

Analysis and Improvement of Optical Flow Methods with a Focus on Robustness

Libo Long

Thesis submitted to the University of Ottawa
University of Ottawa
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

© Libo Long, Ottawa, Canada, 2025

Declaration of Authorship

I hereby certify that this thesis is entirely my own original work except where otherwise indicated. I am aware of the University of Ottawa regulations concerning plagiarism, including those regarding consequent disciplinary actions. Any use of the works of any other author, in any form, is properly acknowledged at their point of use.

Copyright

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of uOttawa's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Acknowledgements

I would like to express my gratitude to my supervisor, Professor Jochen Lang, for his invaluable guidance, support, and insightful advice throughout my PhD. His mentorship has been instrumental in shaping both my research and personal growth.

I am also sincerely thankful to my examination committee members, Professor Robert Laganière, Professor Jiying Zhao, Professor Yuhong Guo, Professor Minglun Gong, for their time, constructive feedback, and thoughtful suggestions that have significantly improved the quality of this thesis.

Special thanks go to my collaborators, Xiao Hu and Tianran Liu, whose expertise and collaboration enriched my research experience and contributed to several fruitful discussions.

I am deeply grateful to my parents, Qizhi Long and Zhijian Shen, for their unwavering love, encouragement, and sacrifices. Their support has been a constant source of strength throughout this journey.

Finally, I would like to thank everyone in the Viva Lab for creating such an inspiring research environment. The camaraderie and support from my colleagues have made this journey enjoyable and rewarding.

Abstract

In the past decade, optical flow estimation has garnered increased attention from researchers as a fundamental component of computer vision applications such as autonomous driving and video editing. However, previous research has mainly focused on predicting more accurate results in benchmarks, revealing a gap between experimental results and real-world scenarios. This thesis explores this field and aims to propose optical flow estimation methods that are robust for real-world applications. Specifically, we identify three main challenges that previous methods still face:

- **Small Motion Boundary Error:** Existing optical flow methods generate blurred motion boundaries for small objects, which usually fuse with the background.
- **Generalization to Unseen Data:** Existing optical flow methods struggle to predict accurate optical flow for unseen data.
- **Robustness to Both Clean and Adverse Conditions:** Existing optical flow methods contain a trade-off in performance between clean and adverse conditions.

To address these issues, we conduct experiments and analyze each problem. We observe that the main reason for small motion boundary error is that traditional downsampling methods can weaken edge information. Thus, we propose a novel downsampling module for the encoder of optical flow, which visibly reduces motion boundary errors. Furthermore, we propose a content-aware regularization method that significantly enhances the generalization ability of unseen data. We introduce RobFlow, an efficient training strategy to make optical flow methods robust in both clean and real-world adverse conditions, such as rain and snow. We then propose a robustness evaluation method to help future studies of optical flow in the real world. To further study video frame interpolation (VFI) as an application of optical flow, we observe that optical flow methods fail to deal with large obstructions in scenes, which are commonly encountered in the real world. We propose a large obstruction robustness framework to address this issue and enhance the robustness of VFI methods for large obstructions. Extensive experiments demonstrate that our methods not only improve the robustness for real-world scenarios but also enhance performance in benchmarks. To further explore how optical flow methods affect the VFI method, we plan to discover the relational representation between two frames. Our main idea is to use a masked auto-encoder to learn a better representation which improves both optical flow and video frame interpolation tasks.

Table of Contents

List of Tables	xi
List of Figures	xiv
1 Introduction	1
1.1 Background	1
1.2 Challenges for Optical Flow Estimation in Real-world Applications	2
1.3 Problem Statement	3
1.3.1 Small Motions Boundary Error	3
1.3.2 Generalization to Unseen Scenes	4
1.3.3 Simultaneous Robustness for Clean and Adverse Condition	5
1.3.4 Robustness Evaluation of Optical Flow Estimation	6
1.3.5 Applications for Optical Flow in Video Frame Interpolation	7
1.4 Thesis Statement	8
1.5 Contributions	8
1.6 Thesis Roadmap	9
1.7 Publications	10
1.7.1 Contribution Statement	10

2	Literature Review	11
2.1	Traditional optical flow estimation	11
2.2	Supervised learning methods	13
2.2.1	U-Net architecture	13
2.2.2	Pyramidal processing, warping, and cost volume	15
2.2.3	Iterative refinement	15
2.3	Unsupervised learning methods	16
2.3.1	Transformer-based methods	16
2.4	Optical flow in adverse conditions	16
2.5	Applications of optical flow estimation	17
2.6	Datasets	18
2.7	Summary	19
3	Detail Preserving Residual Feature Pyramid Modules for Optical Flow	20
3.1	Introduction	20
3.2	Overview	20
3.3	Related work	22
3.4	Methods	23
3.4.1	Feature Map Analysis	23
3.4.2	Repair Mask	26
3.4.3	Module Structure	27
3.4.4	RFPM with IRR-PWC	29
3.4.5	RFPM with RAFT	30
3.5	Experimental Evaluation	30
3.5.1	Training and Implementation	30
3.5.2	Results	31
3.5.3	Ablation Study	33
3.5.4	Transfer Learning Approach	34
3.6	Summary	35

4	Regularization for Unsupervised Learning of Optical Flow	36
4.1	Introduction	36
4.2	Overview	36
4.3	Related Work	38
4.4	Methods	39
4.4.1	Network structure	39
4.4.2	Content Aware Regularization Module	41
4.4.3	Shared-weight Teacher-student Strategy	42
4.4.4	Regularization and Unsupervised Loss	43
4.4.4.1	Content Aware Regularization	43
4.4.4.2	Level Dropout as Regularization	44
4.4.4.3	Overall Unsupervised Loss	44
4.5	Experiment	45
4.5.1	Implementation Details and Use of Datasets	45
4.5.2	Regularization Analysis	47
4.5.3	Comparison to the State-of-the-Art	48
4.5.4	Ablation study	50
4.5.5	Cross Dataset Generalization	51
4.5.6	CAR in Unsupervised Scene Flow Estimation	51
4.6	Summary	51
5	Robust Unsupervised Optical Flow Estimation Under Challenging Conditions	53
5.1	Introduction	53
5.2	Overview	54
5.3	Related Work	56
5.4	Method	57
5.4.1	Overview	57
5.4.2	Preliminaries	57

5.4.3	Noisy Motion Reduction	58
5.4.4	Local-Structure Recovering	60
5.5	Experiments	63
5.5.1	Experiments Setup	63
5.5.2	Experiments on Synthetic Images	64
5.5.3	Experiments on Real Images	64
5.5.4	Ablation study	66
5.5.5	Zero-shot Experiment	67
5.5.6	Discussion	68
5.6	Summary	69
6	Shape and Texture: What Influences Reliable Optical Flow Estimation?	70
6.1	Introduction	70
6.2	Overview	70
6.3	Related Works	72
6.3.1	Optical Flow Estimation	72
6.3.2	Adversarial Attacks in Optical Flow Estimation	73
6.4	Methods	73
6.4.1	Data Manipulation	73
6.5	Optical Flow Update due to Shape Changes	76
6.5.1	Upscaling	76
6.5.2	General Affine Transformation	78
6.6	Occlusion Handling	79
6.7	Mask segmentation	80
6.8	Generate Textures with ControlNet	80
6.8.1	Robustness Evaluation	82
6.9	Experiment	84
6.9.1	Texture vs. Shape	85

6.9.2	Does the Network Remember Objects?	86
6.9.3	Failures for Unseen Objects	87
6.10	A Study of Unseen Objects	88
6.10.1	Feature Visualization in the Encoder	88
6.10.2	Feature Visualization in the Decoder	90
6.11	Summary	91
7	Learning to Handle Large Obstructions in Video Frame Interpolation	92
7.1	Introduction	92
7.2	Overview	92
7.3	Related Work	95
7.4	Method	96
7.4.1	Overview	96
7.4.2	Dynamic Mask Mixing	96
7.4.3	Region Similarity Map	99
7.4.4	Feature Repair	100
7.5	Experiments	102
7.5.1	Datasets and Implementation Details	102
7.5.2	Ablation Study	104
7.5.3	Comparison with the State-of-the-Art	106
7.5.4	Qualitative Evaluation	106
7.6	Limitations	107
7.7	Summary	107
8	Conclusion	108
8.1	Thesis Contributions	110
8.2	Future Work	111
	References	113

List of Tables

3.1	Results Comparison on Sintel and KITTI-15. The value in parentheses are the errors on the training dataset, the best training result is underlined, and the best testing result is shown in bold. Sintel performance is evaluated by average end-point error (AEPE) over all valid pixels. F1-all is the percentage of optical flow outliers over all valid pixels. * Please note that AutoFlow by Sun et al. [180] uses a different training dataset but does not change the basic RAFT architecture. The warm-start technique can only be used with long sequences of images (e.g., Sintel).	31
3.2	Number of Pyramids and Use of Repair Masks	32
3.3	Repair mask levels and numbers.	32
3.4	Downsampling layer trade-off.	33
3.5	Transfer Learning Comparison for RAFT [185] variants on KITTI-2015. C+T+S+K+H is the training schedule used in RAFT and is also the full schedule for our RFPM-RAFT. S_t is our small transfer learning schedule, which leads to our method trained as RFPMt-RAFT. The result shows that RFPMt-RAFT surpasses RAFT with a small training schedule	34
3.6	Asymmetric Data Augmentation (ADA) during transfer learning. ⁺ is the average value evaluated by 3-fold cross validation.	35
4.1	End Point Error (EPE) of different regularization methods. bs indicates the block size for Dropblock.	45

4.2	MPI Sintel and KITTI official optical flow benchmark results with PWC-Net variants. We report EPE error (lower is better) to compare performance for Sintel and KITTI 2012 and the percentage of erroneous pixels F1(%) error for KITTI 2015. We report both, supervised and unsupervised methods. Missing entries (-) are for values that are not reported for the specific method by the authors. [‡] MDFlow uses RAFT [185] as the student model. The results show that the two frames and multi-frame versions of our method outperform all unsupervised and even the original supervised PWC-Net and some other supervised variants.	46
4.3	Evaluation with state-of-the-art methods on MPI Sintel and KITTI benchmark. Computational cost (FLOPs) and inference time are measured for Sintel images with 448×1024 resolution (We tested the inference time on a single Tesla P100 GPU).	47
4.5	Cross Dataset Generalization. All methods are trained on Sintel, and are evaluated on KITTI 2012/2015 training data.	47
4.4	Ablation study of the main components of our method. EPE of indicated pixels are reported. DMCV : Deformable Modulation Cost Volume. LDR : Level Dropout Regularization. ARL : Augmentation Regularization Loss. CAR : Content Aware Regularization.	48
4.6	Evaluation on KITTI Scene Flow training. Our regularization method clearly reduces the percentage of stereo disparity (D1-all and D2-all), optical flow (F1-all) and scene flow (SF1-all) outliers compared to its baseline Self-Mono [57]. Other state-of-the-art multi-task approaches are listed for comparison.	48
5.1	Quantitative results on KITTI datasets: D stand for the denoise module, we use JORDER-E [211] for deraining, AECD-Net [199] for defog and MCR [26] for dark images, [†] : only training on clean dataset, zeroshot inference on challenging conditions. Bold text represents the best result for the metric for each test. Whereas, the <u>underlined</u> text represents the second best for the metric of each test.	62
5.2	Ablation study: L_{dph} indicates Equation (5.2), L_{self} indicates Equation (5.1), LSR indicate the local-similarity recovery without confidence map, C.M indicates confidence map, L_{ni} indicates Equation (5.3). $\checkmark^*$ is a small scale loss (e.g. 0.01).	63

5.3	Quantitative results on the GHOF datasets: RTData denotes the real training data in GHOF. All supervised methods are trained only on synthetic data and do not have access to real data before testing. Semi-supervised and unsupervised methods use real data during training. However, our method uses only clean real data and synthetic weather data during training, which constitutes only 5.22% of the total data compared to other unsupervised methods.	65
5.4	Quantitative results on Zero-shot experiments. Left three column are synthetic data, Snow-R are real-world data.	66
5.5	Discussion on feature loss. (i) is i-order neighbourhood.	68
5.6	Discussion on different image translation strategies.	68
6.1	Quantitative results on KITTI 2015 and Flow-R datasets. We report the average end-point error (AEPE) where not otherwise stated, as well as the F1-all measure, which is the percentage of optical flow outliers with an error larger than 3 pixels. "C + T" refers to results that are pre-trained on the Chairs and Things datasets. "S/K (+ H)" refers to methods that are fine-tuned on the Sintel and KITTI datasets, with some also fine-tuned on the HD1K dataset. Flow-R-T donate the data we replace the texture of main object from KITTI 2015. Flow-R-S donate the data we replace the shape of main object from KITTI 2015. . . .	83
6.2	Texture attack and Shape attack. Quantitative results of the attacked first frame and original second frame.	84
6.3	Adding new objects. Quantitative results of adding new objects to original images. "Add Seen" indicates the data where we add KITTI objects to the images. "Add Unseen" refers to the data where we add real-world objects not included in KITTI. Foreground indicates the EPE errors in the area of the new object. Background indicates the EPE error that do not include the new object.	85
7.1	Ablation study. Effect of applying DMM and RSM on the performance of the proposed model. "-D" indicates using DMM and "-DR" indicates using DMM and RSM. (Note: RSM is applied by Equation (7.3), and COA is evaluated in Table 7.2).	105
7.2	Quantitative comparison of VFI methods on three datasets. The best result is in red, and the second best is in blue. "-DRC" indicates using DMM, RSM and COA with Equation (7.6).	105

List of Figures

1.1	An example of small motion boundary error.	3
1.2	An example of failures of optical flow estimation in large obstruction cases. . . .	7
2.1	Visualization of dense optical flow in KITTI [34] dataset.	12
3.1	Failure cases in current pyramid architectures. (a) Low resolution images blur object boundaries. (b) The feature map exhibits the same problem due to the down-sampling in the convolution layer.	21
3.2	Downsampling modules. WFD is the original module used in CNN optical flow estimation [52, 56, 181]. RFD combines WFD and Max Pooling (MP) with a residual-like design.	24
3.3	Illustration of flow error. The top shows two pixels of a high resolution image, red indicates the flow of the object is to the left, and blue indicates the flow of the object is to the right. The top high resolution image predicts the flow of the two objects correctly (circle towards left, triangle towards right). The low resolution image at the bottom demonstrates that fusing the two pixel produces an average flow prediction which is different from either correct flow value.	25
3.4	Architecture of RFPM. Feature maps are extracted with three different pyramids.	26
3.5	The network extracts a pair of feature maps by a shared weights RFPM and feeds them into a correlation layer.	27
3.6	Visualizing the relation between feature map and predicted flow. Failure areas in RAFT are indicated with white squares. These areas have blurred edges in the feature pyramid. In our RFPM, the flow result correctly predicts the motion boundary because of the improved edges in the feature pyramid. RFPM significantly reduces the EPE errors.	28

3.7	Visualization of feature pyramids in RFPM-RAFT. Comparison of the learned features from left pyramid using WFD, middle pyramid using RFD with repair mask (RM), and right pyramid using WFD with repair mask (RM).	29
4.1	Level l of our multi-frame network.	40
4.2	Flow decoder during prediction (top) and with CAR during training (bottom). . .	41
4.3	Visualization of CAR. The second column shows regions that are emphasized by the projection in CAR (large positive values of M , see Equation (4.9)). The third column shows regions that are diminished (large negative values of M), respectively. The fourth column show the estimated flow by our method. The result is from a generalization study on Vimeo-90K [207].	42
4.4	CAR block. The feature map is split into k subset. We randomly choose one subset (red in the graph) to enhance by a residual convolutional module.	43
4.5	Qualitative visual comparison of our method with UFlow [68] in KITTI 2015. (We did not find results of UPFlow on the KITTI website.) Red pixels indicate higher errors while blue pixels indicate less errors. Our result is visually improved over the previous state-of-the-art. The green rectangles highlight some obvious areas.	49
5.1	ARFlow performs well on clean scenes but suffers from degradation in adverse weather conditions. Our method exhibits robustness to various environmental changes.	54
5.2	Overall architecture of RobFlow. The corresponding challenging pairs are generated before training. In each step, we feed the original and a random challenge pair to the network, and align the feature and motion by our proposed methods. .	58
5.3	The relation between features and optical flow. On the left, we show the features difference map between clean and foggy images. Our RobFlow method generates more similar features for clean and foggy images, and the optical flow in foggy conditions is closer to the optical flow in clean conditions. We report the Cumulative Distribution Function (CDF) of the distance between clean and foggy conditions in the KITTI dataset [34]. The top right figure shows the pixel-wise distance between clean and fog features. The bottom right figure shows the pixel-wise accuracy of fog optical flow and ground truth.	61
5.4	Qualitative results of optical flows on GHOF dataset.	67

6.1	Samples of Flow-R. Original and modified images (Row 1 and 3), ground truth optical flows (Row 2 and 4) are shown. The first column displays the original data from the KITTI dataset. The second column shows the data with the texture of the main object of car replaced. The third column shows the data with the shape of the main object rescaled. The fourth row presents the data with an unseen object added.	72
6.2	Pipeline of our methods. Given the data I_1, I_2, f_{gt} from KITTI, we manipulate the data by changing the texture or shape of object or adding new objects.	74
6.3	Texture Attack. The texture of the main object (car) in the first frame is replaced, while the second frame remains unchanged. The resulting optical flow is similar to that derived from the original images.	76
6.4	Screenshot of our APP. (a) Present the GUI of the app. It takes mouse clicks as input, and when you click the "Generate Mask" button, the app will automatically generate a pair of masks as shown in (b) and (c).	81
6.5	The pipeline for texture generation.	82
6.6	A failed example. We add seen and unseen objects to a similar area. The first row indicates that all methods successfully predict the motion of the added seen (car) objects. The second row indicates that all SOTA methods failed to predict the unseen (fox) objects.	83
6.7	Failed example with different positions. Each column indicates the object car being non-occluded, partially occluded, or fully occluded by the unseen object (apple). The predicted flow from FlowFormer++ shows similar results; all flows ignore the unseen objects. The original motion of the car is still predicted even when the car is fully covered in the third column.	84
6.8	Feature Dissimilarity. All levels in the encoder have high dissimilarity compared with the original features for the added object region, indicating the encoder retains the new object.	87
6.9	Example of Feature Dissimilarity. A clear difference of encoder features with or without new objects is shown at each level. The encoder retains the information related to new objects.	88
6.10	Visualization of Intermediate Flows in the Decoder. Decoder GRU iterations in original and failed samples.	89
6.11	SSIM of intermediate flow in each iteration between failed and original samples.	89

6.12	Feature similarity of GRU block. A higher value indicates that the features of the image with the added object are similar to those of the original image.	90
7.1	Predictions in real-world obstruction scenes, standard approaches fail due to occlusions and motion error, our method makes the models robust in those conditions.	93
7.2	Schematic View of our Dynamic Mask Mixing (DMM) Data Augmentation. We first create a binary mask and add motion to it. Then, we fuse the data with another image using the binary mask to generate new training data.	97
7.3	Two examples of occlusions in a scene with obstructions. The first and second columns show the warped frames generated by the VFI network. The pixels in the red rectangle fail to warp to the same position. Our Region Similarity Map clearly indicates the error.	98
7.4	A comparison of different similarity methods for scenes with obstructions. The region cosine similarity method is most stable.	99
7.5	The details of our COA: Each patch and its 8-neighborhood in the appearance feature map P are projected into one token. Attention is computed with the key K from the inter-frame feature x . The correspondence will match if the patch of x is similar to the patch of f and its neighbors. The hidden features are calculated by the dot product of attention and values V , and repairs the inter-frame features in the explicit region indicated by the Region Similarity Map $1 - C$	100
7.6	The architecture of our model, which uses any optical flow based network as the VFI model. We take the input feature map from the encoder and use the feature repair module to fix the feature in the decoder. W indicates the warp operation.	101
7.7	Qualitative results on SNU test dataset.	103
7.8	Qualitative results on our RWO dataset	104

Chapter 1

Introduction

1.1 Background

"Motion perception is the process of inferring the movement and direction of objects within a visual environment by utilizing visual, vestibular, and proprioceptive inputs for a human" [38]. This ability contributes to fundamental functions in humans' daily lives, such as recognizing the movement of objects, sensing one's movement, guiding eye and body movements, avoiding colliding with others and perceiving the shape of moving objects. Based on those functions, humans then can perform complex tasks like walking, sport and driving [13, 81, 115].

Because of this importance, motion estimation has been a crucial component of many computer vision and intelligent systems, such as autonomous vehicles and robots. Motion estimation can provide proper motion and geometry information for those tasks. This thesis focuses on pixel-level motion estimation – optical flow estimation.

Optical flow is the pattern of apparent motion of image objects between two consecutive frames caused by the movement of those objects or the camera [45]. It is commonly used as a fundamental component of many high-level vision-based applications. In self-driving systems, the vehicle uses the optical flow information to avoid collisions and plan its own route. In video editing, optical flow can help to track objects moving through the whole video or increase the smoothness of the video, making a slow-motion video.

Compared to the human brain's motion estimation, robot optical flow estimation reveals an interesting phenomenon: Recent optical flow estimation methods can compute dense motion fields faster and with higher precision in structured, controlled environments than human manual annotation. However, the human visual system remains superior in handling complex, real-world

motion perception tasks with robustness and adaptability. In this section, we will discuss recent optical flow methods and the main challenge they face to deal with real world scene. This thesis is mainly focused on reducing the gap between experimental methods and real-world application of optical flow estimation. We define the real-world environment as involving model inputs that include natural complexity, unpredictability, noise, and variability. Robustness is the ability of a system, model, or process to perform reliably and maintain its functionality even when facing uncertainties, disturbances, errors, or unexpected conditions [236].

1.2 Challenges for Optical Flow Estimation in Real-world Applications

Many learning-based computer vision tasks have been developed. AlexNet [82] is the first convolutional-based deep neural network to achieve good progress in classification tasks. Following that, FlowNet [27], the first deep learning-based optical flow method, was developed. The network takes two frames as input and predicts the dense optical flow. The network learns by backpropagating the errors calculated from the difference between the predicted flow and ground-truth labels. However, collecting labels for optical flow is expensive. The main real-world autonomous driving dataset for optical flow is KITTI [34], which contains 200 labeled training pairs and 200 labeled testing pairs. The data were collected with a Velodyne laser scanner and a GPS localization system in a car. Consequently, many researchers have started to look at learning optical flow without labels. In general, optical flow methods can be broadly classified into two categories: supervised and unsupervised methods.

Supervised methods for optical flow often rely on dense ground-truth data, which can be expensive and time-consuming to obtain through human labeling. As a result, these methods often use large synthetic datasets for training and get promising accuracy, but can struggle to generalize to real-world data. In contrast, unsupervised learning methods do not require ground-truth annotations and instead minimize the differences between the reference and target images, using the predicted optical flow to warp one image to match the other. Unsupervised methods have gained increasing attention in recent years.

To develop optical flow methods, most of the studies focus on how to improve the accuracy of the two benchmarks MPI Sintel [17] and KITTI [34]. Arguably many optical flow methods already achieve higher efficiency in benchmarks than humans. However, those same methods can easily fail in different real-world situations, such as adverse weather, large obstructions, and small objects. Some studies use domain adaptation for optical flow in challenging weather, but those methods fail in clean scenes. In contrast, humans can easily deal with motion in rain, fog,

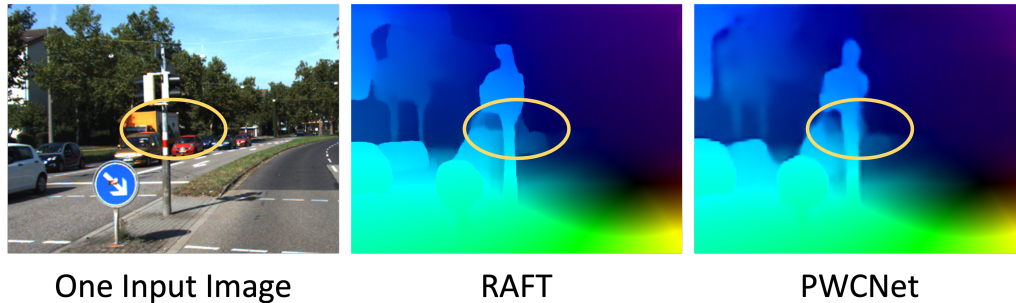


Figure 1.1: An example of small motion boundary error.

and distinguish the motion of tiny objects or partially occluded objects.

1.3 Problem Statement

According to the above discussion, a question arises: can we make optical flow methods robust in real-world settings? In this thesis, we focus on this important and under-explored area and identify the main challenges that hinder the robustness and reliability of optical flow estimation in real-world settings. Specifically, we will discuss three main issues that existing optical flow methods suffer from in the real world and attempt to address them. We propose a robustness evaluation method to examine the behavior of SOTA optical flow methods. Next, we will describe each of the problems.

1.3.1 Small Motions Boundary Error

The human vision system is sensitive to motion boundaries. When driving, we can distinguish the motion of two cars when one car partly occludes the other. However, we noticed that two popular optical flow methods, RAFT [185] and PWC-Net [181], fail to distinguish the motion of small or distant objects. As shown in Figure 1.1, a yellow circle indicates two cars in the input image. However, both RAFT and PWC-Net treat these two objects as a whole in the output optical flow. This could lead to potential risks for high-level applications such as autonomous driving during decision-making. The experiments reported in Chapter 3 demonstrate that most existing networks contain this issue, especially when dealing with background elements.

Previous research has focused less on this problem [56, 181] because the numerical errors are small. For example, the motion boundary of the two cars in Figure 1.1 is small and contains less than 3% of the pixels compared with the whole image. Therefore, even the pixels of motion boundary contains large errors, the overall accuracy will not significantly decline. However, we argue this is an important issue as, in safety-critical applications, distinguishing different moving objects is essential, e.g., distinguishing humans and cars in autonomous driving, distinguishing different organs in surgery.

To address this issue, we first compared the architecture of popular optical flow methods. Basically, we classified them into two categories: coarse-to-fine architecture [56, 60, 181] and iterative refinement architecture [67, 180, 185]. A common ground is that both networks first downsample images into low-resolution features and use low-level features to estimate motions. Imagine downsampling a figure multiple times; closely situated objects could have their motion boundaries weakened, as illustrated in Figure 3.1. This would result in a rough prediction of the optical flow, which serves as a guide and is fine-tuned in subsequent processes. If this error persists in the initial optical flow estimation, it will propagate and potentially mislead future estimations.

With this hypothesis, we propose a novel downsampling module called the Residual Feature Pyramid Module (RFPM). RFPM aims to retain important detail information such as motion boundaries in the downsampling architecture. RFPM incorporates a residual structure between multiple feature pyramids into a downsampling module that corrects the blending of objects across boundaries. The advantage of RFPM is that it can be used as a plug-and-play module for many optical flow methods, reducing the small motion boundary error and improving overall accuracy. The details of the proposed module can be found in Chapter 3.4

1.3.2 Generalization to Unseen Scenes

Compared with methods for other computer vision tasks, we observe that optical flow estimation methods have poor generalization ability. For example, video frame interpolation [77, 112], image segmentation [133], depth estimation [37], and image denoising [219, 224] can undergo one-time training and be tested across multiple benchmarks simultaneously.

The first deep learning-based optical flow method, FlowNet [28], proposes training the network on a synthetic dataset, e.g., FlyingChairs [28]. Then, fine-tuning on benchmarks could yield reasonable performance. Follow-up research has shown that directly training these methods on benchmark datasets does not produce good results [60, 182]. Instead, most methods first pre-train on a large synthetic dataset [53, 60], then fine-tune on each benchmark dataset before

evaluation. This approach may work well in a research setting, but it is not practical for real-world applications.

For example, a video editing application based on optical flow could be trained with real-world movie data, but it would not work well in animation. An autonomous driving system trained with the data from US streets, may not work well in Europe. It is not feasible to constantly collect new data and fine-tune the model to the new environment.

Regularization methods [18, 93, 174] have been used to improve the ability to generalize in many computer vision tasks. My initial thought was to apply regularization techniques like dropout to optical flow methods. However, this proved inefficient in experiments, as detailed in Chapter 4. The reason is that dropout randomly sets some pixels to zero. In optical flow estimation, which takes two images as inputs, randomly zeroing pixels introduces potential occlusion between frames, which can impair the performance of pixel-level flow estimation, especially in unsupervised learning.

In this thesis, We define generalization refers to a model’s ability to perform well on new, unseen data that was not part of the training set. We hypothesize that the main reason for the poor generalization of optical flow methods is their focus on semantic information, such as color. Consequently, when encountering unseen data from a different domain, they produce poor-quality results. However, inspired by human vision, which focuses more on discriminative information such as shape and structure [214], we can still perceive motion even if we do not recognize the objects. Therefore, we propose a student-teacher strategy that employs context-aware regularization to learn more descriptive features. We demonstrate that Context-Aware Regularization (CAR) can improve generalization to unseen environments. Detailed information can be found in Chapter 4.

1.3.3 Simultaneous Robustness for Clean and Adverse Condition

Real-world scenes are often more challenging than synthetic or controlled experiments for optical flow estimation.

One of the most common challenges are adverse weather conditions such as rain and snow and illumination conditions such as darkness. Optical flow methods are sensitive to these situations, and most state-of-the-art (SOTA) methods [56, 97, 114, 185, 205] fail in adverse weather.

To address this issue, previous works have proposed specific network architectures for specific tasks, e.g., considering only darkness [226] or fog [209, 235]. These are far from real-world environments, since these different conditions cannot be dealt with in isolation. Recently, research [234] has focused on adaptive optical flow estimation for multiple adverse weather con-

ditions, but this can degrade performance in clean scenes. Consequently, its applicability in real-world scenarios remains limited, as clear and sunny conditions are more common. Performance in clean and adverse weather faces a trade-off. Furthermore, imagine an autonomous driving system; it needs the ability to handle sunny and rainy conditions and different light changes because of the time-of-day at the same time.

To achieve this, we start by analyzing how optical flow methods work and why adverse weather or light changes can lead to large errors. Given two input frames, we find that original optical flow methods are learned to match similar objects between two input frames by correlation [181, 185], then estimate the distance between them. When dealing with adverse weather such as snow or rain, snowflakes and rain streaks will be treated as objects, and their motion will be included in the final output, thus generating noise in the optical flow. When dealing with low-visibility such as due to darkness, object boundaries will be blurred causing motion boundary errors in the optical flow. Also, those methods designed for adverse weather [234] are forced to learn to ignore small motions, which could degrade performance in clean scenes.

Unlike previous methods, our aim is to make optical flow methods robust in both clean and adverse conditions. The main idea is that the optical flow should be invariant in the same scene but under different challenging conditions. In Chapter 5, we propose a RobFlow training strategy that first constructs corresponding training data with the same scene but under different challenging conditions. Then, we train the network to predict the same optical flow across these varying conditions.

1.3.4 Robustness Evaluation of Optical Flow Estimation

Robustness evaluation is important for safety-critical tasks such as autonomous driving. In this thesis, we focus on real-world applications. However, existing studies mainly focus on improving performance on benchmarks. Some studies try to evaluate the robustness of optical flow methods by proposing more complex scenes. However, optical flow methods may work well in complex scenes but still fail with simple samples. Thus, we need a systematic way to evaluate robustness and try to explain the behaviors of optical flow methods.

We propose a novel method for robustness evaluation by modifying data from original benchmarks. Unlike previous benchmarks that focus on complex scenes, we propose to modify key objects from the original images in order to analyze the sensitivity to these changes observed in the output. Our aim is to identify common failure cases of state-of-the-art (SOTA) methods to evaluate their robustness and understand their behaviors. We show that: Optical flow methods are more sensitive to shape changes than to texture changes; and optical flow methods tend to memorize objects seen during training and may ignore the motion of unseen objects. Our experimental

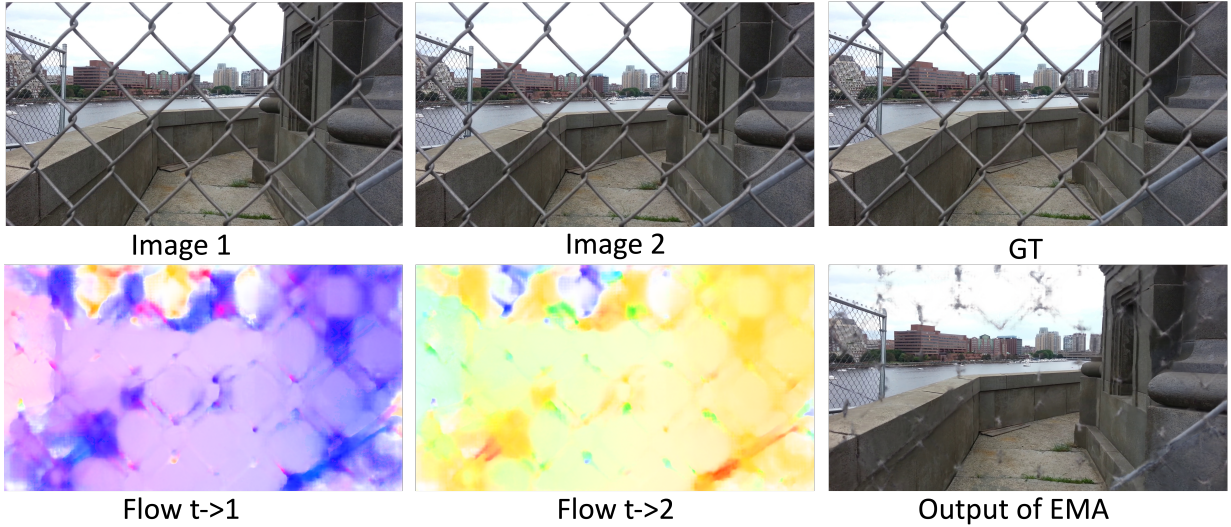


Figure 1.2: An example of failures of optical flow estimation in large obstruction cases.

results and findings provide a more in-depth understanding of the behavior of recent optical flow methods, suggesting a need for more careful design, especially in safety-critical scenarios.

1.3.5 Applications for Optical Flow in Video Frame Interpolation

We further study a high-level application of optical flow in video frame interpolation (VFI). Our main concern is what errors optical flow introduces to VFI tasks. VFI aims to synthesize frames in the middle of two adjacent frames of the original video. It is used to generate slow-motion video or increase video frame rate. The VFI network takes two frames as input. It first calculates the optical flow from the middle to the input images, then warps the two images to the intermediate position, and calculates a residual to fill occlusion or error areas [77, 112, 145].

VFI methods have made significant progress in clean scenes. However, real-world scenarios often present various obstructions and less-than-ideal conditions, which can pose challenges for these methods. We find that optical flow methods usually fail to match and warp frames to the wrong position when dealing with frames containing large obstructions or reflections. This leads to poor quality output frames (see Figure 1.2).

Poor quality output is often caused by large obstructions which generate significant occlusions within the frame, and optical flow methods mismatch those regions. Previous VFI methods

proposed using bi-directional checks or occlusion maps to handle the occlusion issue, but they still suffer from large obstructions. Thus, a more explicit approach needs to be proposed. Instead of only indicating the occlusion map, we use a two-step process: 1. Identify the error regions caused by optical flow. 2. Fix those regions by examining the original input frames. We achieve this in Chapter 7, where we demonstrate that our module significantly improves image quality when dealing with large obstructions and also large motion.

1.4 Thesis Statement

This thesis aims to make optical flow methods robust for real-world scenes. Specifically, we demonstrate that most optical flow methods suffer from three issues: 1. small motion boundary errors, 2. lack of generalization to unseen scenes, and 3. poor robustness under both clean and adverse conditions. We aim to address those issues discussed in Chapter 1.3.1, Chapter 1.3.2 and Chapter 1.3.3. To achieve this, we propose an optical flow framework with unsupervised training strategies. We introduce a RFPM module to address small motion boundary error, content-aware regularization methods to enhance generalization to unseen data, and a RobFlow training strategy to improve the robustness of optical flow methods in both, clean and challenging conditions. We propose a robustness evaluation method to help future studies of optical flow in the real world. We further propose a Large Obstruction Robustness Framework to address the positional errors in video frame interpolation caused by poor optical flow estimation.

1.5 Contributions

The main contributions of this thesis are:

- A Residual Feature Pyramid Module (RFPM) for existing optical flow methods. RFPM retains important details in the feature map and visibly reduces small motion boundary errors in multiple benchmarks [108].
- A Content-Aware Regularization (CAR) approach to enhance the generalization ability of optical flow methods. CAR is randomly applied to certain channels in the convolutional layers during training, which guides predictions in a shared-weight teacher-student strategy. By employing CAR during training, we observe a significant improvement in the generalization ability to unseen data [109].

- RobFlow, an first unsupervised learning framework that works reliably in various outdoor conditions. Our approach is based on two observations regarding the challenges of images taken outdoors. First, noisy motion is mainly caused by the network treating weather image components such as rain streaks as objects, and including their motions in the final output. Second, low quality features due to low-visibility will degrade the estimation of motion boundaries in optical flow.
- Flow-R, a novel method for robustness evaluation by modifying data from original benchmarks. Unlike previous benchmarks that focus on complex scenes, we propose to modify key objects from the original images in order to analyze the sensitivity of the flow estimation to these changes. Our aim is to identify common failure cases of state-of-the-art (SOTA) methods to evaluate their robustness and understand their behaviors.
- Our Obstruction Robustness Framework enhances the robustness of existing VFI networks in the face of large obstructions. The ORF contains two components: (1) A feature repair module that first captures ambiguous pixels in the synthetic frame by a region similarity map, and then repairs them with a cross-overlap attention module. (2) A data augmentation strategy that enables the network to handle dynamic obstructions without additional data [105].

