Learning Deformable Kernels Predict Network for Burst Images Denoising

Wang jincai¹, *, Cai yong ¹

¹School of physics and microelectronics, hunan university, changsha, China

*18774818221@163.com

Abstract: Mobile phones are becoming more common, people has higher requirement about the quality of the pictures that captured by handheld device, in order to reduce the noise in image, we studied the prediction network (KPN) denoising algorithm, to overcome the disadvantage of low efficiency of traditional convolution receptive field, a deformable kernel prediction network (DEF_KPN) algorithm for burst images denoising is proposed in this paper. The deformable convolution structure is used to preprocess the noise image, so that the kernel prediction network can generate a pixel-by-pixel filter core more suitable for the image structure, so as to make the image contour more clearer after denoised. We obtain bayer image by anti-ISP method, and then synthesizing the training data by jitter clipping and adding poissonian-gaussian noise. Applying Adam algorithm and annealing strategy to train neural network to convergence. The experiment results show that: the DEF_KPN algorithm proposed in this paper is superior to the KPN algorithm in both synthetic noise data and real noise data. In terms of synthetic noise data, the PSNR index is improved by about 0.5db. On the real noise data, PSNR index was increased by 0.5dB. The SSIM index has been improved by 0.005. Experiments on burst mobile phone bayer images illustrate that: The neural network trained by our method can successfully de-noise the multi-frame images taken by the phone.

1. Introduction

Image denoising is very important in low level vision and is still an active topic. It also plays an indispensable role in military, industry, remote sensor, satellite and other fields. Image denoising is an ill-posed problem, the aim is to estimate a latent image X from an observed degraded image Y. The degradation model is Y=X+V, and V is generally assumed to be white gaussian noise. To solve this problem, there are many attempts by traditional denoising algorithms, such as NLM[1] and BM3D[2] algorithms which based on structural similarity. There is also k-svd [3] algorithm based on sparse prior. And WNNM [4] algorithm based on low-rank approximation. However, their denoising effect depends on a specific prior or manual input parameters, which takes a long time and is not applicable in complex scenarios.

With the acceleration of GPU in convolution computation, the learn-based convolution neural network algorithm is also applied to image denoising, such as DnCNN,MLP,TNRD,FFDNET[5-8]. These algorithms learn a mapping from a noisy image to a noiseless image by training amount of data. However, no matter how to optimize the neural network architecture, the denoising performance always be restricted by rectangular receptive field of characteristic figure: due to the shape of the object is not rectangular, So the receptive field will contain some independent pixels. Recently, Jifeng Dai proposed a deformable convolution structure [9] to solve this problem. After learning the deformable convolution
structure, the perceptive field can be shifted to the surroundings with the object.

Some multi-frame denoising algorithms have been proposed via that multi-frame images contain more information than single-frame image. Such as the HDR + algorithm [10] proposed by Samuel et al., and the generalize of BM3D algorithm VBM4D [11]. Ben Mildenhall also proposed KPN [12] algorithm, whose main idea is to predict the adaptive checking of multi-frame images per pixel space for adaptive convolution denoising. Unfortunately, the performance of KPN algorithm is also limited by rectangular receptive field.

To solve the above problems, a deformable kernel prediction network algorithm (DEF_KPN) for multi-frame color image denoising is proposed in this paper. Though its structure is similar to the original KPN, but their operating mode is very different: we learn offset of image by little deformable convolution network, then concat with original image and put into nuclear prediction network to extract per-pixel filter core, and then use the generated per-pixel filter to denoise image, this practice is similar to NLM algorithm, to weight weighted denoising of similar pixels, this method not only has denoising flexibility and denoising performance boost, but also ensure the denoising speed; Secondly, we apply anti-ISP to obtain the bayer image to train network, which solves the problem that the current algorithm cannot restore bayer image. The experimental results show that both the PSNR and SSIM[13] indexes of DEF_KPN algorithm are superior to KPN algorithm in both synthetic noise and real noise.

2. The design of algorithm

2.1. deformable convolution

Different from the general convolution kernel sampling is square, the sampling points of the deforming convolution kernel are attached around the object, which is consistent with the structural similarity of traditional denoising methods. The deformation of the sampling grid can be replaced by the sampling point plus the offset of the sampling point.

For the convolution kernel of size 3*3, the coordinates of the sampling point can be denotes as \( R = \{(-1, -1), (-1, 0), \ldots, (0, 1), (1, 1)\} \). After convolution, each point \( P \) of feature graph \( F \) can be represented as

\[
F(P) = \sum_{P_n \in R} W(P_n) \cdot X(P + P_n)
\]

(1)

Where \( W \) denotes the convolution kernel weight and \( X \) denotes the input image. Accordingly, Each point of the characteristic graph after deformable convolution can be represented as

\[
F(P) = \sum_{P_n \in R} W(P_n) \cdot X(P + P_n + \Delta P_n)
\]

(2)

\( \Delta P_n \) on behalf of each pixel offset. As shown in figure 1, firstly, the conventional convolution layer learns the offset of each pixel in the x and y directions, and then obtains the offset by bilinear interpolation.

\[\text{Figure 1. The architecture of deformable convolutional.}\]

2.2. The design of network structure

The input of our denoising network are a set of bayer image with noise \((X_1, X_2, \ldots, X_N)\), and include estimation of noise bayer image \((X_E)\).

\[X_E = \sigma_r + \sigma_s \cdot X_1\]

(3)
The front of the network is a subnetwork with deformable convolution structure, which contains a layer convolution, and its channels is 2 times of the input image frames, the image’s offset of x direction and y direction obtained by input image after convolution layer, and then synthesis the figure map by double linear interpolation, the original image after processing by offset feature maps is the image with offset, we concat original image and the image with offset put into the u-net [14] network, See figure 2 for details.

