Occlusions, Motion and Depth Boundaries with a Generic Network for Disparity, Optical Flow or Scene Flow Estimation

Eddy Ilg*, Tonmoy Saikia*, Margret Keuper, and Thomas Brox

University of Freiburg, Germany
{ilg,saikia,keuper,brox}@cs.uni-freiburg.de

Abstract. Occlusions play an important role in disparity and optical flow estimation, since matching costs are not available in occluded areas and occlusions indicate depth or motion boundaries. Moreover, occlusions are relevant for motion segmentation and scene flow estimation. In this paper, we present an efficient learning-based approach to estimate occlusion areas jointly with disparities or optical flow. The estimated occlusions and motion boundaries clearly improve over the state-of-the-art. Moreover, we present networks with state-of-the-art performance on the popular KITTI benchmark and good generic performance. Making use of the estimated occlusions, we also show improved results on motion segmentation and scene flow estimation.

1 Introduction

When applying dense correspondences to higher level tasks, there is often the desire for additional information apart from the raw correspondences. The areas in one image that are occluded in the other image are important to get an indication of potentially unreliable estimates due to missing measurements. A typical approach to estimate occluded areas is by computing correspondences in both directions and verifying their consistency post-hoc. However, since occlusions and correspondences are mutually dependent [17, 32] and the presence of occlusions already negatively influences the correspondence estimation itself, post-processing is suboptimal and leads to unreliable occlusion estimates.

Another valuable extra information in disparity maps and flow fields are explicit depth and motion boundaries, respectively. Referring to the classic work of Black&Jepson [4], "motion boundaries may be useful for navigation, structure from motion, video compression, perceptual organization and object recognition".

In this paper, we integrate occlusion estimation as well as depth or motion boundary estimation elegantly with a deep network for disparity or optical flow estimation based on FlowNet 2.0 [18] and provide these quantities explicitly as

* equal contribution

output. In contrast to many prior works, this leads to much improved occlusion and boundary estimates and much faster overall runtimes. We quantify this improvement directly by measuring the accuracy of the occlusions and motion boundaries. We also quantify the effect of this improved accuracy on motion segmentation.

Furthermore we improved on some details in the implementation of the disparity and optical flow estimation networks from [29, 11, 18], which gives us state-of-the-art results on the KITTI benchmarks. Moreover, the networks show good generic performance on various datasets if we do not fine-tune them to a particular scenario. While these are smaller technical contributions, they are very relevant for applications of optical flow and disparity. Finally, with state-of-the-art optical flow, disparity and occlusion estimates in place, we put everything together to achieve good scene flow performance at a high frame-rate, using only 2D motion information. Using our predicted occlusions as input, we present a network that learns to interpolate the occluded areas to avoid the erroneous or missing information when computing the motion compensated difference between two disparity maps for scene flow.

2 Related Work

Optical flow estimation with CNNs. Optical flow estimation based on deep learning was pioneered by Dosovitsky et al. [11], who presented an end-to-end trainable encoder-decoder network. The work has been improved by Ilg et al. [18], who introduced a stack of refinement networks. Ranjan and Black [34] focused on efficiency and proposed a much smaller network based on the coarse-to-fine principle. Sun et al. [42] extended this idea by introducing correlations at the different pyramid levels. Their network termed PWC-Net currently achieves state-of-the-art results. The coarse-to-fine approach, however, comes with the well-known limitation that the flow for small, fast-moving objects cannot be estimated. While this does not much affect the average errors of benchmarks, small objects can be very important for decisions in application scenarios.

Disparity estimation with CNNs. For disparity estimation, Zbontar et al. [52] were the first to present a Siamese CNN for matching patches. Post-processing with the traditional SGM method [14] yielded disparity maps. Other approaches to augment SGM with CNNs were presented by [28, 38]. The first end-to-end learning framework was presented by Mayer et al. [29]. The network named DispNetC was derived from the FlowNetC of Dosovitskiy et al. [11] restricted to rectified stereo images. It includes a correlation layer that yields a cost volume, which is further processed by the network. Kendall et al. [21] presented GC-Net, which uses 3D convolutions to process the cost volume also along the disparity dimension and by using a differentiable softargmin operation. Pang et al. [31] extended DispNetC by stacking a refinement network on top, similar to FlowNet 2.0 [18], with the difference that the second network is posed in a residual setting. In this work we also make use of network stacks with up to three networks and use their residual refinement.

Occlusion Estimation. Occlusion and optical flow estimation mutually depend on each other and are thus a typical chicken-and-egg problem [17, 32]. Humayun et al. [16] determine occlusions post-hoc by training a classifier on a broad spectrum of visual features and precomputed optical flow. Pérez-Rúa et al. [32] do not require a dense optical flow field, but motion candidates, which are used to determine if a "plausible reconstruction" exists. Many other methods try to estimate optical flow and occlusions jointly. Leordeanu et al. [27] train a classifier based on various features, including the current motion estimate and use it repeatedly during energy minimization of the flow. Sun et al. [41] make use of superpixels and local layering for an energy formulation that is optimized jointly for layers, optical flow and occlusions. The most recent work from Hur et al. [17] uses consistency between forward and backward flows of two images, by integrating a corresponding constraint into an energy formulation. Since occlusions are directly related to changes in depth [12], it was quite common to consider them explicitly in disparity estimation methods [12, 19, 9, 44].

In this paper, we show that training a network for occlusion estimation is clearly beneficial, especially if the trained network is combined with a network formulation of disparity or optical flow estimation. We do not try to disentangle the chicken-and-egg problem, but instead solve this problem using the joint training procedure.

Depth and motion boundary estimation. In many energy minimization approaches, depth or motion boundary estimation is implicitly included in the form of robustness to outliers in the smoothness constraint. Typically, these boundaries are not made explicit. An exception is Black&Fleet [4], who estimate translational motion together with motion boundaries. Motion boundaries are also explicit in layered motion segmentation approaches. Most of these assume a precomputed optical flow, and only few estimate the segmentation and the flow jointly [40, 8]. Leordeanu et al. [26] introduced a method for combined optimization of a boundary detector that also covers motion boundaries, while most other approaches make use of an external image boundary detector [1, 10]. Sundberg et al. [43] use gPb [1] and LDOF [6] to compute motion differences between regions adjacent to image boundaries. Weinzaepfel et al. [49] use a structured random forest trained on appearance and motion cues. Lei et al. [25] present a fully convolutional Siamese network that is trained on annotated video segmentation. Using only the video segmentation ground-truth for training, they are able to infer the motion of boundary points during inference. For disparity and depth boundaries, the problem is very similar and most of the above mentioned methods could be applied to disparities, too. Jia et al. [20] infer depth boundaries from color and depth images with a Conditional Random Field. In this paper, we obtain depth and motion boundaries also by a joint training procedure and by joint refinement together with occlusions and disparity or flow.

Scene Flow Estimation. Scene flow estimation was popularized for the first time by the work of Vedula et al. [45] and was later dominated by variational methods [15] [33] [47]. Vogel et al. [46] combined the task of scene flow estimation with superpixel segmentation using a piecewise rigid model for regularization.

Schuster et al. [37] proposed a variational approach to interpolate sparse scene flow estimates from sparse matches. Behl et al. [3] proposed a 3D scene flow method, which exploits instance recognition and 3D geometry information to obtain improved performance in texture-less, reflective and fast moving regions.

In this paper, we investigate scene flow estimation based on estimating correspondences only, without the use of 3D geometry information. The only learning based approach in a similar setting was proposed by Mayer et al. [29], but did not perform similarly well.

