Detail Preserving Residual Feature Pyramid Modules for Optical Flow

Libo Long, Jochen Lang
EECS, University of Ottawa
(llong014,jlang)@uottawa.ca

Abstract

Feature pyramids and iterative refinement have recently led to great progress in optical flow estimation. However, downsampling in feature pyramids can cause blending of foreground objects with the background, which will mislead subsequent decisions in the iterative processing. The results are missing details especially in the flow of thin and of small structures. 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. We demonstrate how to integrate our module with two state-of-the-art iterative refinement architectures. Results show that our RFPM visibly reduces flow errors and improves state-of-art performance in the clean pass of Sintel, and is one of the top-performing methods in KITTI. According to the particular modular structure of RFPM, we introduce a special fine-tuning approach that can dramatically decrease the training time compared to a typical full optical flow training schedule on multiple datasets.

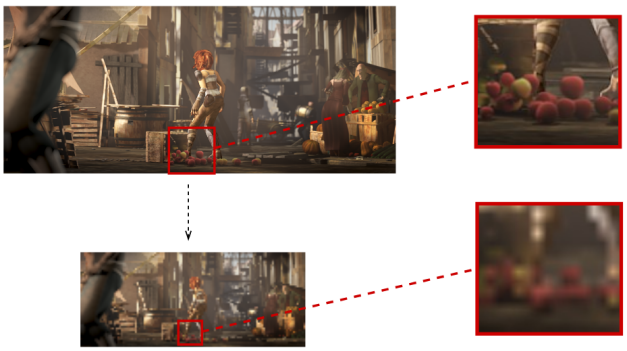
1. Introduction

Optical flow estimation is a key problem in computer vision and a fundamental building block of many high-level computer vision applications [27, 35, 20]. Recent optical flow estimation has greatly benefited from learning-based CNN architectures. Flownet [6] is the first CNN which directly predicts the optical flow with an encoder-decoder architecture. PWC-Net [38] and LiteFlowNet [11] 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 [22, 43, 15, 41, 9]. However, we observe that the iterative refinement design is not without major drawbacks which limits optical flow estimation to improve further. Fig. 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 Fig. 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.

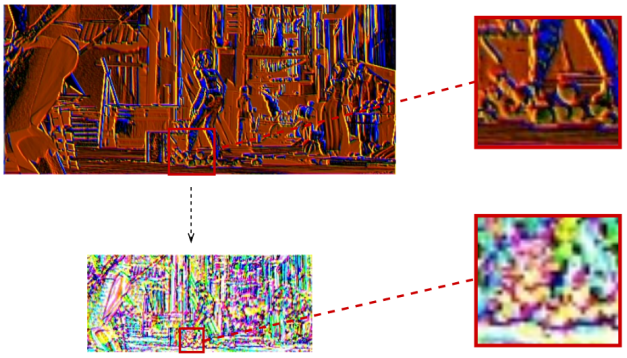
We introduce a new pyramid module, Residual Feature Pyramids Module (RFPM), to address the loss of detail in feature pyramids. We demonstrate how RFPM reduces error in optical flow in Sintel and KITTI, even when integrated into top-performing optical flow methods such as RAFT [40] and IRR-PWC [14]. Our experimental results shows that RFPM-RAFT achieves state-of-the-art performance on MPI Sintel [4] (Clean pass), and KITTI 2015 [28] 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 efficient fine-tuning 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 fine-tuning for RFPM. Next, we will review related optical flow methods, strategies to avoid blur across motion boundaries and the use of feature pyramids.



(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 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.

