Improving Optical Flow on a Pyramid Level

Markus Hofinger[†,‡][0000−0002−9033−9192], Samuel Rota Bulò[†][0000−0002−2372−1367], Lorenzo Porzi[†][0000−0001−9331−2908], Arno Knapitsch[†][0000−0002−5249−8889], Thomas Pock[‡][0000−0001−6120−1058], and Peter Kontschieder[†][0000−0002−9809−664X]

Mapillary Research[†], Graz University of Technology[‡]
research@mapillary.com[†], {markus.hofinger,pock}@icg.tugraz.at[‡]

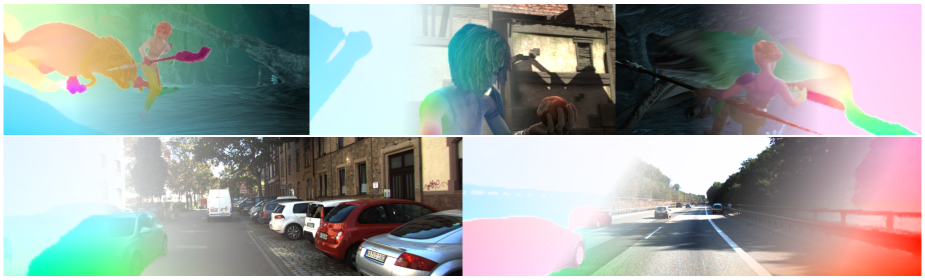

Fig. 1: Optical flow predictions from our model on images from Sintel and KITTI.

Abstract. In this work we review the coarse-to-fine spatial feature pyramid concept, which is used in state-of-the-art optical flow estimation networks to make exploration of the pixel flow search space computationally tractable and efficient. Within an individual pyramid level, we improve the cost volume construction process by departing from a warping- to a sampling-based strategy, which avoids ghosting and hence enables us to better preserve fine flow details. We further amplify the positive effects through a level-specific, loss max-pooling strategy that adaptively shifts the focus of the learning process on under-performing predictions. Our second contribution revises the gradient flow across pyramid levels. The typical operations performed at each pyramid level can lead to noisy, or even contradicting gradients across levels. We show and discuss how properly blocking some of these gradient components leads to improved convergence and ultimately better performance. Finally, we introduce a distillation concept to counteract the issue of catastrophic forgetting during finetuning and thus preserving knowledge over models sequentially trained on multiple datasets. Our findings are conceptually simple and easy to implement, yet result in compelling improvements on relevant error measures that we demonstrate via exhaustive ablations on datasets like Flying Chairs2, Flying Things, Sintel and KITTI. We establish new state-of-the-art results on the challenging Sintel and KITTI 2012 test datasets, and even show the portability of our findings to different optical flow and depth from stereo approaches.

1 Introduction

State-of-the-art, deep learning based optical flow estimation methods share a number of common building blocks in their high-level, structural design. These blocks reflect insights gained from decades of research in *classical* optical flow estimation, while exploiting the power of deep learning for further optimization of *e.g.* performance, speed or memory constraints [14, 37, 44]. Pyramidal representations are among the fundamental concepts that were successfully used in optical flow and stereo matching works like [3]. However, while pyramidal representations enable computationally tractable exploration of the pixel flow search space, their downsides include difficulties in the handling of large motions for small objects or generating artifacts when warping occluded regions. Another observation we made is that vanilla agglomeration of hierarchical information in the pyramid is hindering the learning process and consequently leading to reduced performance.

In this paper we identify and address shortcomings in state-of-the-art flow networks, with particular focus on improving information processing in the pyramidal representation module. For cost volume construction at a single pyramid level, we introduce a novel feature sampling strategy rather than relying on warping of high-level features to the corresponding ones in the target image. Warping is the predominant strategy in recent and top-performing flow methods [44, 14] but leads to degraded flow quality for fine structures. This is because fine structures require robust encoding of high-frequency information in the features, which is sometimes not recoverable after warping them towards the target image pyramid feature space. As an alternative we propose *sampling* for cost volume generation in each pyramid level, in conjunction with the sum of absolute differences as a cost volume distance function. In our sampling strategy we populate cost volume entries through distance computation between features *without* prior feature warping. This helps us to better explore the complex and non-local search space of fine-grained, detailed flow transformations (see Fig. 1).

Using *sampling* in combination with a per-pyramid level *loss max-pooling* strategy further supports recovery of the motion of small and fast-moving objects. Flow errors for those objects can be attributed to the aforementioned warping issue but also because the motion of such objects often correlates with large and underrepresented flow vectors, rarely available in the training data. Loss max-pooling adaptively shifts the focus of the learning procedure towards underperforming flow predictions, without requiring additional information about the training data statistics. We introduce a loss max-pooling variant to work in hierarchical feature representations, while the underlying concept has been successfully used for dense pixel prediction tasks like semantic segmentation [30].

Our second major contribution targets improving the gradient flow *across* pyramid levels. Functions like cost volume generation depend on bilinear interpolation, which can be shown [19] to produce considerably noisy gradients. Furthermore, fine-grained structures which are only visible at a certain pyramid level, can propagate contradicting gradients towards the coarser levels when they move in a different direction compared to their background. Accumulating these gradients across pyramid levels ultimately inhibits convergence. Our proposed solution is as simple as effective: by using level-specific loss terms and smartly blocking gradient propagation, we can eliminate the sources of noise. Doing so significantly improves the learning procedure and is positively reflected in the relevant performance measures.

As minor contributions, we promote additional *flow cues* that lead to a more effective generation of the cost volume. Inspired by the work of [15] that used backward warping of the optical flow to enhance the upsampling of occlusions, we advance symmetric flow networks with multiple cues (like consistencies derived from forward-backward and reverse flow information, occlusion reasoning) to better identify and correct discrepancies in the flow estimates. Finally, we also propose *knowledge distillation* to counterfeit the problem of catastrophic forgetting in the context of deep-learning-based optical flow algorithms. Due to a lack of large training datasets, it is common practice to sequentially perform a number of trainings, first on synthetically generated datasets (like Flying Chairs2 and Flying Things), then fine-tuning on target datasets like Sintel or KITTI. Our distillation strategy (inspired by recent work on scene flow [18] and unsupervised approaches [21, 20]) enables us to preserve knowledge from previous training steps and combine it with flow consistency checks generated from our network and further information about photometric consistency.