3 Network Architectures

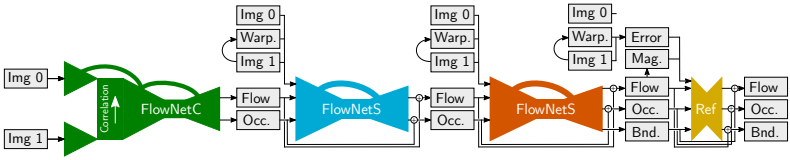
We investigate estimating occlusions and depth or motion boundaries with CNNs together with disparity and optical flow. To this end, we build upon the convolutional encoder-decoder architectures from FlowNet [11] and the stacks from FlowNet 2.0 [18]. Our modifications are shown in Figure 1(a). For simplicity, in the following we mention the flow case. The disparity case is analogous.

In our version of [18], we leave away the small displacement network. In fact, the experiments from our re-implemented version show that the stack can perform well on small displacements without it. We still keep the former fusion network as it also performs smoothing and sharpening (see Figure 1(a)). We denote this network by the letter "R" in network names (e.g. FlowNet-CSSR). This network is only for refinement and does not see the second image. We further modify the stack by integrating the suggestion of Pang et al. [31] and add residual connections to the refinement networks. As in [18], we also input the warped images, but omit the brightness error inputs, as these can easily be computed by the network.

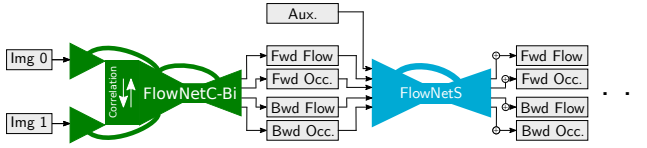
Finally, we add the occlusions and depth or motion boundaries. While occlusions are important for refinement from the beginning, boundaries are only required in later refinement stages. Therefore we add the boundaries in the third network. Experimentally, we also found that when adding depth or motion boundary prediction in earlier networks, these networks predicted details better, but failed more rigorously in case of errors. Predicting exact boundaries early would be contrary to the concept of a refinement pipeline.

Generally, in an occluded area, the forward flow from the first to the second image does not match the backward flow from the second to the first image. If the forward flow is correctly interpolated into the occluded regions, it resembles the flow of the background object. Since this object is not visible in the second image, the backward flow of the target location is from another object and forward and backward flows are inconsistent. Many classical methods use this fact to determine occlusions.

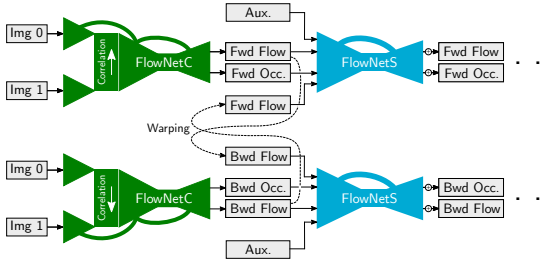
We bring this into the network architecture from Figure 1(b). In this version, we let the network estimate forward and backward flows and occlusions jointly. Therefore we modify FlowNetC to include a second correlation that takes a feature vector from the second image and computes the correlations to a neighborhood in the first image. We concatenate the outputs and also add a second



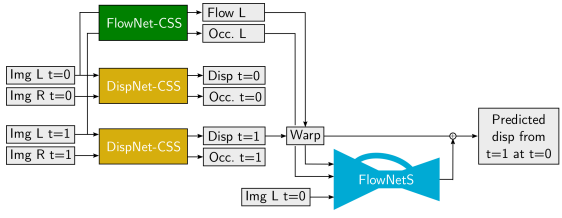
(a) Extension of FlowNet2 with occlusions and residual connections.



(b) Architecture for joint estimation of forward/backward flows and occlusions. See figure caption for symbol explanation.



(c) Dual forward and backward estimation architecture with mutual warping. See figure caption for symbol explanation.



(d) Extending FlowNet-CSS and DispNet-CSS to a full scene flow network.

Fig. 1. Overview of possible refinement stacks for flow, occlusions and motion boundaries. The residual connections are only shown in the first figure and indicated by + elsewhere. Aux. refers to the images plus a warped image for each input flow, respectively. Architectures for the disparity case are analogous.

skip connection for the second image. This setup is shown as FlowNetC-Bi in Figure 1(b). Throughout the stack, we then estimate flow and occlusions in forward and backward directions.

In the third variant from Figure 1(c), we model forward and backward flow estimation as separate streams and perform mutual warping to the other direction after each network. E.g., we warp the estimated backward flow after the first network to the coordinates of the first image using the forward flow. Subsequently we flip the sign of the warped flow, effectively turning it into a forward flow. The network then has the forward and the corresponding backward flow at the same pixel position as input.

Finally, we use our networks for flow and disparity to build a scene flow extension. For the scene flow task, the disparity at $t = 0$ is required and the flow is extended by disparity change [29] (resembling the change in the third coordinate). To compute this disparity change, one can estimate disparities at $t = 1$, warp them to $t = 0$ and compute the difference. However, the warping will be incorrect or undefined everywhere, where an occlusion is present. We therefore add the network shown in Figure 1(d) to learn a meaningful interpolation for these areas given the warped disparity, the occlusions and the image.

4 Experiments

4.1 Training Data

For training our flow networks, we use the FlyingChairs [11], FlyingThings3D [29] and ChairsSDHom [18] datasets. For training the disparity networks, we only use the FlyingThings3D [29] dataset. These datasets do not provide the ground-truth required for our setting per-se. For FlyingChairs, using the code provided by the authors of [11], we recreate the whole dataset including also backward flows, motion boundaries and occlusions. For FlyingThings3D, depth and motion boundaries are directly provided. We use flow and object IDs to determine occlusions. For ChairsSDHom, we compute motion boundaries by finding discontinuities among object IDs and in the flow, by using a flow magnitude difference threshold of 0.75. To determine the ground-truth occlusions, we also use the flow and the object IDs.

4.2 Training Schedules and Settings

For training our networks, we also follow the data and learning rate schedules of FlowNet 2.0 [18]. We train the stack network by network, always fixing the already trained networks. Contrary to [18], for each step we only use half the number of iterations. The initial network is then a bit worse, but it turns out that the refinement can compensate for it well. We also find that the residual networks converge much faster. Therefore, we train each new network on the stack for $600k$ iterations on FlyingChairs and for $250k$ iterations on FlyingThings3D. Optionally, we follow the same fine-tuning procedure for small-displacements on ChairsSDHom as in [18] (we add "-ft-sd" to the network names in this case). We use the caffe framework and the same settings as in [18], with one minor modification: we found that numerically scaling the ground-truth flow vectors

Input	F-measure	
	FlyingThings3D [29]	Sintel clean [7]
Images 0+1	0.790	0.545
Images 0+1, GT fwd Flow	0.932	**0.653**
Images 0+1, GT fwd Flow, GT bwd flow	0.930	-
Images 0+1, GT fwd Flow, GT bwd flow warped+flipped	**0.943**	-
Images 0+1, GT fwd Flow, GT fwd/bwd consistency	**0.943**	-

Table 1. Training a FlowNetS to estimate occluded regions from different inputs. Since Sintel [7] does not provide the ground-truth backward flow, we additionally report numbers on FlyingThings3D [29]. The results show that contrary to literature [32, 27, 16], occlusion estimation is even possible from just the two images. Providing the optical flow, too, clearly improves the results

(by a factor of $\frac{1}{20}$) yields noise for small displacements during optimization. We propose to change this factor to 1. Since these are all minor modifications, we present details in the supplemental material.

To train for flow and disparity, we use the normal EPE loss. For small displacement training we also apply the suggested non-linearity of [18]. To train for occlusions and depth or motion boundaries, we use a normal cross entropy loss with classes 0 and 1 applied to each pixel. To combine multiple losses of different kind, we balance their coefficients during the beginning of the training, such that their magnitudes are approximately equal.

4.3 Estimating Occlusions with CNNs

We first ran some basic experiments on estimating occlusions with a FlowNetS architecture and the described ground-truth data. In the past, occlusion estimation was closely coupled with optical flow estimation and in the literature is stated as "notoriously difficult" [27] and a chicken-and-egg problem [17, 32]. However, before we come to joint estimation of occlusions and disparity or optical flow, we start with a network that estimates occlusions independently of the optical flow or with optical flow being provided as input.

