SDC-Net: Video Prediction Using Spatially-Displaced Convolution

Fitsum A. Reda(✉), Guilin Liu, Kevin J. Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro

Nvidia Corporation, Santa Clara, CA 95051, USA
{freda,guilinl,kshih,rkirby,dtarjan,jbarker,atao,bcatanzaro}@nvidia.com

Abstract. We present an approach for high-resolution video frame prediction by conditioning on both past frames and past optical flows. Previous approaches rely on resampling past frames, guided by a learned future optical flow, or on direct generation of pixels. Resampling based on flow is insufficient because it cannot deal with disocclusions. Generative models currently lead to blurry results. Recent approaches synthesis a pixel by convolving input patches with a predicted kernel. However, their memory requirement increases with kernel size. Here, we present *spatially-displaced convolution* (SDC) module for video frame prediction. We learn a motion vector and a kernel for each pixel and synthesize a pixel by applying the kernel at a displaced location in the source image, defined by the predicted motion vector. Our approach inherits the merits of both vector-based and kernel-based approaches, while ameliorating their respective disadvantages. We train our model on 428K unlabelled 1080p video game frames. Our approach produces state-of-the-art results, achieving an SSIM score of 0.904 on high-definition YouTube-8M videos, 0.918 on Caltech Pedestrian videos. Our model handles large motion effectively and synthesizes crisp frames with consistent motion.

Keywords: 3D CNN · Sampling kernel · Optical flow
Frame prediction

1 Introduction

Video prediction is the task of inferring future frames from a sequence of past frames. The ability to predict future frames could find applications in various domains – ranging from future state estimation for self-driving vehicles to video analysis. For a video prediction model to perform well, it must accurately capture not only how objects move, but also how their displacement affects the visibility and appearance of surrounding structures. Our work focuses on predicting one or more immediate next frames that are sharp, realistic and at high resolution (Fig. 1).

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01234-2_44) contains supplementary material, which is available to authorized users.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11211, pp. 747–763, 2018.
https://doi.org/10.1007/978-3-030-01234-2_44

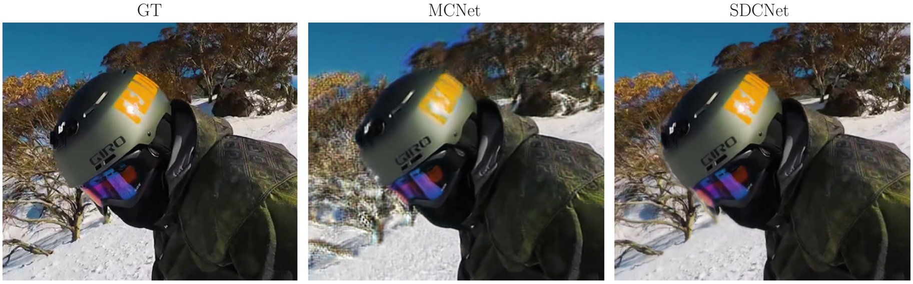


Fig. 1. Frame prediction on a YouTube video frame featuring a panning camera. Left to right: Ground-truth, MCNet [34] result, and our SDC-Net result. The SDC-Net predicted frame is sharper and preserves fine image details, while color distortion and blurriness is seen in the tree and text in MCNet's predicted frame. (Color figure online)

Another attribute of the video prediction task is that models can be trained on raw unlabeled video frames. We train our models on large amounts of high resolution footage from video game sequences, which we find improves accuracy because video game sequences contain a large range of motion. We demonstrate that the resulting models perform well not only on video game footage, but also on real-life footage.

Video prediction is an active research area and our work builds on the literature [2–4,13,18–20,26,33,35,37]. Previous approaches for video prediction often focus on direct synthesis of pixels using generative models. For instance, convolutional neural networks were used to predict pixel RGB values, while recurrent mechanisms were used to model temporal changes. Ranzato et al. [28] proposed to partition input sequences into a dictionary of image patch centroids and trained recurrent neural networks (RNN) to generate target images by indexing the dictionaries. Srivastava et al. [31] and Villegas et al. [34] used a convolutional Long-Short-Term-Memory (LSTM) encoder-decoder architecture conditioned on previous frame data. Similarly, Lotter et al. [17] presented a predictive coding RNN architecture to model the motion dynamics of objects in the image for frame prediction. Mathieu et al. [21] proposed a multi-scale conditional generative adversarial network (GAN) architecture to alleviate the short range dependency of single-scale architectures. These approaches, however, suffer from blurriness and do not model large object motions well. This is likely due to the difficulty in directly regressing to pixel values, as well as the low resolution and lack of large motion in their training data.

Another popular approach for frame synthesis is learning to transform input frames. Liang et al. [14] proposed a generative adversarial network (GAN) approach with a joint future optical-flow and future frame discriminator. However, ground truth optical flows are not trivial to collect at large scale. Training with estimated optical flows could also lead to erroneous supervision signals. Jiang et al. [10] presented a model to learn offset vectors for sampling for frame interpolation, and perform frame synthesis using bilinear interpolation guided by the learned sampling vectors. These approaches are desirable in modeling large motion. However, in our experiments, we found sampling vector-based synthesis results are often affected by speckled noise.

One particular approach proposed by Niklaus et al. [23,24] and Vondrick et al. [36] for frame synthesis is to learn to predict sampling kernels that adapt to each output pixel. A pixel is then synthesized as the weighted sampling of a source patch centered at the pixel location. Niklaus et al. [23,24] employed this for the related task of video frame interpolation, applying predicted sampling kernels to consecutive frames to synthesize the intermediate frame. In our experiments, we found the kernel-based approaches to be effective in keeping objects intact as they are transformed. However, this approach cannot model large motion, since its displacement is limited by the kernel size. Increasing kernel size can be prohibitively expensive.

