The Reverse-Transition Weighting Filter for Effective Edge Detection for Noisy Color Images

1st Seth T. Richey
Department of Mathematics and Statistics
Mississippi State University
Mississippi State, MS 39762 USA
str163@msstate.edu

2nd Luke J. Hand
Department of Mechanical Engineering
Mississippi State University
Mississippi State, MS 39762 USA
ljh307@msstate.edu

3rd Owen C. Burton
Department of Mechanical Engineering
Mississippi State University
Mississippi State, MS 39762 USA
ocb64@msstate.edu

4th J. Brittin Perdue
Department of Electrical Engineering
Mississippi State University
Mississippi State, MS 39762 USA
jbp448@msstate.edu

5th Penelope Prochnow
Data Science
Mississippi State University
Mississippi State, MS 39762 USA
pp872@msstate.edu

6th Jason Shin
Department of Supply Chain Logistics
Mississippi State University
Mississippi State, MS 39762 USA
jshin@business.msstate.edu

7th Seongjai Kim
Department of Mathematics and Statistics
Mississippi State University
Mississippi State, MS 39762 USA
skim@math.msstate.edu. *Contact Author)*

Full/Regular Research Paper; CSCI-RTPC

***Abstract*—Edges in a digital image provide important information about the objects in the image since they constitute boundaries between objects. Most edge detection algorithms are sensitive to noise and attempts to remove noise often weaken not only noise but also the edge strength. This article proposes an innovative denoising method called the *reverse-transition weighting* (RTW) filter, which can suppress noise without weakening the edge strength. Such a property of preserving the edge strength is *unique* to the RTW denoising filter. The new filter is analyzed for its stability and convergence and adopted for the denoising step of the Canny edge detection algorithm, replacing the conventional Gaussian smoothing filter. We also compare gradient-fusion methods which combine the RGB gradients into one. Our goal is to formulate a robust edge detection algorithm for color images, particularly for heavily noisy images. Various examples are given to show the effectiveness of the new denoising filter.**

***Index Terms*—Reverse-transition weighting (RTW) filter, Canny edge detection, denoising, structure tensor**

I. Introduction

Edge detection includes various mathematical methods that aim to identify edges, defined as curves in a digital image at which the image color and brightness change sharply or have discontinuities.

Edge detection methods can be grouped into two categories, *search-based* and *zero-crossing based*. The search-based methods detect edges by first computing a measure of edge strength (such as the gradient magnitude) and then searching for local directional maxima of the gradient magnitude (the gradient direction) [1], [2]. The zero-crossing based methods search for zero crossings in a second-order derivative expression computed from the image (such as the Laplacian) in order to find edges [3]. As a pre-processing step to edge detection, a smoothing stage is usually applied to suppress noise.

In general, differentiation-based edge detection algorithms are sensitive to noise. In order to suppress noise in the image and to improve the performance of edge detection for noisy images, one can employ some spatial smoothing methods such as the Gaussian filter [1], fuzzy filters [4], [5], and anisotropic diffusion operators [6], [7]. For most denoising methods, the main goal is to reduce noise without deteriorating the edge strength.

The results of edge detection methods may vary widely depending on the hysteresis thresholds. An effective edge detection algorithm must incorporate a satisfactory approach for the selection of the thresholds [8]. Additionally for edge detection of color images, a good strategy must be found in order to combine gradient magnitudes (or three separate edges obtained from the image channels) to one magnitude.

In this article, we will introduce an innovative *iterative* denoising algorithm called the *reverse-transition weighting* (RTW) filter, which tries to flatten local extrema in the image, minimizes diffusion in the edge normal direction to preserve the edge strength, and diffuses image values actively in the edge tangential direction to form clean edges. The RTW filter is analyzed for its stability and convergence and adopted for the denoising step of the Canny algorithm [1] for edge detection for noisy color images, replacing the conventional Gaussian smoothing filter.

The article is organized as follows. Section II reviews briefly the Canny edge detection algorithm, strategies for handling color images and noisy images in edge detection, and the structure tensor method. In Section III, the RTW filter is described in detail and analyzed for its stability and convergence. Section IV presents numerical examples which show the effectiveness of the new filter. The section also shows a comparison study for gradient-fusion methods. The last section, Section V, summarizes our findings.

II. Preliminaries

This section presents a brief review of the Canny edge detection algorithm, methods for color images and noisy images in edge detection, and the structure tensor method for combing RGB gradients to one. Note that the Canny algorithm was originally developed for gray-scale signals [1] and it has been applied for the detection of edges on images in either gray or color [9]–[12].

