Normalized Convolution Upsampling for Refined Optical Flow Estimation

Abdelrahman Eldesokey[a] and Michael Felsberg[b]
Computer Vision Laboratory, Linköping University, Sweden

Keywords: Optical Flow Estimation CNNs, Joint Image Upsampling, Normalized Convolution, Spare CNNS.

Abstract: Optical flow is a regression task where convolutional neural networks (CNNs) have led to major breakthroughs. However, this comes at major computational demands due to the use of cost-volumes and pyramidal representations. This was mitigated by producing flow predictions at quarter the resolution, which are upsampled using bilinear interpolation during test time. Consequently, fine details are usually lost and post-processing is needed to restore them. We propose the Normalized Convolution UPsampler (NCUP), an efficient joint upsampling approach to produce the full-resolution flow during the training of optical flow CNNs. Our proposed approach formulates the upsampling task as a sparse problem and employs the normalized convolutional neural networks to solve it. We evaluate our upsampler against existing joint upsampling approaches when trained end-to-end with a a coarse-to-fine optical flow CNN (PWCNet) and we show that it outperforms all other approaches on the FlyingChairs dataset while having at least one order fewer parameters. Moreover, we test our upsampler with a recurrent optical flow CNN (RAFT) and we achieve state-of-the-art results on Sintel benchmark with $\sim 6\%$ error reduction, and on-par on the KITTI dataset, while having 7.5% fewer parameters (see Figure 1). Finally, our upsampler shows better generalization capabilities than RAFT when trained and evaluated on different datasets.

1 INTRODUCTION

Computer vision encompasses a broad range of regression tasks where the goal is to produce numerical output given a visual input. Some of these tasks such as depth prediction and optical flow even require pixel-wise output, which makes theses tasks more challenging. Convolutional neural networks (CNNs) have lead to major breakthroughs in these regression tasks by exploiting deep representations of data. A common design for these regression CNNs is coarse-to-fine where a low-resolution prediction is produced and then progressively upsampled and refined to the full-resolution. This usually requires abundant GPU memory, especially at finer stages as the spatial dimensionality grows. Therefore, the scale of these networks has been throttled by the availability of computational resources, which has been mostly mitigated either by limiting the depth of the networks or reducing the resolution of the data.

As an example, the early work on CNN-based depth estimation in (Eigen et al., 2014) employed an encoder/decoder network where the training datasets were downsampled to half the resolution to fit into the available GPU memory. Similarly, the prevalent optical flow estimation network, FlowNet (Fischer et al., 2015), trains on a quarter of the full resolution and uses bilinear interpolation to restore the full-resolution during test time. This practice has been preserved in subsequent optical flow CNNs, particularly with the increased complexity of these networks and the emergence of the computationally expensive cost-volumes and pyramidal representations

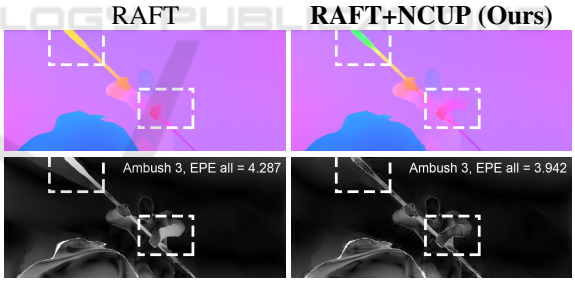


Figure 1: An example from the Sintel (Butler et al., 2012) test set that shows the flow improvement achieved by our proposed upsampler NCUP in comparison with RAFT (Teed and Deng, 2020).

[a] https://orcid.org/0000-0003-3292-7153
[b] https://orcid.org/0000-0002-6096-3648

Eldesokey, A. and Felsberg, M.
Normalized Convolution Upsampling for Refined Optical Flow Estimation.
DOI: 10.5220/0010343707420752
In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021) - Volume 5: VISAPP*, pages 742-752
ISBN: 978-989-758-488-6

(Fischer et al., 2015; Sun et al., 2018; Ilg et al., 2017). Nonetheless, pyramid levels with full and half the resolution were not utilized as they would not fit on the available GPU memory. Unfortunately, operating on a fraction of the full-resolution leads to loss of fine details, which might be crucial in certain tasks.

To alleviate these shortcomings of *coarse-to-fine* approaches, several joint image upsampling approaches have been applied as post-processing to the output from optical flow and depth estimation networks (Li et al., 2019; Su et al., 2019; Wu et al., 2018). These approaches substitute the bilinear interpolation and they utilize RGB images as guidance to perform adaptive upsampling for the predicted flow that preserves edges and fine details. The key idea of theses approaches is to use a guidance modality, *e.g.* . RGB images, to guide the upsampling of a target modality such as flow fields or depth values. However, these approaches act as post-processing and are trained separately from the network of the original task, omitting potential benefits from training them end-to-end. Therefore, we investigate training these joint upsampling approaches within the coarse-to-fine optical flow CNNs, *e.g.* . FlowNet, PWCNet, in an end-to-end fashion to allow optical flow networks to exploit the fine details during training. Moreover, we propose a novel joint upsampling approach (NCUP) that formulates the upsampling as a sparse problem and employs the normalized convolutional neural networks (Eldesokey et al., 2018; Eldesokey et al., 2019) to solve it. Our proposed upsampler that is more efficient (2k parameters) and outperforms other joint upsampling approaches in comparison on the task of end-to-end optical flow upsampling. An illustration for the proposed setup is shown in Figure 2a.

