Published in Image Processing On Line on 2023–12–09.
Submitted on 2023–02–22, accepted on 2023–09–15.
ISSN 2105–1232 © 2023 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2023.467

Implementation of Image Denoising based on Backward Stochastic Differential Equations

Dariusz Borkowski

Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, Poland
dbor@mat.umk.pl

Abstract

In this paper, we give the implementation of an image denoising algorithm based on backward stochastic differential equations. In our algorithm, we consider two stochastic 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 weights to values of pixels. The reconstructed image is characterized by smoothing noisy pixels and at the same time enhancing edges. Our experiments show that the new approach gives very good results and can be successfully used to reconstruct images.

Source Code

The reviewed source code and documentation for this algorithm are available at the IPOL web page of this article[1]. Compilation and usage instructions are included in the `README.txt` file of the archive.

Keywords: denoising; stochastic process; inverse problem

1 Introduction

Let D be a bounded, convex domain in $\mathbf{R}^2$, $u : \overline{D} \to \mathbf{R}$ be an original image and $u_0 : \overline{D} \to \mathbf{R}$ be the observed image of the form

$$u_0 = u + \eta,$$

where η stands for a white Gaussian noise. We assume that $u_0 \in \mathbf{C}^1(\overline{D})$. We are given u_0, the problem is to reconstruct u. This is a typical example of an inverse problem [3].

The problem of image reconstruction using fully automatic and reliable methods is one of the most important issues of digital image processing and computer vision. Efficient and effective reconstruction of images is an essential element of most image processing and recognizing algorithms. Reconstruction algorithms allow us to make initial treatment of data for further analysis, which is very important, especially in astronomy, biology or medicine.

[1] https://doi.org/10.5201/ipol.2023.467

Various techniques have been proposed to tackle this inverse problem. One may quote the linear filtering, DCT [57, 58], wavelets theory [23, 27], variational methods [23, 31, 51] and stochastic modelling which are generally based on the Markov field theory and Bayesian approach [23, 30, 50]. In another class, one could include methods that take advantage of the non-local similarity of patches in the image. Among the most famous, we can name NL-means [18, 19], BM3D [25, 26, 35], NL-Bayes [37] and K-SVD [2, 40]. A different approach is based on neural networks [24, 61]. The concept of deep neural networks has gained wide recognition and its later versions are successfully used to image restoration [33, 41, 49]. The most important direction in image processing has been methods driven by nonlinear diffusion equations [21, 46, 51, 55, 56], where a special case of this approach is restoration based on BSDEs.

The backward stochastic differential equations (in short BSDEs) were introduced by Pardoux and Peng [43], who proved the existence and uniqueness of adapted solutions under suitable assumptions. Independently, Duffie and Epstein [28, 29] introduced stochastic differential utilities in economics models, as solution of certain BSDEs. Since then, it has been widely recognized that BSDEs provide a useful framework for formulating many problems in mathematical finance [34]. They have also appeared to be useful for problems in stochastic control and differential games [32]. Many papers (for instance [44]) show the connections between BSDEs driven by a diffusion process and solutions of a large class of quasilinear parabolic and elliptic partial differential equations (PDEs). These results may be seen as a generalization of the celebrated Feynman-Kac formula. Through all these results, a formal dictionary of the relations between BSDEs and PDEs can be established, which suggests that existence and uniqueness results which can be obtained on the one side should have their counterparts on the other side. However, in our opinion, a stochastic description is more intuitive. Therefore, we treat BSDEs as a starting point in the creation of the reconstruction model. Moreover, using stochastic analysis yields a completely new methodology, what is more important in research.

In image processing one can find theoretical results [1] and some practical aspects [4, 6, 10, 11, 13, 16] of BSDE-based applications. In [4] the problem of reconstruction of the noisy chromaticity is considered. The model of denoising presented there is expressed in terms of a Skorokhod problem associated with the solution of BSDE and an epsilon neighborhood of a two-dimensional sphere. In that paper, BSDE is driven by a trivial drift function ($f \equiv 0$). This means that the presented equation is a model of forward filtering and has properties of smoothing filters. In [6] problems of reconstruction of the noisy grayscale image (smoothing filters) and enhancing of the blurred grayscale image (enhancing filters) are presented. A smoothing filter, similarly as in [4], models an anisotropic forward diffusion with BSDEs with $f \equiv 0$. Enhancing filters presented in [6] are driven by BSDEs with a non-trivial drift function and correspond to an inverse heat equation. This equation is a model of backward filtering (not forward) and in consequence this model fails to perform edge enhancing of the noisy image. The paper [13] is a generalization of [6] to images with values in $\mathbf{R}^n$, in particular to color images. The model of forward anisotropic filtering in a direction perpendicular to the gradient and inverse anisotropic filtering in the gradient direction in terms of BSDEs was presented in [10] and [16] for grayscale and color images respectively. An interesting work using BSDE with reflection to reconstruct images in spaces other than $\mathbf{R}^n$ was presented in [11].

In this work, apart from the implementation of the BSDE algorithm, we generalize the theoretical results from [10]. This generalization will allow us to write a simple and concise reconstruction algorithm. The paper is organized as follows. Section 2 contains definitions and fundamental facts of stochastic analysis. In Section 3 we briefly recall basic ideas of filtering in terms of BSDEs. Section 4 provides new theoretical results for BSDE denoising. We define here the numerical scheme and prove a convergence of this scheme to the continuous model. This chapter is also devoted to presenting details of numerical approximation of the proposed method. Then in Section 5 we introduce algorithms for grayscale images and generalize them in Section 6 to color images. In Section 7, we provide a justification for the selection of parameter values. Finally, in Section 8

experimental results and comparisons to other methods are presented.

2 Mathematical Preliminaries

In what follows, by $(\Omega, \mathcal{F}, \mathcal{P})$ we will denote a probability space and by $W = \{W_t; t \in [0,T]\}$ a two-dimensional Wiener process starting from zero. We assume that we are given a point $x_0 \in \overline{D}$ and a function $\sigma : [0,T] \times \mathbf{R}^2 \to \mathbf{R}^2 \times \mathbf{R}^2$. Consider the stochastic diffusion process with reflection with values in domain $\overline{D}$

$$X_t = x_0 + \int_0^t \sigma(s, X_s)\, dW_s + K_t^{\overline{D}}, \quad t \in [0,T]. \tag{1}$$

Equation (1) (called reflected SDE) characterizes the behaviour of the continuous time stochastic process X as an Itö integral. A heuristic interpretation is that in a small time interval the stochastic process X changes its value by an amount that is normally distributed with variance $\sigma(t, X_t)$ and is independent of the past behavior of the process. This is so because the increments of a Wiener process are independent and normally distributed. The function σ is called the diffusion coefficient. The term $K_t^{\overline{D}}$ is the minimal push needed to keep the process X in $\overline{D}$. For each fixed $\omega \in \Omega$ the function $t \to X_t(\omega)$ is called a trajectory of X and is denoted by $X(\omega)$. The proof of the existence and uniqueness of the solution to reflected SDEs can be found in [53].

The process X can be approximated by the following numerical scheme [48, 52]

$$\begin{aligned} X_0^m &= X_0, \\ X_{t_k}^m &= \Pi_{\overline{D}}[X_{t_{k-1}}^m + \sigma(t_{k-1}, X_{t_{k-1}}^m)(W_{t_k} - W_{t_{k-1}})],\ k = 0, 1, \ldots, m, \end{aligned}$$

where $t_k = \frac{kT}{m}$, $m \in \mathbf{N}$ and $\Pi_{\overline{D}}(x)$ denotes a projection of x on the set $\overline{D}$. Since D is convex, the projection is unique.

Let $(\mathcal{F}_t)$ be a filtration generated by an l-dimensional Wiener process W, $\xi \in \mathbf{L}^2(\Omega, \mathcal{F}_T, P, \mathbf{R}^k)$ be a square integrable random variable and let $f : \Omega \times [0,T] \times \mathbf{R}^k \to \mathbf{R}^k$ be a Lipschitz continuous function in the space variable.

Definition 1. *A solution to the backward stochastic differential equation (BSDE) associated with ξ and f is a pair of $(\mathcal{F}_t)$-measurable processes (Y, Z) with values in $\mathbf{R}^k \times \mathbf{R}^{k \times l}$ satisfying*

$$\mathbf{E}\left[\int_0^T \|Z_s\|^2\, ds\right] < \infty,$$

and

$$Y_t = \xi + \int_t^T f(s, Y_s) ds - \int_t^T Z_s\, dW_s, \quad t \in [0,T]. \tag{2}$$

See [42] for the proof of the existence and uniqueness of the solution to BSDEs. The backward stochastic differential equation is solved starting from $t = T$ until $t = 0$. Note that at time zero the Y_0 is a deterministic value. A drift function f causes the correction of trajectories in expected strength and direction. Later on, we are interested in the process Y, i.e. the first component of the solution to the BSDE. The fact that every martingale on the space with Wiener filtration has a representation as a stochastic integral implies that

$$Y_t = \xi + \int_t^T f(s, Y_s) ds - \int_t^T Z_s\, dW_s = \mathbf{E}\left[\xi + \int_t^T f(s, Y_s) ds | \mathcal{F}_t\right],$$

therefore the process Y can be approximated using the formula [39]

$$\begin{aligned} Y^m_{t_m} &= \xi^m, \\ \hat{Y}^m_{t_k} &= \mathbf{E}[Y^m_{t_{k+1}} | \mathcal{F}^m_{t_k}], \\ Y^m_{t_k} &= \hat{Y}^m_{t_k} + \frac{T}{m} f(t_k, \hat{Y}^m_{t_k}),\ k = m-1, m-2, \ldots, 0, \end{aligned} \tag{3}$$

where $t_k = \frac{kT}{m}$. Here by $\mathbf{E}$ we denote the expected value and by $\mathcal{F}^m$ the filtration generated by the discretization of a Wiener process.

Theorem. *If $\xi^m \to \xi$ $\mathcal{P}-a.s.$ then*

$$\sup_{0 \le k \le n} |Y^m_{t_k} - Y_{t_k}| \to 0,$$

in probability.

3 Continuous Model

A general model of the image reconstruction is the following

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

$$\sigma(s, X_s) = \begin{bmatrix} -\left(1 - \dfrac{c(s)}{c}\right) \dfrac{(G_\gamma * u_0)_{x_2}(X_s)}{|\nabla (G_\gamma * u_0)(X_s)|}, \dfrac{c(s)}{c} \dfrac{(G_\gamma * u_0)_{x_1}(X_s)}{|\nabla (G_\gamma * u_0)(X_s)|} \\ \left(1 - \dfrac{c(s)}{c}\right) \dfrac{(G_\gamma * u_0)_{x_1}(X_s)}{|\nabla (G_\gamma * u_0)(X_s)|}, \dfrac{c(s)}{c} \dfrac{(G_\gamma * u_0)_{x_2}(X_s)}{|\nabla (G_\gamma * u_0)(X_s)|} \end{bmatrix},$$

$$c(t) = \begin{cases} 0 & \text{if} \quad t < S, \\ c & \text{if} \quad t \ge S, \end{cases}$$

where $\{X_t\}_{t \in [0,T]}$ is a stochastic diffusion process, $\{W_t\}_{t \in [0,T]}$ is a two-dimensional Wiener process, the term $\{K_t^{\overline{D}}\}_{t \in [0,T]}$ is the minimal push needed to keep the 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 which determines the measurability of the process Y. G_γ is a normalized 3×3 Gaussian kernel

$$G_\gamma = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix},$$

and $S \in \mathbf{R}_+ < T \in \mathbf{R}_+$, $c \in \mathbf{R}_+$ are parameters of the method. Parameter T defines the size of the neighborhood used in the reconstruction procedure. We deblur from time T to S and smooth out from S to 0. The parameter c is responsible for effect of edge sharpening. The values of all parameters depend on a standard noise deviation ρ. Note that the process X has values in the domain of the image $\overline{D}$ and is driven by some function σ. The process Y has values in the codomain of the image and is driven by some function f.

