Learning Optical Flow with Kernel Patch Attention

Ao Luo[1] Fan Yang[2] Xin Li[2] Shuaicheng Liu[3,1]*
[1]Megvii Technology [2]Group 42
[3]University of Electronic Science and Technology of China

Abstract

Optical flow is a fundamental method used for quantitative motion estimation on the image plane. In the deep learning era, most works treat it as a task of 'matching of features', learning to pull matched pixels as close as possible in feature space and vice versa. However, spatial affinity (smoothness constraint), another important component for motion understanding, has been largely overlooked. In this paper, we introduce a novel approach, called kernel patch attention (KPA), to better resolve the ambiguity in dense matching by explicitly taking the local context relations into consideration. Our KPA operates on each local patch, and learns to mine the context affinities for better inferring the flow fields. It can be plugged into contemporary optical flow architecture and empower the model to conduct comprehensive motion analysis with both feature similarities and spatial relations. On Sintel dataset, the proposed KPA-Flow achieves the best performance with EPE of 1.35 on clean pass and 2.36 on final pass, and it sets a new record of 4.60% in F1-all on KITTI-15 benchmark. Code is publicly available at https://github.com/megvii-research/KPAFlow.

1. Introduction

Optical flow is a fundamental technique used to characterize and quantify motion between two video frames, that facilitates a large number of real-life applications including visual tracking [39], video inpainting [45] and autonomous driving [8]. Recent deep learning based methods have made significant breakthroughs by **i)** learning discriminative features with advanced training paradigms (*e.g.,* reinforcement learning [1] and alternative learning [17, 34, 50]), **ii)** seeking/incorporating extra clues from the given scene [32] and **iii)** utilizing multi-frame information [19]. However, most contemporary works address the cross-image matching problem by learning and measuring feature similarities, overlooking another core component for motion understanding — the spatial relations which reveal the underlying affinities during motion.

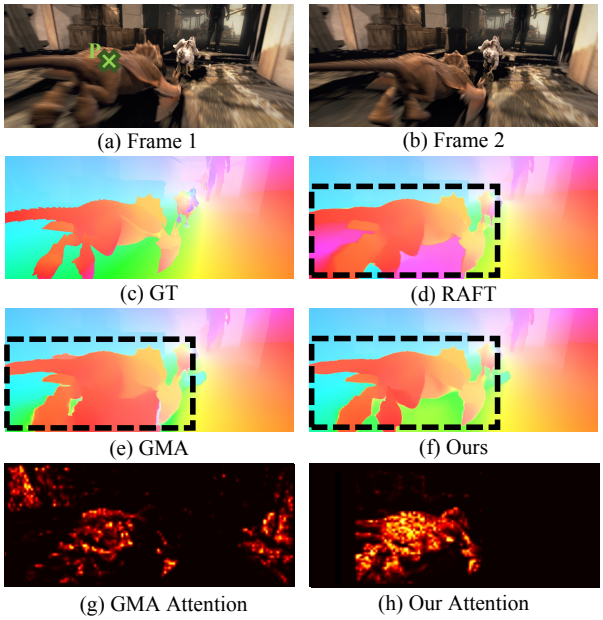


Figure 1. A challenging image pair with heavy motion blur from the final pass of Sintel `test` set. Unlike previous works (RAFT [36] and GMA [20]) suffering from ambiguous matching, our KPA-Flow is able to simultaneously mine the context and spatial affinities for better inferring the flow fields. The attention map of GMA contains many globally misleading clues while ours effectively avoids the noise. "P" indicates the attention point.

Traditional optical flow algorithms formulate the dense matching as an energy minimization problem, explicitly considering both feature similarities and spatial smoothness [5, 6, 13]. Although the hand-crafted features used by those traditional methods suffer from scale and shape variations, the smoothness constraint, to a great extent, helps to fix errors by taking the spatial relations into consideration. In the deep learning era, most contemporary optical flow methods largely focus on addressing the feature-matching similarities (*e.g.,* using 4D correlation volumes [36]), expecting that the extracted features by deep models are dis-

*Corresponding author

criminative enough for estimating flow fields. Indeed, recent advances in deep learning have provided significant improvements in the ability to extract discriminative/invariant features for matching in a data-driven manner, largely improving the reliability of optical flow. However, as optical flow itself is a complicated task, solely counting on 'perfect' features is still sub-optimal, making the deep model become fragile when dealing with some challenging cases (see Fig. 1). We believe that explicitly considering both feature similarities and spatial relations would help to build a powerful deep model that can understand motion at a higher level.

Using deep learning techniques to capture and model local relations is challenging. First, because the motion of object(s) appears locally in the visual scene, effectively capturing the local relations (affinities) is vital for motion analysis. However, common relation modeling approaches, such as non-local operations [41] and graph reasoning [9, 27], focus on solving the global or long-range dependencies. How to reasonably model the local relations is still under-explored. Second, the local relations should be end-to-end learnable, so as to best mine pairwise relationships described by the affinity value. Third, optical flow estimation is a per-pixel matching task, thus it is difficult to efficiently obtain the pixel-wise relations. Last but not the least, the designed module is expected to be easily plugged into contemporary optical flow architectures.

Targeting the above challenges and enabling a deeper understanding of motion, we introduce a novel kernel patch attention (KPA) that explicitly focuses on local relations, and captures motion affinities to better infer the optical flow. To address the first two challenges, we draw inspirations from the non-local operation [41] to learn the pixel-wise relations by comparisons. But differently, our KPA is formulated as a kernel-based operator, which conducts relation reasoning within each local patch and achieves the smoothed motion features in a sliding window manner. Importantly, our KPA operates on both context and motion features; That is, for each location, KPA mines relations over its local surroundings using context features and takes advantage of the mined information to update the motion features. For the third challenge, our KPA is designed to work in a patch-wise manner by treating a group of pixels as 'centres' and computing local relations simultaneously so as to reduce the computational complexity. Finally, we formulate all relation reasoning and feature updating operations with differentiable operations, which are implemented by using neural networks. Thus, it can be easily plugged into existing optical flow architecture and can be trained with other components together.