Inspired by these approaches, we present a spatially-displaced convolutional (SDC) module for video frame prediction. We learn a motion vector and a kernel for each pixel and synthesize a pixel by applying the kernel at a displaced location in a source image, defined by the predicted motion vector. Our approach inherits the merits of both sampling vector-based and kernel-based approaches, while ameliorating their respective disadvantages. We take the large-motion advantage of sampling vector-based approach, while reducing the speckled noise patterns. We take the clean object boundary prediction advantages of the kernel-based approaches, while significantly reducing kernel sizes, hence reducing the memory demand.

The contributions of our work are:

- We propose a deep model for high-resolution frame prediction from a sequence of past frames.
- We propose a spatially-displaced convolutional (SDC) module for effective frame synthesis via transformation learning.
- We compare our SDC module with kernel-based, vector-based and state-of-the-art approaches.

2 Methods

Given a sequence of frames $\mathbf{I}_{1:t}$ (the immediate past t frames), our work aims to predict the next future frame $\mathbf{I}_{t+1}$. We formulate the problem as a transformation learning problem

$$\mathbf{I}_{t+1} = \mathcal{T}\Big(\mathcal{G}\big(\mathbf{I}_{1:t}\big), \mathbf{I}_{1:t}\Big), \tag{1}$$

where $\mathcal{G}$ is a learned function that predicts transformation parameters, and $\mathcal{T}$ is a transformation function. In prior work, $\mathcal{T}$ can be a bilinear sampling operation guided by a motion vector [10,15]:

$$\mathbf{I}_{t+1}(x, y) = f\big(\mathbf{I}_t(x + u, y + v)\big), \tag{2}$$

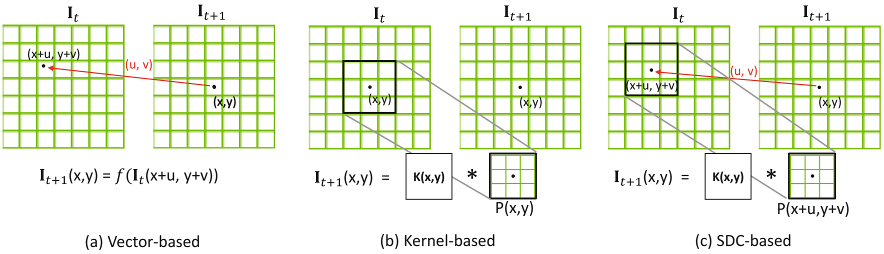
where f is a bilinear interpolator [15], (u, v) is a motion vector predicted by $\mathcal{G}$, and $\mathbf{I}_t(x, y)$ is a pixel value at (x, y) in the immediate past frame $\mathbf{I}_t$. We refer this approach as a vector-based resampling. Figure 2a illustrates this approach.

An alternative approach is to define $\mathcal{T}$ as a convolution module that combines motion or displacement learning and resampling into a single operation [23,24,36]:

$$\mathbf{I}_{t+1}(x, y) = \mathrm{K}(x, y) * \mathbf{P}_t(x, y), \tag{3}$$

where $\mathrm{K}(x, y) \in \mathrm{R}^{\mathrm{N}\times\mathrm{N}}$ is an $\mathrm{N} \times \mathrm{N}$ 2D kernel predicted by $\mathcal{G}$ at (x, y) and $\mathbf{P}_t(x, y)$ is an $\mathrm{N} \times \mathrm{N}$ patch centered at (x, y) in $\mathbf{I}_t$. We refer this approach as adaptive kernel-based resampling [23,24]. Figure 2b illustrates this approach.

Since Eq. (2) considers few pixels in synthesis, its results often appear degraded by speckled noise patterns. It can, however, model large displacements without a significant increase in parameter count. On the other hand, Eq. (3) produces visually pleasing results for small displacements, but requires large kernels to be predicted at each location to capture large motions. As such, the kernel-based approach can easily become not only costly at inference, but also difficult to train.



(a) Vector-based (b) Kernel-based (c) SDC-based

Fig. 2. Illustration of sampling-based pixel synthesis. (a) Vector-based with a bilinear interpolation. (b) Kernel-based, a convolution with a centered patch. (c) our SDC-based method, a convolution with a displaced patch.

2.1 Spatially Displaced Convolution

To achieve the best of both worlds, we propose a hybrid solution – the *Spatially Displaced Convolution* (SDC). The SDC uses predictions of both a motion vector (u, v) and an adaptive kernel $\mathrm{K}(x, y)$, but convolves the predicted kernel with a patch at the displaced location $(x + u, y + v)$ in $\mathbf{I}_t$. Pixel synthesis using SDC is computed as:

$$\mathbf{I}_{t+1}(x, y) = \mathrm{K}(x, y) * \mathbf{P}_t(x + u, y + v). \tag{4}$$

The predicted pixel $\mathbf{I}_{t+1}(x, y)$ is thus the weighted sampling of pixels in an $\mathrm{N} \times \mathrm{N}$ region centered at $(x + u, y + v)$ in $\mathbf{I}_t$. The patch $\mathbf{P}_t(x + u, y + v)$ is bilinearly sampled at non-integral coordinates. Figure 2c illustrates our SDC-based approach.

Setting $K(x, y)$ to a kernel of all-zeros except for a one at the center reduce the SDC to Eq. (2), whereas setting u and v to zero reduces it to Eq. (3). However, it is important to note that the SDC is not the same as applying Eqs. (2) and

(3) in succession. If applied in succession, the N × N patch sampled by $K(x, y)$ would be subject to the resampling effect of Eq. (2) as opposed to being the original patch from $\mathbf{I}_t$.

Our SDC effectively decouples displacement and kernel learning, allowing us to achieve the visually pleasing results of kernel-based approaches while keeping the kernel sizes small. We also adopt separable kernels [24] for $\mathrm{K}(x, y)$ to further reduce computational cost. At each location, we predict a pair of 1D kernels and calculate the outer-product of them to form a 2D kernel. This reduces our kernel parameter count from N^2 to $2\mathrm{N}$. In total, our model predicts $2\mathrm{N} + 2$ parameters for each pixel, including the motion vector. We empirically set $N = 11$. Inference at 1080p resolution uses 174 MB of VRAM, which easily fits in GPU memory.

