IM-Net for High Resolution Video Frame Interpolation

Tomer Peleg Pablo Szekely Doron Sabo Omry Sendik
Samsung Israel R&D Center
{tomer.peleg,pablo.sz,doron.sabo,omry.sendik}@samsung.com

Abstract

Video frame interpolation is a long-studied problem in the video processing field. Recently, deep learning approaches have been applied to this problem, showing impressive results on low-resolution benchmarks. However, these methods do not scale-up favorably to high resolutions. Specifically, when the motion exceeds a typical number of pixels, their interpolation quality is degraded. Moreover, their run time renders them impractical for real-time applications. In this paper we propose IM-Net: an interpolated motion neural network. We use an economic structured architecture and end-to-end training with multi-scale tailored losses. In particular, we formulate interpolated motion estimation as classification rather than regression. IM-Net outperforms previous methods by more than 1.3dB (PSNR) on a high resolution version of the recently introduced Vimeo triplet dataset. Moreover, the network runs in less than 33msec on a single GPU for HD resolution.

1. Introduction

In video frame interpolation (VFI) one synthesizes middle non-existing frames from the original input frames. This is a well-known problem in the field of video processing. A classical application requiring VFI is frame rate up-conversion [3, 4, 12, 14, 16] for handling issues like display motion blur and judder in LED/LCD displays. Other applications include frame recovery in video coding and streaming [10, 11], slow motion effects [13] and novel view synthesis [7, 26].

Conventional approaches to VFI typically consist of the following steps: bi-directional motion estimation (ME), motion interpolation (MI) and occlusion reasoning, and motion-compensated frame interpolation (MC-FI). Such approaches are prone to various artifacts, such as halos, ghosts and break-ups due to insufficient quality of any of the components mentioned above.

In the past few years deep learning and specifically convolutional neural networks (CNNs) have emerged as the leading method for numerous image processing and computer vision tasks. Many computer vision tasks, such as image classification, object detection, and semantic segmentation, require accurate and exhaustive labeling. VFI however can be readily learned by *simply watching videos* [18]. Straightforward sub-sampling of videos can provide frame triplets, in which every middle frame can serve as ground truth for interpolation given the two other input frames.

The self-supervised nature of VFI makes it appealing for deep learning approaches. Indeed, a long series of works [13, 17–20, 22–24, 27, 31] have attempted to replace all or some of the steps in the VFI's algorithmic flow with CNNs.

Figure 1. Example results from our high resolution in-house clips (best viewed in color), from top to bottom: previous input frame, current input frame, middle frame generated by TOFlow [31], SepConv [24] and IM-Net.

978-1-7281-3293-8/19/$31.00 ©2019 IEEE
DOI 10.1109/CVPR.2019.00250

Despite the significant progress achieved by recent CNN-based methods for VFI, existing approaches are still limited in their performance. They do not handle well strong motions and wide occlusions, and are far from meeting real-time processing requirements for standard high resolutions such as HD and FHD. In Fig. 1 we show two examples for failure cases of two recent CNN-based approaches on high resolution frames with strong motions. These methods suffer from severe break-up, ghost and halo artifacts around the moving ball and persons.

IM-Net proposed in this paper aims at closing this performance gap. It can handle strong motions and wide occlusions in high resolution and runs in less than 33msec on a single GPU for HD resolution. Fig. 1 demonstrates our superiority over previous CNN-based methods in this type of scenes. We can see that artifacts observed in previous methods are much reduced in IM-Net: the shape of the ball is clear, the legs are not broken and the faces show no ghosts.

2. Contributions

This work presents IM-Net, a solution for video frame interpolation. It focuses on an important and challenging setup which remains unsolved up to date: real-time temporal interpolation of high resolution videos consisting of strong motions. The contribution of IM-Net is three-fold:

1. It is a deep CNN with a large receptive field that explicitly covers **strong motions** and is well suited for high resolutions.
2. This is an efficient solution for VFI – the CNN is deep, yet achieves **real-time** performance, and the middle frame is synthesized by a simple FI module.
3. IM-Net is trained using a **multi-scale loss** that combines both separable adaptive convolution and trilinear interpolation terms.

3. Related Work

CNNs have been successfully applied for numerous image processing tasks, such as image deconvolution [30] and single image super-resolution (SR) [5,6,29]. In these works the last convolutional layer directly produces the pixels of the output image. Inspired by the success of these CNNs, the early works on CNN-based VFI [18] and video frame prediction [19] attempted to adopt a similar approach. However, this typically led to blurred outputs and unsatisfactory image quality.

To overcome the weakness of these initial attempts, later approaches suggested more structured neural networks. In the AdaConv [23] and SepConv [24] methods, instead of directly producing the output pixels, their CNNs estimate *adaptive* filters for every pair of corresponding patches in the consecutive input frames. These output filters are then applied on the paired patches in both frames to produce the interpolated middle frame. SepConv outperformed all previous CNN-based methods at the time of publication. However, it is important to note that this method is limited to motions up to 51 pixels between consecutive input frames, and thus cannot cope with strong motions and occlusions. Furthermore, it requires high computational cost, for example when applied on FHD resolution, SepConv will estimate $0.4G$ filter weights (204 weights per output pixel).

Another direction revisited the classical VFI algorithmic flow and focused on replacing some of its steps by one or more CNNs. Deep Voxel Flow [17] and van Amersfoort et al. [27] focused on replacing all classical steps aside from FI with a single CNN. Here the network receives as input a pair of consecutive frames and outputs estimations for the interpolated motion vector field (IMVF) and occlusion map.