Using 4D all-pairs correlation volume (CV), recurrent refinement scheme [36] and our proposed KPA, we design a powerful deep flow model, called KPA-Flow, to explicitly handle both feature similarities and spatial affinities within a unified framework for optical flow. Our KPA-Flow is able to infer reliable flows in different challenging cases (see Fig. 1), achieving top-ranked performance on both Sintel and KITTI benchmarks. The contributions of this work are summarized as:

- **A fully-differentiable approach for explicitly conducting the smooth constraint.** To the best of our knowledge, we are the first to explicitly handle the local relations for optical flow based on the context and spatial affinities. We present a kernel-based function to effectively mine local relations and use the mined information to infer the flow fields.
- **A novel operator for comprehensive optical flow estimation.** We propose kernel patch attention (KPA) operator with a specific patch-based sliding window strategy, which is simple yet effective for reliable motion understanding.
- **State-of-the-art results on widely-used benchmarks.** The fully-equipped KPA-Flow can reliable infer the optical flow in challenging scenes, which sets new records on both Sintel and KITTI benchmarks with limited extra computational cost.

2. Related Work

Optical Flow Estimation. Optical flow is the task of estimating per-pixel motion between video frames. In the early stage, researchers [4–6, 13] treat this task as an energy minimization problem, leveraging both feature similarities (*i.e.,* data term) and motion smoothness (*i.e.,* smoothness term) to infer the flow fields. Although the handcrafted features are weak in describing high-level context information, the smoothness term helps to largely reduce ambiguity in dense matching with the carefully designed optimization objective. However, considering the overall accuracy and speed, traditional methods can only handle some simple cases reliably. In the deep learning era, early attempts [2, 43] simply treat the prediction of flow fields as a mapping process — directly mapping two given frames to the corresponding flow fields, and expect that the matching can be implicitly learned by neural networks. Thanks to the power of data, these approaches achieve better performance than traditional approaches. However, without an explicit matching process, the performance gains achieved by those approaches are still limited. Recent deep learning approaches design different neural modules to explicit pixel-wise-relation modeling [15, 30, 35, 46] to largely improve the reliability of optical flow estimation. However, as motion itself would introduce a lot of clutters and create unpredictable variations, those approaches are still unreliable for ambiguous regions, *e.g.,* occlusions and object-

boundary regions. To alleviate the above difficulties, advanced strategies, such as the joint representation learning [49], feature-driven flow regularization [16], iterative refinement [15,17,35] and holistic motion reasoning [20,27] are applied to further improve the reliability of feature matching. Moreover, the smoothness constraint is also considered in the loss function by existing models [24, 47], with a desire to enforce the model to learn consistent features for creating smooth flow fields. However, as these approaches cannot take the spatial information into consideration for handling the smoothness issue, the results are still sub-optimal. Unlike all existing approaches, we propose to explicitly mine the local relations and conduct flow smoothness for better flow estimation.

Self-Attention Operation. Self-attention was originally designed for machine translation by computing relations within the sequence [38]. In computer vision, the self-attention operation has been widely used to mine global relationship/context of images or videos [41, 42, 51]. For example, in [41], non-local operations are used to capture long-range dependencies for video classification, object detection and segmentation, and pose estimation. Wang *et. al* [42] integrates non-local operations with U-Net [31] to conduct more reliable media image analysis. Luo *et. al* [26] leverage the mined low-dimensional supportive knowledge to enhance the semantic feature with an attention approach. To reduce the computational complexity, Zhu *et. al* [51] propose the asymmetric non-local operation to explore the long-range spatial relevance among features. Similarly, Huang *et. al* [14] propose to adaptively captures contextual information on the criss-cross path. For optical flow estimation, GMA [20] mines the global context to resolve ambiguities caused by occlusions via self-attention. Different from these works, we focus on how to effectively and efficiently capture local relations to refine the motion features, which has been never explored before.

3. Methodology

3.1. Problem Formulation

Given a pair of input consecutive images, *i.e.*, source image $\mathbf{I}_1$ and target image $\mathbf{I}_2$, the task of optical flow estimation is to predict a dense displacement field between them. Deep learning based flow networks commonly employ an encoder-decoder pipeline to first extract context feature $\mathbf{f}_c$ and motion feature $\mathbf{f}_m$, and then make flow prediction based on the combination of these two features in a recurrent/coarse-to-fine manner.

In our approach, we propose a simple yet effective operator, named kernel patch attention (KPA), and plug it into the feature fusion procedure, which is formulated as $\hat{\mathbf{f}}_m = \mathcal{F}_{\texttt{KPA}}(\mathbf{f}_c, \mathbf{f}_m)$. Specifically, KPA operator explicitly takes scene context and spatial affinities into consideration, and exploits the learned feature relations to better infer the flow fields. After the motion feature refinement, context and updated motion features are fed into decoder modules for several runs of residual flow estimation.

3.2. Kernel Patch Attention for Optical Flow

The architecture of the proposed kernel patch attention (KPA) for optical flow is depicted in Fig. 2. We follow prior works [20, 21, 36] to develop our model in a recurrent refinement scheme. Specifically, given an input image pair $(\mathbf{I}_1, \mathbf{I}_2)$, we first employ two residual-based encoders [12] to extract a feature pair $(\mathbf{f}_1, \mathbf{f}_2)$ and context feature $\mathbf{f}_c$, respectively. Then, 4D correlation volumes can be built upon all vector pairs within the feature $\mathbf{f}_1$ and $\mathbf{f}_2$ in four scales. Based on a pre-defined searching window, we capture the motion feature $\mathbf{f}_m$ by applying a motion encoder on the segmented matching costs.

