Summary: Deep Learning for Joint Video Denoising and Debayering

Laurent Gudemann
Stuttgart Media University, ARRI Munich

Abstract

This summary presents the main contributions of the author's master's thesis on joint video denoising and debayering using deep learning. Several convolutional and transformer-based neural network architectures are proposed that perform these tasks on raw data from an ARRI cinema camera. It is shown that a joint model which is trained end-to-end is more computationally efficient and produces better quality images than separately trained models for denoising and debayering. However, the joint models also suffer from temporal inconsistency when the individually processed frames are viewed in sequence. To address this issue, an efficient temporal fusion mechanism is incorporated that improves the spatiotemporal models' performance and consistency.

A novel method for generating clean ground truth from any raw video recording is also presented, which overcomes the limitations of existing datasets and methods for supervised learning. The effectiveness of the proposed models and the training dataset is demonstrated on real footage and compared with industry-standard methods.

Supplementary video material is available at this URL: https://laurentgudemann.github.io/joint-video-denoising-and-debayering-thesis/

1. Introduction

This paper investigates how deep learning can improve image denoising and debayering in a digital motion picture camera's image processing pipeline. Denoising and debayering are essential for producing high-quality images, but they are also challenging tasks that involve trade-offs between accuracy, efficiency, and flexibility. Traditional methods based on hand-crafted filters or optimization techniques often suffer from limitations such as color artifacts, loss of fine details, or high computational cost.

In contrast, deep learning can extract complex patterns from large amounts of data, learn functions that are more sophisticated than hand-crafted methods, and exploit the capabilities of modern hardware such as GPU and AI accelerators. These advantages make deep learning a promising approach for denoising and debayering. In fact, image denoising has been one of many computer vision tasks that have benefited from the remarkable success of convolutional neural networks (CNNs).

As part of this summary, the task of removing noise from images will be introduced as a generalized image restoration problem. Afterward, several related works will be examined which utilize a U-shaped CNN architecture design to extract features at multiple scales. Through the use of dense residual connections [18], depthwise separable convolutions [12, 26], and attention mechanisms [9, 14, 25, 41], these publications provide a selection of strategies that will later be employed to find an efficient architecture for denoising raw image data.

Like denoising, the image debayering problem has been tackled by an increasing number of deep learning researchers. In many ways, debayering can be considered an image restoration problem that is not dissimilar from denoising and super-resolution [39]. Thus, it is less common to find recent publications that focus solely on debayering. There is, however, a silver lining: Because these tasks are somewhat similar, recent advancements in image restoration models can likely be leveraged to design an efficient debayering model.

1.1. Research Focus and Contributions

The thesis explores how deep learning can be used for image denoising and debayering in a digital motion picture camera's image processing pipeline. It addresses the following aspects of this problem:

> How can deep learning models be designed and trained to achieve high-quality image reconstruction on real-world images captured by digital motion picture cameras?
>
> Can such a model produce high-quality outputs with limited computational budgets?

The goal is to answer these questions by applying existing techniques and proposing novel methods for denoising and debayering using deep learning, demonstrating their effectiveness and efficiency.

The main contributions can be summarized as follows:

1. The first high dynamic range raw video dataset suitable for supervised learning of spatiotemporal denoising and debayering is proposed.
2. A convolutional neural network architecture is designed that combines access to spatiotemporal information with attention mechanisms to perform joint denoising and debayering in a single step.
3. A training procedure is designed that uses hyperparameter optimization and image brightness augmentation to improve the robustness and generalization of the models.

The practical applicability of the proposed methods is demonstrated on real-world images captured by digital motion picture cameras.

2. Related Work

2.1. Single Image Denoising

The use of CNNs in image denoising was first explored in a paper published in 2016 [42]. In it, the authors propose a feed-forward denoising CNN (DnCNN) that achieved state-of-the-art (SOTA) denoising performance and efficiency for various image restoration tasks. Rather than learning the denoising function f_{Dn} directly, the authors employ a global residual learning approach, in which the model f_θ is tasked to estimate the noise from the signal:

$$f_\theta(x) = x + f_{res}(x, \theta) \tag{1}$$

x and y denote the noisy observation and the clean target respectively. The prediction $\hat{y}$ of the model f_θ is passed to a loss function ℓ subject to minimization. It was shown that that DnCNN outperformed SOTA methods on several image restoration tasks, such as Gaussian denoising, single image super-resolution, and JPEG image deblocking [42].

2.1.1 U-Nets

Since the publication of DnCNN, countless other learning-based methods for image restoration have been developed. Many of the current SOTA denoising architectures design the residual path f_{res} in a U-shape with skip connections [9, 41]. This inter-block design aims to better extract features at multiple scales and increase the spatial receptive field of the model.

To better understand how U-Nets are employed in recent SOTA architectures, the high-level inter-block design is first introduced decoupled from the specific lower-level implementation. This way, the U-Net can later be viewed as an interchangeable module. At large, it consists of these fundamental building blocks:

Basic Block: The basic block is inserted at each level during encoding and decoding. It preserves the shape of the features, allowing it to be stacked sequentially.

Input and Output Block: Because the number of features in the input and output data does not necessarily match the chosen number of base features C_0, the data needs to be transformed accordingly. This is the purpose of the input and output blocks.

Down and Up Block: As part of the multi-stage U-Net, the spatial dimensions of the data need to be reduced or increased to move between levels. The down block usually doubles the channel dimension and halves each of the spatial dimensions. The opposite is true for the up block.

Skip Connection: The output of a series of basic blocks during encoding is stored in memory and later merged with the upsampled features during decoding. This can be done by either concatenating the features and then reducing their channel dimension [41] or by element-wise-adding the upsampled features to the encoder features [9]. Both merging approaches maintain the channel dimension.

2.1.2 Residual and Dense Layer Connections

As the size of neural networks increases, it becomes more difficult to train them in a fast and stable manner. ResNet, short for Residual Network, is a type of artificial neural network that reformulates the layers as learning residual functions with reference to the layer inputs [19]. While DnCNN used only a single residual connection between the network input and output, ResNet's residuals are employed every few layers throughout the architecture.

Dense connections, on the other hand, are built on the idea of feature reuse: By concatenating the features from all previous layers to the input of a late layer, more information can be accessed without spending computational resources and learnable parameters on transporting the information forward [21]. This design has been shown to improve computational efficiency over ResNets on a range of computer vision tasks [21]. The concepts of U-Nets, residual connections, and dense connections were later combined in a denoising architecture called RDUNet [18].

2.1.3 ConvNeXt

In search of the most efficient intra-block design, researchers at Facebook recently published their work on a computer vision CNN called ConvNeXt [26]. At the heart of their proposed architecture are so-called depthwise separable convolutions. Rather than performing 2-D convolution as a weighted sum of neighboring pixels in both spatial and channel dimensions, depthwise separable convolutions first convolve only in the spatial dimension followed by a pointwise regular convolution over only the channel dimension. Compared to the regular convolution with the

same kernel size D_K and N output channels, the depthwise separable convolution's cost is reduced by a factor of $\frac{1}{N} + \frac{1}{D_K^2}$ [20]. For large kernel sizes and output channels, the theoretical complexity of the depthwise approach can be orders of magnitude lower. ConvNeXt's usage of depthwise separable convolutions and an inverted bottleneck is not domain specific to image classification, making it potentially useful for image restoration. In fact, the ConvNeXt basic block has been shown to perform well at denoising tasks when integrated into a U-shaped architecture [12].

2.1.4 Transformers for Computer Vision

Since their introduction in the landmark paper "Attention is all you need" from 2017 [38], Transformer models have seen widespread success in various machine learning domains. Unsurprisingly, significant efforts have been made to leverage attention mechanisms for computer vision.