Our combined contributions lead to significant, cumulated error reductions over state-of-the-art networks like HD^3 or (variants of) PWC-Net [44, 37, 15, 2], and we set new state-of-the-art results on the challenging Sintel and KITTI 2012 datasets. We provide exhaustive ablations and experimental evaluations on Sintel, KITTI 2012 and 2015, Flying Things and Flying Chairs2, and significantly improve on the most important measures like *Out-Noc* (percentage of erroneous non-occluded pixels) and on *EPE* (average end-point-error) metrics.

2 Related Work

Classical approaches. Optical flow has come a long way since it was introduced to the computer vision community by Lucas and Kanade [23] and Horn and Schunck [13]. Following these works, the introduction of pyramidal coarse-to-fine warping frameworks were giving another huge boost in the performance of optical flow computation [4, 34] – an overview of non learning-based optical flow methods can be found in [1, 35, 9].

Deep Learning entering optical flow. Many parts of the classical optical flow computations are well-suited for being learned by a deep neural network. Initial work using deep learning for flow was presented in [40], and was using a learned matching algorithm to produce semi-dense matches then refining them with a classical variational approach. The successive work of [29], whilst also relying on learned semi-dense matches, was additionally using an edge detector [7] to interpolate dense flow fields before the variational energy minimization. End-to-end learning in a deep network for flow estimation was first done in FlowNet [8].

They use a conventional encoder-decoder architecture, and it was trained on a synthetic dataset, showing that it still generalizes well to real world datasets such as KITTI [11]. Based on this work, FlowNet2 [16] improved by using a carefully tuned training schedule and by introducing warping into the learning framework. However, FlowNet2 could not keep up with the results of traditional variational flow approaches on the leaderboards. SpyNet[27] introduced spatial image pyramids and PWC-Net [36, 37] additionally improved results by incorporating spatial feature pyramid processing, warping, and the use of a cost volume in the learning framework. The flow in PWC-Net is estimated by using a stack of flattened cost volumes and image features from a Dense-Net. In [15], PWC-Net was turned into an iterative refinement network, adding bilateral refinement of flow and occlusion in every iteration step. ScopeFlow [2] showed that improvements on top of [15] can be achieved simply by improving training procedures. In the work of [28], the group around [36] was showing further improvements on Kitti 2015 and Sintel by integrating the optical flow from an additional, previous image frame. While multi-frame optical flow methods already existed for non-learning based methods [6, 41, 10], they were the first to show this in a deep learning framework. In [44], the hierarchical discrete distribution decomposition framework HD^3 learned probabilistic pixel correspondences for optical flow and stereo matching. It learns the decomposed match densities in an end-to-end manner at multiple scales. HD^3 then converts the predicted match densities into point estimates, while also producing uncertainty measures at the same time. Devon [22] estimates the flow at fixed quarter resolution and uses a deformable, sampling based cost-volume to iteratively estimate the flow. While they showed reasonable results on synthetic data, their performance on real images from KITTI remained sub-optimal, indicating that sampling alone is not sufficient. Recently, Volumetric Correspondence Networks (VCN) [43] showed that the 4D cost volume can also be efficiently filtered directly without the commonly used flattening but using separable 2D filters instead. Generating dense and accurate flow data for supervised training of networks is a challenging task. Thus, most large-scale datasets are synthetic [5, 8, 17], and real data sets remained small and sparsely labeled [26, 25].

Unsupervised methods. Unsupervised methods do not rely on that data, instead, those methods usually utilize the photometric loss between the original image in the warped, second image to guide the learning process [45]. However, the photometric loss does not work for occluded image regions, and therefore methods have been proposed to generate occlusion masks beforehand or simultaneously [24, 42].

Distillation. To learn the flow values of occluded areas, DDFlow [20] is using a student-teacher network which distills data from reliable predictions, and uses these predictions as annotations to guide a student network. SelFlow [21] is built in a similar fashion but vastly improves the quality of the flow predictions in occluded areas by introducing a superpixel-based occlusion hallucination technique. They obtain state-of-the-art results when fine-tuning on annotated data after pre-training in a self-supervised setting. SENSE [18] tries to integrate optical flow, stereo, occlusion, and semantic segmentation in one semi-supervised setting. Much like in a multi-task learning setup, SENSE [18] uses a shared encoder for all four tasks, which can exploit interactions between the different tasks and leads to a compact network. SENSE uses pre-trained models to "supervise" the network on data with missing ground truth annotations using a distillation loss [12]. To couple the four tasks, a self-supervision loss term is used, which largely improves regions without ground truth (*e.g.* sky regions).

3 Main Contributions

In this section we review pyramid flow network architectures [36, 44], and propose a set of modifications to the pyramid levels (§ 3.2) and their training strategy (§ 3.3), which work in a synergistic manner to greatly boost performance.

3.1 Pyramid flow networks

Pyramid flow networks (PFN) operate on pairs of images, building feature pyramids with decreasing spatial resolution using "siamese" network branches with shared parameters. Flow is iteratively refined starting from the top of the pyramid, each layer predicting an offset relative to the flow estimated at the previous level. For more details about the operations carried out at each level see § 3.2.

Notation. We represent multi-dimensional feature maps as functions $I_i^l : \mathcal{I}_i^l \to \mathbb{R}^d$, where $i = 1, 2$ indicates which image the features are computed from, l is their pyramid level, and $\mathcal{I}_i^l \subset \mathbb{R}^2$ is the set of pixels of image i at resolution l. We call *forward flow* at level l a mapping $F_{1\to2}^l : \mathcal{I}_1^l \to \mathbb{R}^2$, which intuitively indicates where pixels in I_1^l moved to in I_2^l (in relative terms). We call *backward flow* the mapping $F_{2\to1}^l : \mathcal{I}_2^l \to \mathbb{R}^2$ that indicates the opposite displacements. Pixel coordinates are indexed by u and v, *i.e.* $x = (x_u, x_v)$, and given $x \in \mathcal{I}_1^l$, we assume that $I_1^l(x)$ implicitly applies bilinear interpolation to read values from I_1^l at sub-pixel locations.

3.2 Improving pyramid levels in PFNs

Many PFNs [36, 44] share the same high-level structure in each of their levels. First, feature maps from the two images are aligned using the coarse flow estimated in the previous level, and compared by some distance function to build a cost volume (possibly both in the *forward* and *backward* directions). Then, the cost volume is combined with additional information from the feature maps (and optionally additional "flow cues") and fed to a "decoder" subnet. This subnet finally outputs a residual flow, or a match density from which the residual flow can be computed. A separate loss is applied to each pyramid layer, providing deep supervision to the flow refinement process. In the rest of this section, we describe a set of generic improvements that can be applied to the pyramid layers of several state of the art pyramid flow networks.

