Unsupervised Learning of Optical Flow with Deep Feature Similarity

Woobin Im[1], Tae-Kyun Kim[2,1], and Sung-Eui Yoon*[1]

[1] School of Computing, KAIST, Daejeon, South Korea
iwbn@kaist.ac.kr, sungeui@kaist.edu
[2] Department of Electrical and Electronic Engineering, Imperial College London, UK
tk.kim@imperial.ac.uk

Abstract. Deep unsupervised learning for optical flow has been proposed, where the loss measures image similarity with the warping function parameterized by estimated flow. The census transform, instead of image pixel values, is often used for the image similarity. In this work, rather than the handcrafted features i.e. census or pixel values, we propose to use deep self-supervised features with a novel similarity measure, which fuses multi-layer similarities. With the fused similarity, our network better learns flow by minimizing our proposed feature separation loss. The proposed method is a polarizing scheme, resulting in a more discriminative similarity map. In the process, the features are also updated to get high similarity for matching pairs and low for uncertain pairs, given estimated flow. We evaluate our method on FlyingChairs, MPI Sintel, and KITTI benchmarks. In quantitative and qualitative comparisons, our method effectively improves the state-of-the-art techniques.

Keywords: Unsupervised, self-supervised, optical flow, deep feature, similarity

1 Introduction

In computer vision, optical flow estimation is a fundamental step towards motion understanding. It describes the velocity of each point in the 3D world as the projection of points to the 2D motion field. Thanks to its effective motion description, it has been largely used for many applications, e.g., video recognition [31, 7], frame interpolation [4, 36], and inpainting [38], to name a few.

Recently, end-to-end deep networks for optical flow estimation [9, 16, 28, 32] have made architectural progresses, resulting in significant improvements in terms of flow accuracy, efficiency, and generalization capability. For supervised optical flow training, large-scale synthetic datasets, e.g., FlyingChairs [9], have been primarily used. While there are real-world datasets including Middlebury [4] and KITTI [10, 26], their sizes are limited to few hundred images and each of them is limited to a specific scenario. This limitation is mainly due to the extremely prohibitive cost to obtain or manually label accurate matching points in thousands of video frames in the wild.

* Corresponding author

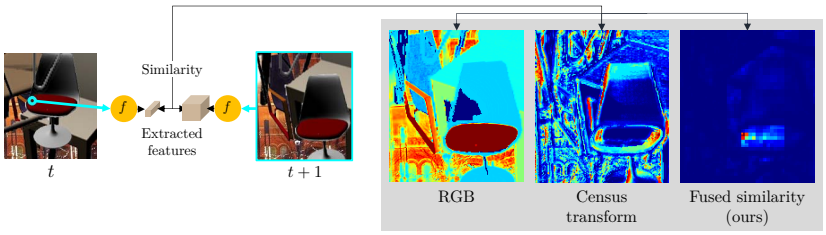


Fig. 1: This figure shows different similarity maps of the reference point at time step t to all pixels in the target image at $t+1$; red means higher similarity. We compare the similarity computed by our deep feature against ones computed by census transform and RGB. The fused similarity shows improved discriminative response, while the census transform tends to be sensitive to local edge appearance and RGB shows high similarity with similar colors. For simple visualization, we compute similarity in the spatial domain

To train deep networks without ground-truth flows, unsupervised approaches have been proposed. In principle, unsupervised methods exploit an assumption that two matching points have similar features and learn to generate flows maximizing the similarity. In this line of research, choosing appropriate features is critical for accurate optical flow estimation. The early work [19, 29] applies RGB pixel values and image gradients as the feature, and recently it has been shown that the census transform [39] is highly effective for optical flow learning [25, 22].

Another interesting aspect of the unsupervised optical flow networks is that a network learns more than just the loss it is trained with. A recent work [22] found that configuring the loss function only with the data term does work without the smoothness term. This observation implies that the network feature learns meaningful patterns in moving objects only with the photometric constancy assumption. A similar observation is also found in the literature on self-supervision [27, 11], where deep features learn semantic patterns by conducting simple unsupervised tasks like a jigsaw puzzle or guessing rotation.

In this work, we learn self-supervised network features and use them to improve the unsupervised optical flow. To learn from self-features, we propose to use the similarity based on the product fusion of multi-layer features (Sec. 3.2). We visualize similarity maps computed by different features in Fig. 1. Our fused similarity demonstrates discriminative matching points highlighting the matching pair, while lessening unmatched areas. On the other hand, the similarity map (e.g., computed by the cosine) with RGB or the census transform shows many high-response points across the whole area. This is mainly because those features only encode patterns in a local area and do not represent semantic meanings. We propose three loss functions utilizing the feature similarity for optical flow training (Sec. 3.3). Across various quantitative and qualitative validations, we demonstrate the benefits of the proposed feature similarity and the loss function. (Sec. 4). When compared to other deep unsupervised methods,

our method achieves state-of-the-art results under various measures across FlyingChairs, MPI Sintel, and KITTI benchmarks.

2 Related Work

End-to-end supervised deep methods. FlowNet [9] is the first end-to-end framework that exploits a deep network for optical flow estimation. To train the network, a large-scale labeled dataset called FlyingChairs was constructed [9]. Following the first work, FlowNet2 [16], SpyNet [28], and PWC-Net [32] have made architectural progresses, resulting in significant improvements in terms of flow accuracy, efficiency, and generalization capability.

