Published in Image Processing On Line on YYYY–MM–DD.
Submitted on YYYY–MM–DD, accepted on YYYY–MM–DD.
ISSN 2105–1232 © YYYY IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
http://www.ipol.im/

Image Segmentation using Backward Stochastic Differential Equations

Dariusz Borkowski [1]

[1] Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, Poland (dbor@mat.umk.pl)

PREPRINT December 21, 2025

Abstract

We introduce a novel image segmentation algorithm based on the methodology of approximating solutions to backward stochastic differential equations (BSDEs). The segmentation method repeats the BSDE reconstruction process, with the parameters of these equations changing in subsequent steps. We are interested in a sequence of images driven by BSDE solutions. As the segmentation result, we define the limit of these images. The segmentation algorithm is based on the BSDEs approximation methodology. By their nature, stochastic tools, particularly the Monte Carlo method, have high computational complexity. There are concerns about the running time of the proposed method, especially if we are considering a sequence of stochastic solutions. Experimental segmentation results show that it is possible to obtain results quickly and that the algorithm yields excellent results for images with intense noise.

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 instruction are included in the README.txt file of the archive.

Keywords: segmentation, stochastic process, stochastic differential equations, noise

1 Introduction

Digital image segmentation is a fundamental process in computer vision and image processing that involves dividing an image into meaningful regions or segments. These segments can correspond to objects, features, or areas of interest within the image. Image segmentation aims to simplify an image's representation, making it easier to analyse and extract valuable information. Segmentation algorithms allow us to make initial data treatment for further analysis, which is very important in astronomy, biology or medicine.

[1]http://www.ipol.im/

Various methods and techniques have been developed in image segmentation to cater to different types of images and segmentation objectives. Threshold-based segmentation is a straightforward method where pixels are categorised into segments based on a specified intensity threshold. It is beneficial for images with apparent intensity differences [44]. Edge-based segmentation aims to detect and highlight edges within an image. It identifies abrupt changes in pixel intensity, which often correspond to object boundaries or prominent features [15, 51]. Level sets are an essential category of modern image segmentation techniques based on partial differential equations. Level sets advance a contour like a rubber band until the contour hits an object boundary [36, 37, 43, 57]. Region-based segmentation groups pixels with similar characteristics into regions. This method is suitable for segmenting images into coherent areas based on attributes like colour, texture or intensity [2, 27, 49, 50]. Region merging hierarchical segmentation creates a hierarchy of image regions, allowing for multi-level analysis. This method is valuable for identifying large and small structures within images [35, 41, 52]. The mean shift method is based on density estimations in a feature space and efficiently finds peaks in a high-dimensional data distribution without explicitly computing the complete function [19, 20]. Active contour (snakes) segmentation is ideal for capturing complex and irregular object boundaries. It minimises an energy function, making it adaptable to various contours [4, 16, 17, 33, 42, 55]. Segmentation using artificial neural networks leverages deep learning techniques, primarily convolutional neural networks, for precise and efficient image segmentation [3, 18, 24, 25, 28, 29, 30, 31, 34, 38, 47, 56]. These segmentation methods cater to various applications, from medical image analysis and object tracking to scene understanding and industrial quality control. The choice of method depends on the specific segmentation task, the characteristics of the images, and the desired level of detail. In this project, we will address the problem of segmenting noisy images and propose a new method based on backward stochastic differential equations.

The backward stochastic differential equations (BSDEs) were introduced by Pardoux and Peng [45], who proved the existence and uniqueness of adapted solutions under suitable assumptions. Duffie and Epstein [21, 22] introduced stochastic differential utilities in economics models as the solutions for certain BSDEs. Since then, it has been widely recognised that BSDEs provide a valuable framework for formulating many problems in mathematical finance [32]. They have also proven helpful in addressing issues in stochastic control and differential games [26]. Many papers (for instance, [46]) 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 generalisation of the celebrated Feynman-Kac formula. Through all these results, a formal dictionary of the relations between BSDEs and PDEs can be established, suggesting that the existence and unique results that can be obtained on the one side should have their counterparts on the other. In image processing, one can find both theoretical results [1] and some practical aspects [5, 7, 10, 11, 12, 13, 14] related to the applications of BSDEs for image denoising. To our knowledge, this type of equation has never been used for image segmentation. Based on the methodology of approximating solutions to BSDEs, we introduce a novel image segmentation algorithm in this paper.

The paper is organised as follows: section 2 briefly reviews basic denoising concepts related to BSDEs and contains the segmentation methodology in terms of BSDEs. Chapter 3 presents details of the numerical approximation of the proposed method. The algorithm is given in section 4. The selection of parameter values is presented in chapter 5, with experimental results in section 6 and concluding remarks in section 7.

2 Methodology

Let D be a bounded, convex domain in $\mathbf{R}^2$, $u : \overline{D} \to \mathbf{R}^3$ be an original colour RGB image and $u_0 : \overline{D} \to \mathbf{R}^3$ be the observed image of the form

$$u_0 = B\,u + \eta,$$

where η stands for white Gaussian noise and B is a linear operator representing the blur. We assume that u and u_0 are appropriately regular. We are given u_0, the problem is to reconstruct u. This is a typical example of an inverse problem.

We consider one of the most popular methods of reconstructing digital images. It is a convolution of the noisy image with a two-dimensional Gaussian mask:

$$u(x) = \int_{\mathbf{R}^2} G_{\sqrt{T}}(x-y)u_0(y)\,dy,$$

where u_0 is a noisy image, $G_\gamma(x) = \frac{1}{2\pi\gamma^2}e^{-\frac{|x|^2}{2\gamma^2}}$ is two-dimensional Gaussian mask with variance γ^2. A mean value may represent an image reconstructed with this filter:

$$u(x) = \mathbf{E}\left[u_0\left(x + W_T\right)\right],$$

where W is a two-dimensional Wiener process starting from zero.

It is known that reconstruction with a Gaussian filter is unsatisfactory, and the obtained image has blurred edges. Therefore, we use the diffusion process X instead of the Wiener process. Then, the reconstructed image u takes the following form:

$$u(x) = \mathbf{E}\left[u_0\left(x + X_T\right)\right].$$

The construction of a process X is modelled by describing diffusion for models of reconstruction images based on partial differential equations. The appropriate form of this process gives us images without noise and with preserving edges.

When we described the above example, we assumed that the image is defined on the whole plane. Since images are considered functions defined on a bounded convex set $\overline{D}$, we have to introduce additional restrictions on process X, which takes values in $\overline{D}$. For this purpose, we use the so-called Skorokhod problem.

The Skorokhod problem is a mathematical tool that lets us find, for a given function, another function (stochastic process) with similar properties to the original but taking values in a given set. In our case, after solving the Skorokhod problem, we have a model:

$$u(x) = \mathbf{E}\left[u_0\left(x + X_T + K_T^{\overline{D}}\right)\right],$$

where $X + K^{\overline{D}}$ is a solution to the Skorokhod problem associated with X and $\overline{D}$.

It should be noted that in multidimensional cases, there are no explicit formulas for solving the Skorokhod problem. To calculate $u(x)$, we have to perform multiple simulations of the trajectory of the diffusion process and then average the obtained results (Monte Carlo method).

