Mobile Aware Denoiser Network (MADNet) for Quad Bayer Images

Pavan C. Madhusudana Jing Li Zeeshan Nadir Hamid R. Sheikh

Seok-Jun Lee

Samsung Research America

Abstract

Pixel binning is a term that is gaining popularity lately. It consists of using high pixel density camera sensors where the pixels are grouped together when low light levels are encountered, and, in the case of bright light scenes, the pixels are not grouped together. One such pixel arrangement is Quad Bayer or Tetra. Historically, significant efforts have been dedicated to demosaicing and denoising Bayer images, yet limited consideration has been directed towards Quad Bayer sensors, owing to their recent introduction. One unique challenge in training deep learning networks for Quad Bayer images is how to encode such data (spatial vs depth arrangement). Conventionally, when training denoising networks on bayer images, the input is split in to individual color channels, however, as results would show, taking that approach in case of Quad Bayer images produces inferior quality results.

In this paper, we present an efficient way of grouping the pixels of a tetra sensor that achieves the best trade off between image quality and inference speed. Due to very large number of pixels, the network training requires enormous amounts of data, making the network prone to over-fitting in case of limited data. In order to regularize the network so as to not overfit, we present a novel inter channel loss function that effectively regularizes the network training. Finally, we do an ablation study to analyze the loss function that we present, pixel grouping for tetra sensor and the proportion of input data with different amounts of noise level. Results show that the techniques presented in this paper produce denoised tetra images that are of better quality than traditional methods. We hope that this paper will inspire further research in developing algorithms for the new Quad Bayer, Hexa Deca and Nona sensors.

1. Introduction

Image denoising belongs to the class of problems known as image restoration and is considered to be an important component of many image processing pipelines [4, 6, 35]. Conventionally, image denoising is done on single gray scale images or demosaiced 3 channel RGB images [7, 11, 18]. However, many times, there is a need to denoise these images while they are in linear raw format prior to demosaicing [3, 9]. Noise in raw images tends to be less correlated and therefore less challenging to handle; in addition, raw images have 3 times less number of unknown pixel values as comapred to RGB images. Nevertheless what makes denoising images in raw format difficult is that these images consists of pixels of different colors adjacent to each other [1, 29]. The exact arrangement of different color channels on the pixel grid depends on the specific color filter array used in the sensor.

As the smartphone camera industry trends towards high mega pixel cameras such as 200 MP, the latest CMOS sensors are now adopting more novel CFA patterns such as Quad Bayer (Tetra), Nona and Hexa Deca (Tetra^2) pattern [12–14, 16]. These new CFA patterns offer the flexibility of binning the adjacent pixels consisting of the same color channel for better imaging signal in low light scenarios. Denoising these non-Bayer raw images presents unique challenges because of the very large number of pixels that need to be processed and increased pixel distance between pixels of different colors [16].

Many different types of denoising techniques have been proposed in the past. Some of the more notable denoising methods include patch based denoising methods that exploit image self similarity [2, 24, 25], dictionary learning [6], low rank approximation [10, 30], bayesian modelling [19, 20, 26, 34], and frequency domain approaches [8, 31]. More recently with the advancements in GPUs, deep learning has opened up new avenues of approaching the denoising problem [3, 17, 33]. However, most of these deep neural networks are trained for RGB or the traditional bayer pat-

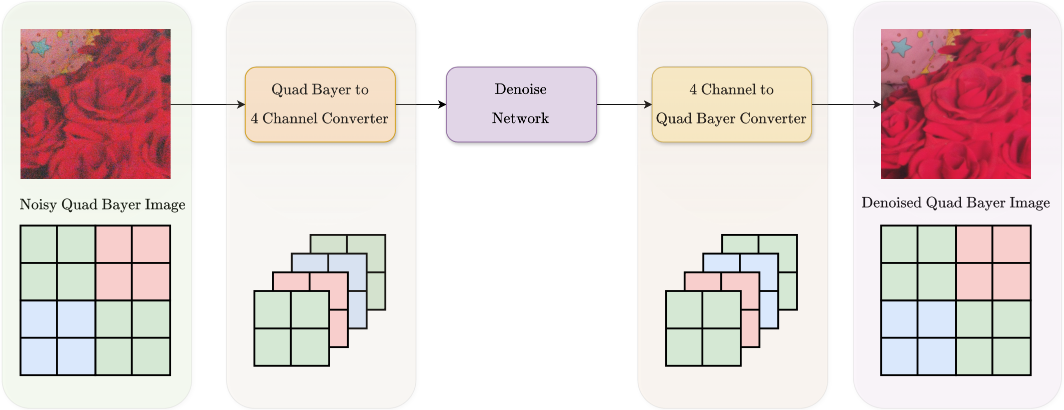


Figure 1. An overview of the Quad Bayer denoise algorithm: In the first step, the Quad Bayer to 4 channel converter converts the input into a four channel data format. Then we use a network to denoise the noisy input. We use the 4 channel to Quad Bayer converter to convert the 4 channel back to Quad Bayer format.

tern raw images [3, 9]. Traditionally, in order to train a deep neural network on bayer raw images, the practice has been to separate individual color phases in to a 4 channel RGGB image [3]. However, a crucial shortcoming of such an approach is that when you separate out the 16 channels in a Quad Bayer image, you get aliasing artifacts due to spatial downsampling.

