Photosequencing of Motion Blur using Short and Long Exposures

Vijay Rengarajan1*, Shuo Zhao1, Ruiwen Zhen2, John Glotzbach2, Hamid Sheikh2, Aswin C. Sankaranarayanan1
1Carnegie Mellon University, 2Samsung Research America
vangarai@andrew.cmu.edu

Abstract

Photosequencing aims to transform a motion blurred image to a sequence of sharp images. This problem is challenging due to the inherent ambiguities in temporal ordering as well as the recovery of lost spatial textures due to blur. Adopting a computational photography approach, we propose to capture two short exposure images, along with the original blurred long exposure image to aid in the aforementioned challenges. Post-capture, we recover the sharp photosequence using a novel blur decomposition strategy that recursively splits the long exposure image into smaller exposure intervals. We validate the approach by capturing a variety of scenes with interesting motions using machine vision cameras programmed to capture short and long exposure sequences. Our experimental results show that the proposed method resolves both fast and fine motions better than prior works.

1. Introduction

Capturing photosequences of fast events is a challenge in the photography milieu. High speed capture is costly to implement since it requires specialized electronics capable of handling high data bandwidth as well as highly sensitive image sensors that can make low noise measurements at short exposure periods. As a consequence, it is still a major obstacle in commodity cameras, especially without sacrificing the full spatial resolution of the sensor. This paper provides a pathway for the implementation of such a capability with no additional hardware.

One approach to obtain photosequences is from motion blur. A single motion-blurred photograph embeds motion events inherently, but recovering them can be hard due to ill-posed temporal ordering, as well as the erasure of spatial textures. While prior single-blur sequencing approaches by Jin et al. [10] and Purohit et al. [17] handle both these issues using data-driven priors encoded using deep neural networks, they are incapable of correctly sequencing multiple local motion events happening across the sensor space. The recent multi-blur sequencing technique of Jin et al. [9] handles the overall sequencing problem by processing multiple blurred photographs captured in succession. As much as texture recovery from blur is conditioned using deep neural networks, fine textures can still be lost due to fast motion since all of the captured photographs are blurred. An orthogonal approach to motion blur is to first capture a short-exposure photosequence at higher frame-rates followed by frame-wise denoising as done by Suo et al. [25]. Even though this looks promising, texture loss could still be an issue due to denoising. Further, static regions, which would have benefited from long exposures, suffer due to postprocessing which would otherwise be clean.

This paper proposes to capture a short-long-short exposure triplet to handle both motion ordering and texture recovery in photosequencing. The two additional short-
exposure photographs are designed to have an exposure that is much smaller than the original long exposure photograph. Hence, while these additional images might be noisy, they are practically free of motion-blur. Further, the short exposure photographs resolve temporal ordering by acting as peripheral keypoints of the long exposure.

Once we capture the short-long-short exposure images, we computationally recover the photosequence equivalent to a high-speed capture of the underlying scene. The blurred image is first decomposed into two half-blurred images with reduced blur corresponding to the two halves of the exposure period and a single sharp image corresponding to the mid-point. We learn this decomposition by training a neural network using images synthesized from high frame-rate video datasets. A recursive decomposition leads to the sharp photosequence without any enforced temporal order. The recovered images gain from the complementary goodness of the short and long exposures for static and moving scene regions.

Figs. 1(a,b,c) show the short-long-short exposure images captured in succession of a scene with a moving jellyfish. The short exposures are noisy while the long exposure embeds motion blur. The intricate tentacles are captured in the noisy image but are blurred in the long exposure. Our recovered 15x photosequence corresponding to the long exposure duration is shown in (d). Both texture and motion are recovered correctly in our reconstruction. Most modern sensors, found in machine vision and cellphone cameras, can implement this exposure capture functionality. While we have shown our results on a machine vision camera, applying it to other cameras is a matter of having access to the correct API (which cellphone manufacturers invariably do).

The main contributions of this work are as follows:

1. We propose the novel idea of capturing short-long-short exposure images for the problem of photosequencing of motion blur.
2. We propose a recursive blur decomposition strategy in which increase in temporal super-resolution can be achieved through multiple levels of decomposition till the removal of blur.
3. We present a new short-long-short exposure dataset for a variety of dynamic events captured using a machine vision camera. We provide qualitative evaluation and comparisons to prior art on this data.