2. Related work

Optical Flow Estimation. In optical flow the use of multi-resolution representations goes back to Lucas and Kanade [25] who used bandpass-filtering to register features across scales. Bouguet [3] reported a popular implementation using image pyramids. We refer the reader to the work of Sun et al. [36] and their analysis of classic optical flow methods derived from Horn and Schunck [8]. FlowNet [6] is the first end-to-end trainable CNN network based on a U-Net [33] 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. [16] 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 [30] introduces a light-weight network by using the feature pyramid and warping within the image pyramid. Sun et al. [38] 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 [22, 14, 31]. Recently, IRR [14] and RAFT [40] 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 [46, 2] segment the image into regions using shape or color, then estimating motion by matching the regions. In recent CNN based methods, Sevilla et al. [34] and Hur et al. [13] add a pretrained semantic segmentation neural network as an additional component to improve optical flow but these methods are not end-to-end trainable. SegFlow [5] 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 [24, 1]. 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. [23] 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. [7], SSD fails to detect small instances. Subsequently, Feature Pyramid Networks [21] 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. [19] 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 [14, 38, 45] 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. Recently, [32] propose a residual skip connections of feature pyramid which improved accuracy in some classic optical flow methods [38, 11].

To our knowledge, most optical flow methods still use an

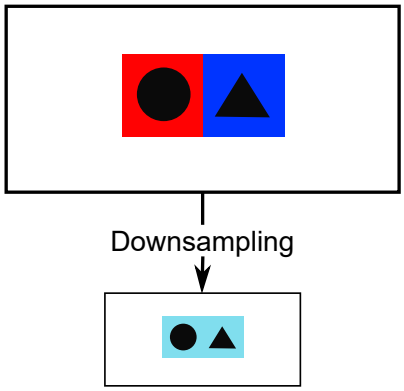


Figure 2. Illustration of flow error. The top shows two pixels of a high resolution image, red indicates the flow of the object is to the left, and blue indicates the flow of the object is to the right. The top high resolution image predicts the flow of the two objects correctly (circle towards left, 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.

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. 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.

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 [38, 14] 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

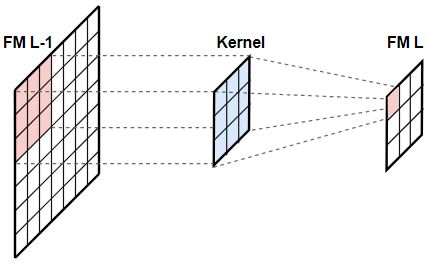
$$V = c(\mathcal{F}_t^l, w(\mathcal{F}_{t+1}^l)), \tag{1}$$

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.

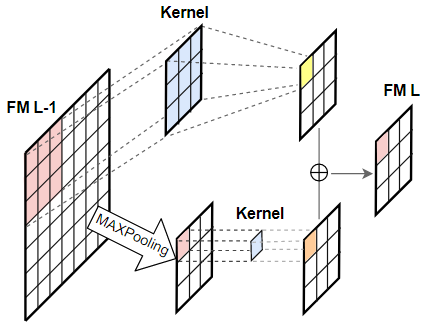
At each level l of the feature pyramid, the output pixel value $x_{i,j}^l$ is a weighted sum of pixel value of the previous level

$$z_{i,j}^l = \sum_{x,y} z_{i+x,j+y}^{l-1} k_3(x,y), \tag{2}$$

where i,j are the indices of the pixel, k_3 is the 3*3 convolution kernel, $-1 \leq x, y \leq 1$ (3*3 kernel). We refer to this as Weighted Feature Downsampling (WFD) in Fig. 3(a).



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



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

Figure 3. Downsampling modules. WFD is the original module used in CNN optical flow estimation [38, 10, 14]. RFD combines WFD and Max Pooling (MP) with a residual-like design.