1.6 Thesis Roadmap

In this chapter, we have discussed the importance of optical flow and the main issues it faces for real-world implementation. In Chapter 2, we will conduct a general survey of background material and related works. In Chapter 3, we find that small motion boundary errors are primarily caused by the feature pyramid architecture of existing methods; we address this by proposing a Residual Feature Pyramid Module. In Chapter 4, we propose a regularization technique to improve the generalization ability to unseen data for optical flow methods. In Chapter 5, we introduce RobFlow, an efficient training strategy that enhances the robustness of existing optical flow methods for both clean and adverse weather conditions. In Chapter 6, we propose a robustness evaluation method for optical flow estimation. In Chapter 7, we further analyze video frame interpolation, which is a high-level application of optical flow. Finally, we provide a conclusion and suggestions of future work in Chapter 8.

1.7 Publications

The following publications have resulted from the this thesis.

- L. Long and J. Lang. Detail Preserving Residual Feature Pyramid Modules for Optical Flow. in IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2022.
- L. Long and J. Lang. Regularization for Unsupervised Learning of Optical Flow. Sensors, 2023.
- L. Long and J. Lang. Learn to Reconstruct Large Occlusions in Video Frame Interpolation. in Proceedings of the ACM International Conference on Multimedia (ACMMM), 2024.
- L. Long, T. Liu, R. Laganieri, and J. Lang. Robust Unsupervised Optical Flow Estimation Under Low-Visibility Conditions. in Proceedings of the European Conference on Computer Vision (ECCV) Workshop, 2024.
- L. Long, X. Hu, and J. Lang. Robustness Analysis: Are Optical Flow Methods Safe to Use? The IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR), 2025.

1.7.1 Contribution Statement

Chapter 5 was done in collaboration with Tianran Liu, whose main contribution was drawing the graphs and creating the visualizations. Chapter 6 was done in collaboration with Xiao Hu, whose main contribution was drawing the graphs and handling the training part of the SAMFlow [237].

Chapter 2

Literature Review

This chapter reviews relevant literature related to the focus of this thesis. Each section contains its own related work. The current literature review primarily aims to provide a broad overview of the overall thesis.

Optical flow estimation is a fundamental task in computer vision, and Chapter 2.1 gives a brief introduction to traditional optical flow methods. Modern optical flow methods based on deep neural networks are introduced in Chapter 2.2. Unsupervised learning of optical flow is covered in Chapter 2.3. These two sections are mainly related to Chapter 3 and Chapter 4, which deal with minor object problems in supervised methods and regularization problems in the unsupervised method, respectively. In Chapter 2.4, we review optical flow methods that deal with challenging weather, which can affect the visibility and contrast of the scene. This is relevant to Chapter 5, which tries to improve the robustness in both clean and challenging conditions. The goal of robust optical flow is to supply accurate motion for high-level tasks. Thus, we consider relevant tasks that use optical flow in Chapter 2.5. Chapter 7 also discusses dealing with optical flow errors in video frame interpolation.

2.1 Traditional optical flow estimation

The concept of optic flow was proposed by James Gibson in the 1950s [36] and involves the estimation of the apparent motion between two images based on the difference in their 2D projections of the 3D scene. The optical flow field, which represents the apparent motion between two images, can be either sparse or dense. A sparse optical flow field includes displacement estimates for only a subset of pixels, such as image features or corners. A dense optical flow field



Figure 2.1: Visualization of dense optical flow in KITTI [34] dataset.

includes a displacement vector for every pixel in the image. Figure 2.1 visualizes an example of dense optical flow. The third column presents the optical flow from frames 1 to 2. Different colors indicate the direction and distance that are present in colour coding. e.g., red and purple indicate motion in the positive x or y direction. Cooler colors such as blue and yellow indicate negative x or y direction motion. The intensity of the color indicates the magnitude of the motion, with brighter colors indicating larger displacements.

The gradient-based method refers to the differential approach, which computes optical flow by estimating the displacements using derivatives of the image intensities. This technique purely relies on the brightness constancy assumption (BCA), which assumes the brightness of the same object in different frames of a video should be the same. A constraint is needed to solve the BCA equation. This constraint can either be local, based on the assumptions about the surrounding neighborhood or global, with a smoothness constraint applied to the entire image.

The Lucas-Kanade method [116] is a well-known local approach for optical flow estimation. This technique determines the displacement of a single pixel by using only the intensities of that pixel and its neighbors in two images being compared. For a local neighborhood of pixels, the Lucas-Kanade method derives an equation that relates the brightness of each pixel to its flow. The set of equations for all the pixels within the local neighborhood is solved to estimate the optical flow by performing least-squares minimization. To solve the equations, one assumes that the displacement is constant over a larger area. However, this assumption is generally not realistic for videos of real-world scenes, as the displacement of objects can vary significantly within a single frame. Moreover, increasing the size of the local neighborhood may produce a denser field of optical flow estimates, but it may lead to less accurate results due to the assumption of a constant displacement. Another approach [10] address that problem involves using Fourier transforms to detect the orientation of structures in an n-dimensional Euclidean space. It produces both an

orientation estimate and a measure of certainty based on the fit error. Following works improves the robustness by using a gray-value structure tensor [133], or second order derivatives [188].

The variational method proposed by Horn and Schunck [46] is known as a typical global method. It solves the optical flow estimation problem by introducing a global smoothness constraint, which assumes that neighboring pixels come from the same object and therefore experience similar motion. Nagel et al. [132] further proposes discontinuity-preserving constraints for non-linear partial differential equations. The function can be solved using numerical methods such as Successive Over-Relaxation (SOR) [15] or efficient multi-grid methods [16]. Other approaches focus on minimizing the global energy function, particularly through variational methods [6, 15, 46]. To optimize these formulations, techniques such as gradient descent [11, 198] and message passing algorithms [47, 131] have been employed.

In the past decades, deep learning achieved remarkable progress in many computer vision tasks, including optical flow estimation. Deep learning is a technique that trains neural networks on a large dataset. These neural networks are made up of layers of interconnected nodes or kernels and are designed to learn patterns from the dataset by gradient descent. Learning-based optical flow estimation can be classified into supervised learning and unsupervised learning methods. We will discuss the important progress in following sections.

2.2 Supervised learning methods

Supervised learning methods can use the labeled data for training. The methods can be classified into U-Net and spatial pyramid networks. In this section, we first review the details of each architecture and briefly describe the progress of various methods, including the milestone methods for each architecture.

2.2.1 U-Net architecture

The U-Net architecture is proposed by Ronneberger et al. [159] which consists of a series of convolutional and max pooling layers that downsample the input image, followed by a series of up-sampling and convolutional layers that up-sample the image back to its original size. The down-sampling and up-sampling operations are implemented using convolutional layers with stride 2 and transposed convolutional layers, respectively. The key feature of the U-Net architecture is the use of skip connections between the downsampling and upsampling layers. These skip connections allow the model to preserve fine-grained spatial information as it processes the input image at different scales. There are usually four transposed convolution layers in the refinement

part, and for computational reasons, the flow is finally upsampled to full resolution using bilinear upsampling. Skip connections are used to connect layers in the contractive part to the expanding (refinement) part, providing additional information about flow-level features at the upsampling stage.

Fisher et al. introduced the first deep-learning optical flow architecture, FlowNet [27]. FlowNet consists of two different model architectures: FlowNetS (FlowNet Simple) and FlowNetC (FlowNet Correlation). Both networks are based on U-Net architecture with differences in the encoder part. FlowNetS takes two stacked RGB frames with one encoder. FlowNetC, on the other hand, uses two identical branches for the input frames and merges them at a later stage using a correlation layer. This correlation layer performs cross-correlation (cost volume) between the feature maps of the two input frames, enabling the network to compare each patch without needing additional learned parameters. FlowNet shows that cross-correlation is an efficient operation for optical flow methods. They also propose a synthetic Flying Chairs dataset to increase training samples.

After this, Ilg et al. introduced deep network FlowNet 2.0 [60], which combines multiple networks, including FlowNet and FlowNetC, to deal with large displacement. They first calculate a global optical flow, and use a warp operation to align the second image with the first image by using the calculated optical flow. Then, local motion is calculated between first image and second warped image. The stacked iteration of sub-networks greatly improves the accuracy of optical flow estimation. However, it also comes with a significant increase in model size and running time.

To reduce blurry flow fields caused by unrefined features and low-resolution, Zhai et al. [221] propose a network that combines an attention mechanism and dilated convolutional layers. This method enhances the discriminative and representational abilities.

To improve flow estimation accuracy, Xiang et al. [203] use prior knowledge assumptions as regularization terms, such as brightness constancy, gradient constancy, and spatial smoothness. To capture large displacement, Zhai et al. [220] designed a dilated residual network that increases the receptive field for the convolutional kernel.

Cheng et al. [20] proposed a multi-task learning framework that simultaneously estimates pixel-wise foreground object segmentation and optical flow. This method trains a single model to perform multiple tasks instead of training separate models for each task. The framework aims to improve performance by learning shared representations that can be useful for both segmentation and optical flow estimation. By jointly training the two tasks, the model can learn to take advantage of their relationships.

Ilg et al. [61] modify FlowNet 2.0 by eliminating brightness error and estimating occlusion areas jointly with disparities or optical flow. Another approach [59] uses FlowNetC as a building block to estimate the confidence interval on the estimated optical flow.

2.2.2 Pyramidal processing, warping, and cost volume

To improve performance for large displacements, In this method, the optical flow is first calculated from the low-resolution image of the image pyramid and is treated as global motion. Then, for each level of the image, finer optical flow is calculated based on low-resolution flow. This approach makes the network much smaller than a corresponding U-net architecture and it can better handle large motions.

PWC-Net [181] first proposed the feature pyramid, warping, and cost volume architecture. PWC-Net first extracts the feature pyramid from two input images. For each level/resolution of the feature map, they warp the second feature map by coarse-level flow and compute the cost correlation between the first and warp the second feature map, which tries to match a similar region. Then, they calculate the intermediate flow based on previous optical flow and match information.

2.2.3 Iterative refinement

PWC-Net [181] is a millstone in deep-learning-based optical flow estimation with many follow-up works [56, 79, 229]. One important PWC-Net variant is IRR-PWC [56], To addresses the issues of high computational cost and lack of robustness to occlusions in existing methods, Hur and Roth [56] proposed an approach called Iterative Residual Refinement (IRR), is based on the combination of classical energy minimization approaches and residual networks. The IRR approach aims to reduce the number of parameters and increase accuracy by using weight sharing and iterative refinement. Additionally, they suggest to integrate occlusion prediction and bi-directional flow estimation into the IRR scheme to further boost the accuracy.

Zachary Teed and Jia Deng propose the Recurrent All-Pairs Field Transforms (RAFT [185]) network. Unlike PWC-based networks, RAFT presents an iterative refinement approach that utilizes a Gated recurrent unit (GRU) block. RAFT begins by extracting per-pixel features from the input images and then builds multi-scale 4D correlation volumes for all pixels. The flow field is iteratively updated through a recurrent unit that performs lookups on the correlation volumes.

Many recent works are based on RAFT. Some works [67, 114] incorporate different attention-based modules to bring long-range dependencies from features.

2.3 Unsupervised learning methods

Jason et al. [218] propose the first unsupervised deep-learning optical flow method. They propose a combination of photometric loss and smoothness loss to train the network. The smoothness loss hypothesis is that the motion of pixels should be similar to their neighbours in most cases.

2.3.1 Transformer-based methods

Some [176, 205, 230] applied global matching to find an initialized flow to handle large displacement. FlowFormer [50] proposes a full transformer-based architecture.

Due to the lack of ground truth, occlusion handling is the main problem in unsupervised optical flow methods. Occlusion occurs when some pixels in the first image are not visible in the second image due to occluding objects, large displacements, or image boundaries. Many previous studies [123, 192] focus on addressing the occlusion problem. Some approaches [123, 192] try to utilize occlusion information by checking the consistency of backward and forward optical flow.

Selfflow [100] proposes a self-supervised learning framework. Given a pair of non-occluded training data, they first add occlusion by masking some pixels in one frame. Selfflow contains a teacher model and a student model. It first trained the teacher model with non-occluded data. Then, it feeds the occluded frames to the student model and uses the output flow from the teacher model to guide the student model. DDFlow [99] uses natural occlusion pixels to guide a student model that is trained with cropped frames, while a teacher model is trained with uncropped frames. Luo et al. [119] proposed OccInpFlow, a framework that uses an appearance-flow network to inpaint occluded flows based on the image structure and texture. Wang et al. [192] proposed a self-supervised method that uses a generative adversarial network to inpaint occluded flows by learning from non-occluded regions. Other extensions improve unsupervised learning by using multiple frames [63, 175], geometric constraints [193] or co-training strategies [42]. Jonschkowski et al. [68] experimented to analyze each technique and combine them into a uniform architecture.

2.4 Optical flow in adverse conditions

The above methods do not work well in challenging conditions. The main issue are adverse weather, such as rain or fog, and darkness, limiting visibility and preventing correct correspondences between two frames. Another problem is the labels of optical flow are much harder to

collect under challenging weather. So far, only a handful of works have explored optical flow estimation with challenging weather. Zheng et al. proposed a data-driven approach that employs a noise model for obtaining optical flow from low-light images [231]. Li et al. proposed two uses of two modules to handle the rain veiling effect and rain streak effect, respectively, that help predict optical flow in heavy rain [92]. Yan et al. proposed a semi-supervised network that converts foggy images into clean images to deal with dense foggy scenes [209]. Zhou et al. [235] proposed a domain adaption framework that transforms the domain from clean scenes to fog scenes. However, most only discuss making optical flow adapt to challenging weather. In real-world applications, we are dealing with both clean and challenging scenes. Therefore, a method of robustness for both senses is necessary.

2.5 Applications of optical flow estimation

Optical flow is usually treated as extra information in many tasks. In action recognition [24, 40, 148, 166, 177, 184], optical flow captures movement patterns between frames, such as frequency and speed, helping to distinguish subtle motion in human activities (e.g., walking vs. running). Instead of relying solely on static RGB frames, integrating optical flow allows models to better recognize actions that depend heavily on motion cues. In video segmentation [21, 137, 138, 173, 190], optical flow provides motion consistency between frames, improving the consistency of the segmentation between different frames. By tracking pixel-level motion, segmentation models can maintain temporal coherence, reducing flickering artifacts in video object segmentation. In video frame interpolation [23, 143–145, 154, 223], VFI synthesizes intermediate frames between two input frames, often by first estimating bidirectional optical flow to approximate motion trajectories. The flow fields guide the warping of input frames before refinement. In medical image registration [87, 124, 130, 227, 228], optical flow-based registration aligns scans taken at different times. Unlike rigid registration, flow-based methods handle deformable tissue motion, improving alignment accuracy in dynamic MRI or ultrasound sequences. In autonomous driving [30, 168], and robot navigation [2], optical flow helps detect moving objects (e.g., pedestrians, vehicles) by distinguishing their motions from the static environment. This aids in collision avoidance and path planning by providing real-time motion cues. In this thesis, we choose video frame interpolation as a high-level application to examine how limitations of optical flow could affect VFI tasks. Given two or more consecutive frames, Video Frame Interpolation (VFI) aims to synthesize intermediate frames between them. Recent VFI methods first predict the optical flow from both sides to the intermediate frame, warp the input images toward the middle frame, and refine the warped result to correct errors.

2.6 Datasets

This section briefly introduces the main data set used in optical flow research.

- **FlyingChairs** is a synthetic dataset that contains 22,232 images of chairs flying in front of different backgrounds. The dataset covers a wide range of motions, scales, rotations, occlusions and illumination changes. The ground truth optical flow is obtained by rendering the 3D models of the chairs and the backgrounds [28].
- **ChairsSDHom** is an extension of FlyingChairs that adds synthetic depth, occlusion and motion blur effects to the original images. It is used for training the model performs well on real-world data, particularly in scenarios with relatively small displacements. [121].
- **FlyingThings3D** is another synthetic dataset that contains 25,824 images of various objects flying in indoor and outdoor scenes. The dataset has complex 3D motions, large displacements, reflections and shadows. The ground truth optical flow, depth, disparity and segmentation are provided for each image pair [120].
- **MPI-Sintel** is a flow dataset derived from the open source 3D animated short film, Sintel. The dataset contains 1,064 images from 23 scenes with different styles and effects. The dataset challenges optical flow algorithms with very long sequences, large motions, specular reflections, motion blur, defocus blur and atmospheric effects [17, 200].
- **Monkaa** is a similar dataset to Sintel, but derived from another open source 3D animated short film, Monkaa. The dataset contains 1,024 images from 24 scenes with diverse contents and motions. The dataset also provides ground truth depth, disparity and segmentation for each image pair [200].
- **KITTI2012** and **KITTI2015** are two real-world datasets captured from a driving platform in urban environments. The datasets contain 194 and 200 image pairs respectively, with corresponding ground truth optical flow obtained by a high-accuracy laser scanner. The datasets are suitable for evaluating optical flow algorithms for autonomous driving applications [33, 125].
- **HD1k** is a dataset for autonomous driving, featuring over 1000 high-resolution frames designed to evaluate optical flow in diverse lighting and weather conditions [76]. However, the ground truth for optical flow contains uncertainties and errors.
- **Shift** is a novel dataset that contains 1,000 image pairs of natural scenes with large view-point changes. The dataset is generated by shifting a camera horizontally or vertically by

a random distance between two consecutive frames. The dataset challenges optical flow algorithms with large displacements, occlusions and perspective distortions [192].

- **Spring** is a high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo estimation. The dataset contains 1,000 image pairs of indoor and outdoor scenes captured by a stereo camera rig with synchronized flashes. The dataset provides ground truth scene flow obtained by a structured light scanner with sub-millimeter accuracy [122].

FlyingChairs is the simplest dataset for optical flow. After that, ChairsSDHom and FlyingThings3D introduce extra occlusions and more objects. MPI-Sintel is created from a 3D animated film, containing more complex motion and large displacements. Later, datasets like KITTI and HD1K incorporate real-world noise. Recently, Shift provides a 4K-resolution dataset. These works aim to make the scenes more complex for evaluating the performance of optical flow methods. However, there is still a lack of a systematic way to examine the robustness of optical flow methods. For example, optical flow methods could still work well in most scenes of complex scene in Shift. In Chapter 6, we propose a data manipulation approach to create a robustness evaluation dataset.

2.7 Summary

This chapter presents a comprehensive overview of optical flow estimation, covering both traditional energy-minimization techniques and modern deep learning-based approaches. Classical methods are beyond the scope of this thesis, as our primary focus is on deep learning-based approaches. We then review supervised and unsupervised learning methods, observing that previous research has primarily focused on improving accuracy on benchmarks, with limited discussion on robustness. We adopt [181] and RAFT [185] as our baseline architectures, with an emphasis on robustness. In Chapter 3, we analyze motion boundary errors prevalent in the methods discussed in Chapter 2.2. Subsequently, Chapter 4 and Chapter 5 investigate key limitations, such as the poor generalization of unsupervised learning approaches introduced in Chapter 2.3 and Chapter 2.4. Chapter 6 evaluates state-of-the-art CNN-based and transformer-based optical flow methods. Finally, we explore how optical flow methods are adopted in video frame interpolation approaches in Chapter 7.

Chapter 3

Detail Preserving Residual Feature Pyramid Modules for Optical Flow

3.1 Introduction

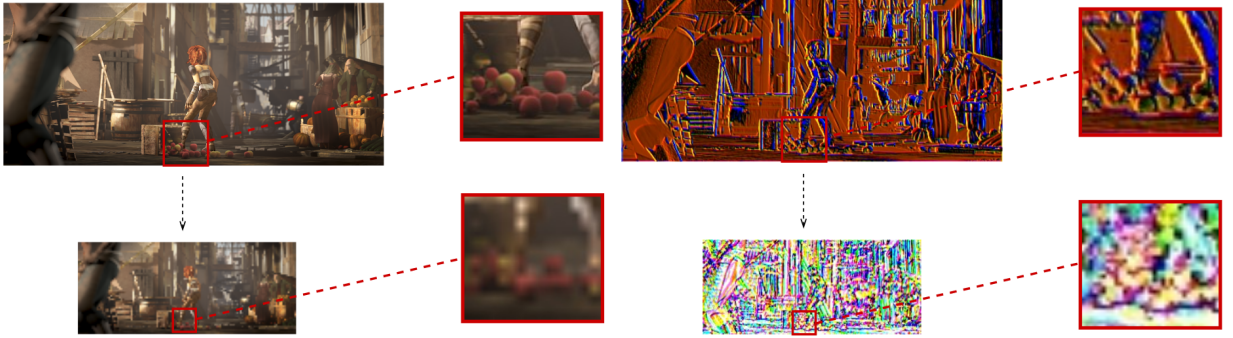
In the previous chapter, we have introduced the progress of optical flow methods. In this chapter, we find that SOTA (state-of-the-art) optical flow methods, e.g., RAFT [185] and IRR-PWC [56], can generate motion boundary errors when dealing with small or large distance objects. We discuss how to address the motion boundary error, which is a significant safety-related issue. We present the main reason that causes motion boundary errors which is the downsampling in the feature pyramid. Thus, we propose a new feature pyramid architecture to address it. Extensive experiments demonstrate that our methods effectively resolve this issue with a minimal increase in parameters.

This chapter has led to a publication [108]¹.

3.2 Overview

Optical flow estimation is a key problem in computer vision and a fundamental building block of many high-level computer vision application [84, 125, 172]. Recent optical flow estimation has greatly benefited from learning-based CNN architectures. Flownet [28] is the first CNN which

¹© 2022 IEEE. Reprinted, with permission, from Libo Long and Jochen Lang, "Detail Preserving Residual Feature Pyramid Modules for Optical Flow," in WACV, 2022.



(a) Boundary blur on a multi-scale image. In the high resolution image (top), we can easily distinguish the outline of individual apples, while in the low resolution image (bottom), it is difficult to recognize individual apples due to their blurred boundaries. (b) Boundary blur on multi-scale feature map. The top image is the feature map at Level 2 of the pyramid where we can see the approximate boundaries of each apple. The bottom image is the feature map at Level 5 where the features in the highlighted area are not discernible.

Figure 3.1: Failure cases in current pyramid architectures. (a) Low resolution images blur object boundaries. (b) The feature map exhibits the same problem due to the down-sampling in the convolution layer.

directly predicts the optical flow with an encoder-decoder architecture. PWC-Net [181] and LiteFlowNet [53] proposed an iterative refinement design. The four fundamental stages in iterative refinement are: first feature maps at different levels of resolution are extracted; second, correlation is used to calculate a cost volume; third, intermediate optical flow is predicted based on the cost volume and on the previous optical flow; and finally, the previous three steps are repeated in an iterative refinement loop. This architecture has been shown to effectively reduce error in large displacement and it has been used as a design in many recent approaches [49, 57, 100, 189, 210]. However, we observe that the iterative refinement design is not without major drawbacks which limits optical flow estimation to improve further. Figure 3.1 shows that the low-resolution image, as well as the low-resolution feature map, exhibit blending across boundaries of foreground objects (the apples in Figure 3.1) and the background because of down-sampling. This can mislead the flow estimation to incorrectly consider foreground and background as one object and hence predict the optical flow incorrectly for both, foreground and background. Because of the iterative design, the erroneous estimate will be amplified in the following iterations.

We introduce a new pyramid module, Recurrent Feature Pyramids Module (RFPM), to address the loss of detail in feature pyramids. We demonstrate how RFPM reduces error in optical flow in Sintel and KITTI, even when integrated into top-performing optical flow methods such as RAFT [185] and IRR-PWC [56]. Our experimental results shows that RFPM-RAFT achieves

state-of-the-art performance on MPI Sintel [17] (Clean pass), and KITTI 2015 [127] benchmarks (two-frame).

We decide to focus on the feature pyramid because we do not want to change the iterative refinement architecture as it has been shown to improve optical flow results, especially for large flow, and because the pyramid module is fully compatible with most optical flow estimation methods or high-level applications. In order to reduce the effort in re-training a complete optical flow method, we also propose a new transfer learning strategy which directly adds RFPM to a trained optical flow method and thereby significantly increasing training progress.

Our main contributions are: First, we show that the traditional pyramid architecture is flawed in optical flow estimation. We present a new pyramid architecture, RFPM, which reduces the error at motion boundaries. We then propose a new strategy of transfer learning for RFPM. Next, we will review related optical flow methods, strategies to avoid blur across motion boundaries and the use of feature pyramids.

3.3 Related work

Optical Flow Estimation. In optical flow the use of multi-resolution representations goes back to Lucas and Kanade [116] who used bandpass-filtering to register features across scales. Bouguet [14] reported a popular implementation using image pyramids. We refer the reader to the work of Sun et al. [179] and their analysis of classic optical flow methods derived from Horn and Schunck [45]. FlowNet [27] is the first end-to-end trainable CNN network based on a U-Net [159] architecture. The model was trained on a synthetic dataset. They propose two basic architectures but the correlation layer of FlowNetC has become a key component in modern architectures. Next, Ilg et al. [58] stack several models of FlowNet into a large system. With a regular training schedule, FlowNet2 achieves significant improvements in the Sintel and KITTI benchmarks. SpyNet [150] introduces a light-weight network by using the feature pyramid and warping within the image pyramid. Sun et al. [181] create PWC-Net that uses an architecture that utilizes a feature pyramid, warping, and a cost volume, which forms the cornerstone of many follow-up works [56, 100, 156]. Recently, IRR [56] and RAFT [185] apply an iterative architecture, which use a fixed CNN component as an unit in the iteration. This architecture achieves remarkable performance with fewer parameters compared to most conventional CNN architectures.

Boundary Blur. Blur across optical flow boundaries is known to be a difficult problem. Many methods use segmentation to prevent blur. Earlier works [12, 241] segment the image into regions using shape or color, then estimating motion by matching the regions. In recent CNN based

methods, Sevilla et al. [167] and Hur et al. [55] add a pre-trained semantic segmentation neural network as an additional component to improve optical flow but these methods are not end-to-end trainable. SegFlow [20] uses an optical flow and an image segmentation model jointly, and construct communications between the two branches of the network. Different from previous work, our method does not use an extra semantic network instead we prevent boundary blur with the proposed pyramid architecture.

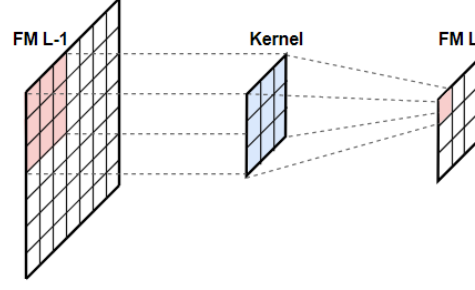
Feature Pyramids. Image pyramids as a multi-resolution representation are widely used in image processing, computer vision and computer graphics [3, 110]. Pyramids of image features are often calculated based on multi-resolution images. This method is slow however, as hand-engineered features on each scale of the images need to be computed. Liu et al. [101] use the convolution layer to predict multi-scale feature maps in their SSD to handle variedly sized objects. However, as noted by Fu et al. [31], SSD fails to detect small instances. Subsequently, Feature Pyramid Networks [94] are a top-down architecture with skip connections in the feature pyramid combining semantically-strong features at low resolution with semantically-weak features at high resolution. This architecture makes the network more robust for different scales of objects. Kong et al. [79] add global attention and local reconfiguration into a SSD-like design. In their work, they compare SSD and its variants showing that global attention enhances multi-scale representations with semantically strong information. The above methods address object detection, semantic segmentation etc. but do not consider optical flow.

Recent optical flow estimation methods [56, 181, 229] use a shared weight pyramid to extract features. The main difference of pyramids in optical flow methods from image detection is that optical flow methods use iterative refinement while in object detection the result is directly predicted. To our knowledge, no prior work has focused on feature pyramids in optical flow in end-to-end trainable optical flow architectures; most of the optical flow methods still use an SSD-like architecture. However, the limitations of SSD for small objects are well known and the problem is exaggerated by the iterative refinement processing in optical flow.

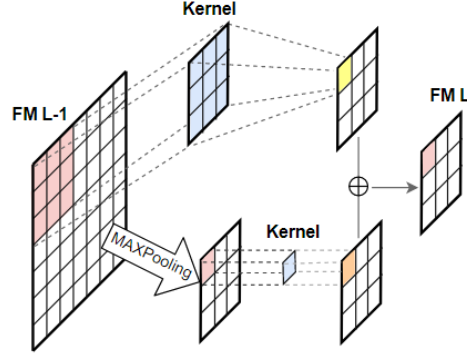
3.4 Methods

3.4.1 Feature Map Analysis

Recent iterative refinement optical flow methods calculate feature pyramids and then use these multi-resolution maps in a correlation step to find a cost volume. We will consider first the currently used approach which we refer to as Weighted Feature Downsampling (WFD) before deriving our alternative approach.



(a) **Weighted Feature Downsampling (WFD)**



(b) **Residual Feature Downsampling (RFD)**

Figure 3.2: Downsampling modules. WFD is the original module used in CNN optical flow estimation [52, 56, 181]. RFD combines WFD and Max Pooling (MP) with a residual-like design.

Given a pair of RGB images, I_t, I_{t+1} , a feature pyramid generates L-level bottom-up pathways, the bottom level is the original input image, i.e., $\mathcal{F}_t^0 = I_t$. Each feature map $\mathcal{F}_t^l$ is generated typically by a 3×3 convolutional filter with a stride of two pixels with respect to $\mathcal{F}_t^{l-1}$. The architecture contains 6 levels of convolutional blocks in recent works [56, 181] while using the last four feature map pairs $\{(\mathcal{F}_t^3, \mathcal{F}_{t+1}^3), (\mathcal{F}_t^4, \mathcal{F}_{t+1}^4), (\mathcal{F}_t^5, \mathcal{F}_{t+1}^5), (\mathcal{F}_t^6, \mathcal{F}_{t+1}^6)\}$ to compute the cost volume. Therefore, at the l-th level, the cost volume can be formulated as

$$V = c(\mathcal{F}_t^l, w(\mathcal{F}_{t+1}^l)), \quad (3.1)$$

where w represents the warping operation that uses bi-linear interpolation to warp I_{t+1} with the optical flow to I_t . The correlation operation c computes the similarity between two feature maps by an element-wise dot product.

At each level l of the feature pyramid, the output pixel value $z_{i,j}^l$ is a weighted sum of pixel values of the previous level

$$z_{i,j}^l = \sum_{x,y} z_{i+x,j+y}^{l-1} k_3(x,y), \quad (3.2)$$

where i,j are the indices of the pixel, k_i is the $i \times i$ convolution kernel, $-1 \leq x, y \leq 1$ (3×3 kernel). We refer to this as Weighted Feature Downsampling (WFD) in Figure 3.2(a).

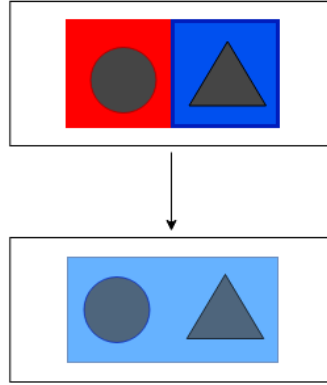


Figure 3.3: Illustration of flow error. The top shows two pixels of a high resolution image, red indicates the flow of the object is to the left, and blue indicates the flow of the object is to the right. The top high resolution image predicts the flow of the two objects correctly (circle towards left, triangle towards right). The low resolution image at the bottom demonstrates that fusing the two pixel produces an average flow prediction which is different from either correct flow value.

We observe a major consequence of this architecture is blurring of motion boundaries. Figure 3.3 illustrates an example: Two tiny objects move in different directions in the original images I_t and I_{t+1} after WFD through several convolution layers. The coarse resolution treats the two objects as a whole, and predicts the same or similar direction for both objects which is incorrect for both and contradicts a possibly correct predictions at the higher resolution. As a result the overall prediction will be in error.

Residual Feature Downsampling (RFD). We propose to apply a joint feature extraction by WFD and Max Pooling (MP) in a residual design [191]. RFD learns to enhance edge areas because MP is expected to keep higher weight pixels for the next level. Although MP itself might

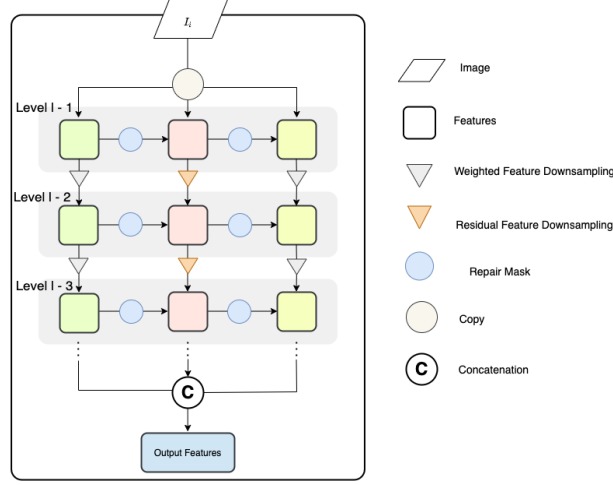


Figure 3.4: Architecture of RFPM. Feature maps are extracted with three different pyramids.

not keep all significant information, it can guide WFD towards the extra information through the residual structure. To this end, we introduce a Residual Feature Downsampling (RFD) pyramid based on the integration of WFD and MP. The RFD extracts features with an addition operation from both WFD and MP (see Figure 3.2(b)). The pixel value at level l is then

$$z_{i,j}^l = \sum_{x,y} MP(z)_{i+x,j+y}^{l-1} k_1(x,y) + z_{i+x,j+y}^{l-1} k_3(x,y) \quad (3.3)$$

Our RFPM includes RFD but it also incorporates multiple different feature pyramids, including pyramids calculated with standard WFD. We also provide additionally pathways for the feature in form of repair masks that act at a chosen level of different pyramids.

3.4.2 Repair Mask

We introduce a learnable Repair Mask (RM) to repair feature maps after downsampling. The RM contains two parts: a multiplicative attention function $\mathcal{A}$ which output a attention mask of shape $(B, 1, H, W)$, and an additive bias function $\mathcal{M}$ which output a bias mask of shape (B, C, H, W) . B is the batch size, C is the channel, and H and W are the height and width of the feature. Assumig two pyramids (and using broadcasting of $\mathcal{A}$), then the feature map of the right pyramid

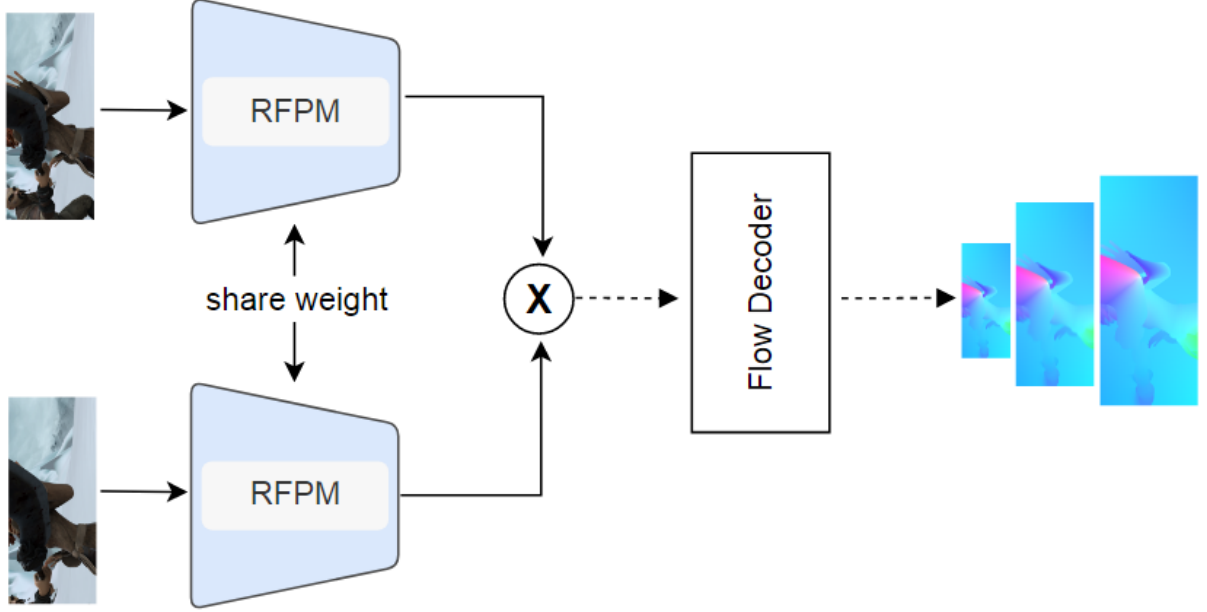


Figure 3.5: The network extracts a pair of feature maps by a shared weights RFPM and feeds them into a correlation layer.

at level $l + 1$ takes the map of the left pyramid into account as in

$$\mathcal{F}_{right}^{l+1} = \text{Conv}(\mathcal{F}_{right}^l \times \mathcal{A}(\mathcal{F}_{left}^{l+1}) + \mathcal{M}(\mathcal{F}_{left}^{l+1})) \quad (3.4)$$

3.4.3 Module Structure

The complete Recurrent Feature Pyramid Module with Repair Masks (RFPM) is constructed as follows (see Figure 3.4):

(a) Multi-kernel top-down pathway: Given a pair of images, features are extracted from the top to bottom via three pathways, the left and right pyramid are constructed with WFD. We call the left pyramid the base pyramid. The middle pyramid uses RFD. When integrating RFPM into different optical flow estimators, the actual pyramid architecture needs to be slightly different. We always keep the original architecture as base pyramid. This design of RFPM is also beneficial for transfer learning which we will describe in Chapter 3.5.4.