Another category of optical flow networks that emerged recently is based on *recurrent networks* (Hur and Roth, 2019; Teed and Deng, 2020), where the predicted flow is iteratively refined. This requires the availability of the flow in full-resolution at the end of each iteration. The bilinear interpolation was used for this purpose in (Hur and Roth, 2019), while a learnable convex combination upsampler was used in (Teed and Deng, 2020). However, this convex upsampler performs the upsampling with a scaling factor of 8 in a single-shot with a limited kernel support of 3×3. Moreover, it has a large number of parameters which encompasses approximately 10% of the entire network. We replace this convex combination module with our efficient upsampler that performs the upsampling at multi-scales, leading to state-of-the-art results on Sintel dataset (Butler et al., 2012), similar results on the KITTI dataset (Menze et al., 2018), better generalization capabilities, and using 5 times fewer parameters. Figure 2b shows an illustration for setup of recurrent networks, where we replace the upsampling module with our proposed upsampler.

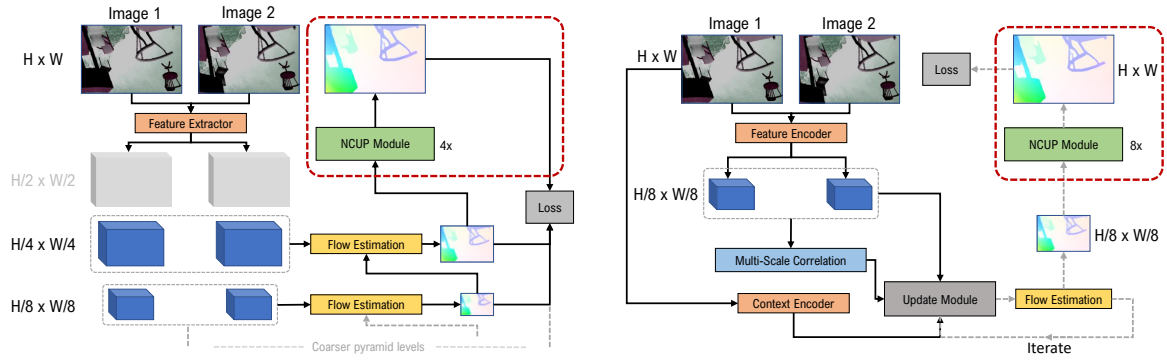
Our Contributions Can Be Summarized as Follows:

- We propose a joint upsampling approach (NCUP) that formulates upsampling as a sparse problem and employs the normalized convolution neural networks to solve it.
- We test our approach with *coarse-to-fine* optical flow networks (PWCNet) to produce the full-resolution flow during training, and we show that it outperforms all other upsampling approaches, while having at least one order fewer parameters.
- When we use our upsampler with a recurrent optical flow CNN, *e.g.* . RAFT (Teed and Deng, 2020), we achieve state-of-the-art results on the Sintel (Butler et al., 2012) benchmark, and perform similarly on the KITTI (Menze et al., 2018) test set using 5 times less parameters than their convex combination upsampler.
- We show that our upsampler has better generalization capabilities than the convex combination in RAFT, when trained on FlyingThings3D (Mayer et al., 2016) and evaluated on Sintel and KITTI.

2 RELATED WORK

CNN-based Optical Flow. Deep learning recently surfaced as a plausible substitute for the classical optimization-based optical flow approaches (Xu et al., 2017; Bailer et al., 2015; Horn and Schunck, 1981). CNNs can be trained to directly predict optical flow given two images avoiding explicitly designing an optimization objective manually in classical approaches. FlowNet (Fischer et al., 2015) introduced the first CNN for optical flow estimation that is trained end-to-end in a coarse-to-fine fashion. Subsequent approaches followed the same scheme where FlowNet2 (Ilg et al., 2017) proposed a stacked version of FlowNet, PWCNet (Sun et al., 2018) introduced a pyramidal variation, and LiteFlowNet (Hui et al., 2018) designed a light-weight cascaded network at each pyramid level. VCN (Yang and Ramanan, 2019) proposed several improvements for matching cost-volumes to expand their receptive field and they added support for multi-dimensional similarities.

Recently, several recurrent approaches were proposed where the flow is iteratively refined similar to the optimization-based approaches. An initial flow



(a) Coarse-to-fine (b) Recurrent

Figure 2: An illustration for how we train our proposed normalized convolution upsampler (NCUP) with coarse-to-fine and recurrent optical flow networks. In coarse-to-fine CNNs, *e.g.* . PWCNet (Sun et al., 2018) in **(a)**, the flow is estimated at different levels of a pyramid of features. However, pyramid levels with full and half the resolution are not utilized as it is not feasible to fit them in GPU memory. We upsample the flow to the full-resolution during training using our proposed approach leading to refined flow predictions. In recurrent CNNs, *e.g.* . RAFT (Teed and Deng, 2020) in **(b)**, the full-resolution flow needs to be available after each iteration. We replace the convex combination upsampler in RAFT, with our more compact upsampler NCUP and we achieve state-of-the-art results using fewer parameters.

prediction is produced at the first iteration and it is refined for a number of iterations. IRR (Hur and Roth, 2019) proposed to use either FlowNetS (Fischer et al., 2015) or PWCNet (Sun et al., 2018) as a recurrent unit that iteratively estimates the residual flow from the previous iteration. However, the number of iterations was limited either by the size of the network in FlowNet, or the number of pyramid levels in PWC-Net. RAFT (Teed and Deng, 2020) introduced a light-weight recurrent unit that is coupled with a GRU cell (Cho et al., 2014) as an update operator. This cell allowed performing more iterations and led to refined flow predictions at a relatively lower computations.

