IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier xxxx

Enhanced edge detection using SR-guided threshold maneuvering and window mapping: Handling broken edges and noisy structures in Canny Edges

DEEPAK DHILLON (Graduate Student Member, IEEE), and RAJLAXMI CHOUHAN (Senior Member, IEEE)
Department of Electrical Engineering, Indian Institute of Technology, Jodhpur, India
Corresponding author: Deepak Dhillon (e-mail: deepak.2@iitj.ac.in).

This work is supported by Department of Science & Technology, Government of India, under project #ECR/2016/001606

ABSTRACT Preserving edges in a noisy environment is a challenging task as even some of the latest end-to-end deep learning (DL) algorithms continue to struggle in achieving high pixel-level accuracy. As the Canny Edge Detector (CED) continues to be one of the most popular edge detection operators, this paper presents an *enhanced* CED using Stochastic Resonance (SR) guided threshold maneuvering and window mapping, which takes the same input parameter set as that of the conventional Canny but produces the edge map with better-connected edges and reduced noise. The SR-based analysis informs the steps that should be followed to enhance the performance of the classical CED. We also propose a new measure for efficient edge detection; a unique, efficient way of edge content extraction and its combination for various channels; and a framework to handle repercussions of the randomness of the noise. Since the proposed solution comes in the form of a modular patch-based framework, it can be easily incorporated into other algorithm developments. Qualitative and quantitative results are presented along with the BSDS500 & BIPED benchmarking to showcase the proposed algorithm's effectiveness. On BIPED benchmarking, our algorithm gives the human-level performance ($F1$ score .79), which is appreciable considering that it is a non-DL–based algorithm.

INDEX TERMS Image edge detection, stochastic resonance, smart cameras, noise, digital cameras, image edge analysis, thresholding, image filtering, feature extraction.

I. INTRODUCTION

WHEN working with edges, pixel-level details hold pertinent information. Even some of the latest popular end-to-end deep learning algorithms like HED (Holistically-Nested Edge Detection) [1], RCF (Richer Convolutional Features) [2], etc. give thick grayscale edges as the output. These thick edges are usually then processed with some thinning algorithm and thresholded to get the final thin binary edge map. These processes are usually not that accurate; a threshold value decided based on an ensemble of the dataset could give a incorrect result on a single instance (single image) from the same dataset (because of the high variability within the dataset). Further, the edge-thinning algorithms somewhat deteriorate the wide-edge input provided to them. In short—Edges are tough to handle.

Canny Edge Detector (CED) [3], though presented more than three decades ago, is still one of the most popular edge detection algorithms. CED algorithm or its implementation function in OpenCV [4] or MATLAB® [5] mainly takes three inputs:

1) an input image
2) a low threshold
3) a high threshold

and gives a *binary edge map* as the output. Even for the cases of a single input (only image) or two inputs (image & a threshold) in MATLAB® implementation of CED, the two thresholds are first calculated in the back-end, and then the main CED algorithm is applied.

A. THE PROBLEM UNDER CONSIDERATION

Consider the example shown in Figure 1, where CED [3] with an input parameter set is applied to a digital photo-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

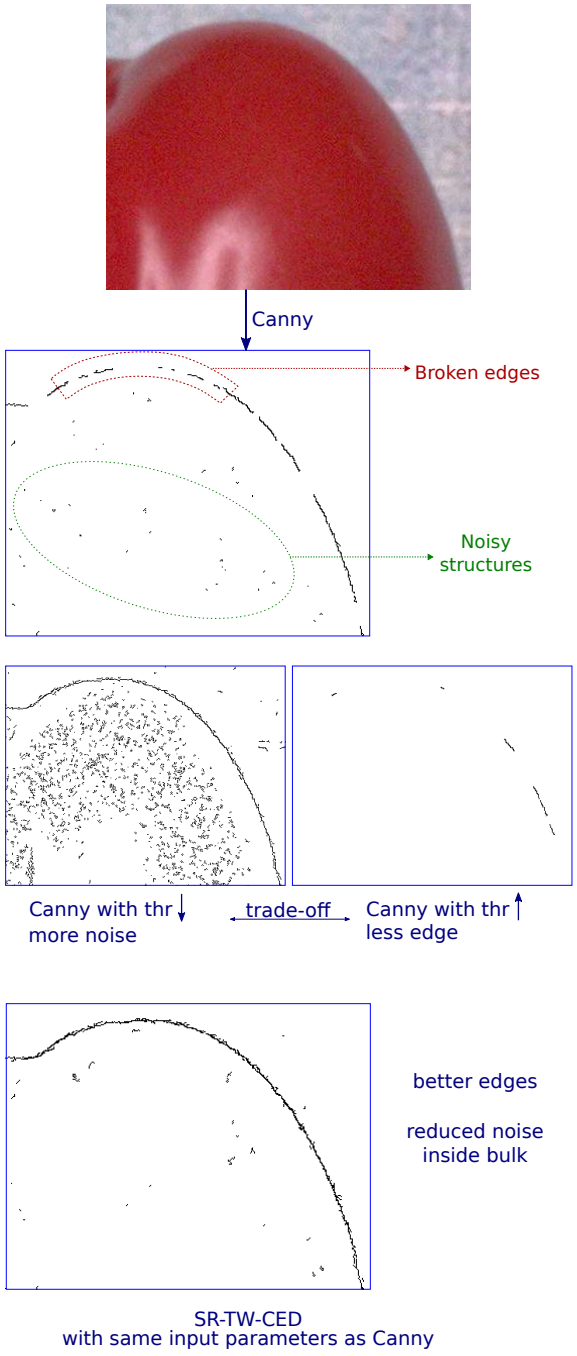

Figure 1. The Problem

graph captured from a Google Pixel™ smartphone [6]. Two of the main problems with CED [3] are:

1) Broken edges
2) Noisy structures

as is shown in Figure 1. These noisy structures are not necessarily one-or-two pixels long but can be of a larger lengths and unpredictable shapes. To get better edges from the CED, we can decrease the thresholds; this, however, will increase the noisy structures in the edge map. On the other hand, to decrease the noisy structures, we can increase the thresholds but this leads to more broken edges. There actually exists a trade-off between *Better edges* and *less noise* in the CED. This problem can be mitigated using some pre-processing or post-processing steps but can not altogether be eradicated. Further, pre/post-processing can unfavorably alter our data in addition to the extra computations added. Many a times, there does not exist a sweet spot of the threshold values, such that we get very good edges with very low noise. This paper focuses exactly on this problem of getting *Better edges* and *less noisy structures* in the edge map of an input image. Figure 1 shows the result of the proposed **SR-guided enhanced CED using Thresholding maneuvering and Window mapping (SR-TW-CED)**, with the same input parameter set as that of CED, without any pre/post-processing, but by modifying the core of the CED algorithm. In our result, some noisy structures can be seen in the areas near the edges, as we have prioritized 'edges and their neighborhood' over 'noise'. This effect, however, would be less visible when the full size of the image is considered. The edge maps presented in Figure 1 are inverted (black↔white) to save the printing resources (when printed on a white paper). More insight over the problem can be gained from Section II. Further details of the proposed algorithm are presented in Section III.

B. EXISTING LITERATURE AND RELATED WORK

1) Edge Detection

Work in edge detection dates as early as seven decades ago and is still an important topic in current research. Various classical as well as learning-based algorithms have progressed the work of edge detection. Gradient has an important property that it always points in the direction of maximum change, and edges occur only at the places where there is a change (i.e. no change, no edge). Various classical algorithms were based on this property of the gradient. These algorithms include Roberts [7], Prewitt [8], Sobel [9], Kirsch [10], & Marr-Hildreth [11]. Canny [3] considered edge detection as an optimization problem satisfying three objectives—Low error, good localization, & single edge response.

With a small loss in detection accuracy, SMED ((Scale Multiplication Edge Detection)) [12] uses scale multiplication to improve the localization accuracy in CED [3]. To implement Canny [3] faster, Distributed Canny [13] breaks the whole image into blocks and implements a parallel FPGA implementation. It calculates the local variance at each pixel, which is then used to evaluate the threshold of the enclosing block. Instead of manually choosing the threshold & sigma for CED [3], Kalbasi *et al.* [14] created a lookup table based on the noise intensity present in the image for selecting these parameters.

Instead of relying on thresholding, Edge Drawing [15] takes edge extraction as a dot by dot (anchors) boundary joining problem. Whereas PEL (Predictive Edge Linking) [16] works on joining the small edge segments to create

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

longer edges. Yang *et al.* [17] used 2D entropy to adjust the threshold automatically and kept linking the edge segments until the edge percentage achieves a reference value.