3.2.1 Kernel Function Based Definition

Given the context feature $\mathbf{f}_c$ from $\mathrm{F}_c \in \mathbb{R}^{c\times h\times w}$ and motion feature $\mathbf{f}_m$ from $\mathrm{F}_m \in \mathbb{R}^{c\times h\times w}$ extracted by encoder networks, we design KPA to extract feature relations from scene context, and then treat the mined relations as the smooth constraint to guide the learning of motion features. Specifically, inspired by image and point convolutions [23,37], we formulate KPA as a kernel-based operator, which is generally given by:

$$(\mathrm{F} * \mathcal{K})(x) = \sum_{x_i \in \mathcal{N}_x} \mathcal{K}_{(x_i - x)}(\tilde{\mathbf{f}}_c, \mathbf{f}_c)\rho(\mathbf{f}_m), \qquad (1)$$

where x_i indicates the position i in 2D grid space of neighborhoods $\mathcal{N}_x$ (also termed as *kernel window*) around x. $\mathcal{K}_{(x_i-x)}(\cdot)$ is a kernel function taking the neighbor points as inputs, and $\rho(\cdot)$ is a linear projection that maps the input motion feature to an embedding space. $\tilde{\mathbf{f}}_c$ denotes the context feature in the center patch window of $\mathcal{N}_x$ (please refer to Fig. 3 and Sec. 3.2.2 for more details about the kernel and its center patch windows). From the perspective of attention mechanism, $\tilde{\mathbf{f}}_c$ is used to produce the query vectors, and the corresponding key and value vectors can be embedded from $\mathbf{f}_c$ and $\mathbf{f}_m$ within $\mathcal{N}_x$.

The kernel function is the core component in our KPA, which can be commonly defined as:

$$\mathcal{K}_{(t_i)}(\tilde{\mathbf{f}}_c, \mathbf{f}_c) = \mathcal{S}(t_i, \hat{x}_n)\mathcal{W}_{(t_i)}(\tilde{\mathbf{f}}_c, \mathbf{f}_c), \qquad (2)$$

where $t_i = x_i - x$, and $\hat{x}_n$ means the coordinate of position n in kernel region. Then, $\mathcal{S}(t_i, \hat{x}_n)$ stands for a scale function, which produces a scale map based on the Euclidean distance between t_i and $\hat{x}_n$ in grid space. The weight function $\mathcal{W}_{(t_i)}(\tilde{\mathbf{f}}_c, \mathbf{f}_c)$ is used to produce kernel weights.

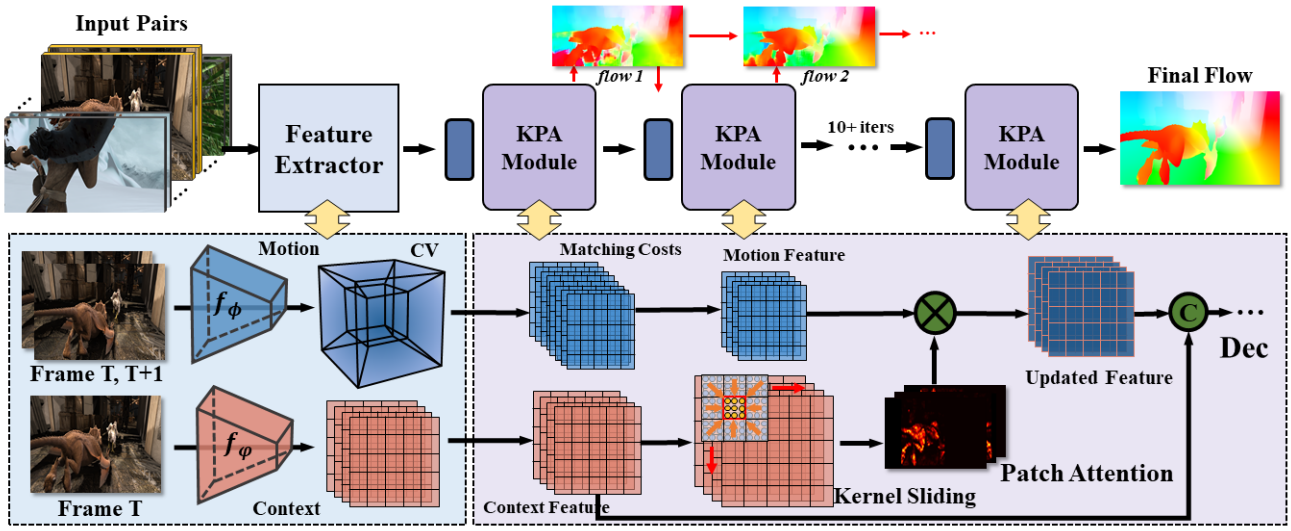


Figure 2. Architecture of the proposed kernel patch attention for optical flow (KPA-Flow). "C" indicates concatenation and "×" denotes multiplication. "CV" means the 4D correlation volumes. "Dec" denotes flow decoding process. Best viewed in color.

Similar to widely-used 2D convolution kernels, we devise $\mathcal{W}_{(t_i)}(\cdot)$ to provide different weights for each position of t_i. However, different from image convolution kernels, which simply define a learnable matrix W_i, we take a step further by taking the context relations in kernel region into consideration. Specifically, given the context features $\tilde{\mathbf{f}}_c$ and $\mathbf{f}_c$, we utilize the embedded Gaussian with normalization [41] to measure the correlations between all pairs among them, which can be formulated as:

$$\mathcal{W}_{(t_i)}(\tilde{\mathbf{f}}_c, \mathbf{f}_c) = \frac{\exp(\theta(\tilde{\mathbf{f}}_c^{\,u})^T \phi(\mathbf{f}_c^{\,v})))}{\sum_{\forall v} \exp(\theta(\tilde{\mathbf{f}}_c^{\,u})^T \phi(\mathbf{f}_c^{\,v}))}, \quad (3)$$

where $\theta(\mathbf{f}) = \mathrm{W}_\theta \mathbf{f}$ and $\phi(\mathbf{f}) = \mathrm{W}_\phi \mathbf{f}$ are two linear projections to preform feature embeddings. The dimension of the produced adaptive weights are $N \times K$, where $N = h \times w$ denotes the pixel-wise spatial dimension of the entire feature map, and $K = k \times k$ indicates the size of kernel window.