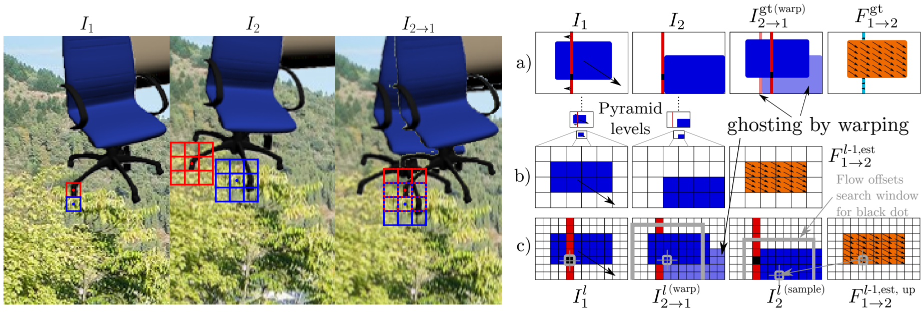
I_1
I_2
$I_{2\to1}$
I_1
I_2
$I_{2\to1}^{\mathrm{gt\,(warp)}}$
$F_{1\to2}^{\mathrm{gt}}$
a)
Pyramid levels
ghosting by warping
b)
$F_{1\to2}^{l\text{-}1,\mathrm{est}}$
Flow offsets search window for black dot
c)
I_1^l
$I_{2\to1}^{l\,\mathrm{(warp)}}$
$I_2^{l\,\mathrm{(sample)}}$
$F_{1\to2}^{l\text{-}1,\mathrm{est,\ up}}$

Fig. 2: Sampling vs. Warping. **Left**: Warping leads to image ghosting in the warped image $I_{2\to1}$; Also, neighbouring pixels in I_1 must share parts of their search windows in $I_{2\to1}$, while for sampling they are independently sampled from the original image I_2. **Right**: A toy example; a) Two moving objects: a red line with a black dot and a blue box. Warping with $F^{\mathrm{gt}}_{1\to2}$ leads to ghosting effects. b) Zooming into lowest pyramid resolution shows loss of small details due to down-scaling. c) *Warping* I_2^l with the flow estimate from the coarser level leads to distortions in $I^l_{2\to1}$ (the black dot gets covered up). Instead, direct *sampling* in I_2^l with a search window(gray box) that is offset by the flow estimate avoids these distortions and hence leads to more stable correlations.

Cost volume construction The first operation at each level of most pyramid flow networks involves comparing features between I_1^l and I_2^l, conditioned on the flow $F^{l-1}_{1\to2}$ predicted at the previous level. In the most common implementation, I_2^l is warped using $F^{l-1}_{1\to2}$, and the result is cross-correlated with I_1^l. More formally, given I_2^l and $F^{l-1}_{1\to2}$, the warped image is given by $I^l_{2\to1}(x) = I^l_2(x + F^{l-1}_{1\to2}(x))$ and the cross-correlation is computed with:

$$V^{\mathrm{warp}}_{1\to2}(x,\delta) = I_1^l(x)\cdot I^l_{2\to1}(x+\delta) = I_1^l(x)\cdot I_2^l(x+\delta+F^{l-1}_{1\to2}(x+\delta))\,, \quad (1)$$

where $\delta \in [-\Delta, \Delta]^2$ is a restricted search space and $\cdot$ is the vector dot product. This warping operation, however, suffers from a serious drawback which occurs when small regions move differently compared to their surroundings.

This case is represented in Fig. 2: A small object indicated by a red line moves in a different direction than a larger blue box in the background. As warping uses the coarse flow estimate from the previous level, which cannot capture fine-grained motions, there is a chance that the smaller object gets lost during the feature warping. This makes it undetectable in $I^l_{2\to1}$, even with an infinite cost volume range (CVr/CV-range) δ. To overcome this limitation, we propose a different cost volume construction strategy, which exploits direct sampling operations. This approach always accesses the original, undeformed features I_2^l, without loss of information, and the cross-correlation in Eq. (1) now becomes:

$$V^{\mathrm{samp,Corr}}_{1\to2}(x,\delta) = I_1^l(x)\cdot I_2^l(x+\delta+F^{l-1}_{1\to2}(x))\,. \quad (2)$$

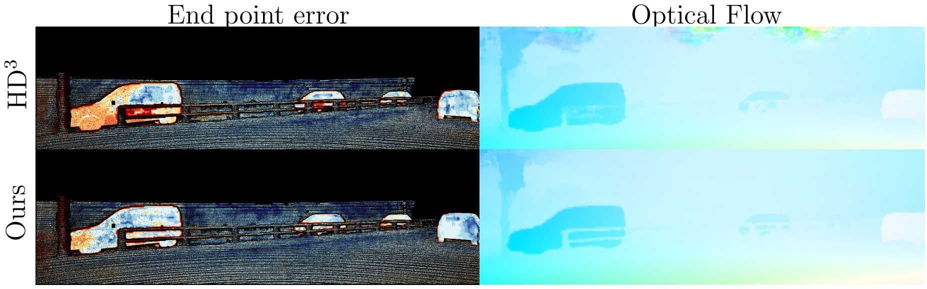

Fig. 3: Predicted optical flow and end point error on KITTI obtained with HD^3 from the model zoo (top) and our version (bottom). Note how our model is better able to preserve small details.

For this operator, the flow just acts as an offset that sets the center of the correlation window in the feature image I_2^l. Going back to Fig. 2, one can see that the sampling operator is still able to detect the small object, as it is also exemplified on real data in Fig. 3. In our experiments we also consider a variant where the features are compared in terms of Sum of Absolute Differences (SAD) instead of dot product:

$$V_{1\to2}^{\mathrm{samp,SAD}}(x,\delta) = \|I_1^l(x) - I_2^l(x+\delta+F_{1\to2}^{l-1}(x))\|_1 \,. \tag{3}$$

Loss Max Pooling We apply a Loss Max-Pooling (LMP) strategy [30], also known as Online Hard Example Mining (OHEM), to our knowledge for the first time in the context of optical flow. In our experiments, and consistent with the findings in [30], we observe that LMP can help to better preserve small details in the flow. The total loss is the sum of a pixelwise loss ℓ_x over all $x \in \mathcal{I}_1$, but we optimize a weighted version thereof that selects a fixed percentage of the highest per-pixel losses. The percentage value α is best chosen according to the quality of the ground-truth in the target dataset. This can be written in terms of a loss max-pooling strategy as follows:

$$L = \max\left\{\sum_{x\in\mathcal{I}_1} w_x \ell_x \,:\, \|w\|_1 \le 1\,, \|w\|_\infty \le \frac{1}{\alpha|\mathcal{I}_1|}\right\}\,, \tag{4}$$

which is equivalent to putting constant weight $w_x = \frac{1}{\alpha|\mathcal{I}_1|}$ on the percentage of pixels x exhibiting the highest losses, and setting $w_x = 0$ elsewhere.