The TOFlow method [31] utilized three sub-networks: one for estimating the motion of each input frame with respect to the middle frame, a second for occlusion reasoning, and a third for frame synthesis given warped frames and occlusion masks. The main contribution of this work was demonstrating that each video processing task requires a different optical flow.

In Super Slomo [13] a CNN is used for bi-directional ME, then a simplified MI method is applied, and finally a second CNN performs ME refinement and occlusion reasoning. This work achieved overwhelming quality when applied on videos taken at high frame-rates. However, it seems that they do not aim at covering a wide range of motions.

Contex-Aware Synthesis (CtxSyn) [22] also utilizes a CNN for bi-directional ME, which is followed by classical MI and occlusion reasoning. Their main focus is on a second CNN for frame synthesis, which is based on a GridNet architecture [8]. This allowed them to replace the standard weighted blending scheme by a learned and locally adaptive synthesis method. Their algorithm outperformed SepConv for complex scenes. Another advantage of both Super Slomo and CtxSyn is their ability to produce as many of intermediate frames as one desires.

Finally, two recent works suggested utilizing a per-pixel phase-based motion representation for VFI. Phasenet [20] incorporated such a representation within a CNN-based approach. This allows them to handle a wider range of motion, compared to a classical phase-based method [21]. Their main advantage is the ability to better cope with inherent matching ambiguities in challenging scenes containing illumination changes and motion blur. However, Phasenet is inferior to SepConv in terms of the level of detail.

4. Method

In our work we propose a fully convolutional neural network for estimating the IMVF and occlusion map. Unlike

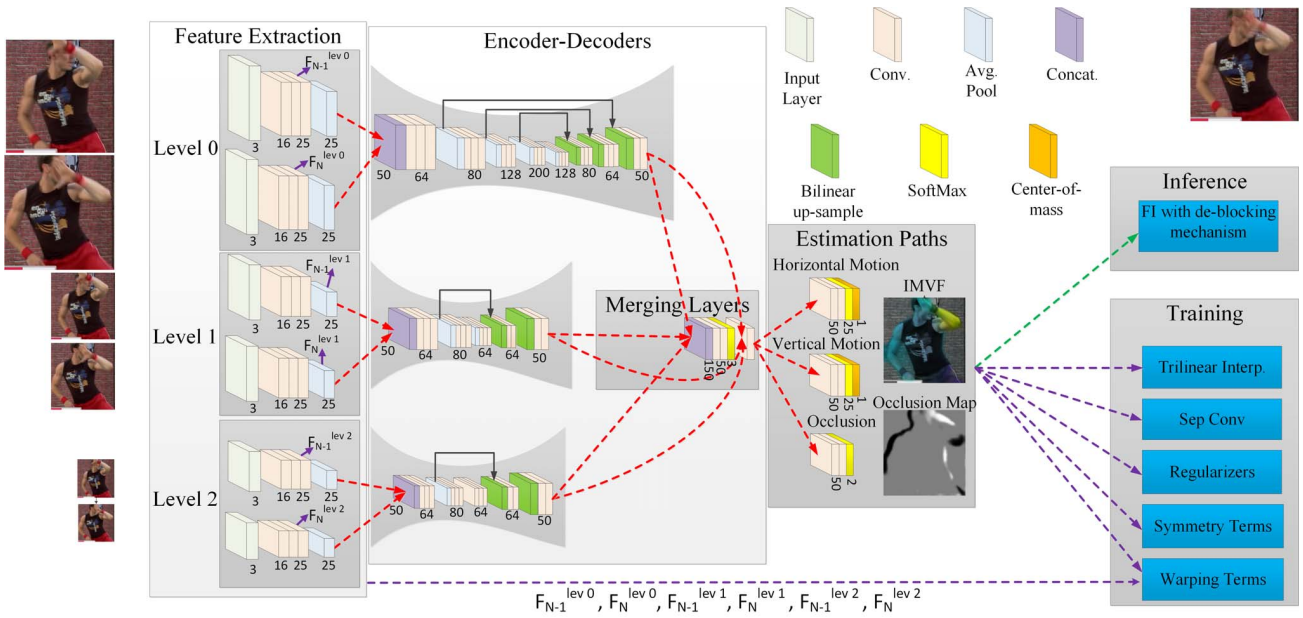

Figure 2. Overview of the IM-Net (best viewed in color). Left – pairs of input frames at three resolutions are inserted to the network. Middle – the CNN architecture. Right – the Inference and Training paths. ReLU activation is applied after every Conv layer which is not followed by SoftMax. The IMVF estimated by the network is overlaid on the interpolated frame.

previous works [17,27] that obtain pixel-wise estimates, we aim at block-wise versions. This is reasonable for high resolutions thanks to the piecewise smooth nature of motion. The estimated IMVF and occlusion map are then passed, along with the input frames, to a classical FI method that synthesizes the interpolated middle frame.

A widely used choice of architecture in the VFI domain is the encoder-decoder module [13,17,20,24]. IM-Net also uses such a module, but only as a basic processing building block.

In this section we will describe in detail our hand-tailored architecture which includes non-conventional layers. We further explain how the training loss is built upon this choice of architecture and how the contributions are manifested.