Consistent with KPN, the network finally outputs a group of pixel by pixel filtering core (K*K) of images, and then noise bayer images with offset are processed by the filtering core to get a group of clean bayer images. Then aligned and averaged to produce a clean bayer image. The number of images in our network N=8, the size of the filter core K=5.

2.3. loss function

Although the input and output of our network are bayer images in linear space, the images will eventually be observed in color domain, so our loss function adopts the image in RGB domain. As shown in figure 3, the noiseless image Y_1, Y_2 ...Y_N after ISP pipeline post-processing is a set of color images(\(\vec{V}_1, \vec{V}_2 \ldots \vec{V}_N\)) in RGB color domain. The Y is averaged by noiseless bayer image Y_1, Y_2 ...Y_N, and then through the ISP post-processing pipeline converted to \(\vec{Y}\) which in RGB color domain. Accordingly, our label X is obtained from a noiseless bayer image processed by an ISP pipeline. Our basic loss function use the square of L2 norm of pixel difference plus the L1 norm of difference of pixel gradient.

![Figure 2. The architecture of deformable convolutional neural network.](image)

![Figure 3. The algorithm flow.](image)

Owing to the noiseless image Y averaged by bayer image Y_1, Y_2 ... Y_N, If no alignment between bayer image, the clean image will fuzzy, and loss function convergence value is bigger. We introduce the annealing loss function like KPN to improve this phenomenon. The final loss function is:

\[
L(\vec{V}, X) = \|\vec{V} - X\|_2^2 + \|\nabla \vec{V} - \nabla X\|_1
\]

\[
L(\vec{V}, X, t) = L(\vec{V}, X) + \mu \alpha^t \sum_{i=1}^{N} L(\vec{V}_i, X)
\]

The \( Y = \sum_{i=1}^{N} \vec{V}_i \) as anneal term, t is the training steps, \( \mu \alpha^t \) is annealing time control table, we also take \( \mu = 100, \alpha = 0.9998 \).

3
2.4. synthesis of training data
We use method that provided by TimBrooks in [15] to make sRGB image to bayer matrix through a series of anti-ISP processing. After verification, The bayer matrix obtained by this method is similar to matrix which captured by camera. Subsequently, we added Poissonian-Gaussian [16] noise to bayer matrix, and the clean bayer image processed by ISP pipeline to creat a label. According to the jitter characteristics of images captured by the handheld device, N frames bayer noise image were cut out from one bayer noise image as network inputs.

Add noise: we counted the distribution of noise parameters in DND[17] as follows: The range of noise we added is shown in the orange shaded part of figure 4, which covers the noise parameter points of the DND data set, among which the orange points are test points.

We use tensorflow to implement our algorithm, using Adam optimization algorithm [18] training network to convergence, learning rate we set to $10^{-4}$, we choose DIV2K sRGB image data sets in DIV2K_train_HR to synthesis training data by method which 2.4 stated, and inputs image block size is 128 * 128, batch is 4, and training 400,000 steps in total.

3. Experimental results and analysis

3.1. test on composite noise
We selected the image in DIV2K_valid_HR for the experiment on composite noise. As shown in figure 4, a total of 4 noise levels were selected to test (orange points in figure 4). DIV2K_valid_HR contains a total of 100 images. For each noise level, we took the average of the 100 images, and we used PSNR and SSIM to evaluate the image quality. To avoid unfairness, we trained the KPN network in the same way, and we also compared it with the NLM, BM3D and VBM4D algorithms. All algorithms are denoised on the bayer image, and then converse into RGB domain to evaluation. The results are shown in table 1. The best results of PSNR index in the table have been marked in orange. The best SSIM results are shown in yellow. Compared with KPN, it can be seen that the algorithms we proposed have varying improvement degree on PSNR (about 0.5dB).

|                | PSNR=36.7095 | PSNR=31.3091 | PSNR=26.1019 | PSNR=22.9394 |
|----------------|--------------|--------------|--------------|--------------|
| NLM            | 31.7057      | 0.816        | 30.6828      | 0.796        | 29.1939      | 0.7614       | 28.4441      | 0.7267       |
| BM3D           | 32.7668      | 0.8517       | 32.084       | 0.8382       | 30.587       | 0.8021       | 28.2852      | 0.6956       |
| VBM4D          | 37.2231      | 0.9423       | 34.6176      | 0.9083       | 32.4857      | 0.8702       | 31.2843      | 0.8235       |
| KPN            | 37.5497      | 0.968        | 35.6755      | 0.9415       | 34.195       | 0.9149       | 33.1082      | 0.8793       |
| DEF_KPN        | 38.4065      | 0.9692       | 36.3097      | 0.9432       | 34.3572      | 0.9142       | 33.1436      | 0.878        |