In the most basic case, we only provide the two input images to the network and no optical flow, i.e., the network must figure out by itself on how to use the relationship between the two images to detect occluded areas. As a next step, we additionally provide the ground-truth forward optical flow to the network, to see if a network is able to use flow information to find occluded areas. Since a classical way to detect occlusions is by checking the consistency between the forward and the backward flow as mentioned in Section 3, we provide different versions of the backward flow: 1.) backward flow directly; 2.) using the forward flow to warp the backward flow to the first image and flipping its sign (effectively turning the backward flow into a forward flow up to the occluded areas); 3.) providing the magnitude of the sum of forward and backward flows, i.e., the classical approach to detect occlusions. From the results of these experiments in Table 1, we conclude:

Configuration	EPE	F-measure
FlowNetC estimating flow	3.21	-
FlowNetC estimating occlusions	-	**0.546**
FlowNetC estimating flow + occlusions	**3.20**	0.539
FlowNetC-Bi estimating fwd/bwd flow and fwd occlusions	3.26	0.542

Table 2. Joint estimation of flow and occlusions with a FlowNetC from Sintel train clean. Estimating occlusions neither improves nor degrades flow performance

Occlusion estimation without optical flow is possible. In contrast to existing literature, where classifiers are always trained with flow input [32, 27, 16, 26] or occlusions are estimated jointly with optical flow [41, 17], we show that a deep network can learn to estimate the occlusions directly from two images.

Using the flow as input helps. The flow provides the solution for correspondences and the network uses these correspondences. Clearly, this helps, particularly since we provided the correct optical flow.

Adding the backward flow marginally improves results. Providing the backward flow directly does not help. This can be expected, because the information for a pixel of the backward flow is stored at the target location of the forward flow and a look-up is difficult for a network to perform. Warping the backward flow or providing the forward/backward consistency helps a little.

4.4 Joint Estimation of Occlusions and Optical Flow

Within a single network. In this section we investigate estimating occlusions jointly with optical flow, as many classical methods try to do [41, 17]. Here, we provide only the image pair and therefore can use a FlowNetC instead of a FlowNetS. The first row of Table 2 shows that just occlusion estimation with a FlowNetC performs similar to the FlowNetS of the last section. Surprisingly, from rows one to three of Table 2 we find that joint flow estimation neither improves nor deproves the flow or the occlusion quality. In row four of the table we additionally estimate the backward flow to enable the network to reason about forward/backward consistency. However, we find that this also does not affect performance much.

When finding correspondences, occlusions need to be regarded by deciding that no correspondence exists for an occluded pixel and by filling the occlusion area with some value inferred from the surroundings. Therefore, knowledge about occlusions is mandatory for correspondence and correct flow estimation. Since making the occlusion estimation in our network explicit does not change the result, we conclude that an end-to-end trained network for only flow already implicitly performs all necessary occlusion reasoning. By making it explicit, we obtain the occlusions as an additional output at no cost, but the flow itself remains unaffected.

Configuration	EPE	F-measure
Only flow as in FlowNet2-CS [18]	2.28	-
+ occlusions (Figure 1(a))	**2.25**	**0.590**
+ bwd direction (Figure 1(b))	2.77	0.572
+ mutual warping (Figure 1(c))	2.25	0.589

Table 3. Results of refinement stacks on Sintel train clean. Simply adding occlusions in a straightforward manner performs better or similar to more complicated approaches. In general, adding occlusions does not perform better than estimating only flow

4.5 With a refinement network

In the last section we investigated the joint estimation of flow and occlusions, which in the literature is referred to as a "chicken-and-egg" problem. With our first network already estimating flow and occlusions, we investigate if the estimated occlusions can help refine the flow ("if a chicken can come from an egg").

To this end, we investigate the three proposed architectures from Section 2. We show the results of the three variants in Table 3. While the architectures from Figures 1(a) and 1(c) are indifferent about the additional occlusion input, the architecture with joint forward and backward estimation performs worse.

Overall, we find that providing explicit occlusion estimates to the refinement does not help compared to estimating just the optical flow. This means, either the occluded areas are already filled correctly by the base network, or in a stack without explicit occlusion estimation, the second network can easily recover occlusions from the flow and does not require the explicit input.

We finally conclude that occlusions can be obtained at no extra cost, but do not actually influence the flow estimation, and that it is best to leave the inner workings to the optimization by using only the baseline variant (Figure 1(a)). This is contrary to the findings from classical methods.

4.6 Comparing occlusion estimation to other methods

In Tables 4 and 5 we compare our occlusion estimations to other methods. For disparity our method outperforms Kolmogorov et al. [24] for all except one scene. For the more difficult case of optical flow, we outperform all existing methods by far. This shows that the chicken-and-egg problem of occlusion estimation is much easier to handle with a CNN than with classical approaches [44, 24, 17, 27] and that CNNs can perform very well at occlusion reasoning. This is confirmed by the qualitative results of Figure 2. While consistency checking is able to capture mainly large occlusion areas, S2DFlow [27] also manages to find some details. MirrorFlow [17] in many cases misses details. Our CNN on the other hand is able to estimate most of the fine details.

Method	F-Measure					
	Cones	Teddy	Tsukuba	Venus	Sintel clean	Sintel final
Kolmogorov et al.[24]	0.45	**0.63**	0.60	0.41	-	-
Tan et al.[44]	0.44	0.40	0.50	0.33	-	-
Ours	**0.91**	0.57	**0.68**	**0.44**	**0.76**	**0.72**

Table 4. Comparison of estimated disparity occlusions from our DispNet-CSS to other methods on examples from the Middlebury 2001 and 2003 datasets (results of Kolmogorov et al. [24] and Tan et al. [44] taken from [44]) and the Sintel train dataset. Only in the scene Teddy of Middlebury our occlusions are outperformed by Kolmogorov et al. [24]

Method	Type	F-Measure	
		clean	final
FlowNet2 [18]	consistency	0.377	0.348
MirrorFlow [17]	estimated	0.390	0.348
S2DFlow [27]	estimated	0.470	0.403
Ours	estimated	**0.703**	**0.654**

Table 5. Comparison of the occlusions from FlowNet-CSSR-ft-sd to other occlusion estimation methods on the Sintel train dataset. For the first entry, occlusions were computed using forward/backward consistency post-hoc. The proposed approach yields much better occlusions

4.7 Motion Boundary Estimation

For motion boundary estimation we compare to Weinzaepfel et al. [49], which is to the best of our knowledge the best available method. It uses a random forest classifier and is trained on the Sintel dataset. Although we do not train on Sintel, from the results of Table 6, our CNN outperforms their method by a large margin. The improvement in quality is also very well visible from the qualitative results from Figure 3.