The starting point of the next consideration is the following observation modelled on methods of solving backward stochastic differential equations. The fact that every martingale in the space with Wiener filtration has representation as stochastic integral implies that a process Z exists (meeting appropriate technical conditions) with property.

$$Y_t = \mathbf{E}\left[u_0\left(x + X_T + K_T^{\overline{D}}\right)|\mathcal{W}_t\right] = u_0(x + X_T + K_T^{\overline{D}}) - \int_t^T Z_s\,dW_s.$$

In particular, Y is a process with continuous trajectories, and its initial value Y_0 equals $u(x)$:

$$Y_0 = \mathbf{E}\left[u_0\left(x + X_T + K_T^{\overline{D}}\right)\right] = u(x).$$

Following this idea, we can generalise the model and consider backward stochastic differential equations of the form

$$Y_t = u_0(x + X + K^{\overline{D}}) + \int_t^T f(s, Y_s, X_s + K_s^{\overline{D}})ds - \int_t^T Z_s\, dW_s, \quad t \in [0, T],$$

keeping in mind that the value of the process Y at time zero is a reconstructed pixel. A proper form of random variable $u_0(x + X + K^{\overline{D}})$ and drift function f give us a better smoothing effect while preserving edges and enhancing and sharpening them.

The segmentation method repeats the BSDE reconstruction process, with the parameters of these equations changing in subsequent steps. In the case of image segmentation, we consider the sequence of BSDEs:

$$Y_t^{x,n} = u_0^n(x + X^n + K^{n,\overline{D}}) + \int_t^T f_n(s, Y_s^{x,n}, X_s^n + K_s^{n,\overline{D}})ds - \int_t^T Z_s^n\, dW_s, \quad t \in [0, T],$$

where $u_0^1 = u_0$. We are interested in a sequence of images $(u^n)_{n=1,2,\cdots+\infty}$ driven by BSDE solutions $(Y^{x,n}, Z^n)_{x\in\overline{D}}$, i.e. $u^n(x) = Y_0^{x,n}$, where $u_0^n = u^{n-1}$. The result of segmentation is defined as a limit

$$u(x) = \lim_{n\to+\infty} u^n(x), \;\; x \in \overline{D}.$$

3 Approximation

BSDE solution

For the segmentation problem, we will use the case of the BSDE equation described in [12], where

$$f_n(t, y) = \begin{cases} c_n(t)\left(y - u^{n-1}(X_t^n + K_t^{n,\overline{D}})\right), & t \in [S,\, T] \\ b(t)\left(g(X_t^n, u_0, x)u_0(X_t^n + K_t^{n,\overline{D}}) - y\right), & t \in [0,\, S). \end{cases}$$

We will use the same values for functions $b(.)$ and $g(.)$, which are responsible for the weight values of pixels near the reconstructed one. A more significant value of $b(t)$ increases the importance of pixels closer to the reconstructed one, while function g determines the weights based on the similarity of patches. We will make changes to function $c_n(.)$, whose values will change in the next n steps, i.e.

$$c_n(t) = \begin{cases} 0 & \text{if} \quad t \in [0,\, S), \\ c_n & \text{if} \quad t \in [S,\, T], \end{cases}$$

The parameter c_n is responsible for the effect of edge sharpening, which takes place from time T to S.

Theorem ([12]). *Let $S < T$, $u^{n-1} : \overline{D} \to \mathbf{R}$, $x \in \overline{D}$. Assume that $\xi^n = u^{n-1}(X_S^n + K_S^{n,\overline{D}})$, where $X^n + K^{n,\overline{D}}$ is a two-dimensional diffusion process with reflection with values in $\overline{D}$ and starting from x and*

$$f_n(t, y) = \begin{cases} c_n(t)(y - u_0(X_t^n + K_t^{n,\overline{D}})), & t \geq S, \\ b(t)\Big(g(X_t^n, u_0, x)u_0(X_t^n + K_t^{n,\overline{D}}) - y\Big), & t < S. \end{cases}$$

If $(Y^{x,n}, Z^n)$ *is a solution to the BSDE*

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

then

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

where

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

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

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

According to the above theorem, the reconstructed pixel Y_0^m is the sum of the pixel values from the neighbourhood multiplied by weights a_k and some measure of similarity g. The function f_n is chosen so that the coefficient values have the following properties:

1. $\sum_{k=0}^{m-1} a_k = 1,$
2. $a_k > 0, \;\; k = 0, 1, ..., j,$
3. $a_k < 0, \;\; k = j+1, j+2, ..., m-1,$
4. $\sum_{k=j}^{m-1} a_k > 0.$

The above properties mean that for times $k = 0, 1..., j$, we want to achieve a smoothing effect in the direction of the edges, and for times $k = j+1, j+2, ..., m-1$, we wish to accomplish a sharpening impact in the direction of the gradient vector. In the case of reconstructing the image background, we relinquish the sharpening effect, and the last weight takes the value $\sum_{k=j}^{m-1} a_k$.

Monte Carlo method

The reconstructed pixel can be approximated by the following formula

$$\begin{aligned} u^n(x) \approx Y_0^m &= \sum_{k=0}^{j-1} a_k \mathbf{E}\left[g(X_{t_k}^n + K_{t_k}^{n,\overline{D}}, u_0, x) u_0(X_{t_k}^n + K_{t_k}^{n,\overline{D}})\right] + \sum_{k=j}^{m-1} a_k \mathbf{E}\left[u_0(X_{t_k}^n + K_{t_k}^{n,\overline{D}})\right] \\ &= \mathbf{E}\Big[\sum_{k=0}^{j-1} a_k g(X_{t_k}^n + K_{t_k}^{n,\overline{D}}, u_0, x) u_0(X_{t_k}^n + K_{t_k}^{n,\overline{D}}) + \sum_{k=j}^{m-1} a_k u_0(X_{t_k}^n + K_{t_k}^{n,\overline{D}})\Big]. \end{aligned}$$

The expected value $\mathbf{E}$ is approximated using the Monte Carlo method as follows

$$u^n(x) \approx \frac{1}{N} \sum_{v=1}^{N} \Big[\sum_{k=0}^{j-1} a_k g(X_{t_k}^n(\omega_v), u_0, x) u_0(X_{t_k}^n(\omega_v)) + \sum_{k=j}^{m-1} a_k u_0(X_{t_k}^n(\omega_v))\Big],$$

where $X^n(\omega)$ is trajectory of the process $X^n + K^{n,\overline{D}}$. To get values of this trajectory, we use the classical Euler scheme [53] and the modified diffusion method [6, 8, 9, 12].

Euler approximation

The classical Euler scheme in the gradient direction, we can write as Algorithm 1, where $\mathcal{N}(0,1)$ is generator of the normal distribution and ∇ is Di Zenzo RGB gradient [58].