We develop deep neural networks to learn motion vectors and kernels adapted to each output pixel. The SDC is fully differentiable and thus allows our model to train end-to-end. Losses for training are applied to the SDC-predicted frame. We also condition our model on both past frames and past optical flows. This allows our network to easily capture motion dynamics and evolution of pixels needed to learn the transformation parameters. We formulate our model as:

$$\mathbf{I}_{t+1} = \mathcal{T}\Big(\mathcal{G}\big(\mathbf{I}_{1:t}, \mathbf{F}_{2:t}\big), \mathbf{I}_t\Big), \tag{5}$$

where transformation $\mathcal{T}$ is realized with SDC and operates on the most recent input $\mathbf{I}_t$, and $\mathbf{F}_i$ is the backwards optical flow (see Sect. 2.3) between $\mathbf{I}_i$ and $\mathbf{I}_{i-1}$. We calculate $\mathbf{F}$ using state-of-the-art neural network-based optical flow models [7,9,32].

Our approach naturally extends to multiple frame prediction $\mathbf{I}_{t+1:t+D}$ by recursively re-circulating SDC predicted frames back as inputs. For instance, to predict a frame two steps ahead, we re-circulate the SDC predicted frame $\mathbf{I}_{t+1}$ as input to our model to predict $\mathbf{I}_{t+2}$.

2.2 Network Architecture

We realize $\mathcal{G}$ using a fully convolutional network. Our model takes in a sequence of past frames $\mathbf{I}_{1:t}$ and past optical flows $\mathbf{F}_{2:t}$ and outputs pixel-wise separable kernels $\{\mathrm{K}_u, \mathrm{K}_v\}$ and a motion vector (u, v). We use 3D convolutions to convolve across width, height, and time. We concatenate RGB channels from our input images to the two optical flow channels to create 5 channels per frame. The topology of our architecture gets inspiration from various V-net type typologies [7,22,29], with an encoder and a decoder. Each layer of the encoder applies 3D convolutions followed by a Leaky Rectified Unit (LeakyRELU) [8] and a convolution with a stride $(1, 2, 2)$ to downsample features to capture long-range spatial dependencies. Following [7], we use $3 \times 3 \times 3$ convolution kernels, except for the first and second layers where we use $3 \times 7 \times 7$ and $3 \times 5 \times 5$ for capturing large displacements. Each decoder sub-part applies deconvolutions [16] followed by LeakyRELU, and a convolution after corresponding features from the contracting part have been concatenated. The decoding part also has several heads, one head for (u, v) and one each for K_u and K_v. The last two decoding layers of K_u and K_v use upsampling with a trilinear mode, instead of normal deconvolution, to minimize the checkerboard effect [25]. Finally, we apply repeated convolutions in each decoding head to reduce the time dimension to 1 (Fig. 3).

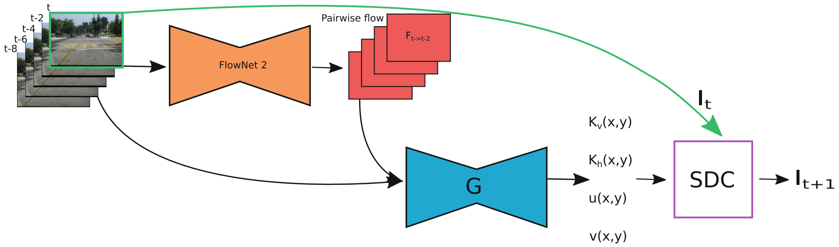


Fig. 3. Our model $\mathcal{G}$ takes in a frame sequence and pairwise flow estimates as input, and returns parameters for the SDC module to transform $\mathbf{I}_t$ to $\mathbf{I}_{t+1}$.

2.3 Optical Flow

We calculate the inter-frame optical flow we input to our model $\mathcal{G}$ using FlowNet2 [9], a state-of-the-art optical flow model. This allows us to extrapolate motion conditioned on past flow information. We calculate backwards optical flows because we model our transformation learning problem with backwards resampling, i.e. predict a sampling location in $\mathbf{I}_t$ for each location in $\mathbf{I}_{t+1}$.

It is important to note that the motion vectors (u, v) predicted by our model $\mathcal{G}$ at each pixel are not equivalent to optical flow vectors $\mathbf{F}_{t+1}$, as pure backwards optical flow is undefined (or zero valued) for dis-occluded pixels (pixels not visible in the previous frame due to occlusion). A schematic explanation of the disocclusion problem is shown in Fig. 4, where a 2×2 square is moving horizontally to the right at a speed of 1 pixel per step. The ground-truth backward optical flow at $t = 1$ is shown in Fig. 4b. As shown in Fig. 4c, resampling the square at $t = 0$ using the perfect optical flow will duplicate the left border of the square because the optical flow is zero at the second column. To achieve a perfect synthesis via resampling at $t = 1$, as shown in Fig. 4e, adaptive sampling vectors must be used. Figure 4d shows an example of such sampling vectors, in which a -1 is used to fill-in dis-occluded region. A learned approach is necessary here as it not only allows the disocclusion sampling to adapt for various degrees of motion, but also allows for a learned solution for which background pixels from the previous frame would look best in the filled gap.

2.4 Loss Functions

Our primary loss function is the L1 loss over the predicted image: $\mathcal{L}_1 = \left\|\mathbf{I}_{t+1} - \mathbf{I}^g_{t+1}\right\|_1$, where $\mathbf{I}^g_i$ is a target and $\mathbf{I}_i$ is a predicted frame. We found the L1 loss to be better at capturing small changes compared to L2, and generally produces sharper images.

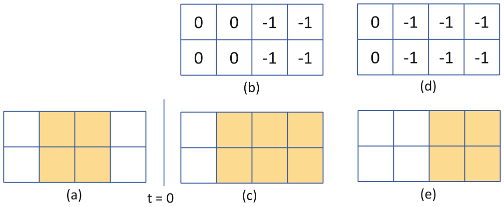


Fig. 4. Disocclusion illustration using backwards optical-flow. Values in top-row indicate vector magnitude in the horizontal axis. (a) frame at $t = 0$; (b) optical flow at $t = 1$; (c) output of resampling (a) using (b); (d) correct sampling vectors; and (e) resampling of (a) using (d). A direct use of optical-flow for frame prediction leads to undesirable foreground stretching in dis-occluded pixels.