Joint Image Upsampling. The notion of joint (guided) image upsampling is to use a guidance image to steer the upsampling of another target image, where both the guidance and the target images could be from the same or different modalities. Several classical approaches were proposed that are based on variations of the bilateral filtering (Yang et al., 2007; **?**). Li *et al.* (Li et al., 2019) proposed a CNN-based architecture for joint image filtering that can be applied to joint upsampling. They employed two sub-networks for target and guidance features extraction followed by a fusion block. Wu *et al.* (Wu et al., 2018) proposed a trainable guided filtering network that was applied to clone the behavior of several vision tasks. Su *et al.* (Su et al., 2019) proposed pixel-adaptive convolutions that modifies the convolution filter with a spatially varying kernel. Wannenwetsch *et al.* (Wannenwetsch and Roth, 2020) extended the pixel-adaptive convolutions to incorporate pixel-wise confidences.

Optical Flow Upsampling. For coarse-to-fine networks, FlowNet (Fischer et al., 2015) suggested the use of an iterative variational approach (Brox and Malik, 2010) to produce the full-resolution flow during test time. However, this approach is computationally expensive and is not possible to train jointly with the network. For recurrent networks, the full-resolution flow is required during the training at the end of each iteration. IRR (Hur and Roth, 2019) attempted a residual upsampling block, but found to be futile with optical flow and they used the bilinear interpolation. RAFT (Teed and Deng, 2020) produces the flow in 1/8 of the full-resolution and employed a convex combination upsampler to construct the full-resolution. However, their upsampler has a limited receptive field and has a large number of parameters.

For coarse-to-fine networks, we look into employing differentiable joint upsampling approaches to upsample the flow during training. Moreover, we propose a joint upsampling approach (NCUP) that maps the upsampling task to a sparsity densificiation problem and employ the efficient normalized convolutional neural networks (Eldesokey et al., 2018; Eldesokey et al., 2019) to solve it. Experiments show that our upsampler performs better than other approaches in comparison on optical flow upsampling. Different to other joint upsampling approaches, our upsampler estimates the guidance on the low-resolution data instead of the full-resolution ones, which leads to fewer computations and memory requirements compared to other approaches.

For recurrent networks, *i.e.* . RAFT (Teed and Deng, 2020), we replace the convex module with our proposed upsampler, which performs the upsampling

at multi-scales and has 5 times fewer parameters. This modification leads to state-of-the-art results on Sintel (Butler et al., 2012) dataset with $\sim 6\%$ error reduction, similar performance on the KITTI (Menze et al., 2018) dataset, while using 7.5% fewer parameters. Finally, our approach shows better generalization capabilities when trained on FlyingThings (Mayer et al., 2016) and tested on Sintel and the KITTI datasets.

3 APPROACH

In joint image upsampling task, it is desired to train a network θ to upsample a low-resolution input $\mathbf{I}_{\text{LR}}$ to a high-resolution output $\mathbf{I}_{\text{HR}}$, guided by some high-resolution guidance data $\mathbf{g}_{\text{HR}}$; $\theta : \mathbf{I}_{\text{LR}} \rightarrow \mathbf{I}_{\text{HR}} | \mathbf{g}_{\text{HR}}$. The guidance data is typically the RGB image, but can be of any modality or even intermediate feature representations from a CNN. In this section, we briefly describe the normalized convolutional neural networks (Eldesokey et al., 2018) followed by our proposed Normalized Convolution Upsampler (NCUP).

3.1 Normalized Convolutional Neural Networks

Eldesokey *et al.* (Eldesokey et al., 2018; **?**) proposed the normalized convolution layer, which is a sparsity-aware convolution operator that was used to interpolate a sparse depth map on an irregular grid. More formally, they learn an interpolation function $\theta : \tilde{\mathbf{I}}_{\text{HR}} \rightarrow \mathbf{I}_{\text{HR}} | \tau(\tilde{\mathbf{I}}_{\text{HR}})$, where $\tilde{\mathbf{I}}_{\text{HR}}$ is a sparse high-resolution input with missing pixels, and $\tau()$ is a thresholding operator that produces ones at pixels where data is present and zeros otherwise. They recently proposed to replace the thresholding operator τ with a CNN Φ that predicts pixel-wise weights from the sparse input $\theta : \tilde{\mathbf{I}}_{\text{HR}} \rightarrow \mathbf{I}_{\text{HR}} | \Phi(\tilde{\mathbf{I}}_{\text{HR}})$ in a self-supervised manner (Eldesokey et al., 2020). The high-resolution output $\mathbf{I}_{\text{HR}}$ is predicted by a cascade of L normalized convolution layers, where the output for layer $l \in \{1 \dots L\}$, is calculated as:

$$\mathbf{I}^{l}_{\text{HR}}(\mathbf{x}) = \frac{\sum_{\mathbf{m} \in \mathbb{R}^2} \mathbf{I}^{l-1}_{\text{HR}}(\mathbf{x}-\mathbf{m})\, \mathbf{w}^{l-1}(\mathbf{x}-\mathbf{m})\, \mathbf{a}^{l}(\mathbf{m})}{\sum_{\mathbf{m} \in \mathbb{R}^2} \mathbf{w}^{l-1}(\mathbf{x}-\mathbf{m})\, \mathbf{a}^{l}(\mathbf{m})} \quad , \tag{1}$$

where $\mathbf{x}, \mathbf{m}$ are the spatial coordinates of the image, $\mathbf{I}^{0}_{\text{HR}} = \tilde{\mathbf{I}}_{\text{HR}}$, $\mathbf{w}^{0} = \Phi(\tilde{\mathbf{I}}_{\text{HR}})$, and $\mathbf{a}^{l}$ is the interpolation kernel at layer l. The weights are propagated between layers as:

$$\mathbf{w}^{l}(\mathbf{x}) = \frac{\sum_{\mathbf{m} \in \mathbb{R}^2} \mathbf{w}^{l-1}(\mathbf{x}-\mathbf{m})\, \mathbf{a}^{l}(\mathbf{m})}{\sum_{\mathbf{m} \in \mathbb{R}^2} \mathbf{a}^{l}(\mathbf{m})} \quad , \tag{2}$$

At the final layer L, the high-resolution output is produced $\mathbf{I}_{\text{HR}} = \mathbf{I}^{L}_{\text{HR}}$.