Algorithm 1: Euler scheme in the gradient direction

```
input  : $x$ – pixel position
output: $(X_k)_{k=0,1\ldots j}$ – trajectory in the gradient direction
$X_0 = x$
foreach $k = 1, 2, \ldots j$ do
    $X_k = X_{k-1} + dt\nabla(G_\gamma * u_0)(X_{k-1})\mathcal{N}(0,1)$
```

To determine trajectories in the direction perpendicular to the gradient - along the edges, we can also use the Euler scheme. However, this scheme yields good results for small time increments dt, which, when combined with the Monte Carlo method, disqualifies this approach due to the algorithm's lengthy runtime. In [8, 9], it is shown that this simulation can be accelerated (almost 50 - fold) by using a modified diffusion scheme.

Modified diffusion

The modified diffusion scheme relies on checking the difference in values between subsequent steps (the theta condition) before determining the next step in the trajectory. Failure to satisfy this condition means we remain in the same place, while fulfilling it allows us to take a step forward with a relatively large value of dt.

In references [6] and [8], two extreme approaches to this simulation were proposed, with a focus on simulation time. In [6], a predetermined number of attempts is performed to determine new trajectory values; in the worst-case scenario, the trajectory may remain stationary. In [8], the scheme was improved by enforcing a predetermined number of distinct trajectory steps. However, in the worst-case scenario, this latter approach may run for an extended period due to the lack of control over the number of theta conditions.

For segmentation, we propose a compromise between these solutions to limit the number of false theta tests. In the new numerical scheme, we require that the number of false theta tests not exceed a predetermined value.

Modified diffusion for image segmentation

For denoising, Borkowski [12] proposed a modified diffusion scheme (perpendicular to the gradient direction) in the form

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

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

where by Θ (the theta condition) we mean the condition

$$|(G_\gamma * u^{n-1})(H_{t_k}^n(\omega)) - (G_\gamma * u^{n-1})(X_{t_{k-1}}^n(\omega))| \leq p$$

and

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

A terminal time τ_j provides that the numerical simulation of the diffusion trajectory gives at least j values of $X^n(\omega)$ which differ from the value in the previous step. For this scheme and the large variability of values in the image, there is a risk that the Θ condition will not be met too often. In the case of segmentation, we consider the possibility of noising the image with Gaussian noise with a standard deviation significantly exceeding the value of 100. Therefore, to speed up the trajectory simulation, we allow only a fixed number of times the Θ condition is false, i.e.

$$\tau_j = \min\{k; (\Theta \text{ is true } j \text{ times}) \text{ or } (\Theta \text{ is false } j \text{ times})\}. \tag{1}$$

In pseudocode terms, we write this as Algorithm 2.

Algorithm 2: Modified diffusion for image segmentation

```
input  : $x$ – pixel position
output: $(X_k)_{k=0,1...j}$– trajectory along edges
$X_0 = x$
$Counter = 0$
foreach  $k = 1, 2, \dots j$ do
    $X_k = X_{k-1} + dt \begin{bmatrix} \mathcal{N}(0,1) \\ \mathcal{N}(0,1) \end{bmatrix}$
    if $|(G_\gamma * u^{n-1})(X_k) - (G_\gamma * u^{n-1})(X_{k-1})| > p$ then
        $k = k - 1$
        $Counter = Counter + 1$
        if $Counter \geq j$ then
            $k = j + 1$
```

4 Algorithm

Article [12] presents the details of implementing the BSDE denoising algorithm, including parameter values and the definition of functions. In the paper [12], we will also find an analysis and justification of the chosen parameter values. After considering the changes for segmentation purposes, we obtain an Algorithm 3, whose outcome depends on the parameters $\sigma > 0$ and $c \in [0, 1]$. The parameter σ represents the standard deviation of Gaussian noise, while the parameter c enhances edge effects (the higher the value, the sharper the image). The algorithm is divided into two parts: we distinguish between cases when a pixel belongs to the edge ($g \equiv 0$) or background ($c \equiv 0$). In the case of the background, we do not use the sharpening effect. We also use the patchwise implementation to reduce the computation of the same weights for neighbourhood pixels.

The segmentation algorithm involves the iterative application of a denoising algorithm. Thanks to parameters c_n, we can avoid the blurring effect on the image and preserve a clear boundary between image regions. For images with high noise levels, the value of the parameter c_n should also be higher to ensure that the sharpening effect is noticeable. The BSDE segmentation algorithm is presented in Algorithm 4. The algorithm has a single parameter μ, and its value depends on the noise in the image. The parameter value must also be high for high noise levels. Based on this parameter, we determine the values of σ_n and c_n for the denoising algorithm. In each step, these values decrease because subsequent images exhibit lower noise. In the case of grayscale images, we convert images to RGB by assigning the same value to each coordinate. In practice, the algorithm works in parallel using threads. Since we denoise pixels independently, each thread deals with its part of the image domain. The only thing to remember is to set a synchronisation barrier for threads in the main loop of the segmentation algorithm.

To provide a more accurate representation of the final result, we introduce Algorithm 5, in which we perform binarization using Otsu's method [44]. For RGB images, this operation is conducted individually for each component: red $(u.r)$, green $(u.g)$ and blue $(u.b)$.

5 Parameters

The impact of the μ parameter on the final result is to separate the image background from significant shapes. Experiments have shown that the range of the μ parameter is a good choice from 40 to 100. We examine the convergence of the proposed algorithm for different values of μ, i.e. the existence of the limit

$$u(x) = \lim_{n \to +\infty} u^n(x).$$

In each step, we check the Cauchy condition for this sequence and calculate the behaviour of the mean square error of the last two elements of the sequence i. e.

$$\|u^n - u^{n-1}\|_{\text{MSE}} = \frac{1}{|\overline{D} \cap \mathbf{Z}^2|} \sum_{x \in \overline{D} \cap \mathbf{Z}^2} (u^n(x) - u^{n-1}(x))^2.$$

Figure 1 shows how the error value changes in subsequent algorithm iterations. The results were obtained for different values of the μ parameter and input images, some of which were also noisy with Gaussian noise. Analysis of the graph shows that convergence is achieved after four iterations. We propose to use five as the value of epsilon $\varepsilon = 5$, which, along with the fact that $\mu > 40$, guarantees that the algorithm will execute at least four iterations.

The initial value of μ determines the subsequent values of the parameters for the denoising algorithm. The proposed subsequent values of c_n and σ_n are chosen to use the entire possible range during denoising and create a sequence of decreasing values, i.e. $c_0 > c_1 > \ldots c_{n-1} > c_n \in (0, 1)$ and