For a fixed pixel x we consider a certain BSDE equation. The values of the process X determine the pixels from the domain of the image $\overline{D}$ which we will use in the reconstruction procedure. We

can say that this process determines the neighborhood of the pixel x (with irregular shape). The reconstructed value $u(x)$ is the sum of pixels from its neighborhood multiplied by some weights. The weight values are determined by the process Y (driven by the function f). Appropriate definitions of the function $c(t)$ and f allow us to give weight values (also negative) which depend on direction and distance from the reconstructed pixel. The value of the process Y at time $t = 0$ is the reconstructed pixel $u(x)$.

4 Approximation

Below we formulate the theorem on which the reconstruction algorithm is based. The theorem is a generalization of results from the work [10]. We consider an additional component – the function g. The task of the function g will be to determine the weights based on the similarity of patches like in the NLM algorithm [18, 19]. The concept of using the similarity of patches with stochastic diffusion was presented in [9, 12, 14, 15], but only at the numerical level. In this paper, we have taken that idea to a BSDE continuous model (in the form of the function g).

Theorem. *Let $S < T$, $u_0 : \overline{D} \to \mathbf{R}$, $x \in \overline{D}$. Assume that $\xi = u_0(X_S^x)$, where X^x is a two-dimensional diffusion process with reflection with values in $\overline{D}$ and starting from x and*

$$f(t,y) = \begin{cases} c(t)(y - u_0(X_t^x)), & t \geq S, \\ b(t)\left(g(X_t^x, u_0, x)u_0(X_t^x) - y\right), & t < S. \end{cases}$$

If (Y, Z) is a solution to the BSDE

$$Y_t = \xi + \int_t^T f(s, Y_s)ds - \int_t^T Z_s\, dW_s, \quad t \in [0,T],$$

then

$$\lim_{m \to +\infty} Y_0^m = Y_0,$$

where

$$Y_0^m = \sum_{k=0}^{j-1} a_k \mathbf{E}\left[g(X_{t_k}^x, u_0, x)u_0(X_{t_k}^x)\right] + \sum_{k=j}^{m-1} a_k \mathbf{E}\left[u_0(X_{t_k}^x)\right], \tag{4}$$

$$0 = t_0 < t_1 < \cdots < t_j \leq S < t_{j+1} < \cdots < t_m = T, \quad dt = t_{i+1} - t_i = \frac{T}{m},$$

$$a_k = \frac{b(t_k)T}{m} \prod_{s=0}^{k-1} \left(1 - \frac{b(t_s)T}{m}\right), \ k = 0, 1, \ldots, j-1, \tag{5}$$

$$a_j = \prod_{s=0}^{j-1} \left(1 - \frac{b(t_s)T}{m}\right) \left[\prod_{r=j}^{m-1} \left(1 + \frac{c(t_r)T}{m}\right) - \frac{c(t_j)T}{m}\right], \tag{6}$$

$$a_k = -\prod_{s=0}^{j-1} \left(1 - \frac{b(t_s)T}{m}\right) \frac{c(t_k)T}{m} \prod_{r=j}^{k-1} \left(1 + \frac{c(t_r)T}{m}\right), \ k = j+1, j+2, \ldots, m-1. \tag{7}$$

Proof. Note that

$$
\begin{aligned}
&Y^m_{t_m} = u_0(X^x_{t_j}),\\
&Y^m_{t_{m-1}} = \left(1+\frac{c(t_{m-1})T}{m}\right)\mathbf{E}\left[u_0(X^x_{t_j})|\mathcal{F}^m_{t_{m-1}}\right] - \frac{c(t_{m-1})T}{m}u_0(X^x_{t_{m-1}}),\\
&Y^m_{t_{m-2}} = \left(1+\frac{c(t_{m-2})T}{m}\right)\left(1+\frac{c(t_{m-1})T}{m}\right)\mathbf{E}\left[u_0(X^x_{t_j})|\mathcal{F}^m_{t_{m-2}}\right]\\
&\qquad - \left(1+\frac{c(t_{m-2})T}{m}\right)\frac{c(t_{m-1})T}{m}\mathbf{E}\left[u_0(X_{t_{m-1}})|\mathcal{F}^m_{t_{m-2}}\right] - \frac{c(t_{m-2})T}{m}u_0(X^x_{t_{m-2}}),\\
&\vdots\\
&Y^m_{t_j} = \prod_{r=j}^{m-1}\left(1+\frac{c(t_r)T}{m}\right)\mathbf{E}\left[u_0(X^x_{t_j})|\mathcal{F}^m_{t_j}\right] - \sum_{k=j+1}^{m-1}\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\mathbf{E}\left[u_0(X^x_{t_k})|\mathcal{F}^m_{t_j}\right] - \frac{c(t_j)T}{m}u_0(X^x_{t_j})\\
&\qquad = \left[\prod_{r=j}^{m-1}\left(1+\frac{c(t_r)T}{m}\right) - \frac{c(t_j)T}{m}\right]u_0(X^x_{t_j}) - \sum_{k=j+1}^{m-1}\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\mathbf{E}\left[u_0(X^x_{t_k})|\mathcal{F}^m_{t_j}\right],\\
&Y^m_{t_{j-1}} = \left(1-\frac{b(t_{j-1})T}{m}\right)\mathbf{E}\left[Y^m_{t_j}|\mathcal{F}^m_{t_{j-1}}\right] + \frac{b(t_{j-1})T}{m}g(X^x_{t_{j-1}},u_0,x)u_0(X^x_{t_{j-1}}),\\
&Y^m_{t_{j-2}} = \left(1-\frac{b(t_{j-2})T}{m}\right)\left(1-\frac{b(t_{j-1})T}{m}\right)\mathbf{E}\left[Y^m_{t_j}|\mathcal{F}^m_{t_{j-2}}\right]\\
&\qquad + \left(1-\frac{b(t_{j-2})T}{m}\right)\frac{b(t_{j-1})T}{m}\mathbf{E}\left[g(X^x_{t_{j-1}},u_0,x)u_0(X^x_{t_{j-1}})|\mathcal{F}^m_{t_{j-2}}\right] + \frac{b(t_{j-2})T}{m}g(X^x_{t_{j-2}},u_0,x)u_0(X^x_{t_{j-2}}),\\
&\vdots\\
&Y^m_{t_0} = \prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\mathbf{E}\left[Y^m_{t_j}|\mathcal{F}^m_{t_0}\right] + \sum_{k=0}^{j-1}\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right)\mathbf{E}\left[g(X^x_{t_k},u_0,x)u_0(X^x_{t_k})|\mathcal{F}^m_{t_0}\right]\\
&\qquad = \prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{m-1}\left(1+\frac{c(t_r)T}{m}\right) - \frac{c(t_j)T}{m}\right]\mathbf{E}\left[u_0(X^x_{t_j})\right]\\
&\qquad\quad - \sum_{k=j+1}^{m-1}\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\mathbf{E}\left[u_0(X^x_{t_k})\right]\\
&\qquad\quad + \sum_{k=0}^{j-1}\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right)\mathbf{E}\left[g(X^x_{t_k},u_0,x)u_0(X^x_{t_k})\right]\\
&\qquad = \sum_{k=0}^{j-1}\left(\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right)\right)\mathbf{E}\left[g(X^x_{t_k},u_0,x)u_0(X^x_{t_k})\right]\\
&\qquad\quad + \left(\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{m-1}\left(1+\frac{c(t_r)T}{m}\right) - \frac{c(t_j)T}{m}\right]\right)\mathbf{E}\left[u_0(X^x_{t_j})\right]\\
&\qquad\quad + \sum_{k=j+1}^{m-1}\left(-\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\right)\mathbf{E}\left[u_0(X^x_{t_k})\right].
\end{aligned}
$$

□

Remark. *Coefficients a_k satisfy the condition*

$$\sum_{k=0}^{m-1} a_k = 1.$$

Proof.

$$\begin{aligned}
\sum_{k=0}^{m-1} a_k &= \prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{m-1}\left(1+\frac{c(t_r)T}{m}\right)-\frac{c(t_j)T}{m}-\sum_{k=j+1}^{m-1}\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\right] \\
&+\sum_{k=0}^{j-1}\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right)=\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)+\sum_{k=0}^{j-1}\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right) \\
&=\prod_{s=0}^{j-2}\left(1-\frac{b(t_s)T}{m}\right)\left(1-\frac{b(t_{j-1})T}{m}\right)+\sum_{k=0}^{j-2}\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right) \\
&+\frac{b(t_{j-1})T}{m}\prod_{s=0}^{j-2}\left(1-\frac{b(t_s)T}{m}\right)=\prod_{s=0}^{j-2}\left(1-\frac{b(t_s)T}{m}\right)+\sum_{k=0}^{j-2}\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right) \\
&=\prod_{s=0}^{1}\left(1-\frac{b(t_s)T}{m}\right)+\sum_{k=0}^{1}\frac{b(t_k)T}{m}\prod_{s=0}^{k-1}\left(1-\frac{b(t_s)T}{m}\right)=\left(1-\frac{b(t_0)T}{m}\right)\left(1-\frac{b(t_1)T}{m}\right) \\
&+\frac{b(t_0)T}{m}+\frac{b(t_1)T}{m}\left(1-\frac{b(t_0)T}{m}\right)=1-\frac{b(t_0)T}{m}-\frac{b(t_1)T}{m}+\frac{b(t_0)T}{m}\frac{b(t_1)T}{m} \\
&+\frac{b(t_0)T}{m}+\frac{b(t_1)T}{m}-\frac{b(t_1)T}{m}\frac{b(t_0)T}{m}=1.
\end{aligned}$$

□

The property below will be used directly in the reconstruction algorithm.

Remark. *Let $(a_k)_{k=0,1,\dots m-1}$ be determined by functions $b(t)$, $c(t)$ and let $(a'_k)_{k=0,1,\dots m-1}$ be determined by $b(t)$, $c'(t)\equiv 0$. Then*

$$a'_k=\begin{cases} a_k & k<j, \\ \sum\limits_{s=j}^{m-1} a_s & k=j, \\ 0 & k>j. \end{cases} \tag{8}$$

Proof.

$$\sum_{k=j}^{m-1} a_k = \prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{m-1}\left(1+\frac{c(t_r)T}{m}\right)-\frac{c(t_j)T}{m}\right]$$
$$-\sum_{k=j+1}^{m-1}\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)=$$
$$\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{m-1}\left(1+\frac{c(t_r)T}{m}\right)-\frac{c(t_j)T}{m}-\sum_{k=j+1}^{m-1}\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\right]$$
$$=\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{m-2}\left(1+\frac{c(t_r)T}{m}\right)\left(1+\frac{c(t_{m-1})T}{m}\right)\right.$$
$$\left.-\frac{c(t_j)T}{m}-\sum_{k=j+1}^{m-2}\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)-\frac{c(t_{m-1})T}{m}\prod_{r=j}^{m-2}\left(1+\frac{c(t_r)T}{m}\right)\right]$$
$$=\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{m-2}\left(1+\frac{c(t_r)T}{m}\right)-\frac{c(t_j)T}{m}-\sum_{k=j+1}^{m-2}\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\right]$$
$$\vdots$$
$$=\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\prod_{r=j}^{j+1}\left(1+\frac{c(t_r)T}{m}\right)-\frac{c(t_j)T}{m}-\sum_{k=j+1}^{j+1}\frac{c(t_k)T}{m}\prod_{r=j}^{k-1}\left(1+\frac{c(t_r)T}{m}\right)\right]$$
$$=\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)\left[\left(1+\frac{c(t_j)T}{m}\right)\left(1+\frac{c(t_{j+1})T}{m}\right)-\frac{c(t_j)T}{m}-\frac{c(t_{j+1})T}{m}\left(1+\frac{c(t_j)T}{m}\right)\right]$$
$$=\prod_{s=0}^{j-1}\left(1-\frac{b(t_s)T}{m}\right)=a'_j.$$