A. The Canny edge detection algorithm

The process of the Canny edge detection algorithm for gray-scale images [1] can be broken down to five different steps:

1) *Noise removal*: Apply a Gaussian filter to smooth the image and remove the noise.
2) *Gradient calculation*: Find the magnitude and angle of the image gradient.
3) *Edge thinning*: Apply thresholding or suppression to the gradient magnitude in order to get thin edges.
4) *Double threshold*: Determine potential edges, weak and strong.
5) *Linking by hysteresis*: Finalize the edge detection by transforming some weak pixels into strong pixels.

We will see some details for the five steps.

1) Noise removal: Since the noise in the image easily affects edge detection results, it is necessary to filter out the noise to prevent false detection. In smoothing the image, a common method is the Gaussian filter in which a kernel is convolved with the image. The equation for a Gaussian filter kernel of size $(2k+1)(2k+1)$ is given by

$$G_{ij} = \frac{1}{2\pi\sigma^2} \exp\Big(-\frac{i^2+j^2}{2\sigma^2}\Big), \quad -k \le i, j \le k. \tag{1}$$

The size of the Gaussian kernel affects the performance of the edge detector; the larger the size is, the lower the detector's sensitivity to noise. However, with the increase of the Gaussian filter kernel size, the edge strength will be weakened and the localization error to detect the edge will increase. A 5×5 kernel is a good size for many cases.

2) Gradient calculation: Edges correspond to a change of pixels' intensities. To detect edges, a common way is to apply filters that can highlight the intensity change in both horizontal (x) and vertical (y) directions. For example, the Sobel gradient can be calculated convolving the Sobel kernels

$$K_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad K_y = K_x^T. \tag{2}$$

Given an image u, let $\nabla u = [u_x, u_y] \approx [K_x * u, K_y * u]$. Then the edge normal direction can be computed by

$$\theta = \text{atan2}(u_y, u_x), \quad -\pi \le \theta \le \pi. \tag{3}$$

3) Edge thinning: This stage of the Canny algorithm for the detection of "true edges" can be formulated as the *non-maximum suppression*. The idea is simple: the final image should ideally have thin edges. Thus the algorithm goes through the points on the gradient intensity matrix and finds the pixels having the maximum value in the edge normal direction.

The non-maximum suppression can be implemented effectively. Note that the output of (3), θ, has values between $-\pi$ to π. However, at a point $\mathbf{x}_{ij}$, the available values of the gradient magnitude are at angles in multiples of $\pi/4$.

```
theta(theta<0) = theta(theta<0)+pi;
R = mod(floor((theta+pi/8)/(pi/4)),4);
```

In the above code in Matlab, the out matrix `R` assigns a value from $\{0,1,2,3\}$ to each pixel. If $\mathtt{R(i,j)} = k$, $k = 0,1,2,3$, the edge normal direction (or its opposite direction) at the (i,j)-pixel is nearest to the angle $k\cdot\pi/4$.

4) Double threshold: This step is to apply a threshold in order to decide whether or not edges are present at an image point. The lower the threshold is, the more edges will be detected. The result will be increasingly susceptible to noise and possibly detect edges of irrelevant features in the image. Conversely a high threshold may miss subtle edges or result in fragmented edges.

A good thresholding strategy is to identify three kinds of pixels: strong, weak, and non-relevant. After setting high and low thresholds:

- The high threshold is used to identify the *strong* pixels.
- The low threshold is used to identify the *non-relevant* pixels.
- All pixels having intensity between both thresholds are flagged as *weak*. The hysteresis mechanism (Step 5) will finalize the edge detection by transforming some weak pixels into strong ones.

We need to choose appropriate thresholding parameters; suitable thresholding values may vary over the image. See [8] for an effective strategy for determining the hysteresis thresholds.

5) Linking by hysteresis: The strategy of thresholding and linking in edge detection is based on the assumption that edges are likely to be in continuous curves. This assumption allows us to analyze weak pixels to find a possibility to transform them to strong ones, hysteresis. Here is a common hysteresis rule:

> *Transform a weak pixel into strong ones if and only if at least one of 8 surrounding pixels of the weak pixel is a strong one.*

B. Color images and noisy images

This subsection briefly reviews applications of the Canny algorithm for color images and noisy images.