The research on edge detection has again blossomed with the availability of exhaustive datasets like BSDS500 [18], NYUD [19], & BIPED [20], and access to the supporting hardware for deep learning techniques. gPb (Globalized Probability of Boundary) [18] uses a new method for gradient signal calculation. At each pixel, it calculates the histogram in each half-circular disk around the pixel. It also includes working on a texture channel that involves convolution with 17 filters, and combines multiple cues from brightness, color, & texture channels to a spectral clustering framework. SE (Structured Edge Forest) [21] articulates the edge detection problem as the local segmentation mask prediction from the image patches. It labels each pixel as Edge or Not Edge and uses a random forest framework to catch the structured information. HED [1] uses VGG net [22] to learn the hierarchical features, & includes side outputs from the last stage of each layer to contribute to the final end-to-end edge detection system. RCF [2] removes all the fully connected layers and uses outputs from all the convolution layers in a way to facilitate the back-propagation. BDCN (Bi-Directional Cascade Network) [23] uses a different supervision strategy; instead of training different intermediate layers with the same ground truth, it uses a layer-specific supervision. It trains the shallow layers for low-level details and the deep layers for object-level details. With this strategy, BDCN achieves the performance with significantly reduced number of parameters. DexiNed (Dense EXtreme Inception Network for Edge Detection) [20] uses an Xception [24]-like network followed by a multi-scale learning network (inspired from HED [1]). The upsampled outputs from various layers are fused together to obtain the final edge map. DexiNed [20] claims to be the first DL-based edge-detector that worked towards thin edge maps. Bhattacharjee and Roy [25] have presented a normalized Pattern of Local Gravitational Force Magnitude (PLGFM), an edge detector inspired by the universal law of gravitation force. For each pixel, the authors propose to evaluate the force exerted on the central pixel by its neighborhood using a novel filter designed for the same. A remarkable feature of PLGFM is that it is illumination-invariant and thus extracts edges from low-illuminated areas as well. Li *et al.* [26] presented a contour sketch algorithm that generates and detects the object boundaries, salient inner edges, and salient background edges using a GAN-based network. It also includes an artistic style while generating the contour sketch.

2) Stochastic Resonance

Most of the natural systems are intrinsically nonlinear and noisy. Noise is usually cursed to deteriorate the signal; the higher the noise, higher the deterioration. However, this is not always the case. Stochastic Resonance (SR) [27] [28] is a counter-intuitive phenomenon, where the response of a nonlinear system to the weak input is actually enhanced with an optimal amount of noise. There actually occurs constructive cooperation (resonance) between the feeble deterministic signal and the stochastic noise. As the noise increases, the response of the system to the weak signal improves, reaches a maximum at the optimal level of the noise, and then decreases on a further increase of the noise. The bell-shaped curve (as in Figure 3) is the characteristic curve of the SR phenomenon. Gammaitoni *et al.* [27] states three required gradients for the SR phenomenon:

1) an activation barrier (threshold)
2) a weak input signal
3) noise

Gammaitoni explains the SR mechanism with the motion of a heavily-damped particle in a double-well potential dynamical system. SR has also been experimentally verified in ac-driven Schmitt trigger, bistable ring laser, etc. [27] [29]. In [30] Gammaitoni discussed SR with the focus on the thresholded systems (while we have used SR and 'noise-induced threshold crossing' synonymously in our paper).

Resonant Retina [31] exploits the effect of shaking the optical axis of a camera (AWGN) to detect the edges from the captured images. Chouhan *et al.* [32] used dynamic SR to induce the transition at every pixel in the image from a low-contrast to a high-contrast state, thus enhancing a dark and low contrast image. Asha *et al.* [33] further utilized this image enhancement model in high sub-bands of the shearlet transform domain to enhance the low contrast satellite images and their structural details. Rallabandi *et al.* [34] used SR in the Fourier domain to enhance the MRI images. Dhillon and Chouhan [35] used SR in edge-preservation while denoising an image. Liu *et al.* [36] used the dynamic SR in both the spatial as well as the spectral domains to enhance the shadow areas in hyperspectral images. The dynamic SR enhanced images are then fused with the original data and followed by a CNN classifier. Singh *et al.* [37] used an optimized multistable SR technique for contrast enhancement of MRI images to enhance the detection of a lesion (tumor) in the pituitary gland, which is otherwise very difficult to be detected. The patent [38] talks in general that the performance of any detector can be improved if a suitable amount of noise is added to the input signal before it is passed through the detector.

C. KEY CONTRIBUTION

This paper presents an enhanced Canny Edge Detector, which takes the same input parameter set as that of the Canny, and produces the edge map with better-connected edges and reduced noise. Our key contributions include:

- Demonstration of the exhibition of Stochastic Resonance (noise-induced threshold crossing) in Sobel/CED for Smartphone images (with their corresponding noise, not AWGN) (Section II)
- Analyzing CED from the point of view of noise-enhanced stochastic resonance and inferring the steps

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

to improve it

- Jointly addressing the two-fold problem of broken edges and noisy structures of CED
- A new measure—$p_{measure}$ proposed; first instance of use of $e_{measure}$ & its combination with $p_{measure}$ for efficient edge detection (Section III-I)
- A unique way of extracting the edge content from patches and combining various channels efficiently (grayscale, R, G, & B) (Section III-G, III-H)
- A framework to handle repercussions of the randomness of the noise using Processed Window Mapping (connectivity, line filling, hole filing, & isolation removal) (Section III-E)
- Two versions proposed to expand the usefulness of our algorithm (Section IV-C)
- Modular framework and patch-based processing to enable easy future developments and multiprocessing

II. EXHIBITION OF STOCHASTIC RESONANCE BY SOBEL/CANNY EDGE DETECTOR IN SMARTPHONE IMAGES

Stochastic Resonance (SR) is a phenomenon that manifests only in non-linear systems, where a weak signal is optimized/amplified by the *assistance* of noise [27]. The basic idea of observing SR is that the sensitivity of a weak signal in a non-linear system can be amplified by *addition* of controlled amounts of noise. At an optimal amount of noise, a maximization of the SNR or any other performance is observed. Equally important to the exhibition of SR is that how can we use it to improve our systems. As per the authors' knowledge, this is the first instance of demonstrating SR (noise-induced threshold hopping) in Sobel/CED in images captured by smartphone cameras (& thus their corresponding noise, not AWGN). The authors have tried to make it as simple and crisp as possible.

A. SR MANIFESTATION: DIFFERENT NOISE, SAME THRESHOLD

Figure 2 shows the results of applying the Sobel edge detector on the image scene shown in Figure 1. By Sobel edge detector, the authors are referring to *Sobel operator* followed by a *threshold*. In Figure 2, on the left is shown the result on the Pristine image, and on the right is that for the noisy image. Therefore, the noise level has increased from left to right, but the threshold value was kept the same. The result shows that edges are better visible from the noisy image than that from the pristine image, which is **counter-intuitive**! The True-Positive edge detection has increased from 41 to 362, further reinforcing that more edges have appeared in the noisy case. The reason for this is 'noise-induced threshold crossing' or SR [30]. *Red peaks in the surface plots are the ridges that have crossed the threshold.* As can be seen, only a small number of ridges from the pristine image could cross the threshold. Whereas, in the noisy case, the noise has actually helped more edge peaks to cross the threshold. Here, the noise has proved to be *useful* for edge

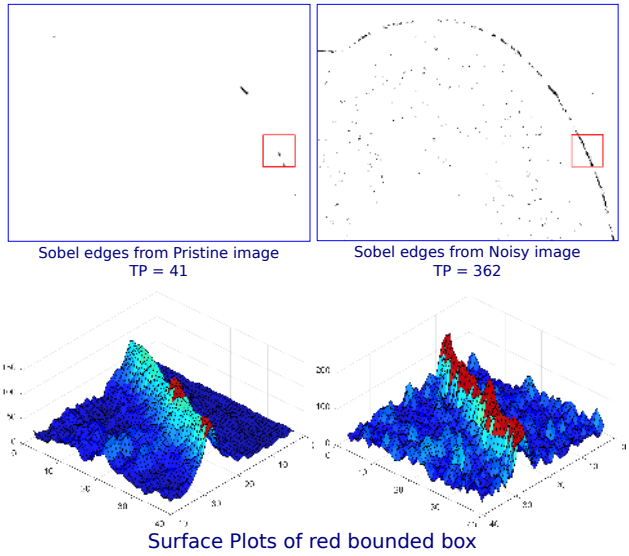

Figure 2. SR in Sobel edge detector on a smartphone image

detection. Similar results were observed for the case of CED, as Canny mainly uses two thresholds instead of just one. Also, similar results were observed for DSLR images.

B. SR MANIFESTATION: SAME NOISE, DIFFERENT THRESHOLD

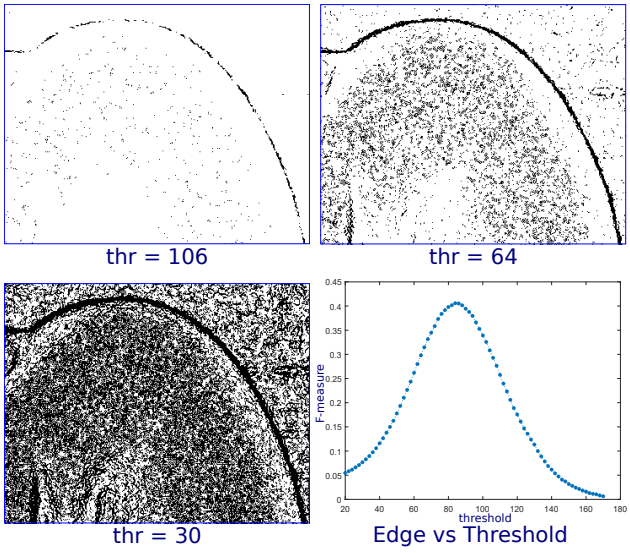

Figure 3. SR in Sobel edge detector on a smartphone image

Another manifestation of SR considered in this paper is by not adding up any noise but changing any system's internal parameter like a threshold. Consider again the noisy image shown in Figure 1; figure 3 shows the results of applying Sobel edge detection with decreasing thresholds. As observed, as we keep on decreasing the threshold, more and more edges keep on appearing (here, as black pixels). This can also be easily visualized using the surface plots of Figure

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

2, as we will decrease the threshold, more & more ridges would be able to cross the threshold, & appear as edges. '$F1$ measure (edges) vs. threshold' curve in Figure 3 shows that decreasing the threshold helps the edges until an optimal threshold is reached, after which the noise starts dominating over the edges. This bell-shaped curve is characteristic [27] of SR, where the optimal value of the parameter lies between the low & high values.