(b) Repair left-right pathway: The feature map of the base pyramid extracts a repair mask,

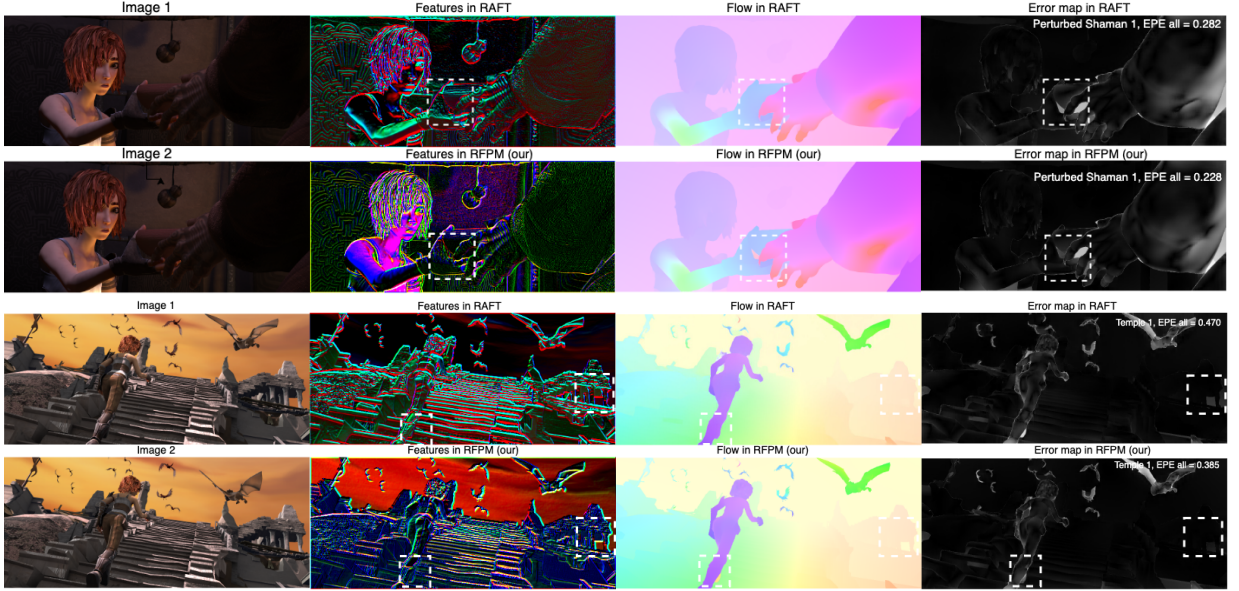


Figure 3.6: Visualizing the relation between feature map and predicted flow. Failure areas in RAFT are indicated with white squares. These areas have blurred edges in the feature pyramid. In our RFPM, the flow result correctly predicts the motion boundary because of the improved edges in the feature pyramid. RFPM significantly reduces the EPE errors.

the mask passes to the middle pyramid to repair missing information due to maximum pooling. The middle pyramid in turn extracts a repair mask to restore the right pyramid as well. Note, we do not add a repair mask at each level. In practice, we found adding repair masks only on some levels at the bottom of the pyramid is sufficient to improve performance.

Therefore, at level l , the cost volume is defined as in Equation (3.1) with the feature maps:

$$\mathcal{F}_t^l = [\mathcal{F}_{left,t}^l, \mathcal{F}_{mid,t}^l, \mathcal{F}_{right,t}^l] \quad (3.5)$$

$$\mathcal{F}_{t+1}^l = [\mathcal{F}_{left,t+1}^l, \mathcal{F}_{mid,t+1}^l, \mathcal{F}_{right,t+1}^l] \quad (3.6)$$

Next, we will discuss how to integrate RFPM with two state-of-the-art optical flow methods: IRR-PWC [56] and RAFT [185].

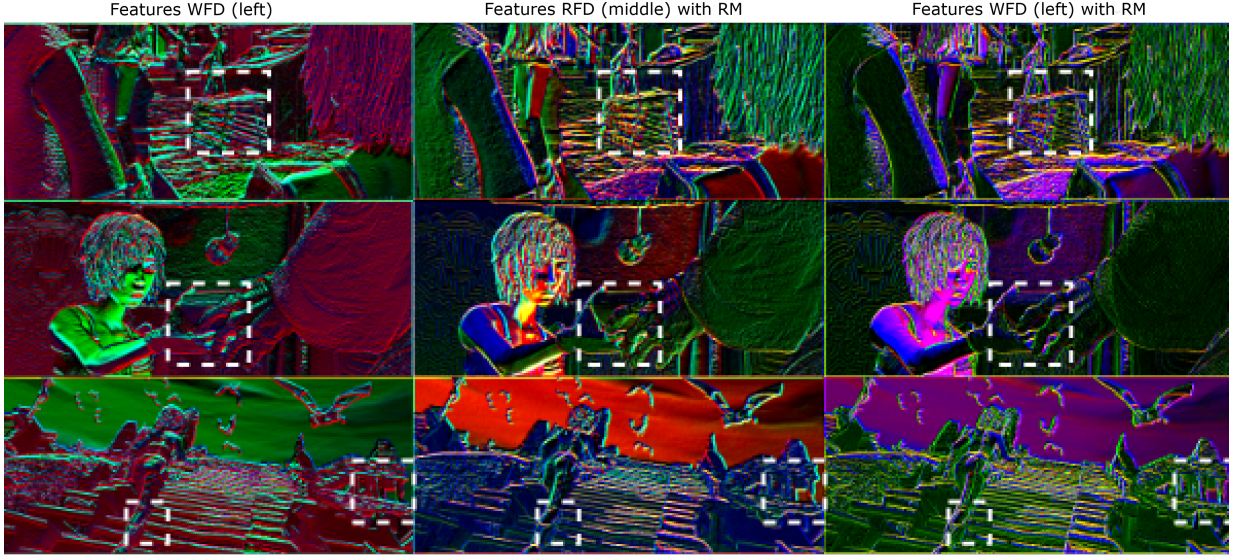


Figure 3.7: Visualization of feature pyramids in RFPM-RAFT. Comparison of the learned features from left pyramid using WFD, middle pyramid using RFD with repair mask (RM), and right pyramid using WFD with repair mask (RM).

3.4.4 RFPM with IRR-PWC

RFPM-IRR-PWC consists of a 6-level shared-weight base pyramid with 16, 32, 64, 96, 128 and 196 feature channels, respectively. Each level from 6 down to 2 makes predictions in a coarse-to-fine manner. The first predictions at level 6 are at $\frac{1}{64}$ and the final predictions at level 2 are at $\frac{1}{4}$ of the resolution of the original image in width and height. The feature channels of the pyramids are 16, 32, 64, 88, 112 and 136, respectively. Figure 3.4 illustrates the pyramids with repair masks generated at level 1 to 3. The base pyramid provides mask $\mathcal{A}(\mathcal{F}_{left}^l)$ and $\mathcal{M}(\mathcal{F}_{left}^l)$, which feed into the same level of the middle pyramid. This avoids unnecessary down- and upsampling. The middle pyramid generates masks $\mathcal{A}(\mathcal{F}_{mid}^l)$ and $\mathcal{M}(\mathcal{F}_{mid}^l)$, which feed into the same level of the right pyramid. Due to the bilateral refinement of flow and occlusion structures, feature maps of all three pyramids feed into the correlation layer. In contrast, only the feature map of the base pyramid is used to predict occlusions and the context flow.

3.4.5 RFPM with RAFT

RAFT contains a three-level feature pyramid, each level consists of two residual blocks, at $\frac{1}{2}$, $\frac{1}{4}$, and $\frac{1}{8}$ resolutions of the original image in width and height, respectively. Instead of a coarse-to-fine manner, RAFT builds a multi-scale 4D correlation volume by all pairs of pixels in the feature map of the last level. RFPM-RAFT uses a similar modification than in RFPM-IRR-PWC, except RFPM-RAFT uses the three-level pyramid architecture of RAFT and the repair masks are generated at level 1 and level 2.

3.5 Experimental Evaluation

We first detail the training for our implementation of RFPM-IRR-PWC and RFPM-RAFT before presenting results on MPI Sintel [17] and KITTI-2015 [127]. We present ablation studies for the configurations of the downsampling approaches as well as the number and level of repair masks in RFPM. We then discuss our reduced effort training approach for our module based on transfer learning and data augmentation.

3.5.1 Training and Implementation

We follow the training configurations of IRR-PWC [56] and RAFT [185] for a fair comparison. We first train RFPM-IRR-PWC on the FlyingChairs_OCC [28] dataset (learning rate schedule S_{short}) and then fine tune on FlyingThings3D [121] (half schedule learning rate of S_{short}). When fine-tuning on Sintel and KITTI, we use a mini-batch size of 4 with the cyclic learning rate proposed by PWC-Net+ [182]. IRR-PWC uses fine-tuning on KITTI based on the checkpoints of FlyingThings3D. We train RFPM-RAFT on FlyingChairs for 200k iterations with a batch size of 6, and on FlyingThing3D for 200k iterations with a batch size of 3. We fine-tune on Sintel by combining data from Sintel [17], KITTI-2015 [127] and HD1K [76] for 200k iterations with a batch size of 3 (the same setting as described in PWC-Net+ [182]). Then, based on the checkpoints of Sintel, we train for 100k iterations with a batch size of 3 (we use half of the batch size and double the iterations in comparison with RAFT because of memory limitations). RFPM is implemented in PyTorch [146] and our experiments use either a single Nvidia 2080Ti GPU with 11 GB of memory, or a Nvidia RTX2070 GPU with 8GB of memory. Some of the ablation study were done with Google Colab. The loss function can be presented as:

$$\mathcal{L} = \sum_{i=1}^N \gamma^{N-i} \cdot \|\mathbf{O}_i - \mathbf{O}_{gt}\|_2 \quad (3.7)$$

Method	Sintel (train)		KITTI-15 (train)		Sintel (test)		KITTI-15 (test)
	Clean	Final	F1-epe	F1-all	Clean	Final	F1-all(%)
SPyNet [150]	(3.17)	(4.32)	-	-	6.64	8.36	35.07
FlowNet2 [58]	(1.45)	(2.19)	(2.36)	(8.88)	4.16	5.74	10.41
FlowNet3 [61]	(1.47)	(2.12)	(1.79)	-	4.35	5.67	8.60
SelFlow [100]	(1.68)	(1.77)	(1.18)	-	3.75	4.26	8.42
PWC-Net [181]	(2.02)	(2.08)	(2.16)	(9.80)	4.39	5.04	9.60
PWC-Net+ [182]	(1.71)	(2.34)	(1.47)	(7.59)	3.45	4.60	7.72
IRR-PWC [56]	(1.92)	(2.51)	(1.63)	(5.32)	3.84	4.58	7.65
LiteFlowNet2 [54]	(1.30)	(1.62)	(1.33)	(4.32)	3.48	4.69	7.62
LiteFlowNet3 [52]	(1.32)	(1.76)	(1.26)	(3.82)	2.99	4.45	7.34
HD3 [216]	(1.70)	(1.17)	(1.31)	(4.10)	4.79	4.67	6.55
VCN [210]	(1.66)	(2.24)	(1.16)	(4.10)	2.81	4.40	6.30
MaskFlowNet [229]	-	-	-	-	2.52	4.17	6.10
RAFT [185]	(0.76)	(1.22)	(0.63)	(1.50)	1.94	3.18	5.10
RAFT(warm-start) [185]	(0.77)	(1.27)	-	-	1.61	2.86	-
RAFT-A* [180]	-	-	-	-	2.01	3.14	4.78
RFPM-IRR-PWC	(1.66)	(2.43)	1.48	5.17	3.63	4.52	7.49
RFPM-RAFT	<u>(0.61)</u>	<u>(1.05)</u>	<u>(0.60)</u>	<u>(1.41)</u>	-	-	4.79
RFPM-RAFT(warm-start)	(0.68)	(1.12)	-	-	1.41	2.90	-

Table 3.1: Results Comparison on Sintel and KITTI-15. The value in parentheses are the errors on the training dataset, the best training result is underlined, and the best testing result is shown in bold. Sintel performance is evaluated by average end-point error (AEPE) over all valid pixels. F1-all is the percentage of optical flow outliers over all valid pixels. * Please note that AutoFlow by Sun et al. [180] uses a different training dataset but does not change the basic RAFT architecture. The warm-start technique can only be used with long sequences of images (e.g., Sintel).

Where $\mathbf{O}_i$ is the predicted optical flow at iteration i , $\mathbf{O}_{gt}$ is the ground truth optical flow, and γ is the weighting factor.

3.5.2 Results

According to Table 3.1, RFPM-RAFT outperforms all published optical flow methods on the MPI Sintel [17] clean pass and KITTI-2015 [127]. Our methods achieve a 12% higher accuracy on

Sintel, and 6% higher accuracy on KITTI compared with RAFT. Figure 3.6 shows a visualization of test results from the Sintel website. To show the benefit of our methods, we compare RAFT with our RFPM-RAFT in both feature map (second column) and output flow (third column). The dashed rectangle corresponds to the same zoomed-in areas as in the feature map and flow results. Comparing features in RAFT and in our proposed RFPM-RAFT, we clearly see that features in RAFT ignore important edges in the zoom-in, and the output flow thus failed to predict motion boundary in the corresponding area (dashed rectangle). In contrast, our RFPM-RAFT significantly reduces the error and better separates different moving objects and environments in these areas. Furthermore, Figure 3.7 visualizes all three feature maps in the same level of RFPM-RAFT, we can see that by adding the repair mask in the RFD (middle) and WFD (right), the edges in the feature map are clearer than in the original WFD (left). Therefore, we think the missing detection of edges in the feature maps is the main reason for a poor prediction of optical flow close to motion boundaries in RAFT. As can be seen, our method enhances the edges in the rectangle and succeeds to estimate the optical flow close to these motion boundaries.

Settings	Trained on Chairs		
	Chairs Testing	Sintel(clean) Training	Sintel(final) Training
L/M	0.79	2.29	4.41
L/M + mask	0.73	2.17	4.30
L/M/R	0.74	2.21	4.33
L/M/R + mask	0.72	2.11	4.28

Table 3.2: Number of Pyramids and Use of Repair Masks

Settings	Chairs	Chairs+Things	
	Chairs Testing	Sintel(clean) Training	Sintel(final) Training
Level 1	0.73	1.32	2.79
Level 2	0.73	1.33	2.82
Level 3	0.75	1.33	2.84
Level 1+2	0.72	1.24	2.74
Level 1+2+3	0.70	1.31	2.83

Table 3.3: Repair mask levels and numbers.

3.5.3 Ablation Study

Our ablation study focuses on RFPM-RAFT, because RAFT has fewer parameters but still obtains lower average end-point error (AEPE) on Sintel and KITTI-15 than RFPM-IRR-PWC. All ablation models are trained on FlyingChairs training and tested on FlyingChairs validation, and the Sintel clean and final training data. The repair mask study uses extra fine-tuning on FlyingThings3D.

We will first look at the number of pyramids. Table 3.2 presents the results of two or three pyramids where L, M and R express Left, Middle and Right pyramid, respectively. We find that the models benefit from the L/M/R architecture with the refinement mask, and the main reduction of the AEPE is from the refinement mask.

In Table 3.3, we look at the level in the pyramids where to use a repair mask. The results indicate that the model with repair masks at both, Level 1 and Level 2 leads to the best performance in terms of AEPE. Using the repair mask at all three levels likely causes overfitting on the Chairs dataset as can be seen from low AEPE error on Chairs testing but the relatively higher AEPE on the other two datasets.

Settings	Trained on Chairs		
	Chairs	Sintel(clean)	Sintel(final)
	Testing	Training	Training
W/M/M + mask	0.81	2.32	4.39
W/M/R + mask	0.76	2.27	4.24
W/R/R + mask	0.68	2.11	4.30
W/R/W + mask	0.70	2.09	4.21

Table 3.4: Downsampling layer trade-off.

Finally, we compare the different downsampling layers discussed in Chapter 3.4.1. Table 3.4 shows the benefits of our module. (Note: W, M and R presents WFD, MP and RFD respectively). W/R/W shows the best performances. We suspect that RFD mainly contributes to the Sintel final pass based on the comparison between W/M/M and W/M/R. W/R/R results in the smallest testing error on Chairs, but it might be overfitting as can be seen from the higher AEPE on the Sintel final pass.

Methods	Schedule	KITTI-15 (test)	
		F1-fg(%)	F1-all(%)
RAFT	C+T+S+K+H	6.87	5.10
RFPm-RAFT	C+T+S+K+H	6.20	4.79
RFPmt-RAFT	S_t	6.69	5.08

Table 3.5: Transfer Learning Comparison for RAFT [185] variants on KITTI-2015. C+T+S+K+H is the training schedule used in RAFT and is also the full schedule for our RFPm-RAFT. S_t is our small transfer learning schedule, which leads to our method trained as RFPmt-RAFT. The result shows that RFPmt-RAFT surpasses RAFT with a small training schedule

3.5.4 Transfer Learning Approach

Transfer learning is to transfer a pre-trained representations to a novel problem which can reduce the training time of a neural network [142]. We present a special transfer learning for our module: Given a pre-trained neural network, our goal is to improve the neural network by incorporating RFPm with only a small training schedule. We add the RFPm into pre-trained RAFT and fine-tune the overall model on KITTI-2015. We can also further improve learning by our novel Asymmetric Data Augmentation (ADA) due to the special structure of our module. ADA keeps the same geometric augmentations for Left/Middle/Right pyramids, but implements different chromatic augmentations for each batch of data. This leads us to the training schedule (S_t) where our module in RAFT is first trained on FlyingThings3D for 50k iteration with a batch size of 3 using Asymmetric Data Augmentation. Then we fine-tune for 100k iterations on KITTI with a batch size of 3. We call the resulting model RFPmt-RAFT to distinguish it from RFPm-RAFT that is trained with the same full C+T+S+K+H schedule as RAFT (see Chapter 3.5.4). Table 3.5 shows that our RFPmt-RAFT surpasses RAFT despite only using a small training schedule with 22.2% iterations used by RAFT.

We also conducted an ablation study of ADA with results shown in Table 3.6. RFPmt-RAFT directly trains on KITTI for 20K iteration with a batch size of three. We compare the effect of different ratios of original and augmented data presented to the RFPmt-RAFT. A probability of 0.2 (20 %) for a sample generated with ADA appears to be most effective. Note that the numbers in Table 3.6 are the average of 3-fold cross validation.

Asymmetric Data Augmentation	KITTI-15(train)	
	F1-epe ⁺	F1-all ⁺ (%)
0	1.21	3.63
0.2	1.19	3.60
0.5	1.20	3.64
0.8	1.22	3.64

Table 3.6: Asymmetric Data Augmentation (ADA) during transfer learning. ⁺ is the average value evaluated by 3-fold cross validation.

3.6 Summary

In the chapter, we describe how the traditional feature pyramid is a major reason for boundary errors in optical flow estimation of small and finely detailed objects. The flow of these objects is lost at low resolution levels of the traditional pyramid. We propose residual feature downsampling that includes max pooling to preserve detail features. We use multiple pyramids in our module that incorporate repair masks at some levels of the pyramids. Our RFPM can be easily integrated into modern iterative refinement optical flow methods as it only modifies the downsampling feature pyramid common to these approaches. We demonstrate that by integrating RFPM in two state-of-the-art methods, their overall error in Sintel (clean) and KITTI-15 improves but more importantly there is clear visual improvement for small and finely detailed objects. Furthermore, We have proposed a special transfer learning strategy for RFPM with novel data augmentation that still achieves state-of-the-art accuracy on KITTI-2015 but with a much smaller training schedule. In the next chapter, we discuss the ability to generalize optical flow estimation methods.

Chapter 4

Regularization for Unsupervised Learning of Optical Flow

4.1 Introduction

In the previous chapter, we demonstrated RFPM's ability to preserve fine details during down-sampling. This approach effectively reduces motion boundary artifacts common in current optical flow methods. In this chapter, we will focus on the generalization ability of optical flow methods. Generalization ability is crucial for optical flow estimation, as methods may encounter unseen scenes in real-world applications. However, the generalization ability of existing methods is poor, and they require fine-tuning in each scene before evaluation. In this chapter, we propose a regularization method that not only improves the performance of unsupervised optical flow methods but also significantly enhances their generalization ability to unseen data.

This chapter has led to a publication [[109](#)] ¹.

4.2 Overview

Estimating optical flow, which is the apparent motion of objects in a scene, is a basic task in computer vision that has many applications [[9](#), [19](#), [66](#), [125](#), [172](#), [206](#), [212](#)]. In recent years, convolutional neural networks (CNN) [[53](#), [151](#), [182](#)] have been used to learn optical flow from data, and

¹© 2023 by the authors. Reprinted under the terms of the Creative Commons Attribution (CC BY) license, from: Libo Long and Jochen Lang, "Regularization for Unsupervised Learning of Optical Flow," *Sensors*, vol. 23, no. 8, 2023. DOI: [10.3390/s23084080](https://doi.org/10.3390/s23084080).

they have shown better performance and efficiency than traditional methods based on variational models [15, 157, 178, 201]. However, learning optical flow from data poses some challenges depending on the type of data and the learning paradigm. In this thesis, we focus on the unsupervised learning of optical flow from real images without ground-truth annotations. We review the existing methods and challenges in this area, and propose a novel approach to improve the accuracy and robustness of optical flow estimation.

To cope with the difficulties of unsupervised learning of optical flow, previous methods have proposed various techniques to reduce the errors caused by occlusion, illumination changes, textureless regions, and motion boundaries [90, 96, 118, 152, 155, 196, 215]. However, these techniques are often specific to certain types of errors and may introduce noise or complexity to the network. In this thesis, we propose a different approach that does not rely on these error-specific techniques, but instead uses a general regularization strategy in a novel teacher-student framework.

The teacher-student framework is a common strategy for unsupervised learning, where a teacher network provides guidance to a student network based on some learned representation. However, a limitation of this strategy is that the teacher and student networks do not improve together during the training process. If the student network learns a better representation than the teacher network, the guidance from the teacher network becomes less effective. Some methods have tried to address this issue by using a co-teaching strategy [42, 194], where two networks teach each other iteratively, but this comes at the cost of training two separate networks. In this thesis, we introduce a novel shared-weight teacher-student strategy, where the teacher and student networks share the same network weights but differ in the use of an additional regularization module. The network with regularization acts as the teacher, and the network without regularization acts as the student. In this way, both networks are updated simultaneously by backpropagation and benefit from each other’s improvements.

Another aspect that we investigate in this thesis is the generalization ability of optical flow networks, which is often limited in unsupervised learning methods. We hypothesize that regularization methods can enhance the generalization ability of unsupervised optical flow networks, as they do for image classification and semantic segmentation networks [35, 62]. We test different regularization methods in our teacher-student learning strategy and show that they have some positive effects on the performance. However, we also discover that our novel content-aware regularization (CAR) module improves the generalization ability more than existing methods.

The CAR module is a simple but effective way to enhance and diminish the features of the network randomly during training in the teacher model, and then remove it during testing. The CAR module works as follows: given an input feature x , it splits x into k subsets, randomly selects one subset, and computes residuals only from that subset. The CAR module is content-

aware because it learns a convolutional layer based on the features without regularization. We argue that this idea can prevent the co-adaptation of channels and help the network learn more discriminative features in each channel. As a result, the shared-weight network improves for both the teacher and student models.

The proposed CAR module has several advantages: it has a low memory footprint, it supports real-time computation and end-to-end training, and it can be easily integrated into different tasks and networks. We implement CAR into a modified version of PWC-Net [181], called PWC-Lite [98], which is a small unsupervised optical flow network. We also extend our method to scene flow estimation, which is a more challenging task that involves estimating both optical flow and depth. The experimental results show that our method significantly improves the original PWC-Net model without any additional space and time costs during inference. Our method solely trained on MPI-Sintel outperforms supervised PWC-Net by 27.9% and 32.9% on KITTI 2012 and 2015, respectively. To summarize, our main contributions are as follows:

- We propose a novel and effective teacher-student unsupervised learning strategy for optical flow and scene flow estimation, where the teacher and student networks share the same weights but differ in the use of a content-aware regularization module.
- We experimentally show that a PWC-Net model trained with our unsupervised framework outperforms all other unsupervised PWC-Net variants on standard benchmarks. The multi-frame version surpasses supervised PWC-Net with lower computational costs and using a smaller model.
- A PWC-Net model trained with our method shows superior cross-dataset generalization compared to supervised PWC-Net and unsupervised ARFlow.

4.3 Related Work

Unsupervised learning of optical flow has been an active research area in recent years. Many methods have adopted the teacher-student strategy to improve the performance and generalization of unsupervised networks. SelFlow [100] proposed a self-supervised learning framework that alternates between a teacher model and a student model. The teacher model is trained with a self-consistency loss that measures the consistency between forward and backward flows. The student model is trained with a self-distillation loss that minimizes the difference between the teacher and student predictions. ARFlow [98] introduced an adaptive robust loss function that adapts to different regions of the image based on the teacher’s confidence. The teacher model is

trained with a photometric loss and a smoothness loss, while the student model is trained with the adaptive robust loss and a distillation loss. MDFlow [78] used multiple teacher models with different levels of difficulty to guide the student model. The difficulty level is determined by the image quality and motion magnitude. The student model is trained with a weighted combination of losses from different teachers. Self-Guidance [232] improved the generalization of deep neural networks via knowledge distillation. The teacher model is trained with a self-guidance loss that encourages the network to learn from its own predictions on unlabeled data. The student model is trained with a distillation loss that transfers the knowledge from the teacher to the student. Our method differs from these methods in several aspects. First, we use a shared-weight network for both teacher and student models, which reduces the memory and computational costs and enables simultaneous improvement of both models. Second, we propose a content-aware regularization module that randomly enhances and diminishes partial features during training in the teacher model. This module helps to break the co-adaptation between channels and to learn more discriminative features for optical flow estimation. Third, we demonstrate that our method can be easily extended to scene flow estimation by adding a depth estimation branch and applying our regularization module to both branches.

4.4 Methods

4.4.1 Network structure

Given a pair of RGB images I_t, I_{t+1} , the estimation of optical flow can be formulated as:

$$f(p) = A(I_t, I_{t+1}, \theta), \quad (4.1)$$

where $f(p)$ is the resulting flow per pixel p , A is the estimation network and θ are the learnable parameters of the network.

Flow estimation networks commonly use an encoder-decoder structure. We use a PWC-Net variant (unsupervised) as backbone [98, 100]. The optical flow is estimated in a coarse-to-fine manner. The network contains a shared-weight Siamese feature pyramid which extracts feature maps at different scales from images. At level l , the feature map of the second image I_{t+1} is warped by the predicted flow $f(p)^{l+1}$ of the (previous) coarser level. Then the cost volume V^l is calculated by correlation which represents the matching costs of pixels between images. The decoder module with shared weights at all levels takes the feature map of I_t , the upsampled flow $f(p)_{\uparrow}^{l+1}$, and the cost volume V^l as inputs and predicts the optical flow $f(p)^l$ of level l . Finally,

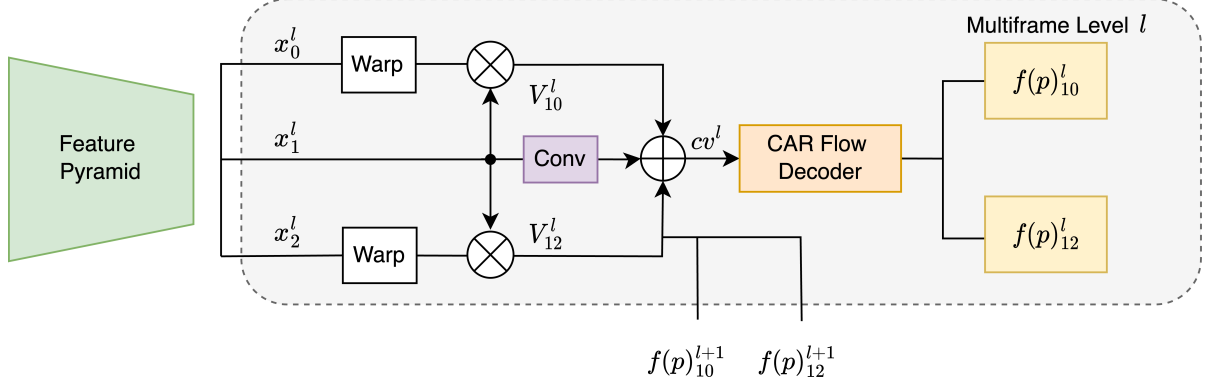


Figure 4.1: Level l of our multi-frame network.

the upsampled optical flow $f(p)_{\uparrow}^l$ is input to the next finer level. Figure 3.4 shows the architecture of our method. We use PWC-Net [181] as backbone with some modifications.

Previous works [41, 113, 204] use deformable convolution to deal with the problem of small objects disappearing in low resolution feature maps. Inspired by that, we use a deformable modulation cost volume (DMCV) to recover distortion and artifacts of small objects by a dense connectivity motion extractor based on deformable convolution (see Figure 3.4):

$$y_t^l = \text{Concat}([x, DC(x)]) \quad (4.2)$$

$$V^l = DC(C(y_t^l, W(y_{t+1}^l))), \quad (4.3)$$

where x and y indicate the feature and dense feature map, DC is deformable convolutional layer, C is the correlation operation. The output flow can be formulated as:

$$cv^l = \text{Concat}([conv(x_t^l), f(p)^{l+1}, V^l]), \quad (4.4)$$

$$f(p)^l = E(cv^l), \quad (4.5)$$

where E presents the optical flow decoder module and $conv$ is a convolution layer.

We optionally extend our method into a three frames network by adding the extra backwards flow from the next frame and its corresponding cost volume as sketched in Figure 4.1. We follow the same multi-frame architecture than ARflow [98], except we added our DMCV and CAR module.

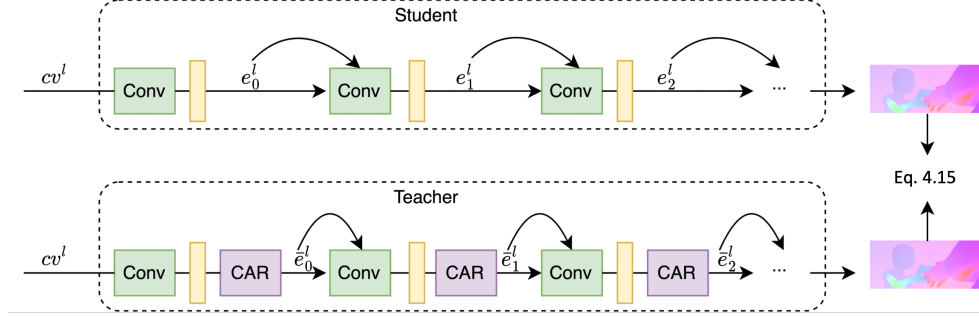


Figure 4.2: Flow decoder during prediction (top) and with CAR during training (bottom).

4.4.2 Content Aware Regularization Module

The main idea of our method is to use only one network but use it twice, once as teacher network with the CAR module and once without the CAR module as the student network in our shared-weight teacher-student strategy. Only the teacher network uses our CAR module which we introduce next. Given an feature map $e \in R^{N \times D}$, where $N = W \times H$. W and H refer to the width and height of the feature map, respectively, and D indicates the depth of the feature map. We first split e into k subsets $v^1, \dots, v^k$ by its depth dimension where $v^i \in R^{N \times D/k}$ is the i -th subset of the feature map. Then, we select a subset j randomly and compute the new feature map via:

$$v^1, \dots, v^k = \text{Split}(e) \quad (4.6)$$

$$j = \text{Random}(1, k), \quad (4.7)$$

$$M = \text{Conv}(v^j) \quad (4.8)$$

$$\tilde{v}^j = v^j (1 + M), \quad (4.9)$$

$$\tilde{e} = \text{concat}([v^1, \dots, \tilde{v}^j, \dots, v^k]) \quad (4.10)$$

Where M is a learnable content-mask which is applied to the j -th subset of the feature map x . Since M is randomly applied to only a subset of e , it stops the channels from co-adapting. $\text{Random}(1, k)$ in Equation (4.7) produces a uniform random number in the range $[1, k]$. We leverage the idea from SpatialDropout [186] to break the co-adaptation of channels, but without simply setting them to zero.

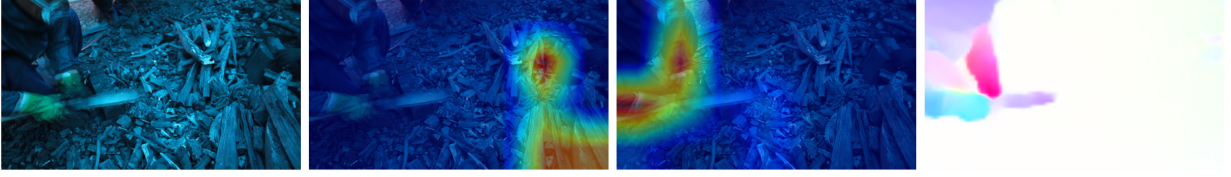


Figure 4.3: **Visualization of CAR.** The second column shows regions that are emphasized by the projection in CAR (large positive values of M , see Equation (4.9)). The third column shows regions that are diminished (large negative values of M), respectively. The fourth column show the estimated flow by our method. The result is from a generalization study on Vimeo-90K [207].

4.4.3 Shared-weight Teacher-student Strategy

During training, we feed each image pair I_t, I_{t+1} through the network twice, once for teaching by invoking the CAR module and once without for the student. The decoder for the student is shown in the upper part of Figure 4.2. Given an input cv^l as defined in Equation (4.4), the optical flow decoder module generates features e_i (i is the index of hidden layer) by densely connected convolutions in each decoder layer i which can be formulated (dropping the layer superscript l for clarity) as

$$e'_i = CR(e_{i-1}) \quad (4.11)$$

$$e_i = \text{Concat}(e'_i, e_{i-1}), \quad (4.12)$$

where CR is a convolutional layer followed by a LeakyReLU activation function. This decoder structure is shown in the upper part of Figure 4.2. The lower part of Figure 4.2 shows the structure of the decoder for teaching.

$$e_i = \text{Concat}(\text{CAR}(e'_i), e_{i-1}) \quad (4.13)$$

Where function $\text{CAR}(\cdot)$ indicates Equation (4.6)-Equation (4.10). The architecture allows us to choose the number of affected channels by selecting k .

The CAR module is randomly applied to a subset of features breaking the co-adaption between channels. In this strategy, the networks for student and teacher are the same except for the CAR module as they share weights, and the common network is trained together instead of sequentially. During the training, if the student learns a better representation guided by the teacher, the teacher can also benefit because of shared-weight parameters.

We conduct an informal study on the generalization abilities of a PWC-Lite model trained with CAR on Vimeo-90K [207]. To show how CAR improves the performance, we use a heatmap

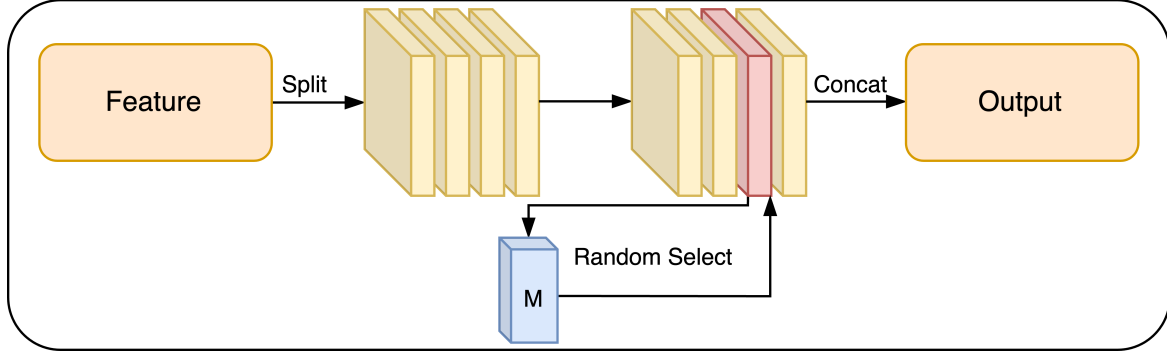


Figure 4.4: **CAR block**. The feature map is split into k subset. We randomly choose one subset (red in the graph) to enhance by a residual convolutional module.

to visualize what regions CAR enhances and diminishes (see Figure 4.3). The first column shows an overlay of the input images. The input images only contain one moving object which is the worker with a static background. The second column in Figure 4.3 shows regions that are emphasized by the projection in CAR and the third columns shows regions that are diminished, respectively. The fourth column shows the estimated flow. We can see that the projection in CAR emphasizes some dark background area. On the other hand, CAR diminishes some areas with large motion preventing co-adaption for large displacement. Co-adaption could otherwise potentially lead to incorrect results for regions with small motion.

4.4.4 Regularization and Unsupervised Loss

4.4.4.1 Content Aware Regularization

During training, the shared-weight teacher-student strategy forward data through the network twice, with and without invoking the CAR module. Then, we obtain two different predicted flow $f^r(p)$ (with CAR module) and $f^o(p)$ (without CAR module). As shown in Figure 4.2, the flow decoders with and without the CAR module share the same layers except for the CAR module which is turned on and off. Therefore, f^r is different from f^o . We regularize the predicted flow by minimizing the difference of two forward passes, i.e.,

$$\mathcal{L}_{ca} = \sum_p \psi(f^r(p) - f^o(p)), \quad (4.14)$$

where $\psi = (|x| + \epsilon)^q$ is a robust function proposed in DDflow [99] with $\epsilon = 0.01$ and $q = 1$.