2. Related Works

Traditional deblurring approaches [2,3,11,16,24,27,29] aim to get a single sharp image along with local motion kernels and camera trajectories. Reconstructing the whole photosequence for combined object and camera motions has been dealt only recently [10,17]. Single-blur techniques produce photosequence from a single image by imposing ordering constraints through explicit temporal image costs as in the work by Jin et al. [10] or through an implicit cost using a video encoder-decoder framework as in the work by Purohit et al. [17]. These methods suffer from ambiguities in temporal ordering of multiple objects. The recent multi-blur approach by Jin et al. [9] does photosequencing using multiple motion-blurred images to impart more information for reconstruction and ordering. All these photosequencing approaches use neural network priors to reconstruct sharp textures from blur.

One can also increase framerate through temporal interpolation of video frames. An interpolation approach based on gradients and phase is employed by Mahajan et al. [13] and Meyer et al. [14], respectively, while Shahar et al. [21] fuse recurring video patches to perform spatiotemporal super-resolution. Recently, the work by Jiang et al. [8] employed neural networks to linearly interpolate optical flows. However, frame interpolation methods are inherently limited from the assumption of video frames being non-blurred.

A hybrid high resolution, lower frame rate and a low resolution, higher frame rate camera setup is used in the works of Ben-Ezra and Nayar [1], and Tai et al. [26] to remove motion blur from the high resolution capture, but a further photosequencing step is not performed. The coded exposure work of Raskar et al. [18] suggests changing exposure design to improve the conditioning of the deblurring problem. The idea of capturing a noisy-blurry pair in the work by Yuan et al. [28] regularizes deblurring better. While this work employs an additional short exposure image to produce a single sharp image, our goal is to recover the whole photosequence.

3. Problem

Let \( I_t \) be the clean ground-truth image of a scene at \( t \in [0,1] \). Let \( Y(a,b) \) be an observed image for the time interval \([a,b]\) defined as \( Y(a,b) = 1/(b-a) \cdot \int_a^b Y dt + n \) where \( n \) represents noise. The goal of motion-blur photosequencing is capture a blurred image \( Y_{0 \rightarrow 1} = Y(0,1) \) and to estimate the images \( \{\hat{t}_j\}_{j=1}^N \) such that \( Y_{0 \rightarrow 1} \approx \sum_{j=1}^N \hat{t}_j / N \) where \( t_j \)s are equi-spaced timepoints and \( N \) represents the sequence-rate. In essence, the recovered sequence represents the series of images of the scene if it had been captured with a hypothetical higher frame-rate camera operating at the same spatial resolution and without the noise level associated with that frame rate.

3.1. Approach

We propose to capture two short exposure images \( Y_{0 \rightarrow -\Delta t} = Y(-\Delta t, 0) \) and \( Y_{1 \rightarrow} = Y(1,1+\Delta t) \), one before and one after the long exposure image \( Y_{0 \rightarrow 1} \). Though the short exposure images will be noisy owing to the very short exposure
interval $\Delta t$, they provide the much-needed information of scene texture. They also act as temporal endpoints for the blurred image to disambiguate motion directions. Our problem statement is to estimate the image sequence $\{\hat{I}_t\}$ happened during long exposure, given the three input images, $\{Y_{0-}, Y_{0\rightarrow1}, Y_{1+}\}$ as inputs.

### 3.2. Recursive Blur Decomposition

We adopt a multi-step sequencing strategy wherein we progressively increase the number of reconstructed sharp images. We first decompose the long exposure into two half-blurred images $\hat{I}_{0\rightarrow0.5}$ and $\hat{I}_{0.5\rightarrow1}$ and the sharp image $\hat{I}_{0.5}$. Estimating just $\hat{I}_{0.5}$ would correspond to deblurring. In addition, we also estimate $\hat{I}_{0\rightarrow0.5}$ and $\hat{I}_{0.5\rightarrow1}$ that contain lesser motion blur corresponding to each half of the original exposure interval. Our core blur decomposition step, hence, is the following:

$$\{Y_{0-}, Y_{0\rightarrow1}, Y_{1+}\} \rightarrow \{\hat{I}_{0\rightarrow0.5}, \hat{I}_{0.5}, \hat{I}_{0.5\rightarrow1}\}. \quad (1)$$

Our idea then is to perform blur decomposition on both the half-blurred images to further split the blur interval and get a sharp image at their respective middle timepoints as shown in Fig. 2(a). The second-level will result in the sharp sequence $\hat{I}_{0.25}, \hat{I}_{0.5}, \hat{I}_{0.75}$ corresponding to 3x frame-rate. We could perform blur decomposition recursively to achieve a desired sequence-rate; a $k$-level decomposition will provide $2^k - 1$ sharp images. In practice, one could stop the decomposition at a desired level when the blur in half-blurred images is negligible.