LMP lets the network focus on the more difficult areas of the image, while reducing the amount of gradient signals where predictions are already correct. To avoid focussing on outliers, we set the loss to 0 for pixels that are out of reach for the current relative search range $\varDelta$. For datasets with sparsely annotated ground-truth, like *e.g.* KITTI [11], we re-scale the per pixel losses ℓ_x to reflect the number of valid pixels. Note that, when performing distillation, loss max-pooling is only applied to the supervised loss, in order to further reduce the effect of noise that survived the filtering process described in § 3.4.

3.3 Improving gradient flow across PFN levels

Our quantitatively most impacting contribution relates to the way we pass gradient information across the different levels of a PFN. In particular, we focus on the bilinear interpolation operations that we implicitly perform on I_2^l while computing Eq.s (1), (2) and (3). It has been observed [19] that taking the gradient of bilinear interpolation w.r.t. the sampling coordinates (*i.e.* the flow $F_{1\to 2}^{l-1}$ from the previous level in our case) is often problematic. To illustrate the reason, we restrict our attention to the 1-D case for ease of notation, and write linear interpolation from a function $\hat{f} : \mathbb{Z} \to \mathbb{R}$:

$$f(x) = \sum_{\eta \in \{0,1\}} \hat{f}(\lfloor x \rfloor + \eta)\left[(1-\eta)(1-\tilde{x}) + \eta\tilde{x}\right] , \tag{5}$$

where $\tilde{x} = x - \lfloor x \rfloor$ denotes the fractional part of x. The derivative of the interpolated function $f(x)$ with respect to x is:

$$\frac{df}{dx}(x) = \sum_{\eta \in \{0,1\}} \hat{f}(\lfloor x \rfloor + \eta)(2\eta - 1) . \tag{6}$$

The gradient function $\frac{df}{dx}$ is discontinuous, for its value drastically changes as $\lfloor x \rfloor$ crosses over from one integer value to the next, possibly inducing strong noise in the gradients. An additional effect, specific to our case, is related to the issues already highlighted in § 3.2: since $F_{1\to 2}^{l-1}$ is predicted at a lower resolution than level l operates at, it cannot fully capture the motion of smaller objects. When this motion contrasts with that of the background, the gradient w.r.t. $F_{1\to 2}^{l-1}$ produced from the sampling at level l will inevitably disagree with that produced by the loss at level $l-1$, possibly slowing down convergence.

While [19] proposes a different sampling strategy to reduce the noise issues discussed above, in our case we opt for a much simpler work around. Given the observations about layer disagreement, and the fact that the loss at $l-1$ already provides direct supervision on $F_{1\to 2}^{l-1}$, we choose to stop back-propagation of flow gradients across levels altogether, as illustrated in Fig. 4.

Evidence for this effect can be seen in Fig. 5, where the top shows the development of the training loss for a Flying Chairs 2 training with an HD^3 model. The training convergence clearly improves when the partial flow gradient is stopped between the levels (red cross in Fig. 4). On the bottom of the figure the Normalized Cross Correlation (NCC) between the partial gradient coming from the next level via the flow and the current levels loss is shown. On average the correlation is negative, indicating that for each level of the network the gradient component that we decided to stop, coming from upper levels, is pointing in a direction that opposes the gradient from the loss directly supervising the current level, thus harming convergence. Additional evidence of the practical, positive impact of our gradient stopping strategy is given in the experiment section § 4.2.

Further evidence on this issue can be gained by analyzing the parameter gradient variance [38] as it impacts the rate of convergence for stochastic gradient descent methods. Also the β-smoothness [33] of the loss function gradient can give similar insights. In the supplementary material (section § A) we provide further experiments that show that gradient stopping also helps to improve these properties, and works for stereo estimation and other flow models as well.

3.4 Additional refinements

Flow cues As mentioned at the beginning of § 3.2, the decoder subnet in each pyramid level processes the raw feature correlations to a final cost volume or direct flow predictions. To provide the decoder with contextual information, it commonly [36, 44] also receives raw features (*i.e.* I_1^l, I_2^l for forward and backward flow, respectively). Some works [39, 15, 17] also append other cues, in the form of hand-crafted features, aimed at capturing additional prior knowledge about flow consistency. Such flow cues are cheap to compute but otherwise hard to learn for CNNs as they require various forms of non-local spatial transformations. In this work, we propose a set of such flow cues that provides mutual beneficial information, and perform very well in practice when combined with costvolume sample and LMP (see § 4.2). These cues are namely forward-backward flow warping, reverse flow estimation, map uniqueness density and out-of-image occlusions, and are described in detail in the supplementary material (§ B).

Knowledge distillation Knowledge distillation [12] consists in extrapolating a training signal directly from another trained network, ensemble of networks, or perturbed networks [31], typically by mimicking their predictions on some available data. In PFNs, distillation can help to overcome issues such as lack of flow annotations on *e.g.* sky, which results in cluttered outputs in those areas. Formally, our goal is to distill knowledge from a pre-trained master network (*e.g.* on Flying Chairs2 and/or Flying Things) by augmenting a student network with

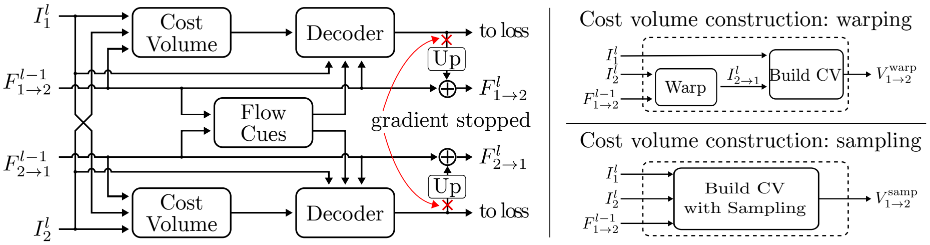

Fig. 4: Left: Network structure – flow estimation per pyramid level; Gradients are stopped at red cross; Right: Cost volume computation with sampling vs. warping.