Algorithm 3: BSDE denoising for image segmentation

```
input  : $u_0$ – noisy image, $\sigma$ – standard deviation of the noise, $c \geq 0$ – enhancing parameter
output: $u$ – reconstructed image
foreach pixel position $x$ do
    if $c > 0$ and $|\nabla(G_\gamma * u_0)(x)| > \sigma$ then                              /* Edge denoising */
        $X_0 = x$
        foreach $n = 1, 2, \dots N$ do                                                  /* Monte Carlo method */
            $u(x) = u(x) + a_0 \cdot u_0(x)$
            $Counter = 0$
            foreach $k = 1, 2, \dots j$ do                         /* Modified diffusion for image segmentation*/
                $X_k = X_{k-1} + dt \begin{bmatrix} \mathcal{N}(0,1) \\ \mathcal{N}(0,1) \end{bmatrix}$
                if $|(G_\gamma * u_0)(X_k) - (G_\gamma * u_0)(X_{k-1})| > p$ then        /* The condition cannot be */
                    $k = k - 1$                                                        /* false more than j times */
                    $Counter = Counter + 1$
                    if $Counter \geq j$ then
                        $X_j = X_{k-1}$
                        $k = j + 1$
                else
                    $u(x) = u(x) + a_k \cdot u_0(X_k)$                                              /* $a_k > 0$ */
            foreach $k = j + 1, j + 2, \dots m - 1$ do                         /* Sharpening in gradient direction */
                $X_k = X_{k-1} + dt\nabla(G_\gamma * u_0)(X_{k-1})\mathcal{N}(0,1)$          /* $\nabla$ – DiZenzo RGB gradient */
                $u(x) = u(x) + a_k \cdot u_0(X_k)$                                                  /* $a_k < 0$ */
    else                                                                                  /* Background denoising */
        $X_0 = x$
        foreach $n = 1, 2, \dots N$ do                                                  /* Monte Carlo method */
            $u(x) = u(x) + a_0 \cdot u_0(x)$
            $Counter = 0$
            foreach $k = 1, 2, \dots j$ do                         /* Modified diffusion for image segmentation*/
                $X_k = X_{k-1} + dt \begin{bmatrix} \mathcal{N}(0,1) \\ \mathcal{N}(0,1) \end{bmatrix}$
                if $|(G_\gamma * u_0)(X_k) - (G_\gamma * u_0)(X_{k-1})| > p$ then        /* The condition cannot be */
                    $k = k - 1$                                                        /* false more than j times */
                    $Counter = Counter + 1$
                    if $Counter \geq j$ then
                        $X_j = X_{k-1}$
                        $k = j + 1$
                else
                    if $k == j$ then
                        $a = a_k$
                        foreach $i = j + 1, j + 2, \dots m - 1$ do                                /* No sharpening */
                            $a = a + a_i$
                        $u(x) = u(x) + a \cdot u_0(X_k)$                               /* $\sum_{i=j}^{m-1} a_i = a > 0$ */
                    else                                                                /* Patchwise implementation */
                        foreach $xx$ such that $|x - xx| \leq radius$ do
                            $u(x + xx) = u(x + xx) + a_k g(X_k, u_0, x) \cdot u_0(X_k + xx)$        /* $a_k > 0$ */
```

Algorithm 4: BSDE segmentation

```
input  : $u_0$ – input RGB image, $40 < \mu < 100$ – noise parameter
output: $u$ – segmented image
$u = u_0$
$\sigma = \mu$
foreach $\sigma > \varepsilon$ do
    $c = 0.01\sigma$                      /* $c_n$ */
    $u$ = Algorithm 3 $(u,\sigma,c)$       /* $u^n$ */
    $\sigma = \frac{\sigma}{2}$                    /* $\sigma_n$ */
```

Algorithm 5: BSDE binarization

```
input  : $u_0$ – input RGB image, $40 < \mu < 100$ – noise parameter
output: $v$ – binarized image
$u$ = Algorithm 4 $(u_0,\mu)$
$v.r$ = Otsu$(u.r)$
$v.g$ = Otsu$(u.g)$
$v.b$ = Otsu$(u.b)$
```

$\sigma_0 > \sigma_1 > \ldots \sigma_{n-1} > \sigma_n \in (0, 100)$. However, our goal is to propose an algorithm free from input parameters. Therefore, we propose to make the value of the parameter μ dependent on the variance of the input image and assume:

$$\mu = \max\left\{50, \sqrt{\operatorname{var}(u_0)}\right\}. \tag{2}$$

In the algorithm, we can pass a value for μ. If the given value exceeds the range $(40, 100)$, then the value of this parameter is determined automatically, and we initialise it with the standard deviation of the input image using the formula (2). Figure 2 shows three different segmentation results: a value μ that is too small, optimal, and too large. The proposed formula for automatically selecting this parameter usually returns a value that is too large compared to the optimal value. We believe selecting too large a value is a lesser evil than if we had provided too small a value. We want to avoid leaving background fragments (noise) in the image. This is also why we give up the range from 40 to 50 in the proposed formula (2).

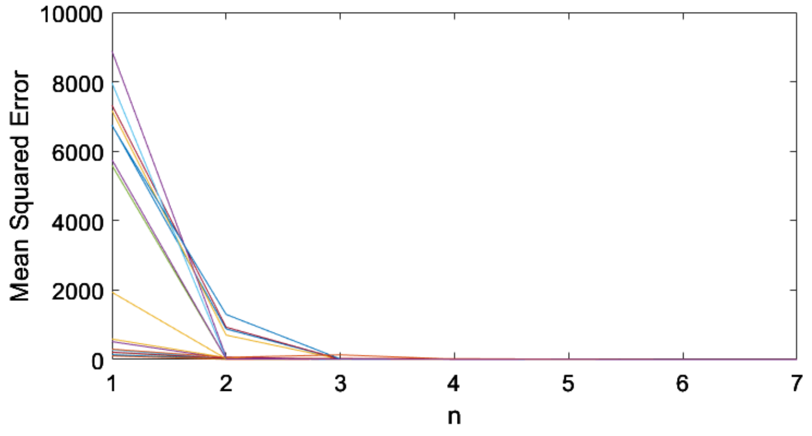


Figure 1: Mean squared error $\|u^n - u^{n-1}\|_{\text{MSE}}$ for different values of the parameter μ and input test images.

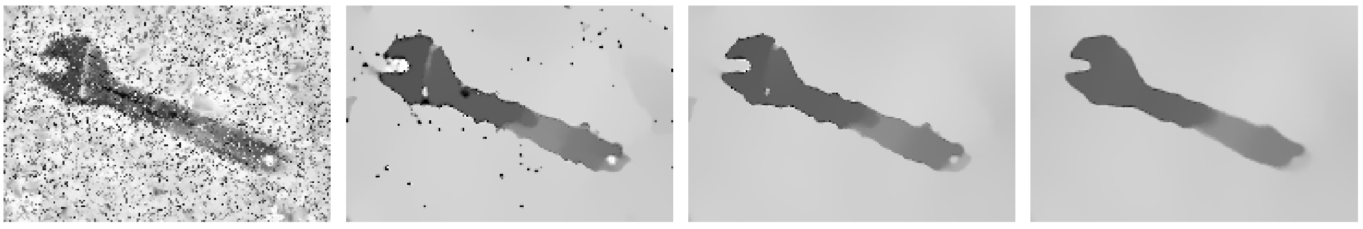


Figure 2: BSDE segmentation results of Wrench image for various values of μ: (b), (c) – manually selected, (d) – formula (2)

Table 1: Segmentation time (in seconds) using the BSDE method for images presented in Figure 3 and Figure 4. The tests were conducted on a CPU Intel(R) Core(TM) i9-9900K and eight threads.

noisy Shamrock 237×278	Coin 182×157	noisy Coin 182×157	Cow 300×200	Bear 255×200	noisy Bear 255×200	noisy Rose Geranium 300×285
11.5	4.8	7.2	9.0	8.4	10.5	14.9

