ISSN 2464-4617 (print)
SSN 2464-4625 (DVD)
Computer Science Research Notes
CSRN 3101
WSCG 2021 Proceedings

# Reduction of JPEG Artifacts using BSDEs

Dariusz Borkowski
Faculty of Mathematics and Computer Science
Nicolaus Copernicus University
Chopina 12/18, 87-100 Toruń, Poland
dbor@mat.umk.pl

Katarzyna Jańczak-Borkowska
Institute of Mathematics and Physics University of Science and Technology
al. prof. S. Kaliskiego 7, 85-789 Bydgoszcz, Poland
kaja@utp.edu.pl

## ABSTRACT

In this paper we propose a novel approach for reduction of JPEG artifacts using advanced techniques of stochastic calculus. In order to solve this problem we use backward stochastic differential equations (in short BSDEs) and the non local means method. In our algorithm we consider two processes. One of them has values in the image domain and determines pixels that will be involved in the reconstruction, the second one has values in the image codomain and gives weight to values of pixels. To calculate the weights, we use the idea of the patches similarity used in the non local means. Our experiments show that the new approach gives very good results and compares favourably with other methods.

## Keywords

Image reconstruction, JPEG artifacts, Stochastic differential equations, Non local means

## 1 INTRODUCTION

Lossy compression algorithms discard data that is least important to the recipient's visual sense, leaving the data of greater importance. The human eye is not sensitive to high frequency information, which is why most lossy image compression methods are implemented by quantization or approximation in the frequency domain. JPEG is a representative standard for lossy compression and is used worldwide. The JPEG compression standard splits the image, performs a discrete cosine transformation and then transformation coefficients are quantized and coded. The amount of data rejected is determined by the quality of compression.

There are many different techniques for the reconstruction of JPEG images. Early preliminary works [Ree84, Lis03, Lee04, Bre12, Wan13, Pou14, Gay15, Pan15] perform filtering to reduce only blocking artifacts. Many interesting works are done by low-pass filtering and frequency domain techniques [Cho98, Lee98, Lia02, Tri03, Abb06, Sin07, Sin11, Gol14]. Another popular method uses the concept of projection onto convex sets [Wee02, Gan03]. Several proposals have been made that make use of wavelet transforms [Hsu98, Cho00]. An important role in reducing of artifacts play the traditional noise reduction algorithms [Dab07, Foi07]. A different approach is based on multiple dictonary and machine learning methods [Cha13, Che15, Li16]. Dong et al. [Don14] first introduced a deep neural network to solve the problem of reduction of artifacts. The concept of deep neural networks has gained wide recognition and its later versions are successfully used to JPEG restoration [Don15, Svo16, Cav17, Zha17, Zha18, Ehr20, Kim20, Wan20].



In this paper we present a new method of artifacts reduction in the case of RGB images. We combine the idea of image denoising based on backward stochastic differential equations from [Bor17] and the concept of applying the non local means method to Feynman-Kac formula from the paper [Bor14].

The contribution of this work is twofold. We give a new noise reduction approach based on BSDE and non local means method. Moreover, we apply the obtained algorithm for JPEG reconstruction.

The rest of the paper is constructed as follows. In Section 2 we recall the method of denoising in terms of backward stochastic differential equations from [Bor17]. Section 3 provides a new method of restoration of the noisy image. Section 4 contains information about using an algorithm in order to remove JPEG

DOI:10.24132/CSRN.2021.3101.1

ISBN 978-80-86943-34-3

artifacts. Finally, in Section 5 experimental results and comparison to other methods are presented.

# 2 IMAGE RECONSTRUCTION BASED ON BSDES

## 2.1 Continuous model

Let $D$ be a bounded, convex domain in $\mathbf{R}^2$, $u : \overline{D} \to \mathbf{R}^3$ be an original RGB image and $u_0 : \overline{D} \to \mathbf{R}^3$ be the observed image of the form $u_0 = u + \eta$, where $\eta$ stands for a white Gaussian noise (independently added to all coordinates). The BSDE model to restoration of the image $u(x)$ is the following

$$\begin{cases} X_t = x + \int_0^t \sigma(s, X_s)\, dW_s + K_t^{\overline{D}}, \quad t \in [0,T], \\ Y_t = u_0(X_S) + \int_t^T c(s)(Y_s - u_0(X_s))ds - \\ \qquad\qquad \int_t^T Z_s\, dW_s, \quad t \in [0,T], \end{cases}$$

where $S < T$, $\{X_t\}_{t\in[0,T]}$ is a stochastic diffusion process, $\{W_t\}_{t\in[0,T]}$ is two-dimensional Wiener process, the term $\{K_t^{\overline{D}}\}_{t\in[0,T]}$ is the minimal push needed to keep process $X$ in $\overline{D}$, $\{Y_t\}_{t\in[0,T]}$ is the first component of the solution to the BSDE, $\{Z_t\}_{t\in[0,T]}$ is the second component of the solution to the BSDE and determines the measurability of the process $Y$,

$$\sigma(s,x) = \left[ \left(1 - \frac{c(s)}{c}\right) \theta_-(G_\gamma * u_0, x), \frac{c(s)}{c} \theta_+(G_\gamma * u_0, x) \right]$$

$$c(t) = \begin{cases} 0 & \text{if} \quad t < S \quad \text{or} \quad N(G_\gamma * u_0, x) < d, \\ c & \text{if} \quad t \geq S \quad \text{and} \quad N(G_\gamma * u_0, x) \geq d, \end{cases}$$

