

# MaskFlowNet: Asymmetric Feature Matching with Learnable Occlusion Mask

Shengyu Zhao\* Yilun Sheng\* Yue Dong Eric I-Chao Chang Yan Xu†

## Abstract

Feature warping is a core technique in optical flow estimation; however, the ambiguity caused by occluded areas during warping is a major problem that remains unsolved. In this paper, we propose an asymmetric occlusion-aware feature matching module, which can learn a rough occlusion mask that filters useless (occluded) areas immediately after feature warping without any explicit supervision. The proposed module can be easily integrated into end-to-end network architectures and enjoys performance gains while introducing negligible computational cost. The learned occlusion mask can be further fed into a subsequent network cascade with dual feature pyramids with which we achieve state-of-the-art performance. At the time of submission, our method, called MaskFlowNet, surpasses all published optical flow methods on the MPI Sintel, KITTI 2012 and 2015 benchmarks. Code is available at <https://github.com/microsoft/MaskFlowNet>.

## 1. Introduction

Optical flow estimation is a core problem in computer vision and a fundamental building block in many real-world applications [2, 22, 29]. Recent development towards fast, accurate optical flow estimation has witnessed great progress of learning-based methods using a principled network design — feature pyramid, warping, and cost volume

\*Equal contribution. †Corresponding author (xuyan04@gmail.com). Shengyu Zhao, Yilun Sheng, and Yue Dong are with Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China. Shengyu Zhao, Yilun Sheng, Eric I-Chao Chang, and Yan Xu are with Microsoft Research, Beijing 100080, China. Yan Xu is with State Key Laboratory of Software Development Environment and Key Laboratory of Biomechanics and Mechanobiology of Ministry of Education and Research Institute of Beihang University in Shenzhen and Beijing Advanced Innovation Center for Biomedical Engineering, Beihang University, Beijing 100191, China.

This work is supported by the National Science and Technology Major Project of the Ministry of Science and Technology in China under Grant 2017YFC0110903, Microsoft Research under the eHealth program, the National Natural Science Foundation in China under Grant 81771910, the Fundamental Research Funds for the Central Universities of China under Grant SKLSDE-2017ZX-08 from the State Key Laboratory of Software Development Environment in Beihang University in China, the 111 Project in China under Grant B13003.

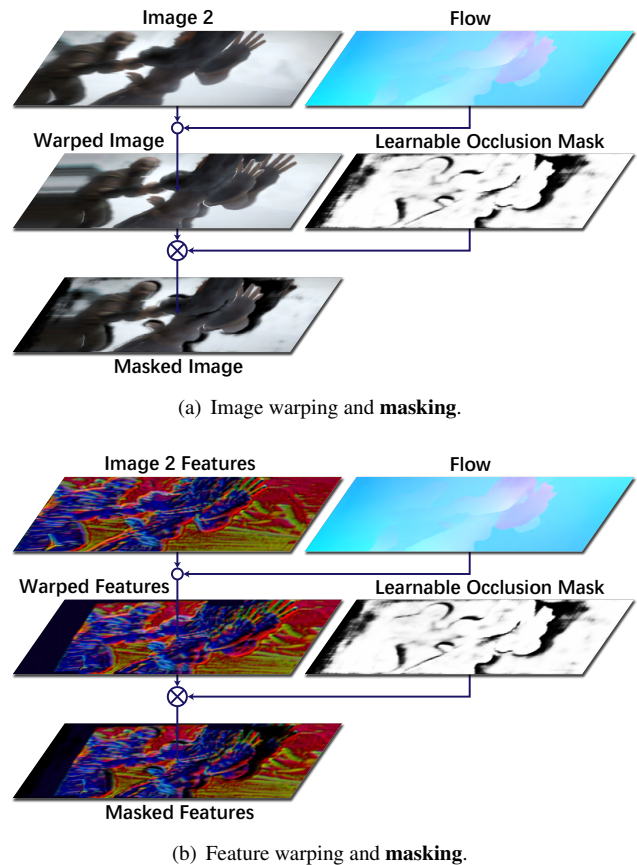


Figure 1. **Motivation of the learnable occlusion mask.** (a) Image warping induces ambiguity in the occluded areas (see the doubled hands). (b) The same problem exists in the feature warping process. Such areas can be masked without any explicit supervision.

— proposed by PWC-Net [32] and LiteFlowNet [11], and used in many follow-up works [12, 13, 18, 24, 31]. Feature warping effectively resolves the long-range matching problem between the extracted feature maps for the subsequent cost volume computation. However, we observe that a major problem of the warping operation is that it introduces unreliable information in the presence of occlusions. As shown in Fig. 1, the warped image as well as the warped feature map can be even “doubled” at the occluded areas (also called the *ghosting effect*). It remains unclear that

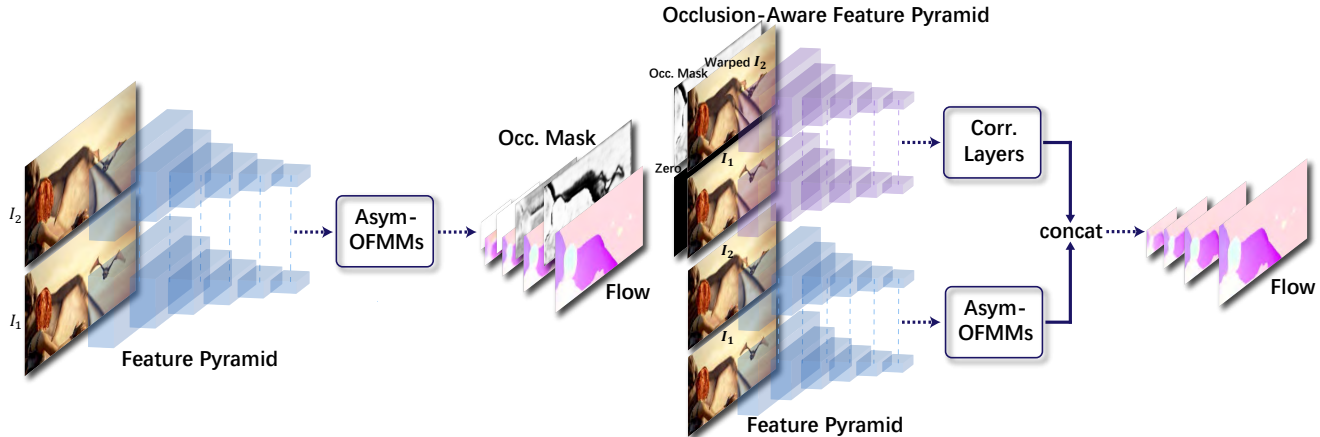


Figure 2. **The overall architecture of MaskFlowNet.** MaskFlowNet consists of two stages — the first end-to-end network named MaskFlowNet-S (left), and the second cascaded network (right) that aims to perform refinements using dual pyramids. Dashed lines across pyramids represent shared weights. MaskFlowNet generally utilizes the proposed AsymOFMM whenever possible. The learnable occlusion mask is coarse-to-fine predicted and fed into the new occlusion-aware feature pyramid. See §4 for the network details.

whether the source image would be mismatched to such areas yet raises a natural question: *are they really distinguishable without being supervised of occlusions?*

We answer this question positively by showing that the network can indeed learn to *mask* such areas without any explicit supervision. Rather than enforcing the network to distinguish useful parts from those confusing information, we propose to apply a multiplicative learnable occlusion mask immediately on the warped features (see Fig. 1). We can see that there is a clear distinction between the black and white areas in the learned occlusion mask, indicating that there exists solid gradient propagation. The masked image (features) has much cleaner semantics, which could potentially facilitate the subsequent cost volume processing.

The masking process interprets how those areas can be distinguished in a clear way. While previous works commonly believe that the feature symmetry is crucial for the cost volume processing, we in contrast demonstrate that the network further benefits from a simple *asymmetric* design despite the explicit masking. The combined *asymmetric occlusion-aware feature matching module* (AsymOFMM) can be easily integrated into end-to-end network architectures and achieves significant performance gains.

We demonstrate how the proposed AsymOFMM would contribute to the overall performance using a two-stage architecture named *MaskFlowNet* (see Fig. 2). MaskFlowNet is trained on standard optical flow datasets (not using the occlusion ground truth), and predicts the optical flow together with a rough occlusion mask in a single forward pass. At the time of submission, MaskFlowNet surpasses all published optical flow methods on the MPI Sintel (on both clean and final pass), KITTI 2012 and 2015 benchmarks while using only *two-frame* inputs with no additional assumption.

## 2. Related Work

**Optical Flow Estimation.** Conventional approaches formulate optical flow estimation as an energy minimization problem based on brightness constancy and spatial smoothness since [10] with many follow-up improvements [3, 21, 36]. Estimating optical flow in a coarse-to-fine manner achieves better performance since it better solves large displacements [4, 37]. Later works propose to use CNN extractors for feature matching [1, 39]. However, their high accuracy is at the cost of huge computation, rendering those kinds of methods impractical in real-time settings.

An important breakthrough of deep learning techniques in optical flow estimation is made by FlowNet [8], which proposes to train end-to-end CNNs on a synthetic dataset and first achieves a promising performance. Although they only investigate two types of simple CNNs (FlowNetS and FlowNetC), the correlation layer in FlowNetC turns out to be a key component in the modern architectures. FlowNet2 [14] explores a better training schedule and makes significant improvements by stacking multiple CNNs which are stage-wise trained after fixing the previous ones.