Therefore, for better edges, we can decrease the threshold. However, this comes with a shortcoming—on decreasing threshold, edge content increases while also increasing noise content in the output. Another observation is—lower threshold favors more edge content, & a higher threshold favors less noise content.

C. IDEA

The idea that we got from this SR phenomenon that drives our algorithm development is—decrease the threshold for edge areas, & increase the threshold for noisy areas. This could enhance the edges as well as reduce the noise.

III. PROPOSED ALGORITHM: SR-GUIDED ENHANCED CANNY EDGE DETECTOR USING THRESHOLDING MANEUVERING AND WINDOW MAPPING (SR-TW-CED)

A. PRINCIPLES KEPT IN MIND

The authors have followed the following principles while developing the algorithm:

- Modular framework
- Lower computations preferred over a small loss of accuracy
- *Better edges* preferred over *lower noise*

B. CHANNEL SEPARATION

Figure 4 shows the proposed modular-framework & algorithm. The proposed algorithm deviates from the MATLAB® [5], & OpenCV [4] implementation of CED from the beginning itself. Consider a colored image from a 'consumer camera' or a 'smartphone camera' is provided as the input image. MATLAB® does not support the multi-channel input, & one has to convert the colored image to the grayscale image to use it further. OpenCV does supports the colored input, but it picks (at a pixel) the highest *gradient magnitude* among all channels, which is quite different from ours.

The proposed algorithm separates the colored input into their ingredient color channels—R, G, & B, and also creates a grayscale component using the HSI model. Thus, in contrast to Matlab® & OpenCV, which uses one & three channels respectively, we use four channels R, G, B, & grayscale. Most of the computations go with the processing over the grayscale channel, whereas only the *Edge Content* is extracted from the R, G, & B channels (subsection III-H). Even though the inclusion of R, G, & B channels in addition to the grayscale channel have added to the computation, the authors have found it to improve the results significantly.

C. CED STEPS

Like in Canny Edge Detector [3], the grayscale image is then smoothened using the convolution with a gaussian kernel. The smoothened image is then operated upon with the Sobel filter to obtain a gradient-intensity image and a gradient-angle image. Non-Max Suppression uses the gradient-angle image to reduce the *wide* edges from the gradient-intensity image to *narrow* edges. The *gradient-intensity image* is then thresholded with T_l, & T_h individually (both provided as the input parameters) to obtain two separate thresholded images. Here T_l denotes *lower threshold*, & T_h denotes *higher threshold*.

D. RAW WINDOW MAPPING

The output from the 'Double Thresholding' comprise two images—a *low-thresholded image* and a *high-thresholded image*. Some of the next few steps of the proposed algorithm depend only on the *low-thresholded image*. The Raw Window Mapping partitions the *low-thresholded image* into square windows of 50×50 pixels (number found analytically) and creates a Raw Window Map (RWM) of reduced size (1/2500 size reduction). Each 50×50-pixel window from *low-thresholded image* is mapped to a single pixel in the RWM. As shown in Figure 4 (color-coded, blue color), each pixel in RWM is associated with two types of flags–Edge flag and Noise flag.

- For Edge flag, the 50×50 window is categorized as:
  - -- *Sure Edge* (SE) if it contains an edge for sure
  - -- *Unsure Edge* (UE) if the window probably has an edge but is not sure
  - -- *No Edge* to cover the remaining set relating the edge flag
- For Noise flag, the 50×50 window is categorized as:
  - -- *Noisy* if it contains noise
  - -- *Not Noisy* (NN) to cover the remaining set relating the noise flag

The *No Edge* flag or *Not Noisy* flag does not ensure that the window would not have any edge or noise, but only that it is very less likely to have them. In other words, focus is on detecting the presence of Edge or Noise, not on detecting their absence. The process of flag selection is explained in detail in Subsection III-I.

E. PROCESSED WINDOW MAPPING

Input to this stage is the Raw Window Map. In the absence of noise, a simple raw window mapping would have sufficed. However, in order to handle the repercussions of the random nature of noise, the proposed algorithm adopts various strategies. All these strategies are enveloped under the name 'Processed Window Mapping' (PWM) as shown in Figure 4 (yellow color). Our inclusion of *Unsure Edge* flag in between the *Sure Edges* and *No Edges* is also a reflection of handling this randomness. Following are the modifications done to the Raw Window Map to get the Processed Window Map:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

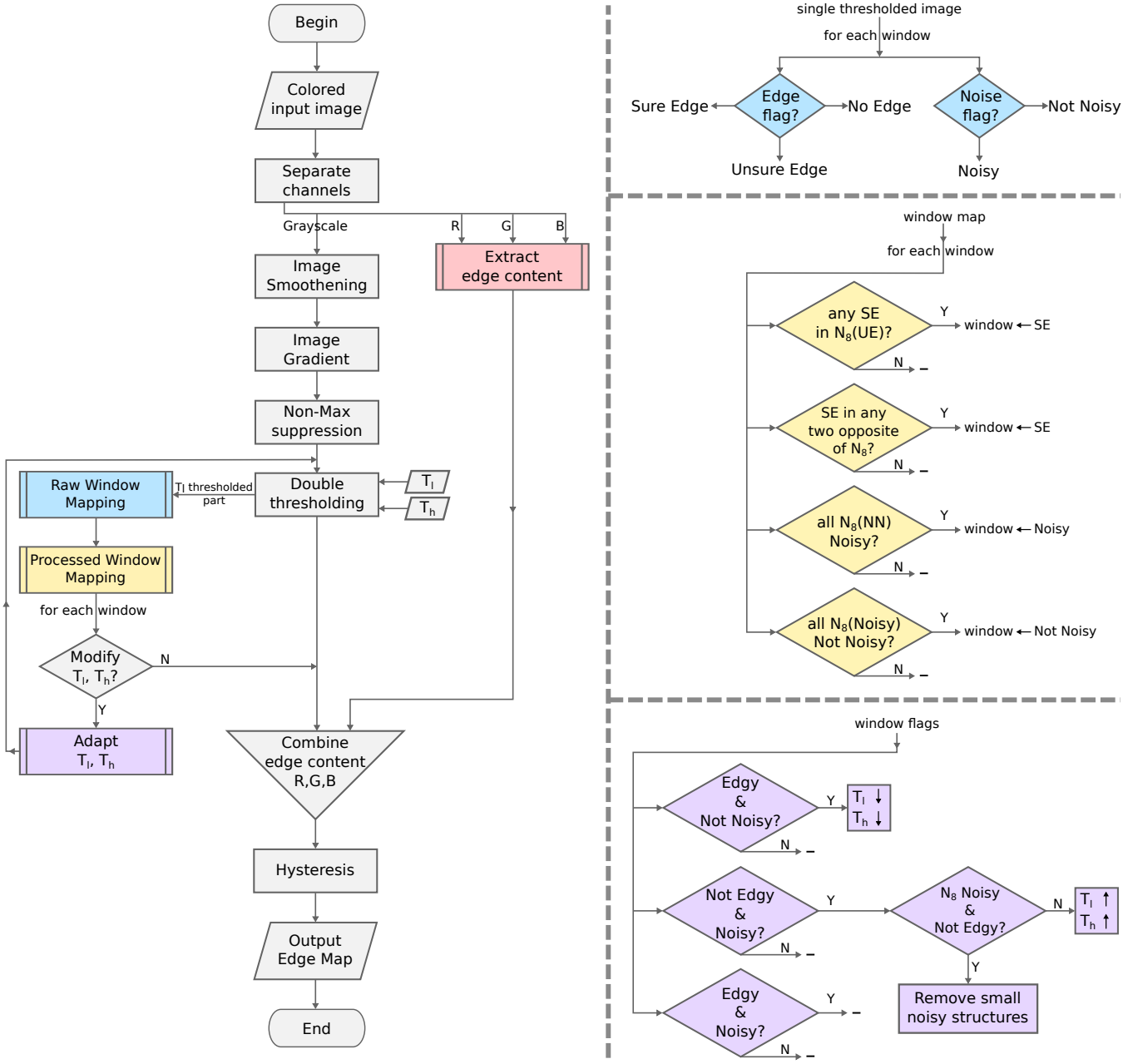

Figure 4. The Proposed Algorithm

- *Connectivity*: If an *Unsure Edge* has a *Sure Edge* in N_8 neighbor, modify its flag to *Sure Edge*
- *Line filing*: If the current pixel has two opposite pixels (i.e. preceding and succeeding pixels along any direction) in N_8 as *Sure Edge*, modify its flag to *Sure Edge*
- *Hole filing*: If all the N_8 open neighborhood of a *Not Noisy* window are *Noisy*, modify its flag to *Noisy*
- *Isolation removal*: If all the N_8 open neighborhood of a *Noisy* window are *Not Noisy*, modify its flag to *Not Noisy*

During PWM, all the *Unsure Edge* flags have been dissolved either into *Sure Edge* or *No Edge*. The PWM is used to decide whether to modify the thresholds of the CED—T_l and T_h.