We observe a major consequence of this architecture is blurring of motion boundaries. Fig. 2 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 [42]. RFD learns to enhance edge areas because MP is expected to keep higher weight pixels for the next level. Although MP itself might 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 ex-

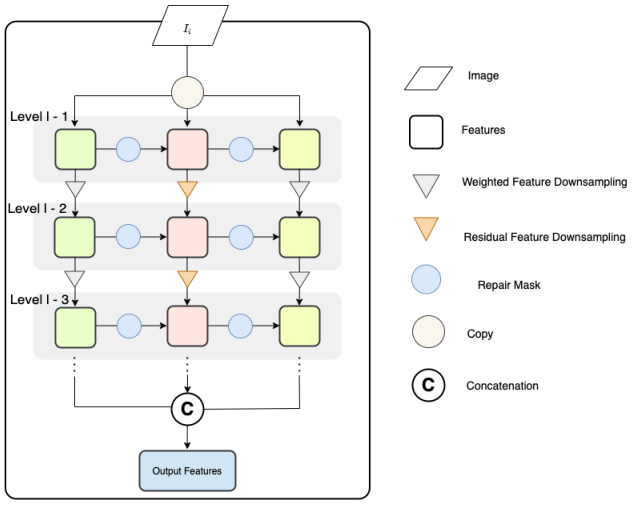


Figure 4. Architecture of RFPM. Feature maps are extracted with three different pyramids.

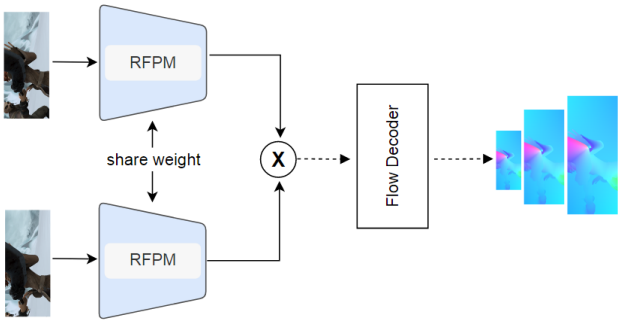


Figure 5. The network extracts a pair of feature maps by a share weight RFPM and feeds them into a correlation layer.

tracts features with an addition operation from both WFD and MP (see Fig. 3(b)). The pixel value at level l is then

$$z^l_{i,j} = \sum_{x,y} M(z)^{l-1}_{i+x,j+y} k_1(x,y) + z^{l-1}_{i+x,j+y} k_3(x,y) \quad (3)$$

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.

3.1. 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). Assuming two pyramids (and using broadcasting of $\mathcal{A}$), then the feature map of the right pyramid at level $l+1$ takes the map of the left pyramid into account as in

$$\mathcal{F}^{l+1}_{right} = Conv(\mathcal{F}^l_{right} * \mathcal{A}(\mathcal{F}^{l+1}_{left}) + \mathcal{M}(\mathcal{F}^{l+1}_{left})) \quad (4)$$

3.2. Module Structure

The complete Residual Feature Pyramid Module with Repair Masks (RFPM) is constructed as follows (see Fig. 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 efficient fine-tuning which we will describe in Section 4.4.

(b) Repair left-right pathway: The feature map of the base pyramid extracts a repair mask, 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 Eqn. 1 with the feature maps:

$$\mathcal{F}^l_t = [\mathcal{F}^l_{left,t}, \mathcal{F}^l_{mid,t}, \mathcal{F}^l_{right,t}] \quad (5)$$

$$\mathcal{F}^l_{t+1} = [\mathcal{F}^l_{left,t+1}, \mathcal{F}^l_{mid,t+1}, \mathcal{F}^l_{right,t+1}] \quad (6)$$

Next, we will discuss how to integrate RFPM with two state-of-the-art optical flow methods: IRR-PWC [14] and RAFT [40].

3.3. 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. Fig. 4 illustrates the pyramids with repair masks generated at level 1 to 3. The base pyramid provides mask $\mathcal{A}(\mathcal{F}^l_{left})$ and $\mathcal{M}(\mathcal{F}^l_{left})$, which feed into the same level of the middle pyramid. This avoids unnecessary down- and upsampling. The middle pyramid generates masks $\mathcal{A}(\mathcal{F}^l_{mid})$ and $\mathcal{M}(\mathcal{F}^l_{mid})$, 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.