4.1. Network Architecture

The network's architecture (see Fig. 2) is composed of three types of modules — *Feature Extraction*, *Encoder-Decoder* and *Estimation*. The *Encoder-Decoder* sub-networks receive features extracted from the pair of consecutive input frames. Their outputs are merged into a high-dimensional representation which is passed on to the *Estimation* sub-network.

To benefit from a multi-scale processing of the pair of previous and current input frames, we constructed a three level pyramidal representation of the input frames. This means that each frame is passed to the CNN at three different scales.

Each of the six input frames (a pair per each pyramid level) is processed by the *Feature Extraction* module, yielding 25 feature channels per input. Since all inputs go through the same layers and these layers share their parameters, we refer to them as *Siamese*.

The extracted features from each pyramid level are passed as inputs to its *Encoder-Decoder* module. We design each *Encoder-Decoder* module with a slightly different architecture[1] so that all decoder outputs are of size $W/8 \times H/8 \times 50$.

Next, the three decoder outputs are merged using locally (per-pixel) adaptive (learned) weights. To produce these weights the decoder outputs are passed to a cascade of Conv layers, followed by a SoftMax layer. The merged output is then computed as a channel-wise weighted average of the three decoder outputs.

Finally, the merged output is sent to three parallel *Estimation* paths, each consisting of Conv layers and ending with a SoftMax layer. The first two paths generate 25 normalized weights each (in a $W/8 \times H/8$ resolution). These pairs of weights are associated with estimation of the horizontal and vertical components of the IMVF, respectively. The third path generates two normalized weights (in a $W/8 \times H/8$ resolution), which are associated with estimation of the occlusion map.

This architecture results in a computationally light-weight CNN with a large receptive field. This is due to

[1]The parameters of corresponding Conv layers in the three encoders are shared, whereas each decoder has its own set of parameters.

Symbol	Definition	Resolution (Training)
$W \times H$	Full image resolution	512×512
$\mathbf{I}_p, \mathbf{I}_c, \mathbf{I}_m$	Previous/current/middle frame (full resolution)	$512 \times 512 \times 3$
$\mathbf{I}_p^{DS}, \mathbf{I}_c^{DS}, \mathbf{I}_m^{DS}$	Previous/current/middle frame downscaled by factor 8	$64 \times 64 \times 3$
$\mathbf{T}$	The estimated occlusion map	64×64
$\mathbf{W}_X, \mathbf{W}_Y$	Output of the horizontal/vertical motion estimation path	$64 \times 64 \times 25$
$\mathbf{S}_X, \mathbf{S}_Y$	The estimated horizontal/vertical component of the IMVF	64×64
$\mathbf{F}_k^{lev_0}$	Features extracted from level 0 of the image pyramid for $\mathbf{I}_k$	$256 \times 256 \times 25$
$\mathbf{F}_k^{lev_i}$	Features extracted from i^{th} level of the image pyramid for $\mathbf{I}_k$, for $i = 1, 2$	$64 \times 64 \times 25$
bilin	Bilinear interpolation over a 2×2 support around a given spatial location	
$\Phi(\mathbf{I}_1, \mathbf{I}_2)$	Average smoothed ℓ_1 metric between two images	
$TV(\cdot)$	Non-isotropic total variation	

Table 1. List of notations

the cost-aware choice of the number of channels at each layer, starting with a small number and increasing it by a small factor (less than 2) after each decrease in spatial resolution. This is in contrast to the common trend in previous work [17, 23, 24]. More details on the computational cost per each sub-network can be found in the supplementary material.

From this point on, we make extensive use of notations. Please refer to Table 1 for the full list.

4.2. Non-Conventional Estimation Layers

The outputs of the estimation paths are further processed for the middle frame synthesis, which requires two components – motion compensated warping (MCW) of the input frames and local blending weights (occlusion map).

The outputs of the horizontal and vertical estimation paths $\mathbf{W}_X$ and $\mathbf{W}_Y$ yield two options for MCW:
(i) *Separable adaptive filtering* — each set of 25 outputs can be utilized as a normalized one-dimensional filter operating on each input frame, downscaled by a factor of 8. The two filters are applied in a separable fashion, where for the previous frame we flip the order of the filter coefficients, yielding two versions of the downscaled middle frame:

$$\begin{aligned}\mathbf{I}_{p\to m}^{DS,SepC} &= \text{SepConv}\left(\mathbf{I}_p^{DS}, -\right)\\ \mathbf{I}_{c\to m}^{DS,SepC} &= \text{SepConv}\left(\mathbf{I}_c^{DS}, +\right)\end{aligned} \tag{1}$$

where

$$\begin{aligned}&\hat{\mathbf{I}}^{SepC}(x,y) = \text{SepConv}(\mathbf{I}, \pm) \doteq \\ &\sum_{v=-12}^{12} \mathbf{W}_Y(x,y,v) \sum_{u=-12}^{12} \mathbf{W}_X(x,y,u)\, \mathbf{I}(x \pm u, y \pm v).\end{aligned} \tag{2}$$

(ii) *Classification probabilities* – we assign each of the 25 classes with a motion component directed from the interpolated frame to the current frame[2]. To cover a large range of motions, we use a uniformly distributed set of values within the range $[-96, 96]$ pixels in full resolution[3], i.e.

