Supplementary Material: Swin-∇: Gradient-Based Image Restoration from Image Sequences using Video Swin-Transformers

Monika Kwiatkowski, Simon Matern, Olaf Hellwich
Computer Vision & Remote Sensing
Technische Universität Berlin, Germany
{m.kwiatkowski,s.matern,olaf.hellwich}@tu-berlin.de

1. 2nd Order Derivatives as Convolution

In this section we go into more details about expressing image derivates as convolutional operation.
Let $I(x, y) \in \mathbb{R}$ be an image. The gradient representation $\nabla I(x, y) = (I_x, I_y)(x, y) \in \mathbb{R}^2$ is computed as a convolution with gradient filters f_x and f_y:

$$I_x(x, y) := (I * f_x)(x, y) \tag{1}$$
$$I_y(x, y) := (I * f_y)(x, y) \tag{2}$$

There exists a variety of different gradient filters. In this paper, we consider the kernel defined by finite difference:

$$f_x = \begin{pmatrix} -1 & 1 & 0 \end{pmatrix}$$
$$f_y = \begin{pmatrix} -1 & 1 & 0 \end{pmatrix}^T$$

In order to extract the original image I^* from its gradients, one can solve the Poisson equation as described by Perez *et al*. [5]:

$$\begin{aligned} \Delta I^* &= \mathbf{div} v && (3) \\ &= \frac{\partial I}{\partial x^2} + \frac{\partial^2 I}{\partial^2 y^2} && (4) \\ &= \Delta I && (5) \end{aligned}$$

The original image can be restored using deconvolution of discrete the Laplace operator. There are various ways to approximate derivatives in the discrete domain. In the following we define the second-order derivatives such that they are consistent with the Laplace operator:

$$\Delta I = \frac{\partial I}{\partial x^2} + \frac{\partial^2 I}{\partial^2 y^2} = I * \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{6}$$

The Laplace operator is defined as the sum of second-order derivatives. We can define the second-order derivatives such that they are consistent with the Laplacian:

$$\begin{aligned} \frac{\partial I}{\partial x^2} &:= I_x * f_x^r && (7) \\ &= I * f_x * f_x^r && (8) \\ &= I * \begin{pmatrix} 0 & -1 & 1 \end{pmatrix} * \begin{pmatrix} -1 & 1 & 0 \end{pmatrix} && (9) \\ &= I * \begin{pmatrix} 1 & -2 & 1 \end{pmatrix} && (10) \\ \frac{\partial I}{\partial y^2} &:= I_y * f_y^r && (11) \\ &= I * f_y * f_y^r && (12) \\ &= I * \begin{pmatrix} 0 & -1 & 1 \end{pmatrix}^T * \begin{pmatrix} -1 & 1 & 0 \end{pmatrix}^T && (13) \\ &= I * \begin{pmatrix} 1 & -2 & 1 \end{pmatrix}^T && (14) \end{aligned}$$

The gradient filters are flipped, inverted and convolved again with their respective gradient images. If we compute their sum, they should be equivalent to the Laplacian.

$$\begin{aligned} \frac{\partial I}{\partial y^2} + \frac{\partial I}{\partial y^2} &= I_x * f_x^r + I_y * f_y^r && (15) \\ &= I * f_x * f_x^r + I * f_y * f_y^r && (16) \\ &= I * (f_x * f_x^r + f_y * f_y^r) && (17) \\ &= I * \left(\begin{pmatrix} 1 & -2 & 1 \end{pmatrix} + \begin{pmatrix} 1 & -2 & 1 \end{pmatrix}^T \right) && (18) \\ &= I * \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} && (19) \end{aligned}$$

An image can be reconstructed from its Laplacian image using deconvolution:

$$\begin{aligned} \Delta I &= I * f_\Delta && (20) \\ \Leftrightarrow \mathcal{F}(\Delta I) &= \mathcal{F}(I) \cdot \mathcal{F}(f_\Delta) && (21) \\ \Leftrightarrow I &= \mathcal{F}^{-1}(\mathcal{F}(\Delta I)/\mathcal{F}(f_\Delta)) && (22) \end{aligned}$$

Here $\mathcal{F}$ describes the Fourier transform. Using the convolutional theorem the convolution is described as a multiplication frequency domain.

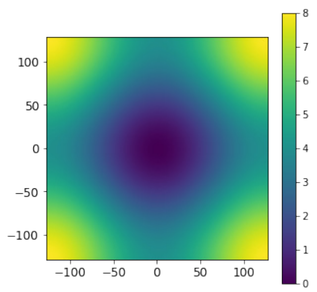


Figure 1. Visualization of the amplitude of the Discrete Fourier Transform of a Laplace filter.