SpyNet [27] explores a light-weight network architecture using feature pyramid and warping, but the learned features are not correlated so it can only achieve a comparable performance to FlowNetS. PWC-Net [32] and LiteFlowNet [11] present a compact design using feature pyramid, warping, and cost volume, and achieve remarkable performance over most conventional methods while preserving high efficiency, and they further make some slight improvements in the later versions [12, 31]. VCN [40] recently exploits the high-dimensional invariance during cost volume processing and achieves state-of-the-art performance.

Note that this paper focuses on the feature matching process prior to the correlation layer, which is independent of the improvement made by VCN. To our knowledge, none of those works realizes that an asymmetric design of the feature matching process can achieve better performance.

**Occlusions and Optical Flow.** Occlusions and optical flow are closely related. Optical flow methods such as FlowNet and FlowNet2 can be easily extended to joint optical flow and occlusion estimation with slight modifications as proposed in [15, 19]. IRR-PWC [13] presents an iterative residual refinement approach with joint occlusion estimation using bilateral refinement. However, all those methods can only explicitly learn from the ground-truth occlusions, which require additional efforts on the training labels that limit their applicability [13, 19].

Unsupervised or self-supervised learning of optical flow is another promising direction. Handling occlusions is clearly a vital aspect in such setting, since the brightness error does not make sense at occluded pixels. Some initial works show the feasibility of the unsupervised learning guided by the photometric loss [17, 28]. Later works realize that the occluded pixels should be excluded from the loss computation [20, 35]. Occlusions also facilitate multi-frame estimation [16, 18, 24]. However, the only occlusion estimation approach used by those kinds of methods is the forward-backward consistency [33] that requires bi-directional flow estimation, which limits its flexibility and could lead to noisy predictions. We would like to remark that our promising approach can jointly estimate occlusions without any explicit supervision in a single forward pass, which we expect can be helpful to future unsupervised or self-supervised learning methods.

**Occlusion-Aware Techniques in Other Applications.** Occlusions commonly exist in object detection and might affect the performance of standard approaches in some scenarios, e.g., crowded pedestrian detection [41]. Recent works propose to explicitly learn a spatial attention mask that highlights the foreground area for occluded pedestrian detection [26, 42], which requires additional supervising information. Occlusions in face recognition is also a major problem which can be addressed by the guided mask learning [30]. Our work is also related to the attention mechanism in computer vision [34], which addresses a different problem of capturing *pixel-wise* long-range dependencies. None of those works realizes a global attention mask can be learned to filter occluded areas with no explicit supervision.

### 3. Occlusion-Aware Feature Matching

Given an image pair  $I_1$  (the source image) and  $I_2$  (the target image), the task is to estimate the flow displacement

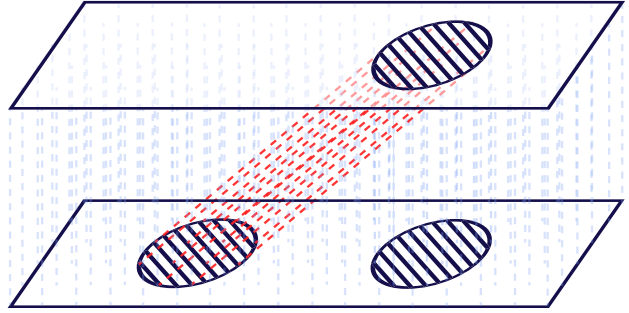


Figure 3. **A simplified case of occlusions.** The top image is warped to the bottom image according to the illustrated flow. The foreground object (shaded area) generates a large displacement (tracked by the red lines) while the background stays still (tracked by the blue lines). However, a copy of the foreground object still stays at the occluded area after warping.

ment  $\phi$ , representing the correspondence between  $I_1(x)$  and  $I_2(x + \phi(x))$ . Image warping is the process of constructing

$$(\phi \circ I_2)(x) \triangleq I_2(x + \phi(x)) \quad (1)$$

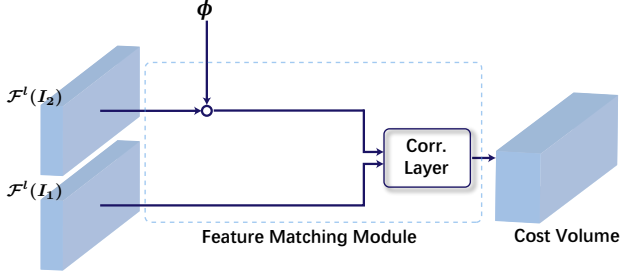
using the estimated displacement  $\phi$ , and ideally we have  $I_1(x) \approx (\phi \circ I_2)(x)$  at all non-occluded pixels  $x$ . This operation is differentiable w.r.t. both inputs because non-integral points can be dealt with bilinear interpolation. Feature warping is similarly defined by replacing  $I_2$  in Eq. (1) with the extracted feature map.

Feature warping followed by a correlation layer is the common practice to compute the cost volume for non-local feature matching in recent works [11, 12, 13, 18, 24, 31, 32]. The feature extractor of an image is a pyramid of convolutional layers, which are proposed to be *symmetric* for  $I_1$  and  $I_2$  in the sense that they share the *same* convolution kernels. The cost volume at pyramid level  $l$  can be formulated as

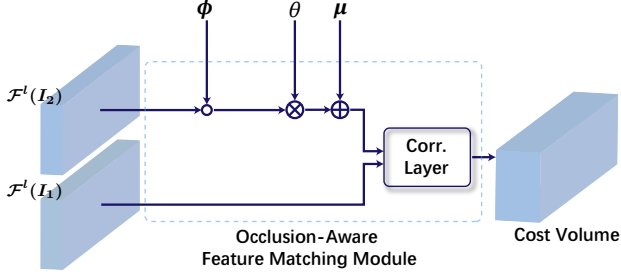
$$c(\mathcal{F}^l(I_1), \phi \circ \mathcal{F}^l(I_2)), \quad (2)$$

where  $\mathcal{F}^l$  denotes the shared feature extractor for level  $l$ , and  $\phi$  denotes the flow displacement predicted by the previous level.  $c$  represents the correlation layer that computes the element-wise dot product between the two feature maps within a maximum displacement. Fig. 4(a) illustrates this process, which we call a feature matching module (FMM).

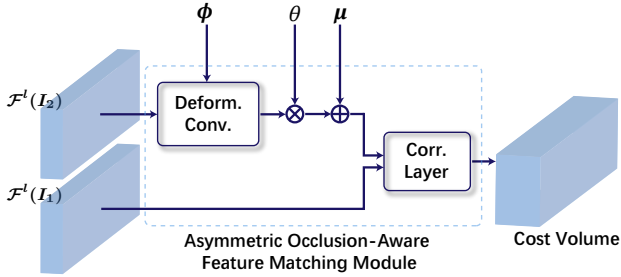
We observe that a major consequence caused by the warping operation is the ambiguity in the presence of occlusions. Fig. 3 illustrates a simplified example, where the foreground object has a large movement while the background stays still. During warping, a copy of the foreground object is revealed in the occluded background area. Such areas in the warped image (features) are useless and cause confusion to the subsequent flow inference.



(a) Feature matching module (FMM) as used in PWC-Net [32].



(b) Occlusion-aware feature matching module (OFMM).



(c) Asymmetric occlusion-aware feature matching module (AsymOFMM).

Figure 4. **Feature matching modules.** The figures illustrate the proposed AsymOFMM and OFMM in comparison with the FMM. FMM warps the target feature maps with the flow displacement  $\phi$ . OFMM introduces the multiplicative learnable occlusion mask  $\theta$  followed by the additive trade-off term  $\mu$ . AsymOFMM further replaces the warping operation by a deformable convolution.

#### Occlusion-Aware Feature Matching Module (OFMM).

The occlusion-aware feature matching module incorporates a *learnable occlusion mask* that filters useless information immediately after feature warping (see Fig. 4(b)). The warped feature tensor of shape  $(B, C, H, W)$  is element-wise multiplied by the (soft) learnable occlusion mask  $\theta$  of shape  $(B, 1, H, W)$  with broadcasting and then added with an additional feature tensor  $\mu$  of shape  $(B, C, H, W)$ . The resulting cost volume at level  $l$  is formulated as

$$c(\mathcal{F}^l(I_1), (\phi \circ \mathcal{F}^l(I_2)) \otimes \theta \oplus \mu). \quad (3)$$

$\theta$  is assumed to be within the range of  $[0, 1]$ .  $\mu$  acts as a trade-off term that facilitates the learning of occlusions, as

it provides extra information at the masked areas.

OFMM learns to mask the occluded areas simply because it realizes they are useless comparing to the trade-off term, even if there is no explicit supervision to the occlusions at all. Although the OFMM itself might not contribute significantly to the performance, it can learn a rough occlusion mask at negligible cost, which can be further fed into the new occlusion-aware feature pyramid (see §4).

#### Asymmetric Occlusion-Aware Feature Matching Module (AsymOFMM).

We suggest that an asymmetric design of the feature extraction layers consistently gains the performance. Intuitively, the warping operation induces ambiguity to the occluded areas and breaks the symmetry of the feature matching process, so an asymmetric design might be helpful to conquer this divergence.