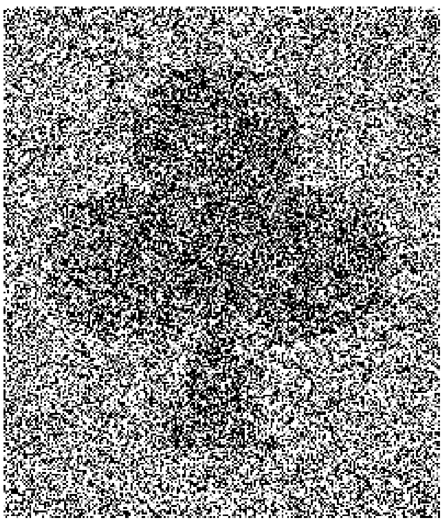
6 Experimental results

This section presents the image segmentation results for the Shamrock, Coin, Cow, Bear and Rose Geranium images. The Cow and Shamrock images are grayscale images, and the Cow, Bear and Rose Geranium images are RGB images. The segmentation algorithm works for RGB images, and in the case of grayscale images, we convert to the same values at each coordinate. The Otsu binarization is performed for each RGB component separately, resulting in eight possible colour values. In the case of four examples, Figure 3 (a), Figure 3 (g) and Figure 4 (g), Figure 4 (j), we added Gaussian noise with a significant standard deviation value. In all presented examples, we use formula (2) to initialise the algorithm - the program is free from parameters. The analysis of examples shows how well the segmentation algorithm works with noisy data. Table 1 shows the runtimes of the parallel version of the algorithm using eight threads. A significant difference in runtime can be seen for noisy and noise-free images. The difference in the number of tests for the modified diffusion condition (1) causes this. The number of false tests increases for pictures with large variability of values.

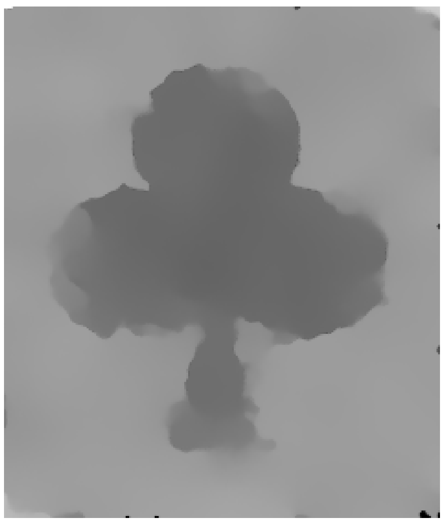
Figure 5 and Figure 6 show the proposed method's use for medical image segmentation from the HAM10000 dataset [54]. We used the algorithm for skin lesion segmentation. The algorithm returns the shape of this lesion as a black-and-white image. After binarisation using our method, the segment located in the central part of the image is displayed as the result. Since the analysed images do not contain noise (only the natural skin texture), our algorithm's value of the μ parameter is set to the lowest value $\mu = 40$ for all medical images.

7 Conclusions

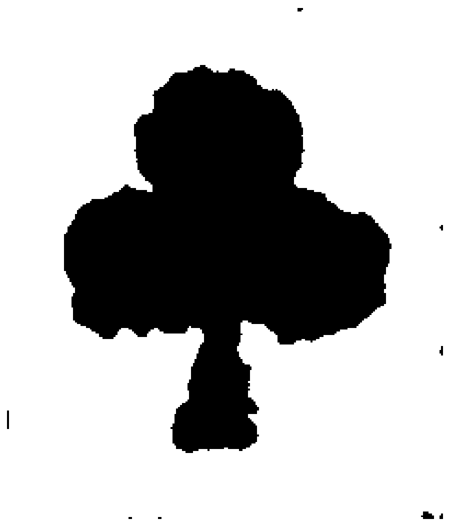
In this work, we introduced a segmentation algorithm that showcases interesting segmentation results for noisy images. The subject presented here is an intriguing area for further research. Specifically, it would be interesting to develop a mathematical model for the solution generated by the segmentation algorithm.



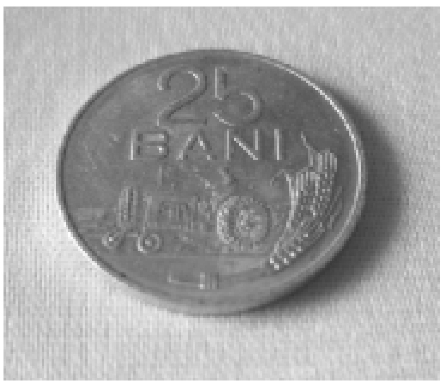
(a) Input image



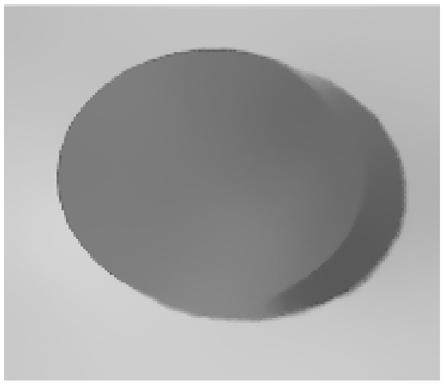
(b) BSDE segmentation for $\mu = 99$



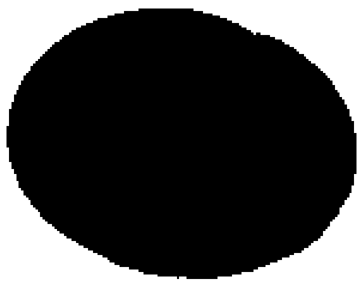
(c) BSDE binarization for $\mu = 99$



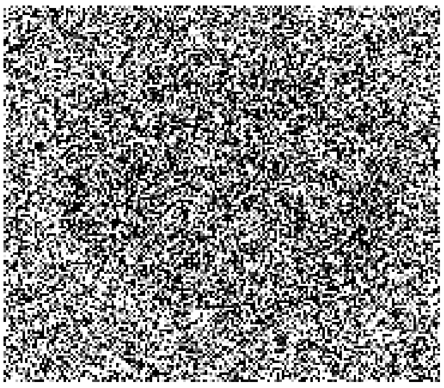
(d) Input image



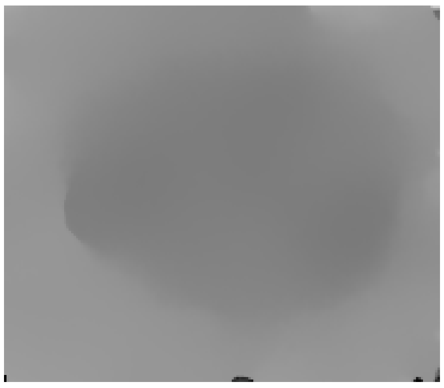
(e) BSDE segmentation for $\mu = 50$



(f) BSDE binarization for $\mu = 50$



(g) Input image

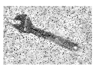


(h) BSDE segmentation for $\mu = 99$

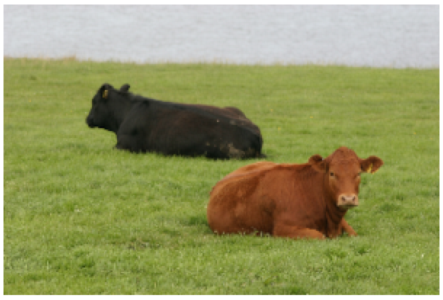
(i) BSDE binarization for $\mu = 99$

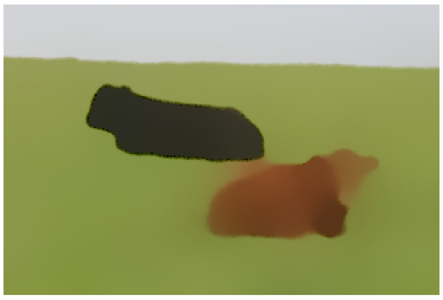
Figure 3: Segmentation and binarization of grayscale images (for each row): Shamrock and Coin. The input images (a) and (g) are corrupted by Gaussian noise. The Shamrock image is corrupted with a standard deviation equal to 500, and the Coin image with a standard deviation equal to 350.