The goal of self-attention is to model long-range dependencies between input tokens in a computationally efficient and parallelizable manner. Intuitively, self-attention can be seen as a process in which a relevancy score is computed between all combinations of token-pairs [16]. The output is then a weighted average of the inputs according to these scores. Concretely, the inputs are first linearly projected with learned matrices into a set of queries and key-value pairs. The attention is then computed with a dot-product:

$$\text{Attention}(Q, K, V) := \text{softmax}\left(\frac{QK^\intercal}{\sqrt{d_k}}\right)V \qquad (2)$$

Compared with convolution, self-attention connects the vectors at all positions rather than only connecting those within the size of the kernel. To enlarge the receptive field, convolutional and recurrent layers need to be applied several times sequentially, which further increases the length that signals in the network have to traverse. Shorter path lengths make it easier for the model to learn dependencies between distant positions and thus, the attention mechanism with its direct paths can better capture the global context [38].

To transfer the architecture of the Transformer from 1-D sequential inputs, to 2-D images, the early pioneering work on the Vision Transformer (ViT) [14] closely adheres to the original Transformer architecture while making some small changes to adapt it to the image classification task. The image is first divided into N non-overlapping patches with size $C \times P \times P$ and each patch is then flattened into a vector The resulting patch matrix with shape $N \times (C \cdot P^2)$ is treated as a sequence of N tokens, whose positionally encoded linear projections are passed to the Transformer's encoder block for self-attention and refinement. Unfortunately, the computation of such an approach is prohibitively expensive for high-resolution images as its time and memory complexity

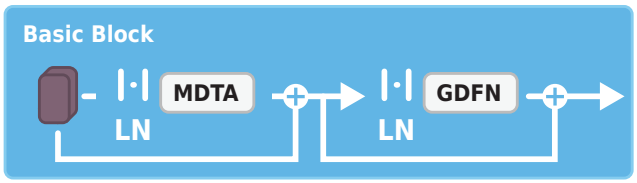


Figure 1. Restormer's basic block

$O(W^2H^2)$ rises quadratically with the number of pixels in the input image [41].

One strategy to limit the complexity growth is to compute attention locally rather than globally. Swin Transformers [25] do this by first dividing the image into small patches of size $C \times 4 \times 4$ which are treated as image tokens. Rather than computing self-attention between all tokens, the image is further divided into windows, each containing 7×7 tokens. Self-attention is then performed only within each window. Consequently, the computational cost of the attention rises only linearly with the spatial resolution and quadratically with the size of the context window [25]. With a limited compute budget, a small context window is hypothesized to yield little benefit over regular convolutions [41].

Restormer

In an effort to more efficiently capture the global context of an image, a channel-attention mechanism was introduced by the Restormer architecture. The authors propose a *Multi-Dconv Head Transposed Attention* module, or MDTA for short, which "computes cross-covariance across channels to generate an attention map" [41]. This module first projects the block's input into a set of queries, keys and values using a depthwise separable convolution. Then, a global channel attention matrix A of shape $C \times C$ is computed from the dot product of the queries and keys, which is then matrix-multiplied by each pixel-vector in V. For a visual illustration, please refer to Fig. 2.

Formally, the computation of the channel attention can be described as follows:

$$\begin{gathered}\text{Attention}(\hat{Q}, \hat{K}, V_{flat}) = V_{flat} \cdot \overbrace{\text{softmax}\left(\alpha \cdot \hat{Q}\hat{K}\right)}^{A} \\ \text{with} \\ \hat{Q} := \frac{Q_{flat}}{|Q_{flat}|} \\ \hat{K} := \frac{K_{flat}^\intercal}{|K_{flat}|}\end{gathered} \qquad (3)$$

where Q_{flat}, K_{flat}, and V_{flat} are matrices containing query, key, and value features that have been flattened along the

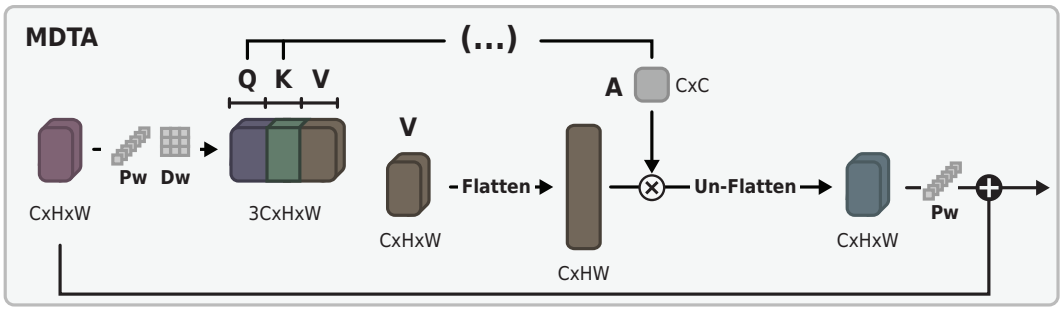


Figure 2. MDTA block illustrated with a single attention head for simplicity

spatial dimensions while preserving the number of channels. The major difference between the Restormer's attention mechanism and the Swin-Transformer is the former's ability to capture a global context. The queries and keys reduce information from all pixels into a single-channel attention matrix, that is then used to weight the values across the channel dimension at each location in the image. The complexity of this approach rises only linearly with the number of pixels in the input image and quadratically with the number of channels, rather than quadratically with the size of the context window.

The second component of the Restormer's basic block is the GDFN module. In it, the features are split in half; the second half is activated with a GeLU function and element-wise multiplied with the first half. The intuition here is that the GeLU-activated features can control the information flow in a gate-like manner, which has been shown to improve the efficiency of the model [41]. By integrating the MDTA attention mechanism and GDFN gating mechanism into a basic block which is then arranged in a U-shaped architecture, the authors demonstrate SOTA performance on denoising benchmarks.

Test-time Local Conversion

The inference domain is different from the training domain - not only because the training data in this work is degraded synthetically but also because the image size during inference is larger. This causes a training-inference gap for the Restormer, whose channel-attention mechanism reduces global information into a single channel-attention matrix. While the original Restormer publication does not take this effect into account, the problem was discovered by Chu *et al*. [10] in a later publication. The authors propose a method called test-time local conversion (TLC) which selectively converts global operations into local image regions to match the size of the training data. While the local conversion itself has a negligible performance penalty, it was found to render visible stitching artifacts at attention boundaries. Applying the method to the joint Restormer models only produced artifact-free results once a more computationally expensive blended attention conversion was implemented. A comparison of local conversion methods is given in Tab. 1.

	Global Attention	TLC	Vectorized Local Attention	Blended Attention
Runtime	409 ms	2442 ms	418 ms	1216 ms
Memory	11.45 GiB	11.5 GiB	11.721 GiB	29.83 GiB

Table 1. Resource usage of various channel attention implementations.

NAFNet: A simplified Restormer that does not require local conversion

The performance penalty of local conversion of the Restormer makes it undesirable for GPU inference. Thus, alternative transformer architectures that could be candidates for a real-time model were investigated next. Conveniently, the authors of TLC later published their work on a simplified image restoration transformer called NAFNet [9]. Its channel attention mechanism is invariant to the input dimensions and therefore does not require TLC.

2.2. Temporal Fusion Mechanisms

The previously introduced denoising architectures were all designed with single-image denoising in mind. Therefore, the latent noise-free image had to be inferred from one noisy observation only. Analogous to traditional denoising approaches, it can be argued that these denoising models leverage self-similarity, local low-level features such as edges, and memorized image priors to perform this task [36]. Video denoising methods, on the other hand, can make use of temporal redundancies [29] to better remove noise from images that do not exhibit self-similarity in the spatial dimensions. This section reviews four main categories of temporal fusion for deep video denoising: sliding window, MIMO, recurrent, and temporal shift techniques [31].