Based on the OFMM, we introduce an extra convolutional layer prior to the warping operation, which is asymmetrically drawn on the feature extraction process of only  $I_2$ . In practice, we replace the extra convolutional layer and the warping operation by a *deformable convolution*. In the general setting of deformable convolutional networks [7], different locations in each of the convolution kernels are associated with different offsets, but here we introduce a specialized setting where each convolution kernel is warped in parallel according to the corresponding flow displacement at center. The deformable convolution slightly differs from the initial design since it reverts the order of convolution and (bilinear) interpolation, which is proved to be better in the experiments. As illustrated in Fig. 4(c), the resulting cost volume can be formulated as

$$c(\mathcal{F}^l(I_1), \mathcal{D}^l(\mathcal{F}^l(I_2), \phi) \otimes \theta \oplus \mu), \quad (4)$$

where  $\mathcal{D}^l(\cdot, \phi)$  denotes the deformable convolution layer at level  $l$  using the displacement  $\phi$ .

## 4. MaskFlowNet

The overall architecture of the proposed MaskFlowNet is illustrated in Fig. 2. MaskFlowNet consists of two cascaded subnetworks. The first stage, named MaskFlowNet-S, generally inherits the network architecture from PWC-Net [32], but replaces the feature matching modules (FMMs) by the proposed AsymOFMMs.

MaskFlowNet-S first generates a 6-level shared feature pyramid as PWC-Net, and then makes predictions from level 6 to 2 in a coarse-to-fine manner. The final predictions at level 2 are 4-time upsampled to level 0. Fig. 5 illustrates the network details at each level  $l$  (modifications needed at level 6 and level 2). The previous level is responsible for providing  $\phi^{l+1}$ ,  $\theta^{l+1}$ , and  $\mu^{l+1}$ , which are then upsampled and fed into the AsymOFMM.  $\phi^{l+1}$ ,  $\theta^{l+1}$  are upsampled using bilinear interpolation;  $\mu^{l+1}$  is upsampled



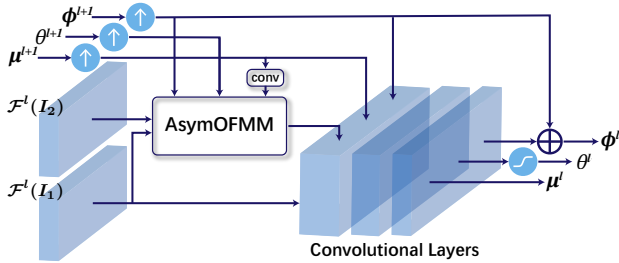


Figure 5. **Network connections at each level.** This figure adopts to the first stage (MaskFlowNet-S). The learnable occlusion mask is generated through the sigmoid activation from the previous level, and then upsampled and fed into the AsymOFMM.

using a deconvolutional layer (of 16 channels), followed by a convolutional layer to match the channels of the pyramid feature extractor. All convolutional layers have a kernel size of 3; all deconvolutional layers have a kernel size of 4. The feature matching module at level 6 is simply a correlation layer since there is no initial flow for warping. The maximum displacement of the correlation layers is kept to be 4. The resulting cost volume is concatenated with  $\mathcal{F}^l(I_1)$ , the upsampled displacement, and the upsampled features, and then passed through 5 densely connected convolutional layers as PWC-Net. The final layer predicts the flow displacement  $\phi^l$  with residual link from the previous flow estimation, the occlusion mask  $\theta^l$  after the sigmoid activation, and the features  $\mu^l$  passing to the next level. Level 2 only predicts the flow displacement, ended with the context network (as PWC-Net) that produces the final flow prediction.

**Occlusion-Aware Feature Pyramid.** The learned occlusion mask, concatenated with the warped image, is fed into the occlusion-aware feature pyramid for the subsequent flow refinement. The occlusion mask is subtracted by 0.5 before concatenation; a zero mask is concatenated with  $I_1$  for symmetricity. The occlusion-aware pyramid extracts features for the concatenated images (both with 4 channels) with shared convolutional layers as usual. We suggest that the occlusion mask facilitates the feature representation of the warped image, given the vast existence of occluded areas during warping.

**Cascaded Flow Inference with Dual Pyramids.** We propose to cascade the network by utilizing *dual* feature pyramids. The occlusion-aware feature pyramid provides abundant information about the warped image from the previous flow estimation for refinement, but it cannot feedback to the new coarse-to-fine flow predictions. Hence, we suggest that the network can still gain complementary information from the original feature pyramid.

The network architecture of this stage is similar to the

former stage except some modifications and the incorporation of the new occlusion-aware feature pyramid (see Fig. 2). The original feature pyramid is directly placed into this stage using the same parameters. The maximum displacement of all correlation layers in this stage is set to 2 since we expect it to perform mainly refinements. Correlation layers are used as the feature matching modules for the occlusion-aware feature pyramid since there is no need for feature warping. At each level, the resulting cost volumes from dual feature pyramids are concatenated together with the other terms including an extra flow predicted from the previous stage at the current level. As suggested in FlowNet2 [14], we fix all the parameters in the first stage (MaskFlowNet-S) when training the whole MaskFlowNet.

## 5. Experiments

### 5.1. Implementation

We implement a Python-trainable code framework using MXNet [6]. Mostly, we follow the training configurations as suggested in PWC-Net+ [31, 32] and FlowNet2 [14]. More details can be found in the supplementary material.

**Training Schedule.** MaskFlowNet-S is first trained on FlyingChairs [8] and then tuned on FlyingThings3D [19] following the same schedule as PWC-Net [32]. When fine-tuning on Sintel, we use the same batch configuration (2 from Sintel, 1 from KITTI 2015, and 1 from HD1K) and a longer training schedule (1000k iterations) referring to the cyclic learning rate proposed by PWC-Net+ [31]. When training the whole MaskFlowNet, we fix all the parameters in MaskFlowNet-S as suggested in FlowNet2 [14] and follow again the same schedule except that it is shorter on FlyingChairs (800k iterations). For submission to KITTI, we fine-tune our model on the combination of KITTI 2012 and 2015 datasets based on the tuned checkpoint on Sintel, while the input images are resized to  $1280 \times 576$  (before augmentation and cropping) since the decreased aspect ratio better balances the vertical and horizontal displacement.

**Data Augmentation.** We implement geometric and chromatic augmentations referring to the implementation of FlowNet [8] and IRR-PWC [13]. We suppress the degree of augmentations when fine-tuning on KITTI as suggested. For sparse ground-truth flow in KITTI, the augmented flow is weighted averaged based on the interpolated valid mask.

**Training Loss.** We follow the multi-scale end-point error (EPE) loss when training on FlyingChairs and FlyingThings3D, and its robust version on Sintel and KITTI, using the same parameters as suggested in PWC-Net+ [31]. Weight decay is disabled since we find it of little help.

Method	Time (s)	Sintel <i>clean</i>		Sintel <i>final</i>		KITTI 2012		KITTI 2015	
		AEPE <i>train</i>	AEPE <i>test</i>	AEPE <i>train</i>	AEPE <i>test</i>	AEPE <i>train</i>	AEPE <i>test</i>	Fl-all <i>train</i>	Fl-all <i>test</i>
FlowNetS [8]	0.01	3.66	6.16	4.76	7.22	6.07	7.6	-	-
FlowNetC [8]	0.05	3.57	6.08	5.25	7.88	7.31	-	-	-
FlowNet2 [14]	0.12	<b>2.02</b>	3.96	<b>3.14</b>	6.02	4.09	1.8	28.20%	11.48%
SpyNet [27]	0.16	4.12	6.64	5.57	8.36	9.12	4.1	-	35.07%
MR-Flow [38]	480	-	2.53	-	5.38	-	-	-	12.19%
LiteFlowNet [11]	0.09	2.48	4.54	4.04	5.38	4.00	1.6	28.50%	9.38%
LiteFlowNet2 [12]	0.04	2.24	3.45	3.78	4.90	3.42	1.4	25.88%	7.74%
PWC-Net [32]	0.03	2.55	3.86	3.93	5.13	4.14	1.7	33.67%	9.60%
PWC-Net+ [31]	0.03	-	3.45	-	4.60	-	1.5	-	7.90%
SelFlow [18]	0.09	-	3.74	-	4.26	-	1.5	-	8.42%
VCN [40]	0.18	2.21	2.81	3.62	4.40	-	-	25.1%	6.30%
MaskFlownet-S	0.03	2.33	2.77	3.72	4.38	3.21	<b>1.1</b>	23.58%	6.81%
MaskFlownet	0.06	2.25	<b>2.52</b>	3.61	<b>4.17</b>	<b>2.94</b>	<b>1.1</b>	<b>23.14%</b>	<b>6.11%</b>

Table 1. **Results of different methods on the MPI Sintel, KITTI 2012 and 2015 benchmarks.** Values listed in the *train* columns only consider those models which are *not* trained on the corresponding training set and thus comparable. AEPE: average end-point error over all valid pixels. Fl-all: percentage of optical flow outliers over all valid pixels. Running times are referred to [32]; our time is measured on an NVIDIA TITAN Xp GPU, which is comparable to the NVIDIA TITAN X used by [32].