$\theta_+(u,x) \in \mathbf{R}^2$, $\theta_-(u,x) \in \mathbf{R}^2$ and $N(u,x) \in \mathbf{R}$ is Di Zenzo [Diz86, Der02] geometry of the RGB image $u((x_1,x_2)) = (R((x_1,x_2)), G((x_1,x_2)), B((x_1,x_2)))$ at point $x$

$$\theta_\pm(u,x) = \frac{\nu_\pm(u,x)}{|\nu_\pm(u,x)|},$$

$$N(u,x) = \sqrt{\lambda(u,x)},$$

$$\lambda(u,x) = \frac{\sqrt{\chi(u,x)} + \frac{\partial R}{\partial x_1}(x)^2 + \frac{\partial G}{\partial x_1}(x)^2 + \frac{\partial B}{\partial x_1}(x)^2}{2} + \frac{\frac{\partial R}{\partial x_2}(x)^2 + \frac{\partial G}{\partial x_2}(x)^2 + \frac{\partial B}{\partial x_2}(x)^2}{2}$$

$$\nu_\pm(u,x) = \begin{bmatrix} 2\Big(\frac{\partial R}{\partial x_1}(x)\frac{\partial R}{\partial x_2}(x) + \frac{\partial G}{\partial x_1}(x)\frac{\partial G}{\partial x_2}(x) \\ + \frac{\partial B}{\partial x_1}(x)\frac{\partial B}{\partial x_2}(x)\Big) \\ \\ \frac{\partial R}{\partial x_2}(x)^2 + \frac{\partial G}{\partial x_2}(x)^2 + \frac{\partial B}{\partial x_2}(x)^2 - \frac{\partial R}{\partial x_1}(x)^2 \\ - \frac{\partial G}{\partial x_1}(x)^2 - \frac{\partial B}{\partial x_1}(x)^2 \pm \sqrt{\chi(u,x)} \end{bmatrix}$$

$$\chi(u,x) = \frac{\partial R}{\partial x_1}(x)^2 + \frac{\partial G}{\partial x_1}(x)^2 + \frac{\partial B}{\partial x_1}(x)^2 - \frac{\partial R}{\partial x_2}(x)^2 - \frac{\partial G}{\partial x_2}(x)^2 - \frac{\partial B}{\partial x_2}(x)^2 + +4\left(\frac{\partial R}{\partial x_1}(x)\frac{\partial R}{\partial x_2}(x) + \frac{\partial G}{\partial x_1}(x)\frac{\partial G}{\partial x_2}(x) + \frac{\partial B}{\partial x_1}(x)\frac{\partial B}{\partial x_2}(x)\right)^2$$

$G_\gamma$ is a $3 \times 3$ Gaussian kernel and $S \in \mathbf{R}_+ < T \in \mathbf{R}_+$, $d \in \mathbf{R}_+$, $c \in \mathbf{R}_+$ are parameters of the method.

For a fixed pixel $x$ we consider a certain BSDE equation. The values of the process $X$ determines pixels from domain of the image $\overline{D}$ which we will use in process reconstruction. We can say that this process determines neighbourhood of the pixel $x$ (with irregular shape). The reconstructed value $u(x)$ is the sum of pixels from its neighbourhood multiplied by some weights. The weight values are determined by the process $Y$. Appropriate definition of the function $c(t)$ allows as to give weight values (also negative) which depend on direction and distance from reconstructed pixel.

Parameter $T$ defines the size of the neighbourhood used in the reconstruction procedure. We deblur from time $T$ to $S$ and smooth out from $S$ to 0. The parameter $d$ determines which pixels will be reconstructed with using smoothing model and which with using enhancing model. The parameter $c$ is responsible for effect of edge sharpening. The values of all parameters depend on standard noise deviation $\rho$.

## 2.2 Algorithm

Consider a time discretization $0 = t_0 < t_1 < \ldots < t_j \leq S < t_{j+1} < \ldots < t_m = T, t_i - t_{i-1} = \frac{T}{m}$. In the first step we generate a trajectory of the stochastic process $X$ for $k = 0, 1, \ldots, m-1$ using the Euler formula [Slo01]

$$\begin{aligned} X_0 &= x, \\ X_{t_k} &= \Pi_{\overline{D}}[X_{t_{k-1}} + \sigma(t_{k-1}, X_{t_{k-1}})(W_{t_k} - W_{t_{k-1}})], \end{aligned} \tag{1}$$

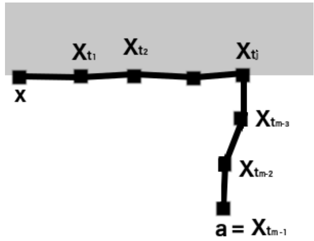
where $\Pi_{\overline{D}}(x)$ denotes a projection of $x$ on the set $\overline{D}$ and $W$ is a Wiener process. The difference $W_{t_k} - W_{t_{k-1}}$ we approximate using random number generator. Since $W_t$ is 2 – dimensional, the value of the difference $W_{t_k} - W_{t_{k-1}}$ is equal to two independent values (vector) obtained with a generator of the normal distribution $\mathcal{N}(0, t_k - t_{k-1})$.

Example of the sequence $X$ given by formula (1) is shown on Figure 1 (a). The image has two areas: gray and white. The process $X$ starts from the reconstructed pixel $x$ located on the edge. Next, the process $X$ has values along the edge until time $t_j$, then after time $t_j$ moves towards the vector $\theta_+$ (gradient vector).

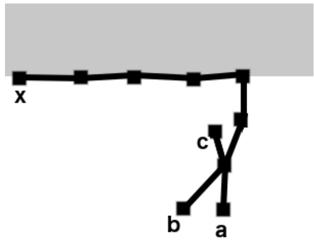
Now, we can define the process $Y$. From a definition of BSDE [Bor17] it starts from

$$Y_{t_m} = u_0(X_{t_j})$$

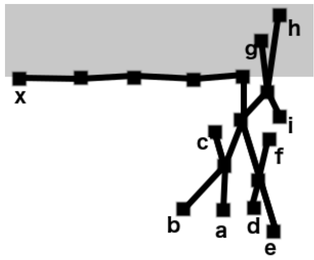
and then we backwardly count values $Y_{t_{m-1}}, Y_{t_{m-2}}, \ldots, Y_0$




(a)




(b)




(c)

Figure 1: Trajectory of the process $X$.

$$Y_{t_k} = \mathbf{E}\left[Y_{t_{k+1}} | \mathscr{F}_{t_k}\right] + \frac{T}{m} c(t_k) \left(\mathbf{E}\left[Y_{t_{k+1}} | \mathscr{F}_{t_k}\right] - u_0(X_{t_k})\right),$$

$k = m-1, m-2, \ldots, 0$, where by $\mathbf{E}$ we denote the expected value and by $\mathscr{F}_t$ the filtration generated by the discretization of the Wiener process [Ma02].