Sliding Window

The sliding-window method is the simplest way to design a spatiotemporal architecture: For each frame whose noise-free latent image is inferred, both the noisy frame and its neighbors are fed into the model. This technique was popularized by DVDnet [34] and FastDVDNet [35], achieving SOTA performance on video denoising at the time.

MIMO

The efficiency of sliding-window-based techniques is not ideal, as redundant computations are performed to extract the features from a frame each time it is fed into the network [31]. Multi-input multi-output (MIMO) methods, as their name suggests, take multiple contiguous frames as inputs and output their respective denoised predictions. This framework can avoid redundant computations by only feeding in each frame once. One major disadvantage of the MIMO scheme is a performance drop at the edges of the temporal window: Because the temporal receptive field is highly asymmetrical at the beginning and end, less recent information is available. This is particularly apparent when a long video is first split into multiple MIMO windows and spliced together afterward, resulting in cyclic changes in the denoising quality [31].

Recurrent Neural Networks

Recently, recurrent frameworks have been applied to achieve SOTA performance in video denoising [7,37]. This temporal fusion method operates in a uni-directional manner, meaning that the current frame has access to information from previous inference steps. In principle, this results in no latency increase when compared to a spatial model, making it suitable for video streaming and image processing pipeline applications.

BSVD

BSVD, which stands for bi-directional streaming video denoising, is a temporal fusion framework from a 2022 paper that promises to solve the problems of all aforementioned temporal fusion strategies [31]. It combines the propagation of features introduced by the temporal shift module (TSM) with an efficient buffer-based inference scheme, achieving bi-directional temporal fusion with a finite receptive field and without cyclic performance drops. The TSM mechanism allows a conventional CNN to access features from a neighboring time step. It "shifts part of the channels along the temporal dimension" [24] allowing temporal information to be exchanged deep in the network architecture. At inference-time, features from the previous inference step can be stored in memory and fused with current features, allowing the current output can be computed with no additional latency and little memory cost [24]. BSVD extends this bufffer-based inference scheme to the bi-directional temporal shift [31]. By delaying the output of the current frame, temporal fusion can be performed in both directions.

2.3. Raw Demosaicing

Like denoising, the image debayering problem has been tackled by an increasing number of deep learning researchers. In many ways, debayering can be considered an image restoration problem that is not dissimilar from denoising and super-resolution [39].

The idea of using CNNs for debayering was first introduced in a 2016 paper by Gharbi *et al*. [17]. They train their model on a large dataset of difficult-to-debayer RGB images by synthetically removing color information from the target image according to the Bayer pattern and performing gradient descent on a loss function between the reconstructed model output and the target image. Their proposed "Demosaicnet" architecture is, at its core, a standard feed-forward CNN with a three-channel masked mosaic residual. The authors also investigated the joint task by adding Gaussian noise to the Bayer pattern input and found that the model can learn to remove this noise. The joint denoising and debayering model achieves competitive results with other joint non-CNN methods at a computational cost that is orders of magnitude lower.

Since then, several other papers have improved upon the architecture design. Both [32] and [39] use residual dense blocks for improved efficiency, demonstrating that findings from tangential image restoration and computer vision research can be leveraged for image debayering.

2.4. Existing Datasets

Early papers for data-driven denoising and debayering adopted a supervised learning approach, where the model's objective is to reproduce the known noise-free RGB target image from a degraded input. However, acquiring the degraded and ground truth image pair is far from trivial - especially with video data. Some recent papers have therefore adopted a self-supervised learning approach, where only a single degraded image is needed. This simplifies the data collection process and reduces the domain gap between the training and testing data. However, these self-supervised methods require significant effort in designing the network architecture and training procedure.

Dewil *et al*. [12] compared the two approaches quantitatively and qualitatively on real noisy images. They showed that a realistic noise degradation model can be used to generate synthetic noisy images from clean raw data without compromising the performance, as long as the noise model's parameters match the sensor's noise characteristics. Their results suggest that unrealistic modeling of clean raw

data affects the quality of the results more than unrealistic modeling of noise.

To better understand why modeling clean raw video data is difficult and imprecise, the following subsections each examine a desideratum for such a dataset and discuss, how existing dataset approaches fall short for supervised training of joint video denoising and debayering.

Noise-Free Ground Truth

A crucial requirement for a clean ground truth image is a low noise level. In the literature, three main approaches have been proposed to obtain such images:

1. **Unprocessing from sRGB:** This method uses a clean sRGB image as the ground truth and synthetically adds noise to create the degraded input image. This way, any color image dataset can be utilized [2, 39]. However, this approach has several drawbacks. The sRGB images have a different distribution from the raw sensor data, and an inverse image processing pipeline needs to be applied to approximate the raw sensor domain [6]. This inverse is an ill-posed problem, as some steps in the processing pipeline are irreversible due to quantization and compression. Moreover, the source images must have much lower noise than the target camera for which the model is trained. Since the ARRI ALEXA 35 camera used in this thesis has a state-of-the-art dynamic range [28], no existing images from other cameras can be inverted to have lower noise relative to the peak signal of the ALEXA 35's sensor.

2. **Darkening:** [30] and [8] photograph a set of two images of a static scene, once with a long and once with a short shutter speed. The long exposure image is brighter but can be darkened to the same exposure of the dark image, yielding less noise than the latter.

3. **Averaging:** [1] and [40] obtain a clean image by averaging multiple noisy images of a static scene. Lehtinen *et al*. [23] provide a statistical proof that training on a pair of noisy images decomposes to the same minimization problem as training on a clean and noisy image pair. This enables a model to be trained on fewer images of a static scene and simplifies the data collection process.

Full RGB Ground Truth

If a model is intended to be trained on the debayering task, having a noise-free raw ground truth does not suffice. In addition, each image must have the full RGB color information at each sensor pixel location. Among the denoising approaches 1.-3., only the sRGB unprocessing method meets this requirement.

However, this method relies on the assumption that the sRGB images have well-sampled RGB information, which may not hold. Khasabi *et al*. [22] note that these images are often derived from an image processing pipeline that involves a demosaicing step. They suggest a raw downscaling scheme to combine a 4x4 block into a single well-sampled RGB pixel.

Another strategy to obtain full RGB resolution of static scenes is to capture multiple images with different color filters at each pixel. This can be, for example, be done with sensor pixel shifting [32] or or a color wheel [4]. This strategy is, however, not desirable for this thesis as it limits the dataset to static scenes.

Realistic Motion and Motion Blur

The only denoising dataset that consists of raw image sequences is a stop-motion dataset by Yue *et al*. [40]. To obtain a noisy-clean image pair, the authors adopted the averaging approach of static images. Motion is achieved by adjusting the scene frame by frame. The scene is therefore entirely static during the exposure, devoid of any motion blur. Another problem with this approach is that the content of the image is limited to a studio setting with controlled lighting and objects small enough to be hand-animated.

Dewil *et al*. [13] use real video sequences and the inverse-pipeline approach to train a temporal video denoising model. While these video sequences contain realistic motion, this method suffers from the same drawbacks as the sRGB still image unprocessing approach.