In this paper, we present a robust deep learning based approach to train a denoising network for raw images including non Bayer patterns such as Quad Bayer images. Our three main contributions in this paper are as follows. First, we propose to use a grouping of pixels that may not appear obvious i.e., instead of encoding Quad Bayer images in to 16 channels, we break it into 4 channels by combining adjacent pixels corresponding to the same color. Second, we propose a novel inter-channel loss function inspired by the traditional total variation loss function but with a key difference; our inter channel loss function combination penalizes the differences between predicted pixel values and the adjacent pixels in the ground truth image. Third, we propose a practical way to regularize the training of denoising neural network by employing two different sets of training data; low exposure data which tends to have high level of noise and high exposure data which tends to be less noisy.

2. Method

The overall architecture of our Quad Bayer denoising algorithm is illustrated in Fig 1. There are three components in the algorithm: Quad Bayer to 4 Channel Converter, Denoise Network, 4 Channel to Quad Bayer Converter. The Quad Bayer to 4 Channel Converter converts the Quad Bayer data into an 4 channel format. Then we feed the 4 channel data to the second component Denoise Network, in the end the 4 Channel to Quad Bayer converter converts the denoised 4 channel output to Quad Bayer format for the downstream tasks. In addition, we created a detail preserving inter-channel loss function. We will introduce each component of our method in details in this section.

2.1. Quad Bayer/4 Channel Converter

In this section, we discuss how we group the pixels in the Quad Bayer images to prepare the input to the network and how we convert back to the Quad Bayer format.

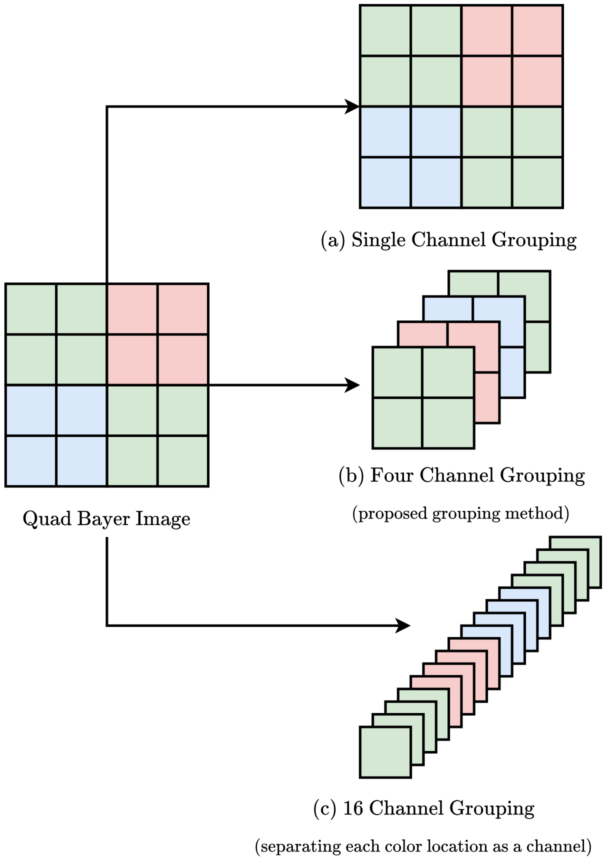
Quad Bayer to 4 Channel Converter As shown in Fig 2, there are two mainstream methods of grouping the pixels before feeding the Quad Bayer into the denoise network. One is to use the Quad Bayer image as it is. The other one is splitting the Quad Bayer pattern into 16 channels as shown in Fig 2-c. Here, we propose a new way of grouping the pixels. We group based on the color clusters within the Quad Bayer pattern. Fig 2-b shows how to organize the pixels. After passing the converter, the Quad Bayer image is converted into a 4 channel data.

4 Channel to Quad Bayer Converter This step is an inverse process of Sec 2.1 which convert the 4 channel data back into Quad Bayer format.

2.2. Denoise Network

We present the detailed network architecture in Fig 3. Our network is based on a U-Net [22] architecture well suited for image denoising. Nevertheless we present a general framework in which any netwrok can be plugged in to denoise the images.