We also incorporated the L1 norm on high-level VGG-16 feature representations [30]. Specifically, we used the perceptual and style loss [11], defined as:

$$\mathcal{L}_{perceptual} = \sum_{l=1}^{\mathrm{L}} \kappa_l \left\| \mathit{\Psi}_l(\mathbf{I}_{t+1}) - \mathit{\Psi}_l(\mathbf{I}^g_{t+1}) \right\|_1, \tag{6}$$

and

$$\mathcal{L}_{style} = \sum_{l=1}^{\mathrm{L}} \kappa_l \left\| \left(\mathit{\Psi}_l(\mathbf{I}_{t+1})\right)^{\intercal} \left(\mathit{\Psi}_l(\mathbf{I}_{t+1})\right) - \left(\mathit{\Psi}_l(\mathbf{I}^g_{t+1})\right)^{\intercal} \left(\mathit{\Psi}_l(\mathbf{I}^g_{t+1})\right) \right\|_1. \tag{7}$$

Here, $\mathit{\Psi}_l(\mathbf{I}_i)$ is the feature map from the lth selected layer of a pre-trained Imagenet VGG-16 for $\mathbf{I}_i$, L is the number of layers considered, and κ_l is a normalization factor $1/C_l H_l K_l$ (channel, height, width) for the lth selected layer. We use the perceptual and style loss terms in conjunction with the L1 over RGB as follows:

$$\mathcal{L}_{finetune} = w_l \mathcal{L}_1 + w_s \mathcal{L}_{style} + w_p \mathcal{L}_{perceptual}. \tag{8}$$

We found the finetune loss to be robust in eliminating the checkerboard artifacts and generates a much sharper prediction than L1 alone.

Finally, we introduce a loss to initialize the adaptive kernels, which we found to significantly speed up training. We use the L2 norm to initialize kernels K_u and K_v as a middle-one-hot vector each. That is, all elements in each kernel are set very close to zero, except for the middle element which is initialized close to one. When K_u and K_v elements are initialized as middle-hot vectors, the output of our displaced convolution described in Eq. (4) will be the same as our vector-based approach described in Eq. (2). The kernel loss is expressed as:

$$\mathcal{L}_{kernel} = \sum_{x=1}^{\mathrm{W}} \sum_{y=1}^{\mathrm{H}} \left(\left\| \mathrm{K}_u(x,y) - \mathbf{1}^{<\frac{\mathrm{N}}{2}>} \right\|_2^2 + \left\| \mathrm{K}_v(x,y) - \mathbf{1}^{<\frac{\mathrm{N}}{2}>} \right\|_2^2 \right), \tag{9}$$

where $\mathbf{1}^{<\frac{\mathrm{N}}{2}>}$ is a middle-one-hot vector, and W and H are the width and height of images.

Other loss functions considered include the L1 or L2 distance between predicted motion vectors (u, v) and target optical flows. We found this loss to lead to inferior results. As discussed in Sect. 2.3, optimizing for optical flow will not properly handle the disocclusion problem. Further, use of estimated optical flow as a training target introduces additional noise.

2.5 Training

We trained our SDC model using frames extracted from many short sequence videos. To allow our model to learn robust invariances, we selected frames in high-definition video game plays with realistic, diverse content, and a wide range of motion. We collected 428K 1080p frames from GTA-V and Battlefield-1 game plays. Each example consists of five $(t = 5)$ consecutive 256×256 frames randomly cropped from the full-HD sequence. We use a batch size of 128 over 8 V100 GPUs.

We optimize with Adam [12] using $\beta_1 = 0.9$, and $\beta_2 = 0.999$ with no weight decay. First, we optimize our model to learn (u, v) using $\mathcal{L}_1$ loss with a learning rate of $1e^{-4}$ for 400 epochs. Optimizing for (u, v) alone allows our network to capture large and coarse motions faster. Next, we fix all weights of the network except for the decoding heads of K_u and K_v and train them using our $\mathcal{L}_{kernel}$ loss defined in Eq. (9) to initialize kernels at each output pixel as middle-one-hot vectors. Then, we optimize all weights in our deep model using $\mathcal{L}_1$ loss and a learning rate of $1e^{-5}$ for 300 epochs to jointly fine-tune the (u, v) and (K_u, K_v) at each pixel. Since we optimize for both kernels and motion vectors in this step, our network learns to pick up small and subtle motions and corrects disocclusion related artifacts. Finally, we further fine-tune all weights in our model using $\mathcal{L}_{finetune}$ at a learning rate of $1e^{-5}$. The weights we use to combine losses are 0.2, 0.06, 36.0 for w_l, w_p, and w_s respectively. We used the activations from VGG-16 layers `relu1_2`, `relu2_2` and `relu3_3` for the perceptual and style loss terms. The last fine-tuning step of our training makes predictions sharper and produces visually appealing frames in our video prediction task. We initialized the FlowNet2 model with pre-trained weights[1] and fix them during training.

3 Experiments

We implemented all our Vector, Kernel, and SDC-based models using PyTorch [27]. To efficiently train our model, we wrote a CUDA custom layer for our SDC module. We set kernel size to 51×51 for the Kernel-based model as suggested in [24]. The kernel size for our SDC-based model is 11×11. Inference using our SDC-based model at 1080p takes 1.66 s, of which 1.03 s is spent on FlowNet2.

[1] https://github.com/lmb-freiburg/flownet2.

3.1 Datasets and Metrics

We considered two classes of video datasets that contain complex real world scenes: Caltech Pedestrian [5,6] (CaltechPed) car-mounted camera videos and 26 high definition videos collected from YouTube-8M [1].

We used metrics L1, Mean-Squared-Error (MSE/L2) [17], Peak-Signal-To-Noise (PSNR), and Structural-Similarity-Image-Metric (SSIM) [38] to evaluate quality of prediction. Higher values of SSIM and PSNR indicate better quality.