3. Method

3.1. Proposed Dataset

As part of this work, a novel dataset for video denoising and demosaicing is presented that addresses the limitations of existing datasets. Concretely, existing datasets either do not provide well-sampled RGB video sequences, do not model the spatial characteristics of the sensor, or do not contain realistic motion and motion blur. To overcome these challenges, a dataset is proposed that combines multiple strategies. The creation of this dataset can be summarized in the following five steps:

1. Real raw video sequences containing realistic motion and motion blur are captured by an ARRI ALEXA 35 cinema camera.

2. Well-sampled RGB video sequences are obtained from raw sensor data by downsampling.

3. A camera-specific spatial frequency response simulation [33] accounts for the effects of the lens, optical low pass filter, and photosensitive pixel area.

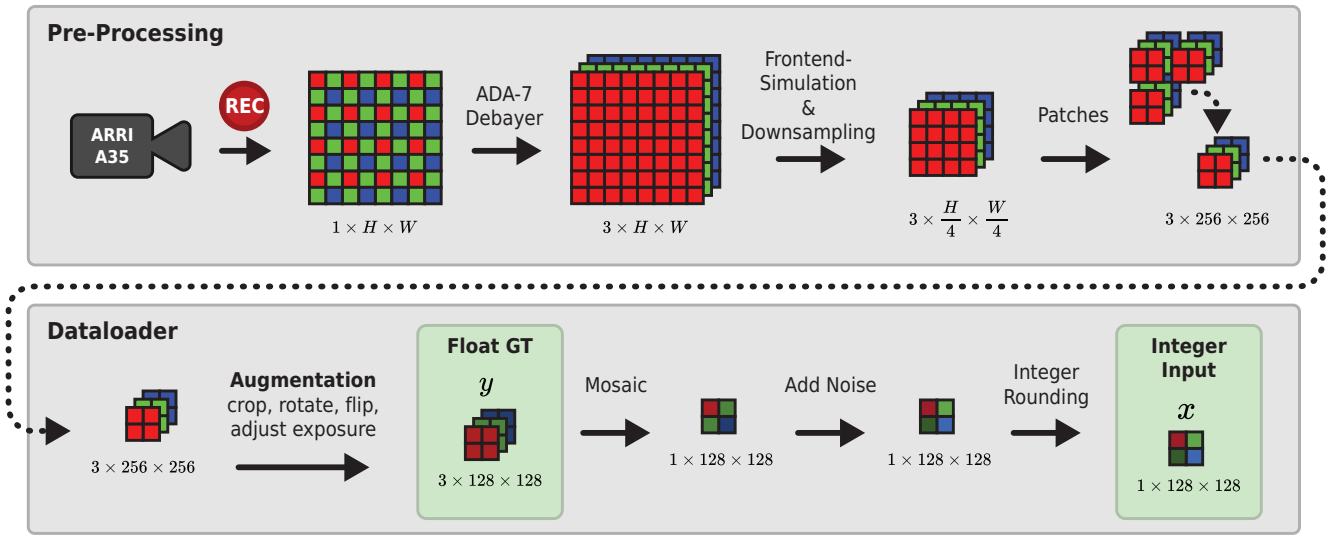


Figure 3. Overview of the proposed dataset processing steps. The actions in the first row are performed once in a pre-processing phase while the actions in the second row are performed ad-hoc during data loading.

4. A random gain factor augmentation during training increases the signal-to-noise ratio of the clean ground truth (GT) at dim exposures.

5. Sensor-specific signal-dependent noise is added to the clean ground truth to synthesize degraded raw data.

3.1.1 Downsampling with a Frontend Simulation

Inspired by Khasabi *et al.*'s approach [22], the fundamental strategy for acquiring full RGB from raw data in this work involves filtered downsampling. Rather than treating each sensor pixel as a point whose value is estimated with interpolation, the proposed method instead treats it as an optical simulation problem. By filtering the image by the spatial response of the camera system, the image's spatial characteristics (e.g. the sharpness of edges) are accounted for.

Concretely, a set of equations from Schoberl *et al.* [33] is employed to simulate the spatial frequency response of the lens, OLPF, and the photosensitive area of each Bayer pixel. The raw data from an ARRI ALEXA 35 can be debayered, filtered with the simulation, and resampled to approximately reconstruct full RGB information as if the image had been recorded with a smaller magnification. Experiments on synthetic Gaussian test images demonstrate that the proposed method not only reconstructs RGB information but also reduces noise significantly (Tab. 2).

During filming, the sequences are exposed brightly to maximize the signal to noise ratio. At training time, the sequences can then be multiplicatively brightened or darkened. By removing color information according to the Bayer pattern and adding heteroskedastic Gaussian noise to the clean ground truth according to the ALEXA 35's sensor characteristics, the clean RGB ground truth can be synthetically degraded in realistic manner. A numerical analysis of the remaining level of noise in the clean ground truth and the synthetic degradation was performed to demonstrate, that the proposed dataset provides realistic clean-noisy RGB-Bayer image pairs.

3.1.2 Dataset Contents

The real raw video sequences are comprised of two sources: newly recorded data and archival test footage. The reader is urged to watch the accompanying supplementary video titled `1_dataset_examples` for visual examples from the real dataset.

The archival test footage consists of ALEXA 35 raw recordings with no denoising pre-applied, which were obtained from an internal database. These recordings were hand-selected to ensure that the recordings have low noise levels, realistic motion blur, and rich textures.

The newly recorded data was obtained specifically to increase the diversity of the training data and to include more challenging scenes. An ARRI ALEXA 35 camera with noise reduction disabled and an ARRI Signature Zoom 24-75mm lens were used for all recordings of to-be-downsampled patches. Each scene was framed with a Signature Prime 18mm lens first to find a reasonable image size and then recorded at 4 times the focal length with the zoom lens. The recorded sequences contain various types of camera movement and lighting conditions. The scenes include printed charts, texture samples, office interior settings, and LED display recordings from a virtual production stage.

Method	σ_r	σ_g	σ_b	Reduction
Sampled Noise	0.1	0.1	0.1	1 (None)
SIDD Averaging [1]	0.0082	0.0082	0.0082	12.2
Filtered Downsampling 4x	0.010	0.0071	0.010	10.0
Weighted Pooling 4x [22]	0.063	0.035	0.063	1.6
Uniform Pooling 3x [22]	$[0.05, 0.1]$	$[0.044, 0.05]$	$[0.05, 0.1]$	1 (None)
Uniform Pooling 5x [22]	$[0.041, 0.05]$	$[0.028, 0.029]$	$[0.041, 0.05]$	2

Table 2. Comparison between the proposed filtered downsampling and weighted pooling with the same downsampling factor of 4. σ_r, σ_g, and σ_b denote the standard deviation in the red, green, and blue channels respectively.

3.2. Model Architectures

The model architectures are based on several concepts from related works. The main idea is to leverage the design strategy of U-shaped CNNs, which have shown remarkable results in various image restoration tasks. For the specific implementation of the U-Nets basic block, the RDUNet, The ConvNeXt Block, and the Restormer block were initially chosen.

The model architectures are divided into two categories: sequential models and joint models. Sequential models perform denoising and debayering in two discrete steps, using separately trained models for each task. Chained together, their reproduction of a clean, RGB image will be compared to a joint model that performs both tasks and is trained end-to-end. Finally, a temporal fusion scheme is incorporated to enhance the joint model with temporal information from multiple frames

3.3. Spatial Denoising Architecture

The denoising model constitutes the first step in the sequential denoising and debayering pipeline. Its goal is to reconstruct a noise-free Bayer pattern image from a single noisy observation. For more stable and faster training, all denoising architectures make use of global residual learning, where the input is added to the model's output. A U-shaped inter-block design is chosen for the residual, whose inputs and outputs are unshuffled 4-channel RGGB Bayer data. To find an architecture well-suited for the denoising task, experiments will be performed that test the efficiency of several basic blocks, whose results will later be shown in Sec. 4.