Datasets. Large-scale synthetic datasets are available for supervised optical flow training including FlyingChairs [9]. Following FlyingChairs, FlyingThings3D [24] and FlyingChairs-Occ [15] datasets are incorporated into the collection with improved reality and additional information. Real-world datasets annotated with a rich optical flow are lacking. Middlebury [4] and KITTI [10, 26] are the most commonly used ones. However, not only the scenes they have are limited to few hundreds of images, but also they are constrained to specific scenarios: indoor static objects for Middlebury and driving for KITTI. This limitation is mainly due to the prohibitive cost to obtain or manually label accurate matching points in thousands of video frames in the wild.

End-to-end unsupervised deep methods. To make use of deep networks for optical flow without expensive ground-truth flows, deep unsupervised approaches have been proposed. Earlier methods [19, 29] brought ideas from the classical variational methods, which adopt the energy functional containing the data and smoothness terms into loss functions of deep learning. The loss function in unsupervised methods can be calculated using the warping technique [17].

In unsupervised optical flow learning, how to filter out unreliable signals from its loss is critical for achieving better results. In terms of the data term, the census transform [39] has been proven to be effective in deep unsupervised optical flow [25, 22]. Similarly, occlusion handling [25, 34] can eliminate possibly occluded points from the loss calculation, where no target pixels exist due to occlusion. Rather than just giving up supervision on occluded ones, hallucinated occlusion [22, 23] can be helpful to give meaningful loss for those occluded points. Additionally, training with multiple frames improves the noisy loss signal. Janai et al. [18] use the constant velocity assumption and Liu et al. [23] build multiple cost-volumes to give more information to a model.

Deep features for matching. In the fields of matching and tracking keypoints or objects, deep network features have been used for robust matching [33, 5, 42]. At a high level, matching techniques are related to optical flow estimation [35, 30, 2, 3, 37, 13]. However, addressing all the pixels and their matching in a dense manner (cf. sparse keypoints) is challenging. To the best of our knowledge, deep features have not been exploited in objective functions for optical flow estimation. On the other hand, there has been successful exploitation of deep features in pixel-wise tasks, e.g., depth estimation [41] and generation by warping [14]. In

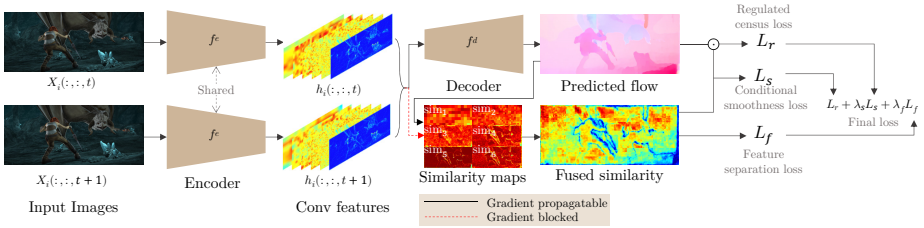


Fig. 2: This shows the overview of our method, which is end-to-end trainable for both optical flow and self-supervised deep features

our evaluations, the Zhan's method [41], which was proposed for stereo matching, or the losses more direct to optimize the deep features, e.g., the triplet losses, have shown poor performance in unsupervised optical flow estimation. Given the high level of noise due to occlusion, deformation, motion blur, and the unsupervised settings of optical flow, the level of optimization, i.e., the loss function for features, needs to be carefully designed.

3 Our Approach

In this work, we successfully learn and exploit deep features for improving unsupervised optical flow estimation. The proposed framework simultaneously improves the features and optical flow while adding a small additional cost for training and no extra cost at runtime. For effective training, we build new loss functions (Sec. 3.3) utilizing the fused similarity (Sec. 3.2). The feature separation loss separates certain and uncertain matchings by our fused similarity like a contrastive loss. Noting that even the best features can fail in cases like occlusions, the separation mechanism effectively improves resulting flows by discouraging to match uncertain pixels, as well as encoraging to improve certain ones. Additionally, the regulated census loss and conditional smoothness loss make use of the census features and the smoothness constraint adaptively considering the deep similarity to compensate for the low precision of deep features.

Figure 2 illustrates our method. Our method is an end-to-end trainable network, which takes a sequence of images as an input for optical flow estimation; we use PWC-Net [32] structure as a base model of the encoder and the decoder, and train it using self-features, i.e., spatial conv features. In the training phase, we add a similarity branch in which similarities of predicted flows are calculated and fused; the resulting fused similarity is actively used in our training process. The aggregated similarity over layers resolves disagreement among multiple-layer features. We apply three loss functions to be minimized upon the fused similarity: feature separation loss, regulated census loss, and conditional smoothness loss. In the learning process, both the encoder and decoder are initialized using the conventional unsupervised optical flow loss. Then, the proposed feature-based losses are minimized. The method converges under different initialization and parameter settings.