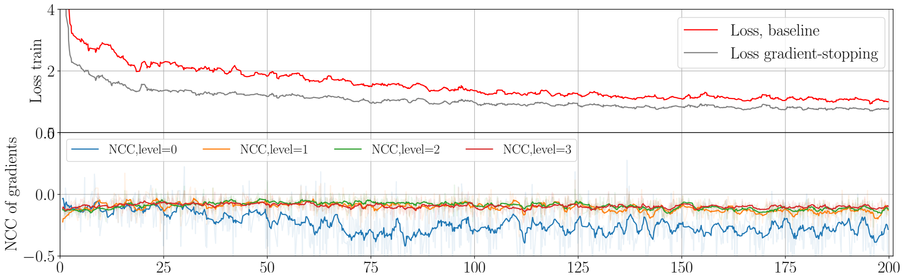

Fig. 5: Top: Loss of model decreases when the flow gradient is stopped; Bottom: Partial gradients coming from the current level loss and the next level via the flow show a negative Normalized Cross Correlation (NCC), indicating that they oppose each other.

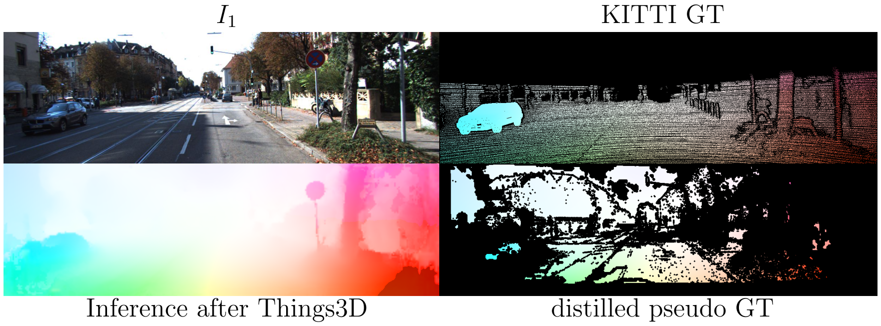

Fig. 6: Illustration of our data distillation process. Left to right: input image and associated KITTI ground truth, dense prediction from a Flying Things3D-trained network and pseudo-ground truth derived from it.

an additional loss term, which tries to mimic the predictions the master produces on the input at hand (Fig. 6, bottom left). At the same time, the student is also trained with a standard, supervised loss on the available ground-truth (Fig. 6, top right). In order to ensure a proper cooperation between the two terms, we prevent the distillation loss from operating blindly, instead enabling it selectively based on a number of consistency and confidence checks (refer to the supplementary material for details). Like for the ground-truth loss, the data distillation loss is scaled with respect to the valid pixels present in the pseudo ground-truth. The supervised and the distillation losses are combined into a total loss

$$\mathcal{L} = \alpha \mathcal{L}_S + (1 - \alpha)\mathcal{L}_D \tag{7}$$

with the scaling factor $\alpha = 0.9$. A qualitative representation of the effects of our proposed distillation on KITTI data is given in Fig. 7.

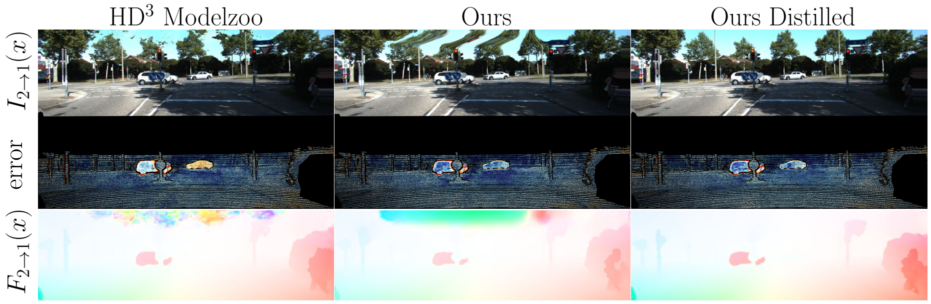

Fig. 7: Qualitative evaluation on KITTI, comparing the HD^3 modelzoo (left), our version with all contributions except distillation (center), and with distillation (right).

4 Experiments

We assess the quality of our contributions by providing a number of exhaustive ablations on Flying Chairs, Flying Chairs2, Flying Things, Sintel, Kitti 2012 and Kitti 2015. We ran the bulk of ablations based on HD^3 [44], *i.e.* a state-of-the-art, 2-frame optical flow approach. We build on top of their publicly available code and stick to default configuration parameters where possible, and describe and re-train the baseline model when deviating.

The remainder of this section is organized as follows. We provide i) in § 4.1 a summary about the experimental and training setups and our basic modifications over HD^3, ii) in § 4.2 an exhaustive number of ablation results for all aforementioned datasets by learning **only** on the Flying Chairs2 training set, and for all reasonable combinations of our contributions described in § 3, as well as ablations on Sintel, and iii) list and discuss in § 4.3 our results obtained on the Kitti 2012, Kitti 2015 and Sintel test datasets, respectively. In the supplementary material we further provide i) more technical details and ablation studies about the used *flow cues*, ii) smoothness and variance analyses for gradient stopping and its impact on depth from stereo or with a PWC baseline iii) ablations on extended search ranges for the cost volume, and iv) ablations on distillation.

4.1 Setup and modifications over HD^3

We always train on 4xV100 GPUs with 32GB RAM using PyTorch, and obtain additional memory during training by switching to In-Place Activated BatchNorm (non-synchronized, Leaky-ReLU) [32]. We decided to train on Flying Chairs2 rather than Flying Chairs for our main ablation experiments, since it provides ground truth for both, forward and backward flow directions. Other modifications are experiment-specific and described in the respective sections.

Flow - Synthetic data pre-training. Also the Flying Things dataset provides ground truth flow for both directions. We always train and evaluate on both flow directions, since this improves generalization to other datasets. We use a batch size of 64 to decrease training times and leave the rest of configuration parameters unchanged w.r.t. the default HD^3 code.

Flow - Fine-tuning on KITTI. Since both the Kitti 2012 and the Kitti 2015 datasets are very small and only provide forward flow ground truth, we follow the HD^3 training protocol and join all KITTI training sequences for the final fine-tuning (after pre-training on Flying Chairs2 and Flying Things). However, we ran independent multi-fold cross validations and noticed faster convergence of our model over the baseline. We therefore perform early stopping after 1.6k (CVr±4)/ 1.4k (CVr±8) epochs, to prevent over-fitting. Furthermore, before starting the fine-tuning process of the pre-trained model, we label the KITTI training data for usage described in the knowledge distillation paragraph in § 3.4.