3.4. Spatial Debayering Architecture

The debayering architecture is based on the idea of method noise removal [12]. Concretely, this means that the model is trained by first applying a bilinear debayering method to a clean Bayer input, which produces a clean RGB image with poorly interpolated colors. Then, the loss between the model output and the clean full RGB ground truth is minimized.

Concretely, the architecture is designed to only predict color information at unsampled locations and preserve the information at sampled locations. This way, the model only learns to fill in the missing color values. The input of this "early debayering" architecture is un-shuffled debayered data of shape $\mathbb{R}^{12\times\frac{H}{2}\times\frac{W}{2}}$. This introduces an inductive bias, which aides the model in identifying whether a given pixel was properly sampled or was interpolated.. The output shape of the U-Net is $\in \mathbb{R}^{8\times\frac{H}{2}\times\frac{W}{2}}$, where each channel corresponds to an unsampled color channel with a stride of (2,2).

For an alternative "late debayering" architecture, a nearest-neighbor method is used instead of debayering with an algorithm that requires compute resources. This way, the model does not have to reverse-engineer the debayering method and is not biased to an existing debayering method. Both the "early" and "late debayering" approaches only describe how the data is handled before the input block and after the output block. Therefore, any of the U-Net-based architectures can be inserted in between.

3.5. Joint Spatial Denoising and Debayering Architecture

The joint spatial architecture is very similar to the spatial debayering architecture, to which two changes were made:

The first change lies in the domain of the input and target data. Rather than training from clean Bayer inputs to predict clean RGB outputs, the joint model will receive noisy Bayer inputs and be trained to predict clean RGB outputs. This way, the model learns to remove both sensor noise and debayering method noise from the input data.

The second change lies in the output domain of the global residual. In the debayering architecture, there was no need to predict the residual at sampled locations, since they already matched the ground truth. Now that the inputs are noisy, the pixels at sampled locations need to be restored as well. This results in a global residual of shape $12 \times \frac{H}{2} \times \frac{W}{2}$ which can then be shuffled into a $3 \times H \times W$ RGB residual.

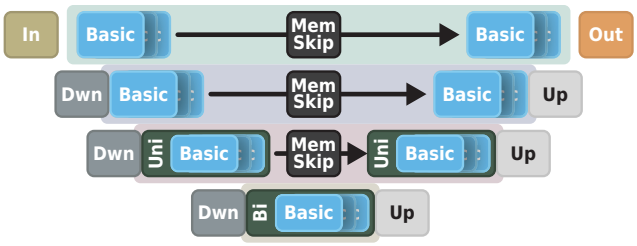


Figure 4. Simplified schematic of a U-Net with asymmetrical BSVD feature fusion. In this example, there are $N_{\text{bi}} = 1$ bi-directional and $N_{\text{uni}} = 2$ uni-directional buffer blocks, resulting in a latency of 1 frame and a temporal receptive field of 5 frames.

3.6. Joint Temporal Architecture

To enrich the CNN with temporal information, BSVD is the temporla fusion method of choice as its receptive field only extends to a fixed number of frames into the past and future [31]. This is desirable, as the inference of a long video sequence can be initiated from an arbitrary starting time. For post-production applications, the BSVD framework would not require pre-processing the entire sequence. Also, if some latency can be afforded, frames from the future can be utilized. BSVD avoids redundant computations by only buffering features near the bottleneck of a U-Net, giving the model the ability to first compress the features to a more information-dense representation which can then be accessed by past and future frames.

However, BSVD has a significant limitation: The latency increases linearly with the number of bi-directional buffer blocks N_{bi}, as they require future frames to be available. If such a model was used as part of an in-camera image processing pipeline, the image presented to the camera operator would be delayed by N_{bi} frames. Recurrent networks, on the other hand, can leverage more past frames without increasing the latency.

To overcome this limitation of BSVD a modification is made where only one buffer block is bi-directional and the rest are uni-directional. A visual example of such an architecture is shown in Fig. 4 and the buffer-based inference of the buffer blocks is shown in Figs. 5 and 6. This way, the latency is reduced to one frame while the temporal receptive field is asymmetrically expanded to $1 + 2N_{\text{bi}} + N_{\text{uni}} = 3 + N_{\text{uni}}$. In other words, the model has access to the current frame, one future frame, and $N_{\text{bi}} + N_{\text{uni}} = 1 + N_{\text{uni}}$ past frames.

4. Experiments

The effectiveness of the proposed dataset and model architectures are tested in a set of experiments that build on top of one another. Together, they are intended to guide the design of an efficient learning-based method that reconstructs noise-free and detailed RGB video sequences from raw camera data.

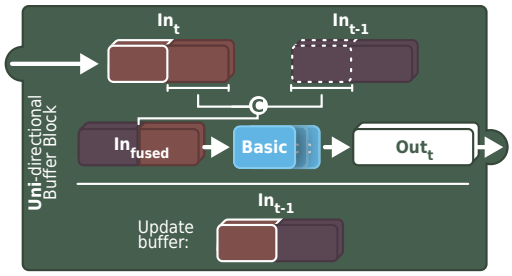


Figure 5. Feature fusion and buffer update inside the uni-directional buffer block (**Uni** in Fig. 4) during inference. 3-D boxes represent features, where the horizontal dimension is the channel dimension  . The dashed lines represent buffered features from the previous inference step that are stored as a property of the buffer block.

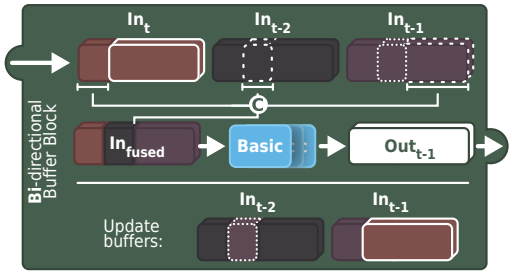


Figure 6. Feature fusion and buffer update inside the bi-directional buffer block (**Bi** in Fig. 4) during inference.

The section begins by presenting the order in which the experiments are conducted, and how insights from one form the starting point of the next one. An overview of the training methods is given, which allow the experiments to be executed in a computationally scalable manner. Then, it is described how each experiment was conducted, and discussed, what insights were gained. The final subsections demonstrate results from optimizing a model for GPU inference which indicate that real-time performance is within reach.

4.1. Experiment Structure

Guiding the structure of the experiments is a set of assumptions:

1. A U-Net's intra-block design can have a profound impact on the performance and efficiency of image restoration models.
2. The efficiency findings from experiments on the denoising task transfer to the debayering task, as both are low-level computer vision image restoration problems.
3. A joint denoising and debayering model is more efficient than two sequentially executed and separately

optimized denoising and debayering models.