2. Singularities

Equation (22) describes a division in the frequency domain. However, this ignores singularities. Figure 1 visualizes the amplitudes of the Discrete Fourier Transform of a Laplace filter. As one can see the 0-th frequency (the center) has an amplitude of zero. When dividing in the frequency domain we set the 0-th frequency to 0, since the division is not defined there. The 0-th frequency of a signal represents the constant offset of the signal. By setting the value to zero, we implicitly assume that the restored signal is also centered around zero. Thus, the solution described by the deconvolution is only correct up to a constant offset. In our restoration model we use the average pixel intensity over all input images as a heuristic to estimate the offset.

3. Evaluation

In this section we show provide more detailed evaluation result. We used the synthetic rendering to generate additional evaluation datasets D_{Eval}, D_{Light} and D_{Occ}. Figure 2 illustrates samples from the different datasets. D_{Eval} is a test set containing samples that follow the same data distribution as our training data. D_{Light} only contains illumination changes and shadows, the occlusions are made invisible. D_{Occ} has only diffuse illumination and only contains occlusion. Figures 3 to 5 show the performance of our models on each dataset. Figures 3 and 4 have very consistent results. *Swin-Mix* has an overall good performance on D_{Eval} and D_{Light}. The results also show that gradient information consistently improves restoration. *Swin-*∇ has a high SSIM, but worse PSNR and RMSE. The loss of the exact color range affects PSNR and RMSE more than SSIM.

As can be seen in Fig. 5 the results from D_{Occ} deviate significantly. Unsupervised methods perform better and result in perfect restoration. For PSNR we clipped the maximal value at 40, since we get numerically unstable values. Pixel-wise median and RPCA have a PSNR of ∞.

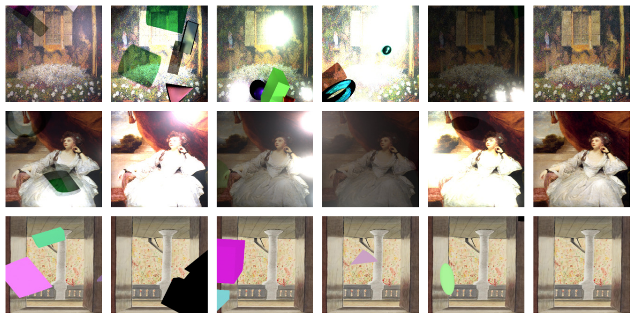


Figure 2. First row show a sample from D_{Eval}. Second row shows samples from D_{Light} and the last row shows D_{Occ}.

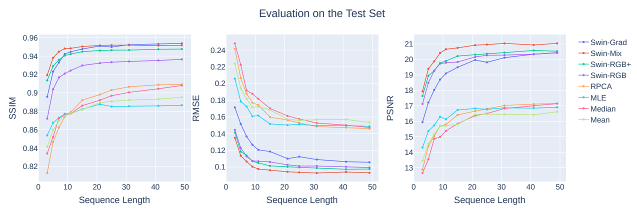


Figure 3. Evaluation results for the test set D_{Eval}

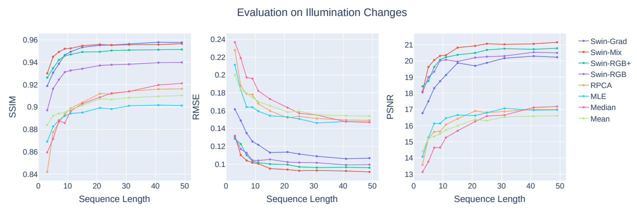


Figure 4. Evaluation results for the dataset containing only illumination changes D_{Light}

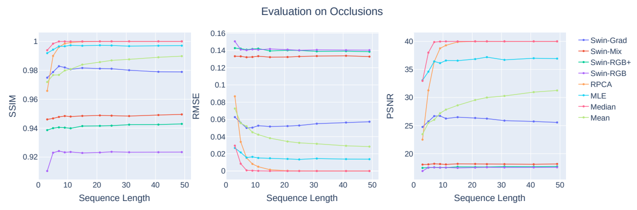


Figure 5. Evaluation results for the dataset containing only occlusion D_{Occ}

4. Ablation Studies

For all models we used the same number of residual blocks and feature maps, except for the corresponding input and output layers. The Video Swin Transformer is parameterized by its token sizes $(T \times H \times W)$ and the attention window $(P \times M \times M)$. We trained all four variations by adjusting these parameters. All models were trained with a batch size of 30 for 100 epochs.