In our network, we use three types of convolutional layers: standard convolutional layer, strided convolutional layer used for downsampling and transpose convolutional layer used for upsampling. In the encoder, we use standard convolutional layer followed by strided convolutional layer



(a) Single Channel Grouping

(b) Four Channel Grouping

(proposed grouping method)

(c) 16 Channel Grouping

(separating each color location as a channel)

Figure 2. Different methods of grouping Quad Bayer pixels together: two traditional way of grouping pixels: (a) using a single channel input as it is, and (c) splitting the Quad Bayer data into 16 channels by using each color component as a channel. Our proposed method (b) separates Quad Bayer pattern into 4 channels with each channel being a group of same color.

to reduce the spatial dimensions of the features by a factor of 2 and increase the channel numbers by 2 (the channel number of the first layer is 16). In total, we have 5 consecutive blocks in the encoder.

In the decoder, we use transpose convolutional layer followed by standard convolutions to increase the spatial dimensions of the features by a factor of 2 and reduce channel numbers by 2 (i.e., 16). Here we use concatenation of features in the decoder and corresponding features from the encoder for skip connection.

2.3. Detail Preserving Inter Channel Loss Function

In the literature Total Variance (TV) objective has been extensively employed for edge preserving image denoising [23]. Inspired by the successes of TV loss, we propose a novel Inter Channel Loss (ICL) with the goal of having similar intensity values for same color neighborhood pixels in the Quad Bayer pattern and also preserve the edges. However, the key distinction between TV loss and ICL is that unlike TV loss which exclusively applies the constraint on the output image, we compute the similarity of neighborhood pixels between the output and ground-truth images. ICL is defined as

$$L_{ICL} = \sum_{S\in\Omega}\sum_{k=0}^{N/4}\sum_{i=1}^{4}\sum_{j=1}^{4} w_{i,j}\|\hat{Y}_{S_{4k+i}} - Y_{S_{4k+j}}\|, \tag{1}$$

where $\Omega = \{R, Gr, B, Gb\}$ is the color space in Quad Bayer, N represents the number of pixels in each image, S denotes the channel index, k is the index of the 4-pixel group, i and j are the indices of the pixel within each group, $w_{i,j}$ is the weight for the difference. For easier interpretability, ICL can be further categorized into two parts : L_{ICL}^{self} and $L_{ICL}^{neighbor}$

- L_{ICL}^{self} : in the equation (1) assigning $w_{i,j} = 0$ for $i \neq j$, and $w_{i,j} = 1$ for $i = j$ reduces the expression to a L_1 loss function.

$$L_{ICL}^{self} = \sum_{S\in\Omega}\sum_{k=0}^{N/4}\sum_{i=1}^{4}\|\hat{Y}_{S_{4k+i}} - Y_{S_{4k+i}}\|$$

- $L_{ICL}^{neighbor}$: This loss is obtained by assigning $w_{i,j} = 1/3$ for $i \neq j$, and $w_{i,j} = 0$ for $i = j$. This loss essentially compares all the neighborhood pixel values with the ground-truth except those located in the same coordinates.

$$L_{ICL}^{neighbor} = \sum_{S\in\Omega}\sum_{k=0}^{N/4}\sum_{i=1}^{4}\sum_{j=1,j\neq i}^{4} \frac{1}{3}\|\hat{Y}_{S_{4k+i}} - Y_{S_{4k+j}}\|$$

In our experiments we determine the relative weighting of L_{ICL}^{self} and $L_{ICL}^{neighbor}$ empirically.

2.4. Overall Loss Function

We trained MADNet using a weighted combination of three terms: L_{ICL}^{self}, $L_{ICL}^{neighbor}$, and L_{SSIM} [1] [28].

$$\begin{aligned} L = \alpha^{self} * L_{ICL}^{self} + \alpha^{neighbor} * L_{ICL}^{neighbor} \\ + \alpha^{SSIM} * L_{SSIM}, \end{aligned} \tag{2}$$

where α^{self}, $\alpha^{neighbor}$ and α^{SSIM} are the weights associated with the three different loss functions subject to the constraint

$$\alpha^{self} + \alpha^{neighbor} + \alpha^{SSIM} = 1.$$

In our experiments α^{self}, $\alpha^{neighbor}$ and α^{SSIM} are hyperparameters values of which are determined empirically using a cross validation dataset.

[1] In our implementation we use the multi-scale version of the SSIM loss function [27].