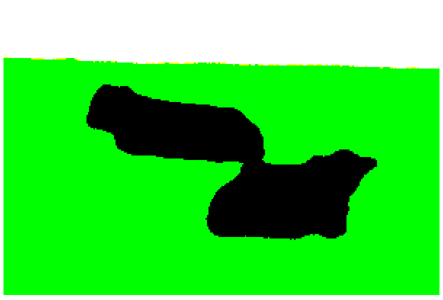
Image Credits

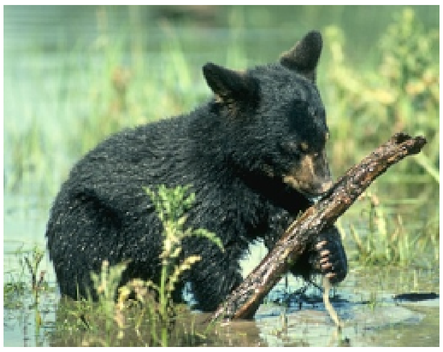
 Extracted from [23]

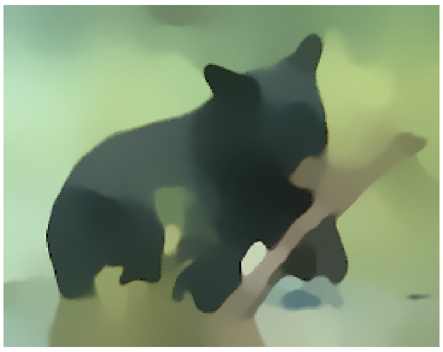
Extracted from [48]

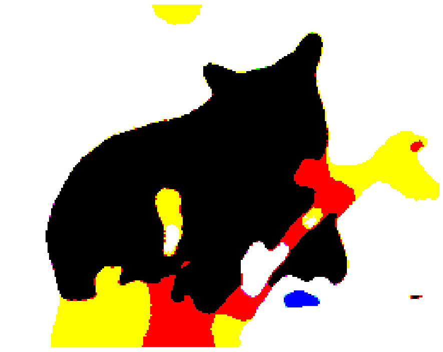

(a) Input image

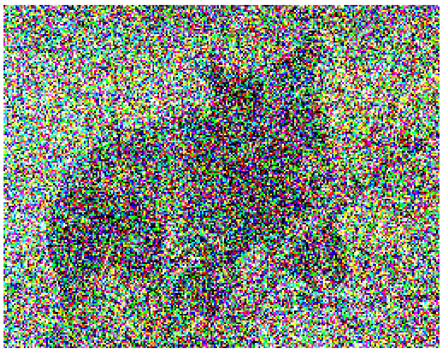

(b) BSDE segmentation for $\mu = 56.85$

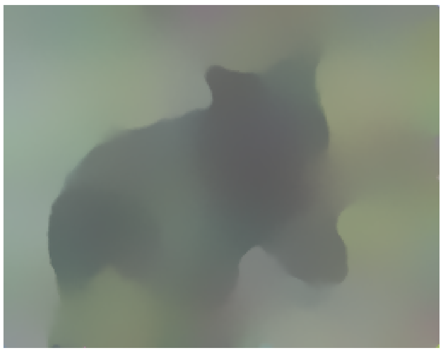

(c) BSDE binarization for $\mu = 56.85$

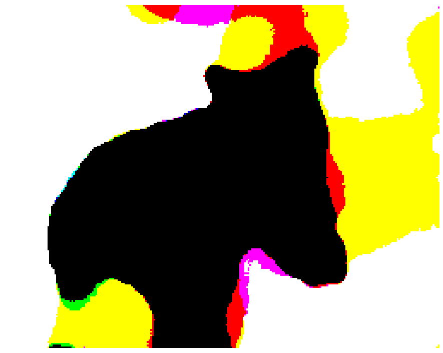

(d) Input image

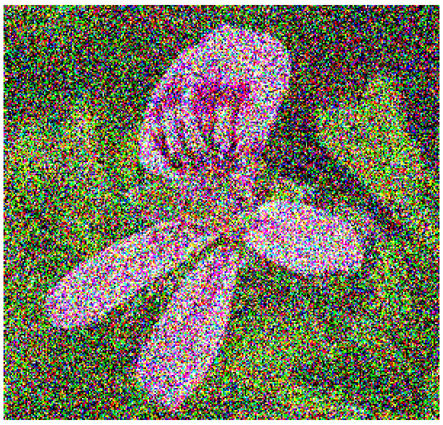

(e) BSDE segmentation for $\mu = 60.89$

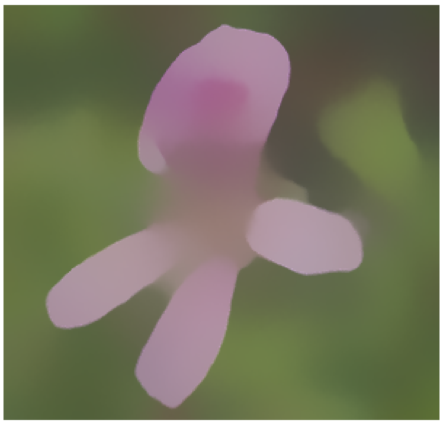

(f) BSDE binarization for $\mu = 60.89$


(g) Input image


(h) BSDE segmentation for $\mu = 99$


(i) BSDE binarization for $\mu = 99$


(j) Input image


(k) BSDE segmentation for $\mu = 98.07$

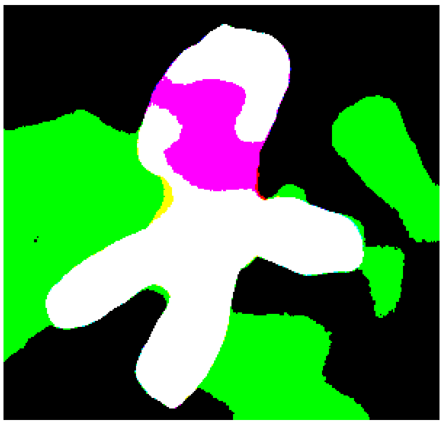

(l) BSDE binarization for $\mu = 98.07$

Figure 4: Segmentation and binarization of RGB images (for each row): Cow, Bear and Rose Geranium. The input images (g) and (j) are corrupted by Gaussian noise (independently added to all coordinates). The Bear image is corrupted with a standard deviation equal to 250, and the Rose Geranium image with a standard deviation equal to 150.