1) Color images: There have been two ways to detect edges of color images. The first way is to transform the color image into a gray one and process it as a gray image, which is a simple and easy procedure. However, the transformation may weaken the strength of edges, and therefore the resulting algorithm may not be effective. The other method is to apply the gray image edge detection algorithm separately for each of three channels of the color image and fuse the three separate results at a certain stage. With fusion-based edge detection, most literature can be placed into two categories: output fusion methods and multi-dimensional gradient methods [13].

- *Output fusion methods* appear to be the most popular. The goal is to perform edge detection three times, once for each red, green, and blue. Then results are fused by a logic rule to determine the edges. For example, if a point has at least two channels identified as an edge, this location is to be determined as an edge pixel.
- *Multi-dimensional gradient methods* short-circuit the above by combining the three gradients into one and detecting edges only once. For this, one can apply the structure tensor [14], [15] or take the average, maximum, or mean square root of the three gradients.

2) Noisy images: Noisy images with low *signal to noise ratio* (SNRs) are common in various application domains. Examples include electron microscopy (EM) images taken under certain protocols (e.g., cryo-EM), fingerprint images with low tissue contrast, photos acquired under poor lighting, etc.. Edges carry important information since they mark the boundaries of objects in the image. Extracting edges from such images is important to allow proper interpretation of their content. Moreover, the study of edge detection under such extreme conditions may potentially lead to better algorithms for handling photographs of natural scenes [16].

Many edge detection algorithms have been developed based on the gradient magnitude, which in general is sensitive to noise in the image. In order to suppress the noise and to improve the performance of edge detection for noisy images, one can employ some spatial smoothing methods, such as the Gaussian filter, applied for either image values, gradient vectors, or the structure tensor [15].

C. Structure tensor

For a function $f : \mathbb{R}^n \to \mathbb{R}$, the *structure tensor* is a matrix derived from the gradient of $f(\mathbf{x})$: the structure tensor is defined as

$$S(\mathbf{x}) = (\nabla f)(\nabla f)^T(\mathbf{x}), \tag{4}$$

which describes the distribution of the gradient in a neighborhood of the point $\mathbf{x}$. When $\mathbf{x} = (x, y) \in \mathbb{R}^2$, the structure tensor in (4) reads

$$S(\mathbf{x}) = (\nabla f)(\nabla f)^T(\mathbf{x}) = \begin{bmatrix} f_x^2 & f_x f_y \\ f_x f_y & f_y^2 \end{bmatrix}(\mathbf{x}). \tag{5}$$

It is not difficult to prove the following.

1) The matrix S is symmetric and positive semidefinite, i.e., eigenvalues are real and nonnegative.
2) $||S||_2 = ||\nabla f||^2$, which is the maximum eigenvalue of S.
3) $\det S = 0$, which implies that the number 0 is the other eigenvalue of S.
4) The two eigenvalue-eigenvector pairs $(\lambda_i, \mathbf{v}_i)$, $i = 1, 2$, are

$$\begin{array}{ll} \lambda_1 = ||\nabla f||^2, & \mathbf{v}_1 = \nabla f \\ \lambda_2 = 0, & \mathbf{v}_2 = [-f_y,\, f_x]^T \end{array} \tag{6}$$

Now, let $I(\mathbf{x}) = (r, g, b)(\mathbf{x})$ be a color image, where $\mathbf{x}$ is a pixel in the image. Then

$$\nabla I(\mathbf{x}) = \begin{bmatrix} (r, g, b)_x \\ (r, g, b)_y \end{bmatrix}(\mathbf{x}), \tag{7}$$

and therefore the structure tensor of I reads

$$\begin{aligned} S_I &= (\nabla I)(\nabla I)^T = \begin{bmatrix} (r, g, b)_x \\ (r, g, b)_y \end{bmatrix} \left[(r, g, b)_x^T \;\; (r, g, b)_y^T \right] \\ &= \begin{bmatrix} r_x^2 + g_x^2 + b_x^2 & r_x r_y + g_x g_y + b_x b_y \\ r_x r_y + g_x g_y + b_x b_y & r_y^2 + g_y^2 + b_y^2 \end{bmatrix}. \end{aligned} \tag{8}$$

For the structure tensor S_I in (8), we have

$$||S_I||_2 = ||\nabla I||_2^2 = \lambda_{\max}(S_I). \tag{9}$$

Rewrite S_I as

$$S_I = \begin{bmatrix} J_{xx} & J_{xy} \\ J_{xy} & J_{yy} \end{bmatrix}. \tag{10}$$

Then the quadratic formula gives the larger eigenvalue

$$\lambda_{\max}(S_I) = \lambda_1 = \frac{J_{xx} + J_{yy} + \sqrt{(J_{xx} - J_{yy})^2 + 4J_{xy}^2}}{2}, \tag{11}$$

of which the square root becomes the gradient magnitude and the corresponding eigenvector reads

$$\mathbf{v}_1 = \begin{bmatrix} J_{yy} - \lambda_1 \\ -J_{xy} \end{bmatrix}, \tag{12}$$

which is the edge normal direction.

