The Five Elements of Flow

Markus Hofinger[†,‡], Samuel Rota Bulò[†], Lorenzo Porzi[†], Arno Knapitsch[†], Thomas Pock[‡], Peter Kontschieder[†]
Mapillary Research[†], Graz University of Technology[‡]
research@mapillary.com[†], {markus.hofinger,pock}@icg.tugraz.at[‡]

Figure 1: Optical flow predicted on the Sintel dataset using HD^3 augmented with our five elements of flow.

Abstract

In this work we propose five concrete steps to improve the performance of optical flow algorithms. We carefully reviewed recently introduced innovations and well-established techniques in deep-learning-based flow methods including i) pyramidal feature representations, ii) flow-based consistency checks, iii) cost volume construction practices or iv) distillation, and present extensions or alternatives to inhibiting factors we identified therein. We also show how changing the way gradients propagate in modern flow networks can lead to surprising boosts in performance. Finally, we contribute a novel feature that adaptively guides the learning process towards improving on under-performing flow predictions. 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 2015 test datasets, and even show the portability of our findings to different optical flow and depth from stereo approaches.

1. Introduction

In this paper we reveal, discuss and overcome a number of shortcomings we have identified in state-of-the-art, deep-learning-based optical flow and stereo matching approaches. By meticulous inspection of best practices, we identified *five* specific actions, leading to significant, cumulated error reductions over state-of-the-art [42]. Across multiple benchmark datasets including Sintel, KITTI 2012 and 2015, Flying Things and Flying Chairs2, we significantly improve on the most important measures like *Out-Noc* (percentage of erroneous non-occluded pixels) and on *EPE* (average end-point-error) metrics. Our findings improve recent and popular optical flow and stereo matching works like HD^3 and PWC-Net [33, 42], and we obtain new state-of-the-art results on Sintel and KITTI 2015. Even more appealing, when applying all our five elements of flow, these improved results can be obtained without adding significant overhead in terms of network parameters.

Our proposed steps towards obtaining *better flow* are particularly effective for improving issues related to the cost volume (and match density) estimation when performed in a coarse-to-fine manner. Feature pyramids are well-known and established in classical, *ante deep learning* optical flow and stereo matching works like [45], and so recently also found their way into deep-learning-based approaches [12, 33, 42]. However, while pyramidal representations allow for computationally tractable exploration of the pixel flow search space, we learned that vanilla agglomeration of hierarchical information is actually hindering performance. Our first and most effective element of improvement is hence simply coined **GRADIENT STOPPING**. We prevent the flow of gradients across pyramid layers, and instead promote explicit supervision at layer-specific loss terms. The intuition behind it is that a hierarchical approach enables propagation of errors between layers, which can be inhibiting for the learning process in subsequent layers. With gradient stopping we found a simple and yet powerful way to overcome this issue, gracefully reflected in the performance development for EPE and Fl-all measures.

We also identify two key elements that lead to a more effective generation of the cost volume. Inspired by the work of [13] that used backward warping of the optical flow to enhance the upsampling of occlusions, we advance symmetric flow networks with multiple cues to better identify and correct discrepancies in the flow estimates. We propose to exploit this information as additional **FLOW CUES**, inherently making the model aware about contradicting predictions and learning to overcome them in the first place.

These cues comprise of consistencies derived from forward-backward and reverse flow information, as well as occlusion reasoning (map uniqueness density [36] and out of image) terms that we make directly available to the network. Another, major improvement we propose replaces warping of learned high-level features at each pyramid level to the corresponding features of the target image, which is a strategy adopted by recent and top-performing flow methods [12, 42]. We found this to be impairing the flow quality for fine structures, which requires robust encoding of high-frequency information in the features, yet recoverable after transformation in the target image pyramid feature space. As alternative we propose **SAMPLING** for cost volume generation, in conjunction with 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.

The fourth element we propose targets **KNOWLEDGE DISTILLATION**, introducing a strategy 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 on first synthetically generated datasets (like Flying Chairs2 and Flying Things), before fine-tuning on target datasets like Sintel or KITTI. Our distillation strategy (inspired by recent work on scene flow [17] and unsupervised approaches [18, 19]) 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. We show that inclusion of this knowledge during training improves results both, qualitatively and quantitatively. Finally, our fifth element targets improvement of the Fl-all measure, which essentially reports the percentage of pixels with prediction errors beyond a given threshold (typically 2 or 3 pixels). Typically, these errors strongly correlate with underrepresented, large flow vectors and/or in conjunction with fine-grained structures. Our proposed solution is coined **LOSS MAX-POOLING** and adaptively shifts the focus of the learning procedure towards under-performing flow predictions, without requiring additional information about the training data statistics. A similar concept has been shown to improve the performance on semantic segmentation by compensating the effect of largely imbalanced training datasets [27]. Our presented, modified variant is adjusted to work in presence of hierarchical feature representations, yielding considerable reductions of outliers.

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 [20] and Horn and Schunck [11]. Main contributions were pyramidal coarse-to-fine warping frameworks, which greatly contributed to the success of optical flow computation [2, 31] – an overview of non learning-based optical flow methods can be found in [1, 7, 32]. Many parts of the classical optical flow computations are well-suited for being learned by a deep neural network.

Deep Learning entering optical flow. Initial work using deep learning for flow was presented in [38], and was using a learned matching algorithm to produce semi-dense matches and further refines them with a classical variational approach. The successive work of [26], whilst also relying on learned semi-dense matches, was additionally using an edge detector [5] 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 [6]. 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 [9]. Based on this work, FlowNet2 [14] improved by using a carefully tuned training schedule and by introducing warping into the learning framework. However, FlowNet2 could now keep up with the results of traditional variational flow approaches on the leaderboards. PWC-Net [34, 35] additionally improved results by incorporating pyramidal processing, warping, and introduction of a cost volume into the learning framework. The flow in PWC-Net is estimated using a stack of flattened cost volumes and image features from a Dense-Net. In [13], PWC-Net was turned into an iterative refinement network, adding bilateral refinement of flow and occlusion in every iteration step. In the work of [25], the group around [35] 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 [4, 8, 39], they were the first to show this in a deep learning framework. In [42], 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. Generating dense and accurate flow data for supervised training of networks is a challenging task. For that reason, most large-scale datasets are synthetic [3, 6, 15], and real data sets remained small and sparsely labeled [22, 23]. Very recently VCN [41] showed that the 4D cost volume can also be efficiently filtered directly without the commonly used flattening but using separable 2D filters instead.

Unsupervised methods. Unsupervised methods do not rely on that data, instead, those methods usually utilize the pho-

tometric loss between the original image in the warped, second image to guide the learning process [44]. However, the photometric loss does not work for occluded image regions, and therefore methods have been proposed to generate occlusion masks beforehand or simultaneously [21, 40].

Distillation. To learn the flow values of occluded areas, DDFlow [18] is using a student-teacher network which **distills data** from reliable predictions, and uses these predictions as annotations to guide a student network. SelFlow [19] 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 [17] tries to integrate optical flow, stereo, occlusion, and semantic segmentation in one semi-supervised setting. Much like in a multi-task learning setup, SENSE [17] 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 [10]. 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 report a number of findings that allow to improve the performance of optical flow networks. Before delving into details, we introduce some notations and definitions that we will use in this section.