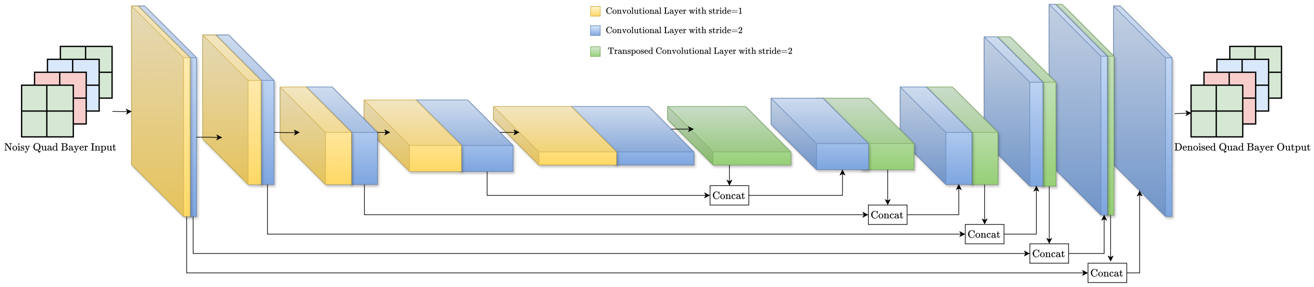

Convolutional Layer with stride=1
Convolutional Layer with stride=2
Transposed Convolutional Layer with stride=2
Noisy Quad Bayer Input
Denoised Quad Bayer Output
Concat
Concat
Concat
Concat
Concat

Figure 3. Illustration of MADNet architecture. MADNet is based on U-Net [22] style architecture containing strided convolutional and trasnposed convolutional layers. Concat denotes concatenation of feature channels.

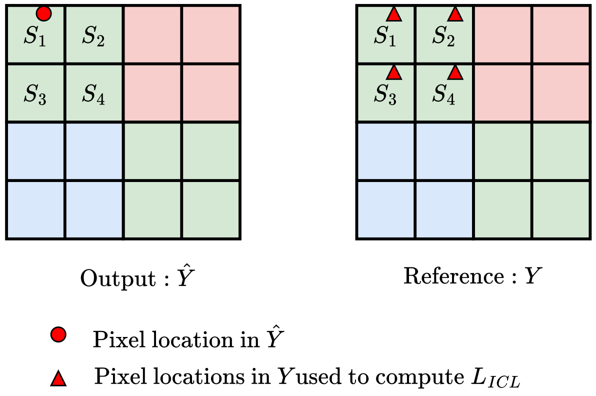

S_1
S_2
S_3
S_4
Output : $\hat{Y}$
S_1
S_2
S_3
S_4
Reference : Y
Pixel location in $\hat{Y}$
Pixel locations in Yused to compute L_{ICL}

Figure 4. Illustration of a L_{ICL}. In the figure, the loss for S_1 in $\hat{Y}$ is computed with all pixels within the same color block, i.e. S_1,S_2, S_3, S_4 in Y .

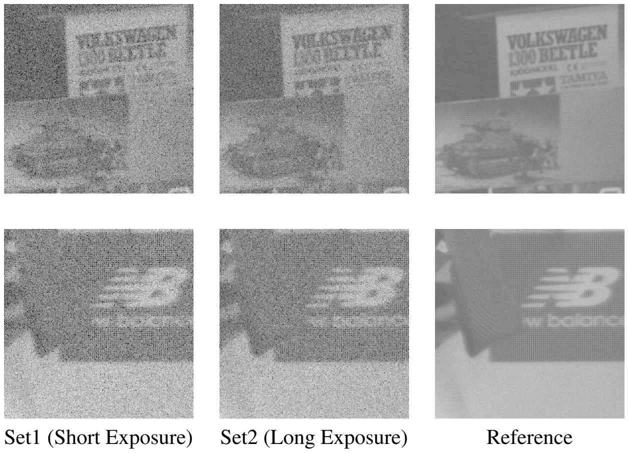


Figure 5. Sample noisy images from the captured datasets Set1 (first column) and Set2 (second column). Reference images are shown in third column. Raw Quad Bayer images are visualized as grayscale images with no additional demosaicing. The difference in noise characteristics between Set1 and Set2 can be better visualized by zooming in.

3. Experiments

In this section we provide details on the experiments we conducted to evaluate our proposed denoising technique. We will first provide details on the experimental settings such as training data, evaluation methodology and compared methods. Then we will objectively and visually evaluate the performance of our model against state-of-the-art (SOTA) denoising models. At the end, we perform several ablation experiments to analyze the importance of ICL objective, performance differences between ICL and the standard Total Variation objectives and the effect of Quad Bayer to 4 channel conversion.

3.1. Experimental Settings

Training Data

We captured a new dataset to train and benchmark raw Quad Bayer images. The images were captured in low-light conditions using Samsung Galaxy S22 smartphone camera which used a specially designed software to obtain raw images in Quad Bayer format of resolution 6120 × 8160. This software also enabled modifying camera settings such as ISO and exposure time. During capture the smartphone was mounted on a tripod and remotely triggered. Both indoor and outdoor scenes were captured in this dataset.