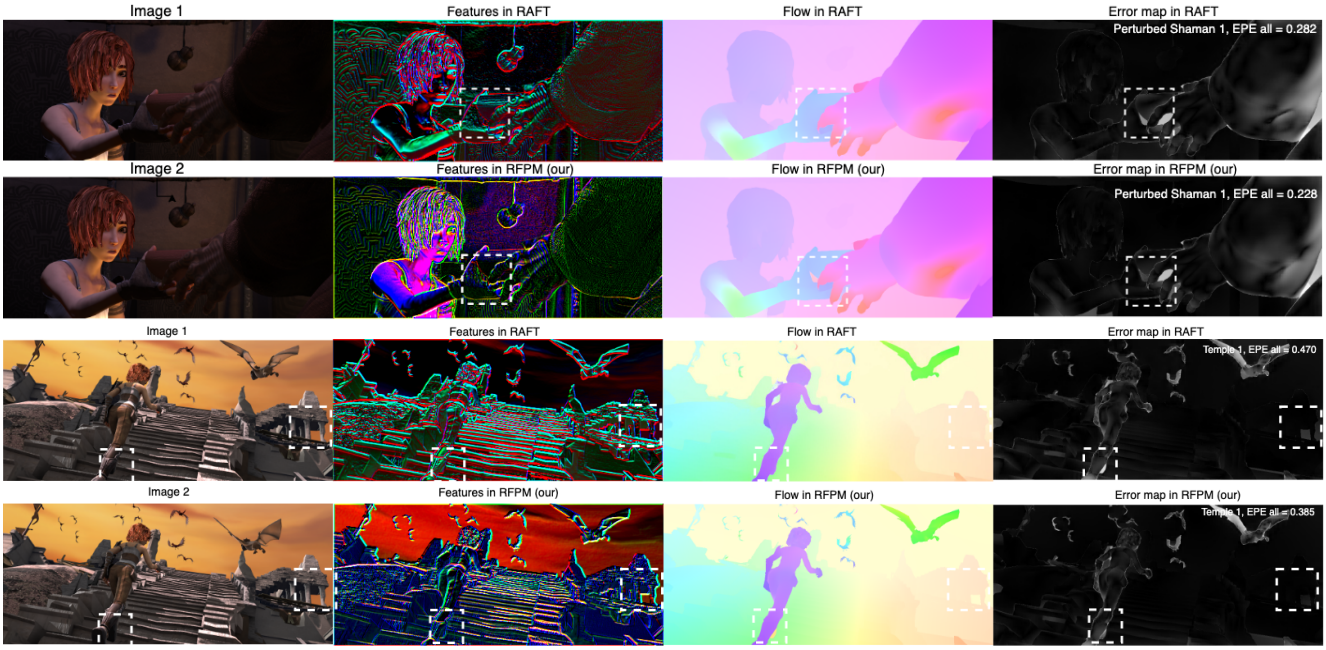


Figure 6. Visualizing the relation between feature map and predicted flow. Failure areas in RAFT are indicated with white squares. These areas have blurred edges in the feature pyramid. In our RFPM, the flow result correctly predicts the motion boundary because of the improved edges in the feature pyramid. RFPM significantly reduces the EPE errors.