III. The Reverse-Transition Weighting Filter

This section introduces an innovative denoising operator called the *reverse-transition weighting* (RTW) filter.

A. Signal denoising

We first consider the RTW filter for one-dimensional (1D) signal as in Fig. 1. Given a noisy signal vector $\mathbf{u}^0$, the RTW filter is an iterative procedure of the form

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \eta\, \mathcal{C}_1 \mathbf{u}^k, \quad k = 0, 1, 2, \cdots, \tag{13}$$

where η is a learning rate, and $\mathcal{C}_1 \mathbf{u}^k$ is the correction term obtained from the last iterate $\mathbf{u}^k$.

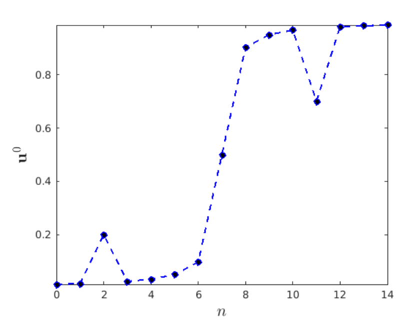

Fig. 1. A noisy synthetic signal, where noise appears at $n = 2$ and 11.

Here our goal is to design an iterative denoising filter which removes noise without altering *monotonic regions* where signal/image values vary monotonically. The goal can be achieved when the correction term of the filter becomes zero in monotonic regions. Note that noise appears as local extrema; in order to suppress local extrema, the correction term must be related to the curvature of the curve. Here we summarize the requirements and desirable outcomes:

- The correction term must be related to the curvature of the signal curve.
- The correction term has nonzero values only at local extrema.
- The resulting iterative filter can suppress noise without weakening monotonic regions, including and around edges.

Now, for each point n, we first define the left and right (one-sided) transitions

$$d_{n,\ell} = |u_n - u_{n-1}|, \quad d_{n,r} = |u_{n+1} - u_n|. \tag{14}$$

Then formulate the correction term, letting the two transitions weight the one-sided slopes reversely:

$$\begin{aligned} \mathcal{C}u_n &= d_{n,\ell}(u_{n+1} - u_n) - d_{n,r}(u_n - u_{n-1}) \\ &= d_{n,r}u_{n-1} - (d_{n,\ell} + d_{n,r})u_n + d_{n,\ell}u_{n+1}. \end{aligned} \tag{15}$$

Claim 1: The correction term $\mathcal{C}u_n$ approximates the curvature at local extrema, while it is zero in monotonic regions. Thus the RTW denoising filter (13) preserves pixel values if they vary monotonically.

Proof. For simplicity, let $n = 2$ in Fig. 1, a local maximum. Then

$$\begin{aligned} d_{2,\ell} &= |u_2 - u_1| = u_2 - u_1, \\ d_{2,r} &= |u_3 - u_2| = -(u_3 - u_2) = u_2 - u_3, \end{aligned}$$

and therefore

$$\begin{aligned} \mathcal{C}u_2 &= d_{2,\ell}(u_3 - u_2) - d_{2,r}(u_2 - u_1) \\ &= (u_2 - u_1)(u_3 - u_2) - (u_2 - u_3)(u_2 - u_1) \\ &= -2(u_2 - u_1)(u_2 - u_3) < 0, \end{aligned}$$

with which the iteration (13) will reduce the image value at $n = 2$ for the next iterate.