4.4.4.2 Level Dropout as Regularization

In general, at each level l , our method predicts the flow f^l based on the coarser level flow f^{l+1} but we also propose to use level dropout as regularization. We use the finest flow as pseudo labels and supervise a level dropped flow. A similar idea can be found in Uflow [68] but with the main difference that we propose to also use a corresponding extra regularization loss. Specifically, we predict the flow with the same data twice: First, we randomly drop the calculation of an intermediate flow at pyramid level i , and instead pass the resized flow f^{i+1} to the level $i - 1$. The resulting flow calculated with a dropped level is written f^{drop} in the following. Second, we predict the flow f^o by going through all the levels of the pyramid from Level 6 to 2. Let $S(\cdot)$ be the stop-gradient, then the loss function $\mathcal{L}_{ld}$ can be written as

$$\mathcal{L}_{ld} = \sum_p \psi(S(f^o(p)) - f^{drop}(p)). \quad (4.15)$$

In practice, we also use f^r , i.e., the flow calculated with the CAR module turned on, to guide f^{drop} as an challenging case with a small probability which is

$$\mathcal{L}_{rd} = \sum_p \psi(S(f^r(p)) - f^{drop}(p)). \quad (4.16)$$

The overall regularization loss is a combination of above three loss functions in Equation (4.14)-Equation (4.16). Thus, our shared-weight teacher-student framework will have multiple forward passes in one training step. The overall loss is

$$\mathcal{L}_{cr} = \alpha(\mathcal{L}_{ca}, \mu_1, \delta_1) + \alpha(\mathcal{L}_{ld}, \mu_2, \delta_2) + \alpha(\mathcal{L}_{rd}, \mu_3, \delta_3), \quad (4.17)$$

where $\alpha(\cdot)$ randomly uses the loss in a training step with a fixed probability μ_i . In each training step, we generate a uniform random number δ_i in $[0,1]$.

$$\alpha(\mathcal{L}_i, \mu_i, \delta_i) = \begin{cases} \mathcal{L}_i, & \text{if } \delta_i \leq \mu_i \\ 0, & \text{Otherwise} \end{cases} \quad (4.18)$$

4.4.4.3 Overall Unsupervised Loss

The overall unsupervised loss is then formulated as

$$\mathcal{L}_{all} = \mathcal{L}_{ph} + w_{sm} \times \mathcal{L}_{sm} + w_{au} \times \mathcal{L}_{au} + w_{cr} \times \mathcal{L}_{cr} \quad (4.19)$$

where $\mathcal{L}_{ph}$ is the photometric loss, $\mathcal{L}_{sm}$ is the smoothness regularization loss [218] and $\mathcal{L}_{au}$ is the augmentation regularization loss [98]². In practice, we set $w_{sm} = 1$, $w_{au} = 0.01$ and $w_{cr} = 0.01$.

4.5 Experiment

Experiment	Sintel Clean	Sintel Final
None	2.79	3.72
Dropout	2.83	3.77
DropBlock, bs=1	2.81	3.73
DropBlock, bs=3	2.77	3.75
DropBlock, bs=7	2.76	3.72
SpatialDropout	2.74	3.69
CAR	2.70	3.63

Table 4.1: End Point Error (EPE) of different regularization methods. *bs* indicates the block size for Dropblock.

4.5.1 Implementation Details and Use of Datasets

We conduct experiments on two commonly used optical flow benchmarks: MPI-Sintel [17] and KITTI (including KITTI 2012 [34] and KITTI 2015 [125]). We follow the same augmentation setting as previous methods [98–100].

Our results on the MPI-Sintel benchmark protocol are obtained by pretraining on KITTI and then fine-tuning on Sintel. We do not pretrain on Sintel raw. Sintel raw contains the raw scenes for the benchmark and hence pretraining on Sintel raw may cause data leakage into the test set. For the KITTI benchmark, we pretrain our network with the KITTI raw dataset (we discard all scenes that contain images that appear in the optical flow KITTI benchmarks) and finetune the network on the KITTI multi-view training dataset. This is the same dataset configuration for KITTI as used by previous works [98–100]. We also implement our method in the related dense pixel matching task of unsupervised scene flow estimation.

We implement our method with PyTorch [146]. For all training, we use the Adam optimizer [74] with $\beta_1 = 0.9$, $\beta_2 = 0.99$. We first train our method for 1000k iterations with learning

²We use the same augmentation setting than ARFlow.

rate 0.001 and batch size 4, followed by 400k iteration with learning rate 0.005 and batch size 1. We increase the crop resolution in the second stage for KITTI from (256, 832) in the first 1000k to (320, 1216) in the following 400k. The total numbers of our model parameters are 2.78M (for two-frame) and 2.97M (for multi-frame) during training. Since the CAR module (0.12M) is dropped after training, the final model in predicting optical flow for a pair of images with a resolution of (448, 1024) is even smaller than during training.

Method	Sintel Clean		Sintel Final		KITTI-12		KITTI-15	
	train	test	train	test	train	test	train	test
PWC-Net [181]	(1.70)	3.86	(2.21)	5.13	(1.45)	1.7	(2.16)	9.60%
IRR-PWC [56]	(1.92)	3.84	(2.51)	4.58	–	–	(1.63)	7.65%
ScopeFlow [8]	-	3.59	-	4.10	–	-	-	6.82%
VCN [210]	(1.66)	2.81	(2.24)	4.40	–	-	(4.1)	6.30%
DICL [195]	(1.11)	2.12	(1.60)	3.44	–	-	(3.6)	6.31%
MaskFlowNet [229]	(2.25)	2.52	(3.61)	4.17	(2.94)	1.1	-	6.11%
UnFlow-CSS [123]	–	–	(7.91)	10.22	3.29	–	8.10	23.30%
OccAwareFlow [†] [90]	(4.03)	7.95	(5.95)	9.15	3.55	4.2	8.88	31.2%
UnFlow [123]	–	9.38	(7.91)	10.22	3.29	–	8.10	23.3%
DDFlow [99]	(2.92)	6.18	(3.98)	7.40	2.35	3.0	5.72	14.29%
EpiFlow [233]	(3.54)	7.00	(4.99)	8.51	(2.51)	3.4	(5.55)	16.95%
SelfFlow [†] [100]	(2.88)	6.56	(3.87)	6.57	1.69	2.2	4.84	14.19%
ARFlow [98]	(2.79)	4.78	(3.87)	5.89	1.44	1.8	2.85	11.80%
ARFlow-MV [†] [98]	(2.73)	4.49	(3.69)	5.67	1.26	1.5	3.46	11.79%
UFlow [68]	(2.50)	5.21	(3.39)	6.50	1.68	1.9	2.71	11.13%
MDFlow [‡] [78]	(2.17)	4.16	(3.14)	5.46	-	-	2.45	8.91%
UPFlow [118]	(2.33)	4.68	(2.67)	5.32	1.27	1.4	2.45	9.38%
CAR-Flow(our)	(2.36)	3.69	(3.28)	5.21	1.16	1.3	2.34	9.09%
CAR-Flow-MV(our) [†]	(2.25)	3.46	(3.23)	4.95	1.02	1.2	2.11	8.40%

[†] indicates that the method uses multiple (3) frames.

Table 4.2: MPI Sintel and KITTI official optical flow benchmark results with PWC-Net variants. We report EPE error (lower is better) to compare performance for Sintel and KITTI 2012 and the percentage of erroneous pixels F1(%) error for KITTI 2015. We report both, supervised and unsupervised methods. Missing entries (-) are for values that are not reported for the specific method by the authors. [‡] MDFlow uses RAFT [185] as the student model. The results show that the two frames and multi-frame versions of our method outperform all unsupervised and even the original supervised PWC-Net and some other supervised variants.

4.5.2 Regularization Analysis

We have conducted experiments on the popular regularization methods: Dropout [186], Spatial-Dropout [186], Dropblock [35], as well as our CAR module. We train the network multiple times but change only the regularization method. We set the dropout rate=0.5, and add regularization in the flow decoder of the network. The results are summarized in Table 4.1.

We observe that Dropout and Dropblock with a small block size hurt the performance of the network. We think the main reason is that the input of the network is a pair of images and random dropout of pixels misleads the network to mismatch dropout pixels between the images (i.e., pixels with its features set to 0). SpatialDropout improves the error. Different from these methods, our method learns the adaptive content mask for both images and improves the error further.

Method	# FLOPs	# params	inference time	Sintel		KITTI	
				clean	final	2012	2015
UPFlow [118]	198.27G	3.49M	271ms	4.68	5.32	1.4	9.38%
SMURF [†] [175]	810.14G	5.26M	413ms	3.15	4.18	-	6.38%
CAR-Flow	50.49G	2.74M	34ms	3.69	5.21	1.3	9.09%
CAR-Flow-MV [†]	108.08G	2.97M	59ms	3.46	4.95	1.2	8.40%

[†] indicates that the method uses multiple (3) frames.

Table 4.3: Evaluation with state-of-the-art methods on MPI Sintel and KITTI benchmark. Computational cost (FLOPs) and inference time are measured for Sintel images with 448×1024 resolution (We tested the inference time on a single Tesla P100 GPU).

Method	Sintel clean	Sintel final	KITTI 2012	KITTI 2015
PWC-Net	(1.86)	(2.31)	3.68	10.52%
ARFlow	(2.79)	(3.73)	3.06	9.04%
CAR-Flow	(2.81)	(3.73)	2.65	7.06%
CAR-Flow-MV [†]	(2.22)	(3.26)	2.23	5.97%

[†] indicates that the method uses multiple (3) frames.

Table 4.5: Cross Dataset Generalization. All methods are trained on Sintel, and are evaluated on KITTI 2012/2015 training data.

DMCV	LDR	ARL	CAR	Sintel Clean			Sintel Final		
				ALL	NOC	OCC	ALL	NOC	OCC
				2.92	1.53	22.14	3.87	2.46	26.24
			✓	2.74	1.32	20.92	3.75	2.31	23.90
✓			✓	2.58	1.17	18.64	3.43	2.14	22.52
		✓	✓	2.53	1.15	18.46	3.51	2.03	22.43
	✓	✓		2.57	1.21	18.72	3.47	1.91	22.16
	✓	✓	✓	2.36	1.04	18.14	3.24	1.77	21.22
✓		✓	✓	2.42	1.16	17.54	3.31	1.84	20.83
✓	✓	✓		2.33	1.11	17.31	3.26	1.81	21.36
✓	✓	✓	✓	2.25	1.01	16.27	3.07	1.62	20.12
✓	✓	✓	✓ [†]	2.13	0.99	16.12	2.83	1.55	19.70

[†] indicates that the method uses multiple (3) frames.

Table 4.4: **Ablation study of the main components of our method.** EPE of indicated pixels are reported. **DMCV**: Deformable Modulation Cost Volume. **LDR**: Level Dropout Regularization. **ARL**: Augmentation Regularization Loss. **CAR**: Content Aware Regularization.

Method	D1-all	D2-all	F1-all	SF1-all
GeoNet [215]	49.54	58.17	37.83	71.32
EPC [213]	26.81	60.97	25.74	(>60.97)
EPC++ [117]	23.84	60.32	19.64	(>60.32)
Self-Mono-SF [57]	31.25	34.86	23.49	47.0
Self-Mono-SF-CAR(our)	29.24	32.49	21.34	43.57

Table 4.6: Evaluation on KITTI Scene Flow training. Our regularization method clearly reduces the percentage of stereo disparity (D1-all and D2-all), optical flow (F1-all) and scene flow (SF1-all) outliers compared to its baseline Self-Mono [57]. Other state-of-the-art multi-task approaches are listed for comparison.

4.5.3 Comparison to the State-of-the-Art

We first compare our method with PWC-Net variants reported on the MPI-Sintel and KITTI benchmarks. Table 4.2 shows our method improves the performance and it has lower errors than

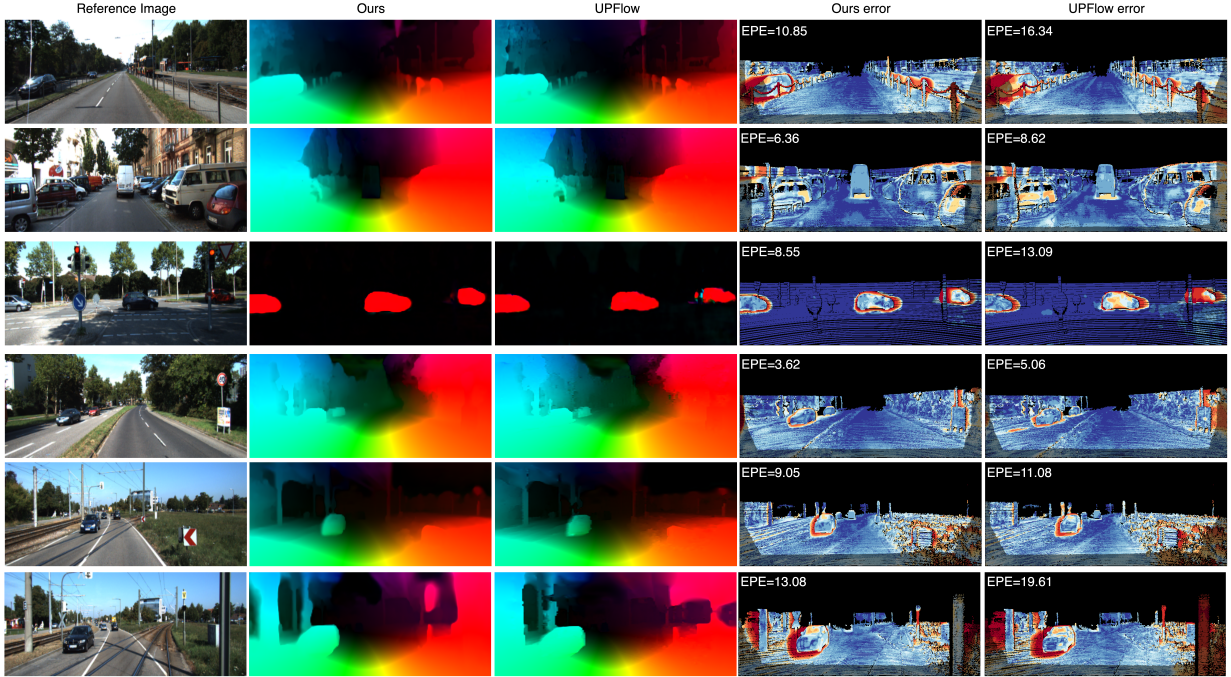


Figure 4.5: Qualitative visual comparison of our method with UFlow [68] in KITTI 2015. (We did not find results of UPFlow on the KITTI website.) Red pixels indicate higher errors while blue pixels indicate less errors. Our result is visually improved over the previous state-of-the-art. The green rectangles highlight some obvious areas.

all other unsupervised PWC-Net methods. On MPI-Sintel, we achieve an AEPE=4.95 on the final pass with 6.9% improvement, and an AEPE=3.46 on the clean pass with 11.3% improvement. We achieve an AEPE=1.2 which is a 14.2% improvement on KITTI 2012, and FI-all=8.40% which corresponds to a 5.7% improvement on KITTI 2015. We also report supervised methods based on PWC-Net for comparison. Our method with 3M parameters surpasses classic supervised methods, e.g., PWC-Net with 8.7M parameters and LiteFlowNet with 5.37M parameters on the benchmarks. Especially, our method is also comparable with some novel supervised methods such as IRR-PWC [56]. Our method even surpasses some of these supervised methods on KITTI 2012 and on the Sintel Clean pass. Figure 4.5 show some qualitative comparisons between our method and previous state-of-art methods. Our method reduces errors in visual comparison to earlier methods.

We analyze the reasons for the performance improvements by our method over the comparators. First, our content-aware regularization module helps to break the co-adaptation between

channels and learns more discriminative features for optical flow and scene flow estimation. Second, our shared-weight teacher-student strategy enables simultaneous improvement of both teacher and student models by sharing weights and transferring knowledge. Third, our method benefits from a simple and efficient design that does not require any extra parameters or computation during inference. Our method has fewer parameters, less computation, and faster inference time than the original PWC-Net. However, as we have chosen to integrate our method into lite PWC-Net, we also inherit possible disadvantages including failures to handle large motion well or errors close to motion boundaries. In future work, we would like to see our method integrated in more powerful optical flow networks, e.g., RAFT [185] but this can be expected to lead to an increase in model size and computation time.

As can be seen from Chapter 4.3, we achieve better performance than UPflow approximately $8\times$ faster with about $4\times$ less computation because we apply our method to a lightweight PWC-Net backbone. Our multi-frame version further improves the accuracy in all benchmarks with less cost than UPFlow. Although the errors of our method are higher than the ones of SMURF, our method leads to a much smaller model and far less computation, and remarkably, is real time on high resolution images with competitive accuracy. As discussed in Section 4.2, SMURF also has the serious drawback of being extremely time and memory expensive in training and hence it is difficult to adapt to new datasets.

4.5.4 Ablation study

To evaluate the capability of each component of our pipeline, we conduct experiments on split training datasets of Sintel and KITTI with a small training schedule (300k iteration with batch size 4). The endpoint error (EPE) of overall pixels (ALL), non-occluded pixels (NOC) and occluded pixels (OCC) are reported for evaluation.

Ablation of main components. Table 4.4 reports the main unsupervised components used in our method which are: Deformable Modulation Cost Volume (DMCV), Level Dropout Regularization (LDR) and Content Aware Regularization (CAR). We also include Augmentation Regularization Loss (ARL) [98] to clarify any possible interactions between the modules in our framework.

We start by only using the photometric loss and the smoothness loss to train the network (see Equation (4.19)). Comparing the first and second row, we can see a significant improvement with our CAR module. Then, we observe DMCV, LDR and ARL all improve the network compared to the first row. Comparing all the combinations, we find that all components improve the performance; the combination of ARL and CAR reduces all errors; the accuracy in the occluded region can be improved by LDR; and the multi-frame version can further improve accuracy. We

also analyzed the regularization rate of CAR and LDR and found a CAR rate=0.5 and LDR=0.9 achieves the highest accuracy, we keep this setting in all other training.

4.5.5 Cross Dataset Generalization

To test the generalization ability of a model, we train it only on Sintel raw and final but evaluate it on the noisy real world KITTI dataset. Table 4.5 reports the results of the fully supervised PWC-Net, the previous SOTA unsupervised method ARFlow and our method. PWC-Net outperforms our method and ARFlow in Sintel in which the model is trained on, but this performance does not generalize well to KITTI. Because of the CAR module, the model trained with our unsupervised method generalizes much better. Note that neither of these models have seen real world images during training.

4.5.6 CAR in Unsupervised Scene Flow Estimation

We extend our method to unsupervised scene flow estimation and implement our CAR module and shared-weight teacher-student strategy in Self-Mono-SF. Self-Mono-SF [57] is a unsupervised Scene Flow method, which uses PWC-Net as backbone. Table 4.6 reports the results of state-of-the-art unsupervised monocular scene flow methods on the KITTI dataset. We follow the evaluation metric of the KITTI Scene Flow benchmark. D1-all, D2-all are the percentage of stereo disparity outliers in the first frame and in the second frame, respectively. F1-all is the percentage of optical flow outliers. SF1-all is the percentage of scene flow outliers. Our method not only improves the performance, but also surpasses other unsupervised multi-task methods. We conclude that our method helps performance in the scene flow estimation task. We suspect that our method is also likely to improve the accuracy in other related dense pixel matching tasks but leave further investigations as future work.

4.6 Summary

In this chapter, we introduced a content-aware regularization module that randomly enhances and diminishes partial features during training in the teacher model. We showed that our method significantly improves the performance and generalization of the original networks and outperforms other popular regularization methods. Furthermore, our method achieves state-of-the-art results on optical flow and scene flow benchmarks and shows superior cross-dataset generalization compared to supervised and unsupervised methods without requiring any extra parameters

or computation during inference. In the next chapter, we discuss the robustness of optical flow methods under different weather conditions.

Chapter 5

Robust Unsupervised Optical Flow Estimation Under Challenging Conditions

5.1 Introduction

In the previous chapter, we discussed and improved the generalization ability of optical flow methods for unseen data. In this chapter, we discuss the robustness of optical flow in clean and adverse conditions. Previous state-of-the-art unsupervised optical flow methods achieve impressive results in clean scenes, they still struggle under adverse weather and in challenging illumination. However, robustness to those conditions is essential in real-world safety-critical settings, such as autonomous driving. We tackle this issue with RobFlow: an effective unsupervised learning framework that works reliably under both standard and challenging conditions. Our approach is based on two observations regarding the challenges of adverse weather images. First, noisy motion is mainly caused by the network treating weather image components such as rain streaks as objects, and including their motions in the final output. Second, low quality features in low-visibility weather or illumination will degrade the estimation of motion boundaries in optical flow. To address these issues, we design a noise reduction strategy to deal with moving disturbances such as rain streaks in the final optical flow estimation, and we design a local-structure recovery to bridge the difference between clean and low-quality features. Additionally, we propose a confidence map to correct possible inconsistencies during unsupervised learning, which will filter out large error pixels in a comparative manner.

This chapter has been partially presented at the ECCV 2024 2nd Workshop on Vision-Centric Autonomous Driving (VCAD) [4].

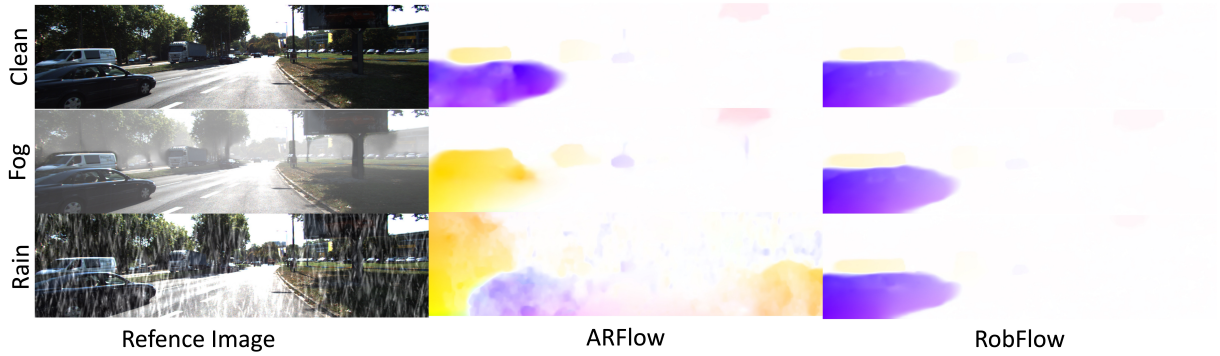


Figure 5.1: ARFlow performs well on clean scenes but suffers from degradation in adverse weather conditions. Our method exhibits robustness to various environmental changes.

5.2 Overview

Optical flow, which aims at estimating pixel-level motion for videos, is fundamental for high-level computer vision applications, such as autonomous driving, video editing or object tracking. While supervised optical flow estimation methods have achieved remarkable progress, they rely on ground-truth labels, which are expensive to collect [34, 127]. Obtaining ground-truth labels requires 3D sensors and manual effort [34, 126]. To evade these issues, geometrical constraints have been widely explored for learning optimal flow in self-supervised frameworks [63, 90, 99, 100]. Such methods are inexpensive and trained with only image sequences captured from RGB cameras.

Unsupervised approaches mainly rely on brightness constancy [155, 218]. Previous methods are able to produce sharp and accurate flow in clean scenes e.g., in good lighting conditions. Recent works have proposed robust methods to handle partly occluded objects or large displacements by adding extra constraints and modules [63, 90, 123]. Adverse weather and low-illumination conditions (e.g., rain, night, and fog) further introduce noise that affects pixel correspondence as shown in Figure 5.1. However, optical flow estimation in safety-critical settings such as autonomous driving [30, 168] require robust optical flow in all conditions.

A few pioneering works have investigated optical flow estimation in adverse weather conditions. However, they are mostly tailored to specific weather scenarios and rely on highly complex

pipelines and architectures [91, 209, 226, 235]. Moreover, weather-agnostic models suffer from a significant degradation when applied to a standard daytime environment [234] or need to introduce additional sensors [88, 89]. Therefore, there is a need to introduce a straightforward framework that can achieve robust optical flow estimation in both clean and challenging conditions with only RGB image input.

First, we identify two major errors in optical flow estimation under challenging conditions: motion boundary errors and noisy motion error in optical flow. The former is mainly caused by low-visibility conditions, such as night or fog, that degrade the image and feature quality and blurs the motion boundaries in optical flow, as shown in Figure 5.1 row 2, col 2. The latter occurs in adverse weather, such as rain, where flow models erroneously estimate and include motion for the rain streak, as shown in Figure 5.1 row 3, col 2. Ideally, the optical flow should be invariant for the same image pair taken under different and challenging conditions. In this work, we leverage the clean optical flow prediction as a reference to investigate the causes of the performance degradation under adverse weather conditions.

We propose a noisy motion reduction strategy with the objective to eliminate the motion of weather components (e.g. raindrops). To achieve this, we propose a duo-photometric loss L_{dph} , which uses the same training signal to guide both clean and noisy scenes. Furthermore, we generate pairs of images by adding moving weather components to static scenes and forcing the network to predict a zero-flow field. Unlike other approaches, our motion noise constraints enable the network to disregard irrelevant patterns. This prevents the network from mistaking weather components for moving objects without compromising the representation ability for clean images.

Regarding the motion boundary error, we observe a clear difference in features between clean and low-visibility images. Intuitively, if the feature is invariant to weather conditions, a model will produce a consistent flow for clean and adverse scenes. Thus, we propose a local-structure recovery method that tries to minimize the structural differences between clean and degraded features by using local cosine similarity. Our key assumption is that minimizing the feature structure differences can result in more similar optical flows. Furthermore, we propose a Confidence Map (CM), which evaluates the reliability of each pixel in clean feature. CM is used to selectively guide the degraded feature with trustworthy pixels. Figure 5.1, column 3, shows that our RobFlow method can handle various environmental changes and produces consistent flow estimates for different weather conditions, which indicates its applicability for safety-critical applications.

We summarize the main contribution as follows:

- We propose a local-structure recovery technique to cope with motion boundary errors caused by low-visibility images such as during fog and at night time. We propose a reliable

self-guidance method that filters out unreliable pixels during self-supervised learning and increases stability.

- We propose a noisy motion reduction strategy to minimize optical flow errors caused by weather elements such as rain.
- Our proposed method simultaneously improves optical flow estimation in clean scenes and in challenging conditions for synthetic and real-world data, without requiring additional parameters. Moreover, our method is the first to train on synthetic adverse weather data and surpass unsupervised methods that train on real-world adverse weather data.

5.3 Related Work

Optical Flow Estimation. Optical flow estimation is the task of estimating pixel-level motion between video frames. Recently, many deep learning based methods have been proposed to learn optical flow in a supervised manner [56, 67, 114, 176, 181, 185, 205]. PWC-Net [181] proposed a warp, cost volume architecture that estimates optical flow in a coarse-to-fine pyramid. RAFT [185] improved it by replacing the encoder with GRU and proposed a 4D cost volume that matches all pairs of pixels. GMA [67] further proposed a transformer module to capture hidden motion. To deal with the lack of real data, unsupervised methods [63, 97, 100, 123, 175] have been proposed with photometric loss. Selfflow [100] proposed to distill reliable flow estimations from non-occluded pixels, and use these predictions as pseudo-label to learn optical flow for hallucinated occlusions. ARFlow [97] proposed to use the original image as a signal to guide augmented images.

Optical Flow under Challenging condition. Optical flow estimation has achieved remarkable results in clean scenes [56, 63, 67, 97, 100, 123, 181, 185]. However, estimation remains challenging in the presence of adverse weather or illumination effects, such as rain, fog, or low-light. Previous research has mainly focused on addressing specific weather effects by designing tailored methods or datasets. RobustFlow [91] was the first method to estimate optical flow in rainy scenes, by using a handcrafted residue channel and its color variant as a prior that is invariant to rain streaks. RainFlow [92] proposed a feature mapping operation that automatically learned rain-invariant and veiling-invariant features in a supervised manner. Zheng et al. [226] developed a method to synthesize large-scale low-light optical flow datasets by simulating the noise model on dark raw images. Yan et al. [209] proposed DenseFogFlow, a semi-supervised method that modeled the flow consistency between clean images and their corresponding rendered foggy images, to overcome the difficulty of collecting ground truth for real fog scenes. Schmalfluss et al [162] propose that augmenting the training data with weather effects enhances

the robustness of optical flow methods in cross-dataset evaluation. Zhou et al. [235] presented an unsupervised domain adaptation method that adapted the model from synthetic fog images to real fog images. They propose to address motion boundary error by transferring the knowledge from a clean weather network to an adverse weather network. In contrast, we propose to address this issue by directly recovering the clean features from the adverse weather features in a single network, which also preserves the performance for clean images. GyroFlow [88] resorted to a hardware scheme to handle adverse weather, and utilized gyroscope data to obtain ego-motion labels of the camera for weakly-supervised optical flow estimation. GyroFlow+ [89] extended GyroFlow by proposing a self-guided fusion module that fused the background gyro field with the optical flow to obtain more detailed motions. However, this approach was still limited by the hardware and the availability of ego-motion labels. Compared with previous methods, our method is the first to handle both clean and challenging conditions in an unsupervised manner, without requiring any additional sensors or labels.

5.4 Method

5.4.1 Overview

In this thesis, we introduce RobFlow, a self-supervised architecture that robustly estimates optical flow in both clean and adverse conditions (e.g., rain, fog, night-time). Our approach utilizes the effectiveness of previous methods in clean scenes to reduce the two motion errors in adverse settings.

We highlight the trade-off existing models face in clean and adverse conditions and lack of generalization to unseen data. The pipeline of our novel framework is shown in Figure 5.2. In our experiments, we use SelfFlow [100] and ARFlow [97] as baseline methods. Our method does not increase parameters and memory during testing.

5.4.2 Preliminaries

Clean-to-adverse translation. To achieve the above, we need correspondence between challenging and clean samples. Following [161], we use a physics-based renderer (PBR) [187] to generate synthetic rain, and fog. CoMoGan [149] is used to generate night time for the KITTI dataset. These two methods can preserve the original image as the background, and overlay adverse effects in the foreground.

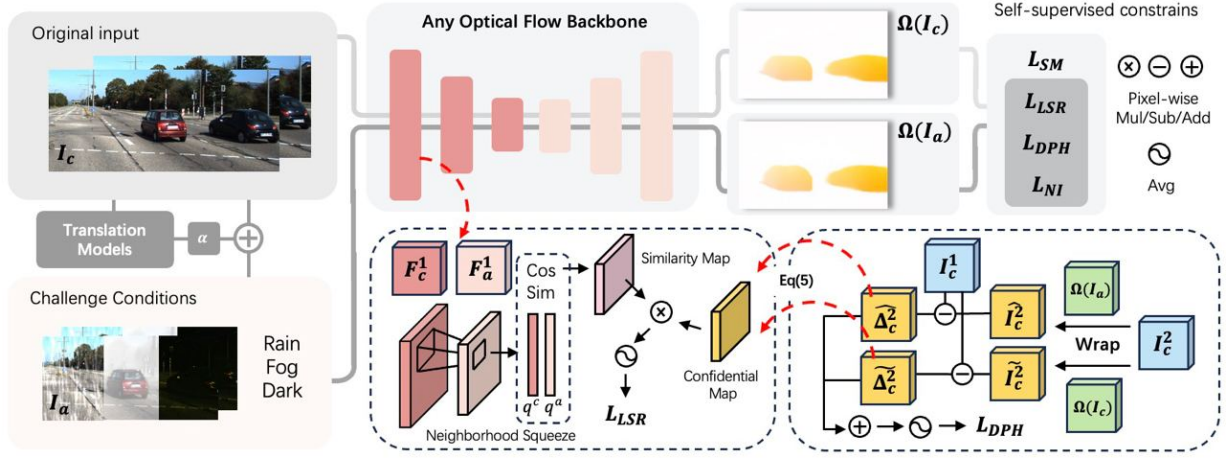


Figure 5.2: Overall architecture of RobFlow. The corresponding challenging pairs are generated before training. In each step, we feed the original and a random challenge pair to the network, and align the feature and motion by our proposed methods.

We apply the above methods to generate one-to-many projections, where one clean image corresponds to rain, fog, and night images. During training, we choose a clean and a random adverse image as a corresponding pair. For clarity, we denote $I_c = (I_c^{(1)}, I_c^{(2)})$ as clean images, and denote $I_a = (I_a^{(1)}, I_a^{(2)})$ as corresponding degraded images because of, e.g., rain or fog. We select ARFlow [97] as backbone. In each training step, the network predicts **clean flow** $\Omega(I_c)$ by $(I_c^{(1)}, I_c^{(2)})$ and generates **degraded flow** $\Omega(I_a)$ by $(I_a^{(1)}, I_a^{(2)})$. Following the original methods, we keep the smoothing loss along with the respective method’s specific constraints for the clean flow $\Omega(I_c)$, and we propose our loss function as extra constraints to enhance the network’s robustness for all the above conditions.

5.4.3 Noisy Motion Reduction

Duo-Photometric Loss. In unsupervised optical flow estimation, a photometric loss L_{ph} is used as the main constraint. Adverse weather breaks the brightness constancy [234], previous methods [234, 235] use clean optical flow as a pseudo-label to guide degraded flow with a self-supervised loss, as shown by L_{self} in Equation (5.1).

$$L_{ph} = \sum |I_c^{(1)} - W(I_c^{(2)} \Omega(I_c))| \quad L_{self} = |\Omega(I_c) - \Omega(I_a)| \quad (5.1)$$

However, this approach is constrained by the accuracy of the clean flow estimation, as the

clean flow may generate errors during the unsupervised training, which may mislead the network when guiding the degraded flow. Moreover, based on Equation (5.1), the clean and degraded flows are trained with different signals: the estimation of clean flow is learned by the photometric loss while estimation of degraded flow is learned from the clean flow. This strategy could result in a discrepancy between the final outputs of clean and adverse conditions.

As an example, imagine in the i -th iteration, we generate a clean flow from a clean image and a rainy flow from a corresponding rainy image. We use the clean flow to guide the rainy flow, which will produce errors if any pixels of the clean image have lower accuracy than the rainy flow. Therefore, we propose an efficient strategy that lets the clean flow and the rainy flow learn from the same signal, i.e., the clean image. By exploiting the corresponding pairs of clean and adverse images, we can leverage the clean image as guidance in calculating the photometric loss for the rainy optical flow. The duo-photometric (DPH) loss L_{dph} , can be defined as follows.

$$L_{dph} = \sum |I_c^{(1)} - W(I_c^{(2)}, \Omega(I_c))| + \sum |I_c^{(1)} - W(I_c^{(2)}, \Omega(I_a))| \quad (5.2)$$

In DPH loss, the degraded optical flow is not influenced by the quality of the clean flow, and both optical flow estimations learn from the same signal, which results in more consistent outcomes.

Noise Invariant Loss. As shown in Figure 5.1, the degrading conditions in images, e.g., rain streaks, will introduce noise in the predicted optical flow, which is led by the correlation operation [56, 97, 181]. This operation matches image regions of similar intensity patterns, and then estimate optical flow. However, the rain streaks is treated as similarly objects and their motion can be incorporated into the optical flow which is unnecessary for real-world flow estimation. To address this issue, we re-examine the relationship between the motion of clean and degraded scenes. Given a pair of corresponding clean images $I_c^{(1)}, I_c^{(2)}$ and a pair of degraded (e.g. rainy) images $I_a^{(1)}, I_a^{(2)} = I_c^{(1)} + \alpha_1, I_c^{(2)} + \alpha_2$, where α is the component due to the degradation. Then the motion of the degraded images can be expressed as $\Omega(I_a) = \Omega(I_c) + \Omega(\alpha) + \Omega(\alpha, I_c)$. Here, $\Omega(\alpha)$ is the motion of the image component causing the degradation (e.g. raindrops or rain streaks). $\Omega(\alpha, I_c)$ is the motion caused by the interaction of image components due the degradation and original objects.

To enhance the robustness of our network to such motion noise, we aim to minimize $\Omega(\alpha)$ and $\Omega(\alpha, I_c)$. We adopt an independent forward pass in the training process to achieve this without hurting the performance on clean scenes. Inspired by Jeong et al. [64], we use one image from the clean pair, i.e., $I_c^{(1)}$, as a static scene, and add different moving degrading image components (rain streaks) to obtain a new pair $I_c^{(1)} + \alpha_1, I_c^{(1)} + \alpha_2$ and force the flow to be estimated as zero. This process is shown in Equation (5.3). Compared with [64], the main difference is that they use a mask with static images to generate occlusion pairs, while we force the network to ignore

moving motions of degrading image components such as rain streaks.

$$L_{NI} = \sum |\Omega(\alpha)| + \sum |\Omega(\alpha, I_c)| \quad (5.3)$$

Building on this idea, we can further reduce the domain gap between different adverse conditions. For example, we use α_f and α_d as fog and darkness components, respectively, and generate a new pair $I_c^{(1)} + \alpha_f, I_c^{(1)} + \alpha_d$ and again force the flow to be estimated as zero.

5.4.4 Local-Structure Recovering

To further reveal the impact of different adverse conditions on the feature extraction of unsupervised models, we conduct analytical experiments on the synthetic KITTI [34] dataset.

Taking ARFlow as an example, we compared the pixel-wise difference of features from the encoder between clean and foggy images. As illustrated in the left panel of Figure 5.3, there is a clean contrast inside the red rectangle between the clean and fog features extracted by ARFlow. As a result, the optical flow for foggy images exhibit a blurred motion boundary in the same region compared to the clean flow. This leads us to hypothesize that having similar feature maps in the encoder part will produce more consistent optical flows.