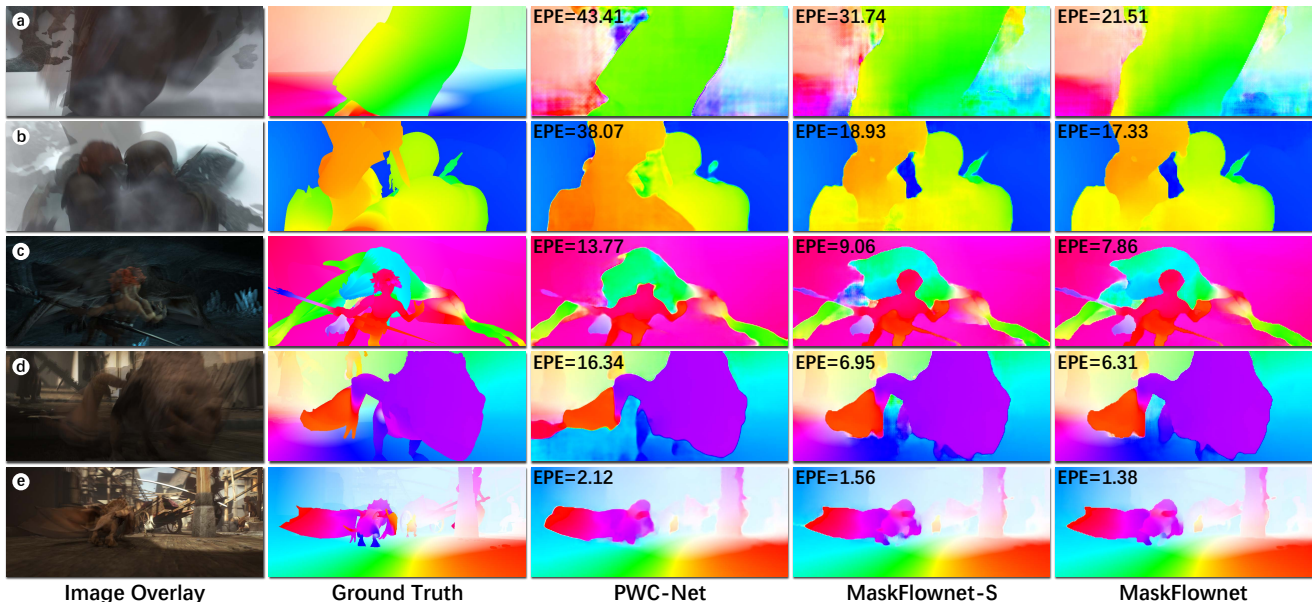


Figure 6. **Qualitative comparison among PWC-Net [32], MaskFlownet-S, and MaskFlownet.** Highlights for each row: (a) weakening the checkerboard effect; (b) separating background from two fighting figures; (c) preserving the weakly connected head of the moving figure; (d) maintaining a fluent flow for the background at left-bottom; (e) preserving boundary details of the flying creature and buildings.

## 5.2. Main Results

MaskFlownet outperforms all published optical flow methods on the MPI Sintel [5], KITTI 2012 [9] and KITTI 2015 [23] benchmarks as presented in Table 1, while the end-to-end MaskFlownet-S achieves a satisfactory result as well. Values listed in the training sets only consider the

models that have never seen them and hence comparable; note that training on the synthetic datasets is of limited generalizability. Fig. 6 visualizes the predicted flows as a comparison. All samples are chosen from the Sintel training set (final pass). We can observe that MaskFlownet in general better separates moving objects from the background,

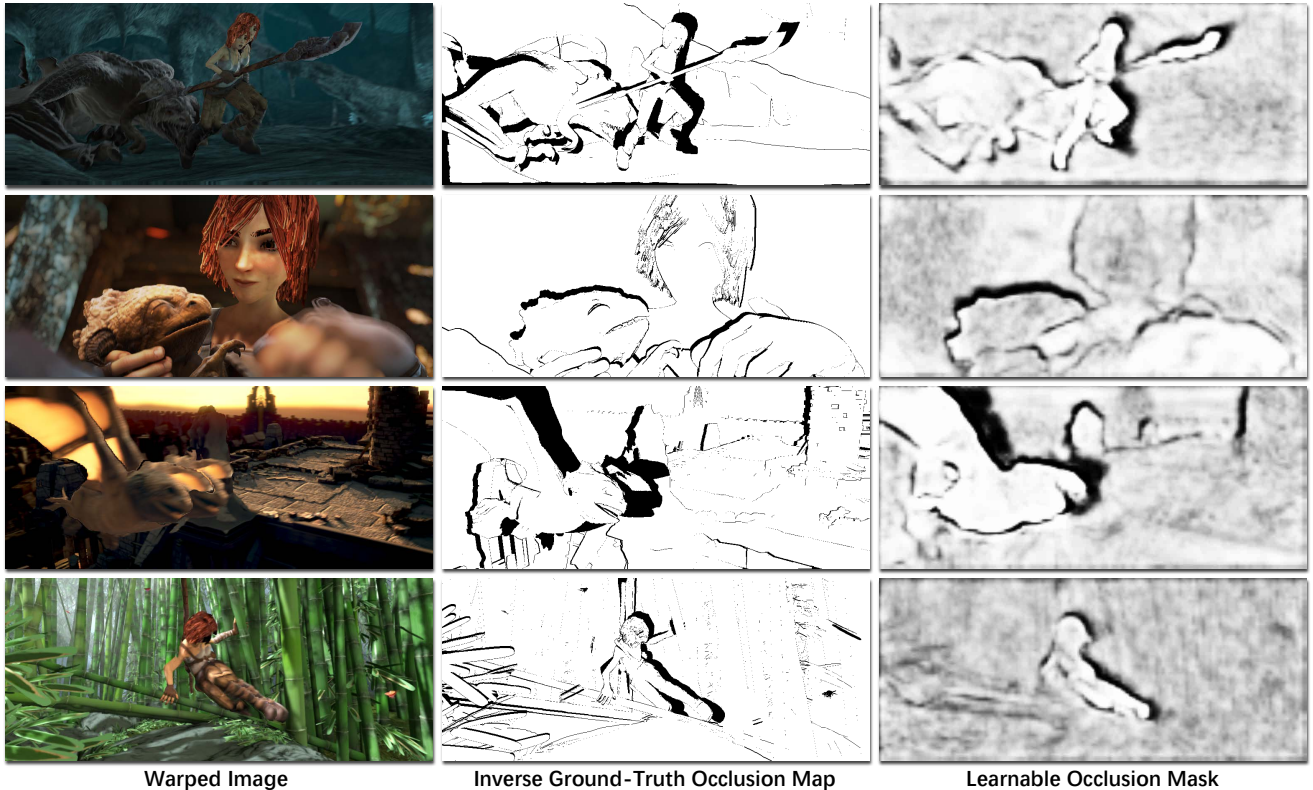


Figure 7. **Learnable occlusion mask.** MaskFlowNet can jointly learn a rough occlusion mask without any explicit supervision.

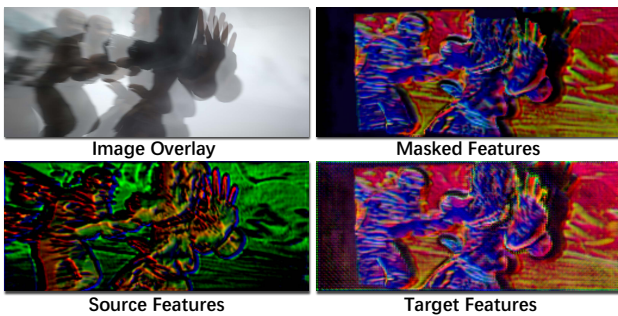


Figure 8. **Asymmetry in the learned feature maps.** The source features and the target features are level-2 feature maps prior to the correlation layer in the AsymOFMM. This figure presents the input image overlay, image 1 features (source features), the warped image 2 features after masking (masked features) and after trade-off (target features). Comparing the source features with the target features, we can see that the AsymOFMM enables the network to learn very different feature representations.

and cascading significantly weakens the checkerboard effect while preserving clear object boundaries. Fig. 7 qualitatively demonstrates that the learnable occlusion mask matches the inverse ground-truth occlusion map fairly well, even if it is learned without any explicit supervision.

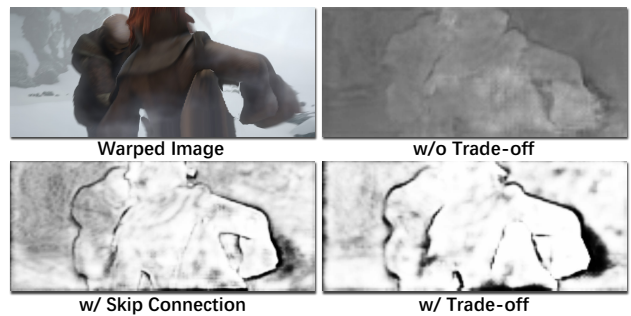


Figure 9. **The trade-off term facilitates the learning of occlusions.** Without the trade-off term, the learnable occlusion mask fails to achieve a clear estimation; if there is an additive shortcut that skips over warping, only motion boundaries are learned. With the trade-off term, large occlusions are successfully learned.

### 5.3. Ablation Study

**Feature Matching Module.** Table 2 presents the results when replacing the AsymOFMM in MaskFlowNet-S with OFMM or FMM. We split about 20% sequences for validation<sup>1</sup> when fine-tuning on the Sintel training set. While

<sup>1</sup>ambush\_2, ambush\_6, bamboo\_2, cave\_4, market\_6, temple\_2.