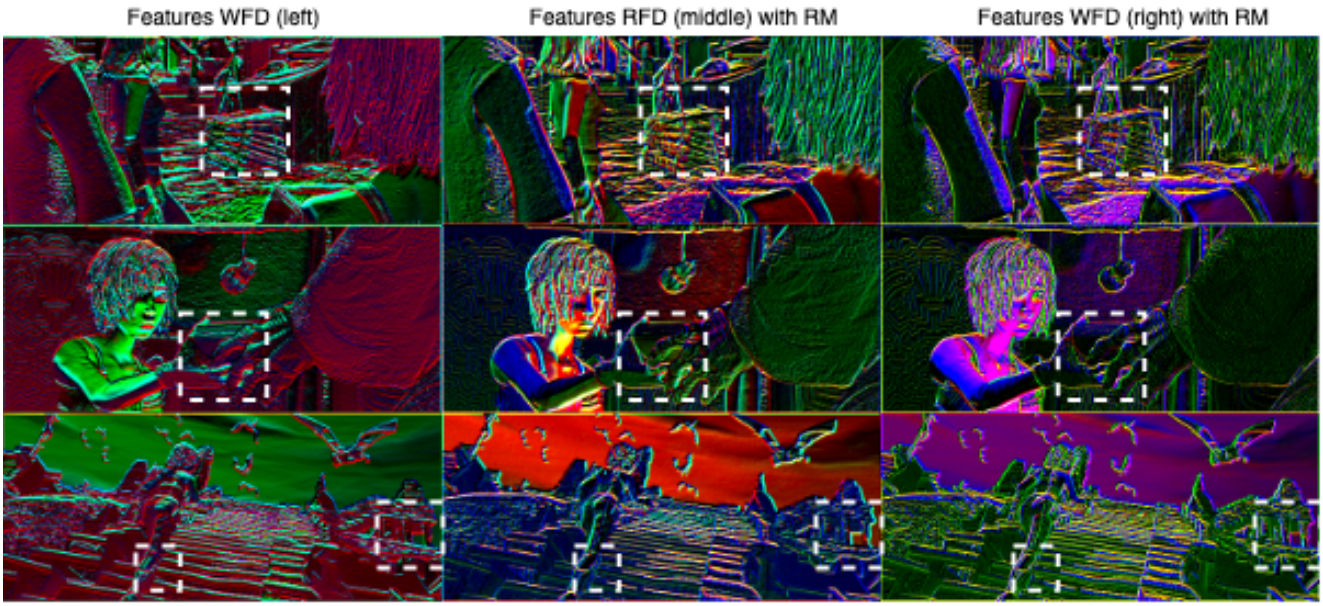


Figure 7. Visualization of feature pyramids. Comparison of the learned features from left pyramid using WFD, middle pyramid using RFD with repair mask (RM), and right pyramid using WFD with repair mask (RM).

3.4. 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. The total number of parameters in RFPM-RAFT is 7.5M.

4. Experimental Evaluation

We first detail the training for our implementation of RFPM-IRR-PWC and RFPM-RAFT before presenting results on MPI Sintel [4] and KITTI-2015 [28]. 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 efficient fine-tuning and data augmentation.

4.1. Training and Implementation

We follow the training configurations of IRR-PWC [14] and RAFT [40] for a fair comparison. We first train RFPM-IRR-PWC on the FlyingChairs_OCC [6] dataset (learning rate schedule schedule S_{short}) and then fine tune on FlyingThings3D [26] (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+ [39]. 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 [4], KITTI-2015 [28] and HD1K [18] for 200k iterations with a batch size of 3 (the same setting as described in PWC-Net+ [39]). 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 [29] 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.

4.2. Results

According to Table 1, RFPM-RAFT outperforms all published optical flow methods on the MPI Sintel [4] clean pass and KITTI-2015 [28]. Our methods achieve a 12% higher accuracy on Sintel, and 6% higher accuracy on KITTI compared with RAFT. Fig. 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, Fig. 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.

4.3. 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 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, 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.

Finally, we compare the different downsampling layers discussed in Section 3. Table 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.