For $k = m-1$ we have

$$Y_{t_{m-1}} = \mathbf{E}\left[Y_{t_m} | \mathscr{F}_{t_{m-1}}\right]$$

$$+\frac{T}{m} c(t_{m-1}) \left(\mathbf{E}\left[Y_{t_m} | \mathscr{F}_{t_{m-1}}\right] - u_0(X_{t_{m-1}})\right).$$

Note that $Y_{t_m}$ is $\mathscr{F}_{t_{m-1}}$ measurable. Therefore

$$Y_{t_{m-1}} = Y^a_{t_{m-1}} = u_0(X_{t_j}) + \frac{T}{m} c(t_{m-1}) \left(u_0(X_{t_j}) - u_0(a)\right).$$

Next, for $k = m-2$

$$Y_{t_{m-2}} = \mathbf{E}\left[Y_{t_{m-1}} | \mathscr{F}_{t_{m-2}}\right]$$

$$+\frac{T}{m} c(t_{m-2}) \left(\mathbf{E}\left[Y_{t_{m-1}} | \mathscr{F}_{t_{m-2}}\right] - u_0(X_{t_{m-2}})\right).$$

Since $Y_{t_{m-1}}$ is not $\mathscr{F}_{t_{m-2}}$ measurable, we need to count $\mathbf{E}\left[Y_{t_{m-1}} | \mathscr{F}_{t_{m-2}}\right]$ by using Monte Carlo method with $M$ iterations. We start $M$-times from point $X_{t_{m-2}}$. Example for $M = 3$ is shown on Figure 1 (b) (in practise we need to use about 10 iterations). As before we count $Y^b_{t_{m-1}}$ and $Y^c_{t_{m-1}}$ and then

$$\mathbf{E}\left[Y_{t_{m-1}} | \mathscr{F}_{t_{m-2}}\right] \approx \frac{Y^a_{t_{m-1}} + Y^b_{t_{m-1}} + Y^c_{t_{m-1}}}{3},$$

$$Y_{t_{m-2}} \approx Y^{a,b,c}_{t_{m-2}} = \frac{Y^a_{t_{m-1}} + Y^b_{t_{m-1}} + Y^c_{t_{m-1}}}{3} +$$

$$\frac{T}{m} c(t_{m-2}) \left(\frac{Y^a_{t_{m-1}} + Y^b_{t_{m-1}} + Y^c_{t_{m-1}}}{3} - u_0(X_{t_{m-2}})\right).$$

Next, value for $t_{m-3}$ is equal to

$$Y_{t_{m-3}} = \mathbf{E}\left[Y_{t_{m-2}} | \mathscr{F}_{t_{m-3}}\right]$$

$$+\frac{T}{m} c(t_{m-3}) \left(\mathbf{E}\left[Y_{t_{m-2}} | \mathscr{F}_{t_{m-3}}\right] - u_0(X_{t_{m-3}})\right)$$

and again, similarly to $Y_{t_{m-2}}$ we need to count it by using Monte Carlo method. For $M = 3$ (see Figure 1 (c)) we have the formula

$$Y_{t_{m-3}} \approx Y^{a,b,c,d,e,f,g,h,i}_{t_{m-3}} = \frac{Y^{a,b,c}_{t_{m-2}} + Y^{d,e,f}_{t_{m-2}} + Y^{g,h,i}_{t_{m-2}}}{3} +$$

$$+\frac{T}{m} c(t_{m-3}) \left(\frac{Y^{a,b,c}_{t_{m-2}} + Y^{d,e,f}_{t_{m-2}} + Y^{g,h,i}_{t_{m-2}}}{3} - u_0(X_{t_{m-3}})\right).$$

The above reasoning is repeated until we determine $Y_0$ which is a reconstructed value i.e. $u(x)$.

# 3 MODIFICATION BASED ON FEYNMAN-KAC FORMULA AND NON LOCAL MEANS

Note, that for times $0 < t_0 < t_1 < \ldots < t_j$ and from definition of the function $c(t)$ ($c(t) = 0$, for $t < S$) the algorithm described in the previous section works as follows

$$Y_{t_{j-1}} = \mathbf{E}\left[Y_{t_j} | \mathscr{F}_{t_{j-1}}\right] \approx \frac{1}{M} \sum_{i=1}^{M} Y^{\omega_i}_{t_j} = \sum_{i=1}^{M} \frac{1}{M} Y^{\omega_i}_{t_j},$$

$$Y_{t_{j-2}} \approx \sum_{i=1}^{M} \frac{1}{M} Y^{\omega_i}_{t_{j-1}},$$

$$\vdots$$

$$Y_{t_0} \approx \sum_{i=1}^{M} \frac{1}{M} Y^{\omega_i}_{t_1}$$

The above approximation is based on Feynman-Kac formula, which means that each value of pixel $Y^{\omega_i}_{t_k}$,

DOI:10.24132/CSRN.2021.3101.1

ISBN 978-80-86943-34-3

$k = 1, 2, \ldots, j$ is weighted with the same value $\frac{1}{M}$. But since pixels have different colours we may consider them with different weights depending on their neighbourhood. We follow the non local means algorithm [Bua05] and propose to think of weights that depend on patches similarity i.e.

$$Y_{t_{j-1}} \approx \sum_{i=1}^{M} w(B_{x,r}, B_{X_{t_j},r}) Y_{t_j}^{\omega_i},$$

$$Y_{t_{j-2}} \approx \sum_{i=1}^{M} w(B_{x,r}, B_{X_{t_{j-1}},r}) Y_{t_{j-1}}^{\omega_i},$$

$$\vdots$$

$$Y_{t_0} \approx \sum_{i=1}^{M} w(B_{x,r}, B_{X_{t_1},r}) Y_{t_1}^{\omega_i},$$

where by $B_{x,r}$ we denote $(2r+1) \times (2r+1)$ RGB pixels centered at point $x$ and by $w(\cdot,\cdot)$ we denote a weight function defined in [Bua11].