4.8 Application to Motion Segmentation

We apply the estimated occlusions to the motion segmentation framework from Keuper et al. [22]. This approach, like [5], computes long-term point trajectories based on optical flow. For deciding when a trajectory ends, the method depends on reliable occlusion estimates. These are commonly computed using the post-hoc consistency of forward and backward flow, which was shown to perform badly in Section 4.6. We replace the occlusion estimation with the occlusions from our FlowNet-CSS. Table 3 shows the clear improvements obtained by the more reliable occlusion estimates on the common FBMS-59 motion segmentation benchmark. In row four, we show how adding our occlusions to flow estimations of FlowNet2 can improve results. This shows that by only adding occlusions, we

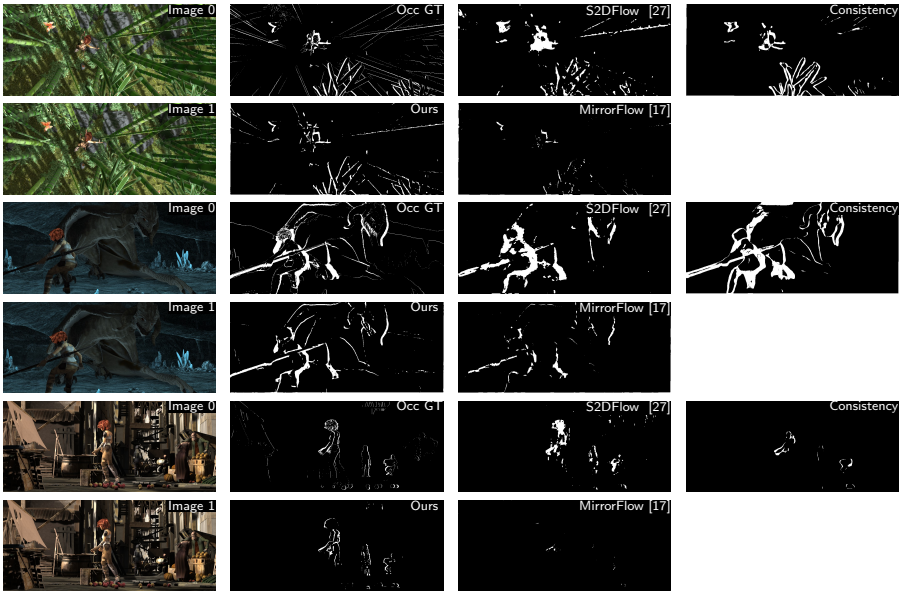


Fig. 2. Qualitative results for occlusions. In comparison to other methods and the forward-backward consistency check, our method is able to capture very fine details.

Method	Sintel clean	Sintel final
Weinzaepfel et al. [49]	76.3	68.5
Ours	**86.3**	**79.5**

Table 6. Comparison of our motion boundary estimation to Weinzaepfel et al. [49] on the Sintel train dataset. The table shows the mean average precision computed with their evaluation code. Although Weinzaepfel et al. [49] trained on Sintel train clean, our method outperforms theirs by a large margin

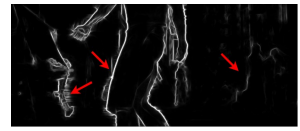
recover 30 objects instead of 26. The last result from our flow and occlusions together further improve the results. Besides the direct quantitative and qualitative evaluation from the last sections, this shows the usefulness of our occlusion estimates in a relevant application. Our final results can produce results that are even better than the ones generated by the recently proposed third order motion segmentation with multicuts [23].

4.9 Benchmark results for disparity, optical flow, and scene flow

Finally, we show that besides the estimated occlusions and depth and motion boundaries, our disparities and optical flow achieve state-of-the-art performance. In Table 8 we show results for the common disparity benchmarks. We also present smaller versions of our networks by scaling the number of channels in each layer


(a) Image 0


(b) Boundary ground-truth


(c) Weinzaepfel et al. [49]


(d) Flow ground-truth


(e) Ours (hard)


(f) Ours(soft)

Fig. 3. Motion boundaries on Sintel train clean. Our approach succeeds to detect the object in the background and has less noise around motion edges than existing approaches (see green arrows). Weinzaepfel et al. detect some correct motion details in the background. However, these details are not captured in the ground-truth.

Method	FBMS test set (30 sequences)			
	Precision	Recall	F-Measure	#Objects
Third Order Multicut [23]	87.77%	71.96%	79.08%	29/69
DeepFlow [48]	88.20%	69.39%	77.67%	26/69
FlowNet2	86.73%	68.77%	76.71%	26/69
FlowNet2 + our occ	85.67%	70.15%	77.14%	30/69
Ours	**88.71%**	**73.60%**	**80.45%**	**31/69**

Table 7. Results of motion segmentation from Keuper et al. [22] on the FBMS-59 test set [5, 30] (with sampling density 8px). The fourth row uses flows from FlowNet2 [18] combined with our occlusions. The improved results show that occlusions help the motion segmentation in general. The last row shows the segmentation using our flow and occlusions, which performs best and also improves over the recent state-of-the-art on sparse motion segmentation using higher order motion models [23]

down to 37.5% as suggested in [18] (denoted by css). While this small version yields a good speed/accuracy trade-off, the larger networks rank second on KITTI 2015 and are the top ranked methods on KITTI 2012 and Sintel.

In Table 9 we show the benchmark results for optical flow. We perform on-par on Sintel, while we set the new state-of-the-art on both KITTI datasets.

In Table 10 we report numbers on the KITTI 2015 scene flow benchmark. The basic scene flow approach warps the next frame disparity maps into the current frame (see [36]) using forward flow. Out of frame occluded pixels cannot be estimated this way. To mitigate this problem we train a CNN to reason about disparities in occluded regions (see the architecture from Figure 1(d)). This yields clearly improved results that get close to the state-of-the-art while the approach is orders of magnitude faster.

Method	**Sintel** (clean) AEE *train*	**KITTI** (2012) AEE *train*	Out-noc *test*	**KITTI** (2015) AEE *train*	D1-all *test*	Runtime *(s)*
Standard						
SGM [14]	19.62	10.06	-	7.21	10.86%	1.1
CNN based						
DispNetC [29]	5.66	1.75	-	1.59	-	0.06
DispNetC-ft [29]	21.88	1.48	4.11%	(0.68)	4.34%	0.06
CRL [31]	16.13	1.11	-	(0.52)	2.67%	0.47
GC-Net [21]	-	-	**1.77**%	-	2.87%	0.90
MC-CNN-acrt [52]	-	-	2.43%	-	3.89%	67
DRR [13]	-	-	-	-	3.16%	0.4
L-ResMatch [39]	-	-	2.27%	-	3.42%	42
With joint occ. est.						
SPS stereo [51]	-	-	3.39%	-	5.31%	2
Our DispNet-CSS	**2.33**	1.40	-	1.37	-	0.07
Our DispNet-CSS-ft	5.53	(0.72)	1.82%	(0.71)	**2.19**%	0.07
Our DispNet-css	2.95	1.53	-	1.49	-	0.03

Table 8. Benchmark results for **disparity estimation**. We report the average endpoint error (AAE) for Sintel. On KITTI, Out-noc and D1-all are used for the benchmark ranking on KITTI 2012 and 2015, respectively. Out-noc shows the percentage of outliers with errors more than 3px in non-occluded regions, whereas D1-all shows the percentage in all regions. Entries in parentheses denote methods that were finetuned on the evaluated dataset. Our network denoted with "-ft" is finetuned on the respective training datasets. We obtain state-of-the-art results on the Sintel and KITTI 2015. Also, our networks generalize well across domains, as shown by the good numbers of the non-finetuned networks and the reduced drop in performance for a network finetuned on KITTI and tested on Sintel

5 Conclusion

We have shown that, in contrast to traditional methods, CNNs can very easily estimate occlusions and depth or motion boundaries, and that their performance surpasses traditional approaches by a large margin. While classical methods often use the backward flow to determine occlusions, we have shown that a simple extension from the forward FlowNet 2.0 stack performs best in the case of CNNs. We have also shown that this generic network architecture performs well on the tasks of disparity and flow estimation itself and yields state-of-the-art results on benchmarks. Finally, we have shown that the estimated occlusions can significantly improve motion segmentation.