Notation. The flow network in this work operates on pairs of images, which are denoted by $I_1 : \mathcal{I}_1 \rightarrow \mathbb{R}^d$ and $I_2 : \mathcal{I}_2 \rightarrow \mathbb{R}^d$. Their sets of pixels $\mathcal{I}_1 \subset \mathbb{R}^2$ and $\mathcal{I}_2 \subset \mathbb{R}^2$ are kept separate for readability to indicate which image we operate on. We call *forward flow* a mapping $F_{1\rightarrow 2} : \mathcal{I}_1 \rightarrow \mathbb{R}^2$, which intuitively indicates where pixels in image I_1 moved to in image I_2 (in relative terms). We call *backward flow* the mapping $F_{2\rightarrow 1} : \mathcal{I}_2 \rightarrow \mathbb{R}^2$ that indicates the opposite displacements. Given a pixel $x \in \mathcal{I}_1$ we denote by $x_{1\rightarrow 2} \in \mathbb{R}^2$ the matching position of x in image I_2 (in absolute terms), *i.e.* $x_{1\rightarrow 2} = x + F_{1\rightarrow 2}(x)$. Similarly, for the opposite direction, we define $y_{2\rightarrow 1} \in \mathbb{R}^2$ for pixels $y \in \mathcal{I}_2$. Pixel coordinates are indexed by u and v, *i.e.* $x = (x_u, x_v)$, and given $x \in \mathcal{I}_1$, we assume that $I_1(x)$ implicitly applies bilinear interpolation to read values from image I_1 at subpixel locations. We denote by $|a|$ the absolute value of a real scalar a, by $[a]_+$ the maximum between a and 0, and by $\mathbb{1}_P \in \{0, 1\}$ an indicator function about the truth of proposition P.

3.1. Gradient Stopping

Our quantitatively most impacting contribution relates to the way we pass gradient information across the different levels of a pyramidal flow network. In particular, this can be applied to architectures that give explicit supervision to the flow (or flow residuals) at each level of the hierarchy, and where the flow prediction at each level is conditioned on the flow prediction from the previous level. This is *e.g.* the case of PWC-Net and HD^3-Net.

We focus here on a minimal two-levels example, which can nevertheless be generalized to multiple levels. Let $f_1(\theta)$ be the flow predicted by the network with parameters θ at the coarse level, and let $f_2(f_1(\theta); \theta)$ be the flow predicted at the finer level by the same network, with the dependency on the flow from the previous level made explicit. Let the loss be $L = L_1(f_1(\theta))) + L_2(f_2(f_1(\theta); \theta))$, where L_1 and L_2 are the level-specific losses that match the predicted flow against some ground-truth. The gradient of L with respect to θ, *i.e.* $\frac{dL}{d\theta}$, can be written as the sum of three terms, namely $\frac{dL_1}{df_1}\frac{df_1}{d\theta}$, $\frac{dL_2}{df_1}\frac{df_1}{d\theta}$ and $\frac{dL_2}{df_2}\frac{df_2}{d\theta}$. The gradient stopping operation cancels the second of those terms, namely the one originated from L_2 via the relation between f_2 and f_1 (see Fig. 3). We found this truncation to be highly beneficial in practice, although we have by now no theoretical explanation of why this is the case, besides showing experimentally consistent improvements over different datasets and different network architectures, namely PWC-Net and HD^3-Net. Our intuitive explanation is that due to the direct supervision of the flows in each level of the hierarchy, the gradient path traversing the hierarchy through the flow predictions is superfluous, and possibly harmful because it might try to compensate for errors that have been committed at the previous levels, increasing the variance of the gradient estimate and thus affecting the convergence of the training algorithm.

To our knowledge, a similar technique has been applied in the context of two-stage, 2d object detection (*e.g.* Faster R-CNN [24]), where the network at the second stage corrects proposals predicted at the previous stage. Indeed, it has been experimentally proven to be beneficial to stop the gradient from flowing from the second stage to the first stage through the predicted position of the proposal bounding box, for the same reasons mentioned above.

3.2. Incorporate Multiple Flow Cues

The use of prior knowledge when computing optical flow has been widely explored in classical methods. More recently, approaches such as PWC-Net [33] have started incorporating these techniques (*e.g.* feature pyramids and warping) into deep-learning frameworks. Following this path, we propose to utilize (i) forward-backward flow warping, (ii) reverse flow estimation, (iii) map uniqueness density, and (iv) out-of-image occlusions as additional cues for our network. While some recent works [37][13][15] have used some of these in a deep learning context, to the best of our knowledge we are the first to combine multiple cues and make them explicitly available to the network as features, to

refine the flow predictions.

First, our architecture keeps jointly track of the forward and backward flows by exploiting Siamese modules with shared parameters, with features from I_1 and I_2 being fed to the two branches in mirrored order. A downside is its increased memory consumption, which we noticeably mitigate by adopting In-Place Activated BatchNorm [29] throughout our networks. Without additional connections, the Siamese modules compute the forward $F_{1\to2}$ and backward $F_{2\to1}$ flow mappings in a completely independent way. However, in practice the true flows are strongly tied to each other, although they reside on different coordinate systems. We therefore provide the network with a Flow Cue Module that gives each branch different kind of cues about its own and the other branch's flow estimates. Each of these cues represents a different mechanism to bring mutually supplementary information from one coordinate system to the other. For the sake of simplicity, we will always present the results of the cues in the coordinate system of the branch that operates on the features of I_1.

Forward-backward flow warping. Since both flow mappings are available, they can be used to bring one flow in the coordinate frame of the other via dense warping. For example, a forward flow estimate $F^{\text{fb}}_{1\to2}$ can be made from the backward flow $F_{2\to1}$ by warping it with the forward flow $F_{1\to2}$:

$$F^{\text{fb}}_{1\to2}(x) = -F_{2\to1}(x_{1\to2})$$

The other direction $F^{\text{fb}}_{2\to1}$ can be computed in a similar way. Comparing the estimated results $F^{\text{fb}}_{1\to2}$ versus $F_{1\to2}$ can be used for consistency checks, and is used in unsupervised flow methods [19, 44] to estimate occlusions in a heuristic manner.

Reverse flow estimation [16]. In contrast to the previous Cue, reverse flow estimation can be used to estimate the forward flow $F_{1\to2}$ directly from backward flow $F_{2\to1}$ alone, although in a non-dense manner. The reverse flow estimates are denoted by $F^{\text{rev}}_{2\to1}$ and $F^{\text{rev}}_{1\to2}$ and are obtained by

$$F^{\text{rev}}_{1\to2}(x) = -\frac{\sum_{y\in\mathcal{I}_2}\omega(x, y_{2\to1})F_{2\to1}(y)}{\omega_1(x)}, \qquad (1)$$

where $\omega(x,x') = [1-|x_u - x'_u|]_+ [1-|x_v - x'_v|]_+$ denotes the bilinear interpolation weight of x' relative to x, and

$$\omega_1(x) = \sum_{y\in\mathcal{I}_2}\omega(x, y_{2\to1})$$

is a normalizing factor. In the dis-occluded areas where the denominator of Eq. (1) is 0, we define the flow values $F^{\text{rev}}_{1\to2}(x) = 0$. In occluded areas $F^{\text{rev}}_{1\to2}$ will become an average of the incoming flows. Similarly, we define $F^{\text{rev}}_{2\to1}$ by swapping 1 and 2 as well as x and y in Eq. (1).

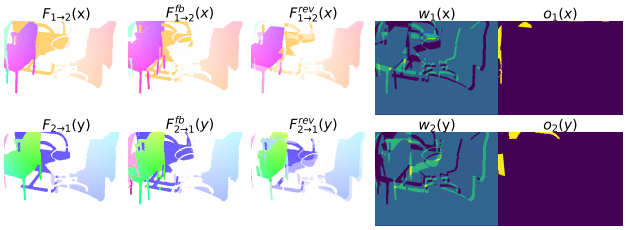

Figure 2: Flow Cues module output illustration for a given ground truth input; Left to right: Ground truth flow, forward-backward estimate, reverse flow estimate, map uniqueness density, out of image occlusions.