3.1 Background on Unsupervised Optical Flow Learning

The learning-based optical flow method commonly works on a dataset to train a model that has a set of spatio-temporal images $\mathcal{X} = \{X_1, X_2, \ldots, X_N\}$, $X_i \in \mathbb{R}^{H\times W\times T\times C}$, and ground-truth flows $\mathcal{Y} = \{F_1, F_2, \ldots F_N\}$, $F_i \in \mathbb{R}^{H\times W\times T-1\times 2}$, where H and W denote height and width, T is its sequence length, and C and N are the numbers of channels and data, respectively. The goal is to train a model f_θ that calculates flow $\hat{F}_i$ from the spatio-temporal sequence $X_i \in \mathcal{X}$. In a supervised case, we train a network by minimizing regression loss: $L_s = \frac{1}{N}\sum_i^N ||F_i - \hat{F}_i||_2^2$. In an unsupervised case, however, we cannot access $\mathcal{Y}$, but only $\mathcal{X}$. We thus configure an unsupervised loss term, L_p, with the photometric consistency assumption:

$$L_p = \frac{1}{N}\sum_i^N \sum_{(x,y,t)\in\Omega} \Psi(X_i(x,y,t) - X_i(x+u, y+v, t+1)), \tag{1}$$

where Ω contains all the spatio-temporal coordinates, $(u, v) = \hat{F}_i(x, y, t)$ is an estimated flow at (x, y, t), and Ψ is the robust penalty function [22]; $\Psi(x) = (|x|+\epsilon)^q$. Note that $X(x+u, y+v, \cdot)$ includes the warping operation using bilinear sampling [17], which supports back-propagation for end-to-end optimization. In this paper, we use the census transform in L_p with the same configuration used in [22] for all experiments unless otherwise stated.

Occlusion handling is performed by checking consistency [25] between forward and backward flows. The estimated occlusion mask is denoted with $\hat{C}_i^o(x, y, t)$, which is 1 if $\hat{F}_i(x, y, t)$ is not occluded, otherwise 0. This geometrically means the backward flow vector should be the inverse of the forward one if it is not occluded. Our loss terms use the occlusion map as previous work [25].

In this work, we use the data distillation loss [22] for occluded pixels:

$$L_d = \frac{1}{N}\sum_i^N \sum_{(x,y,t)\in\Omega} \Psi(\hat{F}_i^s(x,y,t) - \hat{F}_i^t(x,y,t))M_f(x,y,t), \tag{2}$$

where $\hat{F}_i^t$ is a flow from a teacher model, $\hat{F}_i^s$ is a flow from a student model, and M_f is a valid mask. In short, the teacher model processes inputs without artificial synthetic occlusion, and the student model gets inputs with the generated occlusion. The student flow then learns from the teacher flow.

3.2 Feature Similarity from Multiple Layers

We use a model f_θ that estimates optical flow, where the model can be decomposed into an encoder $f_{\theta_e}^e$ and a decoder $f_{\theta_d}^d$, such that $f_\theta(X_i) = f_{\theta_d}^d(f_{\theta_e}^e(X_i))$; $\theta = \theta_e \cup \theta_d$. The encoder f^e is a function that outputs L-layered features $h_i = (h_i^1, h_i^2, \ldots, h_i^L)$; a lower numbered layer indicates a shallower layer in the deep network.

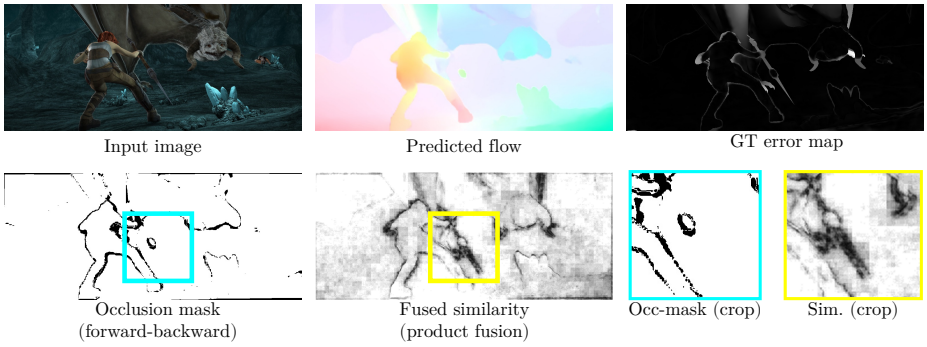


Fig. 3: We visualize the occlusion mask ($\hat{C}^o$) and the similarity map (sim_f). By comparing the ground-truth error map and the fused similarity, we can observe that the similarity is low when the error is high. The bottom row shows occlusion mask and the similarity. Since the occlusion mask does not consider matching confidence, it does not represent how confident the matching is. On the other hand, our fused similarity marks whether the predicted flow is confident. Furthermore, we can use back-propagation since the similarity is differentiable.

Given a matching $\hat{F}(x, y, t)$, we define the similarity between $X_i(x, y, t)$ and $X_i(x + u, y + v, t + 1)$ in a layer l to be:

$$\text{sim}_l(x, y, t; h_i^l, \hat{F}_i) = \frac{h_i^l(x, y, t) \cdot h_i^l(x + u, y + v, t + 1) + 1}{2}, \tag{3}$$

where $(u, v) = \hat{F}_i(x, y, t)$. Since we use l2-normalization for h_i^l, the function $\text{sim}(\cdot)$ is equivalent to the normalized cosine similarity between the reference feature and the target feature. Note that, for flows going out of the frame, no gradient is propagated during training. Additionally, we update encoder weights by indirect gradient, i.e., back-propagation through the decoder, while ignoring direct gradient from sim_l to h_i^l. That strategy is chosen because the direct gradient easily bypasses the decoder, and the encoder easily suffers from overfitting, in the end, the output flow is downgraded.