F. ADAPT T_L AND T_H

The stage of adaptation does careful maneuvering of T_l and T_h so as to enhance the edges as well as reduce the noise. In case of clashes, the edge is given preference over the noise. Following are the modifications done to T_l and T_h based on the edge/noise flags from the Processed Window Map:

- If the flag is *Edge* and *Not Noisy*, decrease both T_l and T_h
- If the flag is *Not Edge* and *Noisy*, increase both T_l and T_h. Also, if all the N_8 open neighborhood is *Noisy* and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

Not Edge, remove small noisy structures
- If the flag is *Edge* and *Noisy*, no modification. Although both T_l and T_h can be increased, if low noise is preferred over a little loss of edges.

For the cases where T_l and T_h are modified, the step of 'Double Thresholding' is executed again with the modified parameters. This way, the step of 'Double thresholding' is executed in total only once for the case of no modification (in T_l & T_h), and twice for the case of modification.

G. COMBINE R-G-B EDGE CONTENT

As the input to this step, we have two thresholded images and an image with *edge content* that was extracted from R, G, & B channels. To combine the extracted edge content, it is sufficient to combine it only with the high-thresholded image.

Next is the step of 'hysteresis' [3], where the two thresholded images are intelligently merged to a single edge map. This edge map is our **final output**.

H. EXTRACT EDGE CONTENT FROM R G B CHANNELS

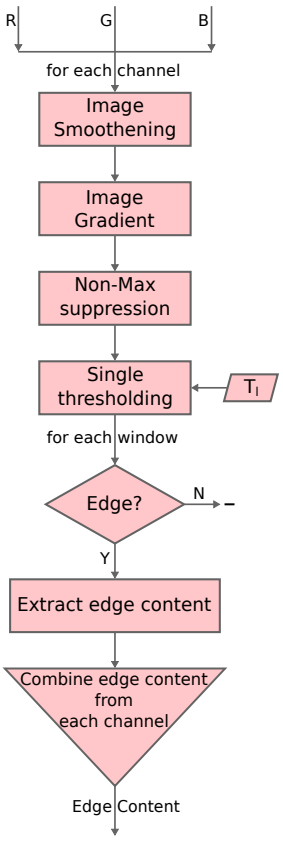

Figure 5. Extract Edge Content from R G B channels

The 'Extract edge content' block from figure 4 is illustrated in Figure 5. Input to this block are three channels R, G, & B. In this block, the focus is only on extracting the edges and completely avoiding the noise. For each channel, the image is first smoothed using convolution with a gaussian kernel. Then the image gradient is calculated using the Sobel operator. Non-Max Suppression is then used to sharpens the edges. After this, instead of double thresholding, only single thresholding with T_l is used. The 'single thresholded image' is then partitioned into windows of 50×50. For each window, the decision is taken, whether it contains an edge or not. If the window contains an edge, the two Largest Connected Components (LCC) are extracted and are considered as the *Edge Content* for that window.On spanning all the windows from a channel, we get the *Edge Content* for that channel. *Edge Content* from each of the three channels R, G, & B are then combined into a single channel which is referred to as the *Edge Content* from RGB channels.

I. EDGE FLAGS & NOISE FLAGS

1) Edge Flags

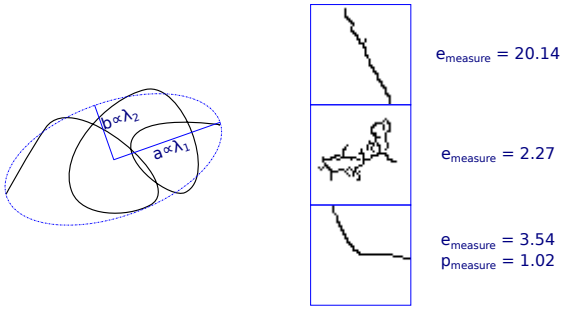

Figure 6. Edge Flag using e-measure, p-measure

To know whether a window has an edge or not, it is categorized for its edge flag. To find the edge flag, the authors created two measures—$e_{measure}$ and $p_{measure}$; which are simple & efficient to calculate. The largest connected component (LCC) from the window is extracted and checked for $e_{measure}$ and $p_{measure}$ tests; $e_{measure}$ is a measure for elongation, which is similar to the feature descriptor *eccentricity* but has the range 1 to ∞. $e_{measure}$ is calculated as:

$$e_{measure} = \sqrt{\frac{\lambda_1}{\lambda_2}} \quad where\ \lambda_1 \geq \lambda_2 \tag{1}$$

where λs are the eigenvalues of the covariance matrix of the curve. Consider an LCC curve extracted be as shown in Figure 6, on the left in black. For this curve, we can create an ellipse with the same second central moment as that of the curve. Length of the major-axis and the minor-axis of this ellipse are proportional to the eigenvalues of the covariance matrix of the curve. Therefore, more elongated the curve is, the more elongated the ellipse becomes, and higher is the $e_{measure}$. Therefore, we consider a window to have *Sure Edge* if it evaluates to a high value of the $e_{measure}$. In Figure 6, on the right are shown three LCC curves. The top one has a high $e_{measure}$ & is thus a *Sure Edge*. The middle one is a noisy structure, and thus have a low $e_{measure}$. The bottom one, though, is an edge but fails the $e_{measure}$ test because it has a corner (change in direction). Since we use a small

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

window size, this sudden change in direction is not observed in most of the edge windows. This is because a long curve can be approximated with small piece-wise straight lines.
To handle these changes in the directions, we use $p_{measure}$ test. We created $p_{measure}$ taking 'plus sign' (+) as the reference. A curve in the shape of the plus sign have $e_{measure}$ value as 1 (equivalent ellipse will be a circle), as $a = b$. $p_{measure}$ is calculated as:

$$p_{measure} = \frac{length\,of\,curve}{k.(\sqrt{\lambda_1} + \sqrt{\lambda_2})} \tag{2}$$

where k is a proportionality constant. $p_{measure}$ puts a constraint on the curve's length and measures its ratio with the sum of the major & minor axes of the equivalent ellipse. In Figure 6, in bottom-right image, $p_{measure}$ is used to conclude if the window has an edge. A low value of $p_{measure}$ assures the *Sure Edge*. For both the $e_{measure}$ and $p_{measure}$ tests, the cut-off (threshold) values have been decided analytically.
Since the covariance framework carries an inherited assumption of *number of data points* to be much larger than the *number of dimensions*; for the cases where there is a possibility of ambiguity in the surety of edges, we assigned those windows as *Unsure Edge*. These Unsure Edges are then handled in the Processed Window Mapping step (III-E).

2) Noise Flags

To evaluate whether a window is noisy or not, we have used a fast and easy method. We assign a window to be noisy if the number of isolated pixels it encompasses exceeds a limit (threshold decided analytically). To extract the isolated pixels from a window, we did a single convolution pass of the following kernel:

1	1	1
1	-1	1
1	1	1

followed by a comparison operation.
To avoid any confusion, we state it explicitly that a window can have *noise* as well as *edge*, and noise and edges are not mutually exclusive. Depending on the flag, a particular window is accordingly processed in the subsequent steps.

IV. RESULTS & DISCUSSION

A. EVALUATION METRICS USED

To evaluate various edge detection algorithms, the following metrics have been used based on the Precision-Recall framework [18]:

- *True-Positive (TP)*: The count of TP denotes the number of correctly detected edge-pixels. Higher the TP count, the better the edge detector.
- *F1 score* [18]: $F1$ score is a metric which in turn depends upon two metrics—Precision (P) and Recall (R).

$$Precision = \frac{TP}{TP + FP} \quad Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1\,\text{score} = 2\frac{P.R}{P + R} \tag{4}$$

where FP denotes the False Positive, and FN denotes the False Negative detected pixel counts. For a given detection algorithm, Precision (P) measures how many of the detected edge pixels are correct. Recall (R) measures how many of the true edge pixels are actually detected by the algorithm. Precision and Recall trade off against one another, filling up the demerits of each other; and $F1$ score, being the harmonic mean of Precision and Recall, balances both Precision and Recall by weighing them equally. Higher the $F1$ score, the better the edge detector.

- *Optimal Dataset Scale (ODS)* [18]: A deep learning-based edge detector generally outputs a probability edge map, which needs to be thresholded to obtain the binary edge map. For the case of ODS, a fixed threshold is applied to the complete dataset, and the corresponding $F1$ score is recorded. By applying various thresholds in this sense, various $F1$ scores are obtained. Reporting the ODS measure refers to reporting the maximum value of the obtained $F1$ scores. Higher the ODS value, the better the edge detector.
- *Optimal Image Scale (OIS)* [18]: Instead of applying a fixed threshold to the complete dataset, OIS applies different thresholds to different images (that suits them best). Reporting the OIS means reporting the aggregate of maximum $F1$ scores obtained per image basis. Higher the OIS value, the better the edge detector.
- *Average Precision (AP)* [18]: Average Precision is the mean Precision weighted by the increase in Recall as the threshold is varied. In figure 12 and 13, it simply denotes the area under the P-R curve. Higher the AP value, the better the edge detector.
- *Detection Common Rate (DCR)* [39]: DCR between two edge maps A and B gives a normalized measure of the common edge pixels between them. For the case where the edge map B is the ground truth edge map, it boils down to the popular evaluation metric *Recall* (Equation 3)

In addition to above metrics, the P-R graph has also been plotted for BSDS500 dataset [18] and BIPED dataset [20] to provide the quantitative analysis. Visual comparison results are provided for qualitative analysis.