3.2 Formulating Upsampling as a Sparse Problem

Typically, the standard interpolation operations, *e.g.* . bilinear, bicubic, employ backward mapping to ensure that each pixel in the output is assigned a value. Contrarily, if forward mapping is used, a sparse grid is formed in the output. Fortunately, normalized convolution layers were demonstrated to perform well with irregular sparse grids, *e.g.* . depth completion, sparse optical flow, and, consequently, can be used to interpolate regular sparse grids.

Given a low-resolution input image $\mathbf{I}_{\text{LR}}$, a high-resolution sparse grid $\tilde{\mathbf{I}}_{\text{HR}}$ can be constructed using forward mapping. The forward mapping from the low-resolution grid coordinates (x', y') to the high-resolution grid (x, y) for a scaling factor s can be realized as:

$$(x, y) = (\text{round}(s \cdot x'), \text{round}(s \cdot y')) \; \forall \; (x', y') \tag{3}$$

Note that the high-resolution grid is regular when $s \in \mathbb{N}$.

The initial pixel-wise weights $\mathbf{w}^{0}$ required for the normalized convolution network can be estimated using a weights estimation network Φ similar to (Eldesokey et al., 2020). But different from (Eldesokey et al., 2020) and other existing joint upsampling approaches, we estimate the pixel-wise weights from the low-resolution guidance image, not the high-resolution one. Predicting weights for the low-resolution image requires less computations and memory requirements making the weights estimation network much smaller and shallower, and therefore, leading to a more efficient upsampling. For instance, the entire upsampling network that we use with coarse-to-fine optical flow networks, *e.g.* . FlowNet and PWCNet, has only 2k parameters (see Figure 3 where `ch1`=16 and `ch2`=8), while being able to outperform other approaches with at least one order of magnitude more parameters.

Another difference from (Eldesokey et al., 2020) is that we employ other modalities, *e.g.* . RGB input image, intermediate CNN features, as guidance for the weights estimation network similar to the existing joint upsampling approaches (Li et al., 2019; Su et al., 2019); $\Phi([\mathbf{I}_{\text{LR}}, \mathbf{g}_{\text{LR}}])$. This allows exploiting other modalities to adapt the weights based on the context. The output from the weights estimation network is also transformed to the high-resolution grid using the forward mapping.

Essentially, we train an upsampling network $\theta : \mathbf{I}_{\text{LR}} \rightarrow \mathbf{I}_{\text{HR}} | \Phi([\mathbf{I}_{\text{LR}}, \mathbf{g}_{\text{LR}}])$, where the sparse high-resolution grid $\tilde{\mathbf{I}}_{\text{HR}}$ is an intermediate stage generated by applying forward mapping to the the low-

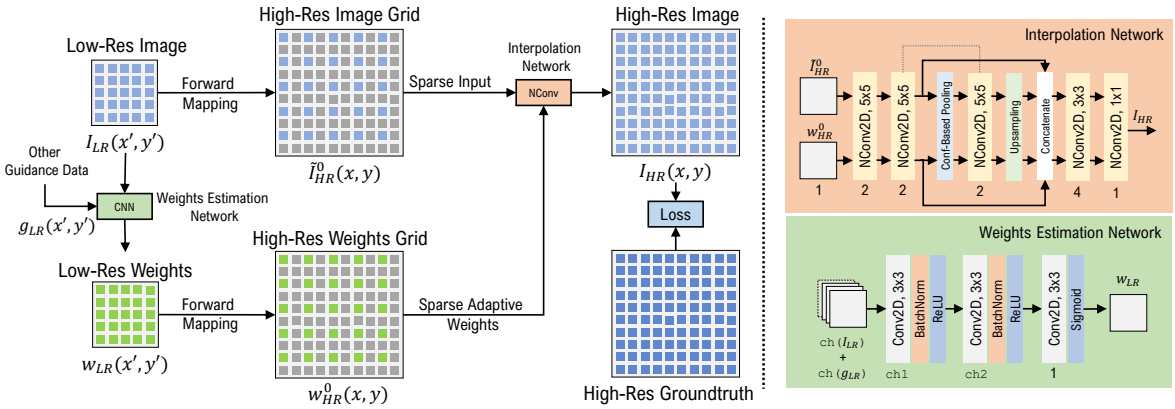


Figure 3: An illustration of our proposed joint upsampling approach (NCUP). First, a sparse high-resolution grid is constructed from the low-resolution image using forward mapping. Pixel-wise weights for the low-resolution image are produced by a *weights estimation network the* (green block) which takes the low-resolution image and other auxiliary data as input. The weights are mapped to the high-resolution grid in a similar fashion using forward mapping. Next, an *interpolation network* that encompasses a cascade of normalized convolution layers (the orange block) receives the high-resolution grid as well as the weights and produce the high-resolution image. Note that the notation `ch()` denotes number of channels.

resolution input. The pixel-wise weights are predicted using a CNN from the low-resolution input and any other guidance data. The weights are similarly mapped to the high-resolution grid using forward mapping. Finally, a cascade of normalized convolution layers is applied to interpolate the missing values in the sparse high-resolution grid. An illustration of the whole pipeline is shown in Figure 3.