□

Using the discretization (4) the reconstructed pixel can be approximated by the following formula

$$u(x)\approx Y_0^m=\sum_{k=0}^{j-1}a_k\mathbf{E}\left[g(X^x_{t_k},u_0,x)u_0(X^x_{t_k})\right]+\sum_{k=j}^{m-1}a_k\mathbf{E}\left[u_0(X^x_{t_k})\right]=$$
$$\mathbf{E}\left[\sum_{k=0}^{j-1}a_k g(X^x_{t_k},u_0,x)u_0(X^x_{t_k})+\sum_{k=j}^{m-1}a_k u_0(X^x_{t_k})\right],$$

$$u(x)\approx\frac{1}{N}\sum_{n=1}^{N}\left[\sum_{k=0}^{j-1}a_k g(X^x_{t_k}(\omega_n),u_0,x)u_0(X^x_{t_k}(\omega_n))+\sum_{k=j}^{m-1}a_k u_0(X^x_{t_k}(\omega_n))\right], \tag{9}$$

where $X^x(\omega)$ means the trajectory of X^x (see Figure 1) starting from x such that

$$\begin{aligned}&X^x_0(\omega)=x,\\&X^x_{t_k}(\omega)=\Pi_{\overline{D}}[X^x_{t_{k-1}}(\omega)+\sigma(t_{k-1},X^x_{t_{k-1}}(\omega))(W_{t_k}(\omega)-W_{t_{k-1}}(\omega))],\\&0=t_0<t_1<\cdots<t_j\le S<t_{j+1}<\cdots<t_m=T,\ t_{i+1}-t_i=\frac{T}{m},\end{aligned} \tag{10}$$

and the parameter N is equal to the number of Monte Carlo iterations.

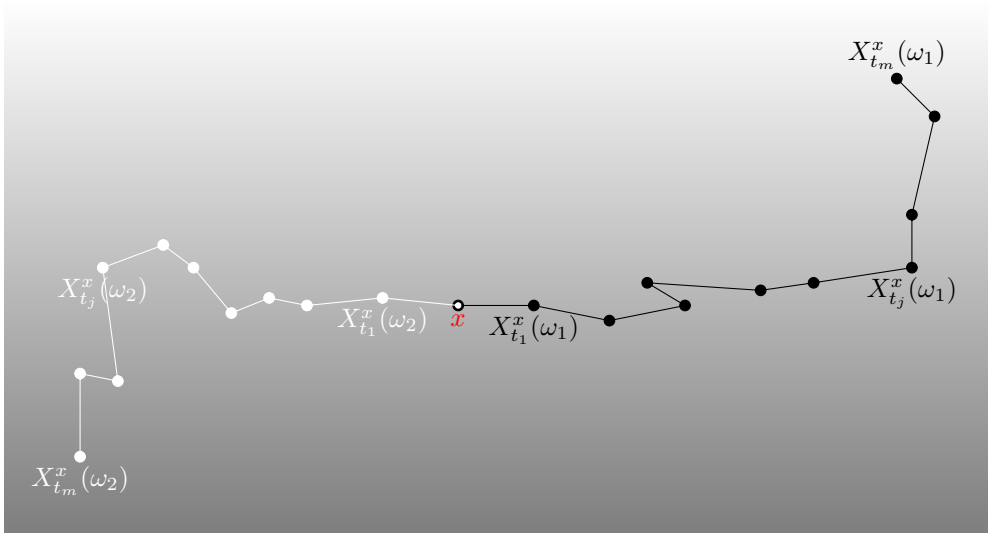


Figure 1: Example of two trajectories of the process X^x. From time t_0 to t_j trajectories have values along edges. From time t_j to t_{m-1} the process diffuses in the gradient direction.

Example 1. $b(t) \equiv 0$
Using the Markov property for diffusion process X^x the expected value can be written as

$$\mathbf{E}\left[u_0(X^x_{t_k})\right] \approx \frac{1}{N}\sum_{n=1}^{N} u_0(X^x_{t_k}(\omega_n)) = \frac{1}{N_0}\sum_{n_0=1}^{N_0}\left(\frac{1}{N_k}\sum_{n_k=1}^{N_k} u_0\left(X_{t_k}^{X^x_{t_j}(\omega_{n_0})}(\omega_{n_k})\right)\right),$$

where N_0, N_k are numbers of Monte Carlo iterations and should be chosen so that $\frac{N_k}{N_0} \approx \frac{t_k - t_j}{t_j}$ for fixed t_j.

$$\begin{aligned} u(x) \approx Y_0^m &\approx \frac{1}{N_0}\sum_{n_0=1}^{N_0}\left(\sum_{k=j}^{m-1}\sum_{n_k=1}^{N_k}\frac{a_k}{N_k}u_0\left(X_{t_k}^{X^x_{t_j}(\omega_{n_0})}(\omega_{n_k})\right)\right) \\ &= \frac{1}{N_0}\sum_{n_0=1}^{N_0}\left(a_j u_0\left(X^x_{t_j}(\omega_{n_0})\right) + \sum_{k=j+1}^{m-1}\sum_{n_k=1}^{N_k}\frac{a_k}{N_k}u_0\left(X_{t_k}^{X^x_{t_j}(\omega_{n_0})}(\omega_{n_k})\right)\right) \\ &= \underbrace{\frac{1}{N_0}\sum_{n_0=1}^{N_0}\underbrace{\left(a_j u_0\left(X^x_{t_j}(\omega_{n_0})\right) + \frac{1}{N_1}\sum_{n_1=1}^{N_1}\sum_{k=j+1}^{m-1} a_k u_0\left(X_{t_k}^{X^x_{t_j}(\omega_{n_0})}(\omega_{n_1})\right)\right)}_{\text{Deblurring in gradient direction at point } X^x_{t_j}(\omega_{n_0})}}_{\text{Smoothing out of deblurring effects in perpendicular to gradient direction}}, \end{aligned}$$

where $\frac{N_1}{N_0} \approx \frac{t_{m-1} - t_j}{t_j}$.

Example 2. $g(X_t^x, u_0, x) \equiv 1$

$$u(x) \approx Y_0^m = \sum_{k=j}^{m-1} a_k \mathbf{E}\left[u_0(X_{t_k}^x)\right] + \mathbf{E}\left[\sum_{k=0}^{j-1} a_k u_0(X_{t_k}^x)\right]$$

$$= \underbrace{\frac{1}{N_0}\sum_{n_0=1}^{N_0}\underbrace{\left(a_j u_0\left(X_{t_j}^x(\omega_{n_0})\right) + \frac{1}{N_1}\sum_{n_1=1}^{N_1}\sum_{k=j+1}^{m-1} a_k u_0\left(X_{t_k}^{X_{t_j}^x(\omega_{n_0})}(\omega_{n_1})\right)\right)}_{\text{Deblurring in gradient direction at point } X_{t_j}^x(\omega_{n_0})}}_{\text{Smoothing out of deblurring effects in perpendicular to gradient direction}}$$

$$+ \underbrace{\frac{1}{N_2}\sum_{n_2=1}^{N_2}\sum_{k=0}^{j-1} a_k u_0(X_{t_k}^x(\omega_{n_2}))}_{\text{Additional smoothing in perpendicular to gradient direction with weights greater for points closer to x, i.e. } a_0>a_1>\cdots>a_{j-1}}$$

Example 3. $c(t) \equiv 0$, $g(X_t^x, u_0, x)$ - *some measure of similarity between neighborhoods of* $u_0(x)$ *and* $u_0(X_t^x)$.

$$u(x) \approx Y_0^m = a_j \mathbf{E}\left[u_0(X_{t_j}^x)\right] + \mathbf{E}\left[\sum_{k=0}^{j-1} a_k g(X_{t_k}^x, u_0, x) u_0(X_{t_k}^x)\right]$$

$$= \underbrace{\frac{1}{N_0}\sum_{n_0=1}^{N_0}\left(a_j u_0\left(X_{t_j}^x(\omega_{n_0})\right)\right)}_{\text{Smoothing in perpendicular to gradient direction}}$$

$$+ \underbrace{\frac{1}{N_2}\sum_{n_2=1}^{N_2}\sum_{k=0}^{j-1} a_k g(X_{t_k}^x(\omega_{n_2}), u_0, x) u_0(X_{t_k}^x(\omega_{n_2}))}_{\text{Additional reconstruction based on a comparison of similarity of pixel neighborhoods and distance from x}}$$

Modified Diffusion with Random Terminal Time

In this section, we focus on the simulation of the trajectory $X^x(\omega)$.

In formula (10) the expression $W_{t_k}(\omega) - W_{t_{k-1}}(\omega)$ is approximated using a random number generator and is equal to two independent values obtained using a generator of the normal distribution with parameters $\mathcal{N}(0, t_k - t_{k-1})$. After considering this observation, we have a simple method of simulation of values

$$X_{t_0}^x(\omega), X_{t_1}^x(\omega), X_{t_2}^x(\omega), \ldots, X_{t_j}^x(\omega), \ldots, X_{t_{m-1}}^x(\omega), X_{t_m}^x(\omega).$$

The numerical scheme (10) gives good results, but only with a small value of the time-step parameter $dt = \frac{T}{m}$ (for example $dt = 0.1$) – especially in the case of diffusion along edges. Calculating the mean value using the Monte Carlo method for small dt is not effective and takes a long time. To avoid this problem, we improve this scheme. The proposed modification is based on the following observation: the cardinality of the deblurring times set $\{t_{j+1}, t_{j+2}, \ldots, t_{m-1}\}$ is several times less

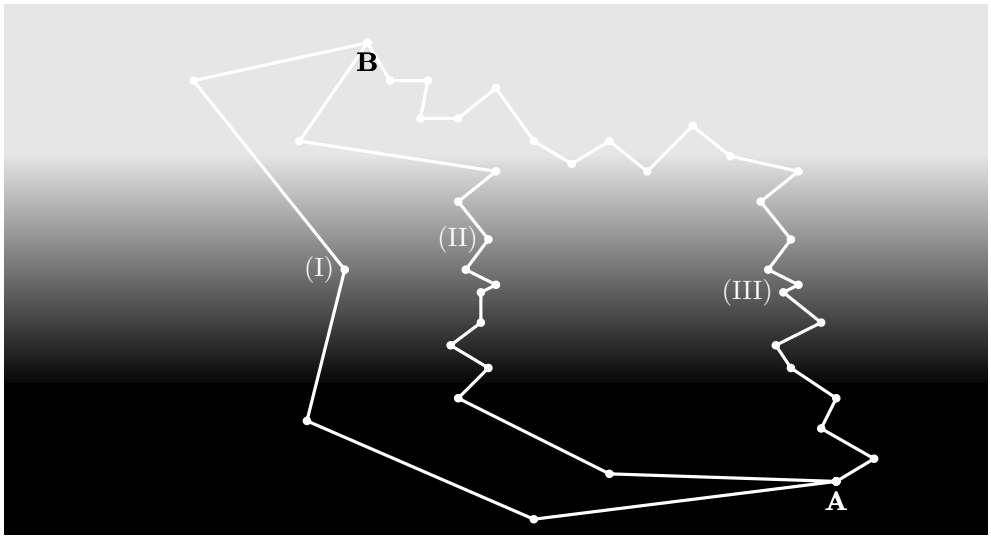


Figure 2: Example of trajectories of the process $(G_\gamma * u_0)(X_t^x)$ from pixel A to B: (I) – using the scheme (10) and large dt, (II) – using the scheme (11) and large dt, (III) – using the scheme (10) and small dt

than the cardinality of the smoothing times set $\{t_0, t_1, \ldots, t_j\}$. In order to reduce the size of the set $\{t_0, t_1, \ldots, t_j\}$ we use the following modification of Euler's approximation taken from [5, 7, 8]

$$X_0^x(\omega) = x, \quad H_{t_k}^x = \Pi_{\overline{D}}[X_{t_{k-1}}^x(\omega) + (W_{t_k} - W_{t_{k-1}})],$$

$$X_{t_k}^x(\omega) = \begin{cases} H_{t_k}^x(\omega) & \text{if} \quad \Theta, \\ & k = 1, 2, \ldots, \tau_j, \\ X_{t_{k-1}}^x(\omega) & \text{elsewhere,} \end{cases} \tag{11}$$

where by Θ we mean the condition

$$|(G_\gamma * u_0)(H_{t_k}^x(\omega)) - (G_\gamma * u_0)(X_{t_{k-1}}^x(\omega))| < p,$$

and $\tau_j = \min\{k; k \geq j \text{ and } \Theta \text{ is true } j \text{ times}\}$.