Captured data consisted of two sets of noisy images : (a) Set1 - containing short exposure images, and (b) Set2 - containing long exposure images. For each of the noisy images in both of the sets, there is a corresponding reference (ground truth) image with significantly long exposure time producing a ground truth image with low noise and better perceptual quality. A sample image pair from this dataset is shown in Fig. 5. To avoid the problem of motion blur, we only used static images without any motion in our experiments. Additionally, same reference may be used for images coming from the two different sets as illustrated in Fig. 5. In total we captured a total of 2876 noisy-reference image pairs in each of Set1 and Set2. The motive to include two different sets is to make denoising models more robust to a wide range of noise levels. To capture noisy images, the exposure time was set between 1/50 to 1/5 seconds. For reference image, the exposure times were increased nearly 10 to 30 times, and ISO values were reduced accordingly to obtained an equally exposed image.

For model training, this dataset was further divided randomly into two sets of 4600 and 1152 image pairs corresponding to training and testing data respectively. We also ensured that images from both sets had equal representation in both training and testing data.

Training and Inference Details

MADNet was trained for 250K iterations with a batch size of 4. Further, each image in the batch was randomly cropped into 3 patches of size 512×512 during training. Adam optimizer [15] with a constant learning rate of 5×10^{-4} ($\beta_1 = 0.9, \beta_2 = 0.999$) was used to train MADNet. The following values were used for hyperparameters in equation (2) : $\alpha^{self} = 0.1$, $\alpha^{neighbor} = 0.05$ and $\alpha^{SSIM} = 0.85$. All the implementations were done in Python using PyTorch[2] framework, and all models were trained on a single Nvidia A100 GPU. To obtain mobile on-device inference, Snapdragon Neural Processing Engine (SNPE) was employed. The trained PyTorch models were converted to on-device compatible format using SNPE toolkit [21].

Comparisons and Evaluation Methodology

We objectively compared the performance of MADNet against two popular SOTA models : BM3D [5], which uses traditional collaborative filtering, and DnCNN [32], which employs an end-to-end trained deep model for noise removal. In our experiments, we ran the BM3D algorithm directly on Quad Bayer images without any prior conversion to another format such as demosaiced RGB images. In case of DnCNN, for fair comparison, we retrained the model with our Quad Bayer images which were encoded in the same manner of 4 channel components as shown in fig. 1.

For objective evaluation we employ standard image quality metrics : Peak Signal to Noise ratio (PSNR) and Structural Similarity Index (SSIM) [28]. Both PSNR and SSIM values are calculated for each image and average value across the entire dataset is reported. Since our emphasis has been efficient processing on a smartphone, we also compare the processing times of the above models on the latest Samsung Galaxy S24 device.

3.2. Denoising Performance

In Table 1 we compare the objective image quality metrics of MADNet against other denoising methods. From the table it can be observed that MADNet outperforms other methods both in terms of PSNR and SSIM numbers. Furthermore, MADNet outperforms other SOTA models in terms of processing time. We do not report processing time for BM3D since it was not optimized for NPU processing and the code released by the authors[3] was taking substantially longer time to denoise when compared to other methods. In Fig. 6 we visually compare the denoising results on two scenes from the testing data. These results show that MADNet preserves more details when compared with DnCNN and BM3D.

Table 1. Comparison of MADNet denoising performance against SOTA models.

Model	PSNR ↑	SSIM↑	Processing Time (ms) ↓
BM3D [5]	34.564	0.954	-
DnCNN [32]	39.662	0.973	1850
MADNet	**40.66**	**0.979**	**540**

3.3. Ablation Experiments

Here we provide details of various ablation experiments we conducted to analyze the robustness and generalizability of the trained MADNet model.

Effect of Short and Long Exposure Images

In Sec. 3.1 we noted that MADNet was trained on a combined dataset containing equal number of short (set-1) and long (set-2) exposure images. The idea to employ images with different exposure levels was to obtain a single model that can be used across a wide range of noise characteristics. In order to verify the utility of combining datasets with different noise levels, we conducted an experiment where MADNet was trained exclusively on either short exposure images or long exposure images. We compared the output quality of such trained models with the model trained on combined dataset from the two sets. The results are shown in Table 2; it can be observed that training the model using the combined dataset yields a slight advantage in terms of image quality metrics. In Fig. 7 we provide visual comparison of the denoised results and from this figure it can be seen that using only long exposure images leads to artifacts since the noise is not completely removed, while using only short exposure images leads to over smoothening. Employing both types of exposure level is essential to get best characteristics from both sets and obtain superior image quality.