3.3 Weights Estimation Network

Since the weights are estimated for the low-resolution input, the receptive field of the weights estimation network can be quite small. Therefore, we use two convolution layers with a 3×3 filters followed by Batch Normalization and ReLU activation. The number of channels per layer is determined based on the guidance data that is used. When RGB images are used, we use 16 and 8 channels for the two convolution layers, while we use 64 and 32 channels when intermediate CNN features are used as guidance (`ch1` and `ch2` values in Figure 3). A last convolution layer with a 1×1 filter is applied to produce the same number of channels as the low-resolution input $\mathbf{I}_{\text{LR}}$. Finally, a Sigmoid activation is applied to produce valid non-negative weights. Other function with a non-negative co-domain can be used, *e.g.* . Softplus, but the Sigmoid function was found to achieve the best results. The estimated weights are transformed to the high-resolution grid using forward mapping as well.

3.4 Interpolation Network

We build a U-Net shaped normalized convolution network inspired by (Eldesokey et al., 2018). However, we perform downsampling only once, *i.e.* . we use two scales instead of four in (Eldesokey et al., 2018), since the sparsity in our case is significantly lower than the LiDAR depth completion problem they were solving. This leads to a smaller network with 224 parameters instead of 480 parameters in (Eldesokey et al., 2018). The interpolation network receives the high-resolution image grid $\tilde{\mathbf{I}}_{\text{HR}}$ and the weights grid $\mathbf{w}^0$ as an input. The weights are propagated and updated within the interpolation network until the final dense output $\mathbf{I}_{\text{HR}}$ is produced at the final layer.

3.5 Optical Flow Upsampling

Optical flow is represented as two channels for vertical and horizontal flow field. We process the two channels jointly within the weights estimation network, *i.e.* . $\texttt{ch}(I_{\text{LR}}) = 2$ in Figure 3. However, for the interpolation network, the two channels are processed separately and then concatenated. In coarse-to-fine optical flow estimation networks, *e.g.* . FlowNet (Fischer et al., 2015) and PWCNet (Sun et al., 2018), the flow is produced at quarter the resolution. We attach the upsampling module to the optical flow estimation network to upsample the flow from $H/4 \times W/4$ to $H \times W$

Typically, the multi-scale loss is employed in coarse-to-fine networks:

$$\sum_{p \in P} \alpha_p \, |\mathbf{f}^p - \mathbf{f}^p_{\text{GT}}|^2 \quad , \tag{4}$$

where $\mathbf{f}^p$ is the flow estimation at pyramid level p in PWCNet or resolution p in FlowNet, where $P = \{3,4,5,6,7\}$, and $\mathbf{f}^p_{\text{GT}}$ is the corresponding downsampled groundtruth. The choice of α_p's were

Table 1: Summary of the results fpr two *coarse-to-fine* optical flow networks trained end-to-end with joint upsampling approaches. Relative Params. indicates the number of parameters for each upsampler. The stated results are the Average End-Point Error (AEPE) on the FlyingChairs (Fischer et al., 2015) test set. The relative improvement is shown between parentheses. The best results are shown in **Bold** and the second best in *Italics*. Our upsampler NCUP outperforms all other approaches DJIF (Li et al., 2019), PAC (Su et al., 2019), ConvComb (Teed and Deng, 2020) with PWCNet , while having the least number of parameters.

	Baseline	Bilinear	DJIF	PAC	ConvComb	**NCUP (Ours)**
PWCNet (Sun et al., 2018)	1.69	1.58 (+6.5%)	1.51 (+10.6%)	*1.50* (+11.2%)	1.52 (+10.0%)	**1.46** (+13.6%)
FlowNetS (Fischer et al., 2015)	2.53	2.23 (+11.8%)	2.16 (+14.6%)	**2.11** (+18.8%)	2.16 (+14.6%)	*2.13* (+15.8%)
Relative Params.	-	-	+56k	+183k	*+44k*	**+2k**

empirically determined in (Fischer et al., 2015) as $\{0.32, 0.08, 0.02, 0.01, 0.005\}$. Note that that $p = 1, p = 2$, where not considered during training as explained earlier. We consider another level/scale in the loss for the full-resolution flow, *i.e.* . we set $P = \{1, 3, 4, 5, 6, 7\}$, and following (Fischer et al., 2015), we found empirically that the best performance is obtained when $\alpha_1 = 0.02$ for most methods. This denotes that the flow is upsampled by a factor of 4 from quarter the resolution to the full-resolution. For the recurrent network RAFT, we use their proposed loss (Teed and Deng, 2020).

4 EXPERIMENTS

In this section, we evaluate our proposed joint upsampling approach with *two types* of optical flow estimation CNNs: coarse-to-fine and recurrent networks.

4.1 Joint Upsampling for Coarse-to-fine Networks

We choose two of the most popular coarse-to-fine optical flow CNNs, *i.e.* . FlowNet (Fischer et al., 2015) and PWCNet (Sun et al., 2018). Different joint upsampling approaches are attached to the two networks and they are trained end-to-end as illustrated in Figure 2a. The joint upsampling approaches that we compare against are DJIF (Li et al., 2019), PAC (Su et al., 2019), the convex combination from RAFT (Teed and Deng, 2020) which we refer to as ConvComb, and the bilinear interpolation. We train only on the FlyingChairs (Fischer et al., 2015) as its spatial resolution is smaller than its counterparts allowing training memory-demanding joint upsampling approaches. For instance, PWCNet trained with PAC fully occupy a 32 GB V100 GPU when trained of FlyingChairs with a batch size of 3. We use the official PyTorch implementations provided by the corresponding authors.