$$\mathbf{W}_j(x,y,k) = \Pr\{\mathbf{S}_j(x,y) = 8k\}, \tag{3}$$

for $j \in \{X, Y\}$ and $k \in \{-12, \ldots, 12\}$. The class probabilities are transformed to values in the IMVF by a center-of-mass (expectation) calculation,

$$\mathbf{S}_j(x,y) = \sum_{u=-12}^{12} 8u \cdot \mathbf{W}_j(x,y,u). \tag{4}$$

The IMVF can be used for obtaining the warped full resolution frames by

$$\begin{aligned}\mathbf{I}_{p\to m}^{Warp} &= \text{Warp}(\mathbf{I}_p, -, 8)\\ \mathbf{I}_{c\to m}^{Warp} &= \text{Warp}(\mathbf{I}_c, +, 8)\end{aligned} \tag{5}$$

where

$$\begin{aligned}&\hat{\mathbf{I}}^{Warp}(x,y) = \text{Warp}(\mathbf{I}, \pm, L) \doteq \\ &\mathbf{I}^{bilin}\left(x \pm \frac{L}{8}\mathbf{S}_X\left(\left\lfloor\frac{x}{L}\right\rfloor, \left\lfloor\frac{y}{L}\right\rfloor\right), y \pm \frac{L}{8}\mathbf{S}_Y\left(\left\lfloor\frac{x}{L}\right\rfloor, \left\lfloor\frac{y}{L}\right\rfloor\right)\right).\end{aligned} \tag{6}$$

In Eq. (5), we assign each estimated motion vector to an 8×8 block in full resolution. In general, using Eq. (6) it can be assigned to an $L \times L$ block in resolution $W \cdot L/8 \times H \cdot L/8$.

The occlusion map serves as local weights for blending the warped input frames and obtaining the final output frame. This map is extracted as the first channel from the output of the occlusion estimation path. The map can get any value between 0 and 1, where 1 is interpreted as *closing*, 0 as *opening* and 0.5 as non-occluded (equal blending).

The low and full resolution versions of the interpolated frame are obtained by

$$\hat{\mathbf{I}}_m^{DS,SepC} = \mathbf{T} \cdot \mathbf{I}_{p\to m}^{DS,SepC} + (1 - \mathbf{T})\, \mathbf{I}_{c\to m}^{DS,SepC} \tag{7}$$

$$\hat{\mathbf{I}}_m^{trilin} = \mathbf{T}^{US\uparrow 8} \cdot \mathbf{I}_{p\to m}^{Warp} + \left(1 - \mathbf{T}^{US\uparrow 8}\right) \mathbf{I}_{c\to m}^{Warp} \tag{8}$$

[2] We assume linear motion between input frames, namely the motion from middle to previous equals minus the motion from middle to current.

[3] This design is flexible with respect to the range of motions on which the network spends its attention during training.

where

$$\mathbf{T}^{US\uparrow_L}(x,y) = \mathbf{T}(\lfloor x/L \rfloor, \lfloor y/L \rfloor). \quad (9)$$

Eq. (8) is essentially the trilinear FI suggested by [17]. In this equation we assigned each occlusion weight to an 8×8 block in the full resolution.

The separable adaptive filtering and trilinear FI operations are applied only during training. At inference time, we replace them by a more elaborate FI module (see Fig. 2). This module exploits a de-blocking mechanism which removes block artifacts from motion boundaries. First, it produces several versions of each output pixel by applying Eq. (8) using the block-wise estimates from neighboring blocks. Then it interpolates these versions according to the pixel location within the block.

4.3. Training Loss

We train the fully convolutional network in an end-to-end manner using only pairs of input frames, along with their middle frames as ground truth. The network's training loss is composed of five terms:

$$\begin{aligned}\textbf{Loss} =& \alpha_1 \Phi\left(\hat{\mathbf{I}}_m^{DS,SepC}, \mathbf{I}_m^{DS}\right) + \alpha_2 \Phi\left(\hat{\mathbf{I}}_m^{trilin}, \mathbf{I}_m\right) + \\ & \alpha_3 \cdot \text{Warp Terms} + \lambda \cdot \text{Regs} + \\ & \gamma \cdot \text{Symmetry Terms} \end{aligned} \quad (10)$$

In all of these terms we shall utilize the smoothed ℓ_1 metric (Φ) when comparing between a pair of image pixels or features. The two first terms are fidelity terms: one is associated with the frames downscaled by a factor of 8 and separable filtering, and the other with the full resolution frames and trilinear interpolation. These terms penalize the network for artifacts in the synthesized frame. However, the root cause for such artifacts is typically inaccuracies in the registration of the input features.

In order to explicitly encourage better alignment between pairs of input features, we added the warping terms. These terms measure the distance between the warped features from the previous and current frames. This is in contrast to [13] that utilizes a loss between the warped input frames and to [23, 24] that incorporate a loss between features of the interpolated and the ground truth middle frames.