4.4. Efficient Fine-tuning Approach

We present an efficient Fine-tuning 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

Method	Sintel (train)		KITTI-15 (train)		Sintel (test)		KITTI-15 (test)
	Clean	Final	F1-epe	F1-all	Clean	Final	F1-all(%)
SPyNet[30]	(3.17)	(4.32)	-	-	6.64	8.36	35.07
FlowNet2[16]	(1.45)	(2.19)	(2.36)	(8.88)	4.16	5.74	10.41
FlowNet3[17]	(1.47)	(2.12)	(1.79)	-	4.35	5.67	8.60
SelFlow[22]	(1.68)	(1.77)	(1.18)	-	3.75	4.26	8.42
PWC-Net[38]	(2.02)	(2.08)	(2.16)	(9.80)	4.39	5.04	9.60
PWC-Net+[39]	(1.71)	(2.34)	(1.47)	(7.59)	3.45	4.60	7.72
IRR-PWC[14]	(1.92)	(2.51)	(1.63)	(5.32)	3.84	4.58	7.65
LiteFlowNet2[12]	(1.30)	(1.62)	(1.33)	(4.32)	3.48	4.69	7.62
LiteFlowNet3[10]	(1.32)	(1.76)	(1.26)	(3.82)	2.99	4.45	7.34
HD3[44]	(1.70)	(1.17)	(1.31)	(4.10)	4.79	4.67	6.55
VCN[43]	(1.66)	(2.24)	(1.16)	(4.10)	2.81	4.40	6.30
MaskFlowNet[45]	-	-	-	-	2.52	4.17	6.10
RAFT[40]	(0.76)	(1.22)	(0.63)	(1.50)	1.94	3.18	5.10
RAFT(warm-start)[40]	(0.77)	(1.27)	-	-	1.61	**2.86**	-
RAFT-A*[37]	-	-	-	-	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	(0.61)	(1.05)	(0.60)	(1.41)	-	-	4.79
RFPM-RAFT(warm-start)	(0.68)	(1.12)	-	-	**1.41**	2.90	-

Table 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.[37] uses a different training dataset but does not change the basic RAFT architecture.

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 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. Repair mask levels and numbers.

Settings	Trained on Chairs		
	Chairs Testing	Sintel(clean) Training	Sintel(final) 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 4. Downsampling layer trade-off.

Methods	Schedule	KITTI-15 (test)	
		Fl-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 5. Efficient Fine-tuning Comparison for RAFT [40] 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 Fine-tuning schedule, which leads to our method trained as RFPMt-RAFT. The result shows that RFPMt-RAFT surpasses RAFT with a small training schedule.

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 Section 4.4). Table 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 6. MFPMt-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 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 6. Asymmetric Data Augmentation (ADA) during efficient fine-tuning. $^+$ is the average value evaluated by 3-fold cross validation.

5. Conclusions

Our analysis has revealed that the traditional feature pyramid is a major reason for 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 a residual feature downsampling that includes max pooling to preserve detail features. We use multiple pyramids in our module incorporating repair masks at some levels of the pyramids. Our RFPM can be easily incorporated 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. We have further proposed an efficient fine-tuning 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.

References

[1] 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.

[2] Michael J Black and Allan D Jepson. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *PAMI*, 18(10):972–986, 1996.

[3] Jean-Yves Bouguet. Pyramidal implementation of the affine Lucas Kanade feature tracker description of the algorithm. *Intel corporation*, 5(1-10):4, 2001.

[4] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*, pages 611–625, 2012.

[5] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *International Conference on Computer Vision*, pages 686–695, 2017.

[6] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision*, pages 2758–2766, 2015.

[7] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C. Berg. DSSD : Deconvolutional single shot detector. *CoRR*, abs/1701.06659, 2017.

[8] Berthold KP Horn and Brian G Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics, 1981.

[9] Yinlin Hu, Rui Song, and Yunsong Li. Efficient coarse-to-fine patchmatch for large displacement optical flow. In *Conference on Computer Vision and Pattern Recognition*, pages 5704–5712, 2016.

[10] Tak-Wai Hui and Chen Change Loy. LiteFlowNet3: Resolving Correspondence Ambiguity for More Accurate Optical Flow Estimation. In *European Conference on Computer Vision*, pages 169–184, 2020.

[11] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 8981–8989, 2018.

[12] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A Lightweight Optical Flow CNN - Revisiting Data Fidelity and Regularization. *PAMI*, 43(8):2555–2569, 2020.