Now, let $n = 6$, a point in a monotonic region. Then

$$\begin{aligned} d_{6,\ell} &= |u_6 - u_5| = u_6 - u_5, \\ d_{6,r} &= |u_7 - u_6| = u_7 - u_6. \end{aligned}$$

Thus

$$\begin{aligned} \mathcal{C}u_6 &= d_{6,\ell}(u_7 - u_6) - d_{6,r}(u_6 - u_5) \\ &= (u_6 - u_5)(u_7 - u_6) - (u_7 - u_6)(u_6 - u_5) = 0, \end{aligned}$$

with which $u_6^{k+1} = u_6^k$ in the iteration (13). Using the same arguments, we can show that the correction term is zero at all points in monotonic regions and nonzero at local extrema. □

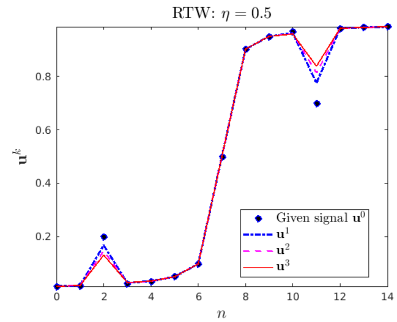

Fig. 2. Three iterations of the RTW filter for the signal in Fig. 1.

Fig. 2 depicts the first three iterates of the RTW filter applied for the synthetic signal in Fig. 1, with $\eta = 0.5$. As Claim 1 shows, the monotonic regions of the signal are preserved, while reducing the noise.

However, the iteration converges slowly; the main reason is that the learning rate η is not easy to set appropriately. To overcome the difficulty, we may scale the correction term by the transition magnitudes. Dividing $\mathcal{C}u_n$ in (15) by the average transition, $(d_{n,\ell} + d_{n,r})/2$, gives

$$\begin{aligned} \mathcal{C}_1 u_n &= \frac{2d_{n,\ell}}{d_{n,\ell} + d_{n,r}}(u_{n+1} - u_n) \\ &\qquad - \frac{2d_{n,r}}{d_{n,\ell} + d_{n,r}}(u_n - u_{n-1}) \\ &= \frac{2d_{n,r}}{d_{n,\ell} + d_{n,r}}u_{n-1} - 2u_n + \frac{2d_{n,\ell}}{d_{n,\ell} + d_{n,r}}u_{n+1}. \end{aligned} \tag{16}$$

The operator $\mathcal{C}_1$ inherits good characteristics of $\mathcal{C}$; only the difference is that the magnitude of the correction is normalized. As a result, the resulting algorithm converges faster and can be analyzed with an appropriate choice of the parameter η. See Remark 2 below.

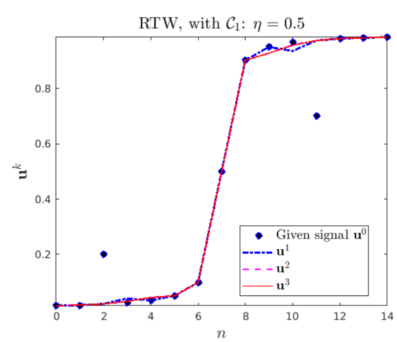

Fig. 3. Three iterations of the scaled RTW filter for the signal in Fig. 1. The iteration converged in two iterations.

In Fig. 3, we present iterates of the scaled RTW filter applied for the same signal in Fig. 1. The scaled RTW filter has eliminated all local extrema in two iterations converging *completely* so that the third iteration results in the exact same signal as $\mathbf{u}^2$, i.e., $\mathbf{u}^3 = \mathbf{u}^2$. Furthermore, the signal values at $n = 6, 7, 8$ (the sharp edge) have remained unaltered during the iteration. The RTW denoising filter is a unique algorithm showing such a property of preserving the edge strength.

B. Image denoising

The RTW denoising filter designed for noisy signals can be easily expanded for the removal of noise in images.

Let $I = (r, g, b)$ be a color image. Then the denoising step (Step 1 of Canny algorithm) can be carried out channel-by-channel. Let $u \in \{r, g, b\}$ be a channel and $u_{i,j} = u(\mathbf{x}_{i,j})$, where $\mathbf{x}_{i,j} = (x_i, y_j)$ is a pixel in the image. At each pixel $\mathbf{x}_{i,j}$, the RTW correction term for the image u can be formulated as the sum of two scaled correction terms in x- and y-directions:

$$\begin{aligned} \mathcal{C}_2 u_{i,j} \;=\; & \frac{2d_{i,j,E}}{d_{i,j,W}+d_{i,j,E}} u_{i-1,j} + \frac{2d_{i,j,W}}{d_{i,j,W}+d_{i,j,E}} u_{i+1,j} \\ & + \frac{2d_{i,j,N}}{d_{i,j,S}+d_{i,j,N}} u_{i,j-1} + \frac{2d_{i,j,S}}{d_{i,j,S}+d_{i,j,N}} u_{i,j+1} \\ & -4u_{i,j}, \end{aligned} \tag{17}$$

where the four one-sided transitions are defined as

$$\begin{aligned} d_{i,j,W} &= |u_{i,j} - u_{i-1,j}| \\ d_{i,j,E} &= |u_{i,j} - u_{i+1,j}| \\ d_{i,j,S} &= |u_{i,j} - u_{i,j-1}| \\ d_{i,j,N} &= |u_{i,j} - u_{i,j+1}|. \end{aligned} \tag{18}$$