The other component of kernel function is the scale function $\mathcal{S}(t_i, \hat{x}_n)$. In image convolutions [10, 23], the scalar is simply set to 1 for all positions in kernel. One reason is that the learnable weights from $\mathcal{W}_{(t_i)}(\cdot)$ can be trained to fit for the weighted summation in Eqn. 1, which inherently contains the scale balance among all points. Besides, the widely-used networks like VGG [33] and ResNet [12] typically tend to stack several convolutions with kernel size of 1×1 and 3×3, thus the distances between t_i and $\hat{x}_n$ should be small and cannot provide additional distinctive information. On the contrary, we hope to ensure that a single KPA operator is capable of obtaining a large receptive field so as to cover instance-level information. Thus it should be equipped with a scale factor mask based on the distance between t_i and $\hat{x}_n$ for spatial constraints. Specifically, we adopt the linear correlation to formulate the scale function as:

$$\mathcal{S}(t_i, \hat{x}_n) = \max(0, l + \mathrm{abs}(a(\|t_i - \hat{x}_n\| - \frac{k}{2}))), \quad (4)$$

where k denotes the size of kernel window, l is the base scalar and a is a learnable parameter indicating the influence of the point-wise distance. This function produces a scalar map, in which the value for each point is in reverse proportion to the Euclidean distance $\|t_i - \hat{x}_n\|$.

3.2.2 Kernel Patch Attention with Sliding Window

Previous work has shown that performing flow feature refinement with global motion constraints is an effective strategy for flow estimation [20]. However, as shown in Fig. 1, directly applying the non-local method based on context feature may involve some unreliable guidance for motion feature refinement.

Here, we introduce KPA to enable the optical flow model to have a suitable receptive field for smooth constraint of motion feature, and avoid the misleading information from long-range context. Specifically, we devise a sliding window based scheme for motion feature smoothing. Given a feature map $\mathbf{f} \in \mathbb{R}^{c \times h \times w}$, we first split it into non-overlapping patches, and each patch with size of $p \times p$ is regarded as a feature group. So all feature vectors within the feature map is divided into $\bar{h} \times \bar{w}$ patch windows, where $\bar{h} = \lceil \frac{h}{p} \rfloor$ and $\bar{w} = \lceil \frac{w}{p} \rfloor$. We treat each group of feature vectors within the patch window as a base element $\bar{\mathbf{f}}$ with size of $c \times p \times p$. Like the basic settings that image convolutions [23] commonly use kernel size of odd numbers, we define the size of kernel window as $k = \bar{k} \times p$, where

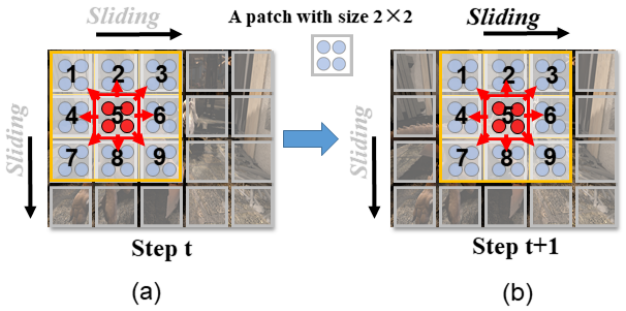


Figure 3. A toy example of KPA with sliding window. The orange box denotes the kernel window, and the red one indicates the corresponding center patch window. Best viewed in color.

$\bar{k} \in \{1, 3, ..., 2i+1\}$. Thus the patch-wise kernel shape is denoted as $(\bar{k} \times \bar{k})$.

A toy example of sliding window implementation of our KPA is illustrated in Fig. 3, where the patch size $p = 2$, the kernel shape $\bar{k} \times \bar{k}$ is 3×3, $\bar{h}$ and $\bar{w}$ are set to 4 and 5, respectively. The left subfigure depicts a sliding kernel window $\mathcal{N}_x$ (the orange one) working on a splitted feature map in step t. Here we treat each patch as a base element, and use e_j, where $j \in \{1, 2, ..., 9\}$), to denote each element (patch) in the kernel window. As can be seen, the core element is e_5 in step t, where the feature representations in the center patch window (the red one) will be updated. In practice, we first employ the weight function $\mathcal{W}_{(t_i)}(\cdot)$ on $\mathbf{f}_c{}^{e_5}$ and $\mathbf{f}_c{}^{E}$ (where E denotes all e_j in $\mathcal{N}_x$) to produce adaptive weights, as in Eqn. 3. Specifically, there are two kinds of dot-product similarity measurements involved in this process, *i.e.*, intra- and inter-patch similarities between some patch windows. For instance, the feature vectors in e_5 can perform self-attention-like similarity measurement with themselves, and simultaneously obtain correlations with the features in eight other patch windows by cross-attention-like interactions. Then the kernel $\mathcal{K}_{(t_i)}$ is built with Eqn. 2. Finally, we perform the kernel patch attention on motion features with Eqn. 1, as $\mathbf{f'}_m{}^{e_5} = \sum_{t_i \in E} \mathcal{K}_{(t_i)} \rho(\mathbf{f}_m)$. The output motion feature $\hat{\mathbf{f}}_m{}^{e_5}$ is finally produced by a residual operation as $\hat{\mathbf{f}}_m{}^{e_5} = \mathbf{f}_m{}^{e_5} + \alpha\, \mathbf{f'}_m{}^{e_5}$, where α indicates a learnable parameter that is initialized as 0 and gradually performs a weighted summation.