[13] Junhwa Hur and Stefan Roth. Joint optical flow and temporally consistent semantic segmentation. In *European Conference on Computer Vision*, pages 163–177, 2016.

[14] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 5754–5763, 2019.

[15] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 7396–7405, 2020.

[16] 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 *Conference on Computer Vision and Pattern Recognition*, pages 2462–2470, 2017.

[17] 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 *European Conference on Computer Vision*, pages 614–630, 2018.

[18] 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 *Conference on Computer Vision and Pattern Recognition Wokshop*, pages 19–28, 2016.

[19] Tao Kong, Fuchun Sun, Chuanqi Tan, Huaping Liu, and Wenbing Huang. Deep feature pyramid reconfiguration for object detection. In *European Conference on Computer Vision*, pages 169–185, 2018.

[20] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *European Conference on Computer Vision*, pages 179–195, 2018.

[21] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Conference on Computer Vision and Pattern Recognition*, pages 936–944, 2017.

[22] Pengpeng Liu, Michael R. Lyu, Irwin King, and Jia Xu. Selflow: Self-supervised learning of optical flow. In *Conference on Computer Vision and Pattern Recognition*, pages 4571–4580, 2019.

[23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37, 2016.

[24] David Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[25] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on AI*, pages 674–679, 1981.

[26] 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 *Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016.

[27] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015.

[28] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, pages 60–76, 2018.

[29] 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. In *NeurIPS*, volume 32, pages 8024–8035, 2019.

[30] Anurag Ranjan and Michael J. Black. Optical flow estimation using a spatial pyramid network. In *Conference on Computer Vision and Pattern Recognition*, pages 2720–2729, 2017.

[31] Zhile Ren, Orazio Gallo, Deqing Sun, Ming-Hsuan Yang, Erik B Sudderth, and Jan Kautz. A fusion approach for multi-frame optical flow estimation. In *Winter Conference on Applications of Computer Vision*, pages 2077–2086, 2019.

[32] Rishav, René Schuster, Ramy Battrawy, Oliver Wasenmüller, and Didier Stricker. ResFPN: Residual skip connections in multi-resolution feature pyramid networks for accurate dense pixel matching. In *International Conference on Pattern Recognition*, pages 180–187. IEEE, 2020.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015.

[34] Laura Sevilla-Lara, Deqing Sun, Varun Jampani, and Michael J Black. Optical flow with semantic segmentation and localized layers. In *Conference on Computer Vision and Pattern Recognition*, pages 3889–3898, 2016.

[35] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, volume 27, pages 568–576, 2014.

[36] Deqing Sun, Stefan Roth, and Michael J Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *IJCV*, 106(2):115–137, 2014.

[37] 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. In *Conference on Computer Vision and Pattern Recognition*, pages 10093–10102, 2021.

[38] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018.

[39] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of CNNs for optical flow estimation. *PAMI*, 42(6):1408–1423, 2019.

[40] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *European Conference on Computer Vision*, pages 402–419, 2020.

[41] Prune Truong, Martin Danelljan, and Radu Timofte. GLU-Net: Global-local universal network for dense flow and correspondences. In *Conference on Computer Vision and Pattern Recognition*, pages 6258–6268, 2020.

[42] Andreas Veit, Michael J. Wilber, and Serge J. Belongie. Residual networks are exponential ensembles of relatively shallow networks. In *NeurIPS*, volume 30, pages 550–558, 2016.

[43] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*, volume 32, pages 794–805, 2019.

[44] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 6044–6053, 2019.

[45] Shengyu Zhao, Yilun Sheng, Yue Dong, Eric I-Chao Chang, and Yan Xu. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *Conference on Computer Vision and Pattern Recognition*, pages 6278–6287, 2020.

[46] C. Lawrence Zitnick, Nebojsa Jojic, and Sing Bing Kang. Consistent segmentation for optical flow estimation. In *International Conference on Computer Vision*, volume 2, pages 1308–1315, 2005.