Therefore, we aim to minimize the discrepancy between the clean and challenging weather features. However, the features from different domains are difficult to project in a homogeneous space [234]. Instead of forcing them to have the same features, we relax the objective to have similar local structures for motion estimation. To this end, we propose the local-structure recovery (LSR) loss. Given the input I_c and I_a , the features $\mathcal{F}_c^i$ and $\mathcal{F}_a^i$ from the i -th block of the encoder can be obtained. Using $\mathcal{N}(p_{ij}, d)$ to represent the d -order neighbourhood of pixel p_{ij} in the feature space, $\bigcup$ represent the flatten operation, the local information $\mathbf{q}$ can be defined as

$$\mathbf{q} = \bigcup_{\mathcal{N}(p_{ij}, d)} p_{ij}. \quad (5.4)$$

Next, we try to recover $\mathbf{q}^a$ from the corresponding $\mathbf{q}^c$. According to our observation in Chapter 5.4.3, the flow from clean features is not guaranteed to be correct. The clean features may also not guaranteed to be better than degraded feature for each pixels. Intuitively, we could reduce the error by ignoring those unreliable pixels. Thanks to our DPH Equation (5.2), both optical flow estimations are optimized using the same clean reference image. We can obtain two error maps by computing the difference between the reference image and the target image warped by either the clean or the degraded flow. By comparing the corresponding pixels in the two error maps, the confidence map is determined by whether a pixel in the clean flow has smaller errors than

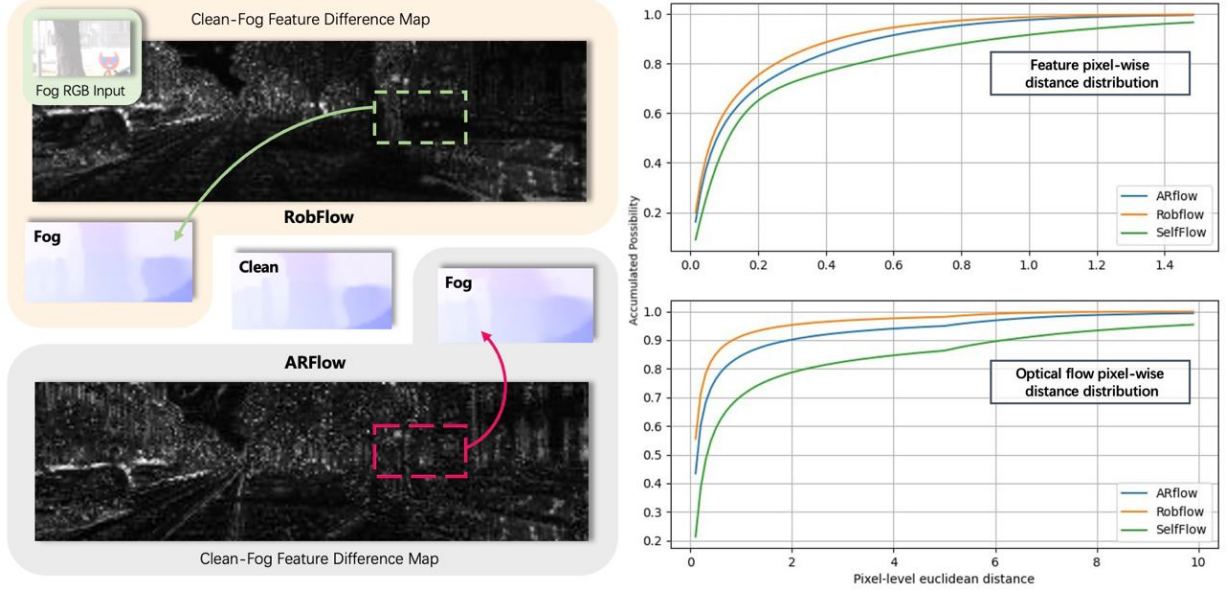


Figure 5.3: The relation between features and optical flow. On the left, we show the features difference map between clean and foggy images. Our RobFlow method generates more similar features for clean and foggy images, and the optical flow in foggy conditions is closer to the optical flow in clean conditions. We report the Cumulative Distribution Function (CDF) of the distance between clean and foggy conditions in the KITTI dataset [34]. The top right figure shows the pixel-wise distance between clean and fog features. The bottom right figure shows the pixel-wise accuracy of fog optical flow and ground truth.

the degraded flow. Inspired by [37], we use it to filter out unreliable pixels if the clean flow error is larger than the degraded flow error. The remaining reliable pixels p_{ij} are obtained by filtering with $\mathcal{K}$ are defined as follows:

$$\mathcal{K} = \{p_{ij} \mid |I_c^{(1)} - \mathcal{W}(\Omega(I_c), I_c^{(2)})| < |I_c^{(1)} - \mathcal{W}(\Omega(I_a), I_c^{(2)})|, p_{ij} \in I_c^{(1)}\} \quad (5.5)$$

Finally, we use the cosine similarity to align features and the local-structure recovery loss can be described by

$$\mathcal{L}_{lsr} = \frac{1}{|\mathcal{K}|} \sum_{p_{ij} \in \mathcal{K}} 1 - \frac{\mathbf{q}^a \mathbf{q}^c}{\|\mathbf{q}^a\| \times \|\mathbf{q}^c\|} \quad (5.6)$$

$\|\cdot\|$ denotes the Euclidean norm and $'$ denotes the stop gradients. In practice, we use the intermediate flow of the same level as the feature map to generate the confidence map for the

Methods	Clean		Rain		Fog		Dark	
	EPE	F1-all	EPE	F1-all	EPE	F1-all	EPE	F1-all
<i>ARFlow</i> [†] [97]	2.84	9.87	15.65	62.68	5.43	15.79	8.22	26.85
ARFlow [97]	4.51	14.42	8.48	47.07	4.78	14.17	7.23	23.58
ARFlow-D	3.27	10.91	8.63	48.11	4.27	13.28	6.82	20.75
SelfFlow [183]	5.58	15.83	10.42	53.21	5.27	15.64	8.13	26.61
SelfFlow-D	5.02	15.12	10.24	52.14	4.92	14.83	8.54	27.25
GMA [67]	3.32	10.73	6.47	27.36	3.60	12.39	5.14	17.90
HMBA-Flow [234]	5.03	15.19	<u>3.62</u>	<u>14.89</u>	<u>2.91</u>	<u>9.94</u>	<u>3.88</u>	<u>14.74</u>
RobFlow*	3.92	13.37	4.25	16.42	4.06	12.94	5.96	19.49
RobFlow	2.68	9.36	3.50	14.58	2.79	9.87	3.47	12.96

Table 5.1: Quantitative results on KITTI datasets: D stand for the denoise module, we use JORDER-E [211] for deraining, AECR-Net [199] for defog and MCR [26] for dark images, †: only training on clean dataset, zeroshot inference on challenging conditions. **Bold** text represents the best result for the metric for each test. Whereas, the underlined text represents the second best for the metric of each test.

feature with the same resolution. Figure 5.3 top left, visualizes the difference map of foggy and clean features of RobFlow. Our method effectively reduces the distance between the clean and foggy features, and predicts a more accuracy flow in green rectangle. Figure 5.3 top right shows the distribution of the level-wise feature distance between fog and clean condition for the KITTI dataset. We observe that our RobFlow has a statistically smaller distance than ARFlow and Selfflow. The bottom right figure shows the pixel-wise accuracy of fog optical flow and ground truth (clean flow is closer to the GT overall). By considering these two figures, we observe that the similarity of fog features and clean features is positively correlated with the similarity of fog optical flow and clean optical flow which supports our claim.

The total loss of our RobFlow framework is described as follows:

$$\mathcal{L}_{all} = L_{dph} + \omega_0 L_{ni} + \omega_1 L_{lsr} + \omega_2 L_{sm} + \omega_3 L_{aug} \quad (5.7)$$

where L_{sm} is smooth loss [123] and L_{aug} is augmentation loss [97].

5.5 Experiments

Methods	L_{dph}	L_{self}	LSR	C.M	L_{NI}	Clean		Rain		Fog		Dark	
						EPE	F1-all	EPE	F1-all	EPE	F1-all	EPE	F1-all
<i>ARFlow</i> [†] [97]						2.84	9.87	15.65	62.68	5.43	15.79	8.22	30.04
ARFlow [97]		✓				11.72	28.04	22.28	73.34	12.19	30.66	14.02	26.85
ARFlow [97]		✓ [*]				4.51	14.42	8.48	47.07	4.78	14.17	7.23	23.58
RobFlow	✓					3.21	12.24	7.79	44.01	3.94	13.94	5.62	18.76
RobFlow	✓		✓			3.18	12.35	6.75	32.35	3.67	12.26	4.27	16.59
RobFlow	✓		✓	✓		2.91	10.71	6.63	29.47	3.42	11.68	3.92	15.27
RobFlow	✓				✓	3.13	12.43	5.77	22.18	3.85	13.76	5.43	16.43
RobFlow	✓		✓	✓	✓	2.68	9.36	3.50	14.58	2.79	9.87	3.47	12.96

Table 5.2: Ablation study: L_{dph} indicates Equation (5.2), L_{self} indicates Equation (5.1), LSR indicate the local-similarity recovery without confidence map, C.M indicates confidence map, L_{ni} indicates Equation (5.3). ✓^{*} is a small scale loss (e.g. 0.01).

5.5.1 Experiments Setup

Datasets. We conduct experiments on synthetic and real-world datasets. Our synthetic dataset is based on KITTI 2015 [34], we generate synthetic rain, fog with physics-based rendering [187], and generate night time images by CoMoGAN [149]. GHOF [89] is a real-world dataset with around 10,000 pairs for training and 530 pairs for evaluation. Beyond this, we also collect more real-world data and evaluate it in the supplemental material. We use EPE [27] and F1-all [34] metrics for evaluation.

Comparison Methods. Two competing methods are GyroFlow+ [89] and HMBA-Flow [234]. However, the authors of HMBA-Flow do not supply any code, or data for evaluation. Thus, we reproduce the method and include it in our evaluation. Furthermore, we select recent SOTA unsupervised methods SelfFlow [100], ARFlow [97], and UPFlow [118] which are designed for clean scenarios and select SOTA supervised methods GMA [67]. For a fair comparison, supervised methods were pre-trained on synthetic datasets with optical flow labels and then fine-tuned on target datasets with a self-supervised strategy [175]. Unsupervised methods were directly trained on the KITTI multiple-view dataset or GHOF training set without seeing testing data. We also

experiment with extra training strategies for unsupervised methods by first denoising images (de-raining or defogging) during training before testing in the denoised images. Our RobFlow uses ARFlow [97] as the backbone. RobFlow* uses Selflow [100] as the backbone. For a fair comparison, we adopt the same hyperparameter settings from the original works, such as batch size, learning rate, optimizer, and image augmentation. Neither of the proposed methods introduces any additional parameters compared to the original method.

5.5.2 Experiments on Synthetic Images

Table 5.1 compares the results of our model, RobFlow, with previous state-of-the-art methods on the Synthetic KITTI 2015 test set. We make the following observations: Both RobFlow and RobFlow* outperform the original network on both clean and challenging conditions. RobFlow outperforms all compared methods on the original KITTI dataset, and achieves clearly better results than all previous state-of-the-art methods on the adverse weather test set. As a baseline model, ARFlow [97] (Row 1), trained only with clean KITTI images, fails to generalize to adverse weather conditions. ARFlow [97] (Row 2), trained with both clean and weather images, improves its performance on the adverse weather test set, but degrades its performance on the clean test set. The denoising methods (rows 3 & 5) improve the results for foggy images, but worsen the results for rainy or night images, because they unintentionally damage to the image details negating the benefits of noise removal.

5.5.3 Experiments on Real Images

In Table 5.3, we report qualitative results on a real-world dataset. The first three rows are state-of-the-art (SOTA) supervised methods trained with synthetic data as reported by Li et al. [89]. Since they do not perform well, we then fine-tune the best method GMA [67] on the GHOF dataset by using a self-supervision strategy [175] listed as GMA-S. We observe an improvement on every metric. In terms of adverse weather data, it is still better than Gyroflow+ and HMBA-Flow, which are well designed for adverse weather. HMBA-Flow achieves four second-best results, however, it shows unsatisfactory results in clean data, because it aims to make robustness for adverse weather conditions by domain adaptation without handling clean data well. Gyroflow+ achieves better results in clean data and competitive results for adverse weather data, but it requires extra gyroscope information as input. On the other hand, our method achieves the best results for both clean and adverse conditions, without extra input. Furthermore, all unsupervised methods are trained with the full GHOF training dataset, while our method is trained with only the clean part of the GHOF training dataset, which is 5.22% of the complete dataset. We use clean images

Methods	Strategy	RTData	Clean		Rain		Fog		Dark	
			EPE	F1-all	EPE	F1-all	EPE	F1-all	EPE	F1-all
PWC-Net [181]	Sup	0%	2.60	-	1.67	-	4.65	-	6.34	-
RAFT [185]	Sup	0%	2.77	-	0.88	-	5.87	-	13.26	-
GMA [67]	Sup	0%	1.31	-	1.73	-	0.69	-	4.87	-
GMA-S [67]	Semi	100%	<u>1.02</u>	<u>4.52</u>	0.51	1.02	1.36	3.64	3.49	14.21
ARFlow [97]	Unsup	100%	1.99	8.83	1.04	2.63	2.86	10.49	5.14	21.52
UPFlow [118]	Unsup	100%	1.08	4.83	0.56	1.26	1.18	3.41	3.39	12.83
HMBA-Flow [234]	Unsup	100%	2.29	12.18	0.46	0.44	<u>1.04</u>	<u>3.26</u>	<u>2.17</u>	9.72
Gyroflow+ [89]	Unsup	100%	1.04	4.79	<u>0.48</u>	<u>0.41</u>	1.14	3.43	2.20	<u>9.65</u>
RobFlow	Unsup	5.22%	0.84	3.54	0.41	0.32	0.94	2.72	1.87	9.45

Table 5.3: Quantitative results on the GHOF datasets: RTData denotes the real training data in GHOF. All supervised methods are trained only on synthetic data and do not have access to real data before testing. Semi-supervised and unsupervised methods use real data during training. However, our method uses only clean real data and synthetic weather data during training, which constitutes only 5.22% of the total data compared to other unsupervised methods.

to generate synthetic weather for rain, fog, and darkness. Therefore, our method never sees real-world adverse weather before testing. We argue that this is an important benefit, since in real-world applications, it is crucial to generalize to new and more challenging conditions that are not included in the training set. Our method achieves 41 ms on images of size 375×1242 using a single 3090 GPU.

Figure 5.4 visualizes qualitative results from ARFlow, HMBA-flow, Gyroflow+ and our method on the GHOF evaluation set. We see that the original ARflow can generate reasonable results in clean scenes, but generates large errors in bad weather scenes. HMBA-flow generates noise in clean scenes and blurs motion boundaries in dark scenes. Gyroflow+ also gets noisy results in rainy scenes and mixes the motion boundary of different objects in dark scenes. Our method achieves smooth results and succeeds in predicting the important optical flow for the car in all conditions.

Methods	Snow		I.C.		Complex		Snow-R	
	EPE	F1-all	EPE	F1-all	EPE	F1-all	EPE	F1-all
SelfFlow [100]	5.82	16.03	8.41	27.24	9.87	51.27	-	-
ARFlow [97]	4.89	14.86	7.72	23.71	8.05	48.12	6.39	30.64
GMA [67]	3.54	11.04	5.64	18.14	5.97	23.41	5.32	21.94
HMBA-Flow [234]	3.20	10.63	4.01	15.32	4.72	18.95	3.11	15.15
RobFlow	2.81	10.42	2.96	10.50	3.52	13.21	1.26	4.27

Table 5.4: Quantitative results on Zero-shot experiments. Left three column are synthetic data, Snow-R are real-world data.

5.5.4 Ablation study

This section presents an ablation study to examine the contribution of each component in our method. We evaluate our model on both clean and adverse weather datasets to demonstrate its robustness and generalization in Table 5.2.

The first row shows the results of ARFlow trained with only clean images, which lacks the ability to handle adverse conditions.

Self-supervision (rows 2 & 3). When training the network with Equation (5.1) that uses clean optical flow to guide adverse condition optical flow, the clean flow may have errors that affect the backpropagation and degrades the model performance for all images. When we reduce the self-supervision loss, the network improves the results for adverse condition images, but deteriorates the results for clean images. This trade-off exists in most of the previous methods.

Duo-photometric loss (row 4). We optimize the network for both the clean and adverse conditions data simultaneously using the duo-photometric loss. We observe improvements for both clean images and adverse condition images. However, the performance of clean scenes is still not comparable with the original network (row 1), which motivates us to further reduce motion errors.

Local-structure recovery (rows 5 & 6). We investigate the effects of LSR (Equation (5.7)) and the confidence Map (Equation (5.5)). First, we reduce the difference between clean and weather features by exploiting the consistency of clean images. This leads to a significant improvement in both clean and images in adverse conditions, especially in dark images. Second, we evaluate the impact of the confidence map that filters out unreliable pixels for guidance. This also results in a reduction of the loss for all test sets. This confirms our hypothesis that reducing the errors in guidance benefits both clean and adverse conditions performance.

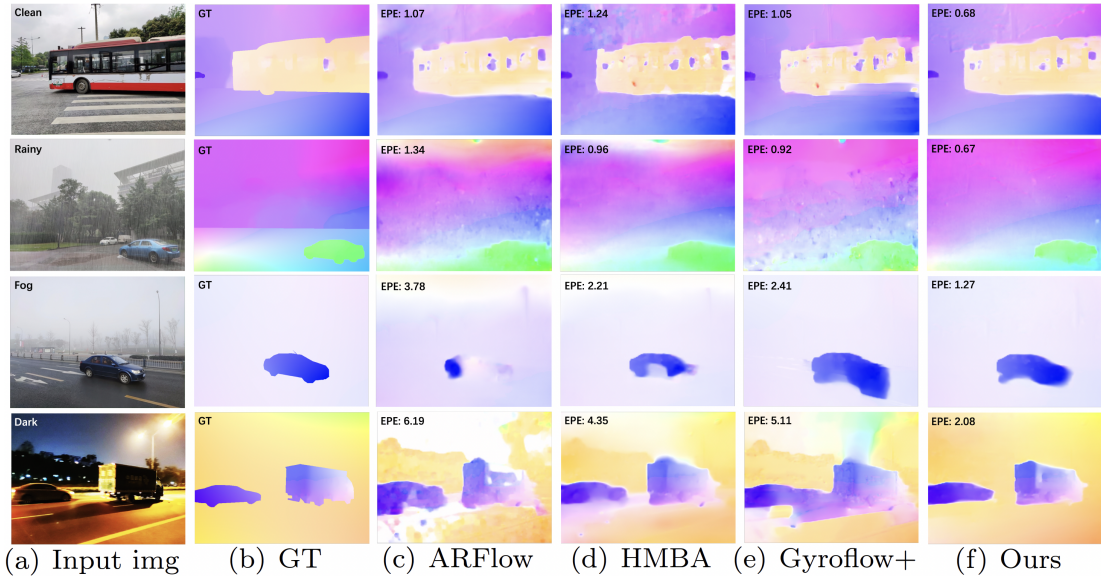


Figure 5.4: Qualitative results of optical flows on GHOF dataset.

Optimising for Weather Motion (row 7). Although local-structure recovery is beneficial, we can further improve the robustness of our method by noisy motion reduction, especially for rainy images. Unlike foggy and dark images, the main challenge of rainy images is that they contain raindrops that interfere with the optical flow estimation. Row 7 shows an increase in the performance for rainy images. The reasoning behind this is that Equation (5.3) forces the network to ignore distracting motion such as raindrop motion and also reduces the domain gap between different conditions.

ALL (row 8). Our model trained with all of the contributions leads to clean results that surpass ARflow and have the best bad weather performance in row 8. We show that all components are effective in reducing the metrics for adverse condition metrics while maintaining the performance of clean data.

5.5.5 Zero-shot Experiment

To test the generalization ability of our method in real-world applications, we also conduct zero-shot experiments on different domains that are not seen during training. We use CoMoGAN [149] to synthesize images with different lighting conditions, such as dawn and dusk, and [1] to generate images with snow from the KITTI 2015 dataset. Then, we combine these techniques to create more complex scenarios, such as fog+dusk, fog+night, and rain+night. We further test

our methods on real-world data; we take all snow testing data from GHOF called Snow-R. We use the same weights that have been trained on previous experiments and evaluate the respective method on this new test set without any fine-tuning. Table 5.4 reports the results of the zero-shot experiments in five categories: snow, illumination change (I. C.) (dawn and dusk), complex (fog+dusk, fog+night, etc) for synthetic data and snow-R. Our method outperforms all comparison and shows robustness on unseen data, which indicates that it could be a potential method for real-world applications. We collect more complex real-world data and evaluate in the supplemental material.

Index	Rain	Fog	Dark
L1	0.57	1.36	2.41
Cosine Similarity	0.48	1.20	2.08
LSR(1)	0.41	0.94	1.87
LSR(2)	0.42	1.02	1.91

Table 5.5: Discussion on feature loss. (i) is i-order neighbourhood.

Image Translation	Rain	Fog	Dark
Paste	0.53	1.04	-
CycleGAN [240]	0.45	-	1.94
CoMoGAN [149]	-	-	1.87
PBR [187]	0.41	0.94	-

Table 5.6: Discussion on different image translation strategies.

5.5.6 Discussion

Image Translation. Our proposed method requires corresponding pairs of images to establish a relationship between the clean and adverse conditions domains. Therefore, the choice of image translation model is crucial. We compare different strategies to generate adverse weather images. 1. We use software tools such as Adobe After Effects and Photoshop to create weather effects and overlay them on clean images. 2. We use generative adversarial networks (GANs) such as CycleGAN [240], or CoMoGAN [149] to produce realistic weather images. 3. We use physics-based rendering (PBR) [187] to simulate the optical effects of weather conditions. We training our methods with synthetic data by the above methods, and test in real data. We present the results in Table 5.6, and observe that CoMoGAN achieves the best performance for dark images, and PBR for rain and fog images. This is because CoMoGAN and PBR preserve the structure of the original images better than the other methods.

Importance of Duo-photometric loss. As shown in Table 5.2, our duo-photometric (DPH) loss outperforms the self-supervised loss based on clean flow. We hypothesize that the main reason is that clean flow introduces errors during training, which degrade the learning quality of flow in adverse conditions. By using the DPH loss, the flow in challenging conditions is less affected by the errors in the clean flow, and thus flow results are improved. Previous experiments

have supported our hypothesis. We analyzed and demonstrated in the supplemental material that the clean flow generates errors while the degraded flow is more accurate in some pixels. We further discuss the limitations in supplemental material.

Importance of Local-structure Recovery. To evaluate the impact of the flow encoder, we conducted an experiment with three different methods for aligning clean and noisy features: L1 distance, cosine similarity, and our proposed local structure recovery. We observed that our local-structure recovery achieves better results than the other two methods. The reason is that the L1 distance or vanilla cosine similarity attempts to align pixel-level features for inputs of different domains, which is hard to achieve in a shared encoder. In an optical flow estimation model, we are more concerned about structure information (shape, geometry) than appearance information. Therefore, we only want to align local-structure information as a soft guidance.

5.6 Summary

In this chapter, we propose RobFlow, a simple and efficient unsupervised framework that can handle both clean and adverse weather conditions, which are a major obstacle to real-world applications (e.g., autonomous driving). To achieve this, we observed that low-quality features caused by low-visibility scenes could lead to boundary errors. Thus, we use local-similarity recovery to reduce the distance between features of clean and low-visibility images. We further propose noisy motion reduction to reduce noisy motion in optical flow caused by weather components. We have performed comprehensive experiments to demonstrate the robustness and generalization ability of the proposed RobFlow. In the subsequent chapter, we discuss how to systematically evaluate the robustness of optical flow methods.

Chapter 6

Shape and Texture: What Influences Reliable Optical Flow Estimation?

6.1 Introduction

In the previous chapter, we discussed how to make optical flow methods work better in real-world environments. However, we are still missing a system evaluation of robustness for optical flow methods. Existing studies mainly focus on improved accuracy in benchmarks and often overlook the analysis of network robustness, which is essential for safety-critical scenarios such as autonomous driving. In this chapter, our aim is to discuss the robustness evaluation of optical flow methods.

This chapter resulted in a publication [107]¹.

6.2 Overview

To enhance the robustness of optical flow methods, various solutions have been proposed. Some studies have addressed occlusion maps using dedicated networks [56, 141, 192]. Others have incorporated attention mechanisms to improve performance, particularly for large displacements [176]. Recently, studies have begun to replace convolutional neural networks (CNNs) with transformer

¹© 2025 IEEE. Reprinted, with permission, from Libo Long, Xiao Hu, and Jochen Lang, "Shape and Texture: What Influences Reliable Optical Flow Estimation?", in *CVPR*, 2025.

blocks to further improve accuracy [50, 114, 169]. However, a systematic analysis of the robustness of deep models in optical flow is still lacking.

Inspired by robustness evaluation datasets in other computer vision tasks [71, 95], we introduce Flow-R, a synthetic test set generated by a diffusion model for optical flow tasks. Unlike existing datasets, we modify the shape, texture, and other properties of a key object (e.g., a car) in the KITTI [34] benchmark and evaluate the change in the output compared to the original results. This allows us to measure the sensitivity of the method to certain types of semantic changes in the input. Through this, we aim to explore two fundamental questions about optical flow methods:

1. Shape vs. Texture: Do optical flow methods estimate motion primarily based on an object’s shape or texture?

2. Memorization: Do these methods tend to memorize objects from previously seen scenes?

Our Flow-R test set includes three subsets: Flow-R-T for texture replacement, Flow-R-S for shape replacement, and Flow-R-O for adding objects. Figure 6.1 presents two samples of data. We develop an UI-based application using the Segment Anything Model (SAM) [75] to generate corresponding masks of the main objects in each pair of images. We then manually select high-quality samples.

To build Flow-R-T, we apply a set of 24 common texture prompts to ControlNet with specific masks to generate new textures for an object. We evaluate the performance of state-of-the-art (SOTA) optical flow methods on Flow-R-T and find that these methods can still match the motion of an object with entirely different textures.

To build Flow-R-S, we directly scale the selected object in the image by a random factor θ and assign an image motion to the object based on its real non-linear motion. We demonstrate that optical flow methods rely more on shape information than texture, and changes of the object flow can significantly affect the optical flow of the background.

To build Flow-R-O, we introduce a set of 13 unseen objects, i.e., objects that do not occur in the training data, and 22 seen objects. We randomly select and add one of these objects to a KITTI image. By comparing the performance of the optical flow method on images with seen vs. unseen objects, we show that SOTA methods can easily estimate the motion of seen objects but tend to ignore the motion of unseen objects.



Figure 6.1: **Samples of Flow-R.** Original and modified images (Row 1 and 3), ground truth optical flows (Row 2 and 4) are shown. The first column displays the original data from the KITTI dataset. The second column shows the data with the texture of the main object of car replaced. The third column shows the data with the shape of the main object rescaled. The fourth row presents the data with an unseen object added.

6.3 Related Works

6.3.1 Optical Flow Estimation

Recent optical flow methods [56, 181, 210, 222] have achieved significant progress in benchmarks [34]. RAFT [185] proposes a 4D cost volume and iterative refinement architecture as a foundational structure for recent methods. CRAFT [176] introduces a cross-attention mechanism to revitalize the cost volume for large displacements. GMFlow [205] proposes directly matching the global motion, then propagating local flow with self-attention. FlowFormer [50] introduces a transformer-based architecture for optical flow estimation. FlowFormer++ [169] enhances the cost volume through masked autoencoding. Recently, SAMFlow [237] embeds the encoder of SAM [75] into FlowFormer to address the fragmentation problem.

However, the primary evaluation of optical flow methods is by measuring performance on benchmarks (e.g., [5]). Common benchmark datasets include the synthetic Chairs [27] and FlyThings3D [121] as well as the computer graphics (CG) based MPI Sintel [17] and real-world

KITTI [34, 125]. Further datasets produced with a game engine, e.g., Virtual KITTI [32] or with the KUBRIC [39] simulator, or based on a CG movie such as the high-resolution SPRING [122]. Real-world datasets are more challenging to collect. HD1K [76] contains driving scenarios but different from KITTI includes various challenging imaging conditions. None of these datasets are designed to study the sensitivity of optical flow method to changes in the texture and shape of objects in the scene or the robustness of optical flow to unseen objects. Unlike previous studies, we focus with Flow-R on identifying the common “weaknesses” of state-of-the-art (SOTA) methods, which is crucial for safety critical real-world applications based on optical flow.

6.3.2 Adversarial Attacks in Optical Flow Estimation

To study the robustness of optical flow estimation, many adversarial attack methods have been proposed. Ranjan et al. [153] proposed learning a small patch to attack optical flow methods. Schrodi et al. [165] introduced a less constrained global attack on optical flow, inspired by the Iterative Fast Gradient Sign Method (I-FGSM) [83]. Koren et al. [80] proposed corrupting the flow predictions of specific object categories using a regularization term that promotes consistency in off-target predictions. Schmalfluss et al. [164] proposed the Perturbation-Constrained Flow Attack (PCFA), which emphasizes the destructiveness of adversarial perturbations while maintaining control over their magnitude. They also proposed adversarial weather attacks [163], which simulate weather noise such as snowflakes using a 3D rendering framework.

Those adversarial attack methods all rely on training and adding perturbations to make the network fail. In contrast, our methods directly add real-world objects instead of perturbations and do not require any training. Most adversarial attack methods aim to explore “worst-case” robustness with specifically designed patches or perturbations, which are unlikely to occur by chance in real-world scenarios. The goal of an adversarial attack is to make the network fail. In contrast, we analyze cases that optical flow methods may face in real-world situations even without an adversary. Our goal is to analyze when and how optical flow methods fail in general scenarios.

6.4 Methods

6.4.1 Data Manipulation

Figure 6.5 indicates the main pipeline of our methods. To construct Flow-R, we utilize the Segment Anything Model (SAM) [75] to identify the corresponding pairs of main objects across

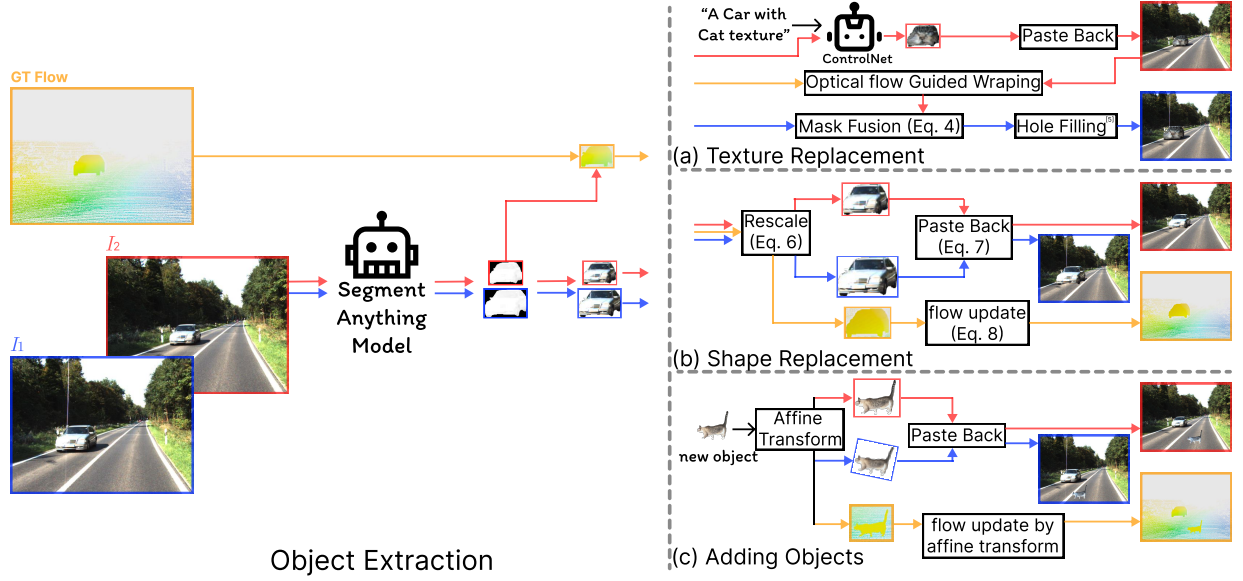


Figure 6.2: **Pipeline of our methods.** Given the data I_1, I_2, f_{gt} from KITTI, we manipulate the data by changing the texture or shape of object or adding new objects.

images. Given a pair of images I_1, I_2 , we generate a mask with SAM for object x :

$$S_1, S_2 = SAM(I_1, I_2, Prompt(P_1, P_2)) \quad (6.1)$$

Where P_1 and P_2 are sets of indices corresponding to object x in each image. S_1 and S_2 are binary masks for object x in I_1 and I_2 .

Flow-R consists of three parts. Flow-R-T includes images with different textures of the same object, which we use to evaluate the performance of methods with unseen textures. Flow-R-S modifies the shape of the object by scaling with a factor θ . Flow-R-O adds new unseen objects or replaces object x with a new object in I_1 and I_2 .

Texture Replacement (Flow-R-T). We employ a stable diffusion model [158] to generate new textures for object x . The diffusion model is capable of producing diverse textures but our approach remains compatible with other generative models that can be directed through language prompts.

To generate the texture with the same shape of the object, we incorporate ControlNet [225]

to guide the generation process. We generate the new texture for I_2 as follows

$$Image(N) = ControlNet(N, S_2) \quad (6.2)$$

where N represents the texture within the region defined by S_2 . S_2 acts as a prompt to focus ControlNet on generating an object with the same shape. The new I_2^N with texture N can be presented as:

$$I_2^N = I_2 \times (1 - S_2) + Image(N) \times S_2 \quad (6.3)$$

To preserve consistency of optical flow, we use ground-truth optical flow f_{gt} to warp the new object from I_2^N to I_1^N .

$$I_1^N = I_1 \times (1 - S_1) + warp(I_2^N, f_{gt}) \times S_1 \quad (6.4)$$

Finally, we use Bi-directional Hole Filling [43] to fill the holes caused by occlusion. The optical flow remains the same as the original. The generated new data is (I_1^N, I_2^N, f_{gt}) .

Shape Replacement (Flow-R-S). We scale the object shape x by a factor θ which allows to use it's real motion instead of adding unreal motion as in [121]. Specifically, we directly upscale the object x in I_1 and in I_2 with a scale factor $\theta > 1$. Then, we paste it back aligned with the center of its own respective bounding box in the image.

$$I_1^x = I_1 \times S_1 \quad (6.5)$$

The center point (c_1^x, c_1^y) of the bounding box of x in I_1^x can be easily calculated. The new image I_1^s can be presented as:

$$I_1^{x\uparrow} = Upsample(I_1^x, \theta) \quad (6.6)$$

$$I_1^s = Paste(I_1, I_1^{x\uparrow}, (c_1^x, c_1^y)) \quad (6.7)$$

Where Paste() is used to paste the scaled x back to the position (c_1^x, c_1^y) in I_1 . Then, by using the ground truth of optical flow, we can trace the center position of the object in I_2 . Using the same process, we get I_2^s .

Given the center position (c_1^x, c_1^y) for x in I_1 and (c_2^x, c_2^y) in I_2 , the optical flow need to be recalculated in object area. We use equations (6.5) to (6.7) to get the upsampled x optical flow $f_{gt}^{x\uparrow}$. Then, for all pixels in $x \uparrow$, the flow can be updated by:

$$f_{ij}^s = \theta f_{ij} + (\theta - 1)(c_1^x - c_2^x, c_1^y - c_2^y), f_{ij} \in x \uparrow \quad (6.8)$$

Where f_{ij} contains the optical flow for the 2D offset at index i, j .



Figure 6.3: **Texture Attack.** The texture of the main object (car) in the first frame is replaced, while the second frame remains unchanged. The resulting optical flow is similar to that derived from the original images.

Adding Objects (Flow-R-O). We collect real-world image I_M include unseen objects. Then, we get new objects y from I_n using the SAM model.

$$S_M = SAM(I_M, Prompt(P_M)) \quad (6.9)$$

$$y = I_M \times S_M \quad (6.10)$$

We paste y in I_2 and move it to I_1 using an affine transform, then recalculate the optical flow f_{aff} . The new data is (I_1^y, I_2^y, f_{aff}) .

Quality Control by Human. The above process will automatically generate new test data. However, SAM and generative models can produce poor quality masks and images that do not match the object x . To ensure the masks and images are correct and of high quality, we resort to human checking.

6.5 Optical Flow Update due to Shape Changes

6.5.1 Upscaling

Given that we have upscaled the objects (x) in frames I_1 and I_2 , it is necessary to rescale the optical flow by reprojection of each point from I_1 to align with the new positions in frame I_2 within the upscaled area $x^\uparrow$.

Since the upscaling is performed based on the center points of the bounding boxes, both the object x and its upscaled counterpart $x^\uparrow$ share the same center point. Let $c_1 = (c_1^x, c_1^y)$ denote the

center point of x in I_1 . We can determine the corresponding center point $c_2 = (c_2^x, c_2^y)$ in I_2 by referencing the ground truth of the optical flow f_{gt} .

Next, for each point $(i_1^\uparrow, j_1^\uparrow)$ in the upscaled object $x^\uparrow$, we need to identify the corresponding point $(i_2^\uparrow, j_2^\uparrow)$ in $I_2^{x^\uparrow}$. The new offset can be calculated by $(i_2^\uparrow, j_2^\uparrow) - (i_1^\uparrow, j_1^\uparrow)$.