For times $S < t_{j+1} < \ldots < t_m = T$ the function $c(t)$ is greater than zero which means that we have negative weights (to obtain the enhancement effect) and the algorithm remains unchanged.

## 4 REDUCTION OF JPEG ARTIFACTS

Note, that if we apply values of parameters as default values recommended by the authors of papers [Bua11, Bor14, Bor17] then algorithm still has one parameter: $\rho$. In this case algorithm can be used successfully to reconstruction of Gaussian noisy images with given standard deviation of the noise.

In the case of reduction of JPEG artifacts we have to combine the $\rho$ parameter with the compression quality $q$. In JPEG standards the compression quality $q$ is always known and is expressed as a percentage. An image at 100% quality has no loss. We propose the following formula to count the $\rho$ parameter:

$$\rho = \max\{-0.3q + 20, 0\}, \qquad (2)$$

where $q$ is a JPEG compression quality of the image. Choosing this function, we followed the principle of maximizing the Peak Signal to Noise Ratio (in short PSNR) :

$$\text{PSNR}(u, \hat{u}) = 10 \log_{10} \left( \frac{255^2}{\text{MSE}(u, \hat{u})} \right),$$

where $u$, $\hat{u}$ denote the original and the restored RGB image and $\text{MSE}(u, \hat{u})$ is a mean square error between $u$ and $\hat{u}$. To determine formula (2) we used several standard test images. For each, fixed $q \in \{5, 10, 20, 30, 40, 50\}$ and for each test image we calculated $\rho$ at which we have the maximum PSNR. These values were averaged – for each $q$ we obtained a mean value of $\rho$. Finally, for these data we used the linear regression model to count the linear function $-0.3q + 20$.

Figure 2: Test RGB images: Toysflash $912 \times 684$, Lighthouse $480 \times 640$, Gantry $400 \times 264$, Onion $198 \times 135$.

## 5 EXPERIMENTAL RESULTS

In this section we present experimental results illustrating the difference between our algorithm and other methods of removing artifacts: SADCT [Foi07], CBM3D [Dab07], ARCNN [Don15], DNCNN [Zha17] and BSDE [Bor17] as the parent method for our approach. We use the MATLAB implementation of compared methods: SADCT [Foi20b], CBM3D [Foi20a], ARCNN [Yuk20], DNCNN [Jpe20]. Parameters of these approaches were set to the default values as recommended by the authors. Some results for our evaluation experiments are presented in Fig. 3, Fig. 4, Fig. 5, Fig. 6, Fig. 7. The results refer to RGB colour images *Toysflash*, *Lighthouse*, *Gantry*, *Onion* corrupted with the JPEG compression algorithm with different values of quality. The maximum values of Peak Signal to Noise Ratio and Structural SIMilarity (in short SSIM) index [Wan04] obtained using tested methods are given in tables: Table 2, Table 3. The reconstruction time of our method on Intel Core i7 is shown in the Table 1.

Table 1: Time of the reconstruction (in seconds) of proposed method. It has been tested for 2 × CPU 1,7 GHz.

| Image\ JPEG Quality | 10 | 20 | 30 | 40 |
|---|---|---|---|---|
| Lighthouse $480 \times 640$ | 4.07 | 4.53 | 4.45 | 5.86 |
| Toysflash $912 \times 684$ | 7.65 | 8.02 | 8.46 | 9.50 |
| Onion $198 \times 135$ | 1.30 | 1.40 | 1.33 | 1.51 |
| Gantry $400 \times 264$ | 2.23 | 2.40 | 2.32 | 2.50 |

The analysis of the measures of image quality from the Table 2 shows that the new method performs better, especially in the case of low value of JPEG quality. In particular, our modification of the algorithm by adding weights improves results of the parent method [Bor17]. Interesting examples are Fig. 3 and Fig. 4 in which one should look at artifacts marked with black arrows. In the Fig. 3 we can see ringing artifacts. These artifacts are mainly due to the coarse quantization of the high-frequency DCT coefficients, making the decompressed image to exhibit noisy patterns known as ringing or mosquito noise near the edge. In the Fig. 4 we can see blocking artifacts, which are mainly due to the coarse quantization of low-frequency DCT coefficients yielding decompressed image look like a mosaic at smooth regions [Ozt07]. Our method removed artifacts very well in both cases.

It can be noticed from tables that the proposed method and SADCT give similar results. However looking at the pictures Fig. 5, Fig. 6, Fig. 7 we see the visual difference between these two methods. After using SADCT, artifacts at the edges are still visible but after using the proposed method these artifacts are smoothed out.

# 6 CONCLUSION

In this paper we present a new method for reducing of JPEG artifacts based on backward stochastic differential equations. The main purpose of this work is to present a new mathematical tool to restore digital images that are qualitatively indistinguishable from other approaches. It seems that further exploration of this tool will also improve the time complexity of this stochastic algorithm.

# 7 REFERENCES

[Abb06] Abboud, I. Deblocking in BDCT image and video coding using a simple and effective method. Information Technology Journal, No. 5(3), pp. 422-426, 2006.

[Bor14] Borkowski, D., Jakubowski, A., and Jańczak-Borkowska, K. Feynman-Kac formula and restoration of high iso images. International Conference on Computer Vision and Graphics, pp. 100-107, 2014.

[Bor17] Borkowski, D. and Jańczak-Borkowska, K. Image denoising using backward stochastic differential equations. International Conference on Man-Machine Interactions, pp. 185-194, 2017.

[Bre12] Bredies, K. and Holler, M. A total variation–based jpeg decompression model. SIAM Journal on Imaging Sciences, No. 5(1), pp. 366-393, 2012.