4. Access to temporally adjacent frames can improve denoising efficiency and temporal consistency.

These assumptions align well with observations from related works in the domain of joint video denoising and debayering [13]. Conveniently, they compound on one another: A joint video denoising and debayering architecture which leverages all four insights can pull from a wide range of image restoration research, improve efficiency by merging tasks into a joint model, and take advantage of the temporal nature of video. By breaking design decisions into 4 discrete experiments, each decision can be supported by efficiency metrics and easily verified. These 4 experiments each concern a specific model **task** and aim to answer a *design-decision question*:

I **Spatial denoising:** Find an efficient intra-block design that scales well across limited computational budgets. *Does the design of the fundamental block of the U-Net have a large impact on efficiency?*

II **Spatial debayering:** Use the best intra-block design from the denoising experiments to construct "early" and "late debayering" architectures. *Does a biliniearly-guided architecture perform more efficiently?*

III **Joint spatial denoising and debayering:** Use the best spatial debayering architecture as the starting point for a joint spatial denoising and debayering model. *Does the joint model perform more efficiently than the two sequential models chained together?*

IV **Joint spatiotemporal denoising and debayering:** Add asymmetrical BSVD feature fusion to leverage temporal redundancies. *Does denoising efficiency and flickering improve? How large should the temporal receptive field be?*

4.2. Training Procedure

Each experiment was executed with an Optuna [3] hyperparamter study using a multivariate TPE sampler [5, 15], which efficiently suggests model architecture and training hyperparamters. Trials that perform poorly in relation to past configurations were pruned with a percentile pruner (25% pruning percentile) and models whose complexity falls outside a user-defined range were pruned right after sampling. The models were trained with the AdamW optimizer [27], whose learning rate and momentum were varied over the duration of each trial with a OneCycle [27] cosine schedule. For the training loss, early models trained in linear raw domain used a relative L1 loss function while later models trained in LogC4 [11] domain use the standard L1 loss. For validation at training-time, the MS-SSIM metric was chosen where applicable. For validation at test-time the higher computational cost of the LPIPS metric could be afforded and thus, the figures presented hereinafter are visualize this perceptual metric. Both the MS-SSIM and LPIPS are computed on brightness-normalized patches to ensure an exposure invariant weighting.

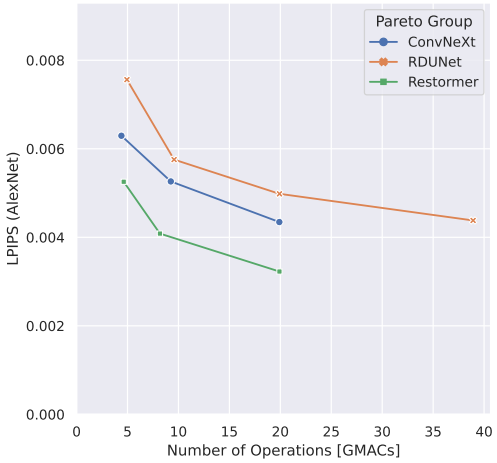


Figure 7. Pareto frontier of bilinearly debayering denoised model output at EI 6400.

4.3. Experiment I: Spatial Denoising

To find an efficient intra-block design for the denoising architecture, this experiment consists of multiple studies where each study has a fixed *design type* and an allowed complexity range. For each design type, three studies were conducted where the complexity was constrained to $x \leq 5$, $5 \leq x \leq 10$, and $10 \leq x \leq 20$ GMACs respectively. The design types to be compared are *RDUNet* [18], *ConvNeXt* [13, 26], and the *Restormer* [41]. The author was not yet aware of the Restormer's local conversion penalty and thus, these early experiments do not use NAFNet.

After all of the studies were completed, the best models were benchmarked on the validation set. By measuring the brightness of each validation patch, it is possible to group the validation results into dim (noisy) and bright (clean) subsets. Fig. 7 illustrates, how well each model type performs at denoising dimly exposed patches (EI 6400). The results demonstrate, that the design of a U-Net's basic block does in fact have a large impact on the model efficiency. Because the Restormer design was most efficient w.r.t. GMAC count and competitive w.r.t GPU runtime, it is chosen for future experiments as the best-performing basic block.

4.4. Experiment II: Spatial Debayering

The goal of this experiment is to investigate which debayering architecture is more efficient: "Early debayering"

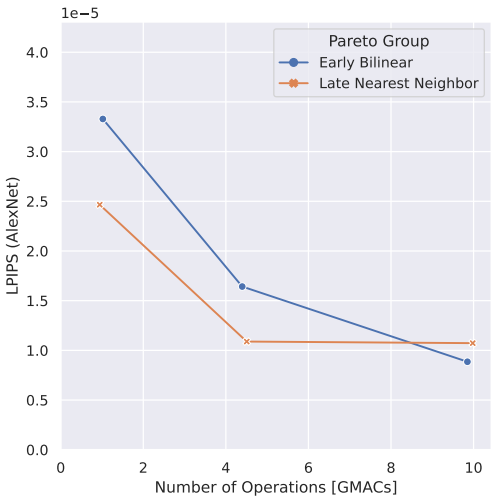


Figure 8. Validation metrics on debayering clean inputs at EI 200.

with bilinearly interpolated inputs or "late debayering" with a nearest neighbor residual. At large, the experiment was conducted in the same manner as the denoising task. The input and output domains of the data were adjusted as needed for the debayering task and hyperparameter optimization studies were were constrained to $x \leq 1$, $1 \leq x \leq 5$, and $5 \leq x \leq 10$ GMACs.

Overall, the efficiency of early and late debayering architectures follows roughly the same trend. The late nearest neighbor method seems more efficient with low compute budgets, but this difference could just as well be within the hyperparameter optimization's margin of error. Because there is no clear efficiency advantage with the "early debayering" architecture, the simpler and more memory-efficient "late debayering" design is used in subsequent experiments.

4.5. Experiment III: Joint Spatial Deoising and Debayering

To train a joint model for spatial denoising and debayering, the late nearest neighbor residual is used as the outer architecture. Rather than inputting clean bayer data, the inputs of this model are noisy. The Restormer design is once again used as the inner architecture. Fig. 9 compares the efficiency of the joint spatial models with the efficiency of a spatial denoising model and a spatial debayering model chained sequentially. It was found, that allocating more than 1 GMAC the debayering model reduced the overall efficiency of the sequential model. Comparing the initial set of joint models (marked *Spatial Restormer* in red) with the most efficient sequential models, it can be seen that the joint model performs worse at debayering (EI 200) and denoising (EI 6400). This seemingly violates the initial assumption, that joint models are more efficient. Upon further experimentation, it was found that this drop in efficiency can be eliminated by training the joint model for more epochs (200 epochs, marked *2x Steps* in Fig. 9). For reference, the sequential models were also trained for longer (200 epochs each) but their performance did not improve as much.

For visual examples, it is recommended to watch the supplementary video titled `2_spatial_models`. It compares the sequential 10 GMAC $\rightarrow$ 1 GMAC models with the joint 10 GMAC model, each trained for 200 epochs. Overall, their denoising performance is similar but the sequential model exhibits discolorization artifacts around edges. This can be attributed to the domain gap between training and inference: The debayering model is trained on clean Bayer inputs and is then given denoised Bayer inputs at inference time. In the future, an end-to-end trained sequential model could be investigated.

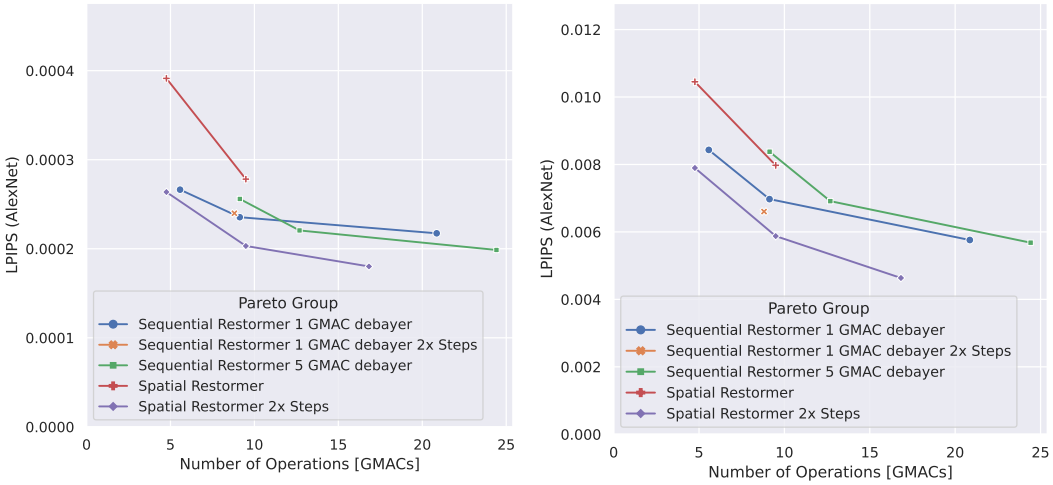
The joint model performs significantly better in brightly exposed scenes as well, outperforming both the sequential model and the current ADA-7 debayering algorithm in reconstructing monochromatic and self-similar imagery.