We proceed by only considering the first coordinate i for notational clarity. Then, given scale factor θ and center point c_1^x , $i_1^\uparrow$ is found by

$$i_1^\uparrow = c_1^x + \theta(i_1 - c_1^x) \quad (6.11)$$

Thus, we can find the original i_1 in I_1 by

$$i_1 = \frac{i_1^\uparrow - c_1^x}{\theta} + c_1^x \quad (6.12)$$

We can find the same corresponding relation in I_2 by

$$i_2^\uparrow = c_2^x + \theta(i_2 - c_2^x) \quad (6.13)$$

$$i_2 = \frac{i_2^\uparrow - c_2^x}{\theta} + c_2^x \quad (6.14)$$

For each i_1 , we can find the corresponding i_2 with the flow f_{gt} .

$$i_2 = i_1 + f_i \quad (6.15)$$

where f_i is the flow for offset i , in i_1 index. Now, we can calculate the update flow by

$$\begin{aligned} f_i^s &= i_2^\uparrow - i_1^\uparrow = c_2^x + \theta(i_2 - c_2^x) - i_1^\uparrow \\ &= c_2^x + \theta(i_1 + f_i - c_2^x) - i_1^\uparrow \\ &= c_2^x + \theta \left(\frac{i_1^\uparrow - c_1^x}{\theta} + c_1^x + f_i - c_2^x \right) - i_1^\uparrow \\ &= c_2^x + i_1^\uparrow - c_1^x + \theta c_1^x + \theta f_i - \theta c_2^x - i_1^\uparrow \\ &= c_2^x - c_1^x + \theta c_1^x - \theta c_2^x + \theta f_i \\ &= (\theta - 1)(c_1^x - c_2^x) + \theta f_i \end{aligned}$$

by adding back the second coordinate j , with $f_{ij} \in x \uparrow$ arrive at the updated flow:

$$\begin{aligned} f_{ij}^s &= \theta f_{ij} + (\theta - 1)(c_1^x - c_2^x, c_1^y - c_2^y) \\ &= \theta f_{ij} + (\theta - 1)(c_1 - c_2) \end{aligned}$$

Optical flow for shape attack. In Chapter 6.4.1 of the shape attack experiment, we rescale I_1 by θ . Consequently, the optical flow can be updated using the expression $i_2 - i_1^\uparrow$ in $x^\uparrow$.

$$\begin{aligned} i_2 - i_1^\uparrow &= i_1 + f_i - i_1^\uparrow \\ &= \frac{i_1^\uparrow - c_1^x}{\theta} + c_1^x + f_i - i_1^\uparrow \end{aligned}$$

Hence, in two dimensions, we have

$$\begin{aligned} (i_2, j_2) - (i_1^\uparrow, j_1^\uparrow) &= \\ &= \frac{(i_1^\uparrow, j_1^\uparrow) - (c_1^x, c_1^y)}{\theta} + (c_1^x, c_1^y) + f_{i,j} - (i_1^\uparrow, j_1^\uparrow) \end{aligned}$$

6.5.2 General Affine Transformation

In this section, we discuss how the optical flow can be updated for many general motion and, we derive the optical flow update due to a general affine transformation which includes (up)scaling but also rotations, shearing and translations.

- $\Theta_1 \in \mathbb{R}^{3 \times 3}$ and $\Theta_2 \in \mathbb{R}^{3 \times 3}$ are the affine transformation matrices of the object in the frame 1 and 2, respectively.
- $c_1 \in \mathbb{R}^2$ is the center position of the object bounding box in frame 1. $C_1 \in \mathbb{R}^{3 \times 3}$ is the corresponding translation matrix between the origin and c_1 . $C_1 = \begin{bmatrix} 1 & 0 & c_1^x \\ 0 & 1 & c_1^y \\ 0 & 0 & 1 \end{bmatrix}$
- $p_1^t \in \mathbb{R}^2$ represents a pixel belonging to the object in the frame 1. t is the pixel index.
- $\hat{p}_1^t \in \mathbb{R}^2$ represents the object pixel p_1^t after the affine transformation in the frame 1.

- $f_{ij} \in \mathbb{R}^2$ is the optical flow for the pixel p^t from frame 1 to frame 2. $F_{ij} \in \mathbb{R}^{3 \times 3}$ is the corresponding translation matrix from p_1^t to p_2^t . $F_{ij} = \begin{bmatrix} 1 & 0 & f_i \\ 0 & 1 & f_j \\ 0 & 0 & 1 \end{bmatrix}$
- $f_{ij}^s \in \mathbb{R}^2$ is the optical flow for the object pixel p^t after the affine transformation.

In our setup, all the affine transformations are centered in the object bounding box. The goal is to calculate the relationship between f_{ij} and f_{ij}^s . p^t and $\hat{p}^t$ are converted into homogeneous coordinates for the affine transformations. The transformation matrix from $\hat{p}_1^t$ to $\hat{p}_2^t$ can be calculated as:

$$\begin{aligned}\hat{p}_2^t &= C_2 \Theta_2 C_2^{-1} p_2^t \\ p_2^t &= F_{ij} p_1^t \\ p_1^t &= C_1 \Theta_1^{-1} C_1^{-1} \hat{p}_1^t \\ \hat{p}_2^t &= C_2 \Theta_2 C_2^{-1} F_{ij} C_1 \Theta_1^{-1} C_1^{-1} \hat{p}_1^t\end{aligned}$$

Considering a scenario when the affine transformations Θ_1 and Θ_2 for two frames are the same and don't contain any translations. Then, with $\theta \in \mathbb{R}^{2 \times 2}$ being the non-translation matrix and I the identity matrix, the above equations can be simplified without using the homogeneous coordinates as:

$$\begin{aligned}\hat{p}_2^t &= c_2 + \theta(-c_2 + f_{ij} + c_1 + \theta^{-1}(-c_1 + \hat{p}_1^t)) \\ &= c_2 - \theta c_2 + \theta f_{ij} + \theta c_1 - c_1 + \hat{p}_1^t \\ f_{ij}^s &= \hat{p}_2^t - \hat{p}_1^t \\ &= (I - \theta)c_2 + \theta f_{ij} + (\theta - I)c_1 \\ &= \theta f_{ij} + (I - \theta)(c_2 - c_1) \\ &= \theta f_{ij} + (\theta - I)(c_1 - c_2)\end{aligned}$$

6.6 Occlusion Handling

In this section, we will discuss how to handle the occlusion for Flow-R. **Occlusion of Flow-R-T.** The optical flow of Flow-R-T is the same as the original flow. To ensure consistency, we use the ground truth optical flow to warp the new texture from I_2 to I_1 . Thus, all occluded pixels should be the same.

Occlusion of Flow-R-S. To easily handle the occlusion area, we only upscale the objects x . Thus, the optical flow should be recalculated in $x^\uparrow$. To indicate the new occlusion area, we first create an occlusion map $Mask_x$ of x . Then, we use nearest neighbor interpolation to generate the new occlusion map for $x^\uparrow$.

Occlusion of Flow-R-O. We use affine transformation to create motion for the new object from I_1 to I_2 . In practice, we avoid using rotation and moving out of the image, as these could generate complex occlusions. Instead, we only add linear motion, ensuring that each pixel of the new object in I_1 can project to the corresponding pixels in I_2 . Thus, there is no occlusion of optical flow in the new object region. We find that even with these linear motion cases, all methods still fail to predict unseen objects, which raises our concern.

6.7 Mask segmentation

To facilitate the corresponding segmentation, we developed a simple GUI-based application using SAM [75]. This application allows users to segment the same object in pairs of images simply by clicking with the mouse.

Given I_1, I_2, f_{gt} , we can get the segmentation of any instance in I_1 by

$$S_1 = SAM(I_1, Prompt(P1)) \quad (6.16)$$

Where $P1$ is a set of indices obtained by a series of mouse clicks. We can match the corresponding points from $P1$ in I_1 to I_2 using the ground truth optical flow f_{gt} . The mask of the same object in I_2 is

$$P2 = P1 + f_{gt}(P1), \quad (6.17)$$

$$S_2 = SAM(I_2, Prompt(P2)) \quad (6.18)$$

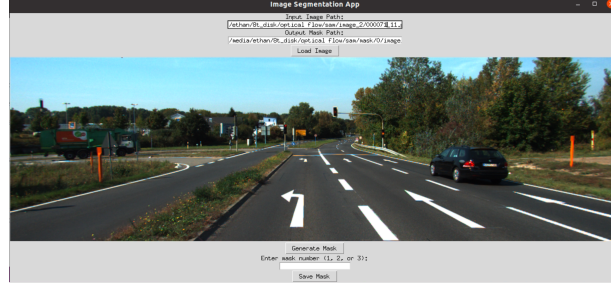
The screenshot of the APP is shown in Figure 6.4.

6.8 Generate Textures with ControlNet

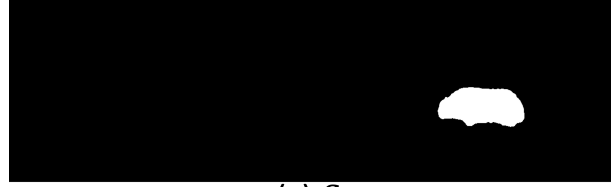
To generate high quality textures with fixed shape, we utilize the Large language model (LLM) [139] to first generate more precise prompt for the mask S_2 . Specially, we feed the S_2 into the GPT4 and ask the following questions:

Q1:Can you recognize which object this mask represents? Please list n of the most likely common objects that fit in the mask.

(a) Screenshot of APP



(b) S_1



(c) S_2



Figure 6.4: **Screenshot of out APP.** (a) Present the GUI of the app. It takes mouse clicks as input, and when you click the "Generate Mask" button, the app will automatically generate a pair of masks as shown in (b) and (c).

Here, n is a number of objects, then we select the most common objects and ask the following question.

Q2: Provide a prompt for ControlNet: [object] with [color/texture] that appears realistic in the real world, ensuring it fits within the provided [mask].

With the output prompt N of and S_2 , the new image can be generated by:

$$Image(N) = ControlNet(N, S_2) \quad (6.19)$$

We use ComfyUI [140, 158] to batch-generate texture images. A brief pipeline is shown in Figure 6.5.

Training Data	Method	KITTI-15		Flow-R-T		Flow-R-S	
		AEPE	Fl-all (%)	AEPE	Fl-all (%)	AEPE	Fl-all (%)
C + T	RAFT [185]	5.04	17.4	5.18	18.3	5.97	24.9
	GMFlow [205]	7.77	23.4	8.67	26.0	8.91	31.0
	GMA [67]	4.69	17.1	4.82	17.9	5.77	24.6
	Flowformer [50]	4.09	14.7	4.28	15.5	5.19	22.3
	Flowformer++ [169]	3.93	14.1	4.30	15.0	5.07	21.8
	SAMFlow [237]	3.44	12.3	3.58	13.0	5.01	20.6
C + T + S/K (+ H)	RAFT [185]	0.63	1.5	0.73	1.66	1.96	10.2
	GMFlow [205]	1.36	5.2	1.55	6.2	2.78	13.9
	GMA [67]	0.57	1.2	0.67	1.4	1.90	10.0
	Flowformer [50]	0.53	1.11	0.64	1.26	1.89	9.9
	Flowformer++ [169]	0.57	1.16	0.71	1.55	1.95	10.1

Table 6.1: **Quantitative results on KITTI 2015 and Flow-R datasets.** We report the average end-point error (AEPE) where not otherwise stated, as well as the Fl-all measure, which is the percentage of optical flow outliers with an error larger than 3 pixels. "C + T" refers to results that are pre-trained on the Chairs and Things datasets. "S/K (+ H)" refers to methods that are fine-tuned on the Sintel and KITTI datasets, with some also fine-tuned on the HD1K dataset. Flow-R-T donate the data we replace the texture of main object from KITTI 2015. Flow-R-S donate the data we replace the shape of main object from KITTI 2015.



Figure 6.6: **A failed example.** We add seen and unseen objects to a similar area. The first row indicates that all methods successfully predict the motion of the added seen (car) objects. The second row indicates that all SOTA methods failed to predict the unseen (fox) objects.

Attack	Texture		Shape	
	AEPE	F1-all	AEPE	F1-all
RAFT [185]	0.82	1.99	2.21	10.5
GMFlow [205]	1.63	6.5	3.03	14.2
GMA [67]	0.72	1.63	2.09	10.5
Flowformer [50]	0.64	1.26	2.04	10.2
Flowformer++ [169]	0.76	1.66	2.13	10.1

Table 6.2: **Texture attack and Shape attack.** Quantitative results of the attacked first frame and original second frame.

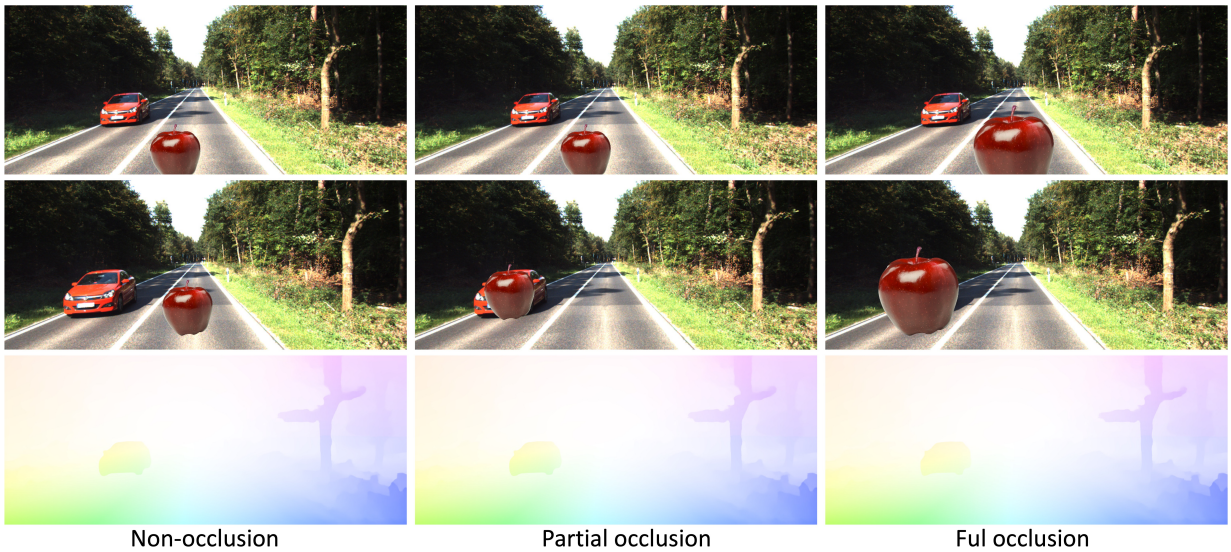


Figure 6.7: **Failed example with different positions.** Each column indicates the object car being non-occluded, partially occluded, or fully occluded by the unseen object (apple). The predicted flow from FlowFormer++ shows similar results; all flows ignore the unseen objects. The original motion of the car is still predicted even when the car is fully covered in the third column.

6.9 Experiment

In this section, we describe the experiments that evaluate the behaviors of state-of-the-art (SOTA) optical flow approaches by our proposed method. We focus primarily on the common behaviors

Methods	Add Seen			Add Unseen				
	all		fore	background	all		fore	background
	AEPE	F1-all	AEPE	AEPE	AEPE	F1-all	AEPE	AEPE
RAFT [185]	2.54	8.2	11.26	0.77	7.43	20.4	25.68	7.18
GMFlow [205]	3.24	12.1	11.32	1.64	9.42	22.6	23.73	9.03
GMA [67]	2.55	8.1	10.89	0.81	7.44	20.2	22.62	7.19
FlowFormer [50]	2.45	7.7	10.51	0.77	7.81	21.0	23.70	7.52
FlowFormer++ [169]	2.55	8.1	10.71	0.83	7.50	20.4	20.53	7.26
SAMFlow [237]	2.43	7.9	10.76	0.85	7.73	20.1	20.57	7.33

Table 6.3: **Adding new objects.** Quantitative results of adding new objects to original images. “Add Seen” indicates the data where we add KITTI objects to the images. “Add Unseen” refers to the data where we add real-world objects not included in KITTI. Foreground indicates the EPE errors in the area of the new object. Background indicates the EPE error that do not include the new object.

and failure cases of methods. These failure cases can potentially guide future studies of optical flow methods.

We mainly use black-box evaluation to explore two important questions: *Texture vs. Shape*: What do optical flow methods rely on for matching? *Memorization*: Do optical flow methods tend to remember objects?

We select following methods in this experiment: RAFT [185], GMA [67], GMFlow [205], FlowFormer [50], FlowFormer++ [169] and SAMFlow [237].

6.9.1 Texture vs. Shape

Understanding how networks match objects is an important problem. It may provide fundamental insights into the functioning and potential failure points of optical flow methods. However, to the best of our knowledge, no studies have explored this aspect. In this section, we conduct three experiments to investigate whether the selected optical flow methods rely on texture or shape.

To investigate the dependency of optical flow methods on texture versus shape, we evaluated their performance using KITTI [34], Flow-R-T, and Flow-R-S datasets as shown in Table 6.1. The findings reveal that most methods yield comparable results on KITTI [34] and Flow-R-T datasets. Conversely, all methods demonstrate a notable increase in errors on the Flow-R-S

dataset, suggesting a higher sensitivity of optical flow methods to shape variations compared to texture variations.

Texture Attack. Given that texture replacement does not significantly impact the performance of SOTA methods, we conduct an additional experiment by altering the texture of only one image. This experiment aims to force the network to match objects with different textures. Given data from the KITTI dataset (I_1, I_2, f_{gt}) , we replace the texture of object x from I_2 by Equation (6.3). Consequently, the texture of object x in I_1 and I_2^N differed, as shown in Figure 6.3. Despite this difference, the object is still matched, and the optical flow is accurately predicted in the result.

Table 6.2 presents the results of the Texture Attack experiment. Surprisingly, we observe that the error rate for texture attack of attention-based methods is relatively small compared with errors in KITTI, suggesting that the matching of objects in optical flow methods is less dependent on texture than shape.

Shape Attack. Shape changes are a common challenge in optical flow estimation, particularly when objects move closer to or further away from the camera, resulting in different shapes between pairs of images.

To assess the robustness of models to shape changes, we design a shape attack experiment, where we scale the object in only one image, forcing the methods to match objects of different sizes.

Specifically, we scale the first frame I_1 by θ (as described by Equation (6.7)), and recalculate the optical flow. Table 6.2 presents the quantitative evaluations of state-of-the-art methods. We observe a significant increase in errors between the shape attack results and the original results, indicating that the matching of objects in optical flow methods is sensitive to shape changes.

6.9.2 Does the Network Remember Objects?

Optical flow methods still suffer from poor cross-dataset generalization. Most state-of-the-art (SOTA) methods require fine-tuning and testing on the same dataset. One hypothesis is that optical flow methods tend to memorize objects during training and attempt to find similar objects during testing.

In this section, we conduct experiments to explore whether this hypothesis holds true. We define an unseen object as one that the methods do not see during training, while a seen object is one that is included in the KITTI dataset. Assuming optical flow methods successfully predict the correct optical flow given data (I_1, I_2, f_{gt}) , we use Equation (6.9) and Equation (6.10) to add unseen objects from real-world images, obtaining $(I_1^{unseen}, I_2^{unseen}, f_{gt}^{unseen})$. We then add seen objects from the KITTI dataset in another scene to obtain $(I_1^{seen}, I_2^{seen}, f_{gt}^{seen})$. By comparing the

results with the original data, data with unseen objects, and data with seen objects in Table 6.3, we find that the accuracy of the flow prediction for unseen objects is significantly worse than for seen objects. This indicates that unseen objects negatively impact the methods, leading to poorer results.

Furthermore, Table 6.3 reports the error in the foreground (EPE error only in the new object area) and the background error (areas not including the new object). We find that adding unseen objects also affects the background accuracy, even if the unseen object is not adjacent to the background area.

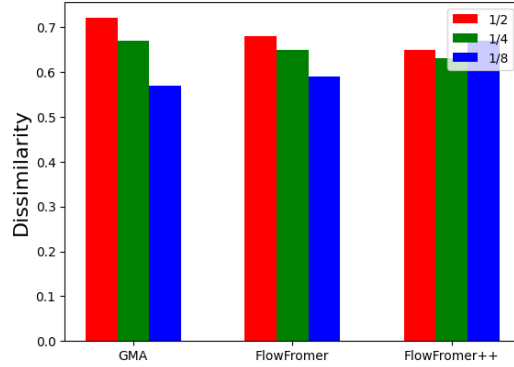


Figure 6.8: **Feature Dissimilarity.** All levels in the encoder have high dissimilarity compared with the original features for the added object region, indicating the encoder retains the new object.

6.9.3 Failures for Unseen Objects

To further study why unseen objects lead to poor results, we visualize the results in Table 6.3. We find that SOTA methods successfully predict the flow for seen objects, i.e., cars, but usually predict worse or even ignore unseen objects in the results. For example, in Figure 6.6, we add a car (first row) and a fox (second row) in similar positions. All methods successfully predict the motion of the car, but all methods fail to predict the motion of the fox. This indicates that SOTA methods tend to reduce or even ignore the flow of unseen objects.

Unseen object attack.

To study how an unseen object x_u affects objects x in original images, we conduct experiments by placing x_u in different positions to observe its impact on the motion of x in the images.

Specifically, we categorize the scenarios into three cases: unoccluded, where x_u does not cover x ; partly occluded, where x_u partially covers x ; and fully occluded, where x_u completely covers x in the target image.

Figure 6.7 presents an example using FlowFormer++ [169]. In the first column, the apple does not interact with the car. In second column, the apple partially covers the car. In the third column, the apple fully covers the car. However, the flow results for all three scenarios ignore the motion of the apple and successfully predict the car and its flow, even when it is covered by the apple. In the third example, where the car is fully covered by the apple, the cost volume should indicate that the car in the first frame cannot be matched to the second image due to occlusion. However, the method still predicts the car’s motion, similar to the original results. This indicates that the methods tend to remember the motion of the scene or object and produce similar results as output.

6.10 A Study of Unseen Objects

To investigate why certain methods ignore unseen objects, we collect all the failure cases where the object is ignored and conducted a visualization focusing on the encoder and decoder components of GMA [67], FlowFormer [114], and FlowFormer++ [169]. As GMA is primarily based on a CNN architecture, FlowFormer on a transformer architecture and FlowFormer++ incorporates a Masked AutoEncoder [44] pretraining strategy, it allows us to investigate whether these designs contribute to robustness.

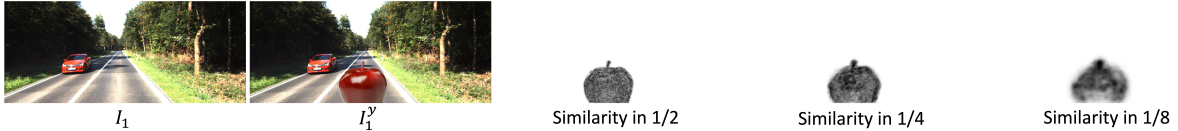


Figure 6.9: **Example of Feature Dissimilarity.** A clear difference of encoder features with or without new objects is shown at each level. The encoder retains the information related to new objects.

6.10.1 Feature Visualization in the Encoder

In the encoder, we extract the contextual features from the original image I_1 and the image containing the unseen object I_1^y . For each feature in level (1/2, 1/4, and 1/8), we treat each pixel



Figure 6.10: **Visualization of Intermediate Flows in the Decoder.** Decoder GRU iterations in original and failed samples.

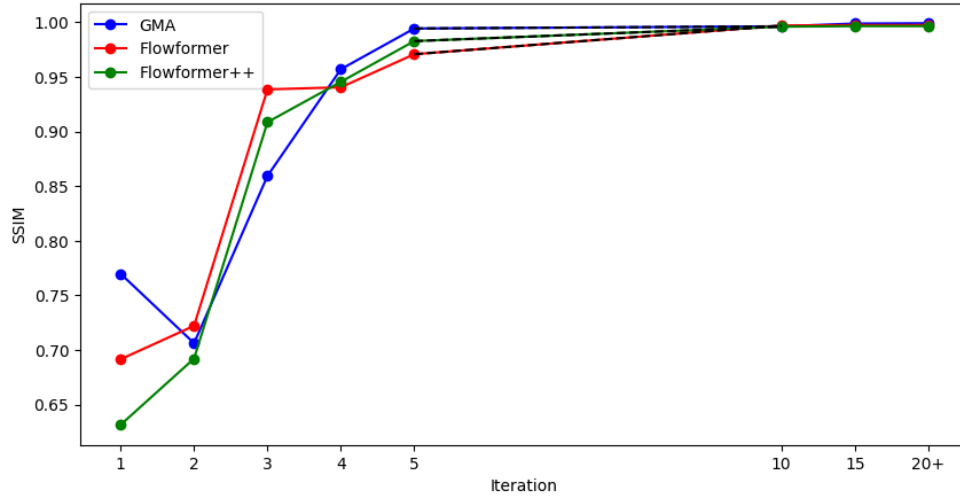


Figure 6.11: SSIM of intermediate flow in each iteration between failed and original samples.

as a vector and perform cross-similarity calculations between the original feature and unseen object feature to generate a similarity map, as illustrated in Figure 6.9. Where 1/2, 1/4 and 1/8 refer to the levels of features corresponding to the image resolution. The results reveal a clear difference in the areas corresponding to the new object, indicating that the encoder successfully extracts features related to the added object at all levels rather than ignoring it.

In Figure 6.8, we statistically analyze the dissimilarity of features across the various methods. We measure the dissimilarity by calculating the similarity score, normalize the score to a range of 0 to 1 and then inverse it. A larger value indicates greater dissimilarity between the two feature sets. We only evaluate the dissimilarity within the new object region. We found that at all three levels (1/2, 1/4, and 1/8), the features exhibited a relatively high dissimilarity, suggesting that the

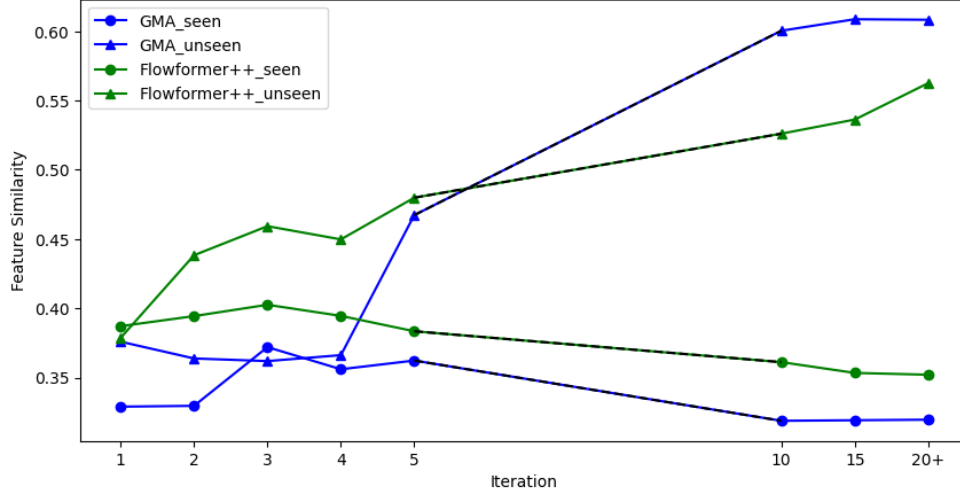


Figure 6.12: **Feature similarity of GRU block.** A higher value indicates that the features of the image with the added object are similar to those of the original image.

encoder effectively retains information about the unseen object. The feature difference between the original image and the image with the apple using cross-similarity in GMA is visualized in Figure 6.9. It clearly indicates the dissimilarity in the apple region.

6.10.2 Feature Visualization in the Decoder

Recent methods employ an iterative refinement architecture. We chose GMA as a representative for such a framework and visualize each iterative step of the optical flow decoder. As illustrated in Figure 6.10, we initially observe the motion of an unseen object (e.g., an apple) during the coarse optical flow estimation. This object is subsequently treated as noise and removed in later iterations.

We utilize SSIM to quantitatively assess the structural similarity between the optical flow of the original input images I_1, I_2 and the images containing the unseen object I_1^y, I_2^y . The evaluation is only in the regions of new object. As depicted in Figure 6.11, we observe significant differences in SSIM values during the first four iterations, which converge to a much closer similarity after ten iterations. As illustrated in Figure 6.10, the first row displays the initial five iterations of optical flow applied to the original image input, while the second row presents the first five iterations for an image with unseen object (an apple). The apple is distinctly visible in

the first iteration, but decreases with each subsequent iteration, and becoming nearly imperceptible by the fourth iteration. This observation suggests that the decoder component may accidentally interpret the unseen object as noise or error, leading to its gradual removal throughout the iterations.

Next, we conduct experiments to analyze each main component of the decoder, including the cost memory, attention block, GRU block, and upsampler. By comparing the features extracted from I_1 , I_2 and I_1^y , I_2^y , we observe that the GRU block exhibits behavior similar to that observed in the flow iterations. As illustrated in Figure 6.12, We report the feature similarity between the original features and the features with new objects within the GRU block. Circles indicate images with seen objects, while triangles denote images with unseen objects. In both GMA and Flowformer++, the GRU block effectively retains the features of seen objects across iterations, while it weakens the features of unseen objects. This suggests that the GRU block is designed to remember and preserve the characteristics of seen objects, while attempting to eliminate the influence of unseen objects during the iterations.

6.11 Summary

In this chapter, we propose a robustness evaluation method by manipulating the shape and texture of original objects or by adding new objects. We generate a dataset and conduct a comparative evaluation of the robustness of recent state-of-the-art (SOTA) methods. Our findings reveal two notable common behaviors among these SOTA methods: 1. The matching performance of optical flow methods relies more on shape than on texture. 2. The GRU block tends to remember seen objects while attempting to diminish the influence of unseen objects. Given these observations, we express concerns regarding the safe usage of current optical flow methods. We hope that our methodology and findings will guide further studies aimed at enhancing the robustness of optical flow techniques in safety-critical scenarios.

Chapter 7

Learning to Handle Large Obstructions in Video Frame Interpolation

7.1 Introduction

In previous chapters, we have discussed the existing limitations of optical flow estimation. In this chapter, we focus on how optical flow affects the video frame interpolation task. We focus on real-world applications of video frame interpolation based on optical flow estimation. We observe that large obstructions can lead to significant occlusion, which produces errors in optical flow. The optical flow warps the objects in consecutive frames to incorrect positions and causes large errors and artifacts in synthetic flow. Thus, we address this issue in this chapter.

This chapter resulted in a publication [106]¹.

7.2 Overview

Video frame interpolation (VFI) is to synthesize intermediate images from given image sequences. Related algorithms are widely implemented in real-world applications, such as video super-resolution [72, 202], slow motion generation [7, 65, 135], novel view synthesis [69, 238] and video compression [111].

¹© 2024 Association for Computing Machinery. Reprinted, with permission, from Libo Long, Xiao Hu, and Jochen Lang, "Learning to Handle Large Obstructions in Video Frame Interpolation," in *ACM MM*, 2024. DOI: [10.1145/3664647.3681006](https://doi.org/10.1145/3664647.3681006).

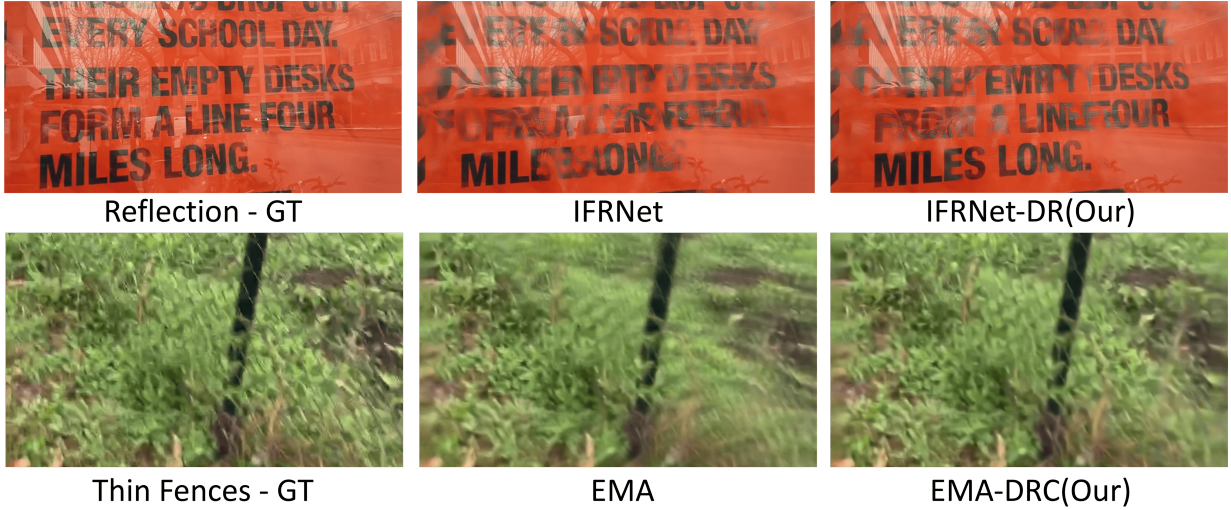


Figure 7.1: Predictions in real-world obstruction scenes, standard approaches fail due to occlusions and motion error, our method makes the models robust in those conditions.

Deep learning-based methods have achieved remarkable results on VFI benchmarks. Previous methods utilize kernels [22, 25, 104] or optical flow [77, 112, 134, 143, 144, 223] to align each object with its location in the neighboring frames.

Progress so far mainly focuses on interpolation quality [112, 143] and interpolating efficiency [48, 77]. Most existing video frame interpolation methods assume that scene objects are in clear view without any external interference. However, in many real-world applications, complex obstructions are often present due to various factors. For example, video captured when driving, during sports and in outdoor settings may involve obstructions caused by fences, rain-drops or reflection. Obstructions can cause large blurred areas and artifacts in the interpolated frames degrading the visual quality and temporal consistency of the output [7, 134]. As a result, applying previous VFI methods directly on videos with obstructions leads to subpar performance. Calculating the motion field in a video with obstructions is difficult, as pixels of an object often lack matches between two input frames.

Although previous methods use depth information [7] or bi-directional optical flow [23, 145] to explore the occlusion information. However, those methods still generate blur and artifacts in scenes with large occlusion caused by obstructions (see examples in Figure 7.1). This shows that mere awareness of occlusions is insufficient for reconstructing large occlusions caused by obstructions. A more explicit approach is needed to identify blurred regions within the features

and subsequently recover them.

To achieve this goal, we propose three techniques to handle videos with obstructions. First, we propose a novel data augmentation method called Dynamic Mask Mixing (DMM), which synthesizes dynamic obstructions and mixes it into clean videos. We want occlusions such as fences to move with the camera. We use DMM to generate new data during training. By using our augmentation, VFI networks can be robust on both videos with and without obstructions without any new dataset.

Second, to indicate possible ambiguous pixels/region which could cause blurred, we propose a lightweight module called Region Similarity Map (RSM). RSM determines pixel similarity in warped adjacent input frames. We observe that most of warped input frames contain a large proportion of similar pixels which can be easily estimated by VFI methods. Therefore, we want the model to focus on ambiguous pixels in dissimilar regions. We demonstrate that our RSM is effective in capturing ambiguous pixels in challenging scenes that contain large occlusions.

Third, to repair large occlusion error caused by obstructions, we propose a cross-overlap attention (COA) mechanism. COA calculates the appearance feature by matching the appearance information of objects between input frames and the synthesized frame. Then we use the appearance feature to fix in the ambiguous region of the synthesized frame previously identified by RSM.

To prove the versatility and adaptability of our RSM and COA module, we apply it to various state-of-the-art VFI models. We then construct a special test set called Real World Obstructions (RWO) dataset which contains various real-world obstructions close to the camera. The experiments shows our approach significantly improves the result compared to the other methods on the RWO dataset.

Our main contributions are:

1. **New Data Augmentation Strategy.** We propose a new data augmentation strategy called **DMM** which generates obstructions based on a dynamic binary mask. By applying DMM during training, models can handle videos with or without obstruction.
2. **Novel Module.** We propose a novel repair feature module which contains **RSM** and **COA**. RSM first marks possible blur regions in features, then COA fixes the region by aggregating appearance information of objects.
3. **Performance.** By integrating our modules with previous methods, we demonstrate that our method achieves significant performance improvement on our RWO dataset and attains state-of-the-art results on multiple large occlusion benchmarks.

7.3 Related Work

Existing VFI methods can be classified into three categories: kernel-based, phase-based and flow-based approaches.

Kernel-based methods regard motion estimation as being joint with motion compensation. AdaConv [135] proposes estimating a pair of spatially-adaptive convolution kernels using a CNN and predicting intermediate flow by convolving input frames with the proposed kernels. A follow-up work [136] uses adaptive separable convolutions to reduce the large computational memory demand. Other alternatives improve it by estimating extra kernel offset vectors [22, 85], integrating deformable convolution to increase the receptive field [22], and proposing a multi-scale tailored loss function [147]. However, they are generally computationally expensive and inefficient in dealing with occlusion.

Earlier phase-based approaches (e.g. [129]) use phase information to learn the motion relationship, and they interpolate phase across the levels of a multi-scale pyramid. This approach works well with small motions but fails to handle challenging interpolation. PhaseNet [128] proposes a neural network that directly estimates the phase decomposition of the intermediate frame.

Previous flow-based works improve performance of occlusion scenes by various techniques including by estimating more robust optical flow for warping pixels [51, 65, 145, 208] or by determining occlusions with a depth map [7]. Recently, flow-based approaches have made good progress. CIAN [23] proposes to use channel attention and PixelShuffle to replace the extra flow estimation network. RIFE [51] proposes a sub-network IFNet that support arbitrary-timestep frame interpolation with the temporal input. FILM [154] proposes a multi-scale feature extractor that shares weights at all scales for large motion videos. IFRNet [77] proposes a lightweight encoder-decoder based network architecture for real-time applications. VFIfomer [112] is a transformer-based U-net framework which enlarges the receptive field for capturing long-range information. Yoo et al. [217] proposed to use video object segmentation as an auxiliary task to train VFI models, which can learn extra segmentation information to interpolate frames with more precise object boundaries. Kim et al. [73] proposed to use event voxel grids as an additional input modality to capture the changes in pixel intensity asynchronously and with high temporal resolution, which can handle large and fast motions. Lee et al. [86] proposed a subnet to estimate D-map that handles discontinuity in frames. EMA [223] proposed a transformer block that extracts motion and appearance information separately to better capture global motions compared with mixed feature map. BiFormer [143] proposed a transformer based architecture including blockwise bilateral cost volumes, local motion module and global motion refinement that handles 4K video frame interpolation. However, those methods still face the challenge of dealing

with videos with large obstructions. In this thesis, we mainly focus on handling close camera obstruction for real world frame interpolation.