Significance of ICL

MADNet was trained was trained using a combination of L_{ICL} and SSIM loss detailed in equation (2). In this experiment we analyze the significance of each component present in this objective by training various models with a

[2] https://pytorch.org/

[3] https://pypi.org/project/bm3d/

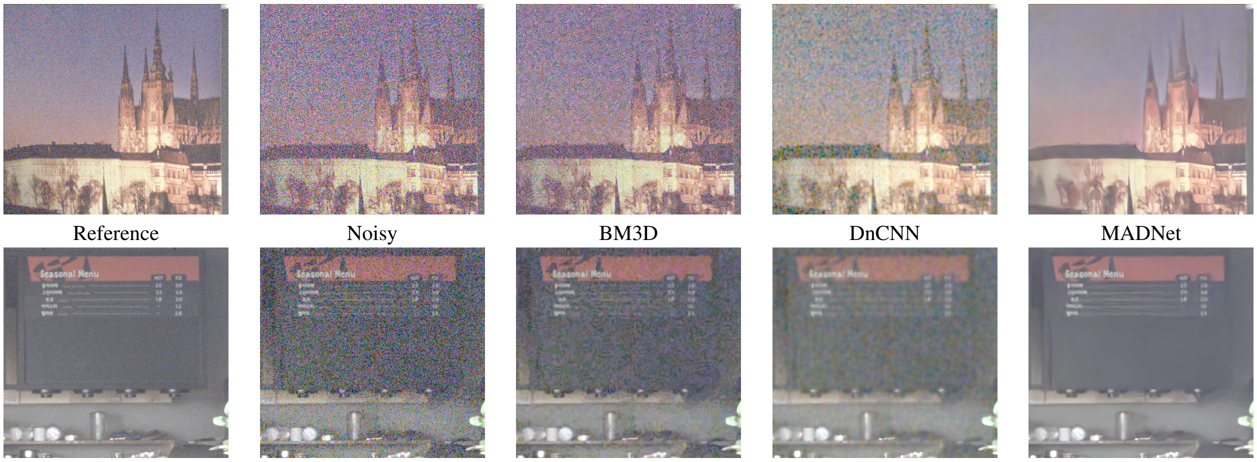


Figure 6. Visual comparison of denoising results across two test images.

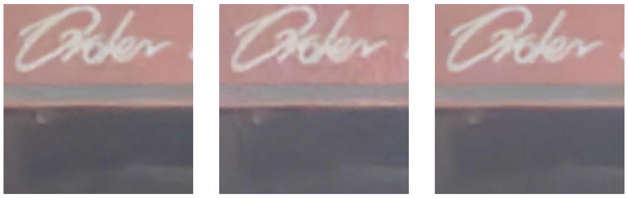


Figure 7. Comparison of denoising results when MADNet is trained exclusively on short or long exposure images with MADNet trained on combined dataset.

Table 2. Evaluation of MADNet when trained only on short and long exposure images.

Model	PSNR ↑	SSIM↑
Trained only on short exposure dataset	40.42	0.978
Trained only on only long exposure dataset	40.53	0.978
Trained on combined dataset	**40.66**	**0.979**

Table 3. Comparison of trained MADNet models with different choice of loss functions

L_{ICL}^{self}	$L_{ICL}^{neighbor}$	L_{SSIM}	PSNR ↑	SSIM↑
✓			38.36	0.972
	✓		38.69	0.971
		✓	38.35	**0.979**
	✓	✓	38.92	0.974
✓	✓		38.63	0.975
✓		✓	39.65	0.978
✓	✓	✓	**40.66**	**0.979**

combination of loss functions. The results are shown in Table 3 and from the reported values, using the proposed combination of L_{ICL} and SSIM objective produces the highest values in terms of objective quality metrics.

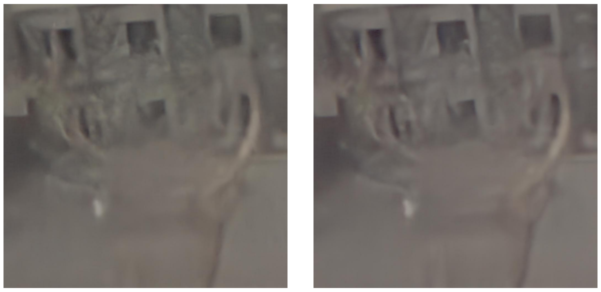


Figure 8. Visual comparison of denoised results using L_{ICL} and TV regularization. Results with TV loss yield smoother details (near edges,corners) when compared to L_{ICL}.