We evaluate the models on D_{Eval}. Section 4 shows that

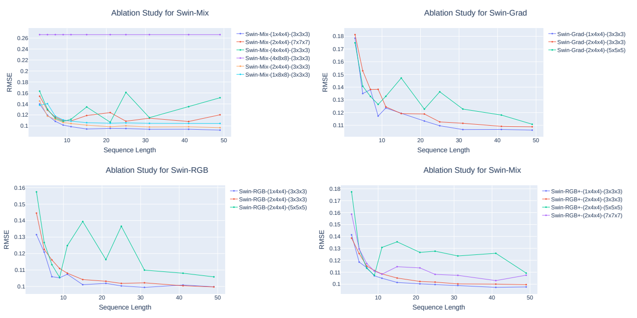


Figure 6. Performance comparison for all four Swin-Transformer variations using various patch sizes and attention windows.

smaller patch sizes and smaller window sizes are consistently better. Overall $(1 \times 4 \times 4)$ is the preferable patch size $(3 \times 3 \times 3)$ the preferable window size.

5. Visualizations

The following section compare restoration results on the evaluation set of the synthetic dataset SIDAR [3]. The results visualize the generalization of the method to new image sequences.
Additionally, we do qualitative comparisons of the same methods for background subtraction. The WALT dataset is used [6]. It contains static time-lapse videos of traffic cameras. The images were sampled at random and contain moving foreground objects and changing illumination.

5.1. Synthetic Results

Figures 7 and 8 visualize the restoration results on several synthetic image sequences. Overall Swin-Grad has consistently better SSIM results, but RMSE/PSNR varies for each sequence. As can be seen in Fig. 7 Swin-Grad has a worse RMSE/PSNR, while the restoration seems the most correct. This effect is likely due to incorrect color estimation. Swin-RGB restores the exact colors better resulting in lower scores. SSIM seems less sensitive to color variation and compares the image content better.

5.2. Real Results

The image restoration methods are also evaluated on real image sequences. Note that the supervised methods were only trained on synthetic image artifacts. Figures 9 to 11 show that Swin-∇ generalizes better than all the other supervised methods using the RGB domain. The results indicate that the other methods overfit to the image distortions of the synthetic dataset. As seen in Fig. 11 Swin-Mix tends to remove the most artifacts, e.g. cars, lights, shadows etc., however as with all RGB models it also darkens the image and the restoration doesn't seem as visually consistent as in Swin-∇.

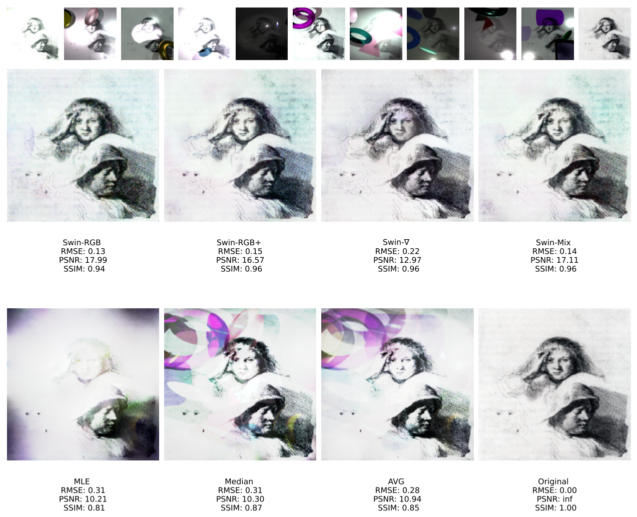


Figure 7. Input sequence and restoration results for a synthetic image sequence.

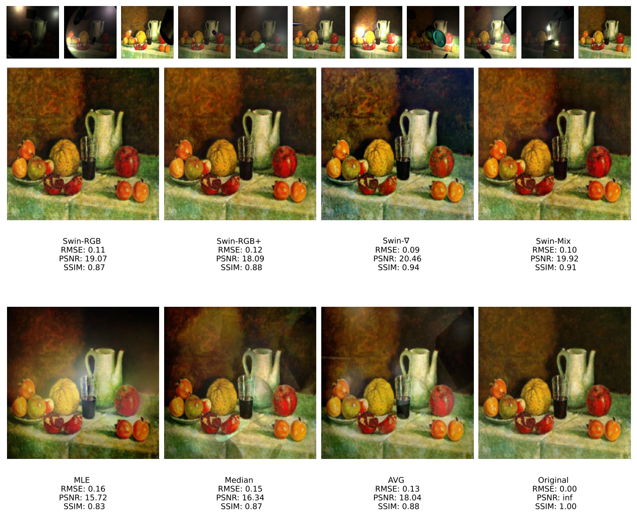


Figure 8. Input sequence and restoration results for a synthetic image sequence.