### 4. Implementation

We learn the blur decomposition mapping in Eq. (1) by training a neural network.

#### 4.1. Network Architecture

We adopt an encoder-decoder architecture similar to U-net [7, 19] as shown in Fig. 3. We use dense residual blocks [30] which have rich local connections, instead of serialized convolutional layers. We use carry-on convolutional layers through the skip connections from encoder to decoder. The inputs are the short-long-short exposure observations $\{Y_{0-}, Y_{0\rightarrow1}, Y_{1+}\}$ and the outputs are the estimates of half-blurred images and the deblurred image, $\{\hat{I}_{0\rightarrow0.5}, \hat{I}_{0.5}, \hat{I}_{0.5\rightarrow1}\}$. The input layers pass through different initial convolutional layers; similarly the output layers are different for the half-blurred and deblurred images. The input layer of the original exposures share the same weights; the output layer of the half-blurred images share the same weights. The image intensities are in the range $[0, 1]$. All the convolutional layers are followed by Leaky ReLU nonlinearity except for the last layer which is followed by rescaled-tanh to enforce outputs to $[0, 1]$. All convolutional layers use 3x3 filters except for the first layers of both noisy and blurred images which use a filter size of 7x7. More details are provided in the supplementary material.

#### 4.2. Training Data

Our goal is to have the neural network learn to decompose different amounts of blur in the long exposure image with the help of the noisy short exposure image. Training such a network requires a large number of blurred,
half-blurred, noisy and clean images according to Eq. (1). Hence, to create training images, we follow the same process as existing photo-sequence works [9, 10] that use high-speed video datasets – which we augment with a physically-accurate implementation of photon and read noise parameters on-the-fly during training based on the variance given by

\[
\var(N) = \alpha + \beta, \quad \alpha \text{ is the observed mean intensity, and } \beta \text{ are the calibrated noise parameters. We use the machine vision Blackfly BFS-U3-16S2C camera to capture our short and long exposure images.}
\]

Since our technique uses recursive decomposition, the inputs to the network beyond the first level would have denoised short exposure images. However, we observed no significant difference in our test results between employing three separate trained networks with two, one, and zero noisy images for the short-exposure input images appropriately at different decomposition levels, and a single trained network with two noisy images. Hence, we report our results for the single trained model approach, that takes in noisy short exposure images as inputs, and provide comparisons to the three model approach in the supplemental material.

4.3. Optimization

We train the neural network by employing the following costs during optimization: (a) supervised cost and sum cost for \(I_{0 \rightarrow 0.5-\rightarrow 0.5} \), \(I_{0.5} \), and \(I_{0.5 \rightarrow 1} \), and (b) gradient, perceptual, and adversarial costs on the sharp image \(I_{0.5} \).

The supervised cost is defined as the mean square error between the estimated and ground-truth outputs corresponding to \(I_{0 \rightarrow 0.5-\rightarrow 0.5} \), \(I_{0.5} \), and \(I_{0.5 \rightarrow 1} \). The sum cost is defined by the mean square error according to the blur decomposition process. The gradient cost is based on the isotropic total-variation norm [20] on the sharp image that encourages sharp edges with homogeneous regions. The perceptual cost is defined as the mean squared error between the VGG [12, 22] features at the conv54 layer of the estimated and ground truth sharp images. We also train a two-class discriminator alongside our network following a generative adversarial training procedure [5], which contributes the generator adversarial cost \(p_{adv} \) to encourage the sharp image to lie in the natural image distribution.