B. RESULTS WITH PRISTINE IMAGE

Although the algorithm was designed keeping in mind the general case of noisy image, it is very important for the algorithm to work well with the pristine set also. Figure 7 presents the results with pristine images. As can be visually seen, a lot more edges are present with the proposed algorithm as compared to that with CED. TruePositive (TP) & $F1$ score improvements further strengthen our visual claims. These results correspond to a particular (though general) set of threshold parameters, and the results of both the algorithms will improve on decreasing the threshold (Subsection IV-D).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

Figure 7. Results with pristine images with a particular threshold set. Top to bottom—Pristine Image, CED, Proposed edge detector (high-resolution weblink)

C. TWO VERSIONS OF THE PROPOSED SR-TW-CED

One shoe doesn't fit all. How useful an algorithm is, depends on the need of the application at hand. To increase the usefulness of our work, we present two versions of our algorithm:

1) Version 1 (Default): *Edges* preferred over *Low noise*
2) Version 2: *Low noise* preferred over *Edges*

Version 1 is used throughout this paper unless otherwise stated; it prioritizes *Edges* over *Low noise*. We also propose Version 2 for the case when a lower noise is preferred, even at the cost of losing some edges. For this case, we only output the *Edge Content*, ignoring the noise present in the image. This also results in lower computations. Comparing visual results of Version 1 with that of Canny in figure 8, more edges can be seen in 2nd & 4th *Capsicum*, and 3rd *Capsicum* shows far less noise. The increase in TruePositive (edge pixels) and $F1$ score is also significant (approx **twice**). Comparing Version 2 with that of Version 1, the 3rd Capsicum shows further less noise, though some edges can be seen missing at some places. Result of Version 2 is still better than that of Canny, both qualitatively and quantitatively (approx 50% improvement). All these algorithms are evaluated for the same set of input parameters.

D. EFFECT OF CHANGING INPUT THRESHOLD

In all the results shown till now in this paper, both the algorithms—Canny and the proposed SR-TW-CED are fed with a general but single input parameter set. This section presents how the results change when the input parameter

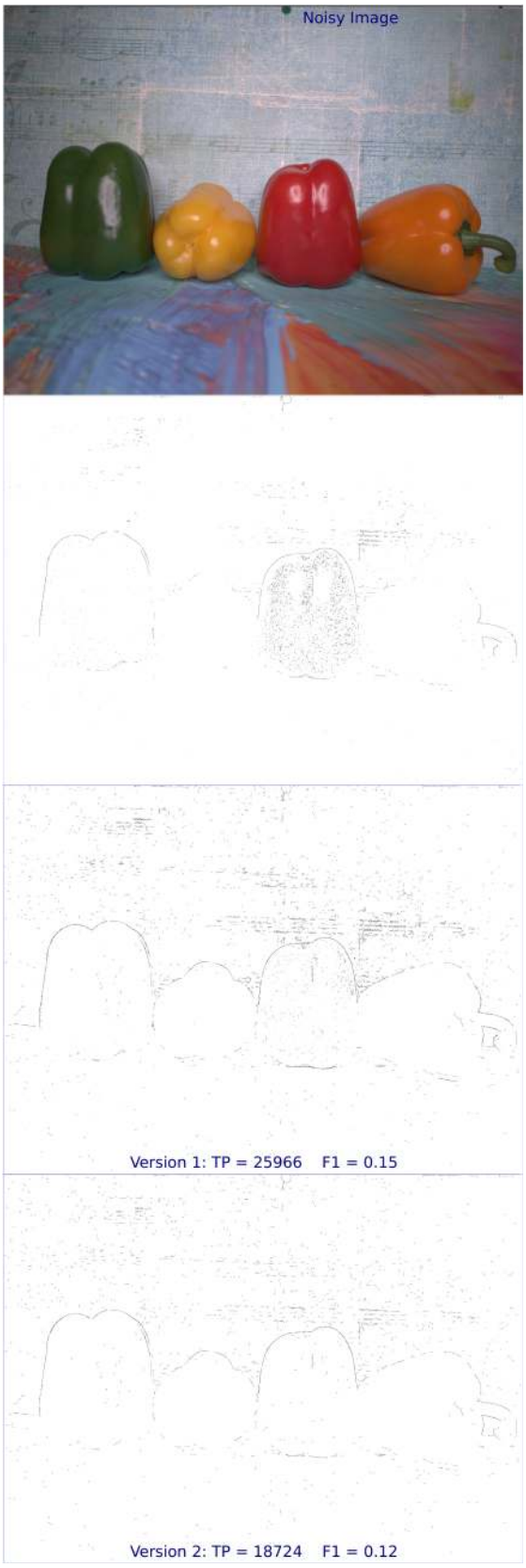

Figure 8. Two Proposed Versions (high-resolution weblink)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

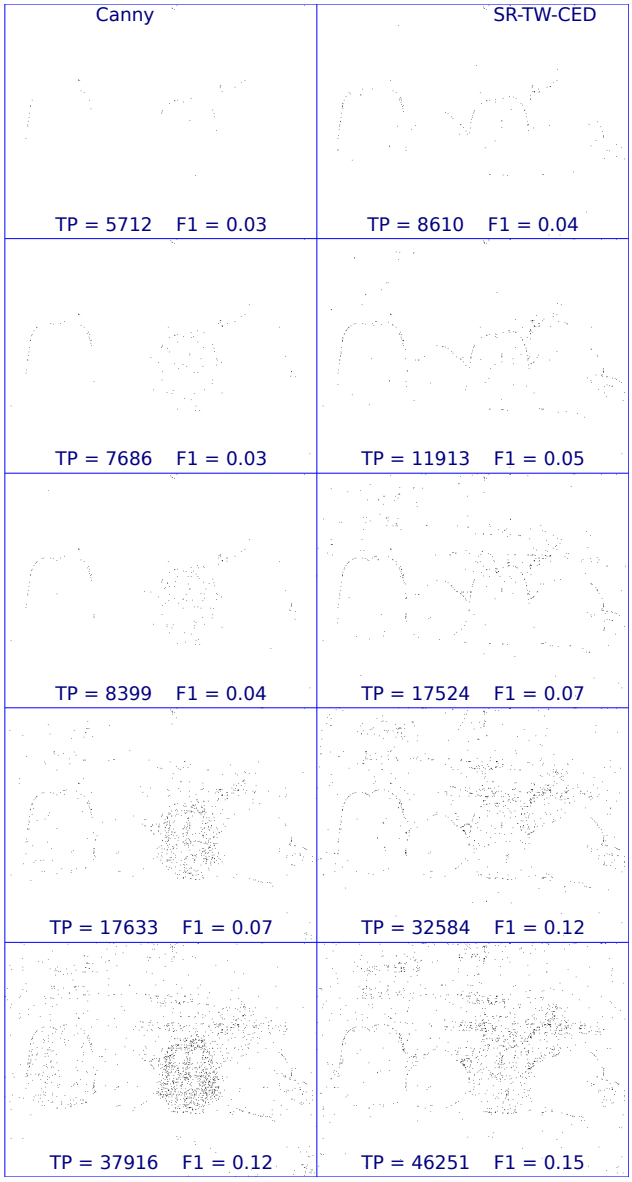

Figure 9. Result with decreasing input threshold (top to bottom)

set is varied. Figure 9 shows the visual comparative results of Canny (on the left) & proposed SR-TW-CED (on the right) with decreasing threshold from top to bottom, for the image scene shown in Figure 8. In all these images, the image on the right always has more edges (specifically 2nd & 4th Capsicum) and less noise (specifically 1st & 3rd Capsicum). In case this is not properly visible in the printed/ full-screen resolution, the reader may zoom and confirm that the black pixels above the Capsicum are actually the horizontal lines and edges present in the texture in the background of the input image (Figure 8). TruePositive & $F1$ score improvement from left to right also fortify our visual claims in all these images. The authors would also like to bring to attention, the case of extreme values of the *input parameter set*. If we further increase the thresholds, the edge map would move towards all-white i.e. losing all edges; and if we further decrease the thresholds, the edge map would move towards all-black i.e. overpowered by noise. Both the cases of very high & very low input thresholds result in less usable edge maps and are mostly avoided in practice.