3.2 Comparison on Low-Quality Videos

Table 1. Next frame prediction accuracy on Caltech Pedestrian [5, 6]. L2 results are in 1e−3.

Methods	L2	SSIM
BeyondMSE [21]	3.42	0.847
PredNet [17]	3.13	0.884
MCNet [34]	2.50	0.879
DualGAN [14]	2.41	0.899
CopyLast	5.84	0.811
Our Vector-based	2.47	0.902
Our Kernel-based	2.19	0.896
Our SDC-based	**1.62**	**0.918**

Table 1 presents next frame prediction comparisions with BeyondMSE [21], PredNet [17], MCNet [34], and DualGAN [14] on CaltechPed test partition. We also compare with CopyLast, which is the trivial baseline that uses the most recent past frame as the prediction. For PredNet and DualGAN, we directly report results presented in [17] and [15], respectively. For BeyondMSE[2] and MCNet[3], we evaluated using released pre-trained models.

Our SDC-based model outperforms all other models, achieving an L2 score of 1.62×10^{-3} and SSIM of 0.918, compared to the state-of-the-art DualGAN model which has an L2 score of 2.41×10^{-3} and SSIM of 0.899. The MCNet which was trained on dataset that is equally as large as ours shows inferior results with L2 of 2.50×10^{-3} and SSIM of 0.879. CopyLast method has significantly worse L2 of 5.84×10^{-3} and SSIM of 0.811, making it a significantly less viable approach for next frame prediction. Our Vector-based approach has higher accuracy than our Kernel-based approach. Since the CaltechPed videos contain slightly larger motion, the Vector-based approach, which is advantageous in large motion sequences, is expected to perform better.

In Fig. 5, we present qualitative comparisons on CaltechPed. SDC-Net predicted frames are crisp, sharp and show minimal un-natural deformation of the highlighted car (framed in red). All predictions were able in picking up the right motion as shown with their good alignment with the ground-truth frame. However, both BeyondMSE and MCNet create generally blurrier predictions and unnatural deformations on the highlighted car.

3.3 Comparison on High-Definition Videos

Table 2 presents next frame prediction comparisons with BeyondMSE, MCNet and CopyLast on 26 full-HD YouTube vidoes. Our SDC-Net model outperforms

[2] https://github.com/coupriec/VideoPredictionICLR2016.

[3] https://github.com/rubenvillegas/iclr2017mcnet.

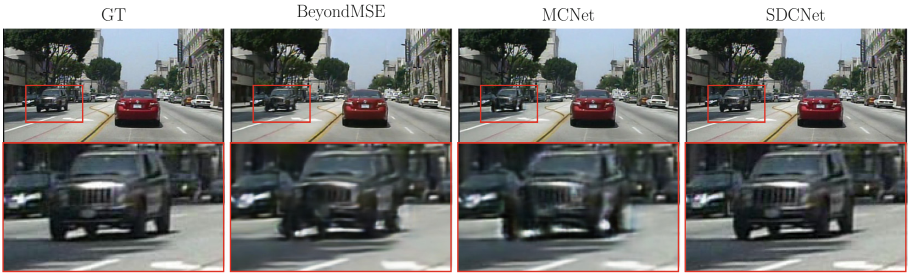


Fig. 5. Qualitative comparison for Caltech (set006-v001/506th frame). Left to right: Ground-truth, BeyondMSE, MCNet, and SDC-Net predicted frames. (Color figure online)

Table 2. Next frame prediction accuracy on YouTube-8M [1].

YouTube8M	L1	L2	PSNR	SSIM
BeyondMSE [21]	0.0271	0.00328	33.33	0.858
MCNet [34]	0.0216	0.00255	35.64	0.895
CopyLast	0.0260	0.00506	36.63	0.854
Our Vector	0.0177	0.00270	37.24	0.905
Our Kernel	0.0186	0.00303	**37.33**	0.904
Our SDC	**0.0174**	**0.00240**	37.15	**0.911**

all other models, achieving an L2 of 2.4×10^{-3} and SSIM of 0.911, compared to the state-of-the-art MCNet model which has an L2 of 2.55×10^{-3} and SSIM of 0.895.

In Fig. 6, SDCNet is shown to provide crisp and sharp frames, with motion mostly in good alignment with the ground-truth frame. Since our models do not hallucinate pixels, they produce visually good results by exploiting the image content of the last input frame. For instance, instead of duplicating the borders of foreground objects, our models displace to appropriate locations in the previous frame and synthesize pixels by convolving the learned kernel for that pixel with an image patch centered at the displaced location.

Since our approach takes FlowNet2 [9] predicted flows as part of its input, the transformation parameters predicted by our deep model are affected by inaccurate optical flows. For instance, optical flow for the ski in Fig. 6 (bottom right) is challenging, and so the ski movement was not predicted by our model as well as the movement of the skiing person.

In Fig. 7, we qualitatively show comparisons for MCNet, our Kernel-, Vector-, and SDC-based methods for a large camera motion. MCNet shows significantly blurry results and ineffectiveness in capturing large motions. MCNet also significantly alters the color distribution in the predicted frames. Our Kernel-based method has difficulty predicting large motion, but preserves the color