7.4 Method

7.4.1 Overview

VFI is a technique to generate intermediate frames between two consecutive video frames $I_0, I_1 \in \mathbb{R}^{H \times W \times 3}$, with H and W being the height and width, respectively. The general framework of flow-based VFI usually contains two steps. First, the network calculates intermediate optical flow $O_{t \rightarrow 0}$ and $O_{t \rightarrow 1}$ ($t=0.5$ in practice). Second, it predicts a residual δI_t and an image mask H based on intermediate optical flow and input frames. The intermediate frame I_t is estimated as:

$$\tilde{I}_0, \tilde{I}_1 = \text{warp}(I_0, O_{t \rightarrow 0}), \text{warp}(I_1, O_{t \rightarrow 1}) \quad (7.1)$$

$$\hat{I}_t = \tilde{I}_0 \times H + \tilde{I}_1 \times (1 - H) + \delta I_t. \quad (7.2)$$

Most of the existing VFI networks [70,77,145,223] are designed to deal with obstruction-free scenes. Our main goal is to develop an efficient and robust method for both obstruction-free and obstruction scenes. There are three main challenges we need to address. 1) Methods are mainly trained on obstruction-free scenes in datasets such as Vimeo90K [208], and they could experience a domain shift when they encounter scenes with obstructions. To solve this problem without using any additional datasets, we propose a data augmentation method called DMM, which is introduced in Chapter 7.4.2. 2) Most methods rely on motion information, such as optical flow, to estimate the intermediate frames. However, when occlusion occurs in the frames, optical flow may fail to match and estimate motion correctly. Thus, we propose a Region Similarity Map (RSM) to find error regions caused by mismatching in Chapter 7.4.3. We show that RSM can clearly indicate error regions that optical flow fails to handle correctly. 3) To better reconstruct the information in error regions, we propose a COA module, which captures the appearance information and fills in the error regions in Chapter 7.4.4.

7.4.2 Dynamic Mask Mixing

Most of previous studies applied flip augmentation along spatial and temporal axes. However, this general augmentation is not sufficient to handle videos with obstructions. We propose a new data augmentation method for frame interpolation, called dynamic mask mixing (DMM).

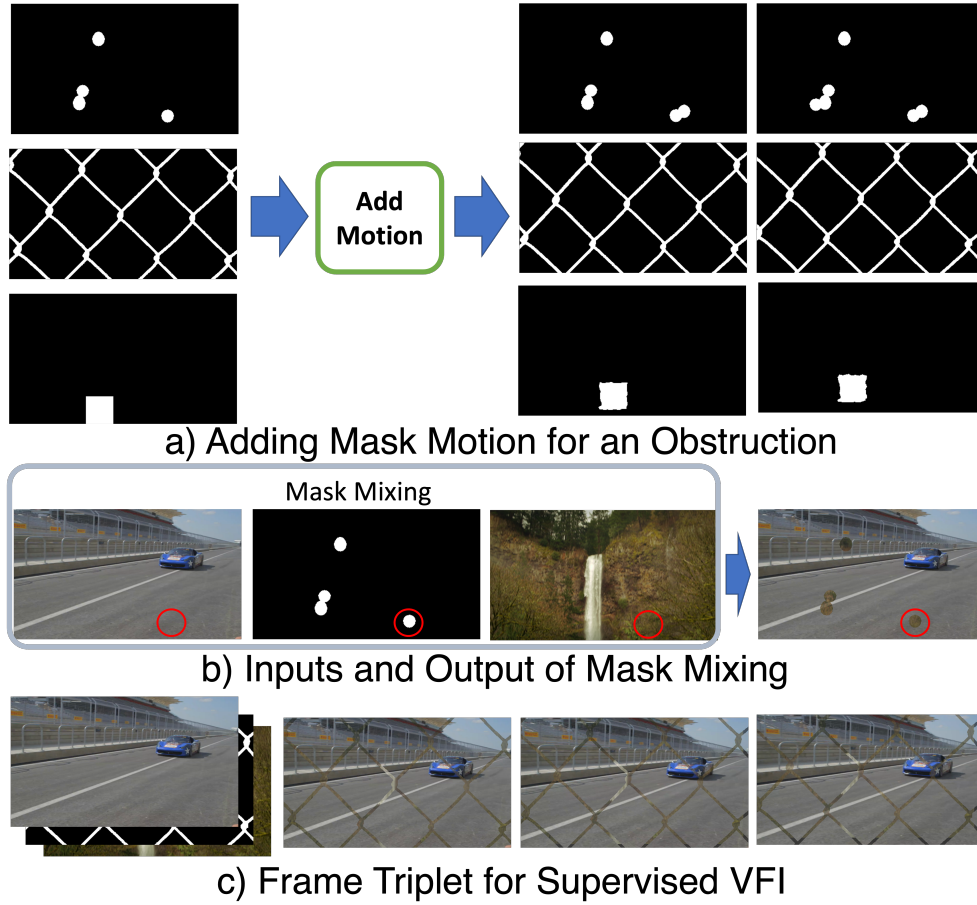


Figure 7.2: Schematic View of our Dynamic Mask Mixing (DMM) Data Augmentation. We first create a binary mask and add motion to it. Then, we fuse the data with another image using the binary mask to generate new training data.

The main idea of DMM is to synthesize obstructions on original video. It has been shown that synthesizing obstructions such as reflections and fences can make the network adapt to real-world scenes with obstructions [102, 170]. However, most of them synthesize obstructions in a single image. In a video, obstructions usually move. The motion may be continuous or discontinuous depending on the physical properties of the obstruction and the imaging setup. Thus, we propose a simple but effective way to simulate the motion of obstructions.

Adding Motion. We simulate three categories of motions for obstructions: translation, de-

formation, and discontinuous motion. Translation refers to the movement of rigid obstructions in a random direction, e.g., motion of fences close to the camera. Deformation occurs when non-rigid obstructions change their shape, e.g., dirt on a window or the lens itself. Discontinuous motion describes the sudden appearance or disappearance of obstructions, e.g., from raindrops on a lens, or due to specular reflection.

As shown in Figure 7.2 a, we first generate a binary mask in the shape of the obstruction. Then we generate masks for the other frames by applying different motions to the first mask. For translation, we randomly add an offset to the position of the obstruction and move the mask for the second and third images. For deformation, we use elastic deformation [171], which modifies the shape of the obstruction by applying random displacements to a grid of pixels. The mask is then interpolated to fit the deformed grid, creating a distorted version of the obstruction. For discontinuous motion, we generate a random number of obstructions of a certain shape, and then randomly add or delete obstructions in the mask for the second and third image consecutively.

Obstruction Mixing. Let (I_{t-1}, I_t, I_{t+1}) be a triple of subsequent images from a video. Then the obstructed version of this triple is $(D(I_{t-1}, M_{t-1}, \tilde{I}_d), D(I_t, M_t, \tilde{I}_d), D(I_{t+1}, M_{t+1}, \tilde{I}_d))$. Where M is the corresponding obstruction mask, $\tilde{I}_d$ is a frame from a different scene, and $D()$ is the mixing function. In practice, we calculate $D(I_t, M_t, \tilde{I}_d)$ by $I_t * \lambda + M_t * \tilde{I}_d * (1 - \lambda)$, with mixing ratio $\lambda \in (0, 1)$ (Figure 7.2 b). Figure 7.2 c is an example of training data.

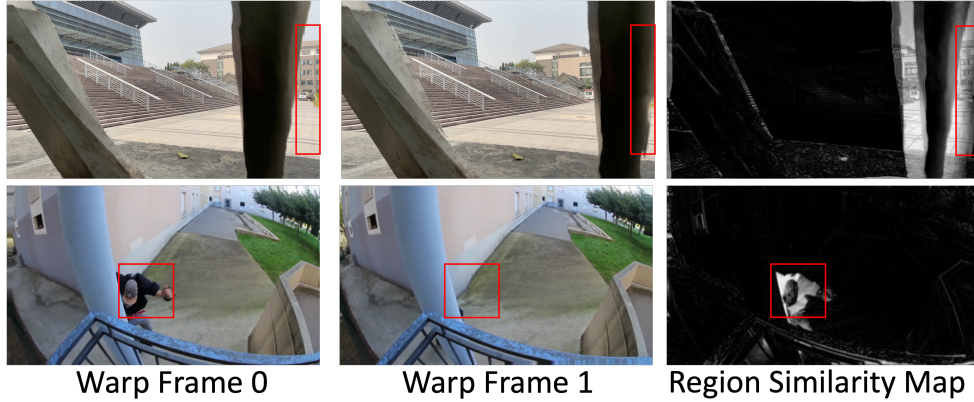


Figure 7.3: Two examples of occlusions in a scene with obstructions. The first and second columns show the warped frames generated by the VFI network. The pixels in the red rectangle fail to warp to the same position. Our Region Similarity Map clearly indicates the error.

7.4.3 Region Similarity Map

Figure 7.3 shows two examples of warped frames with obstructions. The first and second columns show the warped input frames $\tilde{I}_0$ and $\tilde{I}_1$, respectively. The red rectangles highlight the regions where $\tilde{I}_0$ and $\tilde{I}_1$ should be identical or similar. However, we observed that those regions are not correctly aligned by optical flow which could cause large artifacts and distortions in the intermediate frame. The third column shows our proposed Region Similarity Map (RSM), which successfully measures the error regions between the warped frames. We define RSM C as a 1-channel map, where the value of each pixel is in $(0, 1)$. Higher value indicates greater similarity between the corresponding pixels, and vice versa. In practice, we want to keep unambiguous pixels in warped frames, and only repair ambiguous pixels $1 - C$ by our method.

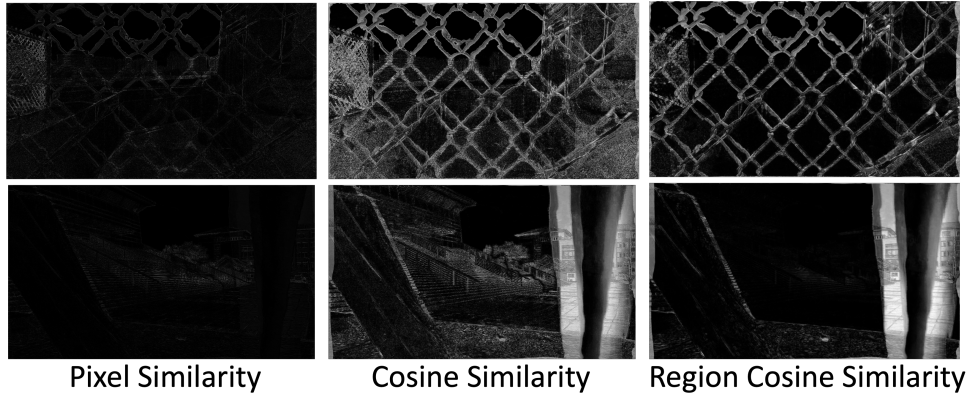


Figure 7.4: A comparison of different similarity methods for scenes with obstructions. The region cosine similarity method is most stable.

To find the best measurement of RSM, we conducted experiments with three different methods: pixel similarity, cosine similarity and region cosine similarity. We extract feature maps $F_0 \in \mathbb{R}^{W \times H \times c}$ and $F_1 \in \mathbb{R}^{W \times H \times c}$ from I_0 and I_1 , as shown in Figure 7.4, and warp F_0 and F_1 to $\tilde{F}_0$ and $\tilde{F}_1$ by using optical flow $O_{t \rightarrow 0}$ and $O_{t \rightarrow 1}$. We use $\tilde{F}_0$ and $\tilde{F}_1$ to calculate the similarity map. The first column indicates the pixel similarity, which is computed by simply taking the pixel-wise difference between $\tilde{F}_0$ and $\tilde{F}_1$. Then averaging along the channel dimension to obtain a 1-channel map. It fails to highlight the most dissimilar regions. The second column is the cosine similarity. We treat the channels of each pixel as a vector $v \in 1 \times c$ and calculate the cosine similarity between each pair of corresponding vectors. Cosine similarity emphasizes the regions with large differences, but also introduces small scale noise which may mislead the network to focus on unimportant areas. Finally, we propose region cosine similarity. For each

pixel in $\tilde{F}_0$, we calculate the similarity with its k by k neighbours and the corresponding region in $\tilde{F}_1$. The map only focuses on regions with large differences and avoids to highlight pixels with small motion. More visualizations of RSM can be found in the supplementary material.

7.4.4 Feature Repair

A naive way to reduce the artifacts and distortions in dissimilar regions is to apply a lightweight residual CNN-based module and hope that those errors at the feature level can be repaired by local information. Let x be the intermediate feature with artifacts. The recovered $\hat{x}$ can be written as.

$$\hat{x} = x \times C + \text{conv}(x) \times (1 - C) \quad (7.3)$$

However, this method may not work well when dealing with large obstructions, as local features may not provide sufficient information. Therefore, we propose to use a transformer module that can model long-range dependencies and learn global context.

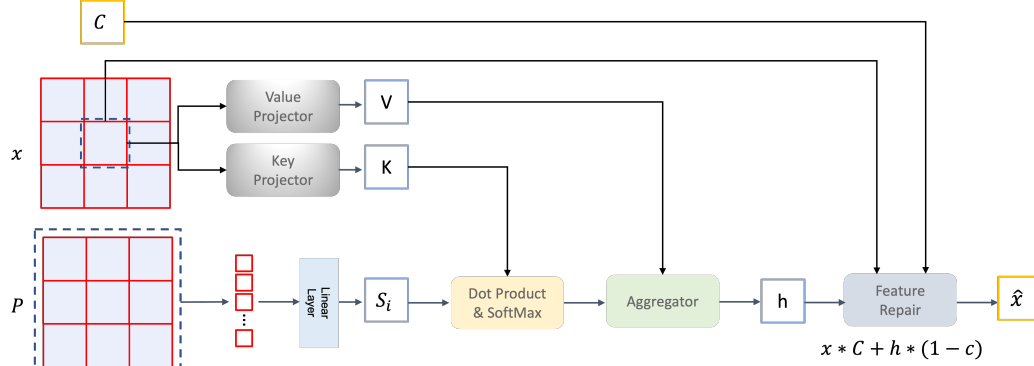


Figure 7.5: The details of our COA: Each patch and its 8-neighborhood in the appearance feature map P are projected into one token. Attention is computed with the key K from the inter-frame feature x . The correspondence will match if the patch of x is similar to the patch of f and its neighbors. The hidden features are calculated by the dot product of attention and values V , and repairs the inter-frame features in the explicit region indicated by the Region Similarity Map $1 - C$.

Cross Overlap Attention. To find good inter-frame features for artifacts and blurred regions, our core idea is to allow the network to globally match the same objects across frames.

As humans, we can easily recover a blurred object with a clear reference image that contains the same non-blurred object. We hypothesize that by comparing non-blurred objects from the original input images and blurred objects in synthetic features, the network can extract clearer appearance and geometrical information for blurred pixels. Based on this idea, we adopt a transformer block [103] that can model long-range dependencies. Unlike self-attention, our query features are projections from appearance features, while our value and key features are projections from inter-frame features. The attention matrix computed from the query and key features is most likely to match the same object, then aggregate the appearance information to the blurred pixels of value features. To reduce the mismatch of the attention map, we project each patch q and its 8-connected neighbors in Query Q into one token, and then compute the attention with the corresponding patch k in Key K . This implies that if patch q and its neighbors are similar to k , they are more likely to belong to the same object. **Formulation of Cross Overlap Atten-**

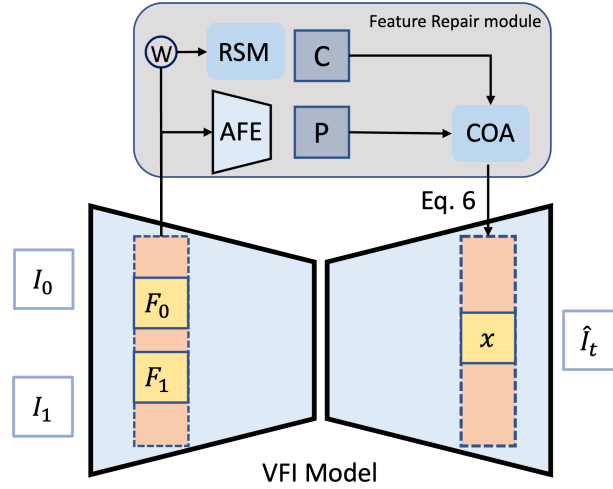


Figure 7.6: The architecture of our model, which uses any optical flow based network as the VFI model. We take the input feature map from the encoder and use the feature repair module to fix the feature in the decoder. W indicates the warp operation.

tion. As show in Figure 7.5, given inter-frame features x and appearance features P of the same spatial resolution. We use P to extract query vectors $Q_f \in \mathbb{R}^{N_q \times d}$, and x to extract key vectors $K \in \mathbb{R}^{N_k \times d}$ and value vectors $V \in \mathbb{R}^{N_v \times d}$, where d is the feature dimension.

For each patch in Q_f , we aggregate its 8-connected neighbors patches into a tensor $S \in \mathbb{R}^{N_q \times 9 \times d}$. We then reshape and linearly project S into $S_i \in \mathbb{R}^{N_q \times d}$. Finally, we compute the

similarity between S_i and K . The output $\tilde{h}$ can be formulated as

$$\begin{aligned}\tilde{h} &= \text{Attention}(L^S(S), K, V) \\ &= \text{Softmax}\left(\frac{L^S(S)K^T}{\sqrt{d}} + B\right) \cdot V\end{aligned}\quad (7.4)$$

where B is the position bias [29], $L^S(\cdot)$ is the linear projection function. We incorporate cross overlap attention into the transformer block. Following [103], we also apply normalization (LN) and a multi-layer perceptron (MLP) after attention as

$$h = COA(x, P) = MLP(LN(\tilde{h})).\quad (7.5)$$

Feature repair with RSM. Figure 7.6 illustrates our proposed method for repairing the intermediate feature with the RSM and the COA module. Given a baseline VFI network N , it extracts the features F_0 and F_1 from the encoder. We warp them to $\tilde{F}_0$ and $\tilde{F}_1$ and compute the RSM from the warped features. We also use a lightweight CNN-based module to estimate the appearance feature P capturing the local color and texture information of the scene. Then, we apply the COA module to refine the intermediate feature by using Equation (7.5). We use the COA module to refine the features in dissimilar regions $(1 - C)$. The refined features $\hat{x}$ are calculated as.

$$\hat{x} = x \times C + h \times (1 - C)\quad (7.6)$$

We choose the optimal layer for each network based on the performance of the RSM and the COA module. The details of our layer selection are provided in the supplementary material.

7.5 Experiments

7.5.1 Datasets and Implementation Details

Training and Evaluation. Our model is trained on the Vimeo90K [208] training set. For hyperparameters, such as learning rates, batch sizes, and optimizers, we follow the settings reported in the original papers of each baseline network. For the data augmentation, we apply random cropping and flipping to the input frames along the horizontal, vertical, and temporal dimensions. We also incorporate the proposed data augmentation DMM, which has been described in Section 7.4.2.



Figure 7.7: Qualitative results on SNU test dataset.

We evaluate our method on four recent state-of-the-art flow-based VFI methods: RIFE [51], IFRNet [77], VFIfomer [112], and EMA [223]. To demonstrate the effectiveness of each component of our method, We first conducted an ablation study on DMM and RSM, which only increased the parameters by less than 1% of the original method. Then, we incorporate all three modules DMM, RSM and COA into VFIfomer and EMA. We use three benchmark datasets for the evaluation: Vimeo90K , which is the most common test set for VFI; SUN-Film, which contains large motion and more occlusions in the hard and extreme data; and our real-world obstruction dataset, which is a new dataset that we construct from real-world obstructions. We collect some of the data ourselves and adapt some from [239]. More detail can be found in supplementary material. We measure the performance of each algorithm by comparing Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) [197].

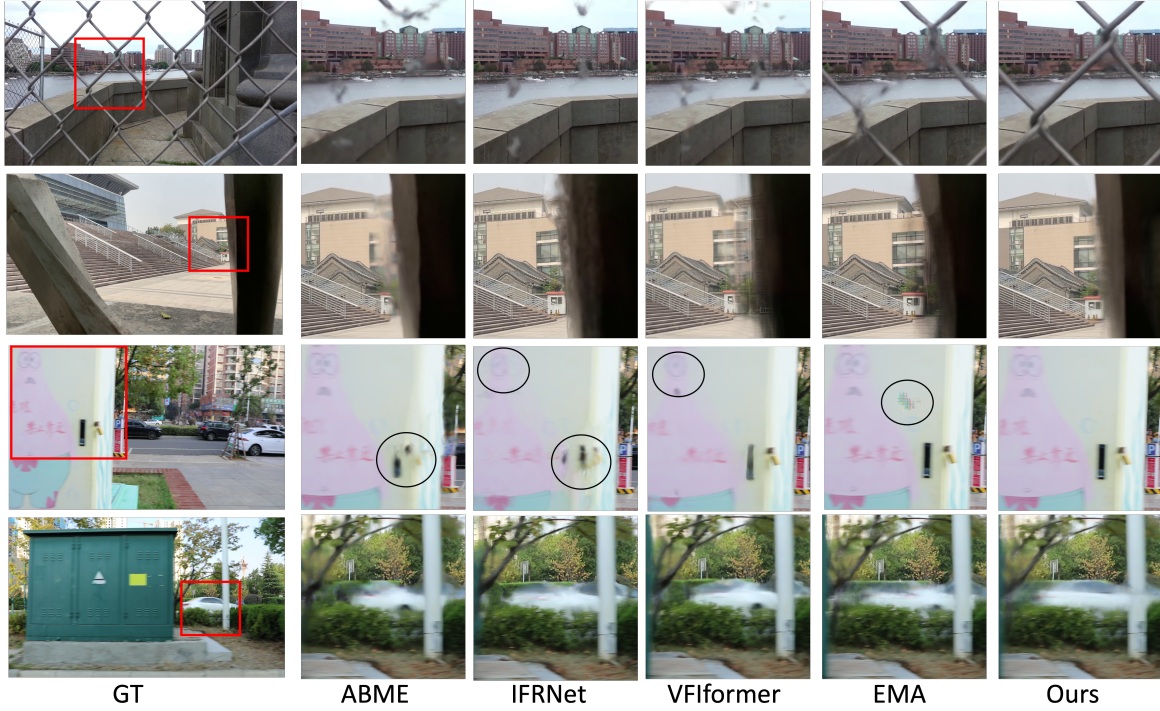


Figure 7.8: Qualitative results on our RWO dataset

7.5.2 Ablation Study

We evaluate the effectiveness of our method by comparing it with previous methods using their own training strategy and using our DMM. Table 7.1 "D" shows that our strategy significantly improves the interpolation performance, especially in the RWO dataset, and also achieves improvement in the Vimeo90K and SNU-FILM datasets in many cases. The reason is that DMM not only adds extra obstructions in the original dataset, but also acts like a mixing strategy that improves the generalization ability of the network by synthesizing more training samples.

We then incorporate our RSM module "-R" with a lightweight residual structure given by Equation (7.3). Therefore, the parameter increase is less than 1% of the corresponding original network. RSM further improves the performance in the large occlusion datasets SNU-FILM hard and extreme and our RWO dataset. Specifically, IFRNet-DR and VFformer-DR show lower quality on Vimeo 90K and SNU-FILM easy and medium, but show large improvements in SNU-FILM extreme and RWO. We attribute this to the fact that SNU-FILM hard and extreme have many large displacements that are difficult to warp to the same position. The RSM can efficiently

Method	Vimeo90K		SNU-FILM EASY		SNU-FILM MED		SNU-FILM Hard		SNU-FILM EX		RWO	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
RIFE [51]	35.61	0.9779	39.99	0.9904	35.68	0.9786	30.08	0.9327	24.83	0.8533	27.52	0.8837
RIFE-D	35.72	0.9780	39.95	0.9900	35.67	0.9783	30.14	0.9329	24.89	0.8548	27.73	0.8863
RIFE-DR	35.80	0.9783	40.03	0.9904	35.72	0.9789	30.23	0.9341	25.01	0.8569	27.82	0.8874
IFRNet [77]	35.80	0.9794	40.03	0.9905	35.94	0.9793	30.40	0.9358	25.05	0.8587	28.25	0.8939
IFRNet-D	35.86	0.9792	40.09	0.9904	36.03	0.9796	30.49	0.9357	25.17	0.8587	28.40	0.8950
IFRNet-DR	35.71	0.9789	39.95	0.9901	35.98	0.9795	30.54	0.9363	25.24	0.8592	28.53	0.8962
VFIformer [112]	36.50	0.9816	40.13	0.9907	36.09	0.9799	30.67	0.9378	25.43	0.8643	28.77	0.8994
VFIformer-D	36.57	0.9819	40.21	0.9912	36.16	0.9781	30.70	0.9380	25.47	0.8659	29.02	0.9014
VFIformer-DR	36.41	0.9804	40.02	0.9897	36.02	0.9776	30.80	0.9383	25.64	0.8679	29.11	0.9028
EMA [223]	36.64	0.9819	39.77	0.9908	35.98	0.9801	30.93	0.9395	25.69	0.8663	28.97	0.8978
EMA-D	36.67	0.9820	39.84	0.9908	35.97	0.9800	30.95	0.9397	25.73	0.8679	29.26	0.9032
EMA-DR	36.63	0.9817	39.88	0.9909	36.07	0.9799	31.02	0.9407	25.86	0.8701	29.33	0.9046

Table 7.1: Ablation study. Effect of applying DMM and RSM on the performance of the proposed model. “-D” indicates using DMM and “-DR” indicates using DMM and RSM. (Note: RSM is applied by Equation (7.3), and COA is evaluated in Table 7.2).

Method	Vimeo90K		SNU-FILM EASY		SNU-FILM MED		SNU-FILM Hard		SNU-FILM EX		RWO	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
AdaCof [85]	34.47	0.9730	39.80	0.9900	35.05	0.9754	29.46	0.9244	24.31	0.8439	26.43	0.8608
CAIN [23]	34.65	0.9729	39.89	0.9900	35.61	0.9776	29.90	0.9292	24.78	0.8507	27.19	0.8805
RIFE [51]	35.61	0.9779	39.99	0.9904	35.68	0.9786	30.08	0.9327	24.83	0.8533	27.52	0.8837
ABME [145]	36.18	0.9805	39.59	0.9901	35.77	0.9789	30.58	0.9364	25.42	0.8639	28.93	0.9029
IFRNet [77]	35.80	0.9794	40.03	0.9905	35.94	0.9793	30.40	0.9358	25.05	0.8587	28.25	0.8939
VFIformer [112]	36.50	0.9816	40.13	0.9907	36.09	0.9799	30.67	0.9378	25.43	0.8643	28.77	0.8994
EMA [223]	36.64	0.9819	39.77	0.9908	35.98	0.9801	30.93	0.9395	25.69	0.8663	28.97	0.8978
VFIformer-DRC	36.50	0.9814	40.11	0.9905	36.07	0.9794	30.87	0.9396	25.73	0.8694	29.20	0.9043
EMA-DRC	36.69	0.9821	39.96	0.9909	36.14	0.9802	31.10	0.9419	25.95	0.8722	29.56	0.9081

Table 7.2: Quantitative comparison of VFI methods on three datasets. The best result is in red, and the second best is in blue. "-DRC" indicates using DMM, RSM and COA with Equation (7.6).

capture those parts (see Figure 7.7). On the other hand, for SNU-FILM easy and medium, the displacement is small, so the RSM does not have much effect, and we obtain small amounts of degradation in some cases. Our DMM and RSM clearly achieve our objective to make VFI models robust to occlusion and obstruction without compromising the performance on clean scenes. Incorporating the COA step improves performance further which we will discuss next.

7.5.3 Comparison with the State-of-the-Art

We compare our methods with previous state-of-the-art VFI methods, including AdaCof [85], CAIN [23], RIFE [51], ABME [145], IFRNet [77], VFIfomer [112] and EMA [223]. We add our methods to the two highest performing previous methods: VFIfomer-DRC and EMA-DRC. "-DRC" means using our full method with all three techniques that we have proposed DMM, RSM and COA. As shown in Table 7.2, our methods outperform the previous algorithms on the RWO dataset by a considerable margin. Although, VFIfomer has a lower performance than ABME and EMA on the RWO dataset, our VFIfomer-DRC improves the results and outperforms all previous methods. The results clearly demonstrate the limitations of previous methods and the robustness of our approach. Moreover, our EMA-DRC also improves large occlusion cases in SNU-FLIM hard and extreme. It even shows competitive performance with SOTA methods on other benchmarks that do not contain large obstructions.

7.5.4 Qualitative Evaluation

Clean Scenes. We select samples from the SNU-Film dataset to compare the quality of different VFI methods. We compare our method with ABME, IFRNet, VFIfomer and EMA. As shown in Figure 7.7, the first column indicates the ground truth. The red rectangle indicates challenging regions. In these regions, previous methods do not perform well, but our method achieves accurate results. Specifically, the first row is a large motion case, where IFRNet and EMA warped pixels to different positions which caused ghosting. ABME and VFIfomer failed to predict this region. Our method preserves the details and the motion consistency. The third row is a tiny object case, where all four previous methods fail to reconstruct the thin sword. However, our method successfully preserves it. The fourth row is an occlusion case, where the person's face is visible in one input frame but partially visible in the other. Other methods produce blurry or distorted faces due to mismatches. Our method generates the clearest results. We suppose the reason is that RSM successfully identified the face, and COA effectively recovered it.

Scenes with Obstructions. We further visualize examples in our RWO dataset (Figure 7.8). We observe two issues of previous methods: 1) They sometimes fail to maintain the structure of the obstruction; and 2) They often have difficulty distinguishing the boundary of the obstruction and the background object and produce artifacts in the frames.

In the first row, ABME, IFRNet and VFIfomer fail to generate fences, and EMA partially fails, but our method succeeds in reconstructing it. In the second row, all four previous methods mixed the boundary between the pillar and the house. In comparison, our method shows a clearer shape. These examples demonstrate that our method successfully reduces errors in cases of large obstructions.

7.6 Limitations

Our method can handle most videos with large obstructions. However, it still fails in some cases, such as videos at night. The main reason is that, when pixels are all similar (darkness) in a scene, our method does not always capture ambiguous pixels/regions. Another case is the use of a flash in a video, which could lead to large dissimilarity for the whole frame and not to a dissimilar region which breaks our hypothesis. The COA module would have difficulties reconstructing the whole frame. However, our method is indeed robust to most cases of obstructions. We hope to address the remaining issues in future research.

7.7 Summary

In this chapter, we observe that large obstructions are difficult to warp to the correct position by optical flow, causing large errors and artifacts. To address this issue, we propose a novel VFI method that can handle videos with both obstructions and clean frames. We introduce a novel feature repair module, which first detects dissimilar regions of the two warped input frames, then repairs those regions by cross overlap attention (COA). We also propose a new data augmentation method (DMM) to make VFI networks robust to obstructions without extra images. Extensive experiments demonstrate that our method shows remarkable improvements on obstruction datasets, and also performs well on standard benchmarks.

Chapter 8

Conclusion

In this thesis, we explored the challenges in optical flow estimation which may currently limit the application of optical flow methods in real-world applications. We define three essential challenges: 1. Small Motions Boundary Error, 2. Generalization to Unseen Scenes, and 3. Simultaneous Robustness in Clean and Adverse Conditions. In each chapter, we covered the specific problem and introduced a solution for improving optical flow methods.

In Chapter 3, we first discovered that the downsampling process of the feature pyramid could cause blending of foreground objects with the background, which misleads the motion of flowing iterative refinement. Thus, we propose a novel residual feature pyramid module (RFPM), which retains important details in the feature map without changing the overall iterative refinement design of the optical flow estimation. RFPM incorporates a residual structure between multiple feature pyramids into a downsampling module that corrects the blending of objects across boundaries. By incorporating the module with the popular methods IRR-PWC and RAFT, respectively, we observe a clear improvement of motion boundaries in the optical flow estimate. Feature visualization also indicates that our RFPM can generate discriminative features for boundaries. Our method applied to RAFT [185] also achieved the top rank in the real-world driving benchmark KITTI at the time of submission. Furthermore, we proposed an efficient approach that allows existing methods to incorporate our module with a small fine-tuning step.

In Chapter 4, we propose to improve the generalization of unsupervised optical flow methods. We find existing generalization methods are not efficient for unsupervised optical flow methods. Thus, we propose a novel shared-weight teacher-student strategy and a content-aware regularization (CAR) module. Based on a tiny, learnable, content-aware mask, CAR is randomly applied to some channels in the convolutional layers during training to guide predictions in a shared-weight teacher-student strategy. CAR prevents motion estimation methods in unsupervised learning

from co-adaptation. By incorporating this technique, we achieved rank 1 in both the Sintel and KITTI benchmarks at the time of submission. Our methods also clearly improve generalization and achieve the highest results in cross-data evaluation, with a large margin compared to previous methods.

Outdoor applications often face various environmental and weather conditions. In Chapter 5, we improve optical flow robustness in both clean and adverse weather conditions. We observed that previous methods work well in clean/sunny weather but usually fail in adverse weather. We classify these weather conditions into two categories: Low-visibility, due to fog or night-time, results in low-quality features, i.e., low-visibility weather or limited illumination degrade the estimation of motion boundaries in optical flow. Motion noise, for example, due to snow or rain, results in methods to include rain streaks or snow in the flow estimation, causing the optical flow to be noisy. We first found that a previous self-guided loss method has flaws, as it assumes optical flow in sunny weather achieves the best results and uses it as a pseudo label to guide adverse condition flow. However, experiments show that more than half of the predicted adverse weather flow pixels are more accurate than the predicted sunny weather flow, which can generate noise during training. To address this, we propose a duo-photometric loss that ensures learning of adverse weather flow is not affected by the quality of clean flow. We propose a local-structure recovery technique to cope with low-visibility situations. We found that the features of low-visibility images have a large discrepancy compared to the features of clean images. Thus, we leverage contrastive learning to reduce the gap between these features, leading to similar results for clean and low-visibility images. We also propose a Noise Invariant Loss that encourages the network to ignore unnecessary noise motion like rain and snow.

In Chapter 6, we propose a systemic robustness evaluation of optical flow methods. We introduce the Flow-R benchmark, which, unlike previous benchmarks that create more complex scenes, replaces the texture or shape of main objects in existing benchmarks. We then collected failure cases where optical flow methods worked on the original image but failed when an element (texture or shape, etc.) was changed. Our analysis reveals two key limitations of optical flow methods: (1) greater sensitivity to shape changes than texture variations, and (2) a tendency to overfit to known objects while disregarding motion in unseen objects. These findings validate our evaluation framework and highlight critical directions for improving robustness in future work.

In Chapter 7, we study how the application of optical flow affects video frame interpolation. We found that video frame interpolation methods based on optical flow face challenges when dealing with large obstructions. Obstructions can cause large occlusions during matching, leading to poor optical flow quality. To address this, we propose an Obstruction Robustness Framework (ORF). The ORF contains two components: 1. A feature repair module that first captures ambiguous pixels in the synthetic frame using a region similarity map, then repairs these

ambiguous pixels with a cross-overlap attention module. 2. A data augmentation strategy that enables the network to handle dynamic obstructions without extra data. Our methods demonstrate state-of-the-art results in existing benchmarks and achieve the best results in the proposed large obstruction benchmarks.

The thesis improves the robustness of optical flow study in real-world scenes. When training data is labeled, the RFPM (Chapter 3) can be used to avoid blending of foreground objects with the background. However, in most outdoor conditions, we may not be able to obtain labeled data. In such cases, the CAR (Chapter 4) and RobFlow framework can be used to improve robustness and stability. To address large occlusion cases, the ORF can be incorporated to detect obstructions and then repair the features. However, there are still remaining issues that need to be addressed, as discussed in Chapter 7. In the following, we discuss their practical implications, limitations, and critical directions for future research.

8.1 Thesis Contributions

The thesis makes several contributions in optical flow estimation. These contributions address some of the gaps in the existing literature of the field discussed in Chapter 2, with a focus on robustness:

- A Residual Feature Pyramid Module (RFPM) that preserves motion boundaries by correcting blending artifacts during feature downsampling, improving flow accuracy in supervised methods.
- A lightweight fine-tuning strategy enabling RFPM to be easily integrated into existing optical flow architectures such as IRR-PWC and RAFT without major retraining.
- A Shared-Weight Teacher-Student framework combined with a Content-Aware Regularization (CAR) module that enhances generalization in unsupervised optical flow learning by preventing feature co-adaptation.
- A RobFlow framework that improves optical flow under adverse weather through a duo-photometric loss for better supervision, a local-structure recovery module to bridge low-visibility feature gaps, and a noise-invariant loss to suppress motion noise from rain and snow.
- A Flow-R benchmark designed for robustness evaluation, which exposes sensitivity to shape and texture changes in optical flow methods and reveals generalization weaknesses overlooked by traditional benchmarks.

- An Obstruction Robustness Framework (ORF) for video frame interpolation that detects and repairs ambiguous regions caused by large occlusions using a cross-overlap attention mechanism.
- A dynamic obstruction data augmentation technique that strengthens video frame interpolation models against dynamic occlusions without the need for extra annotated datasets.