Flow - Fine-tuning on Sintel. We only train on all the images in the *final* pass and ignore the *clean* images like HD^3 for comparability. Also, we only use the forward flow ground truth since backward flow ground truth is unavailable. Although not favorable, our model can still be trained in this setting since we use a single, shared set of parameters for the forward and the backward flow paths. We kept the original 1.2k finetuning iterations for comparability, since our independent three-fold cross validation did not show signs of overfitting.

4.2 Flow ablation experiments

Here we present an extensive number of ablations based on HD^3 to assess the quality of all our proposed contributions. We want to stress that all results in Tab. 1 were obtained by **solely training on the Flying Chairs2 training set**. More specifically, we report error numbers (EPE and Fl-all; lower is better) and compare the original HD^3 model zoo baseline against our own, retrained baseline model, followed by adding combinations of our proposed contributions. We report performance on the target domains validation set (Flying Chairs2), as well as on unseen data from different datasets (Flying Things, Sintel and KITTI), to gain insights on generalization behavior.

Our ablations show a clear trend towards improving EPE and Fl-all, especially on the target domain, as more of our proposed improvements are integrated. Due to the plethora of results provided in the table, we highlight some of them next. GRADIENT STOPPING is often responsible for a large gap w.r.t. to both baseline HD^3 models, the original and our re-trained. Further, all variants with activated SAMPLING lead to best- or second-best results, except for Fl-all on Sintel. Flow Cues give an additional benefit when combined with SAMPLING but not with warping. Another relevant insight is that our full model using all contributions at the bottom of the table always improves on Fl-all compared to the variant with deactivated LMP. This shows how LMP is suitable to effectively reduce the number of outliers by focusing the learning process on the under-performing (and thus more rare) cases.

Table 1: Ablation results when training HD^3 CVr±4 on Flying Chairs2 in comparison to the official model zoo baseline, our re-trained baseline and when adding all our proposed contributions. Results are shown on validation data for Flying Chairs2 and Flying Things (validation set used in the original HD^3 code repository), and on the official training data for Sintel, Kitti 2012 and Kitti 2015, due to the lack of a designated validation split. (Highlighting **best** and second-best results).

Gradient	Sampling	Flow	SAD	LMP	Flying Chairs2		Flying Things		Sintel final		Sintel clean		Kitti 2012		Kitti 2015	
Stopping		Cues			EPE [1]	Fl-all [%]	EPE [1]	Fl-all [%]	EPE [1]	Fl-all [%]	EPE [1]	Fl-all [%]	EPE [1]	Fl-all [%]	EPE [1]	Fl-all [%]
HD^3 baseline model zoo					1.439	7.17	20.973	33.21	5.850	**14.03**	3.70	**8.56**	12.604	49.13	22.67	57.07
HD^3 baseline – re-trained					1.422	6.99	17.743	26.72	6.273	15.24	3.90	10.04	8.725	34.67	20.98	50.27
✓	✗	✗	✗	✗	1.215	6.23	19.094	26.84	5.774	15.89	3.72	10.51	9.469	44.58	19.07	53.65
✓	✗	✓	✗	✗	1.216	6.24	16.294	26.25	6.033	16.26	3.43	9.98	7.879	43.92	17.97	51.14
✓	✓	✗	✗	✗	1.208	6.19	17.161	24.75	6.074	15.61	3.70	9.96	8.673	45.29	17.42	51.23
✓	✓	✓	✗	✗	1.186	6.16	19.616	28.51	7.420	15.99	3.61	9.39	**6.672**	**32.59**	**16.23**	47.56
✓	✓	✓	✓	✗	1.184	6.15	15.136	25.00	5.625	16.35	3.38	9.97	8.144	41.59	17.13	52.51
✓	✗	✗	✗	✓	1.193	6.02	44.068	40.38	12.529	17.85	5.48	10.95	8.778	42.37	19.08	51.13
✓	✓	✓	✗	✓	1.170	5.98	15.752	24.26	5.943	16.27	3.55	9.91	7.742	35.78	18.75	**49.67**
✓	✓	✓	✓	✓	**1.168**	**5.97**	**14.458**	**23.01**	**5.560**	15.88	**3.26**	9.58	6.847	35.47	16.87	49.93

Table 2: Ablation results on Sintel, highlighting **best** and second-best results. Top: Baseline and Flying Chairs2 & Flying Things pre-trained (P) models only. Bottom: Results after additional fine-tuning (F) on Sintel.

Fine-tuned	Gradient	Sampling	Flow	SAD	LMP	CV	Flying Things		Sintel final		Sintel clean	
Pretrained	Stopping		Cues			range ±8	EPE [1]	Fl-all	EPE [1]	Fl-all	EPE [1]	Fl-all
HD^3 baseline – re-trained							12.52	18.06%	13.38	16.23 %	3.06	6.39%
P	✓						7.98	13.41%	**4.06**	**10.62** %	1.86	5.11%
P	✓	✓	✓	✓	✓		7.06	12.29%	4.23	11.05 %	2.20	5.41%
P	✓	✓	✓	✓	✓	✓	**5.77**	**11.48%**	4.68	11.40 %	**1.77**	**4.88%**
F							19.89	27.03%	(1.07)	(4.61 %)	1.58	4.67%
F	✓						13.80	20.87%	(0.84)	(3.79 %)	1.43	4.19%
F	✓	✓	✓	✓	✓		14.19	20.98%	(0.82)	(3.63 %)	1.43	4.08%
F	✓	✓	✓	✓	✓	✓	**11.80**	**19.12%**	**(0.79)**	**(3.49 %)**	**1.19**	**3.86%**

We provide additional ablation results on Flying Things and Sintel in Tab. 2. The upper half shows PreTrained (P) results obtained after training on Flying Chairs2 and Flying Things, while the bottom shows results after additionally fine-tuning (F) on Sintel. Again, there are consistent large improvements on the target domain currently trained on, i.e. (P) for Flying Things and (F) for Sintel. On the cross dataset validation there is more noise, especially for sintel final that comes with motion blur etc., but still always a large improvement over the baseline. After finetuning (F) the full model with CVr±8 shows much better performance on sintel and at the same time comparable performance on Flying Things to the original baseline model directly trained on Flying Things.

4.3 Optical flow benchmark results

The following provides results on the official Sintel and KITTI test set servers.

Sintel. By combining all our contributions and by using a cost volume search range of ±8, we set a new state-of-the-art on the challenging Sintel Final test

Table 3: EPE scores on the Sintel test datasets. The appendix -*ft* denotes fine-tuning on Sintel.