Map uniqueness density [36, 37]. Provides information about occlusions and dis-occlusions and basically corresponds to ω_1 in Eq. (1) for image I_1. The value of $\omega_1(x)$ provides the (soft) amount of pixels in I_2 with flow vectors pointing towards $x \in \mathcal{I}_1$. Occluded areas will result in values ≥ 1 whereas areas becoming dis-occluded in values ≤ 1. ω_1 is therefore an indicator on where the reverse flow is more or less precise. Similarly, we have ω_2(x) for I_2.

Out-of-image occlusions. This represents an indicator function, *e.g.* $o_1 : \mathcal{I}_1 \to \{0,1\}$ for image I_1, providing information about flow vectors pointing out of the other image's domain, *i.e.*

$$o_1(x) = \mathbb{1}_{x_{1\to2}\notin\mathcal{I}_2}$$

and similarly we define $o_2 : \mathcal{I}_2 \to \{0,1\}$ for image I_2.

The Flow Cue Module. We show in Fig. 2 how the flow cues mutually benefit from one another in different areas. *E.g.*, the-out-of-image occlusions o_1 allow to differentiate which dis-occlusions in map uniqueness density ω_1 are real dis-occlusions, *i.e.* areas where the object moved away, and where the low density stems from flow vectors in the second image that are just likely not visible in the current crop.

We therefore provide the network with all the additional flow cues mentioned above, namely $F^{\text{rev}}_{1\to2}$, $F^{\text{fb}}_{1\to2}$, ω_1 and o_1, by stacking them as additional features together with the original forward flow $F_{1\to2}$ for the subsequent part of the network. Therefore, the network now has three differently generated flow estimates including its own prediction $F_{1\to2}$. The following layers can therefore reason about consistency and probable sources of outliers with a far better basis than one single cue alone could provide. Symmetrically, the same is done for the backward stream (see Fig. 2).

3.3. Sampling for Cost Volume Construction

A common point of many pyramidal flow networks, including *e.g.* PWC-Net [35], is the usage of a multi resolution cost volume in combination with a *warping* operator. Starting from the coarsest resolution, the cost volumes are built by computing the correlation between the features of I_1 and those of I_2 warped by the flow estimate from the previous pyramid level. This process iteratively aligns the two

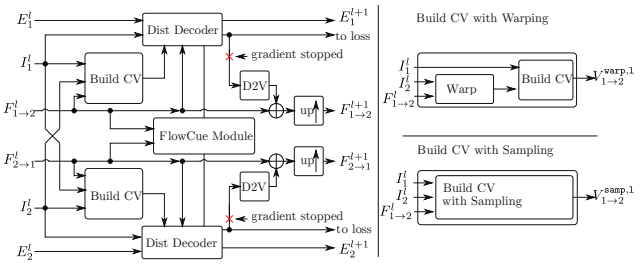

Figure 3: Left: Network structure – symmetric flow estimation per pyramid level; Red cross indicates position of gradient stop point; Right: Cost volume computation with sampling vs. warping

images, and each successive level only has to search for the residual transformation, which can be done using reasonably small search spaces.

More formally, given image I_2 (or a possibly down-scaled d-dimensional feature representation thereof) and a warping flow $F_{1\to2}$, the warped image is given by $I_{2\to1}(x) = I_2(x_{1\to2})$ and the correlation volume via warping is computed as

$$\begin{aligned} V^{\text{warp}}_{1\to2}(x,\delta) &= I_1(x) \cdot I_{1\to2}(x+\delta) \\ &= I_1(x) \cdot I_2(x+\delta+F_{1\to2}(x+\delta))\,, \end{aligned}$$

where $\delta \in [-\Delta,\Delta]^2$ is a restricted search space (in our experiments $\Delta = 4$) and $\cdot$ is the vector dot product.

Our finding is that this warping operation introduces a serious downside: the warped space can transform drastically, and some parts might become unreachable. This generally happens when small regions move different than their surroundings. In this case, since the fine-grained motion cannot be properly represented in the coarser scale, the warping operation has a chance of covering the small object with features from its background, even when flow is predicted perfectly. This case is represented in Figure 4: here, a small object indicated by a red line moves in a different direction than a larger blue box in the background. Since the flow vector must decide on a value for each single pixel the small object is not represented in the flow at the low resolutions. This can lead to a covering of the small objects features in the next level due to the warping. Even with an infinite search space, the cost volume would never be able to recover the correct flow. Since this process repeats from the coarse to the fine level such details can be lost completely.

In order to overcome this limitation, we propose a different cost volume construction strategy, which exploits direct sampling operations. This approach always accesses the original, undeformed feature image I_2, without any loss of information. The computation of the cost volume with our strategy now becomes:

$$V^{\text{samp,Corr}}_{1\to2}(x,\delta) = I_1(x) \cdot I_2(x+\delta+F_{1\to2}(x))\,.$$

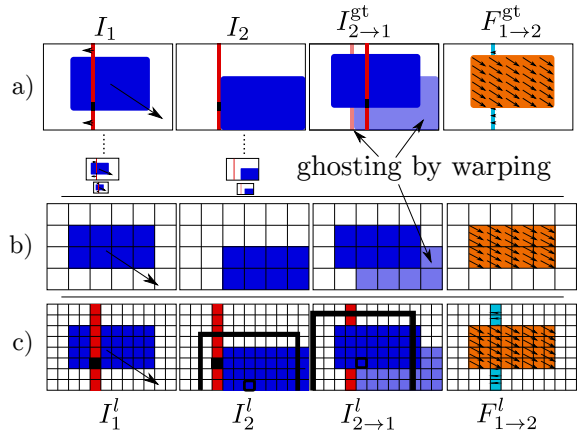

Figure 4: Sampling vs. Warping. a) Two moving objects: a red line with a black dot and a blue box. Warping with $F^{\text{gt}}_{1\to2}$ leads to ghosting effects. b) A zoom into the lowest resolution of pyramid exemplifies how small details can be lost to down-scaling. c) Using the flow from the coarser level on the next finer level results in an erroneous warped image (mind the disappearance of the black dot). The sampling approach does not suffer from this problem and hence leads to more stable correlations within the black window.

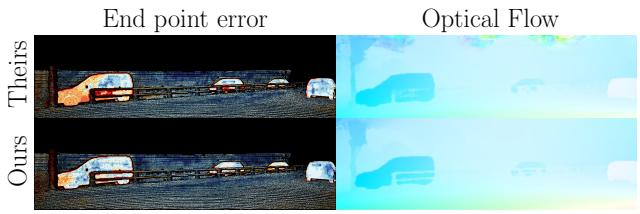

Figure 5: Predicted optical flow and end point error on KITTI obtained with HD^3 from the model zoo (top) and our version (bottom). Note how our model is better able to preserve small details.

For this operator, the flow just acts as an offset that sets the center of the correlation window in the feature image I_2. Going back to Fig. 4, one can see that the sampling operator still is able to detect the small object, as it is also exemplified on real data in Fig. 5.

In networks like PWC-Net or HD^3-Net the cost volume is computed as a cross-correlation between feature maps. It is well-known in optimization that a sum of squares – and for normalized inputs correlation is up to a constant equal to that – is less robust to outliers than a Sum of Absolute Differences (SAD). We will therefore also consider an alternative construction that uses Sum of Absolute Differences (SAD) as a distance measure for building the cost volume:

$$V^{\text{samp,SAD}}_{1\to2}(x,\delta) = \|I_1(x) - I_2(x+\delta+F_{1\to2}(x))\|_1\,.$$

3.4. Knowledge Distillation

Knowledge distillation [10] consists in extrapolating a training signal directly from another trained network, ensemble of networks, or perturbed networks [30], typically by mimicking their predictions on some available data. In our application domain, distillation can help overcome issues such as lack of flow annotations on *e.g.* sky, which results in cluttered outputs in those areas (this seems to be a particularly severe problem for the HD^3 type base models).