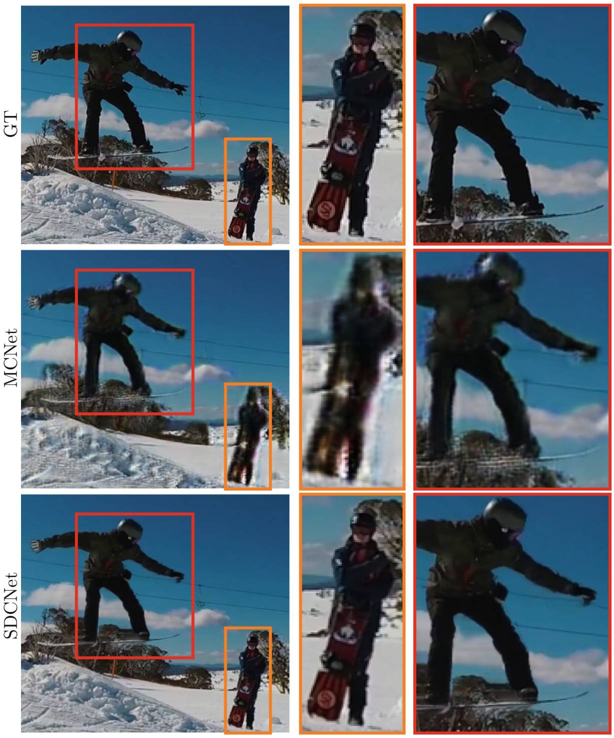


Fig. 6. Comparison of frame prediction methods. Shown from top to bottom are Ground-truth image, MCNet and SDC-Net results. SDCNet is shown to provide crisp and sharp frames, with motion mostly in good alignment with the ground-truth frame. MCNet results on the other hand appear blurry, with artifacts surrounding the persons (framed in red and orange). MCNet results also show checkerboard artifacts near the skis and on the snow background. (Color figure online)

distribution. However, the Kernel-based approach often moves components disjointly, leading to visually inferior results. Our Vector-based approach better captures large displacement, such as the motion present in this sequence. However, its predictions suffer from pixel noise. Our SDC-based method, which combines merits of both our Kernel- and Vector-based approaches, combines the ability of our Vector-based method to predict large motions, along with the visually pleasing results of our Kernel-based approach.

3.4 Comparison in Multi-step Prediction

Previous experiments showed SDCNet's performance in next frame prediction. In practice, models are used to predict multiple future frames. Here, we condition each approach on five original frames and predict five future frames on CaltechPed. Figure 8 shows that SDCNet predicted multiple frames are consistently favourable when compared to previous approaches, as quantified by L1,

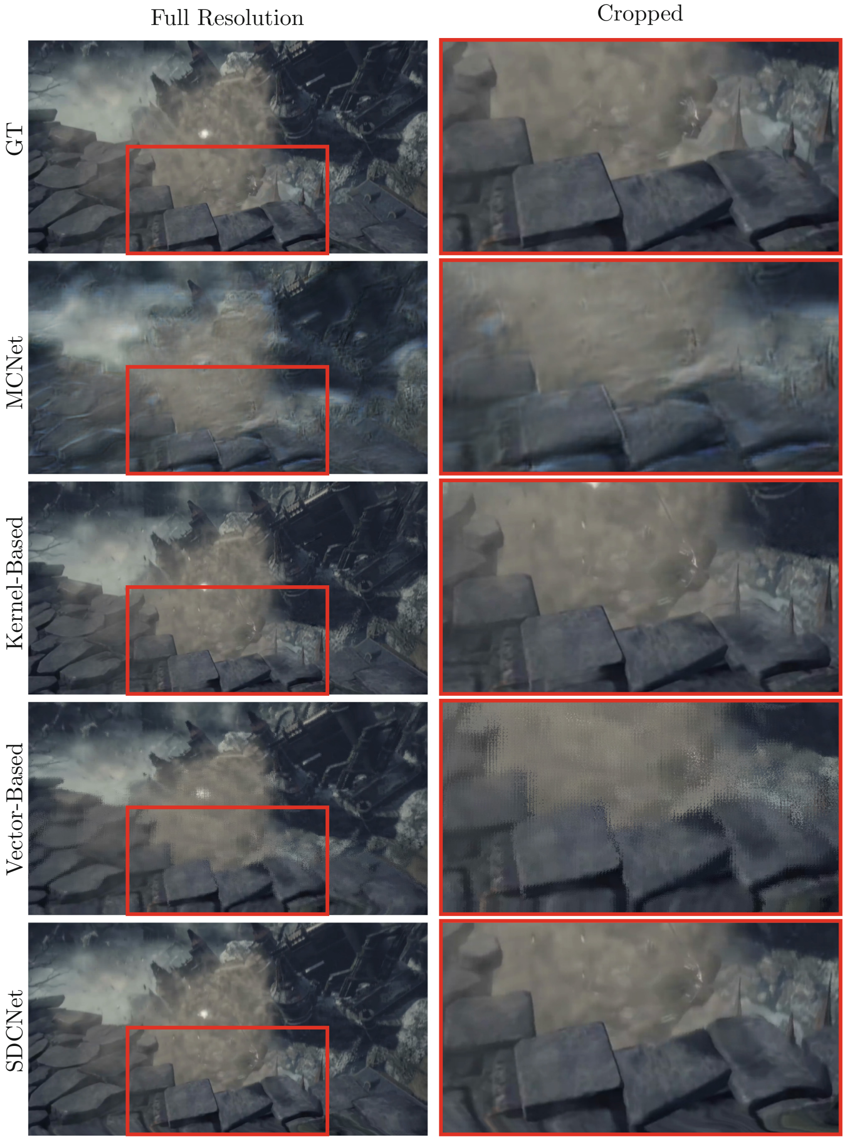


Fig. 7. Comparison of frame prediction for large motion. Expected transformation is an upwards displacement with a slight zoom-in. While the Kernel-based, Vector-based, and SDC-based models were all trained with L1 and fine-tuned with style-loss to promote sharpness, note that the Vector-based result still loses coherence when predicting large displacement. On the other hand, the SDCNet is able to displace as much as the Vector-based model while maintaining sharpness. While the Kernel-based result is relatively sharp, it is conservative about predicting the upwards translation (note the relative distance of tiles to the bottom of the frame compared to the vector and SDC approaches). Further, there is a slight ghosting effect in the right-most tile of the Kernel-based result, which is not present in the SDC result.

L2, SSIM and PSNR over 120,725 unique Caltech Pedestrian frames. Figure 9 presents an example five-step prediction that show SDCNet predicted frames preserving color distribution, object shapes and their fine details.