The right subfigure in Fig. 3 illustrates that the kernel window slides to the next region in step $t+1$. Since we treat each patch as the base element in our approach, the sliding stride of kernel window should be equivalent to the patch size p. Besides, to ensure that the size of feature map can be divisible by p, and the center patch windows are able to cover all positions within the entire feature map by sliding, we apply zero-padding for feature dilation. Therefore, all patches within the motion feature $\mathbf{f}_m$ can be refined for smoothing by our KPA operator. It is worth noting that each sliding step in KPA operation does not rely on the results of others. Thus, like image convolution, the whole operator can be processed in parallel for end-to-end training.

Discussion: Comparison with Convolutions. The function of image convolution can be generally formulated as:

$$(x * w)(t) = \sum_{a \in \mathcal{N}_t} w(t-a)x(a), \tag{5}$$

where a denotes all pixels in kernel region $\mathcal{N}_t$, and $w(\cdot)$ indicates the kernel function. Although our approach shares the same pattern with image convolution as the kernel-based operation, the inherent formulations are totally different. First, image convolutions simply define the kernel function $w(\cdot)$ as learnable weights W, while our KPA defines it with a scale function to provide learnable scalar and an adaptive weight function based on context relations. During inference, our weight function $\mathcal{W}_{(t_i)}(\tilde{\mathbf{f}}_c, \mathbf{f}_c)$ is able to provide the **dynamic** kernel map, in which each weight varying with the corresponding context feature. In contrast, W is a static weight and shared in all sliding positions after training. Second, image convolutions require more parameters to work on both channel dimension and spatial dimension. Nevertheless, our KPA only needs linear projections on channel dimension, which helps to significantly reduce the computational overhead when the kernel region is large. Please refer to Sec. 4.3 for quantitative comparisons with convolutional operators in flow network.

Discussion: Comparison with NL-like Operators. The proposed KPA operator is capable of capturing the region-based scene context to guide the learning of motion features. Therefore, compared to global non-local-like operators [20], KPA consumes less computational overhead. For instance, given a feature with dimension $c \times h \times w$, the computational complexity of our KPA and NL are:

$$\Omega(\text{KPA}) = 4Nc^2 + 2cNK, \tag{6}$$

$$\Omega(\text{NL}) = 4Nc^2 + 2cN^2, \tag{7}$$

where $N = h \times w$ denotes the spatial dimension, and $K = k \times k$ indicates the dimension of kernel window. The feature size is generally larger than kernel size by a large margin, *i.e.*, $N \gg K$. Besides, the shape of attention map in our KPA is $N \times K$, which is smaller than NL attentions of $N \times N$, indicating less GPU memory cost. Moreover, the NL attention maps may involve many global noises from unconstrained scene context, which are not fit for motion guidance (see Fig. 1 and Fig. 5).

3.3. Network Instantiation

Following the success of prior works [20, 21, 36], we develop our KPA-Flow network based on RAFT. Specifically,

Training Data	Method	Sintel (val)		KITTI-15 (val)		Sintel (test)		KITTI-15 (test)
		Clean	Final	EPE	F1-all	Clean	Final	F1-all
C + T	PWC-Net [34]	2.55	3.93	10.35	33.7	-	-	-
	FlowNet2 [18]	2.02	3.54	10.08	30.0	3.96	6.02	-
	RAFT [36]	1.43	2.71	5.04	17.4	-	-	-
	SCV [21]	1.29	2.95	6.80	19.3	-	-	-
	GMA [20]	1.30	2.74	4.69	17.1	-	-	-
	AGFlow [27]	1.31	2.69	4.82	17.0	-	-	-
	SeperableFlow [48]	1.30	**2.59**	4.60	**15.9**	-	-	-
	Flow1D [44]	1.98	3.27	6.69	22.95	-	-	-
	KPA-Flow (ours)	**1.28**	2.68	**4.46**	**15.9**	-	-	-
C + T + S + K (+ H)	IRR-PWC [17]	(1.92)	(2.51)	(1.63)	(5.3)	3.84	4.58	7.65
	VCN [46]	(1.66)	(2.24)	(1.16)	(4.1)	2.81	4.40	6.30
	MaskFlowNet [50]	-	-	-	-	2.52	4.17	6.10
	ScopeFlow [3]	-	-	-	-	3.59	4.10	6.82
	DICL [40]	(1.11)	(1.60)	(1.02)	(3.6)	2.12	3.44	6.31
	RAFT [36]	(0.77)	(1.27)	(0.63)	(1.5)	1.61	2.86	5.10
	SCV [21]	(0.86)	(1.75)	(0.75)	(2.1)	1.77	3.88	6.17
	GMA [20]	(0.62)	(1.06)	(0.57)	(1.2)	1.39	2.47	5.15
	AGFlow [27]	(0.65)	(1.07)	(0.58)	(1.2)	1.43	2.47	4.89
	SeperableFlow [48]	(0.69)	(1.10)	(0.69)	(1.6)	1.50	2.67	4.64
	Flow1D [44]	(0.84)	(1.25)	-	(1.6)	2.24	3.81	6.27
	KPA-Flow (ours)	**(0.60)**	**(1.02)**	**(0.52)**	**(1.1)**	**1.35**	**2.36**	**4.60**

Table 1. Quantitative comparison with state-of-the-arts in EPE and F1-all metrics. We follow prior works [21,36,40] to compare our results with all published works on two standard benchmarks. Models pre-trained on FlyingChairs(C) and FlyingThing(T) are compared in "C + T" for generalization ability evaluation. "+ S + K (+ H)" denotes the training data combining Sintel, KITTI and HD1K(optional [36]). The best results are marked in **bold** for better comparison.

we build the motion encoder based on six residual blocks with three stridings, and the channel dimension of output feature map is set to 256. The context encoder shares the same structure with the motion encoder. We construct the 4D correlation volume in four scales, *i.e.,* by pooling with kernel sizes of $\{1, 2, 4, 8\}$, for all feature pairs.