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 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. Specifically, we apply the following filters, obtaining "pseudo ground-truth" annotations (Fig. 6, bottom right):

- We use forward $F_{1\to2}$ and backward $F_{2\to1}$ flows to estimate occlusions. Specifically, we regard a pixel $y \in I_1$ as not occluded if the following holds [19]
$$\|F_{1\to2}(y) + F_{2\to1}(y_{1\to2})\|^2 - 0.05 < 0.01\left(\|F_{1\to2}(y)\|^2 + \|F_{2\to1}(y_{1\to2})\|^2\right) .$$
- We compute the photometric error using SAD on a per pixel basis and determine a mask of good predictions by thresholding the error.
- We determine the confidence of the network using the method proposed in [42] and retain predictions with a confidence above 95%.
- Finally, we combine all of the previous filters and apply an additional pruning using an erosion operation to remove small patches, in order to only keep regions with sufficient trustable data.

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 to a total loss

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

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

3.5. Loss Max Pooling

We apply a Loss Max-Pooling (LMP) strategy [28], also known as Online Hard Example Mining (OHEM), to optical flow, to our knowledge for the first time. LMP can be quite

Figure 6: Illustration of our data distillation process. Top to bottom, 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.

Figure 7: Qualitative evaluation on KITTI, comparing the HD^3 model from the model zoo (left), our version with all factors enabled except distillation (center), and the one with distillation (right).

sensitive to noise, making it, at least in principle, a bad fit for optical flow. However, in our experiments we found that it can help to better preserve small details, especially when paired with losses that are robust to outliers. This is the case *e.g.* in HD^3-Net, where the loss ℓ_x is a Kullback-Leibler divergence between the predicted and ground-truth vector-to-density representations. When the current gt residual falls out of the search space $d \times d$ of a level, the loss of that pixel is 0, which makes it robust to harsh outliers. The total loss is the sum of 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 gt in the targeted dataset. This can be written in terms of a loss max-pooling strategy as follows:

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

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.

This lets the network focus on the more difficult task still to learn, while reducing the amount of gradient signals where it is already good at. For datasets with sparsely annotated ground-truth, like *e.g.* KITTI [9], 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 prevent the network from learning from potential outliers that survived the

filtering process described in Sec. 3.4.

4. Experiments

We assess the quality of our proposed contributions by providing a number of exhaustive ablations on Flying Chairs, Flying Chairs2, Flying Things, Sintel, Kitti 2012 and Kitti 2015. Our flow elements are not limited to a specific flow model but apply to pyramid-based approaches in general. We ran the bulk of ablations based on HD^3 [43], *i.e.* the current state-of-the-art, 2-frame optical flow approach. We build on top of their code and stick to default configuration parameters where possible, describe and re-train the baseline model where we deviate.

The remainder of this section is organized as follows. We i) summarize the experimental and training setups and basic modifications of the network in § 4.1, ii) provide in § 4.2 an exhaustive list of ablation results for all aforementioned datasets by learning **only** on the Flying Chairs2 training set, and for all reasonable combinations of our proposed five elements of flow described in § 3, and iii) list and discuss in § 4.3 our results obtained on the Kitti 2012, Kitti 2015 and Sintel test datasets, respectively. More analyses about the effect of our proposed FLOW CUES, and about applying gradient stopping to other optical flow works and depth from stereo, are provided in the supplementary material.

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) [29]. 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 are 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 due to decreased 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 1600 epochs, to prevent over-fitting. Furthermore, before starting the fine-tuning process of the pre-trained model, we use it to label the entire KITTI training data for our distillation element.

Flow - Fine-tuning on Sintel. We only train on all the images in the *final* pass and ignore the *clean* images just like HD^3. 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. Due to missing validation data, we again ran cross-validation using a three-fold split over the training data.

4.2. Flow Ablation Experiments

Here we present an extensive number of ablations based on HD^3 and 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 elements. We not only report validation errors on Flying Chairs2, but also EPE and Fl-all numbers for validation data from Flying Things and Sintel. We also list results on training data for KITTI, in order to gain insights regarding generalization behavior of our elements.

Our ablations show a clear trend towards improving EPE and Fl-all as more of our proposed elements get added. 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. However, we also demonstrate in § 4.3, that with activating all our five elements of flow we manage to obtain new state-of-the-art results on the official Sintel *final* test set. Another relevant insight is that our full combination 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 rarer) cases. Another interesting insight we have obtained is that we encounter only minor gains when just replacing traditional warping with our proposed SAMPLING element. However, in combination with our FLOW CUES we obtain considerable improvements, which are less pronounced when combined with warping.

4.3. Optical Flow Benchmark Results

Here we provide the results obtained on the official Sintel and KITTI test sets and their servers, respectively.

Sintel. While the ablation results presented in § 4.2 show sub-optimal performance after pre-training on Flying

Gradient Stopping	Sampling	Flow Cues	SAD	LMP	Flying Chairs2		Flying Things		Sintel		Kitti 2012		Kitti 2015	
					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	20.91	5.850	**14.03**	12.604	49.13	5.850	49.13
HD^3 baseline – re-trained					1.422	6.99	17.743	17.36	6.273	15.24	8.725	34.67	6.273	34.67
✓	✗	✗	✗	✗	1.215	6.23	19.094	16.85	5.774	15.89	9.469	44.58	5.774	44.58
✓	✓	✗	✗	✗	1.208	6.19	17.161	15.74	6.074	15.61	8.673	45.29	6.074	45.29
✓	✗	✓	✗	✗	1.216	6.24	16.294	16.52	6.033	16.26	7.879	43.92	6.033	43.92
✓	✓	✓	✗	✗	1.186	6.16	19.616	17.67	7.420	15.99	**6.672**	**32.59**	7.420	**32.59**
✓	✓	✓	✓	✗	1.184	6.15	15.136	15.89	5.625	16.35	8.144	41.59	5.625	41.59
✓	✗	✗	✗	✓	1.193	6.02	44.068	23.43	12.529	17.85	8.778	42.37	12.529	42.37
✓	✓	✓	✗	✓	1.170	5.98	15.752	15.51	5.943	16.27	7.742	35.78	5.943	35.78
✓	✓	✓	✓	✓	**1.168**	**5.97**	**14.458**	**14.72**	**5.560**	15.88	6.847	35.47	**5.560**	35.47

Table 1: Ablation results when training HD^3 on Flying Chairs2 in comparison to the official model zoo baseline, our re-trained baseline and when adding all our proposed elements of flow. 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).

Chairs2, the story significantly changes when fine-tuning on Sintel data (see Tab. 2). Combining all our elements we set a new state-of-the-art on the challenging Sintel Final test set, improving by 6.5% over the previously best-working approach [25]. We also 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 all our five flow elements on KITTI and report test data results in Tab. 3. We provide state-of-the-art results for EPE errors on both, Kitti 2012 and Kitti 2015. On Kitti 2012 test our Fl-noc scores drop minimally, while EPE shows a strong improvement for test and train. Although, on Kitti 2015 the very recently published VCN[41] yields a slightly better EPE and Fl-all, we perform better on foreground objects (test Fl-fg 8.06 % vs. 8.66 %) and generally improve over the HD^3 baseline (Fl-fg 9.02 %). It is worth noting that all results are obtained when integrating our data distillation element, leading to significantly improved flow predictions on areas where KITTI lacks training data (far away areas including sky, see Fig. 7). We provide qualitative insights and direct comparisons in the supplementary material.

5. Conclusions

In this paper we have reviewed essential building blocks of modern, deep learning based optical flow algorithms. We have raised *five* inhibiting shortcomings and provide simple yet effective *elements of flow* to overcome them. Our most impactful finding improves on issues from agglomerating gradient information over multiple pyramid levels during backpropagation. Instead, we stop gradient flow across layers while keeping per-layer supervision upright. Our other four findings improve i) the consistency of predictions by presenting multiple, learnable flow cues as features to the network, ii) the cost volume generation by replacing conventionally used warping with a novel sampling strategy operating on the original, non-deformed image features, iii) the preservation of knowledge from (synthetic) pre-training stages through distillation throughout multiple fine-tuning stages, and iv) the flow prediction on difficult samples by introducing a loss max-pooling strategy that shifts the focus of the learning process towards them in a dynamic way. We experimentally analyze and ablate all proposed flow elements on a range of benchmark datasets, and obtain new state-of-the-art results on Sintel, Kitti 2012 and Kitti 2015.