E. COMPARISON WITH OTHER ALGORITHMS

Figure 10 and 11 present results with nine diverse-sized images (size written on top) from diverse datasets like SIDD+ [6], PolyU [40], BSDS500 [18], & NIND [41]. The third test image (*Smiling Faces*) in Figure 10, and first test image (*Building1*) in Figure 11 are smaller in size, and thus, their results are observable without much zooming-in. In these Figures, we compared our results with the conventional Canny [3], PLGFM [25], SMED [12], SE [21] , & HED [1]. As SE & HED gives thick grayscale edges, we processed them further with NMS [3] and thresholding to obtain thin binary edges. The PLGFM [25] gives thick binary edges and it does not suggest thinning of the edges. The proposed algorithm shows promising results, both qualitatively & quantitatively. Quantitatively, the 'TruePositive' , '$F1$ score' , & 'DCR' are mentioned below each image in figure 10 & 11. The proposed algorithm performs better than Canny and SE in all the nine images. Comparing with HED, the proposed algorithm performs better in all the nine images except in *Smiling Faces* image, where the HED has a better $F1$ score. Comparing with PLGFM and SMED, the proposed algorithm performs better in most of the images but it lacks in TP and DCR for some images. While the 'TruePositive' , '$F1$ score' , & 'DCR' are mentioned below each image, the visual results are far more suggestive here than the quantitative scores due to the following reasons:

- The nature of thresholds used (as input parameters) is different in these algorithms, and thus each cannot be optimally tuned uniformly. E.g. the deep learning-based algorithm (HED) uses probability-based threshold, while CED uses amplitude-based thresholds.
- For diverse datasets, it is difficult to have a uniform ground truth.
- Some datasets provide the ground truth boundary map; others need to be evaluated using some methods [42] (which can be arguable).
- The quality of ground truth data available with different datasets is different.

It can be seen in the third column of Figure 10 (i.e. the results of the image *Smiling Faces*) that the proposed algorithm maintains a good balance of extracting the texture (from the cloths) as well as the facial features and rejecting the noisy background from behind. Further, it can be seen in the first column of Figure 11 (i.e the results of the image *Building1*) that almost all the algorithms struggle with either the broken edge problem or the noisy structure problem, whereas the proposed algorithm has mitigated both of these problems simultaneously. Similar observations can be made with other images. PLGFM provides thick edges, and is able

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

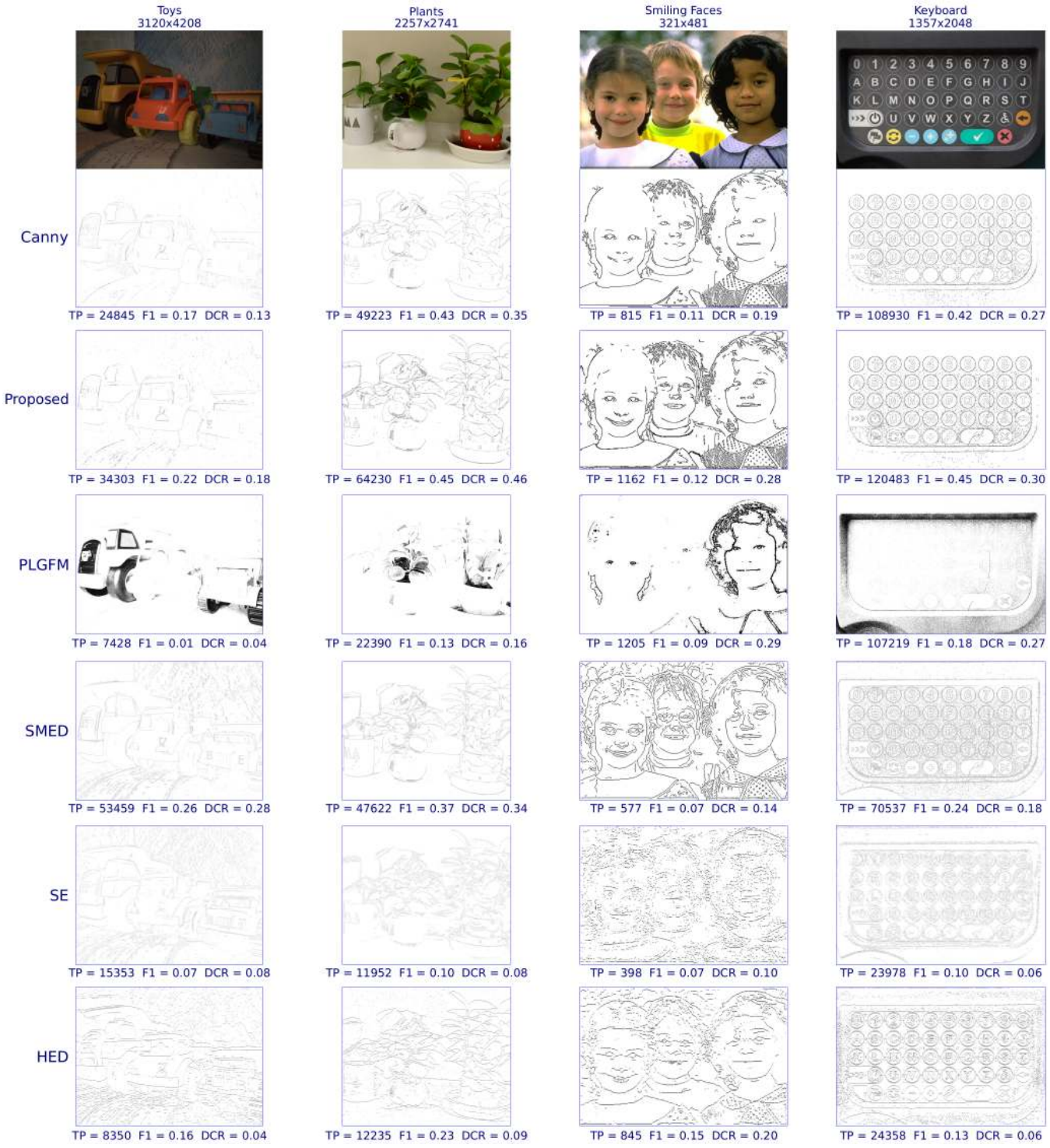

Figure 10. Algorithm comparisons (high-resolution weblink)

to extract edges from low-illuminated areas as well, but it too suffers from broken edges and noisy structures. Please note that the results obtained for SE for the last three images in figure 11 contain very distant detected pixels and as a result appear very faint in this resolution.

Usually the edges are one pixel wide, and the edge map is binary (Edge or Not Edge) [3][18]. Nevertheless, if the application in hand allows wider and grayscale edges, SE and HED results would look better than what is seen in Figure 10 & 11.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

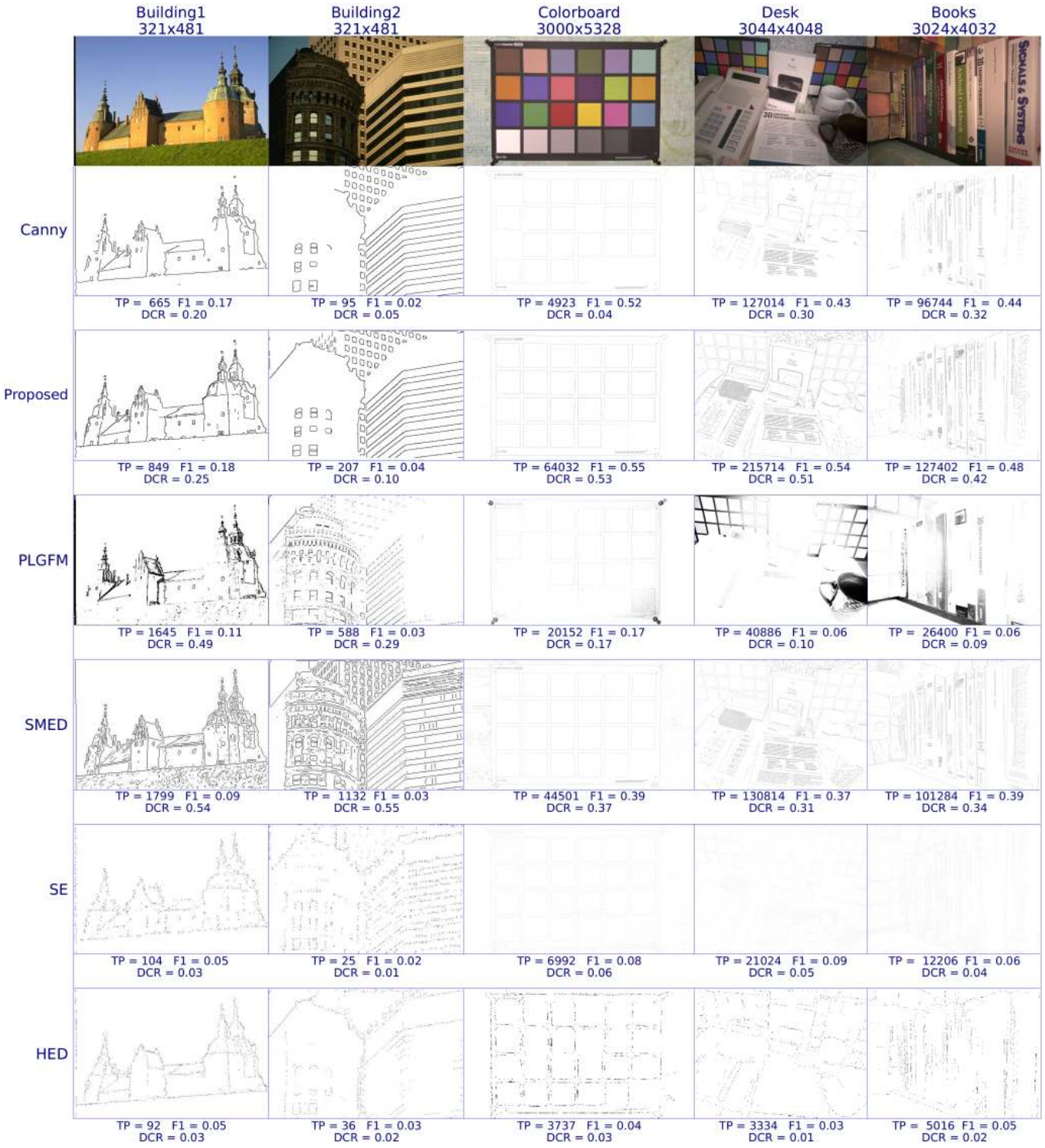

Figure 11. Algorithm comparisons (high-resolution weblink)