The cost function is given as follows:

\[
E = \|\hat{I}_{0 \rightarrow 0.5-\rightarrow 0.5} - I_{0 \rightarrow 0.5-\rightarrow 0.5}\|^2 + \|\hat{I}_{0.5} - I_{0.5}\|^2
+ \|\hat{I}_{0.5 \rightarrow 1} - I_{0.5 \rightarrow 1}\|^2
+ \lambda_{sum} \|Y_{0-1} - (N_1 \hat{I}_{0 \rightarrow 0.5} + \hat{I}_{0.5} + N_2 \hat{I}_{0.5 \rightarrow 1}) / N\|^2
+ \lambda_{perc} \|\text{VGG} (\hat{I}_{0.5}) - \text{VGG} (I_{0.5})\|^2
+ \lambda_{adv} TV_0 (\hat{I}_{0.5}) + \lambda_{adv} p_{adv} (\hat{I}_{0.5})
\]

where \(\lambda_{sum}, \lambda_{perc}, \lambda_{adv}, \text{ and } \lambda_{grad}\) are weights of the individual costs. \(TV_0 (\hat{I}_{0.5}) = \sum_{i,j} \sqrt{\nabla_x^2 \hat{I}_{0.5}(i,j) + \nabla_y^2 \hat{I}_{0.5}(i,j)}\) is the total variation norm. We use \(\lambda_{perc} = 3 \times 10^{-4}, \lambda_{adv} = 10^{-4}, \lambda_{grad} = 10^{-4}, \text{ and } \lambda_{sum} = 10^{-2}\) in our experiments for the image intensity range [0, 1]. We train for \(10^6\) iterations using Adam as our optimizer with initial learning rate as \(10^{-4}\) scaling it by 0.1 every 2.5 \(\times 10^4\) iterations.

5. Experiments

We first demonstrate our blur decomposition through a visualization of blur kernels. We then provide quantitative
vides an elegant way to remove blur and achieve our goal. Thus, our recursive decomposition process with further blur reduction and the corresponding two halflong images. Similarly, in the second level, we get four half-blurred images having shorter blur. One can neatly trace down the long kernel trajectory of $Y_0 \rightarrow 1$ by conjoining the split trajectories of level-1. (The kernel estimation is blind, and therefore, it is always centered.) Also, the blur kernel of the middle image $\hat{I}_{0.5}$ has a close-to-delta kernel indicating a negligible blur. Similarly, in the second level, we get four half-blurred images with further blur reduction and the corresponding two deblurred images. Thus, our recursive decomposition provides an elegant way to remove blur and achieve our goal.

5.1. Successive Reduction of Blur

We demonstrate our blur decomposition through blur kernels estimated using the state-of-the-art blind deblurring method of Pan et al. [16]. A blur kernel describes the motion experienced by a point light source located at the image center, and thus acts as an indicator of residual blur/motion present in the image. Fig. 4(a) shows a patch from a motion blurred image $Y_0 \rightarrow 1$ and its long blur kernel. The network takes this image and the short exposure images as inputs (not shown). At first level of decomposition, the half-blurred images have blur kernels of reduced length indicating shorter blur. One can neatly trace down the long kernel trajectory of $Y_0 \rightarrow 1$ by conjoining the split trajectories of level-1. (The kernel estimation is blind, and therefore, it is always centered.) Also, the blur kernel of the middle image $\hat{I}_{0.5}$ has a close-to-delta kernel indicating a negligible blur. Similarly, in the second level, we get four half-blurred images with further blur reduction and the corresponding two deblurred images. Thus, our recursive decomposition provides an elegant way to remove blur and achieve our goal.

5.2. Quantitative Analysis

We analyze the performance of 3x sequence-rate level, i.e. the PSNR of the middle deblurred image $\hat{I}_{0.5}$ and that of the second-level decomposition images, $\hat{I}_{0.25}$ and $\hat{I}_{0.75}$ against the single-blur sequencing of Jin et al. [9] and multi-blur sequencing of Jin et al. [10]. In addition, we also consider an ablation case of our method by considering only the long exposure blurred image as input without any short exposure images, indicated by long-only. Since each of these methods need different types of inputs, we prepare the testing set as follows. We created a set of eleven examples from the eleven test videos of the GoPro 240fps data [15]. Each example is a sequence of alternating short and long exposures. The short exposure is created by taking a single video frame, while the long exposure is synthesized as an average of 11 frames. Our method takes a single consecutive short-long-short triplet as input with added noise to the short exposures. The single-blur sequencing method takes only a single long exposure image, while multi-blur sequencing takes four long exposure images as inputs.