More specifically, for each pyramid level we use pairs of features from a specific layer in the Siamese sub-network (see Fig. 2). We warp these features according to the estimated IMVF, as follows:

$$\begin{aligned}\mathbf{F}_{p\to m}^{lev_i,SepC} &= \text{SepConv}\left(\mathbf{F}_p^{lev_i}, -\right),\ i=1,2 \\ \mathbf{F}_{c\to m}^{lev_i,SepC} &= \text{SepConv}\left(\mathbf{F}_c^{lev_i}, +\right),\ i=1,2 \\ \mathbf{F}_{p\to m}^{lev_0,Warp} &= \text{Warp}\left(\mathbf{F}_p^{lev_0}, -, 4\right) \\ \mathbf{F}_{c\to m}^{lev_0,Warp} &= \text{Warp}\left(\mathbf{F}_c^{lev_0}, +, 4\right) \end{aligned} \quad (11)$$

Each warping loss term is computed as a conditioned mean over the absolute difference between pairs of warped input features. The condition is that both features are non-negligible and the spatial location does not belong to an occluded region. Let us denote the set of feature indices that satisfy this condition by

$$\begin{aligned}\Omega(\mathbf{F},\mathbf{G},\mathbf{T}) \doteq \{(x,y,c) \mid & \mathbf{F}(x,y,c) > \epsilon, \\ & \mathbf{G}(x,y,c) > \epsilon, \left|\mathbf{T}(x,y) - \frac{1}{2}\right| \le \frac{1}{4}\Big\} \end{aligned} \quad (12)$$

Each conditioned mean is calculated as

$$\kappa(\mathbf{F}_1,\mathbf{F}_2,\mathbf{T}) \doteq \Phi(\mathbf{F}_1,\mathbf{F}_2 \mid \Omega(\mathbf{F}_1,\mathbf{F}_2,\mathbf{T})). \quad (13)$$

Using Eq. (9), (11) and (13) we can formulate the warping terms as:

$$\begin{aligned}\text{Warp Terms} = & \sum_{i=1}^{2} \kappa\left(\mathbf{F}_{p\to m}^{lev_i,SepC}, \mathbf{F}_{c\to m}^{lev_i,SepC}, \mathbf{T}\right) + \\ & \alpha_4 \kappa\left(\mathbf{F}_{p\to m}^{lev_0,Warp}, \mathbf{F}_{c\to m}^{lev_0,Warp}, \mathbf{T}^{US\uparrow_4}\right) \end{aligned} \quad (14)$$

Next, we added regularizers for encouraging piece-wise smoothness in the estimated motion field. Specifically, we apply a non-isotropic total variation over the first moments of the horizontal and vertical motion distributions – $\mathbf{S}_X, \mathbf{S}_Y$, and their second moments:

$$\text{Regs} = TV(\mathbf{S}_X) + TV(\mathbf{R}_X) + TV(\mathbf{S}_Y) + TV(\mathbf{R}_Y), \quad (15)$$

where the second-moment is given by

$$\mathbf{R}_j(x,y) = \sqrt{\sum_{u=-12}^{12} \left[8u - \mathbf{S}_j(x,y)\right]^2 \cdot \mathbf{W}_j(x,y,u)}. \quad (16)$$

Finally, in the last term we encourage the CNN's estimates to be invariant to two symmetries: horizontal flipping and flipping the temporal order of the input frames. When applying these terms we include in each training batch both the original inputs and three combinations of horizontally and/or temporally flipped versions of these inputs.

4.4. Training Dataset

In order to create a large training dataset we started from numerous video clips at HD or FHD resolution, mostly retrieved from YouTube (with common creative license). The chosen video clips include sport events (for example: marathons, basketball and soccer games), scenes with strong hand movements (such as interviews and lectures), and footage with strong camera motion (taken by action-cameras or from a moving vehicle). These clips cover a broad range of lighting conditions and environments (i.e. both indoor/outdoor scenes), and most importantly, diverse types of motion and occlusion.

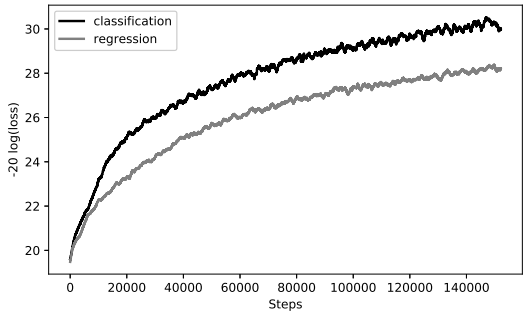

Figure 3. The full resolution fidelity loss term in logarithmic scale (after applying $-20 \cdot log_{10}(\cdot)$) versus the number of iterations for two types of motion estimation – our design choice and plain regression.

Each clip was decomposed to frame triplets, consisting of the previous frame, the current frame and the target middle frame that serves as ground truth. This set of triplets was filtered so that we kept only challenging and interesting scenes. We tested several filtering methods. A straightforward method which proved beneficial is checking that plain blending of the input frame pair substantially differs from the target frame.

From each high resolution frame we cropped a few interesting regions of 512×512 pixels, unlike the common trend of downscaling the input frames. This procedure resulted in approximately 40,000 frame triplets. For enriching this dataset we applied the following data augmentations: horizontal and temporal flipping, as well as adding a random global shift to the input frames.

4.5. Training Protocol

Because our architecture is custom made for this task, we chose to train the parameters of our CNN from scratch, namely we did not use any pre-trained model. We found it useful to apply a series of separate training phases, each with different loss terms or dataset. We started with real-life frames globally and synthetically shifted. The shifts were uniformly distributed in the range $[-192, 192]$ (between the previous and current frame). This ensures that the estimation paths will be adapted from an early stage of their training to a large motion range. Then, we switched to the real-life dataset described in the previous sub-section, gradually adding more loss terms and modifying their contribution by changing the relevant hyper-parameters. For example, we started with $\alpha_1 = 0.9$ and $\alpha_2 = 0.1$, and increased α_2 to 1.5, thus encouraging coarse-to-fine motion estimation.