Comparison of Total Variation and ICL Objectives

One of our key contributions in MADNet is employing L_{ICL}^{self} and $L_{ICL}^{neighbor}$ for training. As mentioned in Sec. 2.3, ICL is mainly inspired by TV loss. Here we conduct an experiment where we replace $L_{ICL}^{neighbor}$ with TV regularization in equation (2) and train MADNet. The TV loss is given as follows

$$L_{TV} = \sum_{S\in\Omega} \sum_{k=0}^{N/4} \sum_{j=1}^{4} \sum_{i=1, i\neq j}^{4} \frac{1}{3} \|\hat{Y}_{S_{4k+i}} - \hat{Y}_{S_{4k+j}}\| \tag{3}$$

$$L = \alpha^{self} * L_{ICL}^{self} + \alpha^{TV} * L_{TV} + \alpha^{SSIM} * L_{SSIM}. \tag{4}$$

In our experiments we choose $\alpha^{TV} = \alpha^{neighbor}$. Note that L_{TV} and $L_{ICL}^{neighbor}$ are very similar, and L_{TV} is computed entirely on $\hat{Y}$ while $L_{ICL}^{neighbor}$ uses both Y and $\hat{Y}$. The objective results are shown in Table 4. From the table it is

Table 4. Comparison of trained MADNet models with and without ICL

Model	PSNR ↑	SSIM↑
TV regularization	39.62	0.978
ICL	**40.66**	**0.979**

Table 5. Impact of channel grouping on denoising performance

Model	PSNR ↑	SSIM↑	Processing Time (ms) ↓
No grouping	**40.75**	**0.980**	1639
4 channels	40.66	0.979	540
16 channels	39.62	0.978	**220**

evident that L_{ICL} obtains higher PSNR and SSIM values when compared with TV regularization. Visual comparison in Fig. 8 also indicates that TV regularization yields smoother images with inferior details when compared to L_{ICL}.

Effect of Channel Grouping

In this experiment, we analyze our channel grouping mechanism presented in Sec. 2.1. We study the effect of our channel grouping by training MADNet with two other variations of encoding: (a) No grouping - in this case the input Quad Bayer image is fed unmodified to the network as shown in Fig. 2-a, (b) 16 channel grouping - this is the traditional grouping mechanism illustrated in Fig. 2-c. Both of these approaches are self evident where (a) produces input images with no spatial downsampling and large sized hidden layers feature whereas (b) produces input images with spatial downsampling and small sized hidden layers features. The results are shown in Table 5 and Fig. 9. Although approach (a) obtains the best image quality; however it comes at the cost of significantly large processing time. On the other hand. 16 channel grouping is the fastest; but it leads to aliasing artifacts as seen in Fig. 9. In our view, the proposed 4 channel grouping offers a good trade-off between the image quality and processing time.

4. Conclusions

In this paper, we presented a deep neural network based framework to denoise recently popular Quad Bayer raw images. We train our denoising network with extensive emphasis on being light weight and ability to run on a smartphone, while preserving details in the images. The proposed model employed a novel channel grouping mechanism and a new Inter Channel Loss function for its training. Further, we used a Quad Bayer dataset containing a total of 5752 noisy-reference image pairs. This Quad Bayer dataset is a first of its kind, and contains a combination of long and short exposure images to capture a wide range of noise characteristics. We conducted a holistic evaluation of the reconstructed image quality and showed that our proposed model outperforms current SOTA methods; both across PSNR and SSIM objective metrics as well as visual quality. We also conducted a range of ablation experiments to analyze the importance of channel grouping and Inter Channel Loss function.

References

[1] SM A Sharif, Rizwan Ali Naqvi, and Mithun Biswas. Beyond joint demosaicking and denoising: An image processing pipeline for a pixel-bin image sensor. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 233–242, 2021. 1

[2] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, pages 60–65. Ieee, 2005. 1

[3] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3291–3300, 2018. 1, 2

[4] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Color image denoising via sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space. In *2007 IEEE international conference on image processing*, pages I–313. IEEE, 2007. 1

[5] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007. 5

[6] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006. 1

[7] Linwei Fan, Fan Zhang, Hui Fan, and Caiming Zhang. Brief review of image denoising techniques. *Visual Computing for Industry, Biomedicine, and Art*, 2:1–12, 2019. 1

[8] Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. *IEEE transactions on image processing*, 16(5):1395–1411, 2007. 1