The results of our analysis are provided in Table 1. First, our network trained with short-long-short exposure inputs performs better than training with only the long exposure image indicating the benefit of capturing additional short exposure images in photosequencing. The multi-blur sequencing performs better than the single-blur sequencing owing to more available information as expected. In turn, we are able to perform better than the multi-blur sequencing. Our method recovers textures missed in heavy blur from the short exposure images. Fig. 5(top) depicts this behavior where the prior works are able to reconstruct the leg on the ground which has lesser blur, while the other leg is not recovered even by the multi-blur technique. Our output shows better textures with minimal residual blur and is practically noise-free.

5.3. Blur Amount Analysis

The amount of blur observed in the long exposure image is a synergy of exposure time and motion. We repeated...
Figure 5. *Quasi-real blur and noise on high-speed video GoProTest [15] data.* Comparisons with single-blur sequencing and multi-blur sequencing prior works.

|-------------|-------------|------------------|---------------|---------------|--------------|--------------|

1ms-15ms 1ms-25ms 1ms-32ms

(a) Short and Long exposure images for different exposure intervals.

(b) Single-blur sequencing outputs [10] (takes a single blurred image as input)

(c) Multi-blur sequencing outputs [9] (takes four blurred images as inputs)

(d) Proposed short-long-short sequencing outputs

Figure 6. *Real Spinner data.* Comparisons with single and multi-blur photosequencing prior works for different blur amounts.

1ms-15ms 1ms-25ms 1ms-32ms

Comparison with multi-blur sequencing.

Fig. 6(a) shows our real captures of a spinner for multiple exposure configurations. At the lowest long exposure, the blur is minimal, while at the longest setting, the spinner texture is almost lost. Three frames from sequencing results of different methods are shown in (b,c) and (d). Single-blur technique (b) shows residual blur artifacts even at medium blur shown in the third column-set. While the multi-blur technique (c) could recover some texture at the highest setting, our method outperforms it. Further, single-blur technique could not disambiguate temporal ordering leading to wrong local spinner motion as shown in Fig. 7, while our short-long-short sequencing correctly recovers the ordering.

5.4. Results on Real Data

We capture sequences of short and long exposures of a wide variety of scenes using our Blackfly camera to show the efficacy of our photosequencing, some of which are shown in Fig. 8. Our captures comprise both indoor and outdoor scenes with different types of motions including linear blur in *Race*, rotations in *Foosball*, human nonlinear motions in *Gym* and *Skate*, fine textured motion of *Jellyfish*, and closeup finger movement in *Keyboard*. Note that we cannot use any of the existing real blurred videos since our technique requires short exposure images as well.

Comparison with multi-blur sequencing. Fig. 9 shows the *Foosball* scene with noisy short exposures, and blurry player and ball in long exposures, wherein the partially occluded ball moves from behind the bar into view. (Please zoom-in to see noise and blur clearly in first two rows.) The third and fourth rows show the cropped results of Jin et al. [9] and our method. The method of Jin et al. [9] recovers the motion direction correctly but the blur is not completely removed. Our method correctly recovers the sharpness of the player and the ball as well as the motion ordering.
Figure 8. Some scenes from our real data captures. The whole sequence is available as supplementary.

Figure 9. Comparison with multi-blur sequencing technique on real Foosball data. Recovering the sequence from multiple blurred images [9] lose out on sharp textures albeit correct motion sequencing. Our short-long-short exposure inputs recover both the motion and texture successfully.

Comparison with frame interpolation. Fig. 10 shows input images and crops of the Skate scene in (a,b), in which heavy noise and blur can be observed. The multi-blur sequencing fails to recover intricate textures of the skateboard and fist as seen in (c). We follow two pipelines to compare with frame interpolation [8]. First, we deblur two long exposures using the state-of-the-art deblurring method of Tao et al. [27] and interpolate frames. Second, we denoise two short exposures using BM3D [4] followed by frame interpolation. In the first case result shown in (d), the heavy residual blur due to poor deblurring of heavy motion is passed on to interpolation which can only recover blurred frames. In the second case (e), we do note that the texture in the fist is devoid of blur since it uses only the short exposure frames, however the denoised image has a washed-out appearance. Further, the static wall undergoes noise-denoise process for no reason only to have its texture erased. Our method correctly exploits textures of static regions (wall) from the blurred image and textures of moving objects (hand, fist) from the short exposure image as shown in (f).