For all training phases we utilized a batch size of 16 and the Adam optimizer [15]. We started with a learning rate of 10^{-4} and decreased it to $5 \cdot 10^{-5}$, as the training progressed. We did not apply batch normalization or dropout.

5. Results

5.1. Comparison to Regression

We first demonstrate that our choice of classification in the motion estimation paths (see Section 4.2) is more efficient than conventional regression. In this experiment, we trained from scratch the network twice: once with our design and once with a modified architecture. In the latter, we avoided using SoftMax layers in the motion estimation paths and center-of-mass computations. Instead, the last convolutional layer in each such path directly produces the values of the IMVF.

To isolate the effect of the motion estimation paths, we used a simplified training process, which allowed for better control for other loss terms. First, we chose to train the network with inputs that have a global shift, as described in Section 4.5. Second, we only kept the fidelity terms, with no other regularizers, warping nor symmetry terms. Finally, we disabled the occlusion estimation path and set all of the values in the occlusion map to 0.5. We trained both networks for $150,000$ iterations. Fig. 3 shows that under these conditions the classification design outperforms regression.

5.2. Testing Dataset

As aforementioned, our approach focuses on real-life scenarios where VFI is required at high resolution with presence of strong motions. Thus we searched for a dataset that meets these requirements.

When inspecting several popular benchmarks we found that many of them are not suitable for our purposes. Both KITTI [9] and Sintel [2] consist of frames at high resolution as we desire. However, the first suffers from low image quality and is limited to urban-traffic scenes, and the second is purely synthetic. Moreover, these benchmarks are typically used for assessing the quality of estimated optical flow between consecutive input frames rather than frame interpolation. Because this measure is not directly linked to our end-to-end training scenario, we did not test on them. UFC-101 [25], which was originally created for action recognition, consists of low resolution frames at low image quality and with mainly weak motions and occlusions (in number of pixels). Finally, while the Middlebury challenge [1] is very popular, it is a small (8 images) and limited benchmark with low resolution frames that does not meet our needs.

The Vimeo dataset, recently introduced by the authors of TOFlow [31], is much more suitable for our needs. It consists of 3,782 frame triplets extracted from real-life videoclips available at the Vimeo website [28]. This dataset has sufficient image quality while covering a large variety of light conditions, environments and motions. Nevertheless, the low resolution of its frames – 448×256, obtained by downscaling the original high resolution frames, is a weak point of this dataset. In contrast, IM-Net was trained with

IM-Net	PSNR	SSIM
w.o. warping and symmetry terms	32.07	0.9319
w.o. symmetry terms	32.94	0.9416
All terms inside	**33.11**	**0.9436**

Table 2. Ablation study – contribution of loss terms

patches of size 512×512 cropped from HD or FHD frames. To benefit from the strong points of this dataset, we created a high-resolution (1344×768) version of it by up-scaling each frame in the original Vimeo dataset using an off-the-shelf CNN-based single image SR method [32]. In our experiments we used both the original dataset and its SR version to conduct an ablation study and to compare IM-Net with prior art.

5.3. Ablation Study

As discussed in Section 4.5 as training progressed, we gradually added more types of loss terms. The symmetry terms are designed to confine the network into the physical world and to reduce biases. The warping terms are added to punish the network outside of occluded areas in the case that the interpolated features from the pair of input frames do not match. When this occurs, ghosts and halos may appear in the synthesized frame.

For the sake of ablation study we evaluated our method on the SR version of the Vimeo dataset, where we report the gain obtained by adding several groups of loss terms. Table 2 summarizes those contributions. We can see that the warping terms contribute $0.87dB$ and that imposing symmetries yields an additional gain of $0.17dB$.

5.4. Comparison to State-of-the-Art

Recent methods achieved impressive results on low resolution VFI benchmarks. For example, SepConv [24], TOFlow [31], Super Slomo [13] and CtxSyn [22] are all ranked at the top ten performers in the Middlebury benchmark. The three former methods also showed state-of-the-art performance on the UCF-101 dataset. By testing two leading methods: SepConv and TOFlow, both on low and high resolution versions of the same benchmark, we shall examine how well these algorithms scale up with resolution and motion-strength.

As described in Section 5.2, we conducted our comparisons using the Vimeo dataset. First, we tried applying IM-Net directly on the original Vimeo frames, despite the fact that there is not a good match between our training set and this test set. We obtained an average PSNR of $32.35dB$, which is $1.4dB$ lower than the best performance achieved by TOFlow. To improve the suitability of our trained network with this dataset, we re-ran our approach with simple pre and post-processing steps (more details are provided in the supplementary material). Next, we repeated the experiment with the SR version of Vimeo (see Section 5.2).

Table 3 summarizes the PSNR and SSIM of these experiments. At low resolution the three networks show comparable quality, but after increasing the resolution SepConv and TOFlow suffer from reductions of $1.6dB$ and $3.2dB$ respectively, whereas IM-Net decreases only by $0.3dB$ compared to the low resolution interpolation quality. We also tested IM-Net on several test videos at various high resolutions, ranging from HD to 4K. On these clips we observed similar performance gaps as in the Vimeo dataset. Results for two of the clips can be found in the supplementary material.

Fig. 4 shows results for five pairs of frames taken from the original Vimeo and its SR version. We can see that for the original low resolution frames, IM-Net is comparable to the two other methods (and all three are quite close to the ground truth). Once the resolution of the frames is increased, severe break-up, ghost and halo artifacts appear in SepConv and TOFlow, whereas for IM-Net, the results remain almost unharmed. More visual demonstrations can be found in the supplementary material.