	Training		Test	
Method	Clean	Final	Clean	Final
FlowNet2 [14]	2.02	3.14	3.96	6.02
FlowNet2-ft [14]	(1.45)	(2.01)	4.16	5.74
PWC-Net [35]	2.55	3.93	-	-
PWC-Net-ft [35]	(2.02)	(2.08)	4.39	5.04
SelFlow [19]	2.88	3.87	6.56	6.57
SelFlow-ft [19]	(1.68)	(1.77)	3.74	4.26
IRR-PWC-ft [13]	(1.92)	(2.51)	3.84	4.58
PWC-MFF-ft [25]	-	-	3.42	4.56
VCN-ft [41]	(1.66)	(2.24)	**2.81**	4.40
HD^3 [43]	3.84	8.77	-	-
HD^3-ft [43]	(1.70)	(1.17)	4.79	4.67
Ours-ft	**1.43**	**(0.82)**	4.39	**4.22**

Table 2: EPE results on Sintel dataset. The appendix *-ft* denotes fine-tuned on Sintel and numbers in the parenthesis are results on data the method has been trained on.

	Kitti 2012			Kitti 2015		
Method	EPE train	EPE test	Fl-noc [%] test	EPE train	Fl-all [%] train	Fl-all [%] test
FlowNet2 [14]	4.09	-	-	10.06	30.37	-
FlowNet2-ft [14]	1.28	1.8	4.82	2.30	8.61	10.41
PWC-Net [35]	4.14	-	-	10.35	33.67	-
PWC-Net-ft [35]	1.45	1.7	4.22	2.16	9.80	9.60
SelFlow [19]	1.16	2.2	7.68	4.48	-	14.19
SelFlow-ft [19]	0.76	1.5	6.19	1.18	-	8.42
IRR-PWC-ft [13]	-	-	-	1.63	5.32	7.65
PWC-MFF-ft [25]	-	-	-	-	-	7.17
VCN [41]	-	-	-	**1.16**	4.10	**6.30**
HD3F [43]	4.65	-	-	13.17	23.99	
HD3F-ft [43]	0.81	1.4	**2.26**	1.31	4.10	6.55
Ours-ft	**0.73**	**1.2**	2.29	1.17	**3.40**	6.52

Table 3: EPE and Fl-all scores on the KITTI test datasets. The appendix *-ft* denotes fine-tuning on KITTI.

The Five Elements of Flow – Supplementary Material

A. Supplementary Material

This document contains supplementary material for the paper 'The Five Elements of Flow'. The structure of this supplementary document is the following:

- Porting gradient stopping
- Qualitative comparisons of validation epe changes
- Histograms of errors
- Ablations on flow cues
- Sidenote on D2V and V2D operation and warping
- Qualitative training results for KITTI (images)
- Qualitative training results for MPI-Sintel (images)

A.1. Porting Gradient Stopping

As indicated in the main part, here we show the applicability of our most effective element, gradient stopping to other works. PWC-Net [33] was chosen as an additional optical flow network, and HD^3 for stereo since it is among the top performing state-of-the-art stereo methods. The following results indeed verify the portability of our gradient stopping element.

Improving HD^3 – Depth from Stereo. In this experiment we use the Stereo training setup of HD^3 in their original publicly available codebase[1] and run a training on the Flying Things Stereo dataset. We choose to use the original version of the code base just with gradient stopping added, and keep the original training procedure that trains only on the *left* disparity. We do this to show that the effect of gradient stopping is not just limited to the simultaneous forward and backward training used in the main paper but is a more general one.

Again, we find significant improvements with our proposed gradient stopping element, as can be seen in Fig. 8. Using our gradient stopping element leads to an improvement of ≈10% on the final EPE. This confirms that gradient stopping also works for stereo estimation networks. Furthermore, it verifies that gradient stopping does not require joint forward- and backward flow training as used in the flow ablations in the main paper, but also leads to significant gains for a standard forward-only training.

Improving PWC-Net – Optical Flow. We use the PWC-Net implementation from the official IRR-PWC [13] publicly available code base[2], and run a training on the Flying Chairs dataset using their provided data processing and augmentation strategy, and follow all default settings for training. We run two experiments, the baseline and an experiment where we apply gradient stopping at the upsampling layer within the pyramid structure used therein. In direct comparison we found both, significantly improved reduction of the training loss for the final high-resolution level as well as the validation EPE (Fl-all is not reported from their inference code).

Fig. 9 shows the validation EPE of an exemplary experimental result on the PWC flow Network. As can be seen, applying gradient stopping leads to a faster convergence of the EPE. This immediately leads to initial gains of more than 10% at 20 epochs and 6% at 100 epochs. Therefore, lower EPE values can be reached faster. We kept the original learning rate schedule for comparability, but even in this setting that was optimized for the original baseline, a difference of approximately 2% remains after 200epoch. Gradient stopping shows a clear positive impact, even though the used PWC-variant directly regresses the flow at each level, whereas the HD^3 baseline that was used for many comparisons in the main paper uses residual estimates together with the D2V and V2D operations. This shows that stopping the gradients for the flow at the upsampling layer leads to

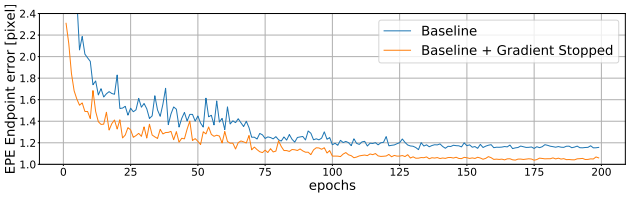

Figure 8: Improving HD^3 Stereo estimation with gradient stopping. Curves show validation Endpoint error (EPE) after each training epoch. Simple gradient stopping leads to faster convergence of the EPE

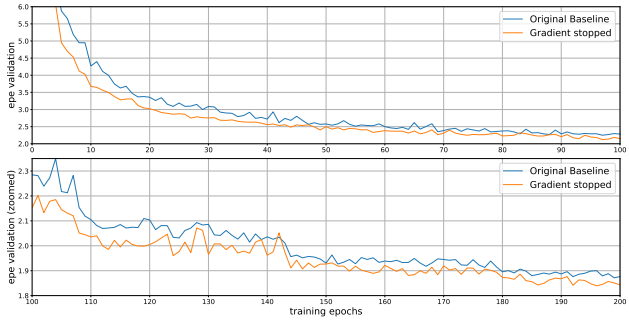

Figure 9: Improving PWC-Net with gradient stopping. Training with gradient stopping vs. original. Gradient stopping leads to faster decrease for the validation EPE.

[1] HD^3 codebase : https://github.com/ucbdrive/hd3/

[2] IRR-PWC codebase: https://github.com/visinf/irr

a faster decrease of the EPE also across multiple types of optical flow networks.

A.2. Qualitative Comparison of Training Convergence

Fig. 10 shows exemplary validation curves from the optical flow Flying Things 3D pre-training on an HD^3 baseline, which is the last part of the pre-training stage before fine-tuning on KITTI or Sintel. We evaluate on forward- and backward-flow for the same validation split provided by the HD^3 codebase.

The validation curves in Fig. 10 illustrate the overall behavior that we observed on the different datasets and models, when adding different elements. When adding gradient stopping to the baseline there is a significant drop in both EPE and Fl-all. Adding *Loss Max Pooling* (LMP) on top mostly affects the Fl-all by focusing on the remaining difficult examples. Adding the rest of the elements (Data Distillation is only applied on KITTI) leads to an additional boost in performance on both EPE and Fl-all.