The proposed technique recovers a sharp frame sequence for a short-long-short triplet. By combining the outputs from a longer sequence of alternating short and long exposures, we can produce high frame-rate videos of long time durations. We showcase some examples of this in the supplemental video. We do note that such sequences have temporal tiling artifacts since each triplet leads to sequences in isolation. We show XT and YT slices for the result of Skate data over multiple triplets in Fig. 11.

Failure Example. A challenging scenario is that of heavy motion of thin structures. Fig. 12(a) portrays the Jumping-Toy scene wherein the thin legs of the jumping metal toy move very fast. The motion is so ragged that the line of blur marked in yellow is disconnected from the two short exposure points (white to red) as shown in the image crops of Fig. 12(b). Our method shows artifacts in the photo-sequence with partial leg reconstructions as shown in (c). In comparison, the multi-blur sequencing method performs worse as shown in (d), and suffers residual blur artifacts in other regions as well where we perform better.
Figure 10. *Results on Skate data.* Our method correctly transfers textures of static regions (wall linings) from the blurred image and textures of moving objects (hand-fist) from the short exposure image.

Figure 11. *Time slices on Skate.* We recovered the photosequence of consecutive short and long exposures using our method. XT (top) and YT (bottom) slices show discontinuities in the horizontal time axis for the input capture indicating lower frame rate. The slices of our photosequencing result on the right show smooth interpolation of motion during the long exposure intervals.

Figure 12. *Results on JumpingToy data.* Very fast jumping of the thin-legged object misaligns the line of blur with the short exposures as in (b). This causes imperfect reconstructions by our method, and also in multi-blur sequencing as shown in (c,d).

6. Conclusion

We proposed a new technique to record fast motion events by capturing short-long-short exposure images. We utilize their complementary information of texture and motion in these images to estimate the sharp high frame-rate photosequence associated with the long exposure. Our technique provides an approach that can be easily implemented on mobile devices, thereby providing the capability of high-speed capture with little change to existing hardware. Hence, we believe this would be a fascinating new application of mobile computational photography.
References

[27] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Ji-aya Jia. Scale-recurrent network for deep image deblurring. In IEEE Conference on Computer Vision and Pattern Recognition, 2018. 2, 7, 8
Photosequencing of Motion Blur using Short and Long Exposures
Supplementary PDF

Vijay Rengarajan\textsuperscript{1*}, Shuo Zhao\textsuperscript{1}, Ruiwen Zhen\textsuperscript{2}, John Glotzbach\textsuperscript{2}, Hamid Sheikh\textsuperscript{2}, Aswin C. Sankaranarayanan\textsuperscript{1}
\textsuperscript{1}Carnegie Mellon University, \textsuperscript{2}Samsung Research America
\textsuperscript{*}vangarai@andrew.cmu.edu

In this supplementary PDF, we provide the details of our neural network architecture, and comparison between inference approaches based on single and three trained models. The accompanying video provides all our results.

S1. Comparison of Single and Three Model Approaches

As mentioned in Sec. 4.2 of the main paper, since our technique uses recursive decomposition, the inputs to the network beyond the first level would have at least one noise-free estimated image. However, our training involves noise for both the short exposure images. We showed results using this single-model approach in the main paper. As a variation to our single trained model, we also trained three different models (with the same architecture) for the three cases with two, one, and zero noisy images for the short-exposure inputs. This is explained in Fig. S1.

We observed no significant difference in our test results for these two approaches. This can be seen in Fig. S3. For four levels of decomposition as depicted in Fig. S1, we show the estimated images $\hat{I}_4$, $\hat{I}_{12}$ inferred using Model-1 and $\hat{I}_6$ using Model-2 in Fig. S3(b) for Skate and Jellyfish scenes. The corresponding estimated images using the single trained model approach inferred using Model-0 is shown in Fig. S3(a). We can see no observable difference. The relative PSNRs between the estimated images of the two approaches are on the high end showing that both the approaches performs very similarly, and hence, we used the single trained model approach for all our results in the main paper.

S2. Network Architecture

The architecture of our neural network with layer numbering is shown in Fig. S2. The details of each layer, viz. input/output channels, filter size, padding, and stride, are provided in Table S1.