The condition Θ with parameter $p > 0$ guarantees that, if the image exhibits a strong gradient then the process X^x diffuses as a process with a small value of the parameter dt (we need to be careful not to lose information in the image) and at locations where variations of the brightness are weak – for background of the image, the process X^x can diffuse with a large value of dt (for example $dt = 4$). A terminal time τ_j provides that the numerical simulation of the diffusion trajectory gives at least j values of $X^x(\omega)$ which differ from the value in the previous step. Figure 2 illustrates a difference between the scheme (10) and the scheme (11). There are shown three examples of trajectories $(G_\gamma * u_0)(X_t^x)$ from the pixel A to the pixel B. Trajectories (I) and (III) were generated using the scheme (10) for large and small value of the parameter dt, respectively. Trajectory (II) was generated using the scheme (11) for large dt. It is easy to see, that at locations where the image is constant, trajectory (II) diffuses as trajectory (I). At locations where the image has strong gradient, the trajectory (II) is similar to the trajectory (III). Observe, that the scheme (11) works well only if the model of the digital image $G_\gamma * u_0$ is continuous. In practice, we can use linear interpolation to get the value of the image $G_\gamma * u_0$, for any point $x \in \overline{D}$. Continuity of the function $G_\gamma * u_0$ guarantees that we will generate random numbers for which the condition Θ will be satisfied.

In this scheme we can also omit the σ function, or in other words we can take σ as an identity

$$\sigma(t, X_t) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The Θ condition guarantees that the diffusion motion is along the edge, and there is no need to duplicate it with σ.

Patchwise Implementation

Let $\mathcal{N}_{x,radius}$ be the square neighborhood of a size $(2 \cdot radius + 1) \times (2 \cdot radius + 1)$ centered at x. The function $g(X_t, u_0, x)$ returns the *weight* which is some measure of distance between intensity of grayscale or color vectors of subimages $u_0(\mathcal{N}_{X_t,radius})$ and $u_0(\mathcal{N}_{x,radius})$. Once this *weight* has been determined, the pixel value $u(x)$ is modified as follows

$$u(x) = u(x) + weight \cdot u_0(X_t).$$

To reduce the number of weight calculations, we will use the same *weight* value for the pixels in the neighborhood, i.e.

$$\forall_{|\hat{x}| \leq radius} u(x + \hat{x}) = u(x + \hat{x}) + weight \cdot u_0(X_t + \hat{x}).$$

The idea of these calculations comes from [20].

5 Algorithm

Below we give algorithms that work for fixed functions $b(t)$, $c(t)$. The first algorithm is an implementation of Example 2, the second one is an implementation of Example 3. In the case of Algorithm 2, we are starting from the functions $b(t)$, $c(t)$ and using the formula (8), we obtain the required assumptions for the function $c(t)$.

In the final Algorithm 3 of the BSDE method, we treat the function $c(t)$ as a parameter. The algorithm code is divided into two parts, depending on whether the reconstructed pixel belongs to an edge or not. For the edge pixels, we run Algorithm 1 and for the remaining pixels, we run Algorithm 2.

6 Generalization to Color Spaces

RGB Space

For RGB images, we need to change in the BSDE algorithm the gradient and partial derivatives to their $\mathbf{R}^n$ space equivalents [54]. To do this, we will use the DiZenzo [60] geometry.

Let $u : D \to \mathbf{R}^n$ be a vector valued image and fix $x \in D$. Consider the function $F_x : V \to \mathbf{R}$, $F_x(v) = \left|\frac{\partial u}{\partial v}(x)\right|^2$, where $V = \{v \in \mathbf{R}^2;\ |v| = 1\}$. We are interested in finding the arguments $\theta_+(u,x), \theta_-(u,x)$ and corresponding values $\lambda_+(u,x) = F_x(\theta_+(u,x))$, $\lambda_-(u,x) = F_x(\theta_-(u,x))$ which maximize and minimize the function F_x, respectively. Note that F_x can be rewritten as $F_x(v) = F_x([v_1, v_2]^T) = v^T \mathbf{G}(x) v$, where in the useful case of color RGB images, $\mathbf{G}$ is defined by the following

$$\mathbf{G}(x) = \begin{bmatrix} \sum_{i=1}^{3} \left(\frac{\partial u_i}{\partial x_1}(x)\right)^2, & \sum_{i=1}^{3} \frac{\partial u_i}{\partial x_1}(x) \frac{\partial u_i}{\partial x_2}(x) \\ \sum_{i=1}^{3} \frac{\partial u_i}{\partial x_1}(x) \frac{\partial u_i}{\partial x_2}(x), & \sum_{i=1}^{3} \left(\frac{\partial u_i}{\partial x_2}(x)\right)^2 \end{bmatrix}, \tag{12}$$

where $u((x_1, x_2)) = (u_1(x_1, x_2), u_2(x_1, x_2), u_3(x_1, x_2))$. Positive eigenvalues $\lambda_+(u,x)$, $\lambda_-(u,x)$ of $\mathbf{G}(x)$ are the maximum and the minimum of F_x while the orthogonal eigenvectors $\theta_+(u,x)$ and $\theta_-(u,x)$ are

Algorithm 1: Pseudo-code for Example 2.

input : u_0 – noisy image, σ – standard deviation of the noise
N, j, m, dt, p – approximation parameters
$(a_k)_{k=0,1\ldots,m-1}$ – coefficients defined by formulas (5), (6), (7)
output: u – reconstructed image
foreach *pixel position* x **do**
  $X_0 = x$
  foreach $n = 1, 2, \ldots N$ **do**
    $weight[0] = a_0$
    $u(x) = u(x) + weight[0] \cdot u_0(x)$
    foreach $k = 1, 2, \ldots j$ **do**
      /* $\mathcal{N}(0,1)$ – *generator of the normal distribution* */
      $X_k = X_{k-1} + dt \begin{bmatrix} \mathcal{N}(0,1) \\ \mathcal{N}(0,1) \end{bmatrix}$
      if $|(G_\gamma * u_0)(X_k) - (G_\gamma * u_0)(X_{k-1})| > p$ **then**
        $k = k - 1$
        /* *Modified Diffusion with Random Terminal Time* */
      else
        $weight[k] = a_k$
        $u(x) = u(x) + weight[k] \cdot u_0(X_k)$
    foreach $k = j+1, j+2, \ldots j-1$ **do**
      $X_k = X_{k-1} + dt \begin{bmatrix} \frac{(G_\gamma * u_0)_{x_1}(X_s)}{|\nabla (G_\gamma * u_0)(X_s)|} \\ \frac{(G_\gamma * u_0)_{x_2}(X_s)}{|\nabla (G_\gamma * u_0)(X_s)|} \end{bmatrix} \mathcal{N}(0,1)$
      $weight[k] = a_k$
      $u(x) = u(x) + weight[k] \cdot u_0(X_k)$
foreach *pixel position* x **do**
  $u(x) = \frac{u(x)}{\sum_k weight[k]}$ /* *Normalization of weights* */

the corresponding variation orientations. We use $N(u,x) = \sqrt{\lambda_+(u,x)}$ as a natural extension of the scalar gradient norm and $\theta_+(u,x)$ and $\theta_-(u,x)$ as equivalents of a gradient and a vector perpendicular to the gradient.

Chromaticity Space

The proposed algorithm works for the model with grayscale and RGB images. Nevertheless, it is worth mentioning an interesting generalization, which is the reconstruction in the chromaticity space.

The chromaticity-brightness model is known to be close to human perception of colors and gives good results [22]. The general idea of the chromaticity-brightness approach is as follows. The brightness component is defined by the Euclidean norm $|u_0|$. The chromaticity component is given by $u_0/|u_0|$ and takes values in S^2, the unit sphere in $\mathbf{R}^3$. The core of this method is to restore these two components independently. In the case of restoring the brightness, we can apply the backward stochastic differential equations. In order to reconstruct the chromaticity component, we can use a method driven by backward stochastic differential equations with reflection (in short RBSDEs). Unfortunately, in the case of RBSDEs we can not apply the above reasoning directly. The problem of the existence and uniqueness of RBSDEs for non-convex domains is presently still open [45]. To

Algorithm 2: Pseudo-code for Example 3.

input : u_0 – noisy image, σ – standard deviation of the noise
N, j, dt, p – approximation parameters
output: u – reconstructed image
foreach *pixel position* x **do**
  $X_0 = x$
  foreach $n = 1, 2, \ldots N$ **do**
    $weight[0] = a_0$
    $u(x) = u(x) + weight[0] \cdot u_0(x)$
    foreach $k = 1, 2, \ldots j$ **do**
      /* $\mathcal{N}(0,1)$ – *generator of the normal distribution* */
      $X_k = X_{k-1} + dt \begin{bmatrix} \mathcal{N}(0,1) \\ \mathcal{N}(0,1) \end{bmatrix}$
      if $|(G_\gamma * u_0)(X_k) - (G_\gamma * u_0)(X_{k-1})| > p$ **then**
        $k = k - 1$
        /* *Modified Diffusion with Random Terminal Time* */
      else
        if $k == j$ **then**
          $weight[k] = a_k$
          foreach $i = j+1, j+2, \ldots m-1$ **do**
            $weight[k] = weight[k] + a[i]$
          /* *The formula* (8) */
          $u(x) = u(x) + weight[k] \cdot u_0(X_k)$
        else
          $weight[k] = a_k g(X_k, u_0, x)$
          foreach xx *such that* $|x - xx| \leq radius$ **do**
            /* *Patchwise Implementation* */
            $u(x + xx) = u(x + xx) + weight[k] \cdot u_0(X_k + xx)$
foreach *pixel position* x **do**
  $u(x) = \frac{u(x)}{\sum_k weight[k]}$ /* *Normalization of weights* */

Algorithm 3: Pseudo-code for the BSDE method

input : u_0 – noisy image, σ – standard deviation of the noise
  $c \geq 0$ – enhancing parameter /* *Definition of the function* $c(t)$ */
output: u – reconstructed image
foreach *pixel position* x **do**
  if $c > 0$ *and* $|\nabla(G_\gamma * u_0)(x)| > \sigma$ **then**
    $u(x)$=Algorithm 1(u_0, σ)
  else
    $u(x)$=Algorithm 2(u_0, σ)

circumvent this problem we consider a model of the chromaticity given by a convex triangle T^2 in $\mathbf{R}^3$ and the brightness as a mean of red, green and blue component.

The RBSDE model to restoration of the chromaticity $u_0^c(x)$ was presented in [11] and has the

form

$$\begin{cases} X_t = x + \int_0^t \sigma(s, X_s)\, dW_s + K_t^{\overline{D}}, \quad t \in [0,T], \\ Y_t = u_0^c(X_S) + \int_t^T c(s)(Y_s - u_0^c(X_s))ds - \int_t^T Z_s\, dW_s + K_T^{T_\epsilon^2} - K_t^{T_\epsilon^2}, t \in [0,T], \end{cases}$$

where

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

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

and $\{K^{T_\epsilon^2} \in \mathbf{R}^3\}$ is a correction process for values of chromaticity i.e. $\{Y_t \in T_\epsilon^2\}_{t\in[0,T]}$. We keep these values in the set

$$T_\epsilon^2 = \{(x_1, x_2, x_3), 1 - \epsilon \leq x_1 + x_2 + x_3 \leq 1 + \epsilon\},$$

where ϵ is a very small number needed just to obtain a domain with a non empty interior. This assumption is required for theoretical results for RBSDEs [47].