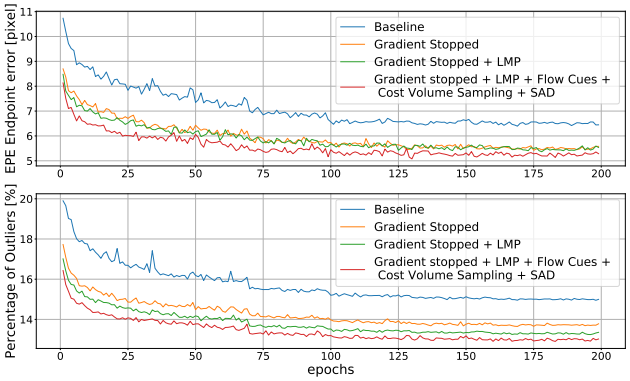

Figure 10: Qualitative comparison of training curves on Things 3D pre-training for optical flow with and HD^3 baseline. Large drop from Baseline to Gradient Stopped version on EPE and percentage of outliers (Fl-all). LMP improves mainly on Fl-all; adding all the rest of the elements gives additional boost on EPE and Fl-all.

A.3. Histogram of Errors

Fig. 11 shows the gains made over the KITTI training sequence as achieved with our submitted model that uses all our proposed five elements. The gains are made visible in form of histograms, where the ground truth flow magnitude is used for the binning. As can be seen, our improvements are not limited to a single range of flow magnitudes but affect the whole spectrum of flow vectors. At this point we want to remind the reader, that adding our elements hardly changes the number of learnable parameters (e.g. $\approx +1\%$ for HD^3) in the network. The gains therefore result from using the provided parameters more effectively via our flow elements.

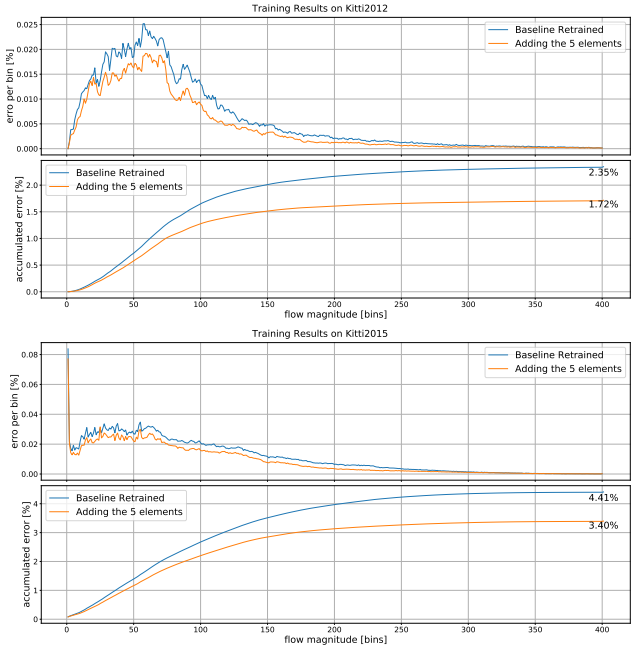

Figure 11: Histogram of errors on the training data of KITTI 2012 (upper half) and KITTI 2015 (lower half). The errors are grouped in bins according to the ground truth flow magnitude on which they occurred. Adding all five elements improves consistently in all areas.

A.4. Ablations on Contributions of Flow Cues.

Here we evaluate the impact of our proposed flow cues in comparison to related ones from prior works [13, 15], demonstrating their effect on relevant error measures on the Flying Chairs2 dataset. The ablations are performed training on Flying Chairs 2. We use averages over the last 10 validation results to reduce the effect of single spikes. In Tab. 4 we list our findings, always on top of activating the elements of gradient stopping and SAMPLING due to its preferable behavior for estimating flow of fine-grained structures.

Providing *Mapping Occurrence Density* (MOD) [36, 37] as the only Flow Cue and hence information about the occlusions an dis-occlusions slightly degrades results in terms of both, EPE and Fl-all. When running the Sampling in combination with Forward-Backward flow warping (FWDBWDFW) we encounter a considerable reduction of errors – particularly on the Fl-all errors. Finally, when combining SAMPLING with all our proposed flow cues (ALL CUES), *i.e.* reverse flow estimation, mapping occurrence density, and out-of-image occlusions, we obtain the lowest errors.

A.5. Sidenote: Sampling vs. Warping – HD^3's D2V and V2D Operations.

One of the key innovations in the HD^3 [43], was the introduction of the D2V and V2D operations that allow to

MOD	FWDBWDFW	ALL CUES	EPE	Fl-all
✗	✗	✗	1.208	6.192
✓	✗	✗	1.217	6.271
✗	✓	✗	1.202	6.171
✓	✓	✓	1.186	6.156

Table 4: Ablation results on Flow Cues on top of Cost Volume Warp and Gradient Stopping

transform match densities into vectors and vice versa. This operation is used for absolute and residual flows and implicitly assumes an equidistant fixed grid spacing for the flow. However, this assumption is actually not always valid since the warping operation can deform the space over which the search window operates in the warped image $I_{2\to1}$. I.e. a movement of a single pixel in the search window in $I_{2\to1}(x)$ can move the correspondence to a completely different position in $I_2(y)$ dependent on the flow $F_{2\to1}(x)$ that was used for the warping.

In the case of sampling, the equidistance of the grid is preserved, since it always uses a single flow vector $F_{2\to1}(x)$ as offset for the entire search window for each individual pixel. Therefore, the spacing of the search window stays equidistant w.r.t. $I_2(y)$ and hence for the D2V and V2D operations.

A.6. Qualitative Comparisons of Training Results on KITTI

In this section various qualitative results on the KITTI training images will be shown. Fig. 12 shows comparisons between the baseline model as taken from the HD^3 modelzoo and our best model that uses all five elements. What can be seen especially well in the error plots, is that our model manages improves a lot on the moving cars. Furthermore, it improves on fine details, which can e.g. be seen e.g. at the guard rails, where it manages to keep sharper edges and a more homogeneous background. At the same time, it does not suffer from the artifacts present in the top region of the baseline model. The figures are best viewed in high-resolution on a PC.

A.7. Results on MPI-Sintel

We outperform the state-of-the-art on the challenging MPI-Sintel Dataset. Fig. 14 shows the *Results and Rankings* for MPI-Sintel test results at the time of submission to the server. For more details please refer to the main paper.

Fig. 13 shows the comparison of a HD^3 baseline model and the improved baseline with our elements trained on the MPI-Sintel training sequence. As can be seen our improved model allows to preserve more fine details like the stick in the bamboo scene or the pike. Also, it seems to be better at detecting and correcting hardly connected moving backgrounds that seem to cause problems for the modelzoo baseline.