Method	Sintel (clean) AEE		Sintel (final) AEE		KITTI (2012)		KITTI (2015)		Runtime (s)
	train	*test*	*train*	*test*	*AEE train*	*OUT-noc test*	*AEE train*	*F1-all test*	
Standard									
EpicFlow [35]	2.27	4.12	3.56	6.29	**3.09**	7.88%	9.27	26.29%	42
FlowfieldsCNN [2]	-	3.78	-	5.36	-	4.89%	-	18.68%	23
DCFlow [50]	-	**3.54**	-	5.12	-	-	-	14.86%	9
CNN based									
FlowNet2 [18]	**2.02**	3.96	**3.14**	6.02	4.09	-	10.06	-	0.123
FlowNet2-ft [18]	(1.45)	4.16	(2.01)	5.74	(1.28)	-	(2.30)	11.48%	0.123
SpyNet [34]	4.12	6.69	5.57	8.43	9.12	-	-	-	**0.016**
SpyNet-ft [34]	(3.17)	6.64	(4.32)	8.36	(4.13)	12.31%	-	35.07%	**0.016**
PWC-Net [42]	2.55	-	3.93	-	4.14	-	10.35	33.67%	0.030
PWC-Net-ft [42]	(2.02)	4.39	(2.08)	**5.04**	-	4.22%	(2.16)	9.80%	0.030
With joint occ est.									
MirrorFlow [17]	-	3.32	-	6.07	-	4.38%	-	10.29%	660
S2D flow [27]	-	18.48	-	6.82	-	-	-	-	2280
Our FlowNet-CSS	2.08	3.94	3.61	6.03	3.69	-	**9.33**	-	0.068
Our FlowNet-CSS-ft	(1.47)	4.35	(2.12)	5.67	(1.19)	**3.45**%	(1.79)	**8.60**%	0.068
Our FlowNet-css	2.65	-	4.05	-	5.05	-	11.74		0.033

Table 9. Benchmark results for **optical flow estimation**. We report the average endpoint error (AAE) for all benchmarks, except KITTI, where Out-noc and F1-all are used for the benchmark ranking on KITTI 2012 and 2015, respectively. Out-noc shows the percentage of outliers with errors more than 3px in non-occluded regions, whereas F1-all shows the percentage in all regions. Entries in parentheses denote methods that were finetuned on the evaluated dataset. On the Sintel dataset, the performance of our networks is on par with FlowNet2. When comparing to other methods with joint occlusion estimation we are faster by multiple orders of magnitude. On KITTI 2012 and 2015 we obtain state-of-the-art results among all optical flow methods (two frame, non-stereo)

Method	D1-all	D2-all	Fl-all	SF-all	Runtime (s)
ISF [3]	4.46	5.95	6.22	8.08	600
SGM+FlowFields (interp.)[36]	13.37	27.80	22.82	33.57	29
SceneFFields (dense) [37]	6.57	10.69	12.88	15.78	65
Ours(interp.)	2.16	13.71	8.60	17.73	0.22
Ours(dense)	2.16	6.45	8.60	11.34	0.25

Table 10. Benchmark results for **scene flow estimation**. "Interp." means the disparity values were automatically interpolated by the KITTI benchmark suite in the sparse regions. Compared to [37] we obtain much improved results and close the performance gap to much slower state-of-the-art methods, such as [3], which use 2D information by a large margin

Acknowledgements

We acknowledge funding by the EU Horizon2020 project TrimBot2020 and by Gala Sports, and donation of a GPU server by Facebook. Margret Keuper acknowledges funding by DFG grant KE 2264/1-1.

References

1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. PAMI **33**(5) (May 2011)
2. Bailer, C., Varanasi, K., Stricker, D.: Cnn-based patch matching for optical flow with thresholded hinge embedding loss. CVPR (2017)
3. Behl, A., Jafari, O.H., Mustikovela, S.K., Alhaija, H.A., Rother, C., Geiger, A.: Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? In: International Conference on Computer Vision (ICCV) (2017)
4. Black, M.J., Fleet, D.J.: Probabilistic detection and tracking of motion boundaries. International Journal of Computer Vision **38**(3), 231–245 (Jul 2000)
5. Brox, T., Malik, J.: Object segmentation by long term analysis of point trajectories. In: ECCV (2010), http://lmb.informatik.uni-freiburg.de//Publications/2010/Bro10c
6. Brox, T., Malik, J.: Large displacement optical flow: descriptor matching in variational motion estimation. PAMI **33**(3), 500–513 (2011)
7. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: European Conference on Computer Vision (ECCV) (2012)
8. Cheng, J., Tsai, Y.H., Wang, S., Yang, M.H.: Segflow: Joint learning for video object segmentation and optical flow (2017)
9. Deng, Y., Yang, Q., Lin, X., Tang, X.: Stereo correspondence with occlusion handling in a symmetric patch-based graph-cuts model (2007)
10. Dollár, P., Zitnick, C.L.: Structured forests for fast edge detection (2013)
11. Dosovitskiy, A., Fischer, P., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., v.d. Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: IEEE Int. Conference on Computer Vision (ICCV) (2015)
12. Geiger, D., Ladendorf, B., Yuille, A.: Occlusions and binocular stereo. IJCV (1995)
13. Gidaris, S., Komodakis, N.: Detect, replace, refine: Deep structured prediction for pixel wise labeling. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
14. Hirschmüller, H.: Stereo processing by semiglobal matching and mutual information. PAMI **30**(2), 328–341 (2008)
15. Huguet, F., Devernay, F.: A variational method for scene flow estimation from stereo sequences. In: 2007 IEEE 11th International Conference on Computer Vision. pp. 1–7 (Oct 2007). https://doi.org/10.1109/ICCV.2007.4409000
16. Humayun, A., Aodha, O.M., Brostow, G.J.: Learning to find occlusion regions. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2011)
17. Hur, J., Roth, S.: Mirrorflow: Exploiting symmetries in joint optical flow and occlusion estimation (2017)
18. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: Evolution of optical flow estimation with deep networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

19. Ishikawa, H., Geiger, D.: Global optimization using embedded graphs (2000)
20. Jia, Z., Gallagher, A., Chen, T.: Learning boundaries with color and depth (2013)
21. Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression (2017)
22. Keuper, M., Andres, B., Brox, T.: Motion trajectory segmentation via minimum cost multicuts. In: ICCV (2015)
23. Keuper, M.: Higher-order minimum cost lifted multicuts for motion segmentation. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
24. Kolmogorov, V., Monasse, P., Tan, P.: Kolmogorov and Zabih's Graph Cuts Stereo Matching Algorithm. Image Processing On Line **4**, 220–251 (Oct 2014). https://doi.org/10.5201/ipol.2014.97, https://hal-enpc.archives-ouvertes.fr/hal-01074878
25. Lei, P., Li, F., Todorovic, S.: Joint spatio-temporal boundary detection and boundary flow prediction with a fully convolutional siamese network. CVPR (2018)
26. Leordeanu, M., Sukthankar, R., Sminchisescu, C.: Efficient closed-form solution to generalized boundary detection. In: European Conference on Computer Vision (ECCV) (2012)
27. Leordeanu, M., Zanfir, A., Sminchisescu, C.: Locally affine sparse-to-dense matching for motion and occlusion estimation. In: IEEE Int. Conference on Computer Vision (ICCV) (2013)
28. Luo, W., Schwing, A.G., Urtasun, R.: Efficient deep learning for stereo matching (2016)
29. N.Mayer, E.Ilg, P.Häusser, P.Fischer, D.Cremers, A.Dosovitskiy, T.Brox: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
30. Ochs, P., Malik, J., Brox, T.: Segmentation of moving objects by long term video analysis. IEEE TPAMI **36**(6), 1187 – 1200 (Jun 2014)
31. Pang, J., Sun, W., Ren, J.S.J., Yang, C., Yan, Q.: Cascade residual learning: A two-stage convolutional neural network for stereo matching. In: IEEE Int. Conference on Computer Vision (ICCV) Workshop (2017)
32. Perez-Rua, J.M., Crivelli, T., Bouthemy, P., Perez, P.: Determining occlusions from space and time image reconstructions. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
33. Quiroga, J., Brox, T., Devernay, F., Crowley, J.L.: Dense Semi-Rigid Scene Flow Estimation from RGBD images. In: ECCV 2014 - European Conference on Computer Vision. Zurich, Switzerland (Sep 2014). https://doi.org/10.1007/978-3-319-10584-0_37, https://hal.inria.fr/hal-01021925
34. Ranjan, A., Black, M.: Optical flow estimation using a spatial pyramid network. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
35. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: Epicflow: Edge-preserving interpolation of correspondences for optical flow. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
36. Schuster, R., Bailer, C., Wasenmller, O., Stricker, D.: Combining stereo disparity and optical flow for basic scene flow. In: Commercial Vehicle Technology Symposium (CVTS) (2018)
37. Schuster, R., Wasenmller, O., Kuschk, G., Bailer, C., Stricker, D.: Sceneflowfields: Dense interpolation of sparse scene flow correspondences. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2018)