F. BSDS500 & BIPED BENCHMARKING

BSDS500 [18] is a contour detection benchmark that uses a precision-recall (PR) framework along with three metrics—ODS, OIS, & AP to evaluate various contour detection algorithms. Similarly, BIPED [20] is a recent edge-detection benchmarking dataset. Table 1 & Table 2 show that our algorithm has better ODS & OIS than that of Canny [3]; and Figure 12 & Figure 13 show that the curve of our algorithm is higher than that of Canny [3], and thus exhibits improvements over Canny [3] in almost all the practical range of the input threshold. Our algorithm has a lower

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3145428, IEEE Access

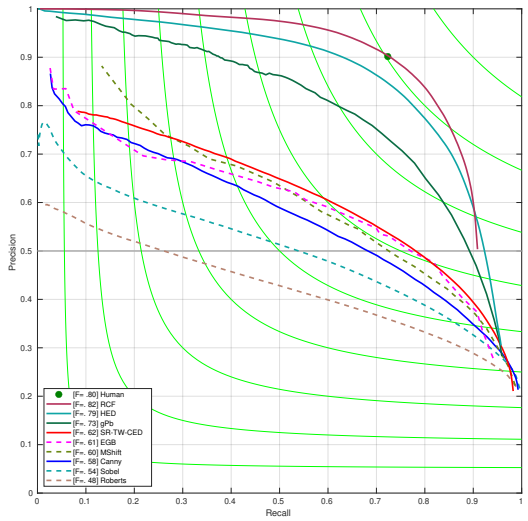

Figure 12. BSDS500 benchmarking

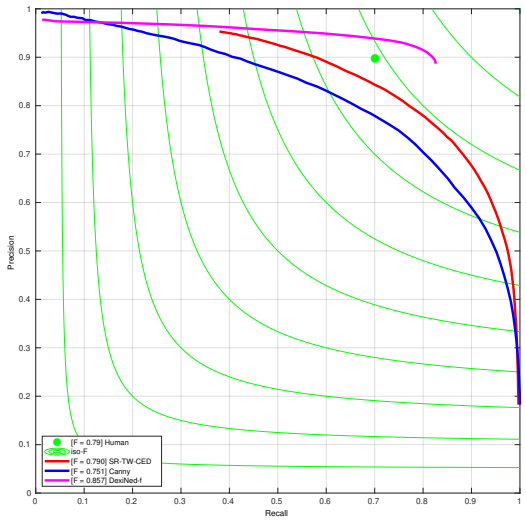

Figure 13. BIPED benchmarking

Table 1. BSDS500 benchmarking

Method	ODS	OIS	AP
RCN [43]	0.824	0.839	0.837
RCF [2]	0.819	0.836	0.846
HED [1]	0.788	0.808	0.840
DexiNed [20]	0.729	0.745	0.583
gPb [18]	0.70	0.72	0.66
SR-TW-CED	0.62	0.64	0.55
EGB [44]	0.614	0.658	0.564
Canny [3]	0.60	0.63	0.58
MShift [45]	0.598	0.645	0.497
Felz-Hutt [46]	0.58	0.62	0.53
Sobel [9]	0.539	0.575	0.498
Roberts [7]	0.483	0.513	0.413

Table 2. BIPED benchmarking

Method	ODS	OIS	AP
DexiNed [20]	0.859	0.867	0.905
RCF [2]	0.843	0.859	0.882
HED [1]	0.829	0.847	0.869
SR-TW-CED	0.790	0.814	0.504
SED [47]	0.717	0.731	0.756
Canny [3]	0.751	0.776	0.818

AP score (area under the curve) because it does not span the lower recall values for the thresholds provided in the benchmarking datasets. The low-recall and high-precision area, i.e., the top-left part of the P-R graph, corresponds to the edge-map having fewer edge pixels. The proposed algorithm identifies this as a broken edge problem. To mitigate this problem, the proposed algorithm lowers the threshold, gets additional edge pixels and thus increases the recall. As per the BSDS500 [18] (Table 1, Figure 12) & BIPED [20] (Table 2, Figure 13) benchmarking, though our algorithm is an improvement over Canny [3], it still underperforms w.r.t. some of the state-of-the-art deep learning (DL)-based algorithms. On BIPED [20] benchmarking, our algorithm gives the human-level performance ($F1$ score .79), which is appreciable considering the fact that it is a non-DL–based algorithm. It is particularly interesting to note the observations in the third column (*Smiling Faces*) of Figure 10, where HED [1] being one of the high performing algorithms, gives a better $F1$ score than ours but lacks in visual quality when the thin-edge version is considered (Their thick-grayscale-edge version looks remarkably better than their thin-binary-edge version).

It is interesting to note the primary differences, both weaknesses and advantages of the proposed non-DL–based approach with conventional deep learning-based methods. The proposed algorithm is a gradient-based approach that detects edge-pixels exactly at those locations where the gradient changes, and therefore produces a pixel-level accurate thin binary edge map. However, a general DL-based algorithm (such as SE and HED) typically produce a thick grayscale edge map, and are therefore not implicitly pixel-level accurate. These algorithms subsequently obtain thin binary edges by post-processing (such as non max suppression and thresholding) unrelated to the DL process. These algorithms are also evaluated in a way that does not penalize the lack of pixel-level accuracy. For example, BSDS500, one of the most popular benchmarking dataset utilizes the concept of corresponding pixel (implemented through *correspondPixels* source file [18]), where the two edge pixels from two edge maps are considered to be matching even if they are certain distance apart (this distance is decided by the *maxDist* parameter). Nevertheless, more recent DL-based works, such as DexiNed [20], do implicitly produce thin maps through an intricate and advanced learning process using a large dataset and outperform the proposed algorithm

(as shown in Table 1 & 2). DL-based edge detectors also consume significant computational resources (GPU, RAM, memory bandwidth, etc.) to train their networks, and require a GPU to find the edge map even for a single test image. On the contrary, the proposed algorithm is not intensive on computational resources and can easily be run on older generation CPUs. The proposed algorithm presents two diverse versions (one preferring more edges, another preferring lower noise). Such an easy customization is usually not available with DL-based algorithms, and one would need to recreate the whole dataset and retrain the network even for a minor change in desired application. Further, DL-based algorithms typically have a tendency to overfit as per the training dataset and the results could vary when tested on new images (although this generalization is improving with time). Since the logic of the proposed algorithm is built from root up, each step, such as calculations involved, decision graph followed, procedure of content extraction, etc. is defined and clearly known to the user. Therefore, the output can be obtained/predicted almost deterministically. On the other hand, a DL-based network is more like a black-box, and the exact process of learning within the network still requires significantly more clarity even in the current state of the art. In terms of non-ML– and attribute-based edge detection approaches, the proposed algorithm shows noteworthy potential in comparison with the state of the art.

V. CONCLUSION AND FUTURE WORK

An enhanced Canny Edge Detector, which takes the same input parameter set as that of the Canny, and produces the edge map with better-connected edges and reduced noise has been presented and discussed. A detailed analysis of CED from the SR point-of-view is also presented. The proposed algorithm is a significant improvement over the CED in almost all the practical ranges of the input threshold. The proposed algorithm is designed by modifying the core of the Canny Edge Detector [3] without any pre/post-processing. However, this non-inclusion of any pre/post-processing has created some limitations in the performance of our algorithm. The proposed algorithm needs to be provided with a threshold as the input (as in CED [3]); it can not work without any input threshold. This limitation can be overcome by using the pre-processing step of an automatic threshold module, like Otsu's threshold method [48] which maximizes inter-class variance or minimizes intra-class variance, or a look-up table [14] computed based on the estimated noise, or a 2-D entropy based automatic threshold calculator [17], etc. Noise in the output edge map can be further reduced by using a denoising module as a post-processing step. Another limitation that the proposed algorithm has (like most other Edge Detectors) is that it is not invariant to illumination; It can be overcome by developing a method incorporating the learnings from PLGFM [25] (PLGFM is an illumination-invariant edge detector). The authors have tried to contribute to the knowledge base of Edge Detection by proposing an improved edge detector. On BIPED [20] benchmarking dataset, the proposed algorithm performs at par with the human level performance ($F1$ score .79), which is appreciable considering the fact that it is a non-DL–based algorithm. The future work includes addressing the limitations of the proposed algorithm and designing a deep learning network with better edge detection capabilities.

ACKNOWLEDGMENT

The authors would like to thank Department of Science & Technology, Government of India, for funding the project titled *Noise-enhanced edge-preserving image denoising using stochastic resonance* (#ECR/2016/001606). The authors would also like to thank Digital India Corporation for its funding through the Visvesvaraya PhD Scheme to the research scholar.

References

[1] Saining Xie and Zhuowen Tu. "Holistically-nested edge detection". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1395–1403.