Fused similarity. To utilize features from all layers, we propose to fuse multi-layer feature similarities ($\text{sim}_1, \text{sim}_2, \ldots, \text{sim}_L$). In CNNs, lower layer features tend to show high response for low-level details, e.g. edge, color, etc., while higher layers focus on objects [40]. As a result, similarity response of a lower layer usually bursts around the whole image, while similarity of a deeper layer has few modes. Therefore, to get stable, yet discriminative feature similarity response as shown in Fig. 1, we define the fused similarity as product of multiple features:

$$\text{sim}_f(\cdot) = \prod_{l=1}^{L} \text{sim}_l(\cdot; h_i^l, \hat{F}_i). \tag{4}$$

For efficient calculation during training, we downsample the flow field $\hat{F}_i$ to the size of each layer feature using area interpolation before calculating the similarity; we use area interpolation, since it can propagate gradient to all source points, while other interpolation methods, e.g., bilinear interpolation, only update few nearest source points.

3.3 Learning Optical Flow with Feature Similarity

In this section, we propose three loss functions to effectively use the similarity map for optical flow estimation.

Feature separation loss. Given deep similarity, a model can learn flow by simply maximizing $\mathrm{sim}(\cdots; h_i, \hat{F}_i)$, since larger similarity possibly means better matching solution. However, this simple approach in practice leads to worse results, because matchings between pixels under occlusion do not get better, even as we increase the similarity. In other words, maximizing the similarity for these points makes the flows incorrectly matched to random pixels with higher similarity.

To address this matching issue with uncertainty, we suppress flows with lower similarity by minimizing their similarity further, while refining flows with higher similarity by maximizing their similarity. First, we define a similarity threshold k, which we separate the values from:

$$k = \frac{1}{2}(k_{noc} + k_{occ}), \tag{5}$$

where k_{noc} and k_{occ} are average similarities of non-occluded pixels and occluded pixels: $k_{noc} = \frac{\sum_\Omega(\mathrm{sim}_f \cdot \hat{C}_i)}{\sum_\Omega(\hat{C}_i)}$, $k_{occ} = \frac{\sum_\Omega(\mathrm{sim}_f \cdot (1-\hat{C}_i))}{\sum_\Omega(1-\hat{C}_i)}$. Since occlusion is an effective criterion to set the boundary value, so that other kinds of difficulties can also be covered as shown in the experiments (Fig. 6).

We then formulate the feature separating loss term as:

$$L_f = \frac{1}{N}\sum_{i}^{N} \sum_{(x,y,t)\in\Omega} -(\mathrm{sim}_f(x,y,t) - k)^2. \tag{6}$$

L_f is a quadratic loss function encouraging the similarity to be far from k, which serves as a boundary value that decides the direction of update. In other words, it suppresses uncertain flows, i.e. $\mathrm{sim}_f(x,y) < k$, down towards 0, and certain flows, i.e. $\mathrm{sim}_f(x,y) > k$, up towards 1. This can be also interpreted as minimizing entropy in semi-supervised learning [12]; we make a network output more informative by regularizing the similarity to be at each polar. A similar approach separating feature similarity from a different domain, image retrieval, has been shown to be effective [21].

One may concern that minimizing the similarity of uncertain flows, i.e., $\mathrm{sim}_f < k$, can lead to an arbitrary matching solution. However, the product operation in the fused similarity (Eq. 4) makes sim_l with a higher similarity

relatively retained, while changing smaller similarity much faster. Given any a, b s.t. $1 \leq a, b \leq L$, whose similarity is not 0, one can derive the following equation:

$$\frac{\partial L_f}{\partial \text{sim}_a(x,y,t)} = \frac{\text{sim}_b(x,y,t)}{\text{sim}_a(x,y,t)} \left(\frac{\partial L_f}{\partial \text{sim}_b(x,y,t)} \right). \tag{7}$$

That is, the scale difference between the two gradients is proportional to the fractional ratio of them, which can grow much faster when the denominator becomes smaller in the scale of the multiplicative inverse. As a result, L_f is minimized by the smaller similarity approaching zero; higher layer similarities can be preserved to prevent arbitrary matching.

Regulated census loss. Since L_f (Eq. 6) is fully self-regulated, using only L_f for training can mislead the network itself. Thus, by modifying the well-known unsupervised loss (Eq. 1), we additionally use a regulated census loss, L_r, controlled by similarity:

$$L_r = \frac{1}{N} \sum_{i}^{N} \sum_{(x,y,t) \in \Omega} \Psi(\cdot) \hat{C}_i^o(x,y,t) \text{sim}_f(x,y,t). \tag{8}$$

The warping operation used in Eq. 1 is the bilinear sampling. This can only take into account four nearest pixels, making it difficult to address the pixel position far from the estimation, as also pointed by Wang et al. [34]. Therefore, the unsupervised loss (Eq. 1) does not give a correct direction when the current estimation is far from the desired target flow; whatever the loss is, it would be a noise in that case. In contrast, deep features have larger receptive fields with global context, so does the fused similarity. Thus, we can suppress the noise signal by multiplying the similarity; the similarity is designed to indicate whether the current estimation is near the desired target point.