4.6. Experiment IV: Joint Spatiotemporal Denoising and Debayering

The goal of this experiment is to further improve the efficiency of the joint model by adding temporal feature fusion. By specifying how many bidirectional and unidirectional blocks are used near the bottleneck, the temporal receptive field can be controlled. At the time of this experiment, the local conversion limitation of Restormer was discovered and thus a set of NAFNet architectures was added for comparison.

The results in Fig. 10 show, that the temporal models are much more efficient at reconstructing noisy image regions (EI 6400) and slightly more capable at debayering clean image regions (EI 200). The depthwise NAFNet models achieve comparable GMAC efficiency to their Restormer counterparts. Additionally, the fused NAFNet models outperform the Restormer models' runtime efficiency. Considering that the real-world Restormer runtime is further increased once TLC is added, the NAFNet architectures have a clear efficiency advantage. Also, the NAFNet architectures benefit from compilation. Using `torch.compile` in PyTorch 2.0, their runtime can be reduced by a factor of 2.

For a comparison between the fully converged models at various temporal receptive fields, the reader is directed to the supplemental video material titled `3_1_temporal_receptive_field`. Viewed in motion, it is immediately apparent that their perceived denoising quality was much improved over the spatial models. Particularly static sequences processed by the spatial model contain distracting flickering artifacts, which are reduced significantly when the temporal receptive field is enlarged. The sequences no longer flicker abruptly, but instead, tran-



(a) EI 200

(b) EI 6400

Figure 9. Comparison of the joint spatial architectures' efficiencies with sequential denoising and debayering models. When training the joint model for a more adequate number of steps (2x steps), it is demonstrably more efficient at both debayering brightly exposed patches (EI 200) and noisy dim patches (6400 EI).

sition smoothly from one state into another in a wave-like manner. To capture this phenomenon quantitatively, an experiment was conducted in which a static sequence was denoised with a receptive field of 1 (0 past / 0 future), 3 (1 past / 1 future), and 5 (3 past / 1 future) frames. A frame from the middle of the sequence was then compared to the temporally adjacent frames using the LPIPS metric. The results in Fig. 11 show that an increased temporal receptive field causes consecutive frames to be perceptually similar.

For a comprehensive comparison between the fully trained spatiotemporal Restormer and NAFNet models, the reader is highly encouraged to watch the supplementary materials titled `3_2_nafnet_models`. Visual examples of difficult-to-debayer imagery are found in video `3_3_nafnet_debayering`. It is demonstrated that the NAFNet24 model is visually competitive with the 10 GMAC Restormer w.r.t. the removal of noise while being much more suitable for GPU inference. The computational cost of the NAFNet16 model is even lower, but its ability to remove noise while preserving details suffers from the limited budget. At the same time, the NAFNet outperforms both ADA-7 and the Restormer at debayering challenging imagery.

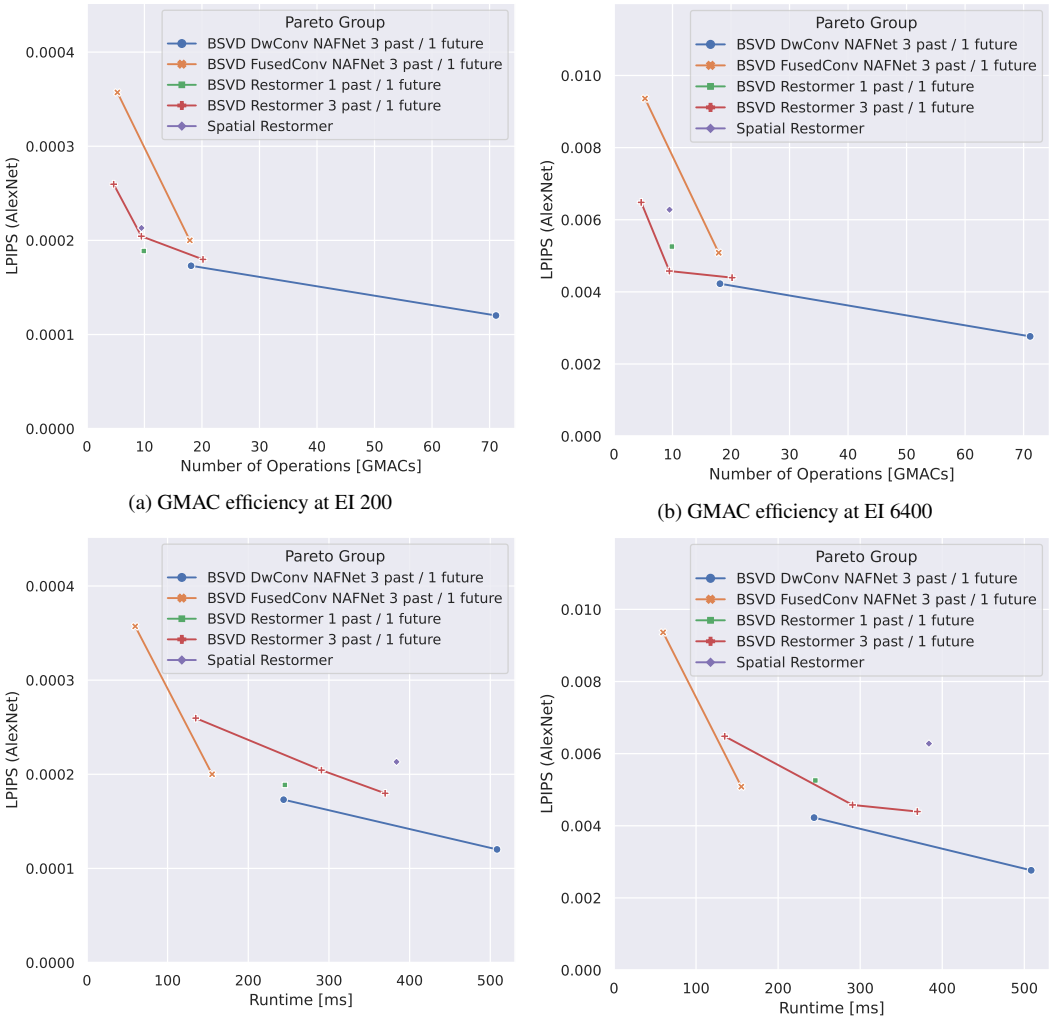
All the denoising models fall into the blurriness failure mode once a certain level of noise is reached. This cannot be avoided, as the signal-to-noise ratio approaches zero in dark image regions. To counteract this unnatural appearance, a method for reducing the denoising strength in the luminance dimension is proposed. By introducing some of the noise back into the image in a controlled and monochromatic manner, the waxy appearance of the denoising methods can be masked while improving the apparent detail in the image. For visual examples, the reader is referred to the video titled `4_denoising_strength_control`.