Finally, we compared the running time of IM-Net with that of SepConv and TOFlow. For our method, we report the time required for running only the CNN. Other operations, such as preparing the input frames at three resolutions and running the FI module, require a negligible computational cost compared to that of the CNN. SepConv and TOFlow include in their neural networks the frame synthesis step and thus the reported times include it. Table 4 summarizes the running times on HD and FHD resolutions, measured on a single Nvidia Titan X GPU. We can see that IM-Net is faster by a factor of 16. This is due to its light-weight architecture and the use of block-wise (instead of pixel-wise) estimates for the IMVF and the occlusion map.

6. Conclusion

In this paper we presented IM-Net, a method for VFI. Our approach is composed of a CNN that outputs block-wise estimates for the IMVF and occlusion map, followed by a FI module that removes block artifacts on motion edges. By careful choice of the network's architecture and training loss, we were able to train a CNN that covers a wide range of motions, handles strong occlusions and at the same time can meet real-time requirements at inference time.

The frame interpolation quality of our approach can be further boosted by adding more CNNs on top of the existing one – a second CNN can refine our estimate for the IMVF and a third CNN can replace the simple FI module, as suggested in several recent works [22, 31].

Acknowledgments. We thank Michael Dinerstein for his hard work and dedication to this project. We are also grateful to Roy Jevnisek, Avital Steinitz and David Tsidkiahu for their insightful comments.

Method	SepConv - L_f [24]		TOFlow [31]		IM-Net	
Objective figure-of-merit	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
448×256 resolution	33.45	0.9509	**33.73**	**0.9515**	33.50	0.9473
1344×768 resolution	31.81	0.9309	30.54	0.9190	**33.11**	**0.9436**

Table 3. Comparison to state-of-the-art on the Vimeo dataset

Method	SepConv -L_f [24]	TOFlow [31]	IM-Net
Platform	Torch	Torch	Caffe
Run time (msec) for HD	500	460	**30**
Run time (msec) for FHD	900	880	**55**

Table 4. Run times (msec) of IM-Net and state-of-the-art methods for HD and FHD resolutions

