IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 29, 2020, accepted February 16, 2020, date of publication February 27, 2020, date of current version March 6, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2976860

Noise-Robust, Reconfigurable Canny Edge Detection and Its Hardware Realization

MAHDI KALBASI[1,2] **AND HOOMAN NIKMEHR**[1,3]

[1]Department of Computer Architecture, University of Isfahan, Isfahan 8174673441, Iran
[2]Department of Electrical and Computer Engineering, Golpayegan University of Technology, Isfahan 8771765651, Iran
[3]School of Electrical and Electronic Engineering, The University of Adelaide, Adelaide, SA 5005, Australia

Corresponding author: Mahdi Kalbasi (kalbasi@gut.ac.ir)

ABSTRACT Edge detection is a common operation in image/video processing applications. Canny edge detection, which performs well in different conditions, is one of the most popular and widely used of these algorithms. Canny's superior performance is due mainly to its provision of the ability to adjust the output quality by manipulating the edge detection parameters, Sigma and Threshold. Calculating values for these two parameters on-the-fly and based on the application's circumstances requires additional preprocessing, which increases the algorithm's computational complexity. To reduce the complexity, several proposed methods simply employ precalculated, fixed values for the Canny parameters (based on either the worst or typical conditions), which sacrifices the edge detection's performance in favor of the computational complexity. In this paper, an adaptive parameter selection method is proposed that selects values for the Canny parameters from a configuration table (rather than calculating in run-time), based on the estimated noise intensity of the input image and the minimum output performance that can satisfy the application requirements. This adaptive implementation of the Canny algorithm ensures that, while the edge detection performance (noise robustness) is higher than state-of-the-art counterparts in different circumstances, the execution time of the proposed Canny remains lower than those of recent cutting-edge Canny realizations.

INDEX TERMS Adaptive systems, Gaussian noise, computational complexity, image edge detection, reconfigurable architectures.

I. INTRODUCTION

Realtime video and image processing are used in a wide range of industrial, medical, consumer electronics and embedded device applications. These applications typically demonstrate an increasing demand for computing power. Therefore, there is a need for high-throughput implementations of image processing algorithms that can perform time consuming computations on considerable amounts of data. Those implementations often need to meet real-time requirements as well.

Edge detection is a fundamental pre-processing step in many machine vision and image processing algorithms such as image segmentation, image enhancement, tracking, and coding [1], [2], exploited to find sharp discontinuities in the image, where the image brightness or intensity changes suddenly. The embedded nature of most recent edge detection applications necessitates developing real-time and noise-robust algorithms. Edge detection has been researched extensively and many such algorithms have been proposed [3]–[5]. Among them, the Canny algorithm [6] is still considered as the standard and basis for many efficient solutions due to its reliable performance, especially in noisy environments, when compared to other similar techniques.

The Canny algorithm applies a Gaussian filter for image smoothing and noise suppression and then filters out low-gradient edge pixels (caused by noise) based on a hysteresis thresholding method. The performance and computational complexity of the algorithm are strongly influenced by its parameters, Sigma and Threshold (TH) [6].

There is a vast number of works on manipulating Sigma and Threshold in order to improve the output quality and/or to decrease the calculation load of Canny edge detection. In one end of the spectrum, there are approaches which calculate and set the parameters' values dynamically for every input image, based on the image characteristics (such as the gradient magnitude histogram) [7]–[9]. This technique, which is referred to as "variable-threshold" in this paper (since there is no research reporting on dynamically calculating the Sigma value), improves the edge detection performance, however, imposes a very high computational complexity. In contrast,

The associate editor coordinating the review of this manuscript and approving it for publication was Sudhakar Radhakrishnan.

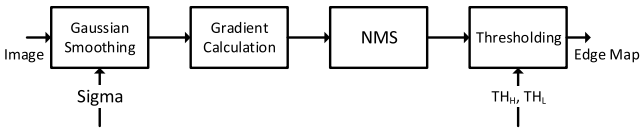


FIGURE 1. Original (conventional) Canny edge detection.

on the other end, to tackle the computation workload problem, there are researchers who use fixed values for the Canny parameters [10]–[12]. While this approach confines the algorithm complexity, it diminishes the performance specifically when environmental conditions and application requirements constantly change.

This paper proposes a new approach which neither calculates dynamically nor assigns fixed values to the Canny parameters, Sigma and TH. In fact it exploits a configuration table with precalculated entries that adaptively provides values for the algorithm parameters based on the environmental conditions and application requirements. Experimental results confirm that the new implementation of the Canny algorithm leads to improved noise robustness and less execution time over the conventional Canny realizations.