Given an image channel $\mathbf{u}^0$, the RTW filter is an iterative procedure of the form

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \eta\, \mathcal{C}_2 \mathbf{u}^k, \quad k = 0, 1, 2, \cdots. \tag{19}$$

where η is a learning rate.

Remark 2: **Stability of the RTW Denoising Filter**
When one of denominators in (17) is zero, the algorithm will be broken. To avoid this, we should modify the four one-sided transitions in (18); take the maxim compared with a small number, e.g., $\varepsilon = 10^{-6}$. This modification may deteriorate the good characteristics of $\mathcal{C}_1$ and $\mathcal{C}_2$, but not much. One can mathematically prove the following.

- For signal denoising, the RTW denoising filter is stable and convergent when $\eta \le 1/2$.
- For image denoising, the RTW denoising filter is stable and convergent when $\eta \le 1/4$.

Remark 3: **Edge Preservation vs. Edge Cleaning**
In image denoising, the RTW filter may alter non-extreme values. At an edge point, the image value is monotonic in the edge normal direction, while it may be extremal in the edge tangential direction. In this case, the RTW filter alters the image value. However, this is not necessarily a drawback of the new filter. Since the filter tries to suppress oscillations in the edge tangential direction, the edge would become clean.
In order to preserve the edge strength more effectively, the correction operator $\mathcal{C}_2$ must be modified to incorporate certain extra edge-preservation operations. This issue will be discussed in a forthcoming research.

IV. Numerical Experiments

The Canny edge detection algorithm is modified and implemented in Matlab for color images. In this section, we will demonstrate the effectiveness of the RTW filter and compare gradient-fusion methods, which are respectively related to Step 1 and Step 2 of the Canny algorithm.

We downloaded three images from the public domain: Fruits, Happy-Fish, and Lena, as shown in Fig. 4. We selected a noisy version for the Happy-Fish image.

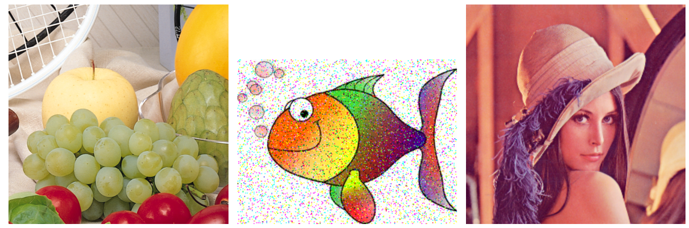

Fig. 4. Three public-domain images: Fruits, Happy-Fish, and Lena.

A. Noise removal

Here we will compare the Gaussian filter and the new RTW filter for edge detection. For a fair comparison, we keep the remained steps (Steps 2–5) the same for both filters. Each of the color images in Fig. 4 is denoised channel-by-channel. Then in Step 2, their Sobel gradients are combined using the structure tensor method. For the Gaussian filter, we use the built-in function `imgaussfilt(I,sigma)`; we set the parameter `sigma` separately for each image to result in its best edges. For the RTW filter, we set $\eta = 0.25$ and the iteration stops in five iterations.

Fig. 5 depicts edge detection results for the three images. As one can see from the figure, the Gaussian filter (left) and the RTW filter (right) perform similarly for clean images. However, the Gaussian filter is not effective for the Happy-Fish image which is a noisy image. The RTW filter has shown its effectiveness for various images.

B. Gradient-fusion methods

In this subsection, we compare three gradient-fusion methods in Step 2: the structure tensor (§II-C), the maximum of RGB gradient magnitudes $\big(\max(|\nabla r|, |\nabla g|, |\nabla b|)\big)$, and the L^2-norm of the RGB gradient magnitudes $\big(\sqrt{|\nabla r|^2 + |\nabla g|^2 + |\nabla b|^2}\big)$. The images are denoised by using the RTW filter in Step 1.

In Fig. 6, we present edge detection results for the Happy-Fish image, with the gradient magnitudes being combined by the structure tensor, (c) the maximum, and (d) the L^2-norm. As one can see from the figure, the structure tensor method gives the best result; there is an observable difference on the grinning mouth.