Experimental Setup. We initialize FlowNetS and PWCNet using pretrained models on the FlyingChairs dataset, while the joint upsampling approaches are initialized randomly. We train each network for 60 epochs with an initial learning rate of 0.0001 that is halved at epochs $\{20, 30, 40, 50, 55\}$. Since we can only fit a batch size of 3 for PAC on a 32GB V100 GPU, we use a batch size of 4 for all other approaches for a fair comparison. We use data augmentation as described in (Hur and Roth, 2019).

Quantitative Results. Table 1 summarizes the results for coarse-to-fine networks. All upsampling approaches lead to performance gains demonstrating the advantage from making the full-resolution flow available for coarse-to-fine networks during training. On PWCNet, our upsampler achieves the best improvement over the baseline despite having at least one order of magnitude lower parameters than its counterparts, while other approaches performs comparably well. On FlowNetS, our upsampler performs second best with a small margin to PAC. We believe that the larger model of PAC allows it to refine the poor predictions from FlowNetS slightly better than our upsampler.

Qualitative Results. A qualitative example for different approaches on the FlyingChairs dataset is shown if Figure 4. All upsampling approaches make edges and details more sharp and defined compared to the standard PWCNet as a result of making the full-resolution flow available during training. Nonetheless, PAC and our upsampler tend to produce sharpest results amongst all. However, our upsampler does a better job preserving small objects in some situations such as the red chair at the bottom of the scene.

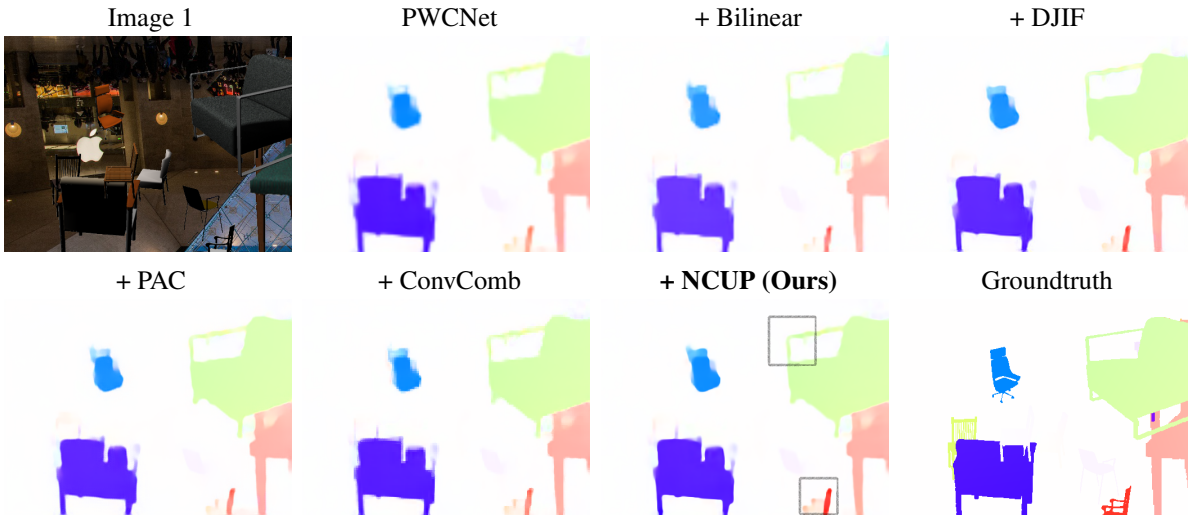


Figure 4: A qualitative example from the FlyingChairs (Fischer et al., 2015) dataset when PWCNet (Sun et al., 2018) is trained end-to-end using different joint upsampling approaches. Our upsampler produces sharp edges and preserves fine details such as the arm of the green chair and the small red chair at the bottom. Better viewed on a computer display.

4.2 Joint Upsampling for Recurrent Networks

We test our proposed upsampler as a substitute for the convex combination upsampler in the recurrent optical flow approach RAFT (Teed and Deng, 2020). The convex upsampler which has $\sim 500k$ parameters is removed and replaced with our upsampler constituting $\sim 100k$ parameters. We use the output from the GRU cell, which has 128 channels as guidance data as they suggested in addition to the low-resolution flow. For efficiency, we upsample the flow from 1/8 to 1/4 the full-resolution and then use our upsampler for restoring the full-resolution.

Experimental Setup. We initialize the network using the pretrained weights provided by the authors (Teed and Deng, 2020). We use the same training hyperparameters as described in (Teed and Deng, 2020) except for the weight decay that we set to 0.00005 and we only train for 50k iterations. For Sintel, we do not include FlyingThings3D and HD1k during finetuning. For KITTI, we disable the batch normalization in the weights estimation network as it leads to better results.

Benchmark Comparison. Table 2 shows the results for Sintel and KITTI benchmarks. On the Sintel benchmark, we outperform the standard RAFT with a 6.3% error reduction on the challenging final pass, while the error is slightly increased by 1.8% on the clean pass. We believe that this performance boost on the final pass is caused by multi-scale interpolation scheme employed by our upsampler that can eliminate large faulty regions in the predicted flow. On the KITTI benchmark, we perform similarly the standard RAFT despite having 7.5% fewer parameters.

Generalization Results. To examine the generalization capabilities of our upsampler, we train it on FlyingChairs followed by FlyingThings3D and evaluate it on the training set of Sintel and KITTI. Table 2 shows that our upsampler outperforms the standard RAFT on clean pass of Sintel and KITTI, while it performs slightly worse on the final pass of Sintel. We believe that the slight degradation on the final pass is due to training the clean and the final pass of FlyingThings3D together without a weighted sampling. However, the large improvement on KITTI significantly indicates that our upsampler posses better generalization.