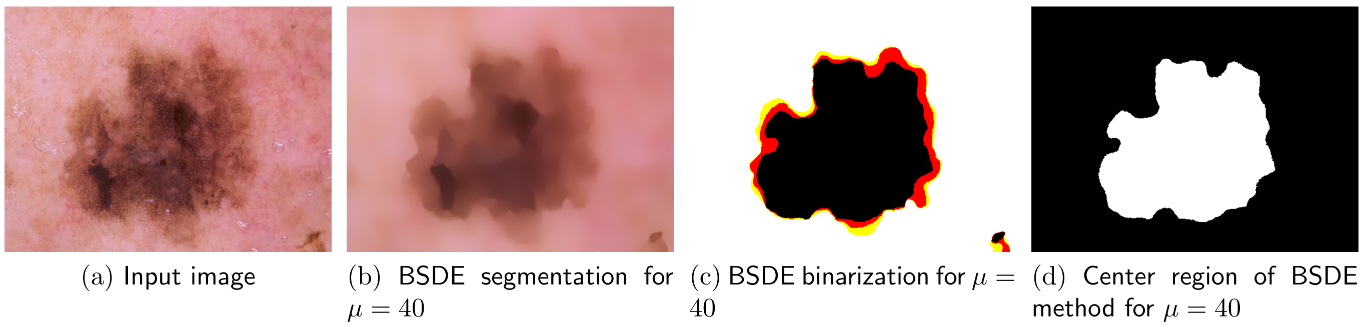

(a) Input image (b) BSDE segmentation for $\mu = 40$ (c) BSDE binarization for $\mu = 40$ (d) Center region of BSDE method for $\mu = 40$

Figure 5: Segmentation, binarization and the centre region of BSDE method. The input image shows a skin lesion, and the algorithm's task is to detect this shape.

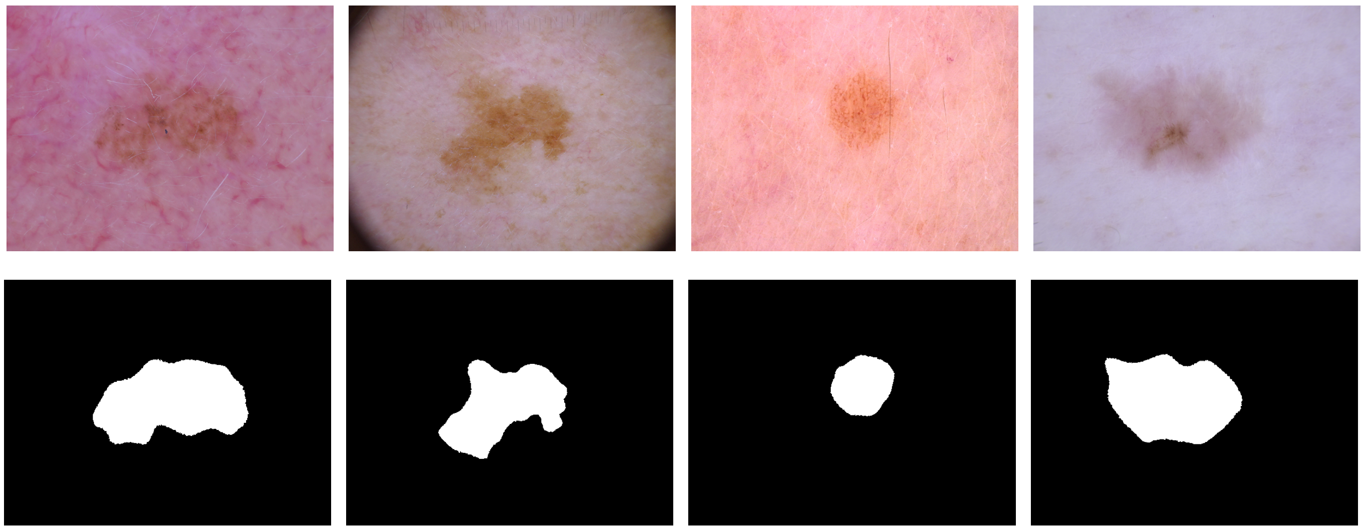
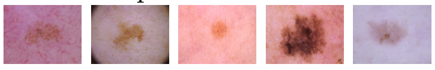


Figure 6: Central regions from the BSDE segmentation method. Each column represents a skin lesion (input image) and the detected shape (center region of BSDE method for $\mu = 40$) .

Extracted from [40]

BSDS300 [39]

Wikipedia

 Harvard Dataverse [54]

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), pp. 184–205.

[2] R. Adams and L. Bischof, *Seeded region growing*, IEEE Transactions on pattern analysis and machine intelligence, 16 (1994), pp. 641–647.

[3] P. Akbarimoghaddam, A. Ziaei, and H. Azarnoush, *Deep active contours using locally controlled distance vector flow*, Signal, Image and Video Processing, 16 (2022), pp. 1773–1781.

[4] G. Aubert, M. Barlaud, O. Faugeras, and S. Jehan-Besson, *Image segmentation using active contours: Calculus of variations or shape gradients?*, SIAM Journal on Applied Mathematics, 63 (2003), pp. 2128–2154.

[5] D. Borkowski, *Chromaticity denoising using solution to the Skorokhod problem*, in Image Processing Based on Partial Differential Equations, Springer, 2007, pp. 149–161.

[6] ——, *Modified diffusion to image denoising*, in Computer Recognition Systems 2, Springer, 2007, pp. 92–99.

[7] ——, *Smoothing, enhancing filters in terms of backward stochastic differential equations*, in International Conference on Computer Vision and Graphics, Springer, 2010, pp. 233–240.

[8] ——, *Euler's approximations to image reconstruction*, in International Conference on Computer Vision and Graphics, Springer, 2012, pp. 30–37.

[9] ——, *Stochastic approximation to reconstruction of vector-valued images*, in Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, Springer, 2013, pp. 393–402.

[10] ——, *Forward and backward filtering based on backward stochastic differential equations*, Inverse Problems & Imaging, 10 (2016), pp. 305–325.

[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.

[12] ——, *Implementation of image denoising based on backward stochastic differential equations*, Image Processing On Line, 13 (2023), pp. 321–349.

[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.

[14] ——, *Image denoising using backward stochastic differential equations*, in Man-Machine Interactions 5: 5th International Conference on Man-Machine Interactions, ICMMI 2017 Held at Kraków, Poland, October 3-6, 2017, Springer, 2018, pp. 185–194.

[15] J. Canny, *A computational approach to edge detection*, IEEE Transactions on pattern analysis and machine intelligence, 6 (1986), pp. 679–698.

[16] T. F. Chan and L. A. Vese, *Active contours without edges*, IEEE Transactions on image processing, 10 (2001), pp. 266–277.

[17] X. Chen, B. M. Williams, S. R. Vallabhaneni, G. Czanner, R. Williams, and Y. Zheng, *Learning active contour models for medical image segmentation*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 11632–11640.