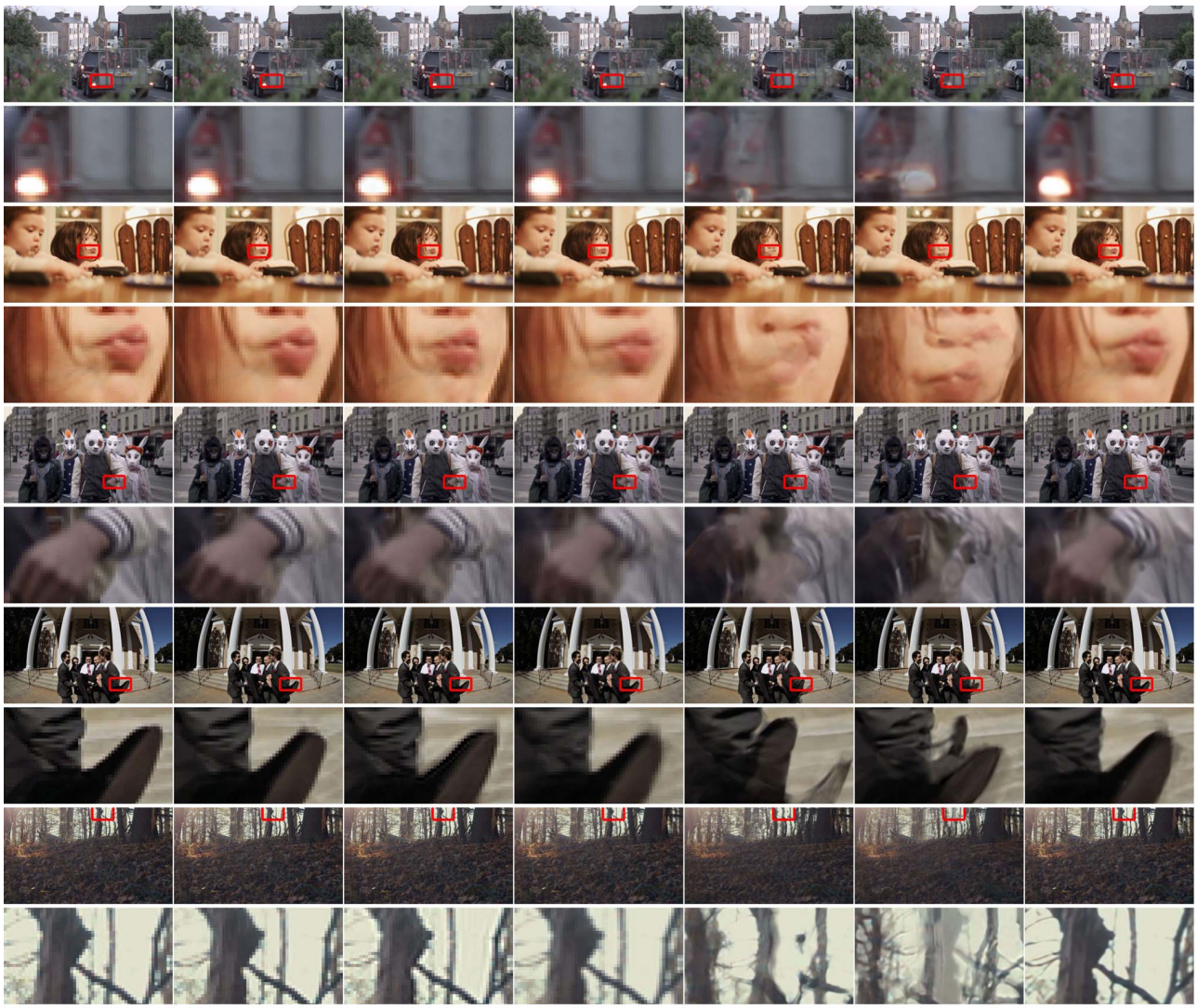

Figure 4. Example results from the Vimeo dataset (best viewed in color), from left to right: 448x256 ground-truth frame, interpolated frame synthesized by TOFlow [31] on 448x256 inputs, SepConv [24] on 448x256 inputs, IM-Net on 448x256 inputs, TOFlow on 1344x768 inputs, SepConv on 1344x768 inputs, and IM-Net on 1344x768 inputs. In each pair of rows we show the full frame on the top and zoom-in of a cropped interesting region (highlighted in a red box) on the bottom.

References

[1] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, pages 1–31, 2011. 6

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*, page 611–625, 2012. 6

[3] R. Castagno, P. Haavisto, and G. Ramponi. A method for motion adaptive frame rate up-conversion. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(5):436–446, 1996. 1

[4] B. D. Choi, J. W. Han, C. S. Kim, and S. J. Ko. Motion-compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(4):407–416, 2007. 1

[5] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, page 184–199, 2014. 2

[6] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016. 2

[7] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1

[8] D. Fourure, R. Emonet, É. Fromont, D. Muselet, A. Trémeau, and C. Wolf. Residual conv-deconv grid network for semantic segmentation. In *British Machine Vision Conference*, 2017. 2

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 6

[10] X. Huang and S. Forchhammer. Cross-band noise model refinement for transform domain Wyner-Ziv video coding. *Signal Processing: Image Communication*, 27(1):16–30, 2012. 1

[11] X. Huang, L. L. Rakêt, H. V. Luong, M. Nielsen, F. Lauze, and S. Forchhammer. Multi-hypothesis transform domain Wyner-Ziv video coding including optical flow. In *IEEE 13th International Workshop on Multimedia Signal Processing*, 2011. 1

[12] B. W. Jeon, G. I. Lee, S. H. Lee, and R. H. Park. Coarse-to-fine frame interpolation for frame rate up-conversion using pyramid structure. *IEEE Transactions on Consumer Electronics*, 49(3):499–508, 2003. 1

[13] H. Jiang, D. Sun, V. Jampani, M. H. Yang, E. Learned-Miller, and J. Kautz. Super Slomo: High quality qstimation of multiple intermediate frames for video interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2, 3, 5, 7

[14] S. J. Kang, K. R. Cho, and Y. H. Kim. Motion compensated frame rate up-conversion using extended bilateral motion estimation. *IEEE Transactions on Consumer Electronics*, 53(4):1759–1767, 2007. 1

[15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 6

[16] S. H. Lee, O. Kwon, and R. H. Park. Weighted-adaptive motion-compensated frame rate up-conversion. *IEEE Transactions on Consumer Electronics*, 49(3):485–492, 2003. 1

[17] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *International Conference on Computer Vision*, 2017. 1, 2, 3, 4, 5

[18] G. Long, L. Kneip, J. M. Alvarez, H. Li, X. Zhang, and Q. Yu. Learning image matching by simply watching video. In *European Conference on Computer Vision*, pages 434–450, 2016. 1, 2

[19] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *International Conference on Learning Representations*, 2016. 1, 2

[20] S. Meyer, A. Djelouah, B. McWilliams, A. Sorkine-Hornung, M. Gross, and C. Schroers. PhaseNet for video frame interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2, 3

[21] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung. Phase-based frame interpolation for video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1410–1418, 2015. 2

[22] S. Niklaus and F. Liu. Context-aware synthesis for video frame interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2, 7

[23] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive convolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2, 4, 5

[24] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive separable convolution. In *International Conference on Computer Vision*, 2017. 1, 2, 3, 4, 5, 7, 8

[25] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. In *CRVC*, 2012. 6

[26] T. Stich, C. Linz, G. Albuquerque, and M. Magnor. View and time interpolation in image space. In *Computer Graphics Forum 27*, page 1781–1787, 2008. 1

[27] J. van Amersfoort, W. Shi, A. Acosta, F. Massa, J. Totz, Z. Wang, and J. Caballero. Frame interpolation with multi-

scale deep loss functions and generative adversarial networks, 2017. arXiv preprint arXiv:1711.06045. 1, 2, 3

[28] Vimeo. https://vimeo.com. 6

[29] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. Deep networks for image super-resolution with sparse prior. In *International Conference on Computer Vision*, page 370–378, 2015. 2

[30] L. Xu, J. SJ. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In *Advances in Neural Information Processing Systems 27*, pages 1790–1798, 2014. 2

[31] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman. Video enhancement with task-oriented flow, 2017. arXiv preprint arXiv:1711.09078. 1, 2, 6, 7, 8

[32] J. Yamanaka, S. Kuwashima, and T. Kurita. Fast and accurate image super resolution by deep CNN with skip connection and network in network. In *International Conference of Neural Information Processing*, pages 217–225, 2017. 7