[Bua05] Buades, A., Coll, B., and Morel, J. M. A non local algorithm for image denoising. IEEE Computer Vision and Pattern Recognition, No. 2, pp. 60-65, 2005.

[Bua11] Buades, A., Coll, B., and Morel, J. M. (2011). Non-local means denoising. Image Processing On Line, No. 1, pp. 208-212, 2011.

[Cav17] Cavigelli, L., Hager, P., and Benini, L. Cascnn: A deep convolutional neural network for image compression artifact suppression. International Joint Conference on Neural Networks, pp. 752-759, 2017.

[Cha13] Chang, H., Ng, M. K., and Zeng, T. Reducing artifacts in jpeg decompression via a learned dictionary. IEEE Transactions on Signal Processing, No. 62(3), pp. 718-728, 2013.

[Che15] Chen, Y., Yu, W., and Pock, T. On learning optimized reaction diffusion processes for effective image restoration. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5261-5269, 2015.

[Cho00] Choi, H. and Kim, T. Blocking-artifact reduction in block-coded images using wavelet-based subband decomposition. IEEE Transactions on Circuits and Systems for Video Technology, No. 10( 5), pp. 801-805, 2000.

[Cho98] Chou,J., Crouse, M., and Ramchandran, K. A simple algorithm for removing blocking artifacts in block-transform coded images. IEEE Signal Processing Letters, No. 5(2), pp. 33-35, 1998.

[Dab07] Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE Transactions on Image Processing, No. 16(8), pp. 2080-2095, 2007.

[Der02] Deriche, R. and Tschumperlé, D. Diffusion pde's on vector-valued images: local approach and geometric viewpoint. IEEE Signal Processing Magazine, No. 19(5), pp. 16-25, 2002.

[Diz86] Di Zenzo, S. A note on the gradient of a multi-image. Computer Vision, Graphics, and Image Processing, No. 33(1), pp. 116-125, 1986.

[Don15] Dong, C., Deng, Y., Change Loy, C., and Tang, X. Compression artifacts reduction by a deep convolutional network. Proceedings of the IEEE International Conference on Computer Vision, pp. 576-584, 2015.

[Don14] Dong, C., Loy, C. C., He, K., and Tang, X. Learning a deep convolutional network for image super-resolution. European Conference on Computer Vision, pp. 184-199, 2014.

[Ehr20] Ehrlich M., Davis L., Lim SN., and Shrivastava A. Quantization guided JPEG artifact correction. Lecture Notes in Computer Science, No. 12353, 2020.

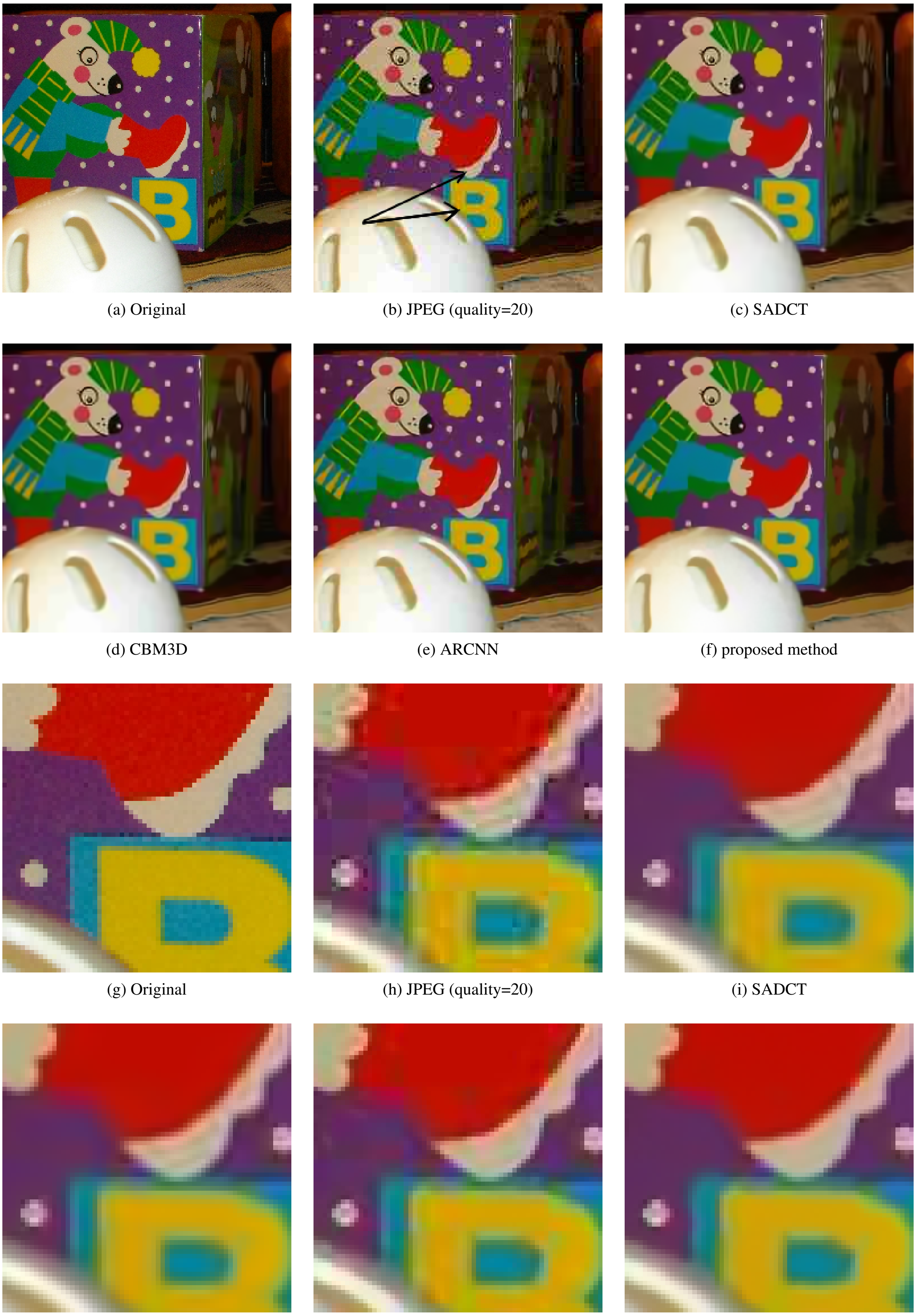
[Foi07] Foi, A., Katkovnik, V., and Egiazarian, K. Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. IEEE transactions on Image Processing, No. 16(5), pp. 1395-1411, 2007.