In the residual updating scheme, a motion encoder is applied to capture motion feature $\mathbf{f}_m$ from matching costs. The channel dimensions c of context feature and motion feature are both set to 128, and the spatial dimension is $\frac{1}{8}$ of input shape. Then context and motion features are fed into the proposed KPA operator for feature refinement. We set the recurrent iteration $N = 12$ for training.

4. Experimental Results

4.1. Implementation Details

We implement KPA-Flow and conduct experiments based on PyTorch toolbox. In our model, we empirically set the kernel shape $\bar{k} \times \bar{k}$ to 3×3. The patch sizes p are set to 19 and 9 on Sintel and KITTI, respectively.

During training, we follow prior work [36] to adopt AdamW optimizer with one-cycle learning rate policy, and conduct model pretraining on synthetic data as the standard optical flow training procedure. The model is pre-trained on FlyingChairs [11] for 120k iterations and then on FlyingThings [28] for 180k iterations. After that, we fine-tune the model on the combined data from Sintel [7], KITTI-2015 [29], and HD1K [22] for 180k iterations, and then submit the flow prediction to Sintel server [7] for online evaluation. Finally, additional 60k iterations of finetuning are performed on KITTI-2015 [29] for KITTI online evaluation. We use 4 GeForce GTX 2080Ti GPUs to train all models, and adopt a single one for evaluation and time testing. The batch sizes are set to 8 and 1 for training and testing, respectively.

4.2. Comparison with State-of-the-Arts

Results on Sintel. The comparisons of generalization ability are shown in "C + T" of Tab. 1. The proposed KPA-Flow achieves an EPE score of 1.28 on clean pass of Sintel dataset, which is better than recent arts, including SCV [21], GMA [20] and SeperableFlow [48], and outstrips the well-known RAFT [36] by 10.5% (1.43 $\rightarrow$ 1.28). On final pass, it obtains an average EPE of 2.68, surpassing the previous state-of-the-art methods SCV and GMA by 9.2% (2.95 $\rightarrow$ 2.68) and 2.2% (2.74 $\rightarrow$ 2.68), respectively. Besides, our approach significantly outperforms recent work Flow1D [44] by 34.3% (1.98 $\rightarrow$ 1.30) in clean pass and 18.0% (3.27 $\rightarrow$ 2.68) in final. The results demonstrate the promising cross dataset generalization of our model.

On Sintel test set, we utilize the warm-start strategy

Figure 4. Qualitative comparisons with RAFT [36] on Sintel `test` set. All results are provided by the official website of Sintel. Quantitative results are shown in corresponding Error map with EPE metric for better comparison (the lower the better).

and submit the predicted results to the official server for online evaluation like prior works [36, 48]. Our KPA-Flow achieves a score of 1.35 in EPE on Sintel clean pass, which surpasses top-ranked methods SeperableFlow [48] and GMA [20] by 10.0% (1.50 → 1.35) and 2.9% (1.39 → 1.35), respectively. Besides, it obtains EPE = 2.36 on final pass, significantly outperforming recent arts SCV [21] and Flow1D [44] by a large margin (39.2% and 38.1%). It is worth noting that our KPA-Flow ranks 1st on both clean and final passes of Sintel benchmark among all published and unpublished approaches at the time of submission.

Fig. 4 provides some qualitative comparisons with the well-known RAFT [36] on the challenging final pass of Sintel dataset. The results demonstrate that our model is able to fully exploit scene context to effectively perform motion refinement with the proposed KPA operator, which helps to reduce the matching uncertainty and produce more accurate and smooth flow fields.

Results on KITTI. The results of our approach on KITTI-15 dataset are also shown in Tab. 1. As we can see, KPA-Flow achieves an average EPE of 4.46 and F1-all score of 15.9% on KITTI-15 validation set, which largely surpasses the recent arts SCV [21] and Flow1D [44] by 29.1% (6.80 → 4.82) in EPE, 11.9% (19.3 → 17.0) in F1-all, and 29.1% (6.80 → 4.82) in EPE, 11.9% (19.3 → 17.0) in F1-all, respectively.

Besides, our approach sets a new records of 4.60% in F1-all on KITTI-15 benchmark, which outperforms top-ranked methods GMA [20] and RAFT [36] by 10.7% (5.15 → 4.60) and 12.2% (5.10 → 4.60), respectively. Some qualitative comparisons with state-of-the-art works on KITTI dataset are illustrated in the supplementary material.

4.3. Ablation Analysis

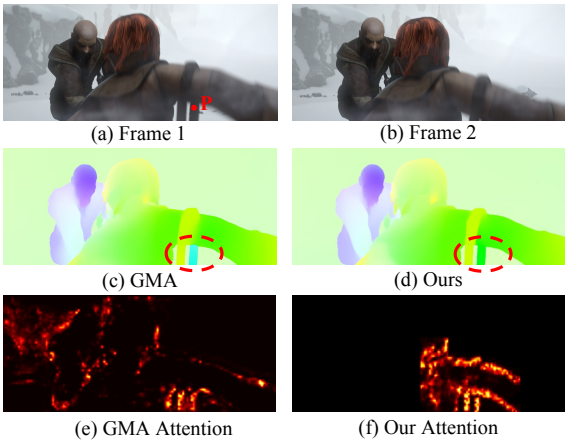
Comparison with Convolutional Operators. We first compare the proposed KPA operator with the widely-used convolutional blocks, dense and dilated convolutions [17, 34], in flow network. Similar to our KPA, we plug the convolutional blocks into the motion feature refinement process. As shown in Tab. 2, dense and dilated convolutions slightly improve the flow accuracy with relatively heavy computation complexity. In contrast, our KPA-Flow achieves better performance, yet only needs additional 0.54 M parameters. This demonstrates that simply applying convolutions is not fit for motion enhancement, while the proposed KPA with dynamic kernel map based on the context and spatial affinities for motion constraint is a simple yet effective approach for flow estimation.