Qualitative Results. Figure 5 shows some qualitative results from the Sintel test set. The use of our upsampler leads to better flow estimations compared to the standard RAFT. The first row shows an example where a large region of faulty flow prediction (the purple region under the dragon) produced by the standard RAFT that is corrected when our proposed upsampler was used. The second row shows another example where the flow is improved at fine details such as the hair. These results clearly demonstrates the impact of upsampling on the quality of the flow. Qualitative examples for the KIITI dataset can be found on the online benchmark: http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=flow .

Table 2: Summary for quantitative results when using our upsampler NCUP with the *recurrent* network RAFT (Teed and Deng, 2020). The best results are shown in **Bold** and the second best in *Italics*. Different datasets are indicated as following: FlyingChairs (Fischer et al., 2015) → C, FlyingThings3D (Mayer et al., 2016) → T, Sintel (Butler et al., 2012) → S, KITTI-Flow 2015 (Menze et al., 2018) → K, and HD1K (Kondermann et al., 2016) → H. Results between brackets are training set score and hence not comparable. Note that we did not use FlyingThings3D and HD1K during finetuning for Sintel. We outperform RAFT on the challenging final pass of Sintel and perform similarly on the test set of KITTI, while having 5 times smaller upsampler. * Indicates that warm starts (Teed and Deng, 2020) were used.

Training Dataset	Method	Sintel (Train)		KITTI (Train)		Sintel (Test)		KITTI
		Clean	*Final*	*AEPE*	*Fl-All*	*Clean*	*Final*	(Test)
C+T	PWCNet (Sun et al., 2018)	2.55	3.93	10.35	33.7	-	-	-
	LiteFlowNet (Hui et al., 2018)	2.48	4.04	10.39	28.5	-	-	-
	VCN (Yang and Ramanan, 2019)	2.21	3.67	8.36	25.1	-	-	-
	MaskFlowNet (Zhao et al., 2020)	2.25	3.61	-	23.1	-	-	-
	FlowNet2 (Ilg et al., 2017)	2.02	3.54	10.08	30.0	3.96	6.02	-
	RAFT-Small (Teed and Deng, 2020)	2.21	3.35	7.51	26.9	-	-	-
	RAFT (Teed and Deng, 2020)	*1.43*	**2.71**	*5.04*	**17.4**	-	-	-
	RAFT+NCUP	**1.41**	*2.75*	**4.83**	**17.4**	-	-	-
C+T+S+K+H	PWCNet+ (Sun et al., 2019)	(1.71)	(2.34)	(1.50)	(5.30)	3.45	4.60	7.27
	VCN (Yang and Ramanan, 2019)	(1.66)	(2.24)	(1.16)	(4.10)	2.81	4.40	6.30
	MaskFlowNet (Zhao et al., 2020)	-	-	-	-	2.52	4.17	6.10
	RAFT* (Teed and Deng, 2020)	(0.77)	(1.27)	(0.63)	(1.50)	**1.61**	*2.86*	**5.10**
	RAFT+NCUP*	(0.71)	(1.09)	(0.67)	(1.68)	*1.66*	**2.69**	*5.14*

4.3 Ablation Study

We conduct an ablation study to justify specific design choices in our proposed approach. Experiments are reported for PWCNet+NCUP on the FlyingChairs (Fischer et al., 2015) test set. Table 3 summarizes the average end-point-error scores for different experiments.

Weights Estimation Network. We replace the final activation with SoftPlus function instead of Sigmoid to get the estimate weights in the range of $[0, \infty[$ instead of $[0, 1]$ produced by the Sigmoid function. The network converges faster when using the SoftPlus function, however the AEPE score is slightly worse. We also attempt to feed the full-resolution guidance data to the weights estimation networks similar to other joint upsampling approaches. The kernel size of the first two convolution layers was increased to 5×5 for a larger receptive field. The results are significantly worse, which is probably because a larger network is needed to exploit the interesting information in the full-resolution data. Finally, we omit the low-resolution flow from being used with guidance data. The results shows that using the low-resolution flow with guidance data contributes significantly to the results.

Interpolation Network. We experiment with two downsamplings, which indicates that the interpolation is performed at three scales instead of two. The results show that the the best results are achieved when using only one downsampling. We also test the standard max pooling for downsampling instead of the confidence-based pooling proposed in (Eldesokey et al., 2018). The results show that the confidence-

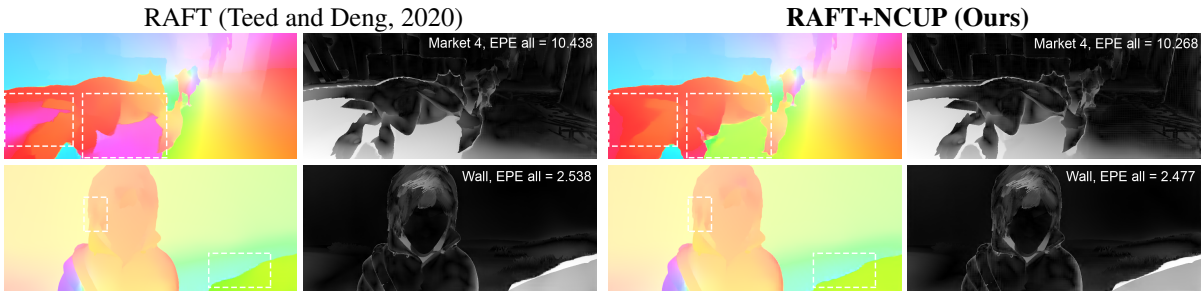


Figure 5: Qualitative examples from the Sintel (Butler et al., 2012) test set.