6. Gradient-Based Image Restoration

6.1. SwinIR3D

To explore the capabilities of gradient-based models for general image restoration we adopt *SwinIR* [4] to our usecase. We name the new architecture *SwinIR3D*. Figure 12 shows the architecture. The only difference to the original implementation is that the 2D Swin transformer is replaced with a 3D Swin transformer and the aggregation of features.

This model is also very similar to our architecture. The main difference is the shallow feature extraction as the input layer and the reconstruction module as the output layer. Otherwise we use the same 3D Swin transformer configura-

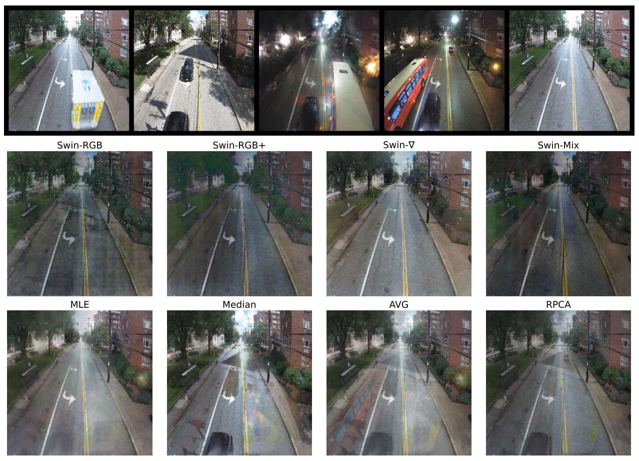


Figure 9. Input sequence and restoration results for a real time-lapse sequence.

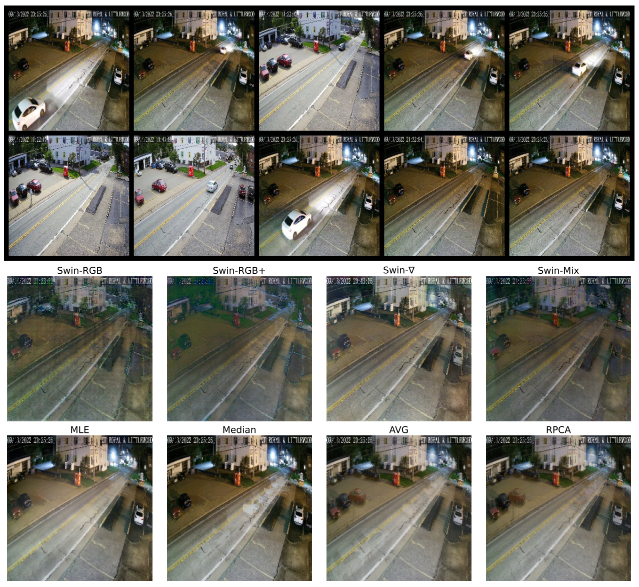


Figure 10. Input sequence and restoration results for a real time-lapse sequence.

tion of $1 \times 4 \times 4$ patches and a windows size of $(3 \times 3 \times 3)$

6.2. Image Distortions

We train and evaluate *SwinIR3D* on a various image restoration tasks. Given any input image we generate a sequence of distortions. Using the library imgaug [2] we can generate various distortions, such as Gaussian noise, blurring, JPEG compression and color changes. Figure 13 show the effect of the image augmentations. Our model should learn to aggregate information across multiple images. That's why we also introduce variation in each distortion. For example, we use various methods for blurring, such as motion blur, Gaussian blur and defocus. We use various levels of JPEG compression. To simulate color changes we multiply or add random values to the image intensity. In addition we also generate random masks before applying the same transformation, this creates effects similar to shadows and specular highlights.