3.2. test on real noise data
We selected the DND data set for the benchmark test. The DND test set included a pair of low-light images with real noise and high-light noiseless images. We submit the bayer image denoising results of KPN and our improved algorithm DEF_KPN. The test results of DND official website are shown in table 2:
Table 2. Result on real noise test.

|        | bayer | RGB |
|--------|-------|-----|
| KPN    | 44.3548 | 0.9693 | 36.1936 | 0.9207 |
| DEF_KPN | 45.1168 | 0.9722 | 36.6989 | 0.9261 |

It can be seen from table 2 that the DEF_KPN algorithm we proposed has higher PSNR indexes than the KPN algorithm in both bayer images and RGB images. The height is 0.7dB and 0.5dB respectively. SSIM also improved by 0.003 and 0.005.

Figure 5 shows the denoising effect of KPN and our algorithm DEF_KPN on the DND data set. The left side of figure 5 is the whole image, 5 (a) is the noise image, 5 (b) is the denoise results of KPN algorithm, and 5 (c) is the denoise results of our algorithm DEF_KPN. It can be seen that our result contains less noise than the result of KPN.

3.3. extend to multi-frame mobile phone images

We choosed a group of multi-frame images from the HDR+ data set which captured by Nexus5 for testing. As shown in Figure. 6, the left of image is one of the multi-frame images. The middle of the image are 8 blocks which we cut from image which has 8 frames, and the is denoised image. It can be seen that the algorithm we proposed can effectively denoising the real multi-frame noise image. Although the position of the female’s right arm is different in each frame, the result of the female’s right arm is very clear after denoised.

4. Conclusion

This paper draws on Deformable Convolutional Networks and Kernel Prediction Networks, and proposed network (DEF_KPN), and applied it in linear space frame bayer image denoising. The experiment proves that proposed denoise algorithm improves the PSNR index by about 0.5dB compared with the KPN algorithm in the RGB color domain, no matter how the experiment on the composite noise image or on the real noise image. And SSIM index is improved by 0.005 compared with KPN algorithm in real noise image’s experiment. The experiment of multi-frame image denoising taken by mobile phone shows that the network trained by our method can successfully denoise the multi-frame image of mobile phone.

References

[1] A. Buades, et al.. A non-local algorithm for image denoising [C]. CVPR, 2005.2: 60-65.
[2] K. Dabov, et al.. Image denoising by sparse 3-d transform-domain collaborative filtering[J]. IEEE Transactions on Image Processing, 2007,16(8):2080–2095.

[3] M. Elad et al.. Image denoising via sparse and redundant representations over learned dictionaries[J]. IEEE Transactions on Image Processing, 2006,15(12):3736–3745.

[4] S. Gu, et al.. Weighted nuclear norm minimization with application to image denoising[C] CVPR, 2014:2862–2869.

[5] K. Zhang, et al.. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising[J]. IEEE Transactions on Image Processing 2017, 26(7):3142–3155.

[6] Y. Chen, et al.. On learning optimized reaction diffusion processes for effective image restoration[C]. CVPR,2015: 5261–5269.

[7] H. C. Burger, C. J. Schuler, and S. Harmeling, Image denoising: Can plain neural networks compete with BM3D? [C]. CVPR, 2012:2392–2399.

[8] Kai Zhang, et al.. FFDnet: Toward a fast and flexible solution for cnn based image denoising. [J]. IEEE Transactions on Image Processing, 2018, 27(9):4608–4622.

[9] J. Dai, H. Qi, Y. Xiong, Y. Li, et al.. Deformable convolutional networks[C]. IEEE International Conference on Computer Vision,2017:764-773

[10] S. W. Hasinoff, et al.. Burst photography for high dynamic range and low-light imaging on mobile cameras [J]. ACM Transactions on Graphics, 2016,35(6): 192:1–192:12.

[11] M. Maggioni, et al.. Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms [J]. TIP, 2012,21(9):3952-3966.

[12] B. Mildenhall, et al.. Burst denoising with kernel prediction networks [C]. CVPR, 2018:2502–2510.

[13] WANG Z,BOVIK A C,SHEIKH H R. Image quality assessment: From error visibility to structural similarity[J].Image Processing,2004,13(4):600-612.

[14] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation[J]. MICCAI, 2015:234-241.

[15] T. Brooks, B. Mildenhall, T. Xue, et al.. Unprocessing images for learned raw denoising[C]. CVPR, 2019: 11036-11045.

[16] Samuel W. Hasinoff. Photon, poisson noise[J]. Computer Vision. 2014 :19-27.

[17] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs[C]. CVPR, 2017:1586-1595.

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization[J]. ICLR, 2015.