References

[1] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, Mar 2011. 2

[2] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004. 2

[3] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012. 2

[4] K. Chaudhury and R. Mehrotra. A trajectory-based computational model for optical flow estimation. *IEEE Transactions on Robotics and Automation*, 11(5):733–741, Oct 1995. 2

[5] Piotr Dollár and C. Lawrence Zitnick. Structured forests for fast edge detection. In *(ICCV)*, 2013. 2

[6] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[7] Denis Fortun, Patrick Bouthemy, and Charles Kervrann. Optical flow modeling and computation. *Comput. Vis. Image Underst.*, 134(C):1–21, May 2015. 2

[8] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. A variational approach to video registration with subspace constraints. *International Journal of Computer Vision*, 104(3):286–314, Sep 2013. 2

[9] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *(IJRR)*, 2013. 2, 6

[10] G. E. Hinton, So. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *Deep Learning Workshop, NIPS*, 2014. 3, 6

[11] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *ARTIFICAL INTELLIGENCE*, 17:185–203, 1981. 2

[12] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A Lightweight Optical Flow CNN - Revisiting Data Fidelity and Regularization. *arXiv preprint arXiv:1903.07414*, 2019. 1, 2

[13] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, 2019. 1, 2, 3, 8, 9, 10

[14] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 2, 8

[15] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *ECCV*, 2018. 2, 3, 10

[16] Snchez Javier, Salgado Agustn, and Monzn Nelson. Direct estimation of the backward flow. Technical report, Institute

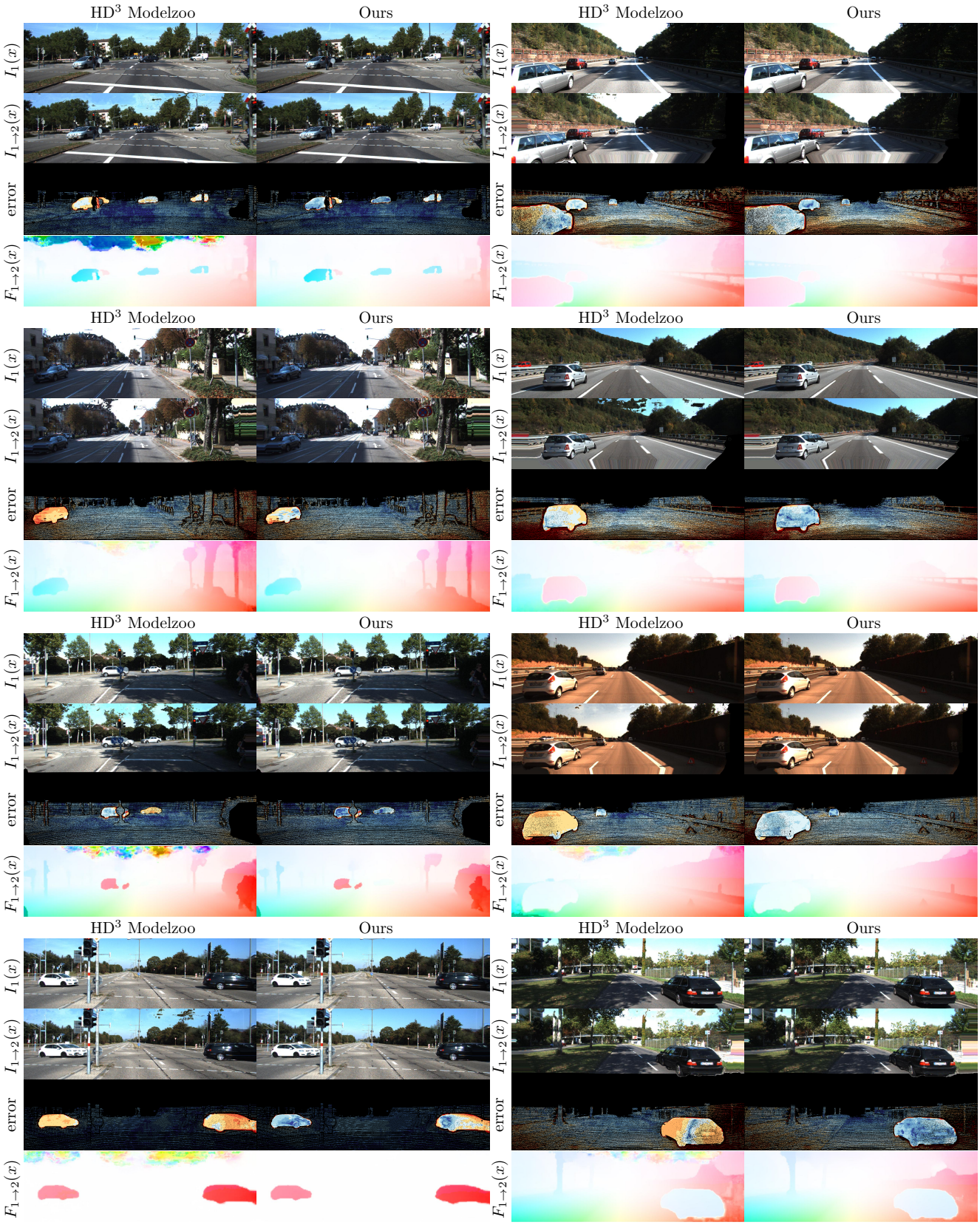

Figure 12: Comparisons on the KITTI 2015 training set. Theirs = HD^3 baseline, ours HD^3 with our elements.

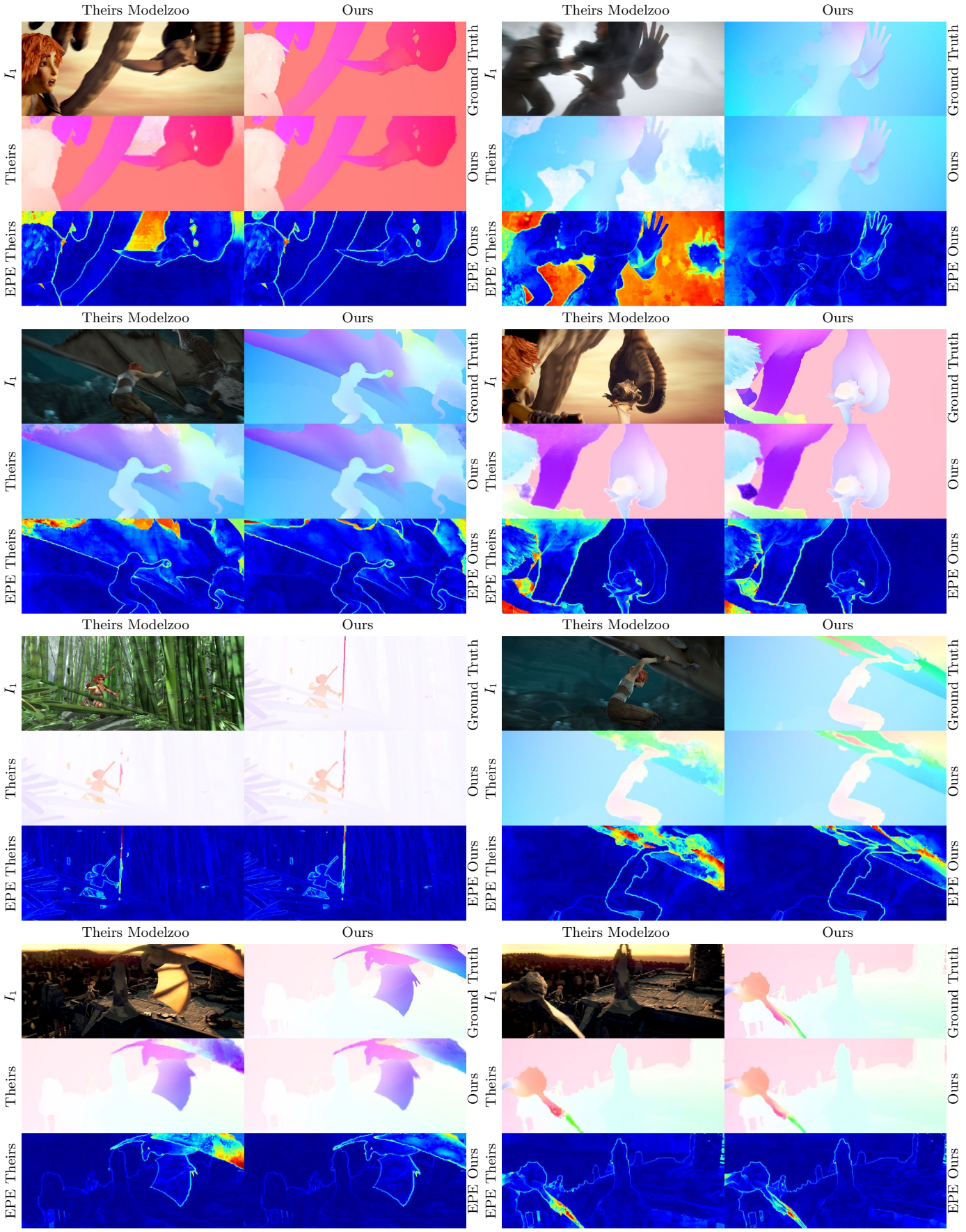

Figure 13: Comparisons on the MPI-Sintel training set. Theirs = HD^3 baseline model from the modelzoo. Ours = with our elements additionally on top (Except for Data Distillation since Sintel has dense GT).