Method	Param ↓ (Δ)	KITTI-15 (val)		Sintel (val)	
		EPE	F1-all	clean	final
RAFT [36]	5.26 (-)	5.04	17.4	1.43	2.71
+ DenseConv [35]	8.6 (+3.34)	4.99	17.1	1.39	2.73
+ DilatedConv [35]	6.11 (+0.85)	5.08	17.3	1.42	2.76
+ SwinTrans [25]	6.15 (+0.89)	4.82	17.1	1.38	2.70
+ GMA [20]	5.80 (+0.54)	4.69	17.1	1.30	2.74
+ **KPA**	5.80 (+0.54)	**4.46**	**15.9**	**1.28**	**2.68**

Table 2. Quantitative comparisons with related methods (refer to Sec. 4.3 for more details). We build KPA-Flow both on RAFT. All methods are trained on C + T for fair comparison.

Comparison with Swin Transformer. Swin Transformer [25] can be directly regarded as a self-attention method for motion feature enhancement, which relies on a few blocks to perform the shifting strategy and inevitably requires more parameters and computational complexity. In contrast, the proposed KPA contains intra- and inter-patch similarity measurements between patch windows in a single operator, which can leverage the kernel shape to change the size of operating region and needs no additional parameters. As shown in Tab. 2, requiring fewer parameters, the proposed KPA outperforms Swin Transformer in two datasets.

Comparison with GMA. We compare our KPA with GMA on the same baseline mode for fair comparison. As can be seen in Tab. 2, with the same parameters, our KPA outperforms GMA [20] by 4.9% (4.69 → 4.46) in EPE and 7.0% (17.1 → 15.9) in F1-all on KITTI. Besides, a qualitative comparison is shown in Fig. 5. As we can see, a strip on the arm moves out of the figure, so the motion must be inferred



(a) Frame 1 (b) Frame 2

(c) GMA (d) Ours

(e) GMA Attention (f) Our Attention

Figure 5. A challenging image pair with fog from the final pass of Sintel `test` set. The strip moves out of the field-of-view. "P" indicates the attention point. (Best viewed in color.)

based on surrounding relevant points. GMA adopts the non-local attention, taking global motions with all feature similarities into consideration, while our KPA focuses on the local region for motion reasoning. The results demonstrate that the region kernel based attention approach is better than global attention for optical flow. This is because the non-local-like attention method takes the whole scene context into consideration, which also contains the long-range similarities in appearance yet improper for motion guidance.

Ablation for Patch Size. We empirically analyze the computational cost and the corresponding performance gain by changing the patch size in our KPA-Flow. As we can see in Tab. 3, when larger patch size are set ($5 \rightarrow 7 \rightarrow 9$), the performance are gradually increased from 4.72 in EPE and 16.8 in F1-all on KITTI validation set to 4.46 and 15.9, respectively. However, if we further enlarge the patch size ($9 \rightarrow 11 \rightarrow 13$), the flow accuracy slightly decreases and the computational overhead is largely increased by around 2 times. This is because the large patch size leads to a large kernel region, where some long-range context affinities bring more unreliable guidance for flow refinement.

Ablation for Kernel Shape. In Tab. 3, we provide the comparisons between different kernel shapes with computational complexity. Following the basic settings in image convolution, we set the kernel shape $k \times k$ to 1×1, 3×3 and 5×5 for comparisons. As we can see, the larger kernel shape directly leads to more computational cost. However, the best flow accuracy is achieved in the middle setting. This is because the large kernel shape also results in some unreliable clues for motion refinement.

Ablation for Feature Summation. We also evaluate the influence of weighted summation in Tab. 3. As we can see, α sum is a cost-effective method, which surpasses other ones yet with negligible computation cost.

Ablation for Scale Function. We evaluate the influence of the scale function $\mathcal{S}(\cdot)$ in Tab. 3. As can be seen, the proposed scale function is better than a static mask with values of 1 for all positions. The results demonstrate that the constraint of spatial affinities can effectively boost the flow performance.

	Settings	FLOPs (G) ↓	KITTI-15 (val)	
			EPE	F1-all
Patch Size p	5	4.3	4.72	16.8
	7	7.3	4.58	16.3
	9	11.2	4.46	15.9
	11	16.1	4.49	16.1
	13	22.1	4.51	16.3
Kernel Shape $\bar{k} \times \bar{k}$	1×1	2.3	4.91	17.2
	3×3	11.2	4.46	15.9
	5×5	29.0	4.57	16.2
Feature Update	sum	11.2	4.59	16.1
	α sum	11.2	4.46	15.9
	replace	11.2	4.71	16.6
Scale Function	On	11.2	4.46	15.9
	Off	11.2	4.58	16.2

Table 3. Ablation analysis for different settings of our KPA-Flow. All settings are trained on C + T for fair comparison. Underline indicates the default settings in our model.

5. limitation

A limitation of our approach is that the fixed kernel shapes are inflexible for handling some challenging cases, like two similar objects on different motions entangling with each other. Under this circumstance, both the context and spatial affinities cannot provide valid information for motion guidance. A possible solution for this problem is to learn deformable kernels for adaptive motion reasoning. We will work on this direction in future work.