Module	Trained on Chairs			Things3D		Sintel	
	Chairs <i>test</i>	Sintel ( <i>train</i> ) <i>clean</i> <i>final</i>		Sintel ( <i>train</i> ) <i>clean</i> <i>final</i>		Sintel ( <i>val</i> ) <i>clean</i> <i>final</i>	
FMM	1.61	3.25	4.59	2.55	4.05	3.02	4.70
OFMM	1.62	3.20	4.50	2.52	4.01	3.06	4.52
AsymOFMM	<b>1.56</b>	<b>2.88</b>	<b>4.25</b>	<b>2.33</b>	<b>3.72</b>	<b>2.70</b>	<b>4.07</b>

Table 2. **Feature matching module.**

Module	Trained on Chairs			Things3D	
	Chairs <i>test</i>	Sintel ( <i>train</i> ) <i>clean</i> <i>final</i>		Sintel ( <i>train</i> ) <i>clean</i> <i>final</i>	
OFMM	1.62	3.20	4.50	2.52	4.01
+ sym-conv	1.61	3.33	4.64	2.54	3.84
+ asym-conv	<b>1.52</b>	2.96	4.29	2.41	3.85
+ deform-conv	1.56	<b>2.88</b>	<b>4.25</b>	<b>2.33</b>	<b>3.72</b>

Table 3. **Asymmetry and deformable convolution.**

Module (AsymOFMM)	Trained on Chairs		
	Chairs <i>test</i>	Sintel ( <i>train</i> ) <i>clean</i> <i>final</i>	
w/o mask w/o trade-off	1.58	3.08	4.29
w/ mask w/o trade-off	1.60	3.06	4.32
w/o mask w/ trade-off	1.58	2.97	4.30
(w/ mask w/ trade-off)	<b>1.56</b>	<b>2.88</b>	<b>4.25</b>

Table 4. **Learnable occlusion mask and the trade-off term.**

Network	Tuned on Sintel	
	Sintel ( <i>val</i> ) <i>clean</i>	<i>final</i>
MaskFlowNet-S	2.70	4.07
+ single pyramid w/o mask	2.53	3.90
+ single pyramid w/ mask	2.55	3.88
+ dual pyramids w/o mask	<b>2.52</b>	3.85
+ dual pyramids w/ mask	<b>2.52</b>	<b>3.83</b>

Table 5. **Network cascading with dual pyramids.**

OFMM achieves relative gains compared with the original FMM, the proposed AsymOFMM significantly outperforms the symmetric variants.

**Asymmetry.** We demonstrate the effectiveness of the asymmetry by comparing AsymOFMM (i.e., “OFMM + deform-conv”, MaskFlowNet-S) with the raw OFMM, “OFMM + sym-conv” (adding an extra convolutional layer at each feature pyramid level), and “OFMM + asym-conv” (adding an asymmetric convolutional layer prior to the warping operation at each level). As shown in Table 3, increasing the depth of convolutional layers has a limited impact in the symmetric setting, while a simple asymmetric design achieves consistently better performance. It also indicates that our deformable convolution can be a better choice over the “asym-conv” version. Although it is commonly believed that matched patterns should be embedded into similar feature vectors, we suggest that the network can

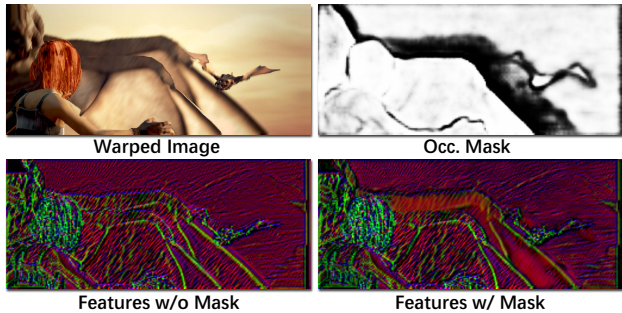


Figure 10. **Occlusion mask facilitates feature extraction in the occlusion-aware feature pyramid.** Comparing the learned features with and without (i.e., replaced by constant) mask, we can see that the occlusion mask draws a significant effect on smoothing the feature map at the occluded areas.

really benefit from learning very different feature representations as visualized in Fig. 8.

**Learnable Occlusion Mask.** Table 4 presents the results if the mask or the trade-off term is disabled. Interestingly, only the two factors combined lead to performance gains. A possible explanation is that the occlusion mask helps if and only if it is learned properly, where the trade-off term plays a vital role (see Fig. 9).

**Network Cascading.** Table 5 indicates that MaskFlowNet consistently benefits from dual feature pyramids over a single new pyramid, while the concatenated occlusion mask gains the performance on the Sintel final pass. We hypothesize that the occlusion-aware feature pyramid mainly contributes to the harder final pass since the occluded areas can be more easily mismatched, but it might be overfitted on the easier clean pass. We demonstrate how the learned occlusion mask could affect the extracted feature map in Fig. 10. The occluded areas are smoothed during feature extraction and hence become more distinguishable.

## 6. Conclusion

We propose the AsymOFMM, which incorporates a learnable occlusion mask that filters occluded areas immediately after feature warping without any explicit supervision. AsymOFMM can be easily integrated into an end-to-end network while introducing negligible computational cost. We further propose a two-stage network — MaskFlowNet — which exploits dual pyramids and achieves superior performance on all modern optical flow benchmarks. Our approach opens a promisingly new perspective on dealing with occlusions for both supervised and unsupervised optical flow estimation, and we also expect it as an initiative and effective component in many other applications.



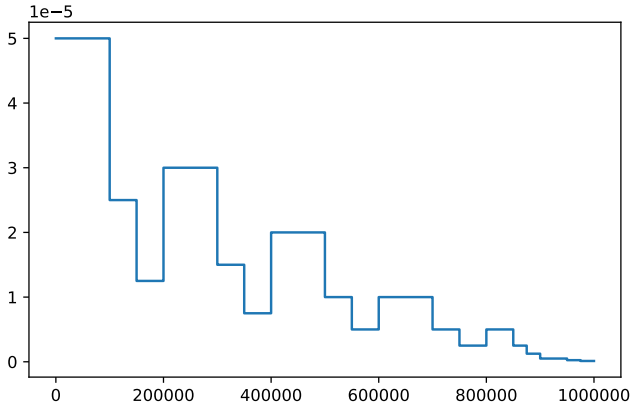
## References

- [1] Christian Bailer, Kiran Varanasi, and Didier Stricker. Cnn-based patch matching for optical flow with thresholded hinge embedding loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3250–3259, 2017. **2**
- [2] Nicolas Bonneel, James Tompkin, Kalyan Sunkavalli, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. Blind video temporal consistency. *ACM Transactions on Graphics (TOG)*, 34(6):196, 2015. **1**
- [3] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision*, pages 25–36. Springer, 2004. **2**
- [4] Thomas Brox and Jitendra Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE transactions on pattern analysis and machine intelligence*, 33(3):500–513, 2010. **2**
- [5] 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. Springer, 2012. **6**
- [6] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015. **5**
- [7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. **4**
- [8] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. **2, 5, 6, 11**
- [9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. **6**
- [10] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. **2**
- [11] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Lite-flownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8981–8989, 2018. **1, 2, 3, 6**
- [12] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow cnn—revisiting data fidelity and regularization. *arXiv preprint arXiv:1903.07414*, 2019. **1, 2, 3, 6**
- [13] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5754–5763, 2019. **1, 3, 5, 11**
- [14] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017. **2, 5, 6**
- [15] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 614–630, 2018. **3**
- [16] Joel Janai, Fatma Guney, Anurag Ranjan, Michael Black, and Andreas Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 690–706, 2018. **3**
- [17] J Yu Jason, Adam W Harley, and Konstantinos G Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *European Conference on Computer Vision*, pages 3–10. Springer, 2016. **3**
- [18] Pengpeng Liu, Michael Lyu, Irwin King, and Jia Xu. Self-low: Self-supervised learning of optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4571–4580, 2019. **1, 3, 6, 11**
- [19] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016. **3, 5**
- [20] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. **3**
- [21] Etienne Mémin and Patrick Pérez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, 1998. **2**
- [22] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015. **1**
- [23] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015. **6**
- [24] Michal Neoral, Jan Šochman, and Jiří Matas. Continual occlusion and optical flow estimation. In *Asian Conference on Computer Vision*, pages 159–174. Springer, 2018. **1, 3**
- [25] Simon Niklaus. A reimplementation of PWC-Net using PyTorch. <https://github.com/sniklaus/pytorch-pwc>, 2018. **13**
- [26] Yanwei Pang, Jin Xie, Muhammad Haris Khan, Rao Muhammad Anwer, Fahad Shahbaz Khan, and Ling Shao. Mask-guided attention network for occluded pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4967–4975, 2019. **3**

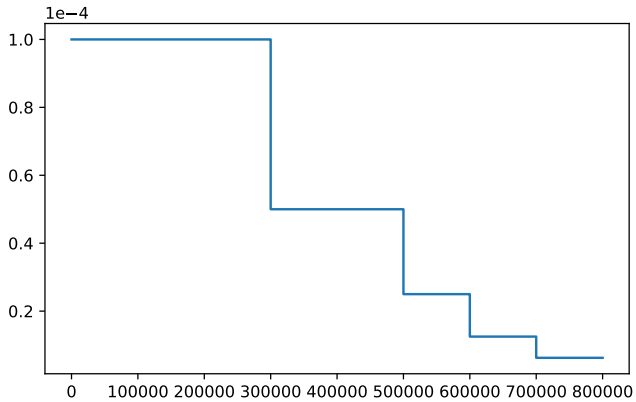
- [27] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017. 2, 6
- [28] Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised deep learning for optical flow estimation. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 3
- [29] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 1
- [30] Lingxue Song, Dihong Gong, Zhifeng Li, Changsong Liu, and Wei Liu. Occlusion robust face recognition based on mask learning with pairwise differential siamese network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 773–782, 2019. 3
- [31] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *arXiv preprint arXiv:1809.05571*, 2018. 1, 2, 3, 5, 6, 11, 13
- [32] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018. 1, 2, 3, 4, 5, 6, 13
- [33] Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *European conference on computer vision*, pages 438–451. Springer, 2010. 3
- [34] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018. 3
- [35] Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. Occlusion aware unsupervised learning of optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4884–4893, 2018. 3
- [36] Andreas Wedel, Daniel Cremers, Thomas Pock, and Horst Bischof. Structure-and motion-adaptive regularization for high accuracy optic flow. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1663–1668. IEEE, 2009. 2
- [37] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1385–1392, 2013. 2
- [38] Jonas Wulff, Laura Sevilla-Lara, and Michael J Black. Optical flow in mostly rigid scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4671–4680, 2017. 6, 11
- [39] Jia Xu, René Ranftl, and Vladlen Koltun. Accurate optical flow via direct cost volume processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1289–1297, 2017. 2
- [40] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *Advances in neural information processing systems*, 2019. 2, 6
- [41] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. Occlusion-aware r-cnn: detecting pedestrians in a crowd. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 637–653, 2018. 3
- [42] Shanshan Zhang, Jian Yang, and Bernt Schiele. Occluded pedestrian detection through guided attention in cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6995–7003, 2018. 3