These contributions advance the robustness and generalization capabilities of optical flow estimation, making them valuable for a wide range of real-world applications such as autonomous driving, video editing, robotics, and augmented reality systems.

8.2 Future Work

Building on the thesis work, we suggest focusing on the following directions.

Although some studies have aimed to improve the accuracy of optical flow, in Chapter 3, we investigated ways to handle small motion boundaries. However, further exploration is needed, particularly in dense scenes, with the presence of numerous overlapping objects, complex textures, and frequent occlusions. One promising idea is to integrate optical flow methods with segmentation or scene graphs to better identify small objects before performing motion estimation.

The Residual Feature Pyramid (RFPM) design effectively preserves motion boundaries, making it ideal for applications requiring fine-grained motion estimation (e.g., autonomous driving, robotic navigation). One may investigate the integration of RFPM into safety-critical systems where labeled data is available. Future work could optimize its computational efficiency for real-time deployment.

CAR’s shared-weight teacher-student framework and content-aware regularization offer a scalable solution for unsupervised optical flow, particularly in dynamic or data-scarce environments. Duo-photometric loss and noise-invariant learning address a critical gap in weather-robust optical flow. These approaches can be combined with CAR for autonomous driving in various environments.

Additionally, collecting optical flow data in adverse weather conditions is challenging. A systematic real-world dataset is needed to study the impact of different weather conditions on optical flow estimation. Future studies may find ways to collect relevant data. To address weather-related challenges, optical flow estimation could be combined with image restoration methods. Alternatively, it could be jointly learned with other perception tasks, such as depth estimation or pose estimation, to handle more complex scenarios effectively.

Using Flow-R alongside traditional benchmarks (Sintel, KITTI) can be used to stress-test new methods. Failure cases on Flow-R can guide improvements in generalization and structural robustness. The framework of Flow-R could be used for other perception tasks such as segmentation, pose estimation, and depth estimation, etc. It could be used to examine the behavior of methods and prevent unexpected failures.

Another relatively underexplored direction is the causal interpretation of the behavior of optical flow networks. According to the experiments in Chapter 7, most optical flow methods match objects primarily based on shape rather than texture. One experiment shows that even with the same shape but completely different textures, the method still matches the objects. This could pose potential hazards in real-world scenarios. For example, in driving scenarios, most cars have similar shapes, and we do not want the method to mismatch different cars in the two respective images as the same object. Thus, the thesis suggests that future work could focus on exploring matchings in optical flow methods, balancing texture and shape. State-of-the-art (SOTA) methods still suffer from poor generalization. The experiments show that these methods tend to memorize objects during training. The error is caused by the decoder. Optical flow methods should focus on motion matching rather than memorizing seen objects.

References

- [1] Automold-road-augmentation-library public. <https://github.com/UjjwalSaxena/Automold-Road-Augmentation-Library>. Accessed: 2024-02-26.
- [2] Using optical flow to control mobile robot navigation. *IFAC Proceedings Volumes*, 35(1):193–198, 2002.
- [3] Edward H Adelson, Charles H Anderson, James R Bergen, Peter J Burt, and Joan M Ogden. Pyramid methods in image processing. *RCA engineer*, 29(6):33–41, 1984.
- [4] FirstName Author and SecondName Coauthor. Robust unsupervised optical flow under low-visibility conditions. In *ECCV Workshop*, 2024.
- [5] Simon Baker, Stefan Roth, Daniel Scharstein, Michael J. Black, J.P. Lewis, and Richard Szeliski. A database and evaluation methodology for optical flow. In *ICCV*, 2007.
- [6] L. Bao, Q. Yang, and H. Jin. Fast edge-preserving patchmatch for large displacement optical flow. *IEEE Transactions on Image Processing*, 23(12):4996–5006, 2014.
- [7] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *CVPR*, 2019.
- [8] Aviram Bar-Haim and Lior Wolf. Scopeflow: Dynamic scene scoping for optical flow. In *CVPR*, 2020.
- [9] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? In *ICCV*, 2017.
- [10] J. Bigun, G.H. Granlund, and J. Wiklund. Multidimensional orientation estimation with applications to texture analysis and optical flow. *IEEE TPAMI*, 1991.

- [11] Michael J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 1996.
- [12] Michael J Black and Allan D Jepson. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE TPAMI*, 1996.
- [13] Randolph Blake, Robert Sekuler, and Emily Grossman. Motion processing in human visual cortex. *Current Directions in Psychological Science*, 2003.
- [14] Jean-Yves Bouguet. Pyramidal implementation of the affine Lucas Kanade feature tracker description of the algorithm. *Intel corporation*, 2001.
- [15] T Brox, A Bruhn, N Papenberg, and J Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, volume 3024 of *Lecture Notes in Computer Science*, pages 25–36, Prague, Czech Republic, May 2004. Springer.
- [16] Andrés Bruhn and Joachim Weickert. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005.
- [17] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, pages 611–625, Florence, Italy, October 2012. Springer.
- [18] Liyan Chen, Philip Gautier, and Sergul Aydore. Dropcluster: A structured dropout for convolutional networks, 2020.
- [19] Jiaxin Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *ICCV*, pages 686–695, Venice, Italy, 2017.
- [20] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *ICCV*, 2017.
- [21] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *ICCV*, pages 686–695, 2017.
- [22] Xianhang Cheng and Zhenzhong Chen. Multiple video frame interpolation via enhanced deformable separable convolution. *IEEE TPAMI*, 2021.
- [23] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *AAAI*, 2020.

- [24] Somayeh Danafar and Niloofar Gheissari. Human action recognition using optical flow and convolutional neural networks. In *IEEE International Conference on Machine Learning and Applications*. IEEE, 2011.
- [25] Tianyu Ding, Luming Liang, Zhihui Zhu, and Ilya Zharkov. Cdfi: Compression-driven network design for frame interpolation. In *CVPR*, 2021.
- [26] Xingbo Dong, Wanyan Xu, Zhihui Miao, Lan Ma, Chao Zhang, Jiewen Yang, Zhe Jin, Andrew Beng Jin Teoh, and Jiajun Shen. Abandoning the bayer-filter to see in the dark. In *CVPR*, 2022.
- [27] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *ICCV*, pages 2758–2766, Santiago, Chile, 2015.
- [28] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladlen Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, page 2758–2766, 2015.
- [29] Philipp Dufter, Martin Schmitt, and Hinrich Schütze. Position information in transformers: An overview. *Computational Linguistics*, 2022.
- [30] Yi Feng, Ruge Zhang, Jiayuan Du, Qijun Chen, and Rui Fan. Freespace optical flow modeling for automated driving. *IEEE/ASME Transactions on Mechatronics*, 2023.
- [31] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C. Berg. DSSD : Deconvolutional single shot detector. *CoRR*, abs/1701.06659, 2017.
- [32] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtualworlds as proxy for multi-object tracking analysis. In *CVPR*, pages 4340–4349. IEEE Computer Society, 2016.
- [33] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.
- [34] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361, Providence, RI, USA, June 2012.
- [35] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V. Le. Dropblock: A regularization method for convolutional networks. 31:10750–10760, 2018.

- [36] J.J. Gibson. *Perception of the Visual World*. Houghton Mifflin, 1950.
- [37] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, 2019.
- [38] Mark W. Greenlee and Hubert Kimmig. Visual perception and eye movements. In *Eye Movement Research*. Springer International Publishing, 2019.
- [39] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J. Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam H. Laradji, Hsueh-Ti Derek Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, A. Cengiz Öztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: A scalable dataset generator. In *CVPR*, pages 3739–3751, 2022.
- [40] Arpan Gupta and M Sakthi Balan. Action recognition from optical flow visualizations. In *International Conference on Computer Vision Image Processing*. Springer, 2018.
- [41] Zhou Haiyun, Xiang Xuezhi, Zhang Rongfang, Zhai Mingliang, and Syed Masroor Ali. Learning optical flow via deformable convolution and feature pyramid networks. In *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICC-SNT)*, pages 26–30, 2019.
- [42] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. 31, 2018.
- [43] Yunhui Han, Kunming Luo, Ao Luo, Jiangyu Liu, Haoqiang Fan, Guiming Luo, and Shuaicheng Liu. Realflow: Em-based realistic optical flow dataset generation from videos. In *ECCV*, pages 288–305, 2022.
- [44] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.
- [45] Berthold KP Horn and Brian G Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*. International Society for Optics and Photonics, 1981.
- [46] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, pages 17(1–3):185–203, 1981.

- [47] Michael Hornaek, Frederic Besse, Jan Kautz, Andrew Fitzgibbon, and Carsten Rother. Highly over parameterized optical flow using patchmatch belief propagation. In *ECCV*, pages 220–234, 2014.
- [48] Ping Hu, Simon Niklaus, Stan Sclaroff, and Kate Saenko. Many-to-many splatting for efficient video frame interpolation. In *CVPR*, 2022.
- [49] Yinlin Hu, Rui Song, and Yunsong Li. Efficient coarse-to-fine patchmatch for large displacement optical flow. In *CVPR*, pages 5704–5712, 2016.
- [50] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer: A transformer architecture for optical flow. In *ECCV*, 2022.
- [51] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *ECCV*, pages 624–642, Tel-Aviv, Israel, 2022. Springer.
- [52] Tak-Wai Hui and Chen Change Loy. LiteFlowNet3: Resolving Correspondence Ambiguity for More Accurate Optical Flow Estimation. In *ECCV*, pages 169–184, 2020.
- [53] Tak-Wai Hui and Xiaoou Tang Chen Change Loy. LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. In *CVPR*, page 8981–8989, Salt Lake City UT, 2018.
- [54] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow CNN - revisiting data fidelity and regularization. *IEEE TPAMI*, 43(8):2555–2569, 2020.
- [55] Junhwa Hur and Stefan Roth. Joint optical flow and temporally consistent semantic segmentation. In *ECCV*, 2016.
- [56] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, page 4761–4770, Long Beach, CA, USA, 2019.
- [57] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *CVPR*, pages 7394–7403, Virtual, June 2020.
- [58] 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 *CVPR*, pages 2462–2470, 2017.

- [59] Eddy Ilg, Ozgün Çiçek, Silvio Galesso, Aaron Klein, Osama Makansi, Frank Hutter, and Thomas Brox. Uncertainty estimates and multi-hypotheses networks for optical flow. In *ECCV*, 2018.
- [60] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, pages 1647–1655, Honolulu, HI, USA, 2017.
- [61] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *ECCV*, pages 614–630, 2018.
- [62] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. volume 37, pages 448–456. PMLR, 07–09 Jul 2015.
- [63] Joel Janai, Fatma Güneş, Anurag Ranjan, Michael J. Black, and Andreas Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *ECCV*, volume Lecture Notes in Computer Science, vol 11220, page 713–731, Munich, Germany, September 2018. Springer, Cham.
- [64] Jisoo Jeong, Jamie Menjay Lin, Fatih Porikli, and Nojun Kwak. Imposing consistency for optical flow estimation. In *CVPR*, 2022.
- [65] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super SloMo: High quality estimation of multiple intermediate frames for video interpolation. In *CVPR*, 2018.
- [66] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super sloMo: High quality estimation of multiple intermediate frames for video interpolation. In *CVPR*, pages 9000–9008, Salt Lake City, UT, USA, June 2018.
- [67] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *ICCV*, 2021.
- [68] Rico Jonschkowski, Austin Stone, Jonathan T Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. In *ECCV*, pages 557–572, Edinburgh, UK, 2020.
- [69] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM TOG*, 2016.

- [70] Tarun Kalluri, Deepak Pathak, Manmohan Chandraker, and Du Tran. Flavv: Flow-agnostic video representations for fast frame interpolation. In *CVPR*, pages 2071–2082, 2023.
- [71] Christoph Kamann and Carsten Rother. Benchmarking the robustness of semantic segmentation models with respect to common corruptions. *International Journal of Computer Vision*, 129(2):462–483, 2021.
- [72] Soo Ye Kim, Jihyong Oh, and Munchurl Kim. FISR: Deep joint frame interpolation and super-resolution with a multi-scale temporal loss. In *AAAI*, 2020.
- [73] Taewoo Kim, Yujeong Chae, Hyun-Kurl Jang, and Kuk-Jin Yoon. Event-based video frame interpolation with cross-modal asymmetric bidirectional motion fields. In *CVPR*, 2023.
- [74] Diederik P Kingma and J Ba. Adam: a method for stochastic optimization 3rd int. In *Conf. for Learning Representations, San*, 2014.
- [75] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.
- [76] Daniel Kondermann, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrulis, Alexander Brock, Burkhard Güssefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, Claus Brenner, and Bernd Jähne. The HCI benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *CVPRW*, pages 19–28, 2016.
- [77] Lingtong Kong, Boyuan Jiang, Donghao Luo, Wenqing Chu, Xiaoming Huang, Ying Tai, Chengjie Wang, and Jie Yang. Ifrnet: Intermediate feature refine network for efficient frame interpolation. In *CVPR*, 2022.
- [78] Lingtong Kong and Jie Yang. Mdflow: Unsupervised optical flow learning by reliable mutual knowledge distillation. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 677–688, 2022.
- [79] Tao Kong, Fuchun Sun, Chuanqi Tan, Huaping Liu, and Wenbing Huang. Deep feature pyramid reconfiguration for object detection. In *ECCV*, pages 169–185, 2018.
- [80] Tom Koren, Lior Talker, Michael Dinerstein, and Roy J. Jevnisek. Consistent semantic attacks on optical flow. In *ACCV*, pages 1658–1674, 2022.
- [81] Eileen Kowler. Eye movements and perception: A selective review. *Journal of Vision*, 2011.

- [82] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012.
- [83] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *ICLR*.
- [84] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *ECCV*, 2018.
- [85] Hyeongmin Lee, Taeoh Kim, Tae-young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. Adacof: Adaptive collaboration of flows for video frame interpolation. In *CVPR*, 2020.
- [86] Sangjin Lee, Hyeongmin Lee, Chajin Shin, Hanbin Son, and Sangyoun Lee. Exploring discontinuity for video frame interpolation. In *CVPR*, 2023.
- [87] Baixin Li, Jie Liu, and Bernard Chiu. Non-rigid medical image registration using optical flow and locally refined control point setup in multilevel free-form deformation. In *IEEE International Symposium on Biomedical Imaging*. IEEE, 2007.
- [88] Haipeng Li, Kunming Luo, and Shuaicheng Liu. Gyroflow: Gyroscope-guided unsupervised optical flow learning. In *ICCV*, 2021.
- [89] Haipeng Li, Kunming Luo, Bing Zeng, and Shuaicheng Liu. Gyroflow+: Gyroscope-guided unsupervised deep homography and optical flow learning. *IJCV*, 2024.
- [90] Jianfeng Li, Junqiao Zhao, Shuangfu Song, and Tiantian Feng. Occlusion aware unsupervised learning of optical flow from video. In *Thirteenth International Conference on Machine Vision*, volume 11605, page 224 – 231, Rome, Italy, 2021. International Society for Optics and Photonics, SPIE.
- [91] Ruoteng Li, Robby T Tan, and Loong-Fah Cheong. Robust optical flow in rainy scenes. In *ECCV*, 2018.
- [92] Ruoteng Li, Robby T Tan, Loong-Fah Cheong, Angelica I Aviles-Rivero, Qingnan Fan, and Carola-Bibiane Schonlieb. Rainflow: Optical flow under rain streaks and rain veiling effect. In *ICCV*, 2019.
- [93] Xiaobo* Liang, Lijun* Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. R-drop: Regularized dropout for neural networks. 2021.

- [94] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017.
- [95] Chang Liu, Yinpeng Dong, Wenzhao Xiang, Xiao Yang, Hang Su, Jun Zhu, Yuefeng Chen, Yuan He, Hui Xue, and Shibao Zheng. A comprehensive study on robustness of image classification models: Benchmarking and rethinking. *International Journal of Computer Vision*, pages 1–23, 2024.
- [96] Liang Liu, Guangyao Zhai, Wenlong Ye, and Yong Liu. Unsupervised learning of scene flow estimation fusing with local rigidity. In *International Joint Conference on Artificial Intelligence, IJCAI*, pages 876–882, July 2019.
- [97] Liang Liu, Jiangning Zhang, Ruifei He, Yong Liu, Yabiao Wang, Ying Tai, Donghao Luo, Chengjie Wang, Jilin Li, and Feiyue Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *CVPR*, 2020.
- [98] Liang Liu, Jiangning Zhang, Ruifei He, Yong Liu, Yabiao Wang, Ying Tai, Donghao Luo, Chengjie Wang, Jilin Li, and Feiyue Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *CVPR*, pages 6488–6497, Virtual, June 2020.
- [99] Pengpeng Liu, Irwin King, Michael R. Lyu, and Jia Xu. Ddflow: Learning optical flow with unlabeled data distillation. In *AAAI*, page 8571–8578, Honolulu, HI, USA, 2019.
- [100] Pengpeng Liu, Michael R. Lyu, Irwin King, and Jia Xu. Selfflow: Self-supervised learning of optical flow. In *CVPR*, page 6707–6716, Long Beach, Ca, USA, 2019.
- [101] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *ECCV*, pages 21–37, 2016.
- [102] Yu-Lun Liu, Wei-Sheng Lai, Ming-Hsuan Yang, Yung-Yu Chuang, and Jia-Bin Huang. Learning to see through obstructions. In *CVPR*, 2020.
- [103] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.
- [104] Gucan Long, Laurent Kneip, Jose M Alvarez, Hongdong Li, Xiaohu Zhang, and Qifeng Yu. Learning image matching by simply watching video. In *ECCV*. Springer, 2016.

- [105] Libo Long, Xiao Hu, and Jochen Lang. Learning to handle large obstructions in video frame interpolation. In *ACM MM*, 2024.
- [106] Libo Long, Xiao Hu, and Jochen Lang. Learning to handle large obstructions in video frame interpolation. In *ACM MM*. ACM, 2024.
- [107] Libo Long, Xiao Hu, and Jochen Lang. Shape and texture: What influences reliable optical flow estimation? In *CVPR*, 2025. ©2025 IEEE. Reprinted, with permission, from *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025, pp. 0000–0000.
- [108] Libo Long and Jochen Lang. Detail preserving residual feature pyramid modules for optical flow. In *WACV*. IEEE, 2022. ©2022 IEEE. Reprinted, with permission, from *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022.
- [109] Libo Long and Jochen Lang. Regularization for unsupervised learning of optical flow. *Sensors*, 2023.
- [110] David Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–, 11 2004.
- [111] Guo Lu, Xiaoyun Zhang, Li Chen, and Zhiyong Gao. Novel integration of frame rate up conversion and hevc coding based on rate-distortion optimization. *IEEE TIP*, 2018.
- [112] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia. Video frame interpolation with transformer. In *CVPR*, 2022.
- [113] Yao Lu, Jack Valmadre, Heng Wang, Juho Kannala, Mehrtash Harandi, and Philip H. S. Torr. Devon: Deformable volume network for learning optical flow. pages 2694–2702, Virtual, June 2020.
- [114] Yawen Lu, Qifan Wang, Siqi Ma, Tong Geng, Yingjie Victor Chen, Huaijin Chen, and Dongfang Liu. Transflow: Transformer as flow learner. In *CVPR*, 2023.
- [115] Zhong-Lin Lu and George Sperling. Motion perception models. *Research Paper Examples*, 2008.
- [116] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, 1981.

- [117] Chenxu Luo, Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, Ramakant Nevatia, and Alan Loddon Yuille. Every pixel counts ++: Joint learning of geometry and motion with 3d holistic understanding. *IEEE TPAMI*, 42(10):2624–2641, 2020.
- [118] Kunming Luo, Chuan Wang, Shuaicheng Liu, Haoqiang Fan, Jue Wang, and Jian Sun. Upflow: Upsampling pyramid for unsupervised optical flow learning. In *CVPR*, pages 1045–1054, Virtual, June 2021.
- [119] Kunming Luo, Chuan Wang, Nianjin Ye, Shuaicheng Liu, and Jue Wang. Occinflow: Occlusion-inpainting optical flow estimation by unsupervised learning. In *arXiv preprint arXiv:2006.16637*, 2020.
- [120] Nikolaus Mayer, Eddy Ilg, Philipp Fischer, Caner Hazirbas, Dilara Aseem, Ali Taha, Oktay Goksel, Stefan Roth, and Thomas Brox. What makes good synthetic training data for learning disparity and optical flow estimation? *International Journal of Computer Vision*, 126(9):942–960, 2018.
- [121] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, pages 4040–4048, Las Vegas, NV, USA, June 2016.
- [122] Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. 2023.
- [123] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, page 7255–7263, New Orleans, La, USA, February 2018.
- [124] Facundo Memoli and Guillermo Sapiro. Medical image registration and interpolation by optical flow with maximal rigidity. *Journal of Mathematical Imaging and Vision*, 2006.
- [125] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, pages 3061–3070, Boston, MA, USA, 2015.
- [126] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis*, 2015.
- [127] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, pages 60–76, 2018.

- [128] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus Gross, and Christopher Schroers. Phasenet for video frame interpolation. In *CVPR*, 2018.
- [129] Simone Meyer, Oliver Wang, Henning Zimmer, Max Grosse, and Alexander Sorkine-Hornung. Phase-based frame interpolation for video. In *CVPR*, 2015.
- [130] Sergiu Mocanu, Alan R Moody, and April Khademi. Flowreg: Fast deformable unsupervised medical image registration using optical flow. *Journal of Machine Learning for Biomedical Imaging*, 2021.
- [131] Mikhail Mozerov. Constrained optical flow estimation as a matching problem. *IEEE Transactions on Image Processing*, 22(5):2044–2055, 2013.
- [132] H.-H. Nagel. Constraints for the estimation of displacement vector fields from image sequences. In *IJCAI*, volume 2, pages 945–951, Karlsruhe, West Germany, August 1983.
- [133] H.-H. Nagel and A. Gehrke. Spatiotemporally adaptive estimation and segmentation of of-fields. In *ECCV*. Springer, 1998.
- [134] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *CVPR*, 2018.
- [135] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *CVPR*, 2017.
- [136] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *ICCV*, 2017.
- [137] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli Kim, and Seon Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *CVPR*, 2018.
- [138] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation with episodic graph memory networks. In *ICCV*, 2019.
- [139] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, and Florencia Leoni Aleman et al. Gpt-4 technical report, 2024.
- [140] Comfy Org. Comfyui: A modular gui for stable diffusion, 2023. Accessed: 2024-11-21.
- [141] Bojun Ouyang and Dan Raviv. Occlusion guided scene flow estimation on 3d point clouds. In *CVPRW*, pages 1049–1058, 2021.

- [142] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 2009.
- [143] Junheum Park, Jintae Kim, and Chang-Su Kim. BiFormer: Learning bilateral motion estimation via bilateral transformer for 4k video frame interpolation. In *CVPR*, 2023.
- [144] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In *ECCV*, 2020.
- [145] Junheum Park, Chul Lee, and Chang-Su Kim. Asymmetric bilateral motion estimation for video frame interpolation. In *ICCV*, 2021.
- [146] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. 32:8024–8035, 2019.
- [147] Tomer Peleg, Pablo Szekely, Doron Sabo, and Omry Sendik. IM-net for high resolution video frame interpolation. In *CVPR*, 2019.
- [148] AJ Piergiovanni and Michael S Ryoo. Representation flow for action recognition. In *CVPR*, page 9939–9948, 2019.
- [149] Fabio Pizzati, Pietro Cerri, and Raoul de Charette. Comogan: continuous model-guided image-to-image translation. In *CVPR*, 2021.
- [150] Anurag Ranjan and Michael J. Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017.
- [151] Anurag Ranjan and Michael J. Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, page 2720–2729, Honolulu, HI, USA, July 2017.
- [152] Anurag Ranjan, Varun Jampani, Lukas Balles, Deqing Sun, Kihwan Kim, Jonas Wulff, and Michael J.Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *CVPR*, pages 12232–12241, Long Beach, Ca, USA, 2019.
- [153] Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J. Black. Attacking optical flow. In *ICCV*, pages 1–10, 2019.

- [154] Fitsum Reda, Janne Kontkanen, Eric Tabellion, Deqing Sun, Caroline Pantofaru, and Brian Curless. Film: Frame interpolation for large motion. In *ECCV*, 2022.
- [155] Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised deep learning for optical flow estimation. In *AAAI*, volume 31, 2017.
- [156] Zhile Ren, Orazio Gallo, Deqing Sun, Ming-Hsuan Yang, Erik B Sudderth, and Jan Kautz. A fusion approach for multi-frame optical flow estimation. pages 2077–2086, 2019.
- [157] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, pages 1164–1172, Boston MA, 2015.
- [158] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10807–10817. IEEE, 2022.
- [159] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. volume 9351 of *LNCS*, pages 234–241. Springer, 2015.
- [160] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global Sensitivity Analysis: The Primer*. John Wiley & Sons, 2008.
- [161] Kieran Saunders, George Vogiatzis, and Luis Manso. Self-supervised monocular depth estimation: Let’s talk about the weather. In *ICCV*, 2023.
- [162] Jenny Schmalfuss, Lukas Mehl, and Andrés Bruhn. Distracting downpour: Adversarial weather attacks for motion estimation. In *ICCV*, 2023.
- [163] Jenny Schmalfuss, Lukas Mehl, and Andrés Bruhn. Adversarial weather attacks for motion estimation. In *ICCV*, pages 10106–10116, 2023.
- [164] Jenny Schmalfuss, Philipp Scholze, and Andrés Bruhn. A perturbation-constrained adversarial attack for evaluating the robustness of optical flow. *ECCV*, 2022.
- [165] Philipp Schrodi, Jenny Schmalfuss, and Andrés Bruhn. A less constrained global attack on optical flow. In *ECCV*, pages 183–200. Springer, 2022.
- [166] Laura Sevilla-Lara, Yiyi Liao, Fatma Güney, Varun Jampani, Andreas Geiger, and Michael J Black. On the integration of optical flow and action recognition. In *ICPR*. Springer, 2018.

- [167] Laura Sevilla-Lara, Deqing Sun, Varun Jampani, and Michael J Black. Optical flow with semantic segmentation and localized layers. In *CVPR*, 2016.
- [168] Hao Shi, Yifan Zhou, Kailun Yang, Xiaoting Yin, and Kaiwei Wang. Csflow: Learning optical flow via cross strip correlation for autonomous driving. In *IEEE Intelligent Vehicles Symposium (IV)*, 2022.
- [169] Xiaoyu Shi, Zhaoyang Huang, Dasong Li, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation. In *CVPR*, 2023.
- [170] Zheng Shi, Yuval Bahat, Seung-Hwan Baek, Qiang Fu, Hadi Amata, Xiao Li, Praneeth Chakravarthula, Wolfgang Heidrich, and Felix Heide. Seeing through obstructions with diffractive cloaking. *ACM TOG*, 2022.
- [171] Patrice Y Simard, Dave Steinkraus, and John C Platt. Best practices for convolutional neural networks applied to visual document analysis. In *International Conference on Document Analysis and Recognition*, 2003.
- [172] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. 27:568–576, 2014.
- [173] Yuxi Song, Chao Ma, Lijun Gong, Jiawei Zhang, Rynson WH Lau, and Ming-Hsuan Yang. Video object segmentation with re-identification. In *Asian Conference on Computer Vision*, 2018.
- [174] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
- [175] Austin Stone, Daniel Maurer, Alper Ayvaci, Anelia Angelova, and Rico Jonschkowski. SMURF: Self-teaching multi-frame unsupervised raft with full-image warping. In *CVPR*, pages 3887–3896, Nashville, TN, USA, June 2021.
- [176] Xiuchao Sui, Shaohua Li, Xue Geng, Yan Wu, Xinxing Xu, Yong Liu, Rick Goh, and Hongyuan Zhu. Craft: Cross-attentional flow transformer for robust optical flow. In *CVPR*, 2022.
- [177] Chen Sun, Abhinav Shrivastava, Carl Vondrick, Kevin Murphy, Rahul Sukthankar, and Cordelia Schmid. Actor-centric relation network. In *ECCV*. Springer, 2018.

- [178] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *CVPR*, page 2432–2439, San Francisco, CA, USA, June 2010.
- [179] Deqing Sun, Stefan Roth, and Michael J Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *IJCV*, 2014.
- [180] Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T. Freeman, and Ce Liu. Autoflow: Learning a better training set for optical flow. *CVPR*, 2021.
- [181] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, page 8934–8943, Salt Lake City, UT, USA, 2018.
- [182] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *IEEE TPAMI*, 42(6):1408–1423, 2020.
- [183] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *IEEE TPAMI*, 2020.
- [184] Shuyang Sun, Zhanghui Kuang, Wanli Ouyang, Lu Sheng, and Wei Zhang. Optical flow guided feature: A fast and robust motion representation for video action recognition. In *CVPR*, 2018.
- [185] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *ECCV*, page 402–419, Glasgow, UK, 2020.
- [186] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *CVPR*, pages 648–656, Boston, Ma, USA, 2015.
- [187] Maxime Tremblay, Shirsendu Sukanta Halder, Raoul De Charette, and Jean-François Lalonde. Rain rendering for evaluating and improving robustness to bad weather. *IJCV*, 2021.
- [188] Oleh Tretiak and Louis Pastor. Velocity estimation from image sequences with second order differential operators. In *ICPR*, Montreal, Canada, July 1984.
- [189] Prune Truong, Martin Danelljan, and Radu Timofte. GLU-Net: Global-local universal network for dense flow and correspondences. In *CVPR*, page 3262–3272, Virtual, 2020.

- [190] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. Video segmentation via object flow. In *CVPR*, 2016.
- [191] Andreas Veit, Michael J. Wilber, and Serge J. Belongie. Residual networks are exponential ensembles of relatively shallow networks. *NeurIPS*, 2016.
- [192] Chuan Wang, Shuaicheng Liu, and Bing Zeng. Occlusion aware unsupervised learning of optical flow. In *IEEE Transactions on Image Processing*, volume 28, page 3077–3087, 2019.
- [193] Guangming Wang, Shuaiqi Ren, and Hesheng Wang. Nccflow: Unsupervised learning of optical flow with non-occlusion from geometry. *IEEE Transactions on Intelligent Transportation Systems*, 2022.
- [194] Hengli Wang, Rui Fan, and Ming Liu. Cot-amflow: Adaptive modulation network with co-teaching strategy for unsupervised optical flow estimation. pages 143–155. PMLR, 2021.
- [195] Jianyuan Wang, Yiran Zhong, Yuchao Dai, Kaihao Zhang, Pan Ji, and Hongdong Li. Displacement-invariant matching cost learning for accurate optical flow estimation. 33:15220–15231, 2020.
- [196] Song Wang and Zengfu Wang. Optical flow estimation with occlusion detection. *Algorithms*, 12(5), 2019.
- [197] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 2004.
- [198] Joachim Weickert and Christoph Schnörr. Variational optic flow computation with a spatio-temporal smoothness constraint. *Journal of Mathematical Imaging and Vision*, 14(3):245–255, 2001.
- [199] Haiyan Wu, Yanyun Qu, Shaohui Lin, Jian Zhou, Ruizhi Qiao, Zhizhong Zhang, Yuan Xie, and Lizhuang Ma. Contrastive learning for compact single image dehazing. In *CVPR*, 2021.
- [200] Jonas Wulff, Daniel J Butler, Garrett B Stanley, and Michael J Black. Lessons and insights from creating a synthetic optical flow benchmark. In *ECCV*, page 168–177. Springer, 2012.
- [201] Jonas Wulff, Laura Sevilla-Lara, and Michael J. Black. Optical flow in mostly rigid scenes. In *CVPR*, pages 6911–6920, Honolulu, HI, USA, 2017.

- [202] Xiaoyu Xiang, Yapeng Tian, Yulun Zhang, Yun Fu, Jan P. Allebach, and Chenliang Xu. Zooming slow-mo: Fast and accurate one-stage space-time video super-resolution. In *CVPR*, 2020.
- [203] Xuezhi Xiang, Mingliang Zhai, Rongfang Zhang, Yulong Qiao, and Abdulmotaleb El Sadik. Deep optical flow supervised learning with prior assumptions. *IEEE Access*, 6:43222–43232, 2018.
- [204] Shuang Xie, Po Kong Lai, Robert Laganiere, and Jochen Lang. Effective convolutional neural network layers in flow estimation for omni-directional images. pages 671–680, Montreal, QC, Canada, 2019.
- [205] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *CVPR*, 2022.
- [206] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *CVPR*, pages 3718–3727, Long Beach, CA, USA, 2019.
- [207] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision (IJCV)*, 127(8):1106–1125, 2019.
- [208] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *IJCV*, 2019.
- [209] Wending Yan, Aashish Sharma, and Robby T Tan. Optical flow in dense foggy scenes using semi-supervised learning. In *CVPR*, 2020.
- [210] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. 32:794–805, 2019.
- [211] Wenhan Yang, Robby T. Tan, Jiashi Feng, Zongming Guo, Shuicheng Yan, and Jiaying Liu. Joint rain detection and removal from a single image with contextualized deep networks. *IEEE TPAMI*, 2020.
- [212] Yanchao Yang, Antonio Loquercio, Davide Scaramuzza, and Stefano Soatto. Unsupervised moving object detection via contextual information separation. In *CVPR*, pages 879–888, Long Beach, CA, USA, 2019.
- [213] Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, and Ramakant Nevatia. Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding. In *ECCV*, pages 691–709, Munich, Germany, September 2018.

- [214] Pradeep Yarlagadda and Björn Ommer. From meaningful contours to discriminative object shape. In *ECCV*.
- [215] Z. Yin and J. Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, pages 1983–1992, Salt Lake City, UT, USA, June 2018.
- [216] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *CVPR*, 2019.
- [217] Jun-Sang Yoo, Hongjae Lee, and Seung-Won Jung. Video object segmentation-aware video frame interpolation. In *ICCV*, 2023.
- [218] Jason J. Yu, Adam W. Harley, and Konstantinos G. Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *Computer Vision–ECCV 2016 Workshops*, pages 3–10, Amsterdam, The Netherlands, October 2016.
- [219] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Learning enriched features for real image restoration and enhancement. In *ECCV*, 2020.
- [220] Mingliang Zhai, Xuezhi Xiang, Rongfang Zhang, Nan Lv, and Abdulmotaleb El Saddik. Learning optical flow using deep dilated residual networks. *IEEE Access*, 7:22566–22578, 2019.
- [221] Mingliang Zhai, Xuezhi Xiang, Rongfang Zhang, Nan Lv, and Abdulmotaleb El Saddik. Optical flow estimation using channel attention mechanism and dilated convolutional neural networks. *Neurocomputing*, 368:124–132, 2019.
- [222] Feihu Zhang, Oliver J. Woodford, Victor Adrian Prisacariu, and Philip H. S. Torr. Separable flow: Learning motion cost volumes for optical flow estimation. In *ICCV*, pages 10807–10817, 2021.
- [223] Guozhen Zhang, Yuhan Zhu, Haonan Wang, Youxin Chen, Gangshan Wu, and Limin Wang. Extracting motion and appearance via inter-frame attention for efficient video frame interpolation. In *CVPR*, 2023.
- [224] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE TIP*, 2017.

- [225] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.
- [226] Mingfang Zhang, Yinqiang Zheng, and Feng Lu. Optical flow in the dark. *IEEE TPAMI*, 2022.
- [227] Yuxuan Zhang, Xiang Zhang, Zhen Li, Shuo Wang, and Shaoting Zhang. A novel optical flow based deformable registration method for brain mr images. *Neurocomputing*, 2018.
- [228] Yuxuan Zhang, Xiang Zhang, Zhen Li, Shuo Wang, and Shaoting Zhang. Optical flow based deformable registration of brain mri images with tumor. In *IEEE International Conference on Bioinformatics and Biomedicine*. IEEE, 2018.
- [229] Shengyu Zhao, Yilun Sheng, Yue Dong, Eric I-Chao Chang, and Yan Xu. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *CVPR*, pages 6277–6286, Virtual, June 2020.
- [230] Shiyu Zhao, Long Zhao, Zhixing Zhang, Enyu Zhou, and Dimitris N. Metaxas. Global matching with overlapping attention for optical flow estimation. In *CVPR*, 2022.
- [231] Yinqiang Zheng, Mingfang Zhang, and Feng Lu. Optical flow in the dark. *IEEE TPAMI*, 2022.
- [232] Zhenzhu Zheng and Xi Peng. Self-guidance: Improve deep neural network generalization via knowledge distillation. page 351–360, 2022.
- [233] Yiran Zhong, Pan Ji, Jianyuan Wang, Yuchao Dai, and Hongdong Li. Unsupervised deep epipolar flow for stationary or dynamic scenes. In *CVPR*, page 12069–12078, Long Beach, Ca, USA, 2019.
- [234] Hanyu Zhou, Yi Chang, Gang Chen, and Luxin Yan. Unsupervised hierarchical domain adaptation for adverse weather optical flow. In *AAAI*, 2023.
- [235] Hanyu Zhou, Yi Chang, Wending Yan, and Luxin Yan. Unsupervised cumulative domain adaptation for foggy scene optical flow. In *CVPR*, 2023.
- [236] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice Hall, 1996.
- [237] Shili Zhou, Ruian He, Weimin Tan, and Bo Yan. Samflow: Eliminating any fragmentation in optical flow with segment anything model. *AAAI*, 2024.

- [238] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *ECCV*, 2016.
- [239] Chengxuan Zhu, Renjie Wan, Yunkai Tang, and Boxin Shi. Occlusion-free scene recovery via neural radiance fields. In *CVPR*, 2023.
- [240] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *ICCV*, 2017.
- [241] C. Lawrence Zitnick, Nebojsa Jojic, and Sing Bing Kang. Consistent segmentation for optical flow estimation. In *ICCV*, volume 2, pages 1308–1315 Vol. 2, 2005.