Conditional smoothness loss. We use the smoothness prior for spatial locations with low similarity. In general, using the smoothness prior for all pixels degrades the accuracy because the flow-field is blurred. Meanwhile, if clear matching is not found, the smoothness constraint can help by being harmonized with surrounding flows. We thus define our smoothness prior loss considering similarity as:

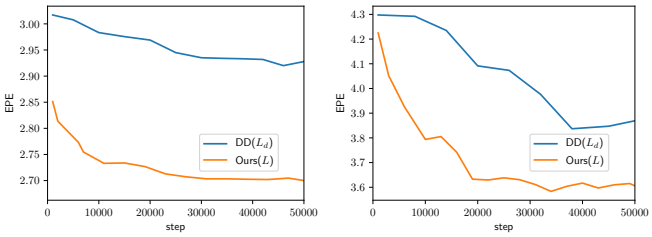
$$L_s = \frac{1}{N} \sum_{i}^{N} \sum_{(x,y,t) \in \Omega} (|\nabla u|^2 + |\nabla v|^2) M_l(x,y,t), \tag{9}$$

$$M_l(x,y,t) = \begin{cases} 1, & \text{if } \text{sim}_f(\cdot) < k, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Loss for training. We jointly use the aforementioned losses to train our model with stochastic gradient descent. Our final loss function is sum of these loss functions:

$$L = L_r + \lambda_f L_f + \lambda_s L_s + \lambda_d L_d, \tag{11}$$

where λs are weight parameters, and L_d is the data distillation loss defined in Eq. 2.



(a) FlyingChairs (b) Sintel

Fig. 4: Training graphs of end-point-error (EPE) on two datasets. For DD(L_d), we use census, occlusion handling and L_d, which is the same setting to [22]. For ours(L), we use the full loss function L (Eq. 11). Ours performs consistently better during training

4 Experimental Results

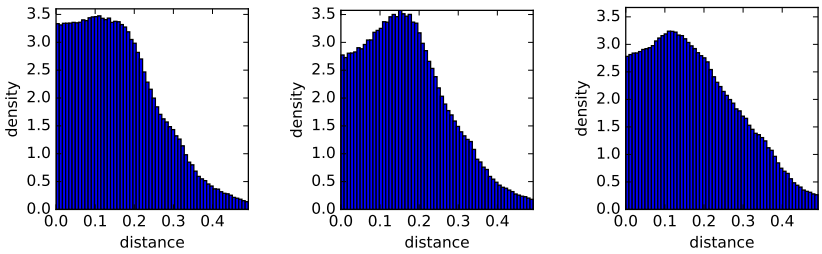
Our network structure is based on PWC-Net [32], which is a deep network that contains warping, cost-volume, and context network to cover large displacements. We train the network from scratch using Adam optimizer [20] for stochastic gradient descent. In all experiment, we set the mini-batch size to 4.

Training procedure. Overall, we follow the training process of DDFlow [22] to initialize the network. To train the model, we first pretrain our network with FlyingChairs [9] and finetune it with each target dataset. For pretraining, we use the conventional photometric loss with RGB (Eq. 1) for 200k steps and additional 300k steps with occlusion handling. The resulting weights become the base network parameters for the following experiments. Next, using each target dataset, we finetune the base network. From this stage, we use the census transform for photometric loss. We train the model for 200k steps with occlusion handling. We then apply the final loss L (Eq. 11). We run first 1k steps without L_d, i.e. $\lambda_d = 0$. Then, the teacher network (Sec. 3.1) is fixed to use L_d and continues training using all the losses to 50k steps. We set hyper-parameters to $\lambda_s = 10^{-4}$ and $\lambda_f = 4$. We follow $\lambda_d = 1$ from the previous work [22].

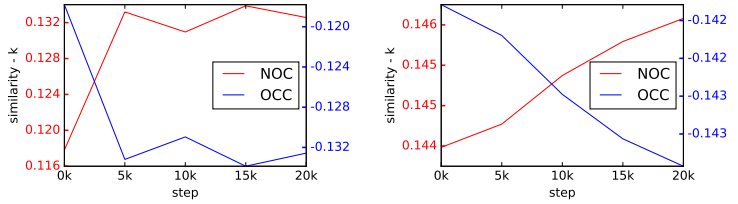
Data augmentation. For better generalization, we augment the training data using random cropping, random flipping, random channel swapping and color jittering; color jittering includes random brightness and saturation. We normalize the input RGB value into $[-0.5, 0.5]$.

FlyingChairs. The FlyingChairs [9] is a synthetic dataset created by combining chair and background images. The dataset consisting of 20 k pairs of images has a given train/test split, thus we use its training set for training our network and evaluate our model on its test set.

MPI Sintel. MPI Sintel [6] is a rendered dataset, originally from an open-source movie, Sintel. For training, we use 1k images from the training set and upload our results of the test set to its benchmark server for evaluation. We use both versions of rendering, i.e., clean and final, for training.