## Appendix A. More Implementation Details

**Training Schedule.** When fine-tuning on Sintel, we use a longer schedule (see Fig. 11(a)) referring to the cyclic learning rate proposed by PWC-Net+ [31]. When training the second stage, we follow again the same schedule as the first stage for all datasets except that it is shorter on FlyingChairs (see Fig. 11(b)). For submission to the test set, we train on the whole training set and reduce randomness by averaging 3 independent runs due to the huge variance.



(a) Schedule for fine-tuning on Sintel.



(b) A shorter schedule for the second stage on FlyingChairs.

Figure 11. Learning rate schedules.

**Data Augmentation.** We implement geometric and chromatic augmentations referring to the implementation of FlowNet [8] and IRR-PWC [13]. Details about the sampling ranges for each training stage are provided in Table 6 (for geometric augmentations) and Table 7 (for chromatic augmentations). We use the same augmentations on FlyingThings3D as FlyingChairs. We finally apply a random crop (within valid areas) using a size of  $448 \times 320$  on FlyingChairs,  $768 \times 384$  on FlyingThings3D,  $768 \times 320$  on Sintel, and  $896 \times 320$  on KITTI. To avoid out-of-bound areas

Geometric Aug.	Chairs	Sintel	KITTI
Horizontal Flip	0.5	0.5	0.5
Squeeze	0.9	0.9	0.95
Translation	0.1	0.1	0.05
Rel. Translation	0.025	0.025	0.0125
Rotation	$17^\circ$	$17^\circ$	$5^\circ$
Rel. Rotation	$4.25^\circ$	$4.25^\circ$	$1.25^\circ$
Zoom	[0.9, 2.0]	[0.9, 1.5]	[0.95, 1.25]
Rel. Zoom	0.96	0.96	0.98

Table 6. Geometric augmentations.

Chromatic Aug.	Chairs	Sintel	KITTI
Contrast	[-0.4, 0.8]	[-0.4, 0.8]	[-0.2, 0.4]
Brightness	0.1	0.1	0.05
Channel	[0.8, 1.4]	[0.8, 1.4]	[0.9, 1.2]
Saturation	0.5	0.5	0.25
Hue	0.5	0.5	0.1
Noise	0.04	0	0.02

Table 7. Chromatic augmentations.

after cropping, we compute the minimum degree of zoom that forces the existence of a valid crop.

## Appendix B. More Visualizations

More visualizations of the learnable occlusion mask and the flow predictions are presented in Fig. 12 and Fig. 13. Note that the learned occlusion masks are relatively vague at the image boundary, since the network cannot learn to mask out-of-bound features that are already zeros. We expect that the estimation results can be further improved if out-of-bound areas are manually regarded as occlusions.

## Appendix C. Screenshots on Benchmarks

At the time of submission, MaskFlowNet ranks first on the MPI Sintel benchmark on both clean pass (see Fig. 14) and final pass (see Fig. 15). Note that the top entry (ScopeFlow) at the time of screenshot (Nov. 23th, 2019) on the final pass is a new anonymous submission, with a relatively poor performance on the clean pass. Remarkably, MaskFlowNet outperforms the previous top entry on the clean pass (MR-Flow [38]) that uses the rigidity assumption while being very slow, as well as the previous top entry on the final pass (SelfFlow [18]) that uses multi-frame inputs.

On the KITTI 2012 and 2015 benchmarks, MaskFlowNet surpasses all optical flow methods (excluding the anonymous entries) at the time of submission (see Fig. 16 and Fig. 17). Note that the top 3 entries on the KITTI 2015 benchmark are *scene flow* methods that use stereo images and thus not comparable.

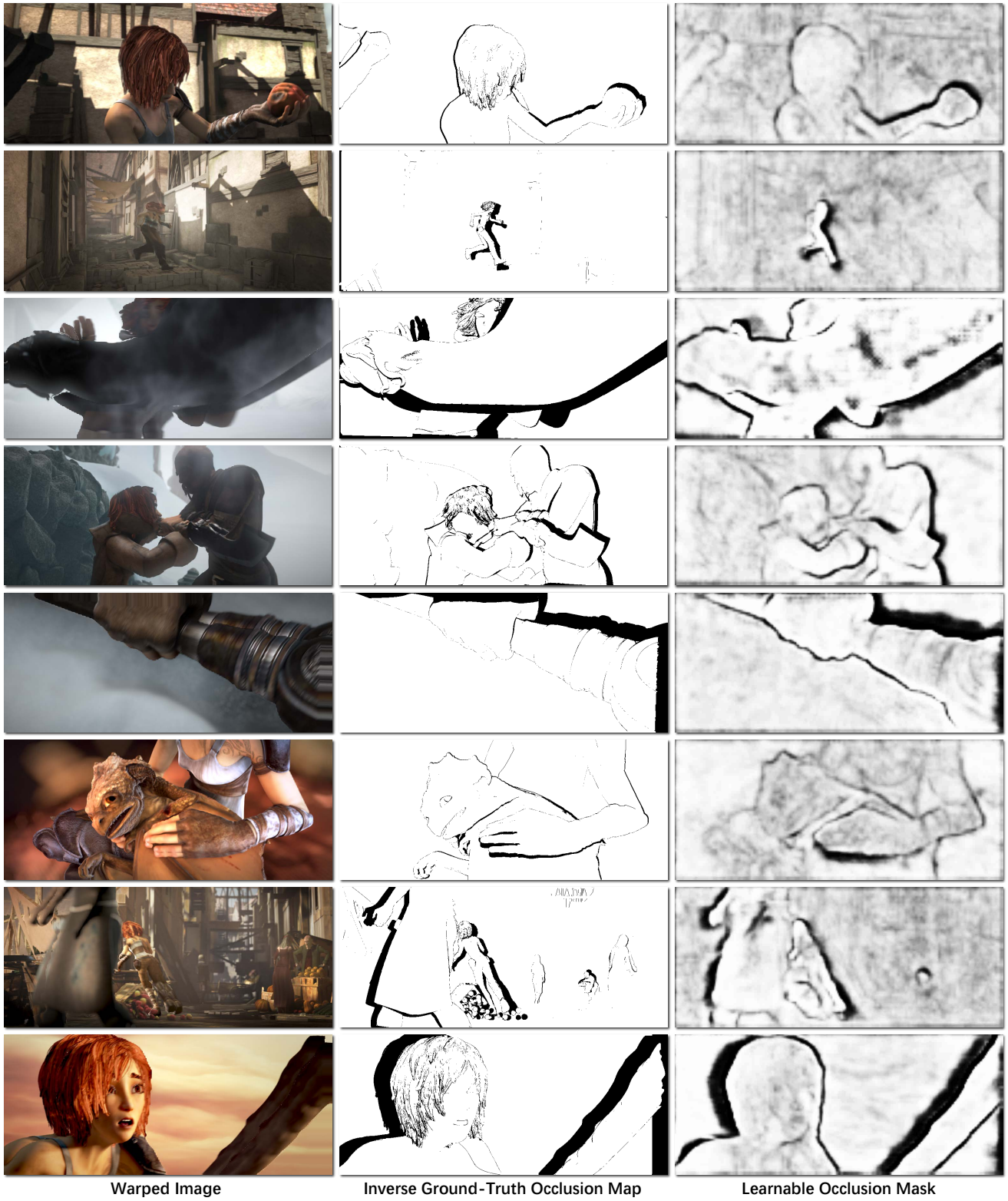


Figure 12. More visualizations of the learnable occlusion mask. All samples are chosen from the Sintel training set (final pass). The learnable occlusion masks are expected to (roughly) match the inverse ground-truth occlusion maps, even if they are learned without any explicit supervision.