Fig. 7 shows edge detection results for the Lena image, with the gradient magnitudes being combined by the structure tensor, (c) the maximum, and (d) the L^2-norm, as in the last figure. For this example, the Lena image is perturbed by a Gaussian noise of $\sigma = 0.1$ and a pepper-and-salt noise of

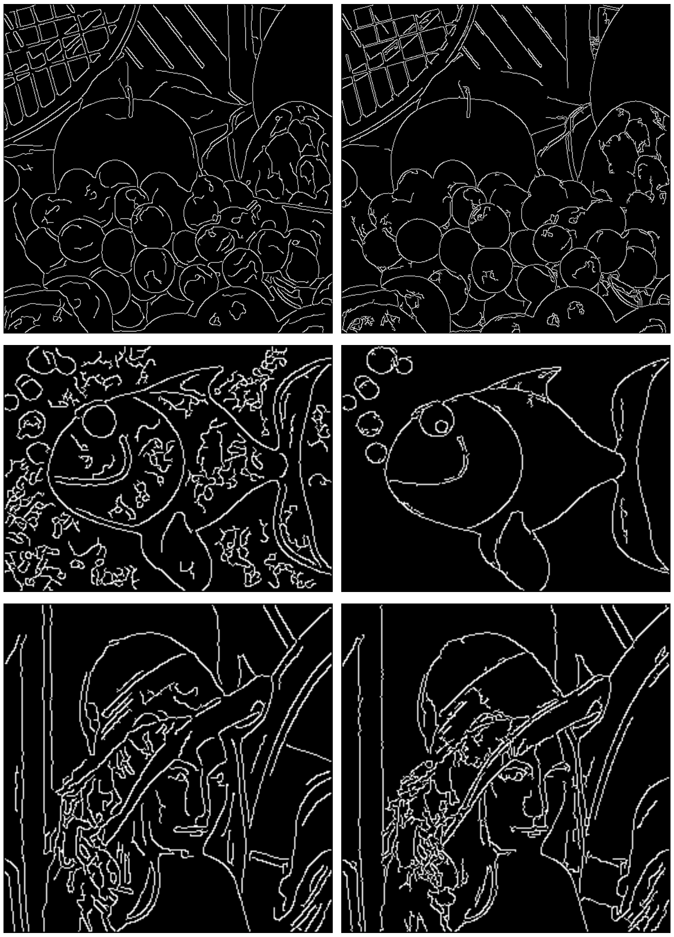

Fig. 5. Edge detection results for the three images: Using the Gaussian filter (left) and the RTW filter (right).

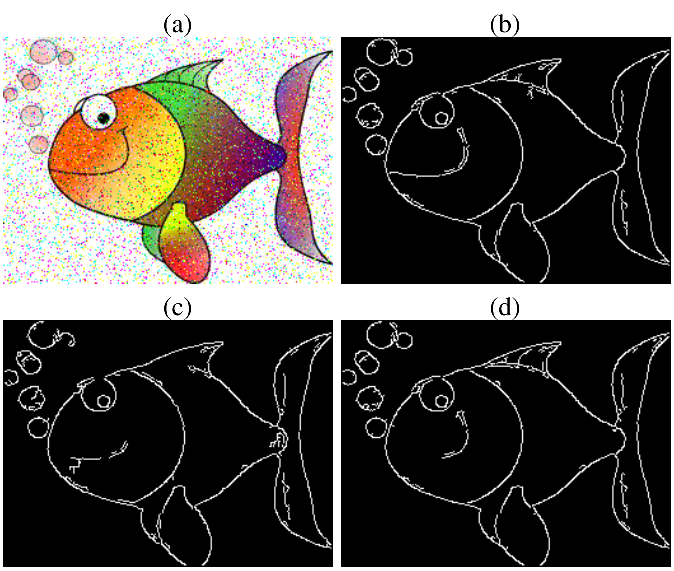

Fig. 6. Gradient-fusion methods for Happy-Fish: (a) The input image and edge detection results, with the gradients combined by (b) the structure tensor, (c) the maximum, and (d) the L^2-norm.

Fig. 7. Gradient-fusion methods for Lena: (a) The input image and edge detection results, with the gradients combined by (b) the structure tensor, (c) the maximum, and (d) the L^2-norm.

Fig. 8. Restored images of Happy-Fish and Lena, obtained by five iterations of the RTW filter.

density 0.1. All of the three gradient-fusion methods result in acceptable edges; the artificial noise must be effectively suppressed by the new RTW filter without weakening the edge strength.