(a) 0 step (4.16) (b) 20k step (3.76) (c) GT, 0 step (0.0)



(d) Estimated flows (e) GT flows

Fig. 5: (a-c) We measure the distance from the boundary k (Eq. 5) to fused similarity on Sintel Final dataset. (a-c) show distance distributions of different steps; we show EPE inside parenthesis. During training, our feature separation loss pushes fused similarity away from the boundary k; note that the greater the distance is, the more separation we have. (d-e) show average $\text{sim}_f - k$ for occluded / non-occluded pixels. Note that occluded pixels show negative value. We measure the distance using ground-truth (GT) flows to observe the effect mainly on the encoder feature excluding the effect from the decoder side.

KITTI. KITTI [10] has a driving scene from the real world. This dataset has only 200 pairs of images with ground-truth flows. We thus train our model using unlabeled images from a multi-view extension set of KITTI without duplicated images in the benchmark training or testing sets, following the previous work [34].

4.1 Evaluation on Benchmarks

Ablation study. The results on benchmark datasets show that the network better learns to estimate flows with the feature similarity (Table 1). As can be seen in the last row of the table, our final method ($L_d + L_f + L_r$) works better in most cases than the other settings on both datasets. Due to low localization precision of deep features, however, using only L_f without L_r does not much improve the result. Interestingly, L_r that adaptively regulates the the conventional census loss is highly effective. When L_r is combined together with L_f, the combined loss ($L_f + L_r$) performs the best.

In Sintel where we report EPE on NOC and OCC, L_f improves better on OCC than on NOC. It implies that the suppression part in L_f takes effect, which

Table 1: Ablation study on various settings. Average end-point error (EPE) is used as a metric. For Sintel, the results are evaluated over all (ALL), non-occluded (NOC), and occluded (OCC) pixels. Results in parenthesis are achieved by testing the model using the data that the model is trained on. In left columns we show types of losses we use: occlusion handling ($\hat{C}$), data-distilation [22] (L_d), and ours: feature separation (L_f) and regulated census (L_r). The first two rows show the performance of our pretrained network trained with FlyingChairs

Base feature	$\hat{C}$	L_d	L_f	L_r	FlyingChairs	Sintel Clean			Sintel Final		
					ALL	ALL	NOC	OCC	ALL	NOC	OCC
RGB					4.01	5.61	3.01	38.64	6.44	3.76	40.45
RGB	✓				3.64	4.40	2.12	33.33	5.42	3.02	36.01
Census	✓				3.10	(3.22)	**(1.26)**	(28.14)	(4.37)	(2.25)	(31.25)
Census	✓	✓			2.93	(3.15)	(1.49)	(24.35)	(3.86)	(2.11)	(26.16)
Census	✓	✓	✓		2.87	(3.25)	(1.46)	(25.95)	(4.15)	(2.27)	(28.01)
Census	✓	✓		✓	2.81	(2.91)	(1.29)	(23.40)	(3.62)	(1.95)	(24.98)
Census	✓	✓	✓	✓	**2.69**	**(2.86)**	(1.28)	**(22.85)**	**(3.57)**	**(1.94)**	**(24.38)**

Table 2: Average EPE on different displacements. ALL: all pixels. *a*-*b*: pixels displaced within $[a, b)$

	Sintel clean				Sintel final			
	ALL	0-10	10-40	40+	ALL	0-10	10-40	40+
L_p	(3.22)	(0.59)	(3.86)	(20.45)	(4.37)	(0.83)	(5.41)	(27.17)
L_d	(3.15)	(0.59)	(4.02)	(19.51)	(3.86)	(0.71)	(5.06)	(23.63)
Ours	**(2.86)**	**(0.49)**	**(3.45)**	**(18.36)**	**(3.57)**	**(0.64)**	**(4.49)**	**(22.36)**

discourages matching with higher similarity for uncertain flows. Table 2 shows that our method effectively covers various ranges of displacements.

We have also tested direct feature learning by the triplet loss [8], it did not improve the baseline accuracy DDFlow [22] i.e. L_d+L_p alone in the experiments. The proposed losses and learning strategy are crucial to unsupervised learning of feature and optical flow.

Loss parameter. We show results with different parameter settings in Table 3. Our smoothness constraint helps the network refine flows with lower similarity, when set to lower weight values; with a high weight value, the flow field becomes over-smoothed. With respect to λ_f, a weight greater than 4 can deteriorate the learning. As can be seen in Figure 4, our final loss (Eq. 11) converges well and it achieves higher performance than the baseline of using L_d.

Analysis on similarity. During training, our method gradually improves flow and encoder feature. Figure 5 illustrates the fused similarity with respect to each training step. Fig. 5a shows the distribution before we apply our feature

Table 3: Average EPE depending on loss weighting parameters

(a) Smoothness weight

λ_s	10^{-2}	10^{-3}	10^{-4}	10^{-5}
FlyingChairs	3.27	2.84	2.85	**2.83**
Sintel Final	(4.41)	(4.34)	**(4.16)**	(4.41)

(b) Separation weight

λ_f	2.0	3.0	4.0	5.0
FlyingChairs	2.95	2.90	**2.85**	3.21
Sintel Final	(4.33)	(4.31)	**(4.16)**	(4.36)

Table 4: Comparison to state-of-the-art deep unsupervised optical flow methods. Results in parentheses indicates it is evaluated using data it is trained on. We report average end-point-error for most categories and percentage of erroneous pixels for KITTI testset calculated from benchmark server. Best results in **red** and second best results in blue.