[2] Yun Liu et al. "Richer convolutional features for edge detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 3000–3009.

[3] John Canny. "A computational approach to edge detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6 (1986), pp. 679–698.

[4] *OpenCV Canny Edge Detector documentation*. 2021. URL: https://docs.opencv.org/master/dd/d1a/group__imgproc__feature.html#ga04723e007ed888ddf11d9ba04e2232de.

[5] *Matlab Canny Edge Detector documentation*. 2021. URL: https://in.mathworks.com/help/images/ref/edge.html?s_tid=srchtitle#d122e65947.

[6] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. "A high-quality denoising dataset for smartphone cameras". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1692–1700.

[7] LG Roberts. "Machine Perception of three-dimensional solids, in Optical and Electro-optical". In: *Formation Processing* (1965).

[8] Judith MS Prewitt. "Object enhancement and extraction". In: *Picture Processing and Psychopictorics* 10.1 (1970), pp. 15–19.

[9] IE Sobel. "Camera Models and Machine Perception. Ph. D. thesis. Palo Alto, Calif.: Stanford University, 1970". In: (1970).

[10] Russell A Kirsch. "Computer determination of the constituent structure of biological images". In: *Computers and Biomedical Research* 4.3 (1971), pp. 315–328.

[11] David Marr and Ellen Hildreth. "Theory of edge detection". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207.1167 (1980), pp. 187–217.

[12] Paul Bao, Lei Zhang, and Xiaolin Wu. "Canny edge detection enhancement by scale multiplication". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.9 (2005), pp. 1485–1490.

[13] Qian Xu et al. "A distributed canny edge detector: algorithm and FPGA implementation". In: *IEEE Transactions on Image Processing* 23.7 (2014), pp. 2944–2960.

[14] Mahdi Kalbasi and Hooman Nikmehr. "Noise-Robust, Reconfigurable Canny Edge Detection and its Hardware Realization". In: *IEEE Access* 8 (2020), pp. 39934–39945.

[15] Cihan Topal and Cuneyt Akinlar. "Edge drawing: a combined real-time edge and segment detector". In: *Journal of Visual Communication and Image Representation* 23.6 (2012), pp. 862–872.

[16] Cuneyt Akinlar and Edward Chome. "PEL: a predictive edge linking algorithm". In: *Journal of Visual Communication and Image Representation* 36 (2016), pp. 159–171.

[17] Yang Liu, Zongwu Xie, and Hong Liu. "An Adaptive and Robust Edge Detection Method Based on Edge Proportion Statistics". In: *IEEE Transactions on Image Processing* 29 (2020), pp. 5206–5215.

[18] Pablo Arbelaez et al. "Contour detection and hierarchical image segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.5 (2010), pp. 898–916.

[19] Nathan Silberman et al. "Indoor segmentation and support inference from rgbd images". In: *European Conference on Computer Vision*. Springer. 2012, pp. 746–760.

[20] Xavier Soria Poma, Edgar Riba, and Angel Sappa. "Dense extreme inception network: Towards a robust cnn model for edge detection". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 1923–1932.

[21] Piotr Dollár and C Lawrence Zitnick. "Fast edge detection using structured forests". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.8 (2014), pp. 1558–1570.

[22] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[23] Jianzhong He et al. "Bi-directional cascade network for perceptual edge detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3828–3837.

[24] François Chollet. "Xception: Deep learning with depthwise separable convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1251–1258.

[25] Debotosh Bhattacharjee and Hiranmoy Roy. "Pattern of local gravitational force (PLGF): A novel local image descriptor". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.2 (2019), pp. 595–607.

[26] Mengtian Li et al. "Photo-sketching: Inferring contour drawings from images". In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2019, pp. 1403–1412.

[27] Luca Gammaitoni et al. "Stochastic resonance". In: *Reviews of Modern Physics* 70.1 (1998), p. 223.

[28] Adi R Bulsara. "Tuning in to noise". In: *Physics today* 49.3 (1996), pp. 39–45.

[29] Gregory Peter Harmer, Bruce R Davis, and Derek Abbott. "A review of stochastic resonance: Circuits and measurement". In: *IEEE Transactions on Instrumentation and Measurement* 51.2 (2002), pp. 299–309.

[30] Luca Gammaitoni. "Stochastic resonance and the dithering effect in threshold physical systems". In: *Physical Review E* 52.5 (1995), p. 4691.

[31] M-O Hongler et al. "The resonant retina: exploiting vibration noise to optimally detect edges in an image". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25.9 (2003), pp. 1051–1062.

[32] Rajlaxmi Chouhan, Rajib Kumar Jha, and Prabir Kumar Biswas. "Enhancement of dark and low-contrast images using dynamic stochastic resonance". In: *IET Image Processing* 7.2 (2013), pp. 174–184.

[33] CS Asha et al. "Optimized Dynamic Stochastic Resonance framework for enhancement of structural details of satellite images". In: *Remote Sensing Applications: Society and Environment* 20 (2020), p. 100415.

[34] VP Subramanyam Rallabandi and Prasun Kumar Roy. "Magnetic resonance image enhancement using stochastic resonance in Fourier domain". In: *Magnetic Resonance Imaging* 28.9 (2010), pp. 1361–1373.

[35] D. Dhillon and R. Chouhan. "Noise-aided Edge-preserving Image Denoising using Non-Local Means with Stochastic Resonance". In: *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. 2018, pp. 21–25.

[36] Xuefeng Liu et al. "Classification of hyperspectral image by CNN based on shadow area enhancement through dynamic stochastic resonance". In: *IEEE Access* 7 (2019), pp. 134862–134870.

[37] Munendra Singh, Ashish Verma, and Neeraj Sharma. "Optimized multistable stochastic resonance for the enhancement of pituitary microadenoma in MRI". In: *IEEE Journal of Biomedical and Health Informatics* 22.3 (2017), pp. 862–873.

[38] Renbin Peng and Pramod K Varshney. *Methods of improving detectors and classifiers using optimized stochastic resonance noise*. US Patent 9,026,404. May 2015.

[39] Animesh Sengupta et al. "Edge information based image fusion metrics using fractional order differentiation and sigmoidal functions". In: *IEEE Access* 8 (2020), pp. 88385–88398.

[40] Jun Xu et al. "Real-world noisy image denoising: A new benchmark". In: *arXiv preprint arXiv:1804.02603* (2018).

[41] Benoit Brummer and Christophe De Vleeschouwer. "Natural image noise dataset". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 0–0.

[42] Rafael Medina-Carnicer et al. "A novel method to look for the hysteresis thresholds for the Canny edge detector". In: *Pattern Recognition* 44.6 (2011), pp. 1201–1211.

[43] André Peter Kelm, Vijesh Soorya Rao, and Udo Zölzer. "Object contour and edge detection with refinecontournet". In: *International Conference on Computer Analysis of Images and Patterns*. Springer. 2019, pp. 246–258.

[44] Pedro F Felzenszwalb and Daniel P Huttenlocher. "Efficient graph-based image segmentation". In: *International Journal of Computer Vision* 59.2 (2004), pp. 167–181.

[45] Dorin Comaniciu and Peter Meer. "Mean shift: A robust approach toward feature space analysis". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.5 (2002), pp. 603–619.

[46] Pedro F Felzenszwalb and Daniel P Huttenlocher. "Efficient graph-based image segmentation". In: *International Journal of Computer Vision* 59.2 (2004), pp. 167–181.

[47] Arash Akbarinia and C Alejandro Parraga. "Feedback and surround modulated boundary detection". In: *International Journal of Computer Vision* 126.12 (2018), pp. 1367–1380.

[48] Nobuyuki Otsu. "A threshold selection method from gray-level histograms". In: *IEEE transactions on systems, man, and cybernetics* 9.1 (1979), pp. 62–66.

DEEPAK DHILLON (GSM'14) is a Ph.D. Student in the Department of Electrical Engineering at IIT Jodhpur (India) under the Visvesvaraya Ph.D. Fellowship, and is pursuing research in the area of image processing. He completed his Bachelor's degree from Panjab University, Chandigarh in 2011, and his Master's degree from Thapar University, Patiala, in 2015.

During 2011-12, Deepak worked as a Software Analyst at Newgen Software Technologies Ltd., New Delhi. He has been an active volunteer of the IEEE Student Branch of IIT Jodhpur and served as Vice-Chair (2017-20). He is also a member of the Rotaract Club of IIT Jodhpur for two years. He received the IEEE Signal Processing Society Travel Grant for presenting his paper in GlobalSIP 2018 in Anaheim, CA, US.

RAJLAXMI CHOUHAN (SM'21) received her Ph.D. in August 2015 from the Department of Electronics and Electrical Communication Engineering at the Indian Institute of Technology (IIT) Kharagpur (India). Dr. Chouhan received her Bachelor's degree in Electronics and Communication Engineering from Jabalpur Engineering College in 2009, and her Master's degree from IIITDM Jabalpur in December 2011.

She is currently an Assistant Professor in the Department of Electrical Engineering at IIT Jodhpur. Her research interests include image quality assessment, noise-aided image processing, education technology, digital learning tools and applications. She served as the Chair of IEEE Women in Engineering Affinity Group of Kharagpur Section in 2014, and currently serves as an advisor of the IEEE WIE Student Chapter Affinity Group of IIT Jodhpur.