	Training		Test	
Method	Clean	Final	Clean	Final
FlowNet2 [16]	2.02	3.14	3.96	6.02
FlowNet2-ft [16]	(1.45)	(2.01)	4.16	5.74
PWC-Net [36]	2.55	3.93	-	-
PWC-Net-ft [36]	(2.02)	(2.08)	4.39	5.04
SelFlow [21]	2.88	3.87	6.56	6.57
SelFlow-ft [21]	(1.68)	(1.77)	3.74	4.26
IRR-PWC-ft [15]	(1.92)	(2.51)	3.84	4.58
PWC-MFF-ft [28]	-	-	3.42	4.56
VCN-ft [43]	(1.66)	(2.24)	**2.81**	4.40
ScopeFlow [2]	-	-	3.59	4.10
Devon [22]	-	-	4.34	6.35
HD^3 [44]	3.84	8.77	-	-
HD^3-ft [44]	(1.70)	(1.17)	4.79	4.67
Ours-no-ft	2.20	4.32	-	-
Ours-ft	1.43	(0.82)	4.39	4.22
OursCVr8-no-ft	1.77	4.68	-	-
OursCVr8-ft	**1.19**	**(0.79)**	3.58	**4.01**

Table 4: EPE and Fl-all scores on the KITTI test datasets. The appendix -*ft* denotes fine-tuning on KITTI.

	Kitti 2012			Kitti 2015		
Method	EPE	EPE	Fl-noc [%]	EPE	Fl-all [%]	Fl-all [%]
	train	test	test	train	train	test
FlowNet2 [16]	4.09	-	-	10.06	30.37	-
FlowNet2-ft [16]	(1.28)	1.8	4.82	(2.30)	8.61	10.41
PWC-Net [36]	4.14	-	-	10.35	33.67	-
PWC-Net-ft [36]	(1.45)	1.7	4.22	(2.16)	9.80	9.60
SelFlow [21]	1.16	2.2	7.68	(4.48)	-	14.19
SelFlow-ft [21]	(0.76)	1.5	6.19	(1.18)	-	8.42
IRR-PWC-ft [15]	-	-	-	(1.63)	5.32	7.65
PWC-MFF-ft [28]	-	-	-	-	-	7.17
ScopeFlow [2]	-	1.3	2.68	-	-	6.82
Devon [22]	-	-	6.99	-		9.16
VCN [43]	-	-	-	(1.16)	4.10	**6.30**
HD3F [44]	4.65	-	-	13.17	23.99	
HD3F-ft [44]	(0.81)	1.4	**2.26**	1.31	4.10	6.55
Ours-no-ft	2.52	-	-	8.32	20.33	-
Ours-ft	**(0.73)**	**1.2**	2.29	1.17	3.40	6.52
OursCVr8-no-ft	2.37	-	-	7.09	18.93	-
OursCVr8-ft	(0.76)	**1.2**	**2.25**	**1.14**	**3.28**	6.35

set, improving over the very recent, best-working approach in [2] (see Tab. 3). Even by choosing the default search range of CVr±4 as in [44] we still obtain significant improvements over the HD^3-ft baseline on training and test errors.

Kitti 2012 and Kitti 2015. We also evaluated the impact of our full model on KITTI and report test data results in Tab. 4. We obtain new state-of-the-art test results for EPE and Fl-all on Kitti 2012, and rank second-best at Fl-all on Kitti 2015. On both, Kitti 2012 and Kitti 2015 we obtain strong improvements on the training set on EPE and Fl-all. Finally, while on Kitti 2015 the recently published VCN [43] has slightly better Fl-all scores, we perform better on foreground objects (test Fl-fg 8.09 % vs. 8.66 %) and generally improve over the HD^3 baseline (Fl-fg 9.02 %). It is worth noting that all KITTI finetuning results are obtained after integrating knowledge distillation from § 3.4, leading to significantly improved flow predictions on areas where KITTI lacks training data (*e.g.* in far away areas including sky, see Fig. 7). We provide further qualitative insights and direct comparisons in the supplementary material (§ C).

5 Conclusions

In this paper we have reviewed the concept of spatial feature pyramids in context of modern, deep learning based optical flow algorithms. We presented complementary improvements for cost volume construction at a single pyramid level, that i) departed from a warping- to a sampling-based strategy to overcome issues like handling large motions for small objects, and ii) adaptively shifted the focus of the optimization towards under-performing predictions by means of a loss max-pooling strategy. We further analyzed the gradient flow across pyramid levels and found that properly eliminating noisy or potentially contradicting ones improved convergence and led to better performance. We applied our proposed modifications in combination with additional, interpretable flow cue extensions as well as distillation strategies to preserve knowledge from (synthetic) pre-training stages throughout multiple rounds of fine-tuning. We experimentally analyzed and ablated all our proposed contributions on a wide range of standard benchmark datasets, and obtained new state-of-the-art results on Sintel and Kitti 2012.

Acknowledgements: T. Pock and M. Hofinger acknowledge that this work was supported by the ERC starting grant HOMOVIS (No. 640156).

References

1. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. International Journal of Computer Vision **92**(1), 1–31 (Mar 2011). https://doi.org/10.1007/s11263-010-0390-2, https://doi.org/10.1007/s11263-010-0390-2
2. Bar-Haim, A., Wolf, L.: Scopeflow: Dynamic scene scoping for optical flow. In: CVPR (June 2020)
3. yves Bouguet, J.: Pyramidal implementation of the lucas kanade feature tracker. Intel Corporation, Microprocessor Research Labs (2000)
4. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: ECCV (2004)
5. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: A. Fitzgibbon et al. (Eds.) (ed.) European Conf. on Computer Vision (ECCV). pp. 611–625. Part IV, LNCS 7577, Springer-Verlag (Oct 2012)
6. Chaudhury, K., Mehrotra, R.: A trajectory-based computational model for optical flow estimation. IEEE Transactions on Robotics and Automation **11**(5), 733–741 (Oct 1995). https://doi.org/10.1109/70.466611
7. Dollár, P., Zitnick, C.L.: Structured forests for fast edge detection. In: (ICCV) (2013)
8. 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 International Conference on Computer Vision (ICCV) (2015), http://lmb.informatik.uni-freiburg.de/Publications/2015/DFIB15
9. Fortun, D., Bouthemy, P., Kervrann, C.: Optical flow modeling and computation. Comput. Vis. Image Underst. **134**(C), 1–21 (May 2015). https://doi.org/10.1016/j.cviu.2015.02.008, http://dx.doi.org/10.1016/j.cviu.2015.02.008
10. Garg, R., Roussos, A., Agapito, L.: A variational approach to video registration with subspace constraints. International Journal of Computer Vision **104**(3), 286–314 (Sep 2013). https://doi.org/10.1007/s11263-012-0607-7, https://doi.org/10.1007/s11263-012-0607-7
11. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The KITTI dataset. (IJRR) (2013)
12. Hinton, G.E., Vinyals, S., Dean, J.: Distilling the knowledge in a neural network. In: Deep Learning Workshop, NIPS (2014)