References

Figure S3. Comparison of single and three trained models approaches. Both the approaches perform very similarly as shown in (a) and (b) for the estimation of three images of the sequence depicted in Fig. S1. The relative PSNRs between the two approaches are quite high denoting this behavior.

Table S1. Layer details of our architecture in Fig. S2

<table>
<thead>
<tr>
<th>Layer</th>
<th>Type</th>
<th>chan-in → chan-out</th>
<th>filt, pad, stride</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip1</td>
<td>Conv</td>
<td>3 → 16</td>
<td>7x7, 3, 1</td>
</tr>
<tr>
<td>ip2</td>
<td>Conv</td>
<td>3 → 32</td>
<td>7x7, 3, 1</td>
</tr>
<tr>
<td>1</td>
<td>ResB</td>
<td>64 → 64</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>2</td>
<td>ResB</td>
<td>64 → 64</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>3</td>
<td>Conv</td>
<td>64 → 128</td>
<td>3x3, 1, 2</td>
</tr>
<tr>
<td>4</td>
<td>ResB</td>
<td>128 → 256</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>5</td>
<td>Conv</td>
<td>128 → 256</td>
<td>3x3, 1, 2</td>
</tr>
<tr>
<td>6</td>
<td>ResB</td>
<td>256 → 512</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>7</td>
<td>ResB</td>
<td>256 → 512</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>8</td>
<td>ResB</td>
<td>256 → 512</td>
<td>3x3, 1, 2</td>
</tr>
<tr>
<td>9</td>
<td>Conv</td>
<td>256 → 512</td>
<td>3x3, 1, 2</td>
</tr>
<tr>
<td>10</td>
<td>ResB</td>
<td>512 → 512</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>11</td>
<td>ResB</td>
<td>512 → 512</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>12</td>
<td>Conv</td>
<td>512 → 1024</td>
<td>3x3, 1, 2</td>
</tr>
<tr>
<td>13</td>
<td>ResB</td>
<td>1024 → 1024</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>14</td>
<td>ConvT</td>
<td>1024 → 256</td>
<td>4x4, 1, 2</td>
</tr>
<tr>
<td>15</td>
<td>ResB</td>
<td>512 → 512</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>16</td>
<td>ResB</td>
<td>512 → 512</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>17</td>
<td>ConvT</td>
<td>512 → 128</td>
<td>4x4, 1, 2</td>
</tr>
<tr>
<td>18</td>
<td>ResB</td>
<td>256 → 256</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>19</td>
<td>ResB</td>
<td>256 → 256</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>20</td>
<td>ConvT</td>
<td>256 → 512</td>
<td>4x4, 1, 2</td>
</tr>
<tr>
<td>21</td>
<td>ResB</td>
<td>128 → 128</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>22</td>
<td>ResB</td>
<td>128 → 128</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>23</td>
<td>ConvT</td>
<td>128 → 256</td>
<td>4x4, 1, 2</td>
</tr>
<tr>
<td>24</td>
<td>ResB</td>
<td>64 → 64</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>25</td>
<td>ResB</td>
<td>64 → 64</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>26</td>
<td>ConvT</td>
<td>64 → 32</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>27</td>
<td>Conv</td>
<td>256 → 128</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>28</td>
<td>Conv</td>
<td>128 → 64</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>29</td>
<td>Conv</td>
<td>64 → 32</td>
<td>3x3, 1, 1</td>
</tr>
</tbody>
</table>

Conv - convolutional layer, ConvT - transpose convolutional layer [1].

Structure of Dense Residual Block (ResB)

<table>
<thead>
<tr>
<th>Layer</th>
<th>Type</th>
<th>chan-in → chan-out</th>
<th>filt, pad, stride</th>
</tr>
</thead>
<tbody>
<tr>
<td>d1</td>
<td>Conv</td>
<td>nchan → nchan</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>d2</td>
<td>Conv</td>
<td>nchan → nchan</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>d3</td>
<td>Conv</td>
<td>nchan → nchan</td>
<td>3x3, 1, 1</td>
</tr>
<tr>
<td>d4</td>
<td>Conv</td>
<td>nchan → nchan</td>
<td>3x3, 1, 1</td>
</tr>
</tbody>
</table>

All convolutional layers are followed by Leaky ReLU. Only op1 and op2 are followed by (tanh()+1)/2.