Method	Chairs	Sintel Clean		Sintel Final		KITTI 2015	
	test	train	test	train	test	train	test(Fl)
BackToBasic [19]	5.3	-	-	-	-	-	-
DSTFlow-ft [29]	5.52	(6.16)	10.41	(6.81)	11.27	16.79	39%
OccAwareFlow-best [34]	3.30	(4.03)	7.95	(5.95)	9.15	8.88	31.2%
UnFlow-CSS-ft [25]	-	-	-	(7.91)	10.22	8.10	23.30%
MultiFrameOccFlow-ft [18]	-	(3.89)	7.23	(5.52)	8.81	6.59	22.94%
DDFlow-ft [22]	2.97	(2.92)	6.18	(3.98)	7.40	5.72	14.29%
SelFlow-ft-Sintel [23]	-	(2.88)†	6.56†	(3.87)†	**6.57†**	**4.84**	14.19%
Ours-Chairs	**2.69**	3.66	-	4.67	-	16.99	-
Ours-ft-Sintel	3.01	**(2.86)**	**5.92**	**(3.57)**	6.92	12.75	-
Ours-ft-KITTI	4.32	5.49	-	7.24	-	5.19	**13.38%**

†: pretrained on the original Sintel movie

separation loss. After 20k steps of training with our method, the distribution (Fig. 5b) becomes similar to ground-truth distribution (Fig. 5c). Figure 5d-5e plots average difference, i.e., $(\text{sim}_f - k)$, of each types of pixels, where we observe the feature similarity becomes more discriminative gradually; average similarities of non-occluded pixels and occluded pixels become higher and lower respectively. The last result using GT flows shows solely on the feature factor (encoder) without flow estimation factor (decoder), which confirms the updated, more discriminative features.

Qualitative results. Training with our feature similarity loss makes the flow and similarity much discriminative. During training, the similarity map changes in a way that higher similarity becomes higher and lower similarity goes lower. As a result, the network can improve its prediction by reinforcing clear matching and suppressing uncertain matching. In Figure 6, we visualize how the similarity loss can be beneficial to flow learning. In the examples, our loss effectively improves the flow estimation by pushing similarity to each polar; that is why our

method performs well in uncertain regions. In Figure 7, we compare ours with SelFlow [23]. In the first two examples, the uncertain snow regions are the most challenging part, so that the state-of-the-art SelFlow fails in such regions. On the other hand, our method is able to handle such regions effectively, since the similarity loss suppresses the flows in that regions.

Quantitative comparison to state-of-the-art. We compare our method with existing deep unsupervised methods in Table 4. Our method effectively improves the baseline framework [22] and shows competitive results against other unsupervised methods. For Sintel, SelFlow gets a better result in the Sintel Final testset; it is trained on 10 k additional Sintel movie frames, while our model uses 1 k frames from the MPI Sintel dataset. Since our method can be used jointly with hallucinated occlusion and multiple-frame schemes from SelFlow [23], we expect a much stronger unsupervised model if the two are combined. In the real dataset KITTI, our method effectively improves over our baseline model (DDFlow), and reduces the percentage of erroneous pixels to 13.38% in the test benchmark. Overall, our approach achieves top-1 or top-2 consistently across different benchmarks. This demonstrates the robustness of our approach and benefits of utilizing deep self-supervised features and fused similarity.

Failure cases. Most unsupervised methods tend to estimate motion in a larger area than it really is, and it occurs more frequently for smaller and faster objects. In contrast, since our method estimates the similarity of a matching and refines it with the similarity, it sometimes reduces flows for small and fast-moving objects. In the last example in Figure 7, ours fails to catch the movement of few birds flying fast over the stairs.

5 Conclusion

In this paper, we have shown that learning flow from self-supervised features is feasible and effective with our fused similarity and feature separation loss. This allows the network to better learn flows and features from a sequence of images in the unsupervised way. We observe that, using the feature separation loss, the flows are updated to make the fused similarity more discriminative, while suppressing uncertain flows and reinforcing clear flows. The experiments show that the proposed method achieves competitive results in both qualitative and quantitative evaluation. The promising results confirm that, without labels, the self-supervised features can be used to improve itself. This kind of self-regulation techniques would be proven more effective under semi-supervised settings, where part of labels are available.

Acknowledgment

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT (NRF-2017M3C4A7066317).

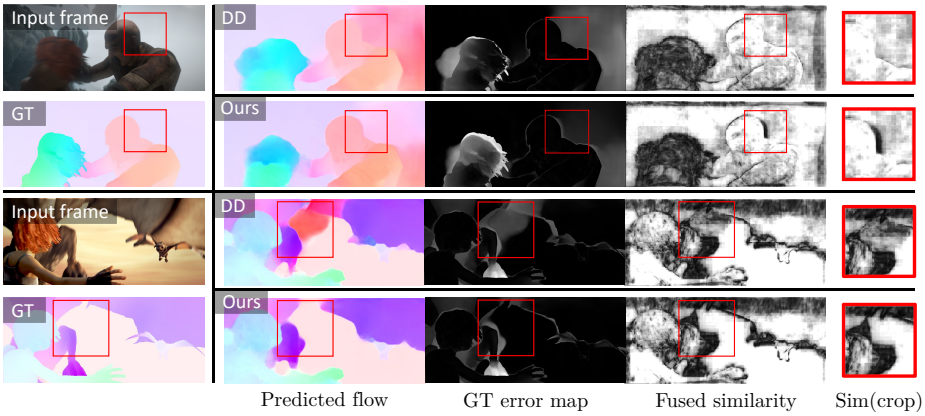


Fig. 6: Fused similarity of models trained w/o and w/ the feature similarity loss, denoted by **DD** and **ours**, respectively. The similarity loss suppresses similarity of the background snow texture behind the fighting man, resulting in the similarity map in the second row. In the second example, the sky region is expanded since the simlilarity increases by the similarity loss. As a result, the flow field effectively separates the brown wing of the dragon and the sky in brown