Table 2: PSNR

| Test Image | JPEG Quality | JPEG | SADCT [Foi07] | CBM3D [Dab07] | ARCNN [Don15] | DNCNN [Zha17] | BSDE [Bor17] | proposed method |
|---|---|---|---|---|---|---|---|---|
| Toysflash $912 \times 684$ | 10 | 27.63 | 28.50 | 28.61 | 28.14 | 27.89 | 28.18 | **28.68** |
| | 20 | 29.88 | 30.48 | 30.68 | 30.25 | 30.18 | 30.17 | **30.69** |
| | 30 | 30.99 | 31.45 | **31.69** | 31.39 | 31.28 | 31.16 | 31.67 |
| | 40 | 31.70 | 32.07 | **32.32** | 32.03 | 31.98 | 31.84 | 32.30 |
| Lighthouse $480 \times 640$ | 10 | 26.60 | 27.67 | 27.51 | 27.38 | 27.08 | 27.13 | **27.69** |
| | 20 | 28.80 | 29.54 | 29.56 | 29.45 | 29.34 | 28.91 | **29.64** |
| | 30 | 30.02 | 30.54 | 30.69 | 30.71 | 30.54 | 30.04 | **30.72** |
| | 40 | 30.84 | 31.24 | 31.43 | 31.36 | 31.29 | 30.87 | **31.45** |
| Onion $198 \times 135$ | 10 | 26.17 | 27.45 | 27.37 | 26.69 | 26.47 | 27.03 | **27.48** |
| | 20 | 28.25 | 29.10 | 29.00 | 28.70 | 28.57 | 28.56 | **29.17** |
| | 30 | 29.46 | 30.30 | 30.23 | 29.89 | 29.80 | 29.70 | **30.36** |
| | 40 | 30.10 | **30.87** | 30.76 | 30.48 | 30.41 | 30.28 | **30.87** |
| Gantry $400 \times 264$ | 10 | 23.15 | 24.06 | 24.01 | 23.80 | 23.53 | 23.56 | **24.10** |
| | 20 | 25.05 | 25.87 | 25.85 | 25.67 | 25.48 | 25.47 | **25.92** |
| | 30 | 25.99 | **26.77** | 26.71 | 26.59 | 26.35 | 26.38 | 26.61 |
| | 40 | 26.68 | **27.51** | 27.42 | 27.16 | 27.01 | 27.08 | 27.13 |

Table 3: SSIM

| Test Image | JPEG Quality | JPEG | SADCT [Foi07] | CBM3D [Dab07] | ARCNN [Don15] | DNCNN [Zha17] | BSDE [Bor17] | proposed method |
|---|---|---|---|---|---|---|---|---|
| Toysflash $912 \times 684$ | 10 | 0.9100 | 0.9273 | 0.9258 | 0.9163 | 0.9139 | 0.9186 | **0.9288** |
| | 20 | 0.9443 | 0.9529 | 0.9537 | 0.9431 | 0.9466 | 0.9485 | **0.9539** |
| | 30 | 0.9556 | 0.9614 | **0.9625** | 0.9557 | 0.9571 | 0.9580 | 0.9620 |
| | 40 | 0.9620 | 0.9663 | **0.9674** | 0.9615 | 0.9633 | 0.9640 | 0.9672 |
| Lighthouse $480 \times 640$ | 10 | 0.9062 | **0.9232** | 0.9180 | 0.9167 | 0.9129 | 0.9108 | 0.9198 |
| | 20 | 0.9490 | **0.9621** | 0.9586 | 0.9550 | 0.9544 | 0.9506 | 0.9600 |
| | 30 | 0.9617 | **0.9693** | 0.9687 | 0.9670 | 0.9660 | 0.9622 | 0.9689 |
| | 40 | 0.9685 | **0.9748** | 0.9742 | 0.9709 | 0.9711 | 0.9692 | 0.9747 |
| Onion $198 \times 135$ | 10 | 0.9419 | 0.9600 | 0.9580 | 0.9486 | 0.9458 | 0.9551 | **0.9601** |
| | 20 | 0.9606 | 0.9711 | 0.9701 | 0.9650 | 0.9636 | 0.9670 | **0.9712** |
| | 30 | 0.9699 | **0.9774** | 0.9765 | 0.9730 | 0.9723 | 0.9736 | 0.9768 |
| | 40 | 0.9736 | **0.9801** | 0.9788 | 0.9763 | 0.9757 | 0.9765 | 0.9790 |
| Gantry $400 \times 264$ | 10 | 0.8367 | 0.8665 | 0.8626 | 0.8526 | 0.8460 | 0.8502 | **0.8665** |
| | 20 | 0.8947 | 0.9158 | 0.9161 | 0.9074 | 0.9035 | 0.9075 | **0.9178** |
| | 30 | 0.9159 | 0.9333 | 0.9334 | 0.9266 | 0.9221 | 0.9269 | **0.9335** |
| | 40 | 0.9244 | **0.9412** | 0.9409 | 0.9334 | 0.9308 | 0.9373 | 0.9399 |