13. Horn, B.K.P., Schunck, B.G.: Determining optical flow. ARTIFICAL INTELLIGENCE **17**, 185–203 (1981)
14. Hui, T.W., Tang, X., Loy, C.C.: A Lightweight Optical Flow CNN - Revisiting Data Fidelity and Regularization. arXiv preprint arXiv:1903.07414 (2019), http://mmlab.ie.cuhk.edu.hk/projects/LiteFlowNet/
15. Hur, J., Roth, S.: Iterative residual refinement for joint optical flow and occlusion estimation. In: CVPR (2019)
16. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: Evolution of optical flow estimation with deep networks. In: CVPR (2017), http://lmb.informatik.uni-freiburg.de/Publications/2017/IMSKDB17
17. Ilg, E., Saikia, T., Keuper, M., Brox, T.: Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In: ECCV (2018)
18. Jiang, H., Sun, D., Jampani, V., Lv, Z., Learned-Miller, E., Kautz, J.: Sense: A shared encoder network for scene-flow estimation. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
19. Jiang, W., Sun, W., Tagliasacchi, A., Trulls, E., Yi, K.M.: Linearized multi-sampling for differentiable image transformation. In: ICCV. pp. 2988–2997 (2019)
20. Liu, P., King, I., Lyu, M.R., Xu, S.J.: Ddflow: Learning optical flow with unlabeled data distillation. CoRR **abs/1902.09145** (2019)
21. Liu, P., Lyu, M.R., King, I., Xu, J.: Selflow: Self-supervised learning of optical flow. In: CVPR (2019)
22. Lu, Y., Valmadre, J., Wang, H., Kannala, J., Harandi, M., Torr, P.: Devon: Deformable volume network for learning optical flow. In: The IEEE Winter Conference on Applications of Computer Vision (WACV) (March 2020)
23. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of Imaging Understanding Workshop. pp. 4884–4893 (1981), http://cseweb.ucsd.edu/classes/sp02/cse252/lucaskanade81.pdf
24. Mac Aodha, O., Humayun, A., Pollefeys, M., Brostow, G.J.: Learning a confidence measure for optical flow. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(5), 1107–1120 (May 2013). https://doi.org/10.1109/TPAMI.2012.171
25. Menze, M., Heipke, C., Geiger, A.: Joint 3d estimation of vehicles and scene flow. In: ISPRS Workshop on Image Sequence Analysis (ISA) (2015)
26. Menze, M., Heipke, C., Geiger, A.: Object scene flow. ISPRS Journal of Photogrammetry and Remote Sensing (JPRS) (2018)
27. Ranjan, A., Black, M.J.: Optical flow estimation using a spatial pyramid network. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2720–2729 (July 2017). https://doi.org/10.1109/CVPR.2017.291
28. Ren, Z., Gallo, O., Sun, D., Yang, M.H., Sudderth, E.B., Kautz, J.: A fusion approach for multi-frame optical flow estimation. In: IEEE Winter Conference on Applications of Computer Vision (2019)
29. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: Epicflow: Edge-preserving interpolation of correspondences for optical flow. CoRR (2015)
30. Rota Bulò, S., Neuhold, G., Kontschieder, P.: Loss max-pooling for semantic image segmentation. In: (CVPR) (July 2017)
31. Rota Bulò, S., Porzi, L., Kontschieder, P.: Dropout distillation. In: Proceedings of The 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 48, pp. 99–107. PMLR, New York, New York, USA (20–22 Jun 2016)
32. Rota Bulò, S., Porzi, L., Kontschieder, P.: In-place activated batchnorm for memory-optimized training of DNNs. In: (CVPR) (2018)

33. Santurkar, S., Tsipras, D., Ilyas, A., Mądry, A.: How does batch normalization help optimization? In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. p. 2488–2498. NIPS'18, Curran Associates Inc., Red Hook, NY, USA (2018)
34. Sun, D., Roth, S., Black, M.J.: Secrets of optical flow estimation and their principles. In: CVPR. pp. 2432–2439 (2010)
35. Sun, D., Roth, S., Black, M.J.: A quantitative analysis of current practices in optical flow estimation and the principles behind them. International Journal of Computer Vision **106**(2), 115–137 (Jan 2014). https://doi.org/10.1007/s11263-013-0644-x, https://doi.org/10.1007/s11263-013-0644-x
36. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In: CVPR (2018)
37. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Models matter, so does training: An empirical study of cnns for optical flow estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **42**(6), 1408–1423 (2020). https://doi.org/10.1109/TPAMI.2019.2894353, to appear
38. Wang, C., Chen, X., Smola, A.J., Xing, E.P.: Variance reduction for stochastic gradient optimization. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26, pp. 181–189. Curran Associates, Inc. (2013), http://papers.nips.cc/paper/5034-variance-reduction-for-stochastic-gradient-optimization.pdf
39. Wang, Y., Yang, Y., Yang, Z., Zhao, L., Wang, P., Xu, W.: Occlusion aware unsupervised learning of optical flow. In: CVPR. pp. 4884–4893 (06 2018). https://doi.org/10.1109/CVPR.2018.00513
40. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: IEEE Intenational Conference on Computer Vision (ICCV). Sydney, Australia (Dec 2013), http://hal.inria.fr/hal-00873592
41. Werlberger, M., Trobin, W., Pock, T., Wedel, A., Cremers, D., Bischof, H.: Anisotropic huber-l1 optical flow. In: Proceedings of the British Machine Vision Conference (BMVC). London, UK (September 2009), to appear
42. Yamaguchi, K., McAllester, D., Urtasun, R.: Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In: ECCV (2014)
43. Yang, G., Ramanan, D.: Volumetric correspondence networks for optical flow. In: Advances in Neural Information Processing Systems 32, pp. 793–803. Curran Associates, Inc. (2019), http://papers.nips.cc/paper/8367-volumetric-correspondence-networks-for-optical-flow.pdf
44. Yin, Z., Darrell, T., Yu, F.: Hierarchical discrete distribution decomposition for match density estimation. In: CVPR (2019)
45. Yu, J.J., Harley, A.W., Derpanis, K.G.: Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In: Computer Vision - ECCV 2016 Workshops, Part 3 (2016)