The three gradient-fusion methods work well for all test cases, with the structure tensor method being the best occasionally.

In Fig. 8, we display the restored images of Happy-Fish and Lena obtained respectively from Fig. 6 (a) and Fig. 7 (a). For both results, the RTW filter has run five iterations for each channel. The images are denoised satisfactorily without weakening the edge strength.

V. Conclusions

Edges carry important information because they mark the boundaries of objects in the image. Thus, edge detection allows us to interpret image content effectively and becomes a necessary or desirable step for various tasks in image processing and computer vision. However, most edge detection algorithms are sensitive to noise and denoising operations may weaken the edge strength. This article has introduced a new iterative procedure, called the *reverse-transition weighting* (RTW) filter, which can suppress noise without an observable weakening of the edge strength. Such a unique feature of the new filter has been verified numerically and mathematically. The new filter has been adopted for the denoising step of the Canny edge detection algorithm and shown its effectiveness for various examples.

Acknowledgment

The results in this article are collected from a project entitled *Canny Edge Detection Algorithm for Color Images* and assigned by Dr. Seongjai Kim, in a junior-level class at Mississippi State University in Fall 2023. Students (the first five authors) worked hard to produce recognizable results in both implementation and numerical tests. Dr. Kim deeply appreciates their efforts and passion.

References

[1] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[2] S. Zhan and R. Mehrotra, "A zero-crossing-based optimal three-dimensional edge detector," *CVGIP: Image Understanding*, vol. 59, no. 2, pp. 242–253, 1994. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1049966084710163

[3] L. Zhai, S. Dong, and H. Ma, "Recent methods and applications on image edge detection," in *2008 International Workshop on Education Technology and Training & 2008 International Workshop on Geoscience and Remote Sensing*, vol. 1, 2008, pp. 332–335.

[4] A. Roy and R. H. Laskar, "Fuzzy SVM based fuzzy adaptive filter for denoising impulse noise from color images," *Multimedia Tools and Applications*, vol. 78, pp. 1785–1804, 2019.

[5] S. Schulte, V. De Witte, and E. E. Kerre, "A fuzzy noise reduction method for color images," *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1425–1436, 2007.

[6] S. Kim, "PDE-based image restoration: A hybrid model and color image denoising," *IEEE Trans. Image Processing*, vol. 15, no. 5, pp. 1163–1170, 2006.

[7] W. Liu, C. Wu, and T. Xu, "Adaptive total variation model for image denoising with fast solving algorithm," *Application Research of Computers*, vol. 28, pp. 4797–4800, 12 2011.

[8] R. Medina-Carnicer, R. Muñoz-Salinas, E. Yeguas-Bolivar, and L. Diaz-Mas, "A novel method to look for the hysteresis thresholds for the canny edge detector," *Pattern Recognition*, vol. 44, no. 6, pp. 1201–1211, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320310005741

[9] Z. Cai, Z. Ma, Z. Zuo, Y. Xiang, and M. Wang, "An image edge detection algorithm based on an artificial plant community," *Applied Sciences*, vol. 13, no. 7, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/7/4159

[10] W. Rong, Z. Li, W. Zhang, and L. Sun, "An improved canny edge detection algorithm," in *2014 IEEE International Conference on Mechatronics and Automation*, 2014, pp. 577–582.

[11] R. Song, Z. Zhang, and H. Liu, "Edge connection based canny edge detection algorithm," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 8, pp. 1228–1236, 11 2017.

[12] L. Xuan and Z. Hong, "An improved canny edge detection algorithm," in *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2017, pp. 275–278.

[13] G. Xin, C. Ke, and H. Xiaoguang, "An improved canny edge detection algorithm for color image," in *IEEE 10th International Conference on Industrial Informatics*, 2012, pp. 113–117.

[14] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, ser. IJCAI'81. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1981, pp. 674–679. [Online]. Available: http://dl.acm.org/citation.cfm?id=1623264.1623280

[15] S. Yuan, Y. Chen, C. Ye, and M. D. Ansari, "Edge detection using nonlinear structure tensor," *Nonlinear Engineering*, vol. 11, no. 1, pp. 331–338, 2022. [Online]. Available: https://doi.org/10.1515/nleng-2022-0038

[16] S. Alpert, M. Galun, B. Nadler, and R. Basri, "Detecting faint curved edges in noisy images," in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 750–763.