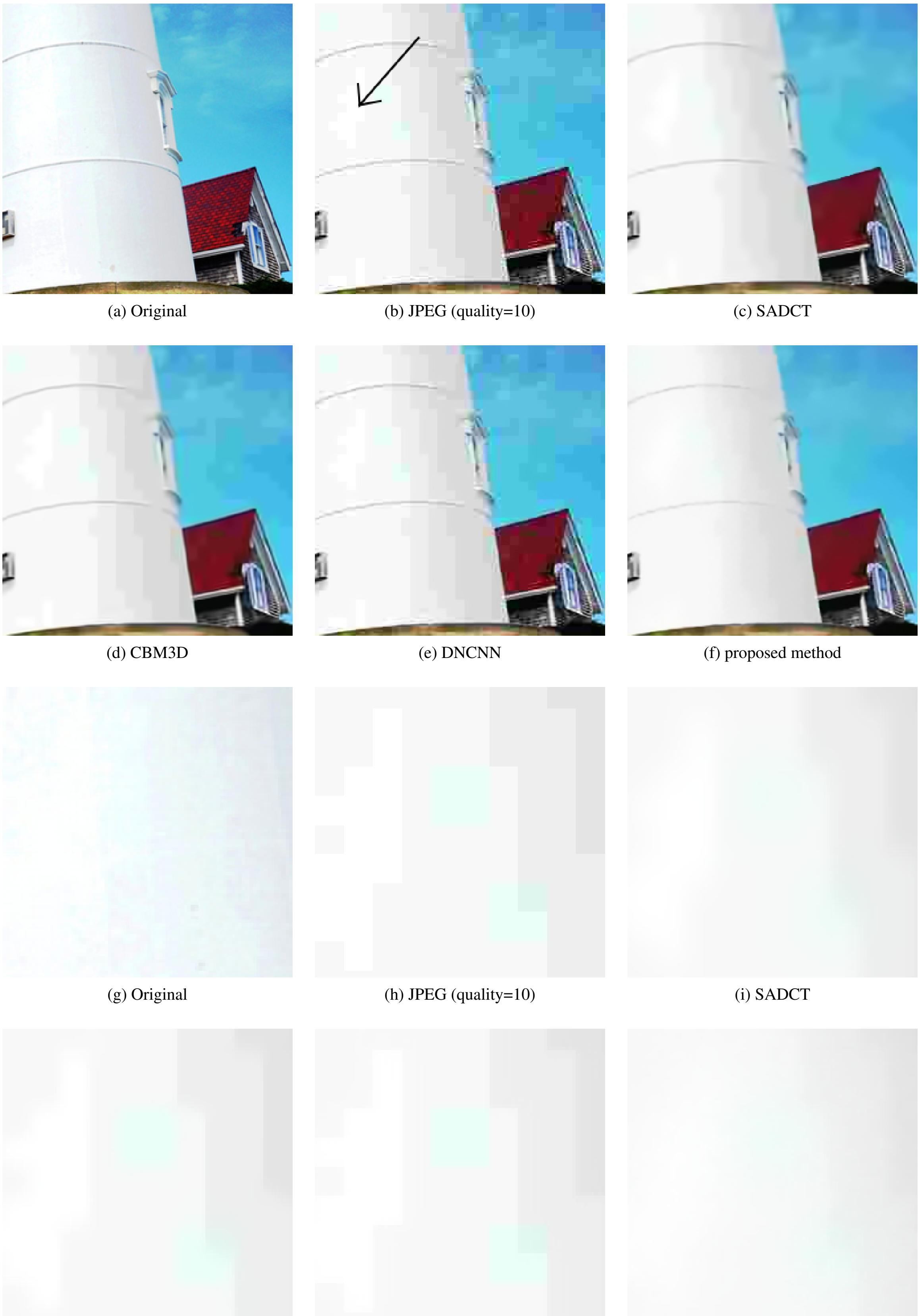
(a) Original (b) JPEG (quality=20) (c) SADCT

(d) CBM3D (e) ARCNN (f) proposed method

(g) Original (h) JPEG (quality=20) (i) SADCT

(j) CBM3D (k) ARCNN (l) proposed method

Figure 3: Restoration of the Toysflash image.



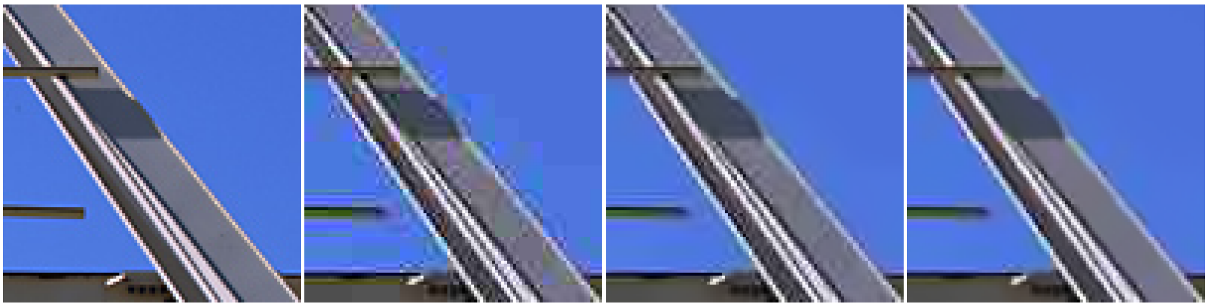
(a) Original (b) JPEG (quality=10) (c) SADCT

(d) CBM3D (e) DNCNN (f) proposed method

(g) Original (h) JPEG (quality=10) (i) SADCT

(j) CBM3D (k) DNCNN (l) proposed method

Figure 4: Restoration of the Lighthouse image.



(a) Original (b) JPEG (quality=20) (c) SADCT (d) proposed method

Figure 5: Restoration of the Gantry image for quality=20.

(a) JPEG (quality=30) (b) SADCT (c) proposed method

Figure 6: Restoration of the Gantry image for quality=30.

(a) JPEG (quality=40) (b) SADCT (c) proposed method

Figure 7: Restoration of the Gantry image for quality=40.

[Foi20a] Foi, A. Image and video denoising by sparse 3D transform-domain collaborative filtering. http://www.cs.tut.fi/~foi/GCF-BM3D/, accessed 2020.

[Foi20b] Foi, A. Shape-adaptive transforms filtering. https://www.cs.tut.fi/~foi/SA-DCT/, accessed 2020.