38. Seki, A., Pollefeys, M.: Sgm-nets: Semi-global matching with neural networks (2017)
39. Shaked, A., Wolf, L.: Improved stereo matching with constant highway networks and reflective confidence learning. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
40. Sun, D., Liu, C., Pfister, H.: Local layering for joint motion estimation and occlusion detection (2014)
41. Sun, D., Liu, C., Pfister, H.: Local layering for joint motion estimation and occlusion detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
42. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. CVPR (2018)
43. Sundberg, P., Brox, T., Maire, M., Arbelez, P., Malik, J.: Occlusion boundary detection and figure/ground assignment from optical flow. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2011)
44. Tan, P., Chambolle, A., Monasse, P.: Occlusion detection in dense stereo estimation with convex optimization. In: 2017 IEEE International Conference on Image Processing (ICIP). pp. 2543–2547 (Sept 2017). https://doi.org/10.1109/ICIP.2017.8296741
45. Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Three-dimensional scene flow. IEEE Trans. Pattern Anal. Mach. Intell. **27**(3), 475–480 (Mar 2005). https://doi.org/10.1109/TPAMI.2005.63, http://dx.doi.org/10.1109/TPAMI.2005.63
46. Vogel, C., Schindler, K., Roth, S.: Piecewise rigid scene flow. In: 2013 IEEE International Conference on Computer Vision. pp. 1377–1384 (Dec 2013). https://doi.org/10.1109/ICCV.2013.174
47. Wedel, A., Brox, T., Vaudrey, T., Rabe, C., Franke, U., Cremers, D.: Stereoscopic scene flow computation for 3d motion understanding. International Journal of Computer Vision **95**(1), 29–51 (Oct 2011). https://doi.org/10.1007/s11263-010-0404-0, https://doi.org/10.1007/s11263-010-0404-0
48. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: ICCV - IEEE International Conference on Computer Vision. pp. 1385–1392. IEEE, Sydney, Australia (Dec 2013). https://doi.org/10.1109/ICCV.2013.175, https://hal.inria.fr/hal-00873592
49. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: Learning to Detect Motion Boundaries. In: CVPR 2015 - IEEE Conference on Computer Vision & Pattern Recognition. Boston, United States (Jun 2015), https://hal.inria.fr/hal-01142653
50. Xu, J., Ranftl, R., Koltun, V.: Accurate Optical Flow via Direct Cost Volume Processing. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
51. Yamaguchi, K., McAllester, D., Urtasun, R.: Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In: ECCV (2014)
52. Zbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. Journal of Machine Learning Research (2016)

Supplementary Material

1 Video

Please see the supplementary video for qualitative results on a number of video sequences at https://www.youtube.com/watch?v=SwOdSaBRysI.

2 Visual Examples

Figures 3, 4 and 5 give examples of our motion boundary, occlusion and flow estimation on some real images. We show our FlowNet-CSS and our FlowNet-CSSR-ft-sd, which includes fine-tuning on small displacements and the final refinement network. One can observe that already FlowNet-CSS performs well, while FlowNet-CSSR-ft-sd provides smoother flow estimations and a bit more details. Generally, one can observe that motion boundaries are estimated well, indicating the usefulness for motion segmentation. This is also visible in the provided video.

Figure 6 shows the case for depth boundary, occlusion and disparity estimation on some examples from Sintel. One can observe that the estimations are very close to the ground-truth and hard to distinguish at first glance. Note how well the DispNet-CSS architecture can estimate the large occlusion areas in the first, and the fine details in the second example.

Figure 7 shows results from the network proposed for filling occluded areas for scene flow. Note that occlusion in general causes hallucination effects (visible e.g. for the trees on the right) and missing values at the boundaries. One can observe that our network is able to remove the hallucination effects and that it provides a meaningful extrapolation to the missing values.

3 Losses

3.1 Scaling of Predictions

For training CNNs it is common to normalize input and output data to a common range. Let $\mathbf{f}$ be the output of the network, $\mathbf{y}^{\text{gt}}$ the ground-truth and $\mathbf{y}$ our prediction. The original FlowNet [11, 18] provided the following implementation:

$$\min \mathcal{L}(\frac{1}{20} \cdot \mathbf{y}^{\text{gt}}, \mathbf{f}) \,,$$

$$\mathbf{y} = 20 \cdot \mathbf{f} \,,$$

(a) FlowNet2-C [18] (b) Scaling (c) Ground-truth

Fig. 1. Comparison of FlowNets trained with different scalings on a Sintel example. (a) shows the first network of the stack published in FlowNet2 [18]. (b) shows the same network with the scaling as proposed by us. (c) shows the ground-truth. Note how the scaling significantly removes noise for small displacements

where $\mathcal{L}$ is the loss function. In other words, the ground-truth was scaled down by a factor of 20. This leads to very small values in the network. We instead propose to scale up the values inside the network by removing the coefficient:

$$\min \mathcal{L}(\mathbf{y}^{\text{gt}}, \mathbf{f}) \,,$$

$$\mathbf{y} = \mathbf{f} \,.$$

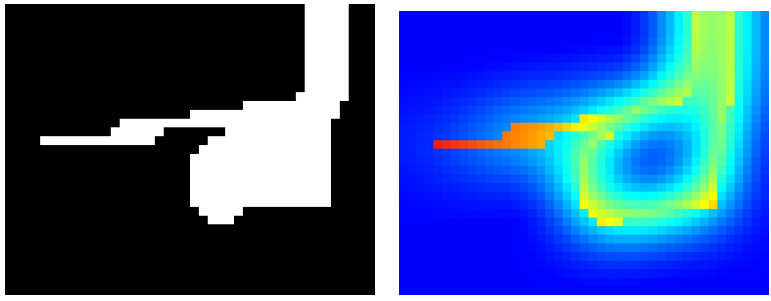
We show a visual example of both scalings in Figure 1. As visible, removing the scaling produces much better results in the case of small displacements. From the first two rows of Table 2 we also see that changing the scaling has no big effect on the EPE. For disparity, the EPE even decreases a bit. Also for the examples from Figures 3, 4 and 5, our network produces very good results for small displacements even without the small displacement fine-tuning and even without the extra small displacement network of FlowNet2 [18].

Note that within the decoder of the network the predictions are always upscaled and concatenated to further decoder stages. Thus, the optimization needs to scale internal activations to fit the ranges of the inserted predictions. In general, such a scale can be arbitrary. However, we conjecture that the different effects come from weight decay, which can yield different results when operating with different ranges.

3.2 Weight Maps for Occlusions and Boundaries

We predict occlusions by using the cross-entropy loss for the two classes occluded and non-occluded. We observed that such a network learns to correctly identify large occluded and non-occluded areas, but tends to ignore thin regions. This is because small regions contribute much less to the total loss.