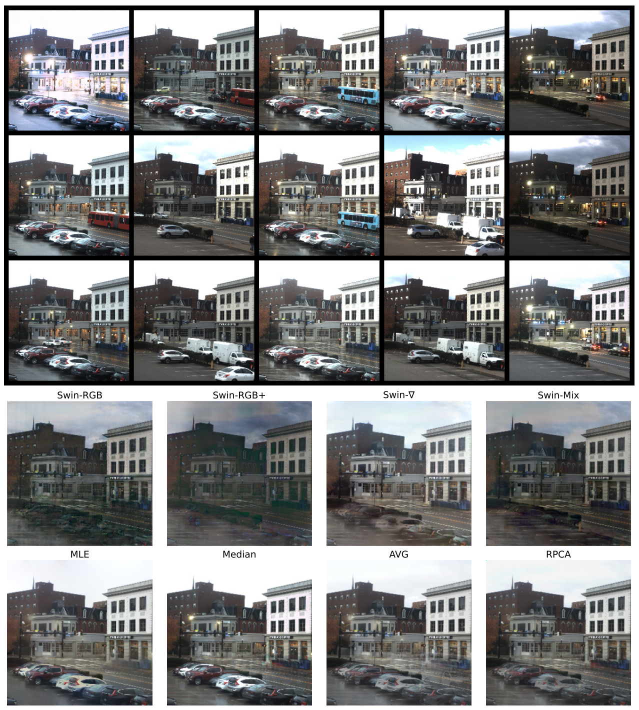


Figure 11. Input sequence and restoration results for a real time-lapse sequence.

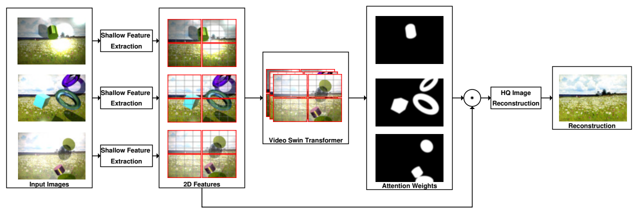
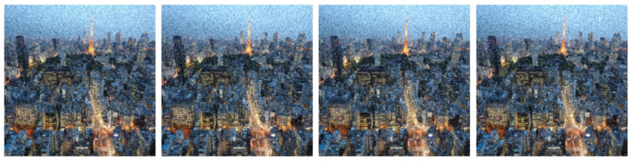


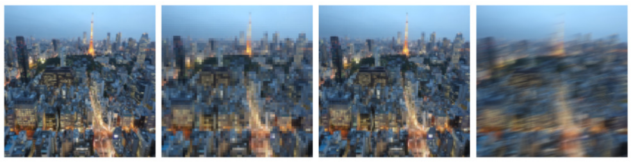
Figure 12. SwinIR3D Architecture

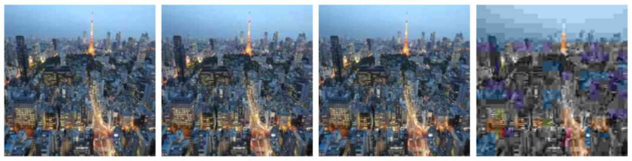
6.3. Training & Evaluation

We create four variations of *SwinIR3D* and train them on the DIV2K dataset [1]. We train each model on each image restoration task. Each input image is augmented into a sequence of 8 distorted images. We use a batch size of 14 and train for 500 epochs. We use an Adam optimizer with a learning rate of $\lambda_{lr} = 10^{-3}$.

For evaluation we use the same testing set (Set5 + Set14 + BSD100 + Urban100 + Manga109) as described in SwinIR [4].


(a) Gaussian Noise


(b) Blur


(c) JPEG compression

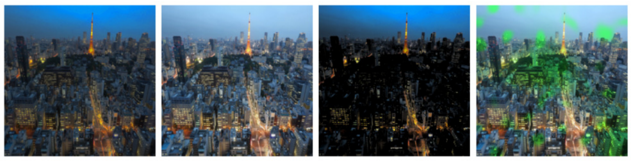

(d) Color changes

Figure 13. An illustration of various image distortions.

References

[1] Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (July 2017) 4

[2] Jung, A.B., Wada, K., Crall, J., Tanaka, S., Graving, J., Reinders, C., Yadav, S., Banerjee, J., Vecsei, G., Kraft, A., Rui, Z., Borovec, J., Vallentin, C., Zhydenko, S., Pfeiffer, K., Cook, B., Fernández, I., De Rainville, F.M., Weng, C.H., Ayala-Acevedo, A., Meudec, R., Laporte, M., et al.: imgaug. https://github.com/aleju/imgaug (2020), online; accessed 01-Feb-2020 4

[3] Kwiatkowski., M., Matern., S., Hellwich., O.: Sidar: Synthetic image dataset for alignment & restoration. In: Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: VISAPP. pp. 175–189. INSTICC, SciTePress (2024). https://doi.org/10.5220/0012391400003660 3

[4] Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L., Timofte, R.: Swinir: Image restoration using swin transformer. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1833–1844 (2021) 3, 4

[5] Pérez, P., Gangnet, M., Blake, A.: Poisson image editing. ACM Transactions on Graphics (TOG) **22**(3), 313–318 (2003) 1

[6] Reddy, N.D., Tamburo, R., Narasimhan, S.G.: Walt: Watch and learn 2d amodal representation from time-lapse imagery. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9356–9366 (June 2022) 3