For a comparison of several traditional SOTA denoising methods and commercially available learning-based denoising methods, the reader is urged to watch `5_comparison_with_industry_standard_commercial_solutions` in the supplemental video material. It shows that out of the tested solutions, only Neat Video can reduce noise to a degree comparable with NAFNet24. Their runtime is similar as well (NAFNet24: 42FPS, Neat Video: 53 FPS at a resolution of 1920×1080), which demonstrates that the proposed spatiotemporal NAFNet24 model already meets the performance standard for inference during post-production.

5. Conclusion

Convolutional neural networks have been demonstrated both in literature and in the experimental results of this thesis to be capable of performing denoising and debayering. Regarding the specific use case in a digital motion picture camera, several limitations of existing datasets led to the conclusion that a camera-specific high dynamic range video dataset is needed to obtain a training distribution that closely matches real camera recordings. The proposed methods for creating such a dataset were demonstrated to be both simple and effective. Models trained on the dataset performed well on real camera data.

The CNNs trained to remove noise using a pixel-wise loss function have a tendency to exhibit a blurriness failure mode: If the noise level is too high or temporal and spa-



(a) GMAC efficiency at EI 200

(b) GMAC efficiency at EI 6400

(c) GPU efficiency at EI 200

(d) GPU efficiency at EI 6400

Figure 10. Efficiency of fully trained joint models at debayering (EI 200) and denoising (EI 6400) in terms of theoretical GMAC (top) and compiled runtime without TLC (bottom). The NAFNet architectures, particularly NAFNet24, strike a good balance between GPU runtime and performance metrics at both tasks.

tial information does not suffice to infer the noise-free latent image, these deep learning models render fine textures in a wax-like manner. This loss of detail was shown to be controllable by reducing the intensity with which the luminance channel is denoised, producing an achromatic film-grain-like texture.

Through automated experimentation and results-driven iterative design, a set of efficient architectures was found that leverage recent advancements on the low-level design of CNNs, on temporal feature fusion, and on optimization for GPU inference. The use of model compilation and mixed precision arithmetic further lowered the runtime of a model to 155 ms without sacrificing image quality. Despite the large input resolution of over 12 million pixels, these architectures are within reach of the real-time threshold on a modern GPU while producing results that are visually competitive with SOTA industry-standard methods. For deployment in post-production environments, where high performance GPU are common, the computational cost of the models can likely be afforded. Integrating it into the image processing pipeline of a battery powered handheld camera system requires further optimization. In future work, power efficient AI accelerator hardware and accelerator-aware network architecture optimization should be investigated.

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S. Brown. A high-quality denoising dataset for smartphone

Figure 11. Temporal consistency of a static scene.

cameras. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2018. 6, 8

[2] Eirikur Agustsson and Radu Timofte. NTIRE 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, jul 2017. 6

[3] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019. 10

[4] Stefano Andriani, Harald Brendel, Tamara Seybold, and Joseph Goldstone. Beyond the kodak image set: A new reference set of color image sequences. In *2013 IEEE International Conference on Image Processing*. IEEE, sep 2013. 6

[5] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. 10

[6] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. Unprocessing images for learned raw denoising, 2018. 6

[7] Jiezhang Cao, Qin Wang, Jingyun Liang, Yulun Zhang, Kai Zhang, Radu Timofte, and Luc Van Gool. Learning task-oriented flows to mutually guide feature alignment in synthesized and real video denoising, 2022. 5

[8] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark, 2018. 6

[9] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration, 2022. 1, 2, 4

[10] Xiaojie Chu, Liangyu Chen, Chengpeng Chen, and Xin Lu. Improving image restoration by revisiting global information aggregation, 2021. 4

[11] Sean Cooper and Harald Brendel. Arri logc4. Technical report, ARRI, 2022. 10

[12] Valéry Dewil, Aranud Barral, Gabriele Facciolo, and Pablo Arias. Self-supervision versus synthetic datasets: which is the lesser evil in the context of video denoising?, 2022. 1, 3, 5, 8

[13] Valery Dewil, Adrien Courtois, Mariano Rodriguez, Thibaud Ehret, Nicola Brandonisio, Denis Bujoreanu, Gabriele Facciolo, and Pablo Arias. Video joint denoising and demosaicing with recurrent CNNs. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE, jan 2023. 6, 10

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020. 1, 3

[15] Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1437–1446. PMLR, 10–15 Jul 2018. 10

[16] François Fleuret. The little book of deep learning, 2023. 3

[17] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Transactions on Graphics*, 35(6):1–12, nov 2016. 5

[18] Javier Gurrola-Ramos, Oscar Dalmau, and Teresa E. Alarcon. A residual dense u-net neural network for image denoising. *IEEE Access*, 9:31742–31754, 2021. 1, 2, 10

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[20] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. 3

[21] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2016. 2

[22] Daniel Khashabi, Sebastian Nowozin, Jeremy Jancsary, and Andrew W. Fitzgibbon. Joint demosaicing and denoising via learned nonparametric random fields. *IEEE Transactions on Image Processing*, 23(12):4968–4981, dec 2014. 6, 7, 8

[23] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data, 2018. 6

[24] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding, 2018. 5

[25] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021. 1, 3

[26] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s, 2022. 1, 2, 10

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017. 10

[28] Gunther Machu. Arri alexa 35 lab test: Rolling shutter, dynamic range and latitude - plus video!, Aug 2022. 6

[29] Matteo Maggioni, Giacomo Boracchi, Alessandro Foi, and Karen Egiazarian. Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms. *IEEE Transactions on Image Processing*, 21(9):3952–3966, sep 2012. 4

[30] Tobias Plötz and Stefan Roth. Benchmarking denoising algorithms with real photographs, 2017. 6

[31] Chenyang Qi, Junming Chen, Xin Yang, and Qifeng Chen. Real-time streaming video denoising with bidirectional buffers. In *Proceedings of the 30th ACM International Conference on Multimedia*. ACM, oct 2022. 4, 5, 9

[32] Guocheng Qian, Yuanhao Wang, Jinjin Gu, Chao Dong, Wolfgang Heidrich, Bernard Ghanem, and Jimmy S. Ren. Rethinking learning-based demosaicing, denoising, and super-resolution pipeline, 2019. 5, 6

[33] Michael Schoberl, Wolfgang Schnurrer, Alexander Oberdorster, Siegfried Fossel, and Andre Kaup. Dimensioning of optical birefringent anti-alias filters for digital cameras. In *2010 IEEE International Conference on Image Processing*. IEEE, sep 2010. 6, 7

[34] Matias Tassano, Julie Delon, and Thomas Veit. DVDNET: A fast network for deep video denoising. In *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2019. 5

[35] Matias Tassano, Julie Delon, and Thomas Veit. Fastdvdnet: Towards real-time deep video denoising without flow estimation, 2019. 5

[36] Chunwei Tian, Lunke Fei, Wenxian Zheng, Yong Xu, Wangmeng Zuo, and Chia-Wen Lin. Deep learning on image denoising: An overview, 2019. 4

[37] Jeya Maria Jose Valanarasu, Rahul Garg, Andeep Toor, Xin Tong, Weijuan Xi, Andreas Lugmayr, Vishal M. Patel, and Anne Menini. Rebotnet: Fast real-time video enhancement, 2023. 5

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 3

[39] Wenzhu Xing and Karen Egiazarian. End-to-end learning for joint image demosaicing, denoising and super-resolution. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2021. 1, 5, 6

[40] Huanjing Yue, Cong Cao, Lei Liao, Ronghe Chu, and Jingyu Yang. Supervised raw video denoising with a benchmark dataset on dynamic scenes, 2020. 6

[41] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration, 2021. 1, 2, 3, 4, 10

[42] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, jul 2016. 2