In general, more non-occluded than occluded pixels are present, thus, the classes are imbalanced. A first solution is to weigh the occluded pixels higher to balance the classes, but this can lead to overflow effects, e.g., pixels of a thin occlusion area have a high weight and are thus more likely to be predicted correctly, but surrounding non-occluded pixels have a low weight and are more easily predicted also as occluded. We propose the following weighting that counteracts this effect:



(a) Occlusions (b) Weighting

Fig. 2. Illustration of weighting for occlusion and boundary loss. Left image shows occlusion ground-truth, right image shows weighting visualized as a heatmap. The weight is higher, the more surrounding pixels have a different occlusion value

Correlation level	EPE
2	5.62
3	3.19

Table 1. Impact of correlation level in DispNetC (the level indicates after which convolution the correlation is defined in the network). We observe a significant performance improvement by just changing the position of the correlation layer to be after the third convolution

$$w(x,y) = \frac{\sum\limits_{i,j \in \mathcal{N}} \delta_{o(x,y) \neq o(i,j)} \cdot g(x-i)g(y-j)}{\sum\limits_{i,j \in \mathcal{N}} g(x-i)g(y-j)} ,$$

$$\text{with } g(d) = e^{-\frac{d^2}{2\sigma^2}} ,$$

where $\mathcal{N}$ is a neighborhood and $\delta_{o(x,y)=o(i,j)}$ determines whether the neighboring pixel has the same occlusion value as the center pixel. The weight determined for the current pixel is the highest if all surrounding pixels have different occlusion value and decreases as surrounding pixels have similar occlusion value. Neighbors are weighted with a Gaussian with parameter σ according to their distance. We show an example of the weights in Figure 2. For boundary estimation, there is a similar problem. Thus, we apply the same weights.

4 Ablation Study of Network Configurations

We first change the DispNet architecture by moving the correlation layer up one level. This results mainly in larger strides and correlation distances. From the

results of Table 1, one can see that this significantly lowers the endpoint error by almost 50%. For an example of improved performance in large displacement regions we refer to the supplementary video.

In Table 2 we provide an ablation study for building our for flow and disparity estimation (in this study we exclude occlusions and boundaries and see only the effects of the stack construction).

In the first step we show that scaling as proposed in Section 3.1 has no big effect on the EPE. In the case of disparity it even improves the results. Adding the second network with residual connections [31] clearly gives an improvement over the normal stacking. As the second and the third network have similar tasks, it turns out that to train the third network, it is beneficial to copy the weights of the second network for initialization. Although we do not train with the full schedule proposed in [18], our final result for FlowNet-CSS is the same. For the case of disparity, our stack significantly improves over DispNetC [29] (see results in main paper).

Configuration	Pred. scaling	Residual refinement	Weight copy	Flow EPE	Disparity EPE
C	No	No	No	3.154	3.335
C	Yes	No	No	3.208	3.194
CS	Yes	No	No	2.340	2.634
CS	Yes	Yes	No	2.280	2.494
CSS	Yes	No	No	2.234	-
CSS	Yes	Yes	No	2.115	2.476
CSS	Yes	Yes	Yes	2.042	2.361

Table 2. Ablation study for training details of a single network and network stacks. Reported errors are from the Sintel train clean dataset. We train each network with the schedule of $600k$ iterations on FlyingChairs [11] and $250k$ iterations on FlyingThings3D [29]. For details see text

5 Motion Segmentation Results

In Table 3 we show results of motion segmentation additionally for the 4px density evaluation. One can observe that the behaviour for the dense version is similar and our approach also performs better in this case.

6 Training Settings for KITTI

We fine-tune each network in our stack individually on KITTI. After fine-tuning the first network, we fix the weights and then fine-tune the second network. This process is repeated for all networks in the stack. We train on a mixture of KITTI

Method	**Test set** (30 sequences)				
	Density	Precision	Recall	F-Measure	#Objects
DeepFlow [48]	0.89%	88.20%	69.39%	77.67%	26/69
FlowNet2	0.85%	86.73%	68.77%	76.71%	26/69
FlowNet2+occ	0.86%	85.67%	70.15%	77.14%	30/69
FlowNetX+occ	0.84%	**88.71%**	**73.60%**	**80.45%**	**31/69**
DeepFlow [48]	3.79%	88.58%	68.46%	77.23%	27/69
FlowNet2	3.66%	87.16%	68.51%	76.72%	26/69
FlowNet2+occ	3.71%	86.29%	69.72%	77.13%	29/69
FlowNetX+occ	3.61%	**89.12%**	**72.77%**	**80.12%**	**32/69**

Table 3. Results of motion segmentation from Keuper et al. [22] on the FBMS-59 test set [5, 30] (with sampling densities 8 and 4px). The third and seventh rows use flows from FlowNet2 [18] combined with our occlusions. The improved results show that occlusions help the motion segmentation in general. The fourth and last rows show the segmentation using our flow and occlusions, which yields the best performance, also in the 4px density case

2012 and 2015 data, which is split into training and validation sets (with sizes 75%/25%). Each network in the stack is trained for 200k iterations with a base learning rate of $1e-5$.

The last network in the stack requires motion boundary labels. Since KITTI does not have ground-truth motion boundary labels, we pre-compute the raw motion boundary features estimated from our network and then tie the predictions to these features using an L2 loss. In this way the network does not forget the already learned boundaries in the case when ground-truth is absent.

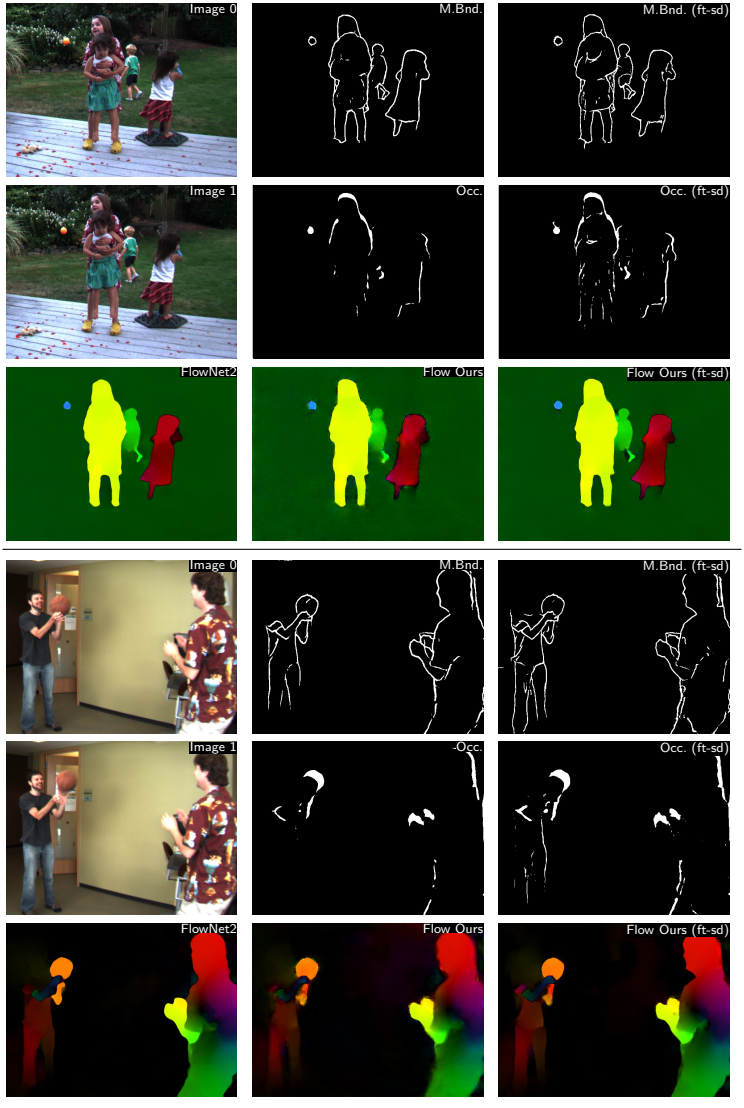


Fig. 3. Examples of our joint motion boundary, occlusion and optical flow estimation on some real images. We provide estimations for FlowNet-CSS and FlowNet-CSSR-ft-sd. One can observe that our method provides very sharp estimations and that the version fine-tuned for small displacements provides a bit more details. In the top example the occlusions from the child in the background become visible