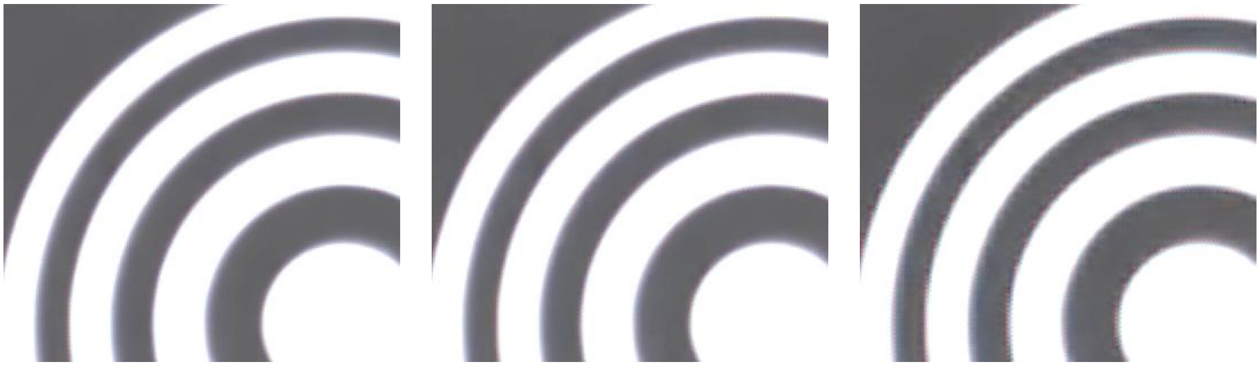
[9] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016. 1, 2

[10] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2862–2869, 2014. 1

[11] Paras Jain and Vipin Tyagi. A survey of edge-preserving image denoising methods. *Information Systems Frontiers*, 18:159–170, 2016. 1

[12] Jun Jia, Hanchi Sun, Xiaohong Liu, Longan Xiao, Qihang Xu, and Guangtao Zhai. Learning rich information for quad bayer remosaicing and denoising. In *Computer Vision – ECCV 2022 Workshops*, pages 175–191, Cham, 2023. 1

[13] Irina Kim, Seongwook Song, Soonkeun Chang, Sukhwan Lim, and Kai Guo. Deep image demosaicing for submicron



No grouping

4 channels

16 channels

Figure 9. Visual comparison of denoising results across different channel grouping methods.

image sensors. *Journal of Imaging Science and Technology*, 63(6):60410–1, 2019.

[14] Irina Kim, Dongpan Lim, Youngil Seo, Jeongguk Lee, Yunseok Choi, and Seongwook Song. On recent results in demosaicing of samsung 108mp cmos sensor using deep learning. In *2021 IEEE Region 10 Symposium (TENSYMP)*, pages 1–4. IEEE, 2021. 1

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Proc. Int. Conf. Learn. Representations*, pages 1–15, 2015. 5

[16] Haechang Lee, Dongwon Park, Wongi Jeong, Kijeong Kim, Hyunwoo Je, Dongil Ryu, and Se Young Chun. Efficient unified demosaicing for bayer and non-bayer patterned image sensors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12750–12759, 2023. 1

[17] Lin Liu, Xu Jia, Jianzhuang Liu, and Qi Tian. Joint demosaicing and denoising with self guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2240–2249, 2020. 1

[18] Peyman Milanfar. A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE signal processing magazine*, 30(1):106–128, 2012. 1

[19] Zeeshan Nadir, Michael S. Brown, Mary L. Comer, and Charles A. Bouman. Tomographic reconstruction of flowing gases using sparse training. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 1733–1737, 2014. 1

[20] Zeeshan Nadir, Charles A. Bouman, Kristin M. Rice, and Michael S. Brown. A hybrid prior model for tunable diode laser absorption tomography. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 1188–1192, 2018. 1

[21] Qualcomm. Snapdragon Neural Processing Engine SDK. https://developer.qualcomm.com/sites/default/files/docs/snpe/index.html, 2024. [Online; accessed 29-January-2024]. 5

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 2, 4

[23] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992. 3

[24] Hiroyuki Takeda, Sina Farsiu, and Peyman Milanfar. Kernel regression for image processing and reconstruction. *IEEE Transactions on image processing*, 16(2):349–366, 2007. 1

[25] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, pages 839–846. IEEE, 1998. 1

[26] Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE global conference on signal and information processing*, pages 945–948. IEEE, 2013. 1

[27] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, pages 1398–1402 Vol.2, 2003. 3

[28] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 3, 5

[29] Wenzhu Xing and Karen Egiazarian. End-to-end learning for joint image demosaicing, denoising and super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3507–3516, 2021. 1

[30] Jun Xu, Lei Zhang, David Zhang, and Xiangchu Feng. Multi-channel weighted nuclear norm minimization for real color image denoising. In *Proceedings of the IEEE international conference on computer vision*, pages 1096–1104, 2017. 1

[31] Guoshen Yu and Guillermo Sapiro. Dct image denoising: a simple and effective image denoising algorithm. *Image Processing On Line*, 1:292–296, 2011. 1

[32] 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, 2017. 5

[33] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018. 1

[34] Ruoqiao Zhang, Charles A Bouman, Jean-Baptiste Thibault, and Ken D Sauer. Gaussian mixture markov random field for image denoising and reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 1089–1092. IEEE, 2013. 1

[35] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *2011 international conference on computer vision*, pages 479–486. IEEE, 2011. 1