Table 3: Ablation results on the FlyingChairs (Fischer et al., 2015) test set. The baseline is PWCNet (Sun et al., 2018) trained with our upsampler NCUP.

Model	AEPE
PWCNet+NCUP (Baseline)	**1.46**
Weights Estimation Network	
Final activation is SoftPlus	1.48
Estimate from High-Res	1.75
Low-Res not used as guidance	1.52
Interpolation Network	
Two downsampling instead of one	1.49
Max instead of Conf. pooling	1.48
Loss Function	
$\alpha_1 = 0.002$	1.48
$\alpha_1 = 0.02$	**1.46**
$\alpha_1 = 0.2$	**1.46**

based pooling is slightly superior to max pooling.

The Loss Function. We experiment with one order of magnitude higher and lower factor α_1 in (4). The results indicates that the choice of $\alpha_1 = 0.02$ and $\alpha_1 = 0.2$ lead to the best results. So, we choose $\alpha_1 = 0.02$ since it works the best for the majority of methods in comparison, but the value of α_1 can be tuned further for our approach.

4.4 What Does Our Upsampler Learn?

Figure 6 shows an example of the predicted weights within our upsampler when used with RAFT on the Sintel dataset in comparison to the bilinear interpolation. The estimated weights essentially highlight edges and fine details with low-weight regions separating them. The width of these regions defines to what extent each object is extrapolated and ensures the separability between objects. Based on the design of the interpolation network, the width of these regions is adapted accordingly. On the other hand, solid regions, *e.g.* . the girl's face, with no texture are assigned uniform weights acting as averaging. This adaptive behavior shows a great potential for using our upsampling with other regression tasks, where the weights estimation network would learn the upsampling pattern that minimizes the reconstruction error.

Figure 6: An example of the predicted weights from NCUP when used with RAFT (Teed and Deng, 2020).

5 CONCLUSION

We introduced an efficient upsampling approach based on the normalized convolutional networks that we incorporated in training coarse-to-fine and recurrent optical flow CNNs. In coarse-to-fine networks, *e.g.* . PWCNet, the full-resolution flow was produced by our upsampler during the training leading to the fines flow estimations compared to other joint upsampling approaches in comparison, while having at least one order of magnitude fewer parameters. When trained with the recurrent optical flow network RAFT, it achieved state-of-the-art results on the Sintel dataset, and achieved a similar score on the KITTI dataset, while having 400k less parameters. Additionally, our approach showed better generalization capabilities compared to the standard RAFT.

ACKNOWLEDGEMENTS

This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) and Swedish Research Council grant 2018-04673.

REFERENCES

Bailer, C., Taetz, B., and Stricker, D. (2015). Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 4015–4023.

Brox, T. and Malik, J. (2010). Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE transactions on pattern analysis and machine intelligence*, 33(3):500–513.

Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. (2012). 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.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374.

Eldesokey, A., Felsberg, M., Holmquist, K., and Persson, M. (2020). Uncertainty-aware cnns for depth completion: Uncertainty from beginning to end. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12014–12023.

Eldesokey, A., Felsberg, M., and Khan, F. S. (2018). Propagating confidences through cnns for sparse data regression. In *The British Machine Vision Conference (BMVC), Northumbria University, Newcastle upon Tyne, England, UK, 3-6 September, 2018*.

Eldesokey, A., Felsberg, M., and Khan, F. S. (2019). Confidence propagation through cnns for guided sparse depth regression. *IEEE transactions on pattern analysis and machine intelligence*.

Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., Van der Smagt, P., Cremers, D., and Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. *arXiv preprint arXiv:1504.06852*.

Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics.

Hui, T.-W., Tang, X., and Change Loy, C. (2018). Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8981–8989.

Hur, J. and Roth, S. (2019). Iterative residual refinement for joint optical flow and occlusion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5754–5763.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470.

Kondermann, D., Nair, R., Honauer, K., Krispin, K., Andrulis, J., Brock, A., Gussefeld, B., Rahimimoghaddam, M., Hofmann, S., Brenner, C., et al. (2016). The hci benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 19–28.

Li, Y., Huang, J.-B., Ahuja, N., and Yang, M.-H. (2019). Joint image filtering with deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1909–1923.

Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:1512.02134.

Menze, M., Heipke, C., and Geiger, A. (2018). Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*.

Su, H., Jampani, V., Sun, D., Gallo, O., Learned-Miller, E., and Kautz, J. (2019). Pixel-adaptive convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11166–11175.

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. (2018). Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943.

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. (2019). Models matter, so does training: An empirical study of cnns for optical flow estimation. *IEEE transactions on pattern analysis and machine intelligence*, 42(6):1408–1423.

Teed, Z. and Deng, J. (2020). Raft: Recurrent all-pairs field transforms for optical flow. *arXiv preprint arXiv:2003.12039*.

Wannenwetsch, A. S. and Roth, S. (2020). Probabilistic pixel-adaptive refinement networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, H., Zheng, S., Zhang, J., and Huang, K. (2018). Fast end-to-end trainable guided filter. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1838–1847.

Xu, J., Ranftl, R., and Koltun, V. (2017). Accurate optical flow via direct cost volume processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1289–1297.

Yang, G. and Ramanan, D. (2019). Volumetric correspondence networks for optical flow. In *Advances in neural information processing systems*, pages 794–805.

Yang, Q., Yang, R., Davis, J., and Nistér, D. (2007). Spatial-depth super resolution for range images. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Zhao, S., Sheng, Y., Dong, Y., Chang, E. I., Xu, Y., et al. (2020). Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6278–6287.