Fig. 7: Comparison to SelFlow [23] on the Sintel testset. We retrieve the resulting images and the visualizations of ground-truth from its benchmark website [1]; it provides only twelve test samples for each method

References

1. Mpi sintel dataset. http://sintel.is.tue.mpg.de/
2. Bailer, C., Taetz, B., Stricker, D.: Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In: Proceedings of the IEEE international conference on computer vision. pp. 4015–4023 (2015)
3. Bailer, C., Varanasi, K., Stricker, D.: Cnn-based patch matching for optical flow with thresholded hinge embedding loss. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
4. Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. International Journal of Computer Vision **92**(1), 1–31 (2011)
5. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: European conference on computer vision. pp. 850–865. Springer (2016)
6. 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)
7. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
8. Dong, X., Shen, J.: Triplet loss in siamese network for object tracking. In: The European Conference on Computer Vision (ECCV) (September 2018)
9. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2758–2766 (2015)
10. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3354–3361. IEEE (2012)
11. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=S1v4N2l0-
12. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: Advances in neural information processing systems. pp. 529–536 (2005)
13. Güney, F., Geiger, A.: Deep discrete flow. In: Asian Conference on Computer Vision. pp. 207–224. Springer (2016)
14. Han, X., Hu, X., Huang, W., Scott, M.R.: Clothflow: A flow-based model for clothed person generation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 10471–10480 (2019)
15. Hur, J., Roth, S.: Iterative residual refinement for joint optical flow and occlusion estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5754–5763 (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: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2462–2470 (2017)
17. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. In: Advances in neural information processing systems. pp. 2017–2025 (2015)

18. Janai, J., Guney, F., Ranjan, A., Black, M., Geiger, A.: Unsupervised learning of multi-frame optical flow with occlusions. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 690–706 (2018)
19. Jason, J.Y., Harley, A.W., Derpanis, K.G.: Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In: European Conference on Computer Vision. pp. 3–10. Springer (2016)
20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
21. Liu, C., Yu, G., Volkovs, M., Chang, C., Rai, H., Ma, J., Gorti, S.K.: Guided similarity separation for image retrieval. In: Advances in Neural Information Processing Systems. pp. 1554–1564 (2019)
22. Liu, P., King, I., Lyu, M.R., Xu, J.: Ddflow: Learning optical flow with unlabeled data distillation. Proceedings of the AAAI Conference on Artificial Intelligence **33**(01), 8770–8777 (Jul 2019). https://doi.org/10.1609/aaai.v33i01.33018770, https://aaai.org/ojs/index.php/AAAI/article/view/4902
23. Liu, P., Lyu, M., King, I., Xu, J.: Selflow: Self-supervised learning of optical flow. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4571–4580 (2019)
24. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4040–4048 (2016)
25. Meister, S., Hur, J., Roth, S.: Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
26. Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3061–3070 (2015)
27. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: European Conference on Computer Vision. pp. 69–84. Springer (2016)
28. Ranjan, A., Black, M.J.: Optical flow estimation using a spatial pyramid network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4161–4170 (2017)
29. Ren, Z., Yan, J., Ni, B., Liu, B., Yang, X., Zha, H.: Unsupervised deep learning for optical flow estimation. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
30. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: Epicflow: Edge-preserving interpolation of correspondences for optical flow. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1164–1172 (2015)
31. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems. pp. 568–576 (2014)
32. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8934–8943 (2018)
33. Ufer, N., Ommer, B.: Deep semantic feature matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6914–6923 (2017)
34. Wang, Y., Yang, Y., Yang, Z., Zhao, L., Wang, P., Xu, W.: Occlusion aware unsupervised learning of optical flow. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4884–4893 (2018)

35. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: Deepflow: Large displacement optical flow with deep matching. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1385–1392 (2013)
36. Werlberger, M., Pock, T., Unger, M., Bischof, H.: Optical flow guided tv-l 1 video interpolation and restoration. In: International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition. pp. 273–286. Springer (2011)
37. Xu, J., Ranftl, R., Koltun, V.: Accurate optical flow via direct cost volume processing. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
38. Xu, R., Li, X., Zhou, B., Loy, C.C.: Deep flow-guided video inpainting. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
39. Zabih, R., Woodfill, J.: Non-parametric local transforms for computing visual correspondence. In: European conference on computer vision. pp. 151–158. Springer (1994)
40. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision. pp. 818–833. Springer (2014)
41. Zhan, H., Garg, R., Saroj Weerasekera, C., Li, K., Agarwal, H., Reid, I.: Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 340–349 (2018)
42. Zhang, Z., Peng, H.: Deeper and wider siamese networks for real-time visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4591–4600 (2019)