7 Approximation Parameters

In this chapter, we will justify the choice of parameter values of the BSDE algorithm.

The primary parameter is the number of iterations of the Monte Carlo method N used to approximate the values of integrals. Clearly, the higher the value of this parameter, the closer the solution is to the ideal one. However, increasing this value also increases the time complexity of the algorithm. We must find a balance between the algorithm's running time and the quality of the resulting image. We define the quality of the resulting image using the PSNR measure (Peak Signal to Noise Ratio). In Figure 3, we can see an example of the algorithm's behavior, specifically the PSNR value of its result based on the number of Monte Carlo iterations. At a value of 20, the increase in PSNR is negligible, which is why we chose this value for our implementation. In Figure 4, we can observe the algorithm's running time dependency on N, which is linear. Please note that reducing N to 10 will decrease the PSNR by approximately 0.2 while doubling the algorithm's speed.

Another parameter that plays a crucial role in approximating a continuous solution is the time discretization, represented as dt. As previously mentioned, the denser the discretization, meaning the smaller the value of dt, the closer we approach the limit solution. However, in the field of stochastic methods, this parameter is pivotal, and setting it too low can disqualify the method due to excessively long running times. To achieve a favorable reconstruction result, a value of $dt = 0.05$ is recommended for the classical Euler scheme [7], albeit at the expense of running time. To address this issue, we introduced a modified Euler scheme (as detailed in Section 4), which introduced another parameter, denoted as p. This modification enabled us to use larger values of dt. The two parameters, dt and p, can be viewed as a single dt with a small value for the classical Euler scheme. In Figure 5, from [7], we can observe that implementing the modified Euler scheme resulted in a remarkable 30-fold acceleration in the reconstruction process! The introduction of this scheme played a pivotal role in the further development of denoising methods based on backward stochastic differential equations. We determined the value of the p parameter to be dependent on the noise level of the input image, with $p = \sigma$, while dt was set at 4 in accordance with recommendations from [7].

The whole time of reconstruction T was divided into two fragments. The period from 0 to t_j and the period from t_j to $T = t_{m-1}$. In the implementation, this means that we have two more parameters j and m at our disposal. The parameter j is responsible for the time in which process

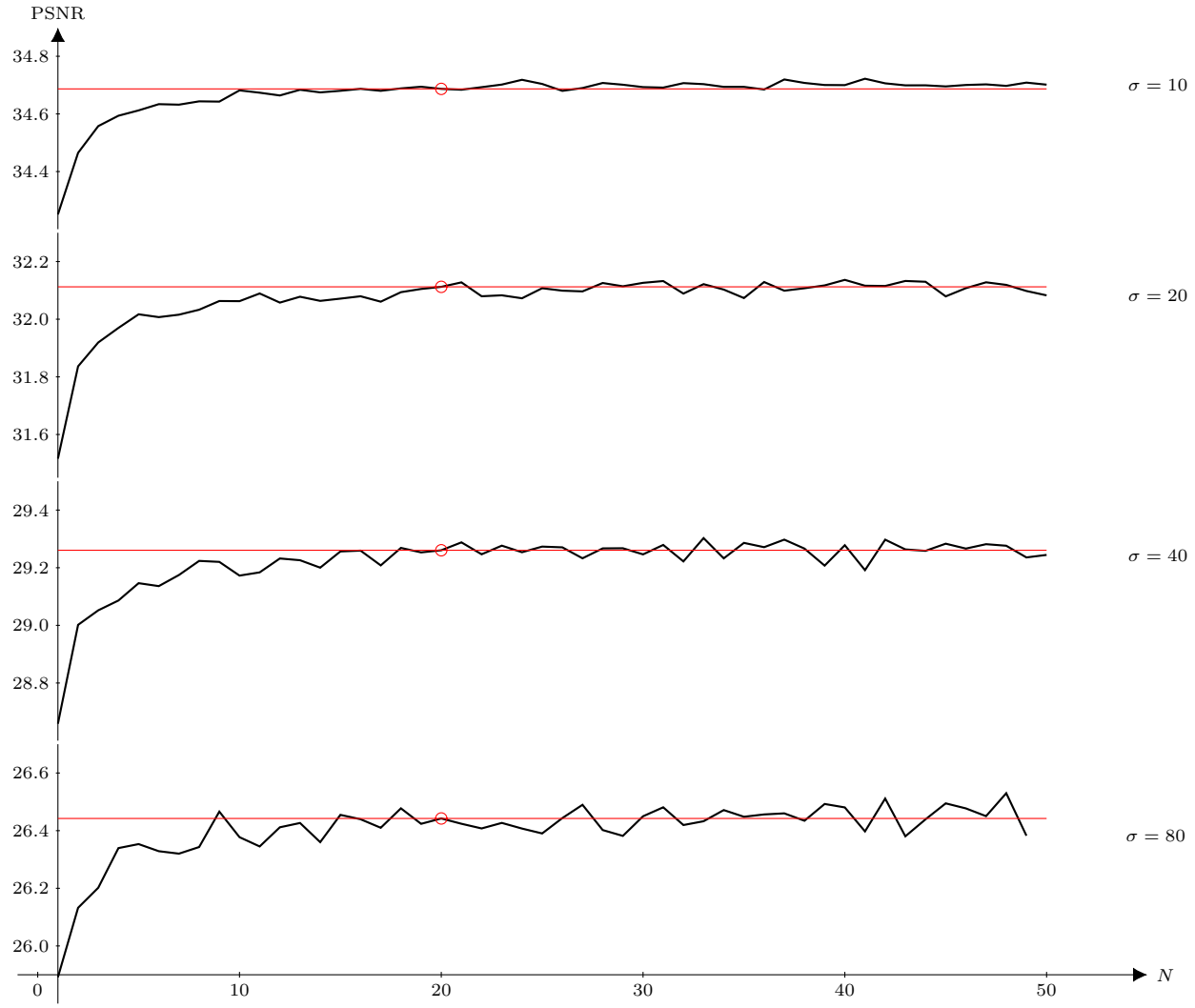


Figure 3: The PSNR measure of image denoising depends on the parameter N. The algorithm was executed using a standard Lena test image with various noise levels σ. Data analysis indicates that selecting a value of N greater than 20 does not yield an improvement in the result's quality.

X determines the neighborhood of the reconstructed pixel with positive weights. An example of a boundary point for an edge point and a non-edge point is shown in Figure 6. The pixels that are used in the reconstruction with positive weights are marked in blue. As before, we made the value of parameter $j = 10 + \sqrt{\sigma}$ dependent on the noise of the input image, following the principle that the greater the noise, the greater the neighborhood.

The parameter k determines the number of additional steps we take with negative weights. This procedure is aimed at counteracting the smoothing effect on the image. In implementation, we take only two steps toward the gradient, i.e., we assume $m-1 = j+2$. Figure 7 illustrates the distribution of negative weights depending on the parameter c for two different functions $b(t)$. A higher value of $b(t)$ increases the significance of pixels located closer to the reconstructed one.

8 Experimental Results

In this section, we present experimental results illustrating the difference between the BSDE algorithm and other methods. We use the implementation of compared methods from Image Processing On Line: K-SVD [38], NLM [20], BM3D [36], NL-Bayes [37] and DCT [59]. Parameters of these

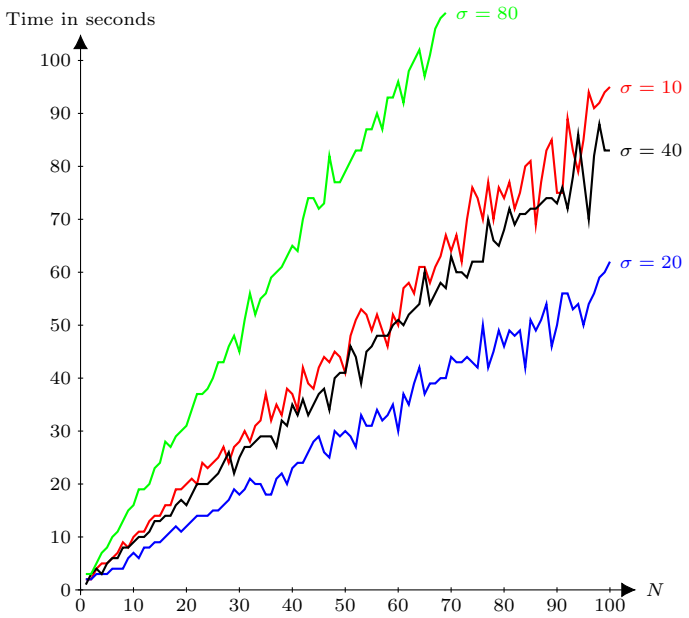
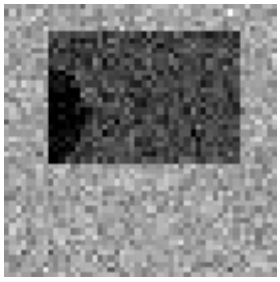
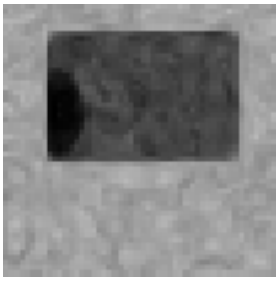


Figure 4: The algorithm's running time is directly proportional to the value of the N parameter. The algorithm was executed using a standard Lena test image with various noise levels σ on a computer equipped with an Intel Core i9 processor.

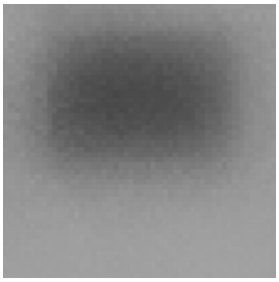
(a) Original image: 50×50 pixels



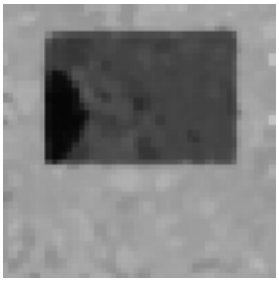
(b) Noisy image: standard deviation of the noise $\sigma = 25$



(c) Euler's scheme: $dt = 0.05$ (480 seconds)



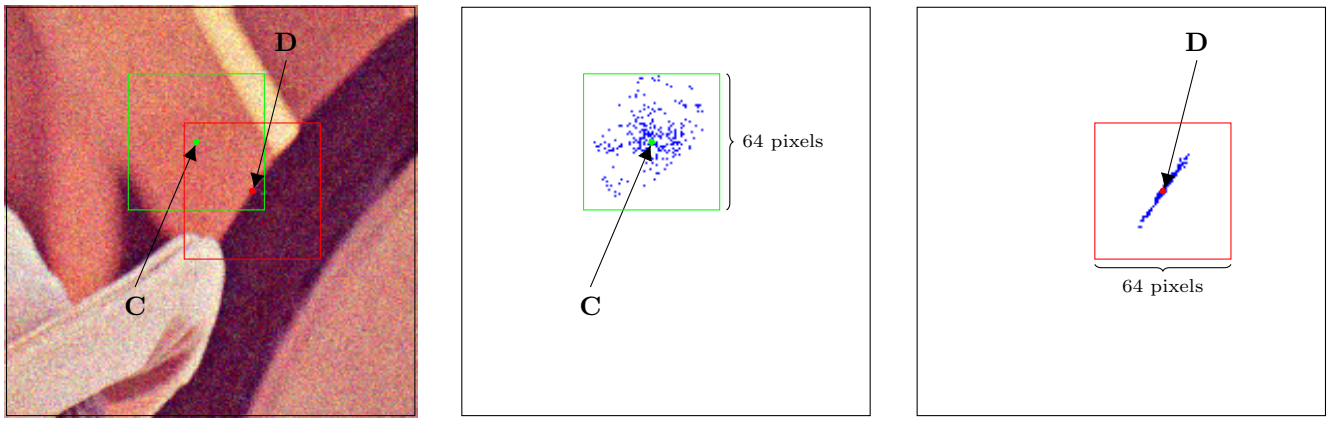
(d) Euler's scheme: $dt = 4$ (7 seconds)



(e) Modified diffusion with random terminal time: $dt = 4$ (14 seconds)

Figure 5: Comparison of reconstruction results using Euler's approximation with long and short time-step discretization and modified diffusion with long-time-step discretization shows that the reconstructed images (c) and (e) are similar, but the reconstruction time has been reduced.

approaches were set to the default values as recommended by the authors. Our algorithm's parameter is a standard noise deviation ρ and it works for a fixed function $b(t) = 0.05$. Other parameters



(a) Noisy Lena image with $\sigma = 20$ (b) Neighborhood of the reconstructed pixel C (c) Neighborhood of the reconstructed pixel D

Figure 6: Here's an example of a reconstruction neighborhood (marked as blue points) for an edge point D and a background point C. In the case of pixel C, the neighborhood is evenly distributed around the center, while for point D, it takes the shape of an edge.