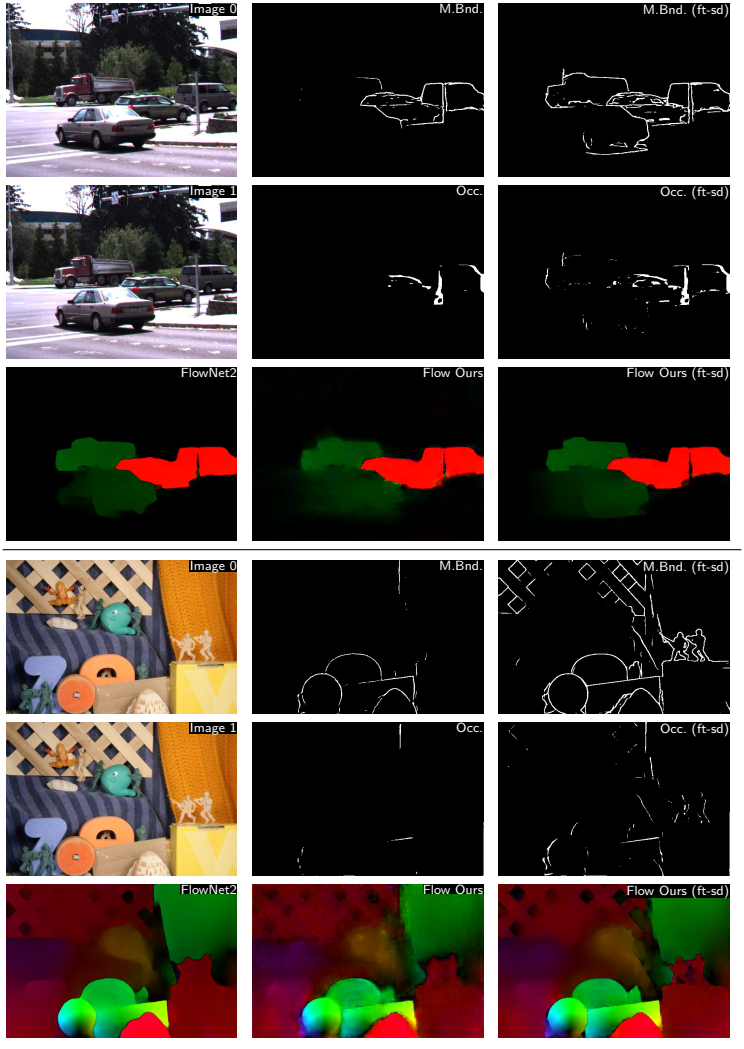


Fig. 4. More examples of our joint motion boundary, occlusion and optical flow estimation on some real images. We provide estimations for FlowNet-CSS and FlowNet-CSSR-ft-sd. One can observe that our method provides very sharp estimations and that the version fine-tuned for small displacements provides a bit more details. In the top example, the boundaries of the cars become apparent. Note the fine details, such as the exhaust pipe of the truck. In the bottom example, motion from the background pattern is visible in the fine-tuned version

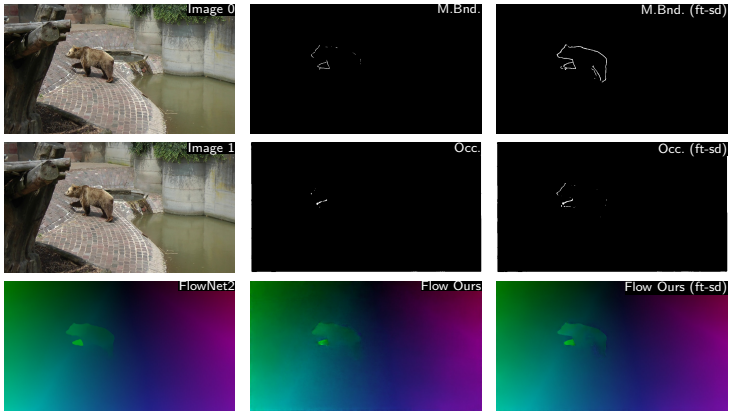


Fig. 5. One more example of our joint motion boundary, occlusion and optical flow estimation on some real images. We provide estimations for FlowNet-CSS and FlowNet-CSSR-ft-sd. One can observe that our method provides very sharp estimations and that the version fine-tuned for small displacements provides a bit more details. In this case the motion boundaries estimated from FlowNet-CSSR-ft-sd segment the bear from the background very well

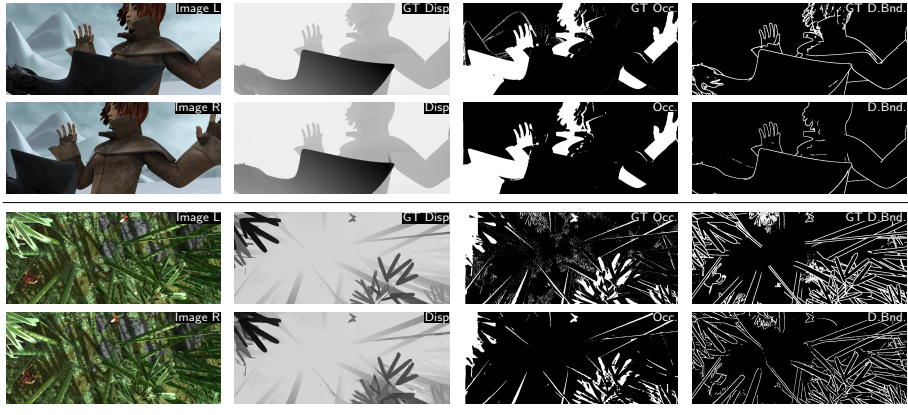


Fig. 6. Examples from our DispNet-CSS for joint depth boundary, occlusion and disparity estimation on some Sintel images. The estimations from our method are in general very close to the ground-truth

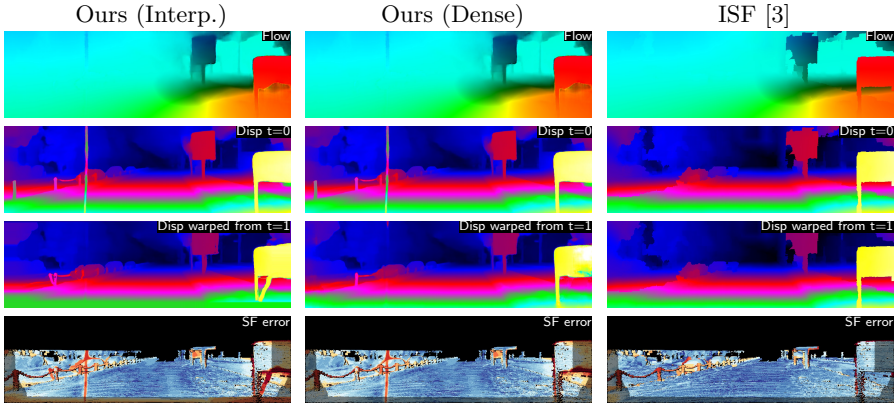


Fig. 7. Example of our occlusion filling network for scene flow. We compare our results on scene flow estimation with the current state of art on KITTI [3] (right) directly with the visualizations from the KITTI benchmark. The first row shows the estimated optical flow on the left images from $t = 0$ to $t = 1$. The second row shows disparity at $t = 0$ with left image as the reference frame. The third row shows the disparity at $t = 1$ warped to $t = 0$ using in the forward flow (shown in the first row). The last row shows the scene flow error map from the KITTI benchmark, where the occluded regions have a dark overlay.
The first column shows results of sparse predictions. The interpolation into occluded regions is the default from the KITTI benchmark. The second column shows the results when using our additional network to fill the occlusion areas and the third column visualizes the results from ISF [3]. We can observe that our scene flow architecture learns to fill reasonable disparity values in the occluded regions. The error in the bottom occlusion area is significantly lower. Also, note that the hallucination effects at the road sign (yellow) are removed by our network