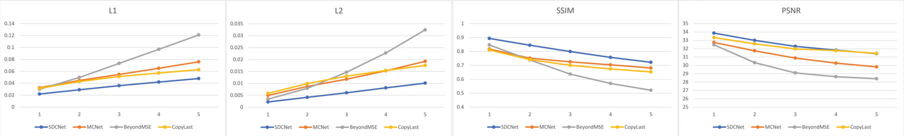


Fig. 8. Quantitative *five-step* prediction results for SDC-Net (blue), MCNet (orange), BeyondMSE (gray) and CopyLast (yellow). SDCNet shows consistently better results as quantified by L1, L2, PSNR and SSIM over 120,725 unique CaltechPed frames. (Color figure online)

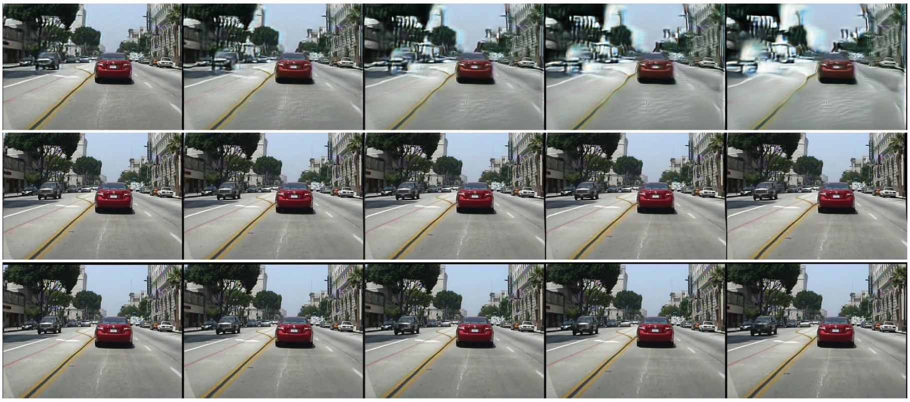


Fig. 9. Qualitative *five-step* prediction results for MCNet (top row), SDCNet (middle row), and Ground Truth (bottom row). Both MCNet and SDCNet were conditioned on the same set of five frames (not seen in the figure).

3.5 Ablation Results

We compare our Vector-based with our SDC-based approach in Fig. 10. Our Vector-based approach struggles with disocclusions (orange box), as described in Sect. 2.3. In Fig. 10, the Vector-based model avoids completely stretching the glove borders, but still leaves some residual glove pixels behind. The Vector-based approach also may produce speckled noise patterns due to large motion (red box). Disocclusion and speckled noise are significantly reduced in the SDC-Net results shown in Fig. 10.

In Fig. 11, we present qualitative results for our SDC-based model trained using $\mathcal{L}_1$ loss alone vs $\mathcal{L}_1$ followed by our $\mathcal{L}_{finetune}$ given by Eq. (8). We note that using $\mathcal{L}_1$ loss alone leads to slightly blurry results, e.g. the glove (red box), and the fence (orange box) in Fig. 11. Figure 11 (center column) shows the same

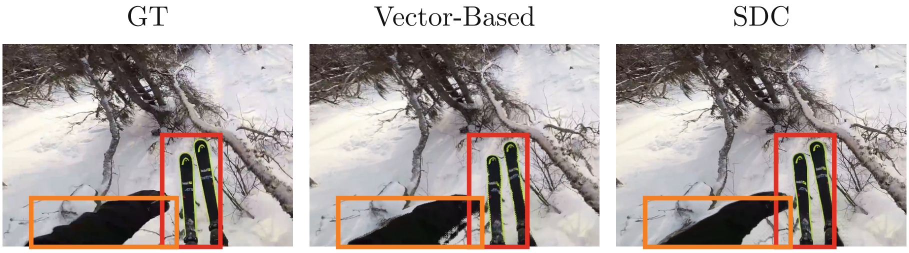


Fig. 10. Comparison of frame synthesis operations. Ground-truth frame (left), Vector-based sampling (middle), and SDC (right). Some foreground duplication (orange box) and inconsistent pixel synthesis (red box, may require zooming in) are present in the Vector-based approach but resolved in the SDC results. (Color figure online)

result after fine-tuning, with finer details preserved – demonstrating that the perceptual and style losses reduce blurriness. We also observed that the $\mathcal{L}_1$ loss helps capture large motions that are otherwise challenging to capture.

Figure 11 represents a challenging example due to fast motion. Since our model depends on optical flow, situations that are challenging for optical flow are also difficult for our model. The prediction errors can be seen with the relatively larger misalignment on the fence compared to the ground truth (orange box). Our approach also fails during scene transitions, where past frames are not relevant to future frames. Currently, we automatically detect scene transitions by analyzing optical flow statistics, and skip frame prediction until enough (five) valid frames to condition our models are available.

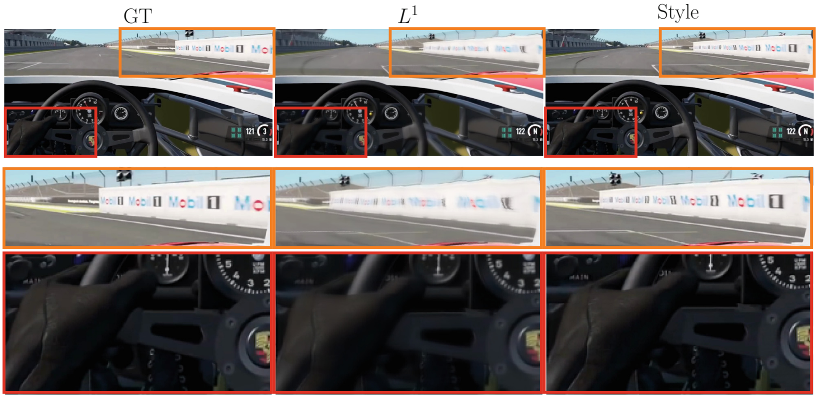


Fig. 11. Comparison of loss functions. Ground-truth (left), L1 loss (middle), and Fine-tuned result with style loss (right). Fine-tuning with style loss can improve the sharpness of results, as seen in the rendered text on the barriers and fence (orange crop) as well as the glove (red crop). (Color figure online)

4 Conclusions

We present a 3D CNN and a novel spatially-displaced convolution (SDC) module that achieves state-of-the-art video frame prediction. Our SDC module effectively handles large motion and allows our model to predict crisp future frames with motion closely matching that of ground-truth sequences. We trained our model on 428K high-resolution video frames collected from gameplay footage. To the best of our knowledge, this is the first attempt in transfer learning from synthetic to real life for video frame prediction. Our model's accuracy is dependent on the accuracy of the input estimated flows, thus leading to failures in fast motion sequences. Future work will include a study on the effect of multi-scale architectures for fast motion.