[Gan03] Gan, X., Wee-Chung Liew, A., and Yan, H. Blocking artifact reduction in compressed images based on edge-adaptive quadrangle meshes. Journal of Visual Communication and Image Representation, No. 14, pp. 492-507, 2003.

[Gay15] Gayathri Tejaswini, S., Ramalakshmi, M., Santhi, M., Rahul H., and Nag, H. Reduction of blocking artifacts of DCT compressed image based on block Wiener filtering. International Journal of Advanced Research in Electronics and Communication Engineering, No. 4(3), pp. 662-665, 2015.

[Gol14] Golestaneh, S. A., and Chandler, D. M. Algorithm for JPEG artifact reduction via local edge regeneration. Journal of Electronic Imaging, No. 23(1), 2014.

[Hsu98] Hsung, T, C., and Lun, D. P. K. Application of singularity detection for the deblocking of JPEG decoded images. IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, No. 45(5), pp. 640-644, 1998.

[Jpe20] JPEG image deblocking using deep learning. https://www.mathworks.com/help/images/jpeg-image-deblocking-using-deep-learning.html, accessed 2020.

[Kim20] Kim, Y., Soh, J. and Cho, N. AGARNet: Adaptively gated JPEG compression artifacts removal network for a wide range quality factor. IEEE Access, No. 8, pp. 20160–20170, 2020.

[Lee98] Lee, Y.L., Kim, H.C., and Park, H.W. Blocking effect reduction of JPEG images by signal adaptive filtering. IEEE Transactions on Image Processing, No. 7(2), pp. 229-234, 1998.

[Lee04] Lee, K., Kim, D. S., and Kim, T. Regression-based prediction for blocking artifact reduction in jpeg-compressed images. IEEE Transactions on Image Processing, No. 14(1), pp. 36-48, 2004.

[Li16] Li, S., Wang, G., Zhao, X. Multiplicative noise removal via adaptive learned dictionaries and TV regularization, Digital Signal Processing, No. 50, pp. 218-228, 2016.

[Lia02] Liaw, Y. C., Lo, W., and Lai, J. Z. C. Image restoration of compressed images using classified vector quantization. Pattern Recognition, No. 35, pp. 329-340, 2002.

[Lis03] List, P., Joch, A., Lainema, J., Bjontegaard, G., and Karczewicz, M. Adaptive deblocking filter. IEEE transactions on circuits and systems for video technology. No. 13(7), pp. 614-619, 2003.

[Ma02] Ma, J., Protter, P., San Martin, J., Torres, S., et al. Numerical method for backward stochastic differential equations. The Annals of Applied Probability, No. 12(1), pp. 302-316, 2002.

[Ozt07] Oztan, B., Malik, A., Fan, Z., and Eschbach, R. Removal of artifacts from JPEG compressed document images. Color Imaging XII: Processing, Hardcopy, and Applications No. 6493, 2007.

[Pan15] Pandey, S. S., Manu, Singh, P., and Pandey, V. Block wise image compression & reduced blocks artifacts using discrete cosine transform. International Journal of Scientific and Research Publications, No. 5(3), 2015.

[Pou14] Pourreza-Shahri, R., Yousefi, S. and Kehtarnavaz, N. Optimization method to reduce blocking artifacts in JPEG images. Journal of Electronic Imaging, No. 23(6), 2014.

[Ree84] Reeve, H. C. and Lim, J. S. Reduction of blocking effects in image coding. Optical Engineering, No. 23(1), 230134, 1984.

[Sin07] Singh, S., Kumar, V., and Verma, H. K. Reduction of blocking artifacts in JPEG compressed images. Digital Signal Processing, No. 17, pp. 225-243, 2007.

[Sin11] Singh, J., Singh, S., Singh, D., and Uddin, M. A signal adaptive filter for blocking effect reduction of JPEG compressed images. International Journal of Electronic and Communications, No. 65, pp. 827-839, 2011.

[Slo01] Słomiński, L. Euler's approximations of solutions of sdes with reflecting boundary. Stochastic Processes and their Applications, No. 94(2), pp. 317-337, 2001.

[Svo16] Svoboda, P., Hradiš, M., Bařina D., and Zemčík, P. Compression artifacts removal using convolutional neural networks. Journal of WSCG, No. 24(2), pp. 63–72, 2016.

[Tri03] Triantafyllidis, G. A., Varnuska, M., Sampson, D., Tzovaras, D., and Strintzis, M. G. An effiecient algorithm for the enhancement of JPEG-coded images. Computers & Graphics, No. 27, pp. 529-534, 2003.

[Wan04] Wang, Z., Simoncelli, E.P., and Bovik, A.C. Multiscale Structural similarity for image quality assessment. The Thirty-Seventh Asilomar Conference on Signals, Systems & Computers, pp. 1398–1402, 2003.

[Wan13] Wang, C., Zhou, J., and Liu, S. Adaptive non-local means filter for image deblocking. Signal Processing: Image Communication, No. 28(5), pp. 522–530, 2013.

[Wan20] Wang, M., Fu, X., Sun, Z., and Zha, Z. JPEG artifacts removal via compression quality ranker-guided networks. International Joint Conferences on Artificial Intelligence Organization, pp. 566–572, 2020.

[Wee02] Weerasinghe, C., Wee-Chung Liew, A., and Yan, H. Artifact reduction in compressed images based on region homogeneity constraints using the projection onto convex sets algorithm. IEEE Transactions on Circuits and Systems for Video Technology, No. 12(10), pp. 891-897, 2002.

[Yuk20] Yu, K., Dong, C., Deng, Y., Loy, C. C. and Tang, X. Deep convolution networks for compression artifacts reduction. http://mmlab.ie.cuhk.edu.hk/projects/ARCNN.html, accessed 2020.

[Zha17] Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing, No. 26(7), pp. 3142-3155, 2017.

[Zha18] Zhang, X., Yang, W., Hu, Y., and Liu, J. Dmcnn: Dual-domain multi-scale convolutional neural network for compression artifacts removal. 25th IEEE International Conference on Image Processing, pp. 390-394, 2018.