6. Conclusion

This work proposes a new model to mine region-based context relations and leverage the learned information to better infer the flow fields. To this goal, we design a learnable kernel patch attention (KPA) module to reason over the local context for obtaining spatial affinities, and use them to refine the motion features. Therefore, our fully-equipped KPA-Flow is able to explicitly deal with both feature similarities and spatial consistencies, which sets new records on both Sintel and KITTI benchmarks. We expect our work to inspire more future efforts towards this promising direction.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (NSFC) No.61872067 and No.61720106004, and in part by the National Key R&D Plan of the Ministry of Science and Technology No.2020AAA0104400.

References

[1] Artemij Amiranashvili, Alexey Dosovitskiy, Vladlen Koltun, and Thomas Brox. Motion perception in reinforcement learning with dynamic objects. In *Conference on Robot Learning*, 2018. 1

[2] Min Bai, Wenjie Luo, Kaustav Kundu, and Raquel Urtasun. Exploiting semantic information and deep matching for optical flow. In *ECCV*, 2016. 2

[3] A. Bar-Haim and L. Wolf. Scopeflow: Dynamic scene scoping for optical flow. In *CVPR*, 2020. 6

[4] Michael J Black and Padmanabhan Anandan. A framework for the robust estimation of optical flow. In *ICCV*, 1993. 2

[5] T. Brox, Andrés Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004. 1, 2

[6] Andrés Bruhn, J. Weickert, and C. Schnörr. Lucas/kanade meets horn/schunck: combining local and global optic flow methods. *IJCV*, 2005. 1, 2

[7] Daniel Butler, Jonas Wulff, Garrett Stanley, and Michael Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 6

[8] Linda Capito, Umit Ozguner, and Keith Redmill. Optical flow based visual potential field for autonomous driving. In *IEEE Intelligent Vehicles Symposium*, 2020. 1

[9] Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Yan Shuicheng, Jiashi Feng, and Yannis Kalantidis. Graph-based global reasoning networks. In *CVPR*, 2019. 2

[10] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 4

[11] A. Dosovitskiy, P. Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, V. Golkov, P. V. D. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 6

[12] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 4

[13] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 1981. 1, 2

[14] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*, 2019. 3

[15] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, 2018. 2, 3

[16] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow cnn—revisiting data fidelity and regularization. *TPAMI*, 2020. 3

[17] Junhwa Hur and S. Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, 2019. 1, 3, 6, 7

[18] Eddy Ilg, N. Mayer, Tonmoy Saikia, Margret Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 6

[19] Joel Janai, Fatma Guney, Anurag Ranjan, Michael Black, and Andreas Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *ECCV*, 2018. 1

[20] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *ICCV*, 2021. 1, 3, 4, 5, 6, 7

[21] Shihao Jiang, Yao Lu, Hongdong Li, and R. Hartley. Learning optical flow from a few matches. In *CVPR*, 2021. 3, 5, 6, 7

[22] D. Kondermann, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrulis, Alexander Brock, Burkhard Güssefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, C. Brenner, and B. Jähne. The hci benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *CVPRW*, 2016. 6

[23] Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. Convolutional networks and applications in vision. In *IEEE international symposium on circuits and systems*, 2010. 3, 4

[24] 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. 3

[25] 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. 7

[26] Ao Luo, Fan Yang, Xin Li, Yuezun Li, Zhicheng Jiao, Hong Cheng, and Siwei Lyu. Robust scene parsing by mining supportive knowledge from dataset. *TNNLS*, 2021. 3

[27] Ao Luo, Fan Yang, Kunming Luo, Xin Li, Haoqiang Fan, and Shuaicheng Liu. Learning optical flow with adaptive graph reasoning. In *AAAI*, 2022. 2, 3, 6

[28] N. Mayer, Eddy Ilg, Philip Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 6

[29] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015. 6

[30] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017. 2

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCIA*, 2015. 3

[32] Laura Sevilla-Lara, Deqing Sun, Varun Jampani, and Michael J Black. Optical flow with semantic segmentation and localized layers. In *CVPR*, 2016. 1

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2014. 4

[34] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 1, 6, 7

[35] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 2, 3, 7

[36] Zachary Teed and Jun Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 1, 2, 3, 5, 6, 7

[37] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J

Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 3

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3

[39] Mikko Vihlman and Arto Visala. Optical flow in deep visual tracking. In *AAAI*, 2020. 1

[40] Jianyuan Wang, Yiran Zhong, Yuchao Dai, K. Zhang, Pan Ji, and Hongdong Li. Displacement-invariant matching cost learning for accurate optical flow estimation. In *NeurIPS*, 2020. 6

[41] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 2, 3, 4

[42] Zhengyang Wang, Na Zou, Dinggang Shen, and Shuiwang Ji. Non-local u-nets for biomedical image segmentation. In *AAAI*, 2020. 3

[43] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *ICCV*, 2013. 2

[44] Haofei Xu, Jiaolong Yang, Jianfei Cai, Juyong Zhang, and Xin Tong. High-resolution optical flow from 1d attention and correlation. In *ICCV*, 2021. 6, 7

[45] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *CVPR*, 2019. 1

[46] Gengshan Yang and D. Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*, 2019. 2, 6

[47] 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 *ECCV*, 2016. 3

[48] Feihu Zhang, Oliver J Woodford, Victor Adrian Prisacariu, and Philip HS Torr. Separable flow: Learning motion cost volumes for optical flow estimation. In *ICCV*, 2021. 6, 7

[49] Chengqian Zhao, Cheng Feng, Dengwang Li, and Shuo Li. Of-msrn: optical flow-auxiliary multi-task regression network for direct quantitative measurement, segmentation and motion estimation. In *AAAI*, 2020. 3

[50] Shengyu Zhao, Yilun Sheng, Yue Dong, E. Chang, and Yan Xu. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *CVPR*, 2020. 1, 6

[51] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In *ICCV*, 2019. 3