Acknowledgements. We would like to thank Jonah Alben, Paulius Micikevicius, Nikolai Yakovenko, Ming-Yu Liu, Xiaodong Yang, Atila Orhon, Haque Ishfaq and NVIDIA Applied Research staff for suggestions and discussions, and Robert Pottorff for capturing the game datasets used for training.

References

1. Abu-El-Haija, S., et al.: YouTube-8M: a large-scale video classification benchmark. arXiv preprint arXiv:1609.08675 (2016)
2. Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R.H., Levine, S.: Stochastic variational video prediction. arXiv preprint arXiv:1710.11252 (2017)
3. Byeon, W., Wang, Q., Srivastava, R.K., Koumoutsakos, P.: Fully context-aware video prediction. arXiv preprint arXiv:1710.08518 (2017)
4. Denton, E., Fergus, R.: Stochastic video generation with a learned prior. arXiv preprint arXiv:1802.07687 (2018)
5. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: a benchmark. In: CVPR, June 2009
6. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: an evaluation of the state of the art. PAMI **34** (2012)
7. Dosovitskiy, A., et al.: FlowNet: learning optical flow with convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2758–2766 (2015)
8. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)
9. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: FlowNet 2.0: evolution of optical flow estimation with deep networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2 (2017)
10. Jiang, H., Sun, D., Jampani, V., Yang, M.H., Learned-Miller, E., Kautz, J.: Super SloMo: high quality estimation of multiple intermediate frames for video interpolation. arXiv preprint arXiv:1712.00080 (2017)
11. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 694–711. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_43

12. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
13. Leibfried, F., Kushman, N., Hofmann, K.: A deep learning approach for joint video frame and reward prediction in Atari games. arXiv preprint arXiv:1611.07078 (2016)
14. Liang, X., Lee, L., Dai, W., Xing, E.P.: Dual motion gan for future-flow embedded video prediction. In: ICCV (2017)
15. Liu, Z., Yeh, R., Tang, X., Liu, Y., Agarwala, A.: Video frame synthesis using deep voxel flow. In: International Conference on Computer Vision (ICCV), vol. 2 (2017)
16. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
17. Lotter, W., Kreiman, G., Cox, D.: Deep predictive coding networks for video prediction and unsupervised learning. In: ICLR (2014)
18. Lu, C., Hirsch, M., Schölkopf, B.: Flexible spatio-temporal networks for video prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6523–6531 (2017)
19. Luc, P., Neverova, N., Couprie, C., Verbeek, J., LeCun, Y.: Predicting deeper into the future of semantic segmentation. In: Proceedings of International Conference on Computer Vision, ICCV 2017, p. 10 (2017)
20. Mahjourian, R., Wicke, M., Angelova, A.: Geometry-based next frame prediction from monocular video. In: 2017 IEEE Intelligent Vehicles Symposium (IV), pp. 1700–1707. IEEE (2017)
21. Mathieu, M., Couprie, C., LeCun, Y.: Deep multi-scale video prediction beyond mean square error. In: International Conference on Learning Representations (2016)
22. Milletari, F., Navab, N., Ahmadi, S.A.: V-Net: fully convolutional neural networks for volumetric medical image segmentation. In: 2016 Fourth International Conference on 3D Vision (3DV), pp. 565–571. IEEE (2016)
23. Niklaus, S., Mai, L., Liu, F.: Video frame interpolation via adaptive convolution. In: IEEE Conference on Computer Vision and Pattern Recognition (2017)
24. Niklaus, S., Mai, L., Liu, F.: Video frame interpolation via adaptive separable convolution. In: IEEE International Conference on Computer Vision (2017)
25. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill (2016). https://doi.org/10.23915/distill.00003. http://distill.pub/2016/deconv-checkerboard
26. Oliu, M., Selva, J., Escalera, S.: Folded recurrent neural networks for future video prediction. arXiv preprint arXiv:1712.00311 (2017)
27. Paszke, A., et al.: Automatic differentiation in PyTorch (2017)
28. Ranzato, M., Szlam, A., Bruna, J., Mathieu, M., Collobert, R., Chopra, S.: Video (language) modeling: a baseline for generative models of natural videos. arXiv preprint arXiv:1412.6604 (2014)
29. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
30. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
31. Srivastava, N., Mansimov, E., Salakhutdinov, R.: Unsupervised learning of video representations using LSTMs. In: ICML (2015)

32. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. arXiv preprint arXiv:1709.02371 (2017)
33. Van Amersfoort, J., Kannan, A., Ranzato, M., Szlam, A., Tran, D., Chintala, S.: Transformation-based models of video sequences. arXiv preprint arXiv:1701.08435 (2017)
34. Villegas, R., Yang, J., Hong, S., Lin, X., Lee, H.: Decomposing motion and content for natural video sequence prediction. In: ICLR (2017)
35. Vondrick, C., Pirsiavash, H., Torralba, A.: Generating videos with scene dynamics. In: Advances in Neural Information Processing Systems, pp. 613–621 (2016)
36. Vondrick, C., Torralba, A.: Generating the future with adversarial transformers. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2992–3000 (2017)
37. Vukotić, V., Pintea, S.-L., Raymond, C., Gravier, G., van Gemert, J.C.: One-step time-dependent future video frame prediction with a convolutional encoder-decoder neural network. In: Battiato, S., Gallo, G., Schettini, R., Stanco, F. (eds.) ICIAP 2017. LNCS, vol. 10484, pp. 140–151. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68560-1_13
38. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)