[18] D. Cheng, R. Liao, S. Fidler, and R. Urtasun, *Darnet: Deep active ray network for building segmentation*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 7431–7439.

[19] Y. Cheng, *Mean shift, mode seeking, and clustering*, IEEE transactions on pattern analysis and machine intelligence, 17 (1995), pp. 790–799.

[20] D. Comaniciu and P. Meer, *Mean shift: Arobust approach toward feature space analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 24 (2002), pp. 603–619.

[21] D. Duffie and L. G. Epstein, *Asset pricing with stochastic differential utility*, The Review of Financial Studies, 5 (1992), pp. 411–436.

[22] ——, *Stochastic differential utility*, Econometrica: Journal of the Econometric Society, (1992), pp. 353–394.

[23] P. Getreuer, *Chan-vese segmentation*, Image Processing On Line, 2 (2012), pp. 214–224.

[24] S. Gur, L. Wolf, L. Golgher, and P. Blinder, *Unsupervised microvascular image segmentation using an active contours mimicking neural network*, in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 10722–10731.

[25] M. Haft-Javaherian, L. Fang, V. Muse, C. B. Schaffer, N. Nishimura, and M. R. Sabuncu, *Deep convolutional neural networks for segmenting 3d in vivo multiphoton images of vasculature in alzheimer disease mouse models*, PloS one, 14 (2019), p. e0213539.

[26] S. Hamadene and J. P. Lepeltier, *Zero-sum stochastic differential games and backward equations*, Systems & Control Letters, 24 (1995), pp. 259–263.

[27] R. M. Haralick and L. G. Shapiro, *Image segmentation techniques*, Computer vision, graphics, and image processing, 29 (1985), pp. 100–132.

[28] K. He, G. Gkioxari, P. Dollár, and R. Girshick, *Mask r-cnn*, in Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969.

[29] A. Hoogi, A. Subramaniam, R. Veerapaneni, and D. L. Rubin, *Adaptive estimation of active contour parameters using convolutional neural networks and texture analysis*, IEEE transactions on medical imaging, 36 (2016), pp. 781–791.

[30] P. Hu, B. Shuai, J. Liu, and G. Wang, *Deep level sets for salient object detection*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2300–2309.

[31] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, *Densely connected convolutional networks*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.

[32] N. E. Karoui, S. Peng, and M. C. Quenez, *Backward stochastic differential equations in finance*, Mathematical Finance, 7 (1997), pp. 1–71.

[33] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contour models*, International journal of computer vision, 1 (1988), pp. 321–331.

[34] T. H. N. Le, K. G. Quach, K. Luu, C. N. Duong, and M. Savvides, *Reformulating level sets as deep recurrent neural network approach to semantic segmentation*, IEEE Transactions on Image Processing, 27 (2018), pp. 2393–2407.

[35] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, *Entropy rate superpixel segmentation*, in CVPR 2011, IEEE, 2011, pp. 2097–2104.

[36] S. Luo, X.-C. Tai, L. Huo, Y. Wang, and R. Glowinski, *Convex shape prior for multi-object segmentation using a single level set function*, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 613–621.

[37] R. Malladi, J. A. Sethian, and B. C. Vemuri, *Shape modeling with front propagation: A level set approach*, IEEE transactions on pattern analysis and machine intelligence, 17 (1995), pp. 158–175.

[38] D. Marcos, D. Tuia, B. Kellenberger, L. Zhang, M. Bai, R. Liao, and R. Urtasun, *Learning deep structured active contours end-to-end*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8877–8885.

[39] D. Martin, C. Fowlkes, D. Tal, and J. Malik, *A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics*, in Proceedings eighth IEEE international conference on computer vision. ICCV 2001, vol. 2, IEEE, 2001, pp. 416–423.

[40] Z. Meng, E. Merkurjev, A. Koniges, and A. L. Bertozzi, *Hyperspectral image classification using graph clustering methods*, Image Processing On Line, 7 (2017), pp. 218–245.

[41] A. P. Moore, S. J. Prince, J. Warrell, U. Mohammed, and G. Jones, *Superpixel lattices*, in 2008 IEEE conference on computer vision and pattern recognition, IEEE, 2008, pp. 1–8.

[42] D. B. Mumford and J. Shah, *Optimal approximations by piecewise smooth functions and associated variational problems*, Communications on pure and applied mathematics, (1989).

[43] S. Osher and J. A. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations*, Journal of computational physics, 79 (1988), pp. 12–49.

[44] N. Otsu, *A threshold selection method from gray-level histograms*, IEEE transactions on systems, man, and cybernetics, 9 (1979), pp. 62–66.

[45] E. Pardoux and S. Peng, *Adapted solution of a backward stochastic differential equation*, Systems & control letters, 14 (1990), pp. 55–61.

[46] ——, *Backward stochastic differential equations and quasilinear parabolic partial differential equations*, in Stochastic Partial Differential Equations and Their Applications: Proceedings of IFIP WG 7/1 International Conference University of North Carolina at Charlotte, NC June 6–8, 1991, Springer, 2005, pp. 200–217.

[47] S. Peng, W. Jiang, H. Pi, X. Li, H. Bao, and X. Zhou, *Deep snake for real-time instance segmentation*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 8533–8542.

[48] F. Pierre, M. Amendola, C. Bigeard, T. Ruel, and P.-F. Villard, *Segmentation with active contours*, Image Processing On Line, 11 (2021), pp. 120–141.

[49] H. Qin, J. M. Zain, X. Ma, and T. Hai, *Scene segmentation based on seeded region growing for foreground detection*, in 2010 Sixth International Conference on Natural Computation, vol. 7, IEEE, 2010, pp. 3619–3623.

[50] M. F. Ramli and K. N. Tahar, *Homogeneous tree height derivation from tree crown delineation using seeded region growing (srg) segmentation*, Geo-Spatial Information Science, 23 (2020), pp. 195–208.

[51] W. Rong, Z. Li, W. Zhang, and L. Sun, *An improved canny edge detection algorithm*, in 2014 IEEE international conference on mechatronics and automation, IEEE, 2014, pp. 577–582.

[52] J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Transactions on pattern analysis and machine intelligence, 22 (2000), pp. 888–905.

[53] L. Słomiński, *Euler's approximations of solutions of sdes with reflecting boundary*, Stochastic Processes and their Applications, 94 (2001), pp. 317–337.

[54] P. Tschandl, C. Rosendahl, and H. Kittler, *The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions*, Scientific data, 5 (2018), pp. 1–9.

[55] L. A. Vese and T. F. Chan, *A multiphase level set framework for image segmentation using the mumford and shah model*, International journal of computer vision, 50 (2002), pp. 271–293.

[56] Z. Wang, D. Acuna, H. Ling, A. Kar, and S. Fidler, *Object instance annotation with deep extreme level set evolution*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7500–7508.

[57] S. Yan, X.-C. Tai, J. Liu, and H.-Y. Huang, *Convexity shape prior for level set-based image segmentation method*, IEEE Transactions on Image Processing, 29 (2020), pp. 7141–7152.

[58] S. D. Zenzo, *A note on the gradient of a multi-image*, Computer vision, Graphics, and Image Processing, 33 (1986), pp. 116–125.