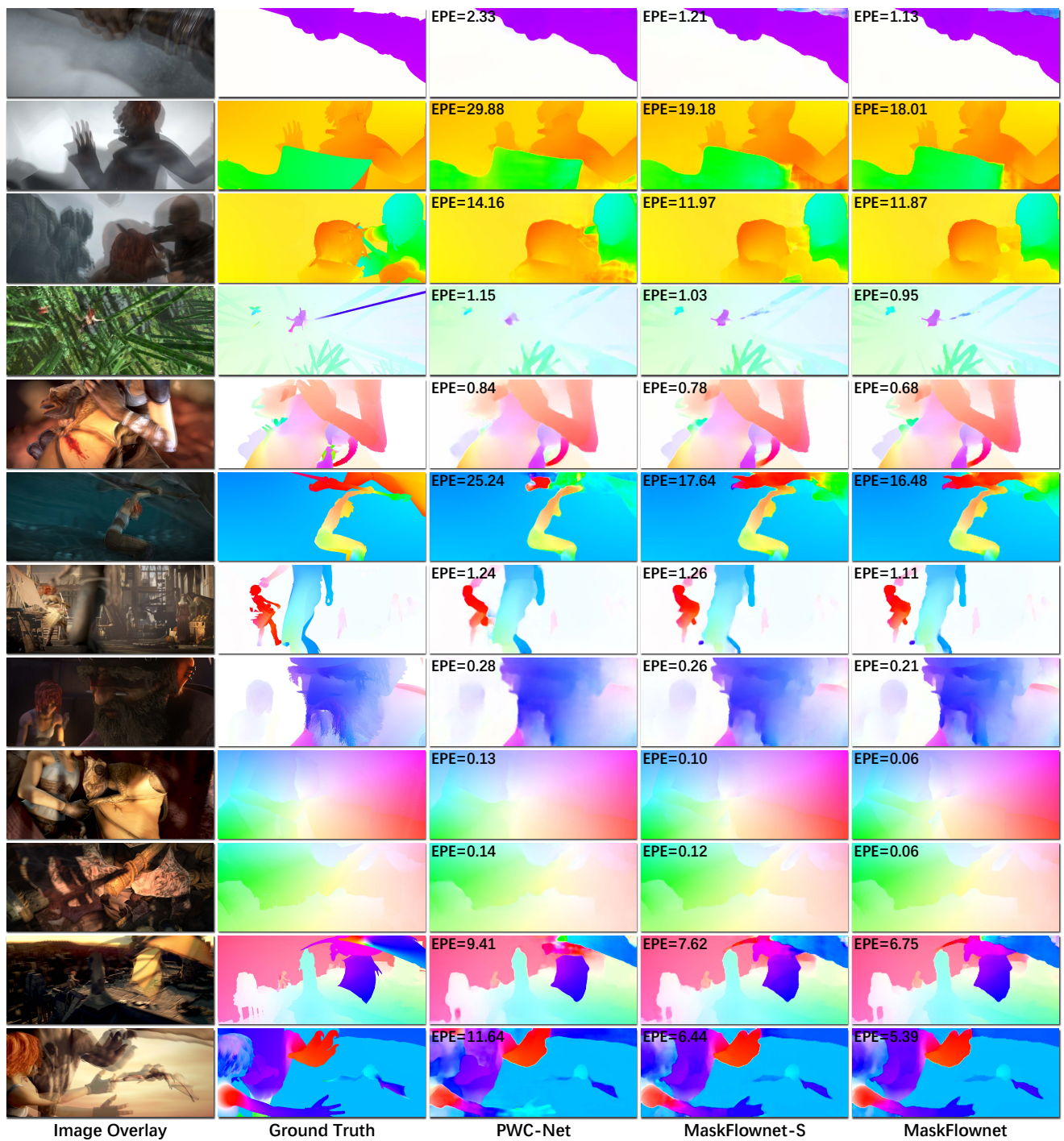
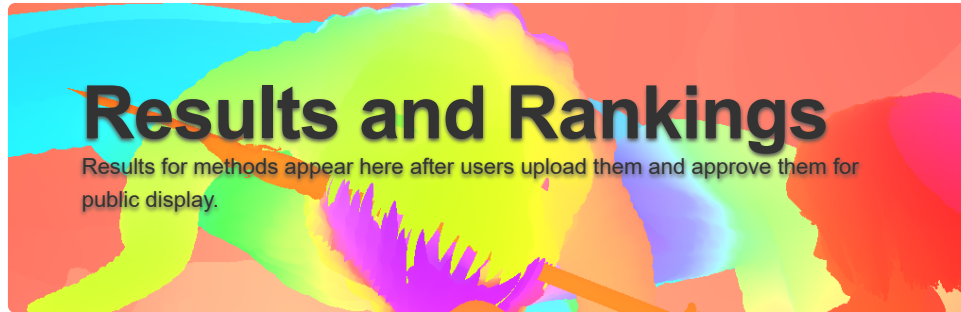


Figure 13. **More visualizations for qualitative comparison among PWC-Net [32], MaskFlownet-S, and MaskFlownet.** All samples are chosen from the Sintel training set (final pass). We replicate PWC-Net using the PyTorch reimplementation [25] that provides a pretrained model of the “PWC-Net\_ROB” version [31].



[Final](#) [Clean](#)

	EPE all	EPE matched	EPE unmatched	d0-10	d10- 60	d60- 140	s0-10	s10- 40	s40+	
GroundTruth <sup>[1]</sup>	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	<a href="#">Visualize Results</a>
MaskFlowNet <sup>[2]</sup>	2.521	0.989	15.032	2.742	0.908	0.291	0.361	1.285	16.261	<a href="#">Visualize Results</a>
MR-Flow <sup>[3]</sup>	2.527	0.954	15.365	2.866	0.710	0.420	0.446	1.715	14.826	<a href="#">Visualize Results</a>
ProFlow_ROB <sup>[4]</sup>	2.709	1.013	16.549	2.843	0.723	0.518	0.485	1.586	16.470	<a href="#">Visualize Results</a>
MaskFlowNet-S <sup>[5]</sup>	2.771	1.077	16.608	2.901	0.996	0.342	0.419	1.404	17.777	<a href="#">Visualize Results</a>
VCN <sup>[6]</sup>	2.808	1.108	16.682	3.267	0.867	0.418	0.646	1.669	16.302	<a href="#">Visualize Results</a>
ProFlow <sup>[7]</sup>	2.818	1.027	17.428	2.892	0.751	0.496	0.469	1.626	17.369	<a href="#">Visualize Results</a>
SfM-PM <sup>[8]</sup>	2.910	1.016	18.357	2.797	0.756	0.479	0.559	1.732	17.431	<a href="#">Visualize Results</a>
FlowFields++ <sup>[9]</sup>	2.943	0.850	20.027	2.550	0.603	0.403	0.560	1.859	17.401	<a href="#">Visualize Results</a>
LiteFlowNet3 <sup>[10]</sup>	2.994	1.148	18.077	3.000	0.985	0.498	0.559	1.670	18.302	<a href="#">Visualize Results</a>
FlowFields+ <sup>[11]</sup>	3.102	0.820	21.718	2.340	0.616	0.373	0.593	1.865	18.549	<a href="#">Visualize Results</a>
DIP-Flow <sup>[12]</sup>	3.103	0.881	21.227	2.574	0.681	0.419	0.548	1.801	18.979	<a href="#">Visualize Results</a>
PST <sup>[13]</sup>	3.110	0.942	20.809	2.759	0.664	0.378	0.635	2.069	17.919	<a href="#">Visualize Results</a>

Figure 14. Screenshot on the MPI Sintel clean pass (printed as PDF).



[Final](#) [Clean](#)

	EPE all	EPE matched	EPE unmatched	d0-10	d10-60	d60-140	s0-10	s10-40	s40+	
GroundTruth <sup>[1]</sup>	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	<a href="#">Visualize Results</a>
ScopeFlow <sup>[2]</sup>	4.098	1.999	21.214	4.028	1.689	1.180	0.725	2.589	24.477	<a href="#">Visualize Results</a>
MaskFlowNet <sup>[3]</sup>	4.172	2.048	21.494	3.783	1.745	1.310	0.592	2.389	26.253	<a href="#">Visualize Results</a>
SelFlow <sup>[4]</sup>	4.262	2.040	22.369	4.083	1.715	1.287	0.582	2.343	27.154	<a href="#">Visualize Results</a>
MaskFlowNet-S <sup>[5]</sup>	4.384	2.120	22.840	3.905	1.821	1.359	0.645	2.526	27.429	<a href="#">Visualize Results</a>
VCN <sup>[6]</sup>	4.404	2.216	22.238	4.381	1.782	1.423	0.955	2.725	25.570	<a href="#">Visualize Results</a>
LiteFlowNet3 <sup>[7]</sup>	4.448	2.089	23.681	3.873	1.755	1.344	0.754	2.503	27.471	<a href="#">Visualize Results</a>
ContinualFlow_ROB <sup>[8]</sup>	4.528	2.723	19.248	5.050	2.573	1.713	0.872	3.114	26.063	<a href="#">Visualize Results</a>
MF <sup>[9]</sup>	4.566	2.216	23.732	4.664	2.017	1.222	0.893	2.902	26.810	<a href="#">Visualize Results</a>
IRR-PWC <sup>[10]</sup>	4.579	2.154	24.355	4.165	1.843	1.292	0.709	2.423	28.998	<a href="#">Visualize Results</a>
PWC-Net+ <sup>[11]</sup>	4.596	2.254	23.696	4.781	2.045	1.234	0.945	2.978	26.620	<a href="#">Visualize Results</a>
PPAC-HD3 <sup>[12]</sup>	4.599	2.116	24.852	3.521	1.702	1.637	0.617	2.083	30.457	<a href="#">Visualize Results</a>
CompactFlow <sup>[13]</sup>	4.626	2.099	25.253	4.192	1.825	1.233	0.845	2.677	28.120	<a href="#">Visualize Results</a>

Figure 15. Screenshot on MPI Sintel final pass (printed as PDF).





background interpolation as explained in the corresponding header file in the development kit. For each method we show:

- **Out-Noc:** Percentage of erroneous pixels in non-occluded areas
- **Out-All:** Percentage of erroneous pixels in total
- **Avg-Noc:** Average disparity / end-point error in non-occluded areas
- **Avg-All:** Average disparity / end-point error in total
- **Density:** Percentage of pixels for which ground truth has been provided by the method

**Note:** On 04.11.2013 we have improved the ground truth disparity maps and flow fields leading to slightly improvements for all methods. Please download the stereo/flow dataset with the improved ground truth for training again, if you have downloaded the dataset prior to 04.11.2013. Please consider reporting these new number for all future submissions. Links to last leaderboards before the updates: [stereo](#) and [flow](#)!