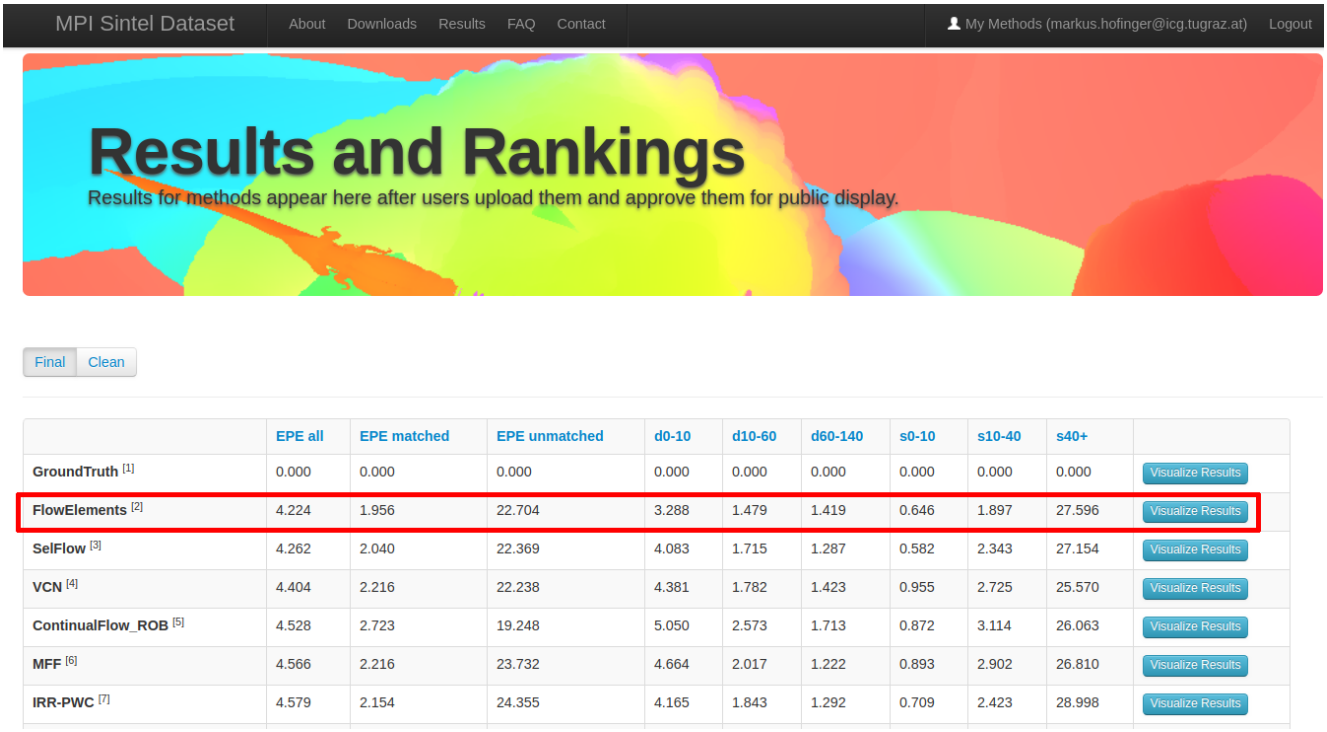

Figure 14: MPI-Sintel *Results and Rankings* - our method improves upon the state of the art.

for Systems and Technologies of Information, Control and Communication, Las Palmas de Gran Canaria, 2013. 4

[17] Huaizu Jiang, Deqing Sun, Varun Jampani, Zhaoyang Lv, Erik Learned-Miller, and Jan Kautz. Sense: A shared encoder network for scene-flow estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2, 3

[18] Pengpeng Liu, Irwin King, Michael R. Lyu, and Susan Jia Xu. Ddflow: Learning optical flow with unlabeled data distillation. *CoRR*, abs/1902.09145, 2019. 2, 3

[19] Pengpeng Liu, Michael R. Lyu, Irwin King, and Jia Xu. Selflow: Self-supervised learning of optical flow. In *CVPR*, 2019. 2, 3, 4, 6, 8

[20] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of Imaging Understanding Workshop*, pages 4884–4893, 1981. 2

[21] O. Mac Aodha, A. Humayun, M. Pollefeys, and G. J. Brostow. Learning a confidence measure for optical flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(5):1107–1120, May 2013. 3

[22] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015. 2

[23] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018. 2

[24] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *(NIPS)*, 2015. 3

[25] Zhile Ren, Orazio Gallo, Deqing Sun, Ming-Hsuan Yang, Erik B. Sudderth, and Jan Kautz. A fusion approach for multi-frame optical flow estimation. In *IEEE Winter Conference on Applications of Computer Vision*, 2019. 2, 8

[26] Jérôme Revaud, Philippe Weinzaepfel, Zaïd Harchaoui, and Cordelia Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. *CoRR*, 2015. 2

[27] S. Rota Bulò, G. Neuhold, and P. Kontschieder. Loss max-pooling for semantic image segmentation. In *(CVPR)*, July 2017. 2

[28] S. Rota Bulò, G. Neuhold, and P. Kontschieder. Loss max-pooling for semantic image segmentation. *arXiv preprint arXiv:1704.02966*, 2017. 6

[29] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. In-place activated batchnorm for memory-optimized training of DNNs. In *(CVPR)*, 2018. 4, 7

[30] S. Rota Bul, L. Porzi, and P. Kontschieder. Dropout distillation. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 99–107, New York, New York, USA, 20–22 Jun 2016. PMLR. 6

[31] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *CVPR*, pages 2432–2439. IEEE, June 2010. 2

[32] Deqing Sun, Stefan Roth, and Michael J. Black. 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. 2

[33] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns

for optical flow estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018. to appear. 1, 3, 9

[34] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation, 2018. 2

[35] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 2, 4, 8

[36] M. Unger, M. Werlberger, T. Pock, and H. Bischof. Joint motion estimation and segmentation of complex scenes with label costs and occlusion modeling. In *(CVPR)*, 2012. 2, 4, 10

[37] Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. Occlusion aware unsupervised learning of optical flow. In *CVPR*, pages 4884–4893, 06 2018. 3, 4, 10

[38] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE Intenational Conference on Computer Vision (ICCV)*, Sydney, Australia, Dec. 2013. 2

[39] Manuel Werlberger, Werner Trobin, Thomas Pock, Andreas Wedel, Daniel Cremers, and Horst Bischof. Anisotropic huber-l1 optical flow. In *Proceedings of the British Machine Vision Conference (BMVC)*, London, UK, September 2009. to appear. 2

[40] Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *ECCV*, 2014. 3

[41] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *Advances in Neural Information Processing Systems 32*, pages 793–803. Curran Associates, Inc., 2019. 2, 8

[42] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *(CVPR)*, 2019. 1, 2, 6

[43] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *CVPR*, 2019. 7, 8, 10

[44] Jason J. Yu, Adam W. Harley, and Konstantinos G. Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *Computer Vision - ECCV 2016 Workshops, Part 3*, 2016. 3, 4

[45] Jean yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000. 1