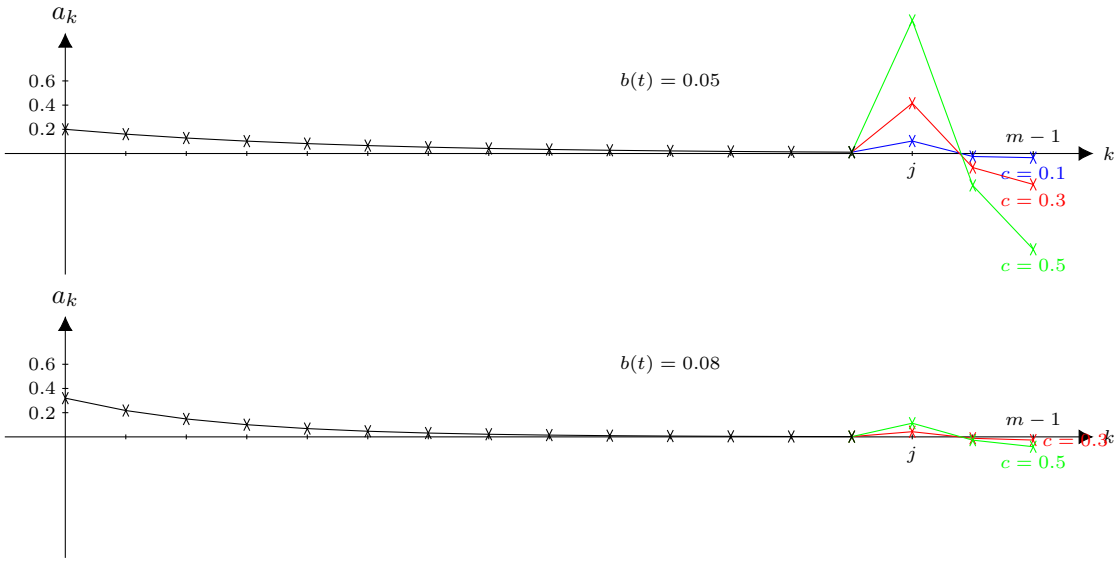
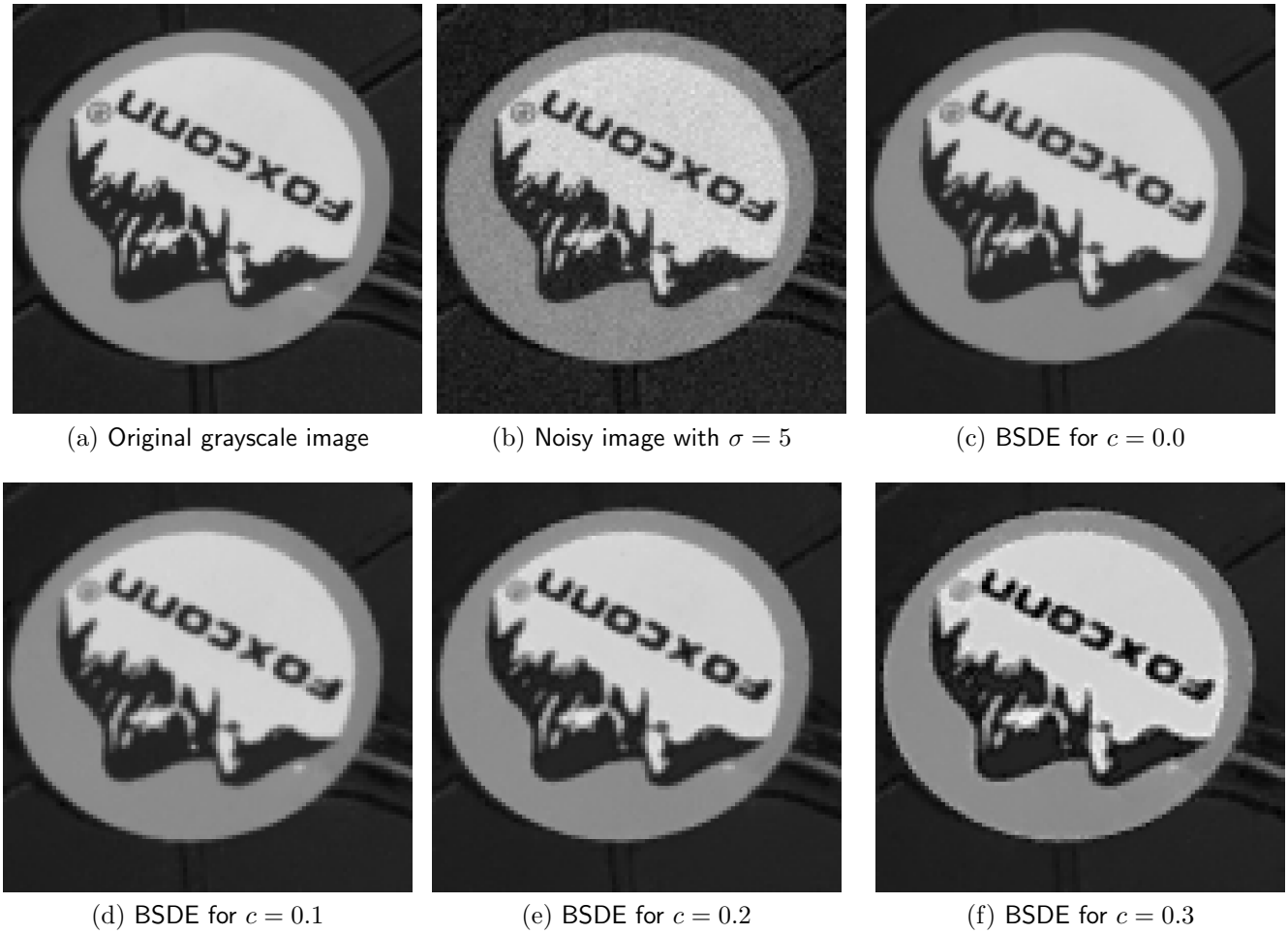


Figure 7: Distribution of weights a_k depending on functions $b(t)$ and $c(t) = c$.

depend only on the value of ρ. Choosing their values, we followed the principle of maximizing the Peak Signal to Noise Ratio (in short PSNR). We can also release parameter c, which is responsible for sharpening the image. However, a too high value of this parameter has a negative impact on the value of PSNR (although the picture is more readable for our eyes). This effect can be seen in Figures 8 and 9.

When comparing the BSDE algorithm with other approaches, we can notice one regularity. The BSDE method smooths out the noise more, as shown in Figure 11 and Figure 13, which may result in a loss of detail in the image. This effect is clearly visible in Figure 11 and the inscription "SeMA" (compare with the original and noisy images in Figure 10). The compared methods have a problem with noise reduction around this word.

The proposed algorithm gives good results in the case of JPEG artifact reduction, which was shown in [17]. In the case of reduction of JPEG artifacts, we have to combine the ρ parameter with the compression quality q. In JPEG standards, the compression quality q is always known and is



(a) Original grayscale image (b) Noisy image with $\sigma = 5$ (c) BSDE for $c = 0.0$

(d) BSDE for $c = 0.1$ (e) BSDE for $c = 0.2$ (f) BSDE for $c = 0.3$

Figure 8: Restoration of Computer image using BSDE method with different values of parameter c. A larger value of c increases the sharpness of the image.

expressed as a percentage. An image at 100% quality has no loss. In [17] the following formula to evaluate the ρ parameter was proposed

$$\rho = \max\{0, -0.3q + 20\}, \tag{13}$$

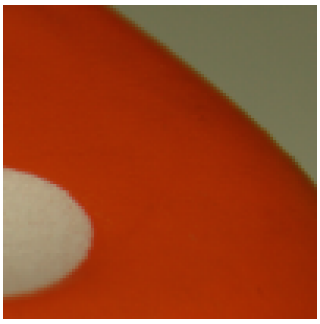
where q is a JPEG compression quality of the image. JPEG artifacts are mainly due to the coarse quantization of the high-frequency DCT coefficients, making the decompressed image to exhibit noisy patterns known as ringing or mosquito noise near the edge. The second type of artifacts are blocking artifacts, which are mainly due to the coarse quantization of low-frequency DCT coefficients, making the decompressed image like a mosaic at smooth regions. As shown in Figure 14 the BSDE method removed artifacts very well in both cases.

(a) Original RGB image

(b) Noisy image with $\sigma = 10$

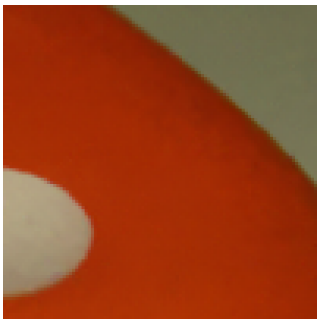
(c) BSDE for $c = 0.0$

(d) BSDE for $c = 0.4$

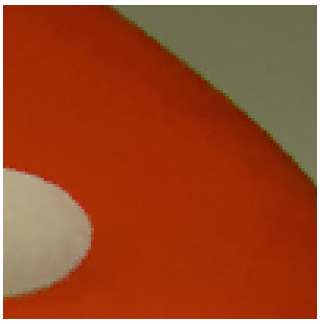


(e) Zoom of the original image

(f) Zoom of the noisy image

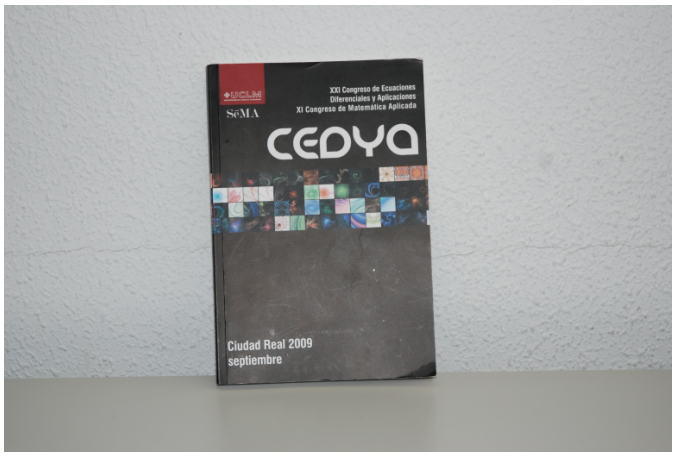


(g) Zoom of the restored image with $c = 0.0$

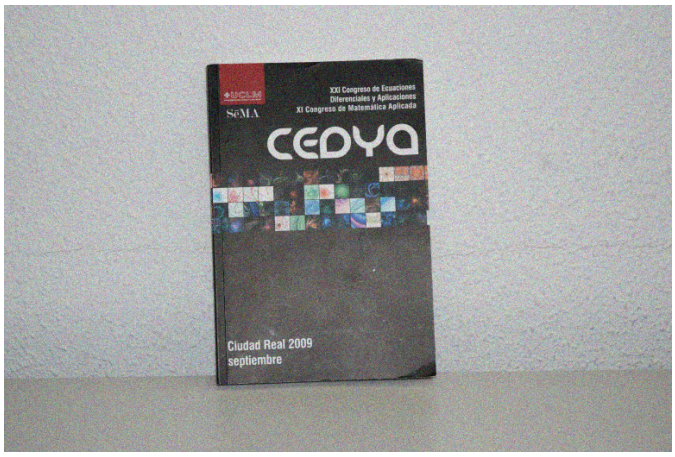


(h) Zoom of the restored image with $c = 0.4$

Figure 9: Restoration of Dice image using BSDE method. A too high value of parameter c causes that the reconstructed image differs significantly from the original image.