**Important Policy Update:** As more and more non-published work and re-implementations of existing work is submitted to KITTI, we have established a new policy: from now on, only submissions with significant novelty that are leading to a peer-reviewed paper in a conference or journal are allowed. Minor modifications of existing algorithms or student research projects are not allowed. Such work must be evaluated on a split of the training set. To ensure that our policy is adopted, new users must detail their status, describe their work and specify the targeted venue during registration. Furthermore, we will regularly delete all entries that are 6 months old but are still anonymous or do not have a paper associated with them. For conferences, 6 month is enough to determine if a paper has been accepted and to add the bibliography information. For longer review cycles, you need to resubmit your results.

#### Additional information used by the methods

-  Stereo: Method uses left and right (stereo) images
-  Multiview: Method uses more than 2 temporally adjacent images
-  Motion stereo: Method uses epipolar geometry for computing optical flow
-  Additional training data: Use of additional data sources for training (see details)

Method	Setting	Code	Out-Noc	Out-All	Avg-Noc	Avg-All	Density	Runtime	Environment	Compare
1	<a href="#">DM-Net-i2</a>	<a href="#">code</a>	0.00 %	0.00 %	0.0 px	0.0 px	0.00 %	0.90 s	1 core @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
2	<a href="#">Anonym</a>		0.00 %	0.00 %	0.0 px	0.0 px	0.00 %	TBD s	1 core @ 2.5 Ghz (Python)	<input type="checkbox"/>
3	<a href="#">PPAC-HD3</a>		2.01 %	5.09 %	0.6 px	1.2 px	100.00 %	0.14 s	NVIDIA GTX 1080 Ti	<input type="checkbox"/>
4	<a href="#">PCF-F</a>		2.07 %	5.45 %	0.6 px	1.2 px	100.00 %	0.08 s	GPU @ 2.5 Ghz (Python)	<input type="checkbox"/>
5	<a href="#">MaskFlowNet</a>		2.07 %	4.82 %	0.6 px	1.1 px	100.00 %	0.06 s	NVIDIA TITAN Xp	<input type="checkbox"/>
6	<a href="#">HD^3-Flow</a>	<a href="#">code</a>	2.26 %	5.41 %	0.7 px	1.4 px	100.00 %	0.10 s	NVIDIA Pascal Titan XP	<input type="checkbox"/>
Z. Yin, T. Darrell and F. Yu: <a href="#">Hierarchical Discrete Distribution Decomposition for Match Density Estimation</a> . CVPR 2019.										
7	<a href="#">MaskFlowNet-S</a>		2.29 %	5.24 %	0.6 px	1.1 px	100.00 %	0.03 s	NVIDIA TITAN Xp	<input type="checkbox"/>
8	<a href="#">PRSM</a>	<a href="#">code</a>	2.46 %	4.23 %	0.7 px	1.0 px	100.00 %	300 s	1 core @ 2.5 Ghz (Matlab + C/C++)	<input type="checkbox"/>
C. Vogel, K. Schindler and S. Roth: <a href="#">3D Scene Flow Estimation with a Piecewise Rigid Scene Model</a> . ijcv 2015.										
9	<a href="#">LiteFlowNet3-S</a>		2.49 %	5.91 %	0.7 px	1.3 px	100.00 %	TBD	NVIDIA TITAN XP	<input type="checkbox"/>
10	<a href="#">LiteFlowNet3</a>		2.51 %	5.90 %	0.7 px	1.3 px	100.00 %	TBD	NVIDIA TITAN XP	<input type="checkbox"/>
11	<a href="#">HTC</a>		2.55 %	7.84 %	0.8 px	1.6 px	100.00 %	0.03 s	1 core @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
12	<a href="#">cvpr-304</a>		2.58 %	5.62 %	0.7 px	1.3 px	100.00 %	-1 s	1 core @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
13	<a href="#">LiteFlowNet2</a>	<a href="#">code</a>	2.63 %	6.16 %	0.7 px	1.4 px	100.00 %	0.0486 s	GTX 1080 (slower than Pascal Titan X)	<input type="checkbox"/>

Figure 16. Screenshot on the KITTI 2012 benchmark (printed as PDF).



- [Download development kit \(3 MB\)](#)





Our evaluation table ranks all methods according to the number of erroneous pixels. All methods providing less than 100 % density have been interpolated using simple background interpolation as explained in the corresponding header file in the development kit. Legend:

- **D1**: Percentage of stereo disparity outliers in first frame
- **D2**: Percentage of stereo disparity outliers in second frame
- **Fl**: Percentage of optical flow outliers
- **SF**: Percentage of scene flow outliers (=outliers in either D0, D1 or Fl)
- **bg**: Percentage of outliers averaged only over background regions
- **fg**: Percentage of outliers averaged only over foreground regions
- **all**: Percentage of outliers averaged over all ground truth pixels

**Note:** On 13.03.2017 we have fixed several small errors in the flow (noc+occ) ground truth of the dynamic foreground objects and manually verified all images for correctness by warping them according to the ground truth. As a consequence, all error numbers have decreased slightly. Please download the devkit and the annotations with the improved ground truth for the training set again if you have downloaded the files prior to 13.03.2017 and consider reporting these new number in all future publications. The last leaderboards before these corrections can be found [here \(optical flow 2015\)](#) and [here \(scene flow 2015\)](#). The leaderboards for the KITTI 2015 stereo benchmarks did not change.

**Important Policy Update:** As more and more non-published work and re-implementations of existing work is submitted to KITTI, we have established a new policy: from now on, only submissions with significant novelty that are leading to a peer-reviewed paper in a conference or journal are allowed. Minor modifications of existing algorithms or student research projects are not allowed. Such work must be evaluated on a split of the training set. To ensure that our policy is adopted, new users must detail their status, describe their work and specify the targeted venue during registration. Furthermore, we will regularly delete all entries that are 6 months old but are still anonymous or do not have a paper associated with them. For conferences, 6 month is enough to determine if a paper has been accepted and to add the bibliography information. For longer review cycles, you need to resubmit your results.

#### **Additional information used by the methods**

-  Stereo: Method uses left and right (stereo) images
-  Multiview: Method uses more than 2 temporally adjacent images
-  Motion stereo: Method uses epipolar geometry for computing optical flow
-  Additional training data: Use of additional data sources for training (see details)






Evaluation ground truth		All pixels		Evaluation area		All pixels					
Method	Setting Code	Fl-bg	Fl-fg	Fl-all	Density	Runtime	Environment	Compare			
1	<a href="#">UberATG-DRISE</a>		<b>3.59</b> %	10.40 %	<b>4.73</b> %	100.00 %	0.75 s	CPU+GPU @ 2.5 Ghz (Python)	<input type="checkbox"/>		
W. Ma, S. Wang, R. Hu, Y. Xiong and R. Urtasun: <a href="#">Deep Rigid Instance Scene Flow</a> . CVPR 2019.											
2	<a href="#">DH-SF</a>		4.12 %	12.07 %	5.45 %	100.00 %	350 s	1 core @ 2.5 Ghz (Matlab + C/C++)	<input type="checkbox"/>		
3	<a href="#">IAOSE</a>		4.56 %	12.00 %	5.79 %	100.00 %	5 min	1 core @ 3.5 Ghz (Matlab + C/C++)	<input type="checkbox"/>		
4	<a href="#">PCF-F</a>		6.05 %	<b>5.99</b> %	6.04 %	100.00 %	0.08 s	GPU @ 2.5 Ghz (Python)	<input type="checkbox"/>		
5	<a href="#">PPAC-HD3</a>		5.78 %	7.48 %	6.06 %	100.00 %	0.14 s	NVIDIA GTX 1080 Ti	<input type="checkbox"/>		
6	<a href="#">MaskFlownet</a>		<b>5.79</b> %	<b>7.70</b> %	<b>6.11</b> %	<b>100.00</b> %	<b>0.06 s</b>	<b>NVIDIA TITAN Xp</b>	<input type="checkbox"/>		
7	<a href="#">ISE</a>		5.40 %	10.29 %	6.22 %	100.00 %	10 min	1 core @ 3 Ghz (C/C++)	<input type="checkbox"/>		
A. Behl, O. Jafari, S. Mustikovela, H. Alhajja, C. Rother and A. Geiger: <a href="#">Bounding Boxes, Segmentations and Object Coordinates: How Important is Recognition for 3D Scene Flow Estimation in Autonomous Driving Scenarios?</a> . International Conference on Computer Vision (ICCV) 2017.											
8	<a href="#">VCN</a>		5.83 %	8.66 %	6.30 %	100.00 %	0.2 s	Titan X Pascal	<input type="checkbox"/>		
G. Yang and D. Ramanan: <a href="#">Volumetric Correspondence Networks for Optical Flow</a> . NeurIPS 2019.											
9	<a href="#">Mono expansion</a>		5.83 %	8.66 %	6.30 %	100.00 %	0.25 s	GPU @ 2.5 Ghz (Python)	<input type="checkbox"/>		

Figure 17. Screenshot on the KITTI 2015 benchmark (printed as PDF). MaskFlownet-S ranks 14th.