(a) Original image



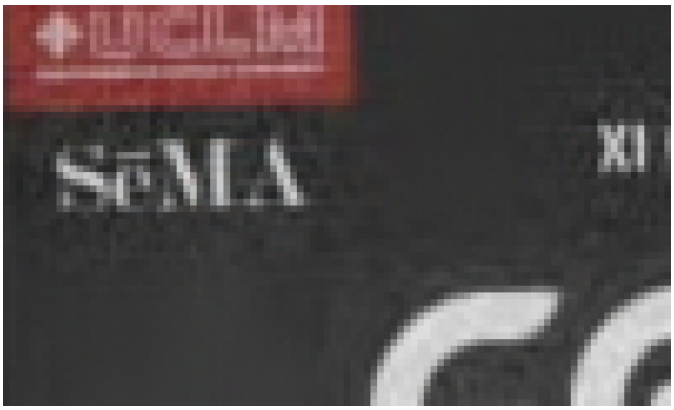
(b) Noisy image with $\sigma = 15$

(c) Zoom of the original image

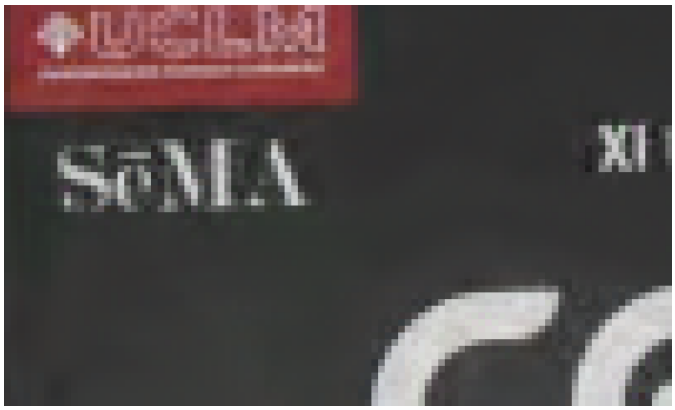
(d) Zoom of the noisy image

Figure 10: The input Book image for the compared denoising algorithms.

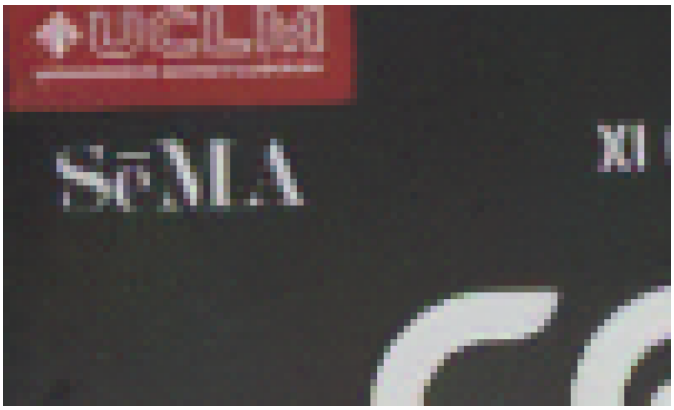
(a) K–SVD $PSNR = 36.58$



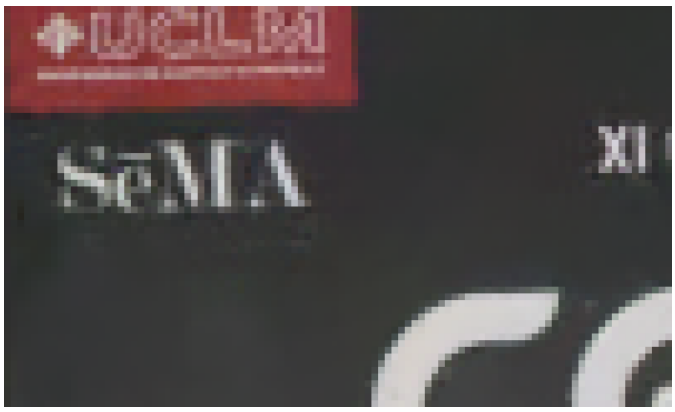
(b) DCT $PSNR = 36.49$



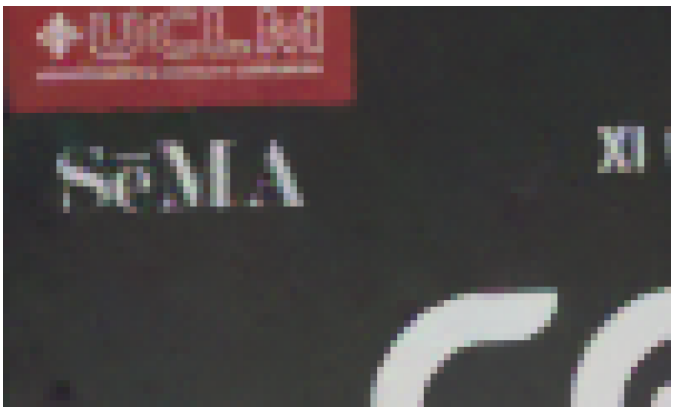
(c) BM3D $PSNR = 37.67$



(d) NLM $PSNR = 36.26$



(e) NL–Bayes $PSNR = 37.18$



(f) BSDE $PSNR = 36.50$

Figure 11: Reconstruction results for various algorithms on the Book image (see Figure 10) and a visual comparison with the proposed method.

(a) Original image

(b) Noisy image with $\sigma = 20$

(c) Zoom of the original image

(d) Zoom of the noisy image

Figure 12: The input Traffic image for the compared denoising algorithms.

(a) K–SVD $PSNR = 30.71$

(b) DCT $PSNR = 29.70$

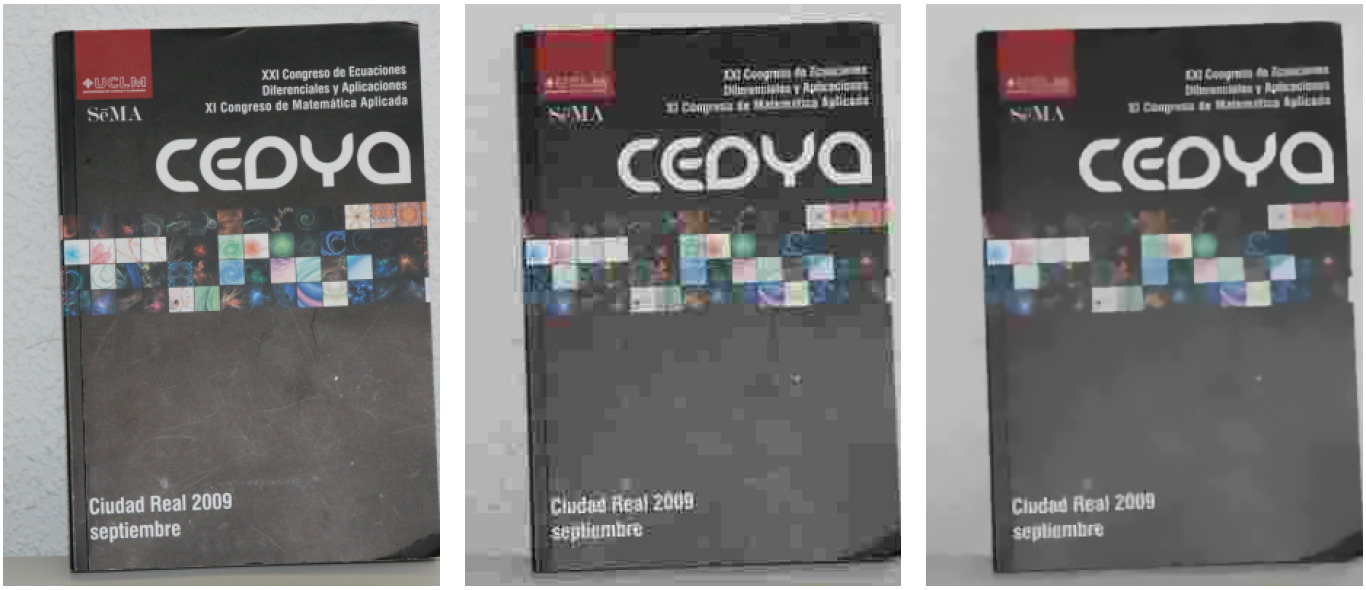
(c) BM3D $PSNR = 30.80$

(d) NLM $PSNR = 30.11$

(e) NL–Bayes $PSNR = 31.25$

(f) BSDE $PSNR = 30.02$

Figure 13: Reconstruction results for various algorithms on the Traffic image (see Figure 12) and a visual comparison with the proposed method.



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

Figure 14: Reduction of JPEG artifacts. The BSDE algorithm was applied with $\sigma = 17$ and $c = 0.0$.

Image Credits

 by M. Colom, CC-BY

 by A. Buades, CC-BY

 Standard test image

 Synthetic image provided by the author

References

[1] R. Abraham and O. Riviere, *Forward-Backward Stochastic Differential Equations and PDE with Gradient Dependent Second Order Coefficients*, ESAIM: Probability and Statistics, 10 (2006), p. 184–205, https://doi.org/10.1051/ps:2006005.

[2] M. Aharon, M. Elad, and A. Bruckstein, *K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation*, IEEE Transactions on Signal Processing, 54 (2006), pp. 4311–4322, https://doi.org/10.1109/TSP.2006.881199.

[3] G. Aubert and P. Kornprobst, *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*, vol. 147, Springer, 2006. ISBN 978-0-387-44588-5.

[4] D. Borkowski, *Chromaticity Denoising Using Solution to the Skorokhod Problem*, in Image Processing Based on Partial Differential Equations, Springer, 2007, pp. 149–161, https://doi.org/10.1007/978-3-540-33267-1_9.

[5] ——, *Modified Diffusion to Image Denoising*, in Computer Recognition Systems 2, Springer, 2007, pp. 92–99, https://doi.org/10.1007/978-3-540-75175-5_12.

[6] ——, *Smoothing, Enhancing Filters in Terms of Backward Stochastic Differential Equations*, in International Conference on Computer Vision and Graphics, Springer, 2010, pp. 233–240, https://doi.org/10.1007/978-3-642-15910-7_26.

[7] ——, *Euler's Approximations to Image Reconstruction*, in International Conference on Computer Vision and Graphics, Springer, 2012, pp. 30–37, https://doi.org/10.1007/978-3-642-33564-8_4.

[8] ——, *Stochastic Approximation to Reconstruction of Vector-Valued Images*, in International Conference on Computer Recognition Systems (CORES), Springer, 2013, pp. 393–402, https://doi.org/10.1007/978-3-319-00969-8_38.

[9] ——, *Non Local Means on Random Anisotropic Sub-Image*, International Journal of Imaging & Robotics, 15 (2015), pp. 1–13. http://www.ceser.in/ceserp/index.php/iji/article/view/3602.

[10] ——, *Forward and Backward Filtering Based on Backward Stochastic Differential Equations*, Inverse Problems & Imaging, 10 (2016), pp. 305–325, https://doi.org/10.3934/ipi.2016002.

[11] ——, *Restoration of Colour Images Using Backward Stochastic Differential Equations with Reflection*, in International Conference on Computer Analysis of Images and Patterns, Springer, 2019, pp. 317–329, https://doi.org/10.1007/978-3-030-29891-3_28.

[12] D. Borkowski, A. Jakubowski, and K. Jańczak-Borkowska, *Feynman-Kac Formula and Restoration of High Iso Images*, in International Conference on Computer Vision and Graphics, Springer, 2014, pp. 100–107, https://doi.org/10.1007/978-3-319-11331-9_13.

[13] D. Borkowski and K. Jańczak-Borkowska, *Application of Backward Stochastic Differential Equations to Reconstruction of Vector-Valued Images*, in International Conference on Computer Vision and Graphics, Springer, 2012, pp. 38–47, https://doi.org/10.1007/978-3-642-33564-8_5.

[14] ——, *Image Restoration Using Anisotropic Stochastic Diffusion Collaborated with Non Local Means*, in IFIP International Conference on Computer Information Systems and Industrial Management, Springer, 2013, pp. 177–189, https://doi.org/10.1007/978-3-642-40925-7_18.

[15] ——, *Random NL-Means to Restoration of Colour Images*, in International Conference on Digital Image Computing: Techniques and Applications (DICTA), IEEE, 2015, pp. 1–8, https://doi.org/10.1109/DICTA.2015.7371298.

[16] ——, *Image Denoising Using Backward Stochastic Differential Equations*, in International Conference on Man-Machine Interactions (ICMMI), Springer, 2018, pp. 185–194, https://doi.org/10.1007/978-3-319-67792-7_19.

[17] ——, *Reduction of JPEG Artifacts Using BSDEs*, in International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2021, pp. 1–10, https://doi.org/10.24132/CSRN.2021.3101.1.

[18] A. Buades, B. Coll, and J. M. Morel, *A Non-Local Algorithm for Image Denoising*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, IEEE, 2005, pp. 60–65, https://doi.org/10.1109/CVPR.2005.38.

[19] A. Buades, B. Coll, and J. M. Morel, *A Review of Image Denoising Algorithms, with a New One*, Multiscale Modeling & Simulation, 4 (2005), pp. 490–530, https://doi.org/10.1137/040616024.

[20] ——, *Non-Local Means Denoising*, Image Processing On Line, 1 (2011), pp. 208–212, https://doi.org/10.5201/ipol.2011.bcm_nlm.

[21] F. Catté, P. L. Lions, J. M. Morel, and T. Coll, *Image Selective Smoothing and Edge Detection by Nonlinear Diffusion*, SIAM Journal on Numerical Analysis, 29 (1992), pp. 182–193, https://doi.org/10.1137/0729012.

[22] T. F. Chan, S. H. Kang, and J. Shen, *Total Variation Denoising and Enhancement of Color Images Based on the CB and HSV Color Models*, Journal of Visual Communication and Image Representation, 12 (2001), pp. 422–435, https://doi.org/10.1006/jvci.2001.0491.

[23] T. F. Chan and J. Shen, *Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods*, SIAM, 2005. ISBN 0-89871-589-X.

[24] Y. W. Chiang and B. J. Sullivan, *Multi-Frame Image Restoration Using a Neural Network*, in Midwest Symposium on Circuits and Systems, IEEE, 1989, pp. 744–747, https://doi.org/10.1109/MWSCAS.1989.101962.

[25] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, *Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering*, IEEE Transactions on Image Processing, 16 (2007), pp. 2080–2095, https://doi.org/10.1109/TIP.2007.901238.

[26] A. Danielyan, V. Katkovnik, and K. Egiazarian, *BM3D Frames and Variational Image Deblurring*, IEEE Transactions on Image Processing, 21 (2011), pp. 1715–1728, https://doi.org/10.1109/TIP.2011.2176954.

[27] D. L. Donoho and I. M. Johnstone, *Ideal Spatial Adaptation by Wavelet Shrinkage*, Biometrika, 81 (1994), pp. 425–455, https://doi.org/10.1093/biomet/81.3.425.

[28] D. Duffie and L. G. Epstein, *Asset Pricing with Stochastic Differential Utility*, The Review of Financial Studies, 5 (1992), pp. 411–436, https://doi.org/10.1093/rfs/5.3.411.

[29] ——, *Stochastic Differential Utility*, Econometrica: Journal of the Econometric Society, (1992), pp. 353–394, https://doi.org/10.2307/2951600.

[30] A. Efros and T. Leung, *Texture Synthesis by Non-Parametric Sampling*, in IEEE International Conference on Computer Vision, vol. 2, IEEE, 1999, pp. 1033–1038, https://doi.org/10.1109/ICCV.1999.790383.

[31] T. Goldstein and S. Osher, *The Split Bregman Method for L1-Regularized Problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 323–343, https://doi.org/10.1137/080725891.

[32] S. Hamadene and J. P. Lepeltier, *Zero-Sum Stochastic Differential Games and Backward Equations*, Systems & Control Letters, 24 (1995), pp. 259–263, https://doi.org/10.1016/0167-6911(94)00011-J.

[33] K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778, https://doi.org/10.1109/CVPR.2016.90.

[34] N. E. Karoui, S. Peng, and M. C. Quenez, *Backward Stochastic Differential Equations in Finance*, Mathematical Finance, 7 (1997), pp. 1–71, https://doi.org/10.1111/1467-9965.00022.

[35] V. Katkovnik, A. Danielyan, and K. Egiazarian, *Decoupled Inverse and Denoising for Image Deblurring: Variational BM3D-Frame Technique*, in IEEE International Conference on Image Processing, IEEE, 2011, pp. 3453–3456, https://doi.org/10.1109/ICIP.2011.6116455.

[36] M. Lebrun, *An Analysis and Implementation of the BM3D Image Denoising Method*, Image Processing On Line, 2012 (2012), pp. 175–213, https://doi.org/10.5201/ipol.2012.l-bm3d.

[37] M. Lebrun, A. Buades, and J. M. Morel, *Implementation of the Non-Local Bayes (NL-Bayes) Image Denoising Algorithm*, Image Processing On Line, 3 (2013), pp. 1–42, https://doi.org/10.5201/ipol.2013.16.

[38] M. Lebrun and A. Leclaire, *An Implementation and Detailed Analysis of the K-SVD Image Denoising Algorithm*, Image Processing On Line, 2 (2012), pp. 96–133, https://doi.org/10.5201/ipol.2012.llm-ksvd.

[39] J. Ma, P. Protter, J. S. Martín, and S. Torres, *Numerical Method for Backward Stochastic Differential Equations*, Annals of Applied Probability, (2002), pp. 302–316. https://www.jstor.org/stable/2699935.

[40] J. Mairal, M. Elad, and G. Sapiro, *Sparse Representation for Color Image Restoration*, IEEE Transactions on Image Processing, 17 (2007), pp. 53–69, https://doi.org/10.1109/TIP.2007.911828.

[41] X. Mao, C. Shen, and Y. B. Yang, *Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections*, Advances in Neural Information Processing Systems, 29 (2016).

[42] E. Pardoux, *Backward Stochastic Differential Equations and Viscosity Solutions of Systems of Semilinear Parabolic and Elliptic PDEs of Second Order*, in Stochastic Analysis and Related Topics VI. Progress in Probability, Vol 42, Springer, 1998, pp. 79–127, https://doi.org/10.1007/978-1-4612-2022-0_2.

[43] E. Pardoux and S. Peng, *Adapted Solution of a Backward Stochastic Differential Equation*, Systems & Control Letters, 14 (1990), pp. 55–61, https://doi.org/10.1016/0167-6911(90)90082-6.

[44] ——, *Backward Stochastic Differential Equations and Quasilinear Parabolic Partial Differential Equations*, in Stochastic Partial Differential Equations and Their Applications. Lecture Notes in Control and Information Sciences, Vol 176, Springer, 2005, pp. 200–217, https://doi.org/10.1007/BFb0007334.

[45] E. Pardoux and A. Râșcanu, *Stochastic Differential Equations, Backward SDEs, Partial Differential Equations*, vol. 69, Stochastic Modelling and Applied Probability, Springer, 2014, https://doi.org/10.1007/978-3-319-05714-9.

[46] P. Perona and J. Malik, *Scale-Space and Edge Detection Using Anisotropic Diffusion*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 12 (1990), pp. 629–639, https://doi.org/10.1109/34.56205.

[47] A. G. Petit and E. Pardoux, *Equations Différentielles Stochastiques Rétrogrades Réfléchies dans Un Convexe*, Stochastics: An International Journal of Probability and Stochastic Processes, 57 (1996), pp. 111–128.

[48] R. Pettersson, *Approximations for Stochastic Differential Equations with Reflecting Convex Boundaries*, Stochastic Processes and their Applications, 59 (1995), pp. 295–308, https://doi.org/10.1016/0304-4149(95)00040-E.

[49] D. Ren, W. Shang, P. Zhu, Q. Hu, D. Meng, and W. Zuo, *Single Image Deraining Using Bilateral Recurrent Network*, IEEE Transactions on Image Processing, 29 (2020), pp. 6852–6863, https://doi.org/10.1109/TIP.2020.2994443.

[50] W. H. Richardson, *Bayesian-Based Iterative Method of Image Restoration*, Journal of the Optical Society of America, 62 (1972), pp. 55–59, https://doi.org/10.1364/JOSA.62.000055.

[51] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear Total Variation Based Noise Removal Algorithms*, Physica D: Nonlinear Phenomena, 60 (1992), pp. 259–268, https://doi.org/10.1016/0167-2789(92)90242-F.

[52] L. Słomiński, *Euler's Approximations of Solutions of SDEs with Reflecting Boundary*, Stochastic Processes and their Applications, 94 (2001), pp. 317–337.

[53] H. Tanaka, *Stochastic Differential Equations with Reflecting Boundary Condition in Convex Regions*, Hiroshima Mathematical Journal, 9 (1979), pp. 163–177, https://doi.org/10.1142/9789812778550_0013.

[54] D. Tschumperle and R. Deriche, *Diffusion PDEs on Vector-Valued Images*, IEEE Signal Processing Magazine, 19 (2002), pp. 16–25, https://doi.org/10.1109/MSP.2002.1028349.

[55] J. Weickert, *Theoretical Foundations of Anisotropic Diffusion in Image Processing*, Springer, 1996, https://doi.org/10.1007/978-3-7091-6586-7_13.

[56] ——, *Coherence-Enhancing Diffusion Filtering*, International Journal of Computer Vision, 31 (1999), pp. 111–127.

[57] L. P. Yaroslavsky, *Local Adaptive Image Restoration and Enhancement with the Use of DFT and DCT in a Running Window*, in Wavelet Applications in Signal and Image Processing IV, vol. 2825, SPIE, 1996, pp. 2–13, https://doi.org/10.1117/12.255218.

[58] L. P. Yaroslavsky, K. O. Egiazarian, and J. T. Astola, *Transform Domain Image Restoration Methods: Review, Comparison, and Interpretation*, Nonlinear Image Processing and Pattern Analysis XII, 4304 (2001), pp. 155–169, https://doi.org/10.1117/12.424970.

[59] G. Yu and G. Sapiro, *DCT Image Denoising: a Simple and Effective Image Denoising Algorithm*, Image Processing On Line, 1 (2011), pp. 292–296, https://doi.org/10.5201/ipol.2011.ys-dct.

[60] S. D. Zenzo, *A Note on the Gradient of a Multi-Image*, Computer Vision, Graphics, and Image Processing, 33 (1986), pp. 116–125, https://doi.org/10.1016/0734-189X(86)90223-9.

[61] Y. T. Zhou, R. Chellappa, and B. K. Jenkins, *A Novel Approach to Image Restoration Based on a Neural Network*, in International Conference on Neural Networks, 1987.