The rest of this paper is organized as follows. Section II briefly describes the classic Canny algorithm. In Section III, the new adaptive Canny edge detection algorithm is presented. Experimental results are shown and discussed in Section IV, followed by a Hardware implementation and its evaluation in Section V. Section VI concludes the paper.

II. CANNY EDGE DETECTION ALGORITHM

The Canny is a multi-stage image edge detection algorithm introduced for the first time in [6]. The algorithm aims to satisfy the following three criteria:

1) Edges need to be detected with a low error rate.
2) Detected edges should be localized as close as possible to the real edges (i.e. good localization).
3) Edges have to be marked only once (i.e. minimal response).

As shown in Fig. 1, edge detection based on the conventional Canny algorithm consists of the following operational blocks.

- **Smoothing**: Using the Gaussian filter, the input image is smoothed to remove noise.
- **Gradient calculation**: Legitimate edges can be found where the image gradients have large magnitudes. The magnitudes can be obtained by convolving the image with the gradient masks.
- **Non-Maximal Suppression (NMS)**: To sharpen the edge features, all values along the gradient line, except for the local maxima, must be suppressed.
- **Thresholding**: In the original Canny algorithm, two hysteresis thresholds, namely TH_H and TH_L, exist. If the pixel's gradient is found greater than TH_H, it is considered as a strong edge. However, if the gradient is smaller than TH_L, the corresponding pixel need to be discarded. In other cases (the gradient is between TH_H and TH_L), the pixel is marked as a weak edge (from now on, in this paper, the term "Threshold" refers to the pair of TH_H and TH_L). A weak edge can turn into a strong edge if it is found connected to another strong edge.

While in general, both Sigma and Threshold can affect the Canny algorithm's sensitivity to noise, its ability to detect fine details, and the amount of localization error in detected edges, in particular, Sigma influences both the algorithm's performance and its computational complexity.

On the other hand, since the Gaussian smoothing stage is implemented using a 2D convolver, the computational complexity of this stage can be expressed proportional to the convolution kernel size. For example, Kumar [13] reports that increasing the kernel size from three to seven, results in about five times increase in the convolver's cycles per pixel (or equivalently CPI). Therefore, given that Gaussian smoothing is a computation intensive operation, appropriate selection of the convolution kernel size can significantly affect the Canny convolver's performance results.

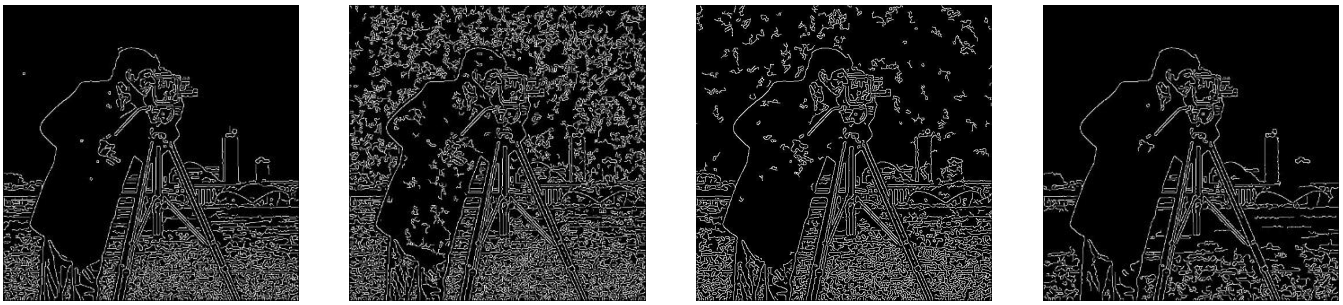
III. ROBUST EDGE DETECTION BASED ON CANNY ALGORITHM

Selecting appropriate values for Sigma and Threshold is one of the most critical decisions to make in Canny edge detection since they directly impact the performance and computational complexity of the algorithm. In this section, their effects are studied and discussed, and then Adaptive Parameter Selection (APS), a new approach to implementing Canny edge detection, is introduced.

A. CANNY PARAMETERS AND PERFORMANCE

Many Canny implementations reported in the literature use fixed values for Sigma and Threshold for every input image. These implementations, which can be categorized as "fixed-parameter" Canny, result in low-performance edge detection, particularly for systems dealing with noisy images. Fig. 2 illustrates edge maps of the Cameraman image obtained by applying the fixed-parameter approach. Fig. 2c, which represents the outcome of the original Canny edge detector for the noise-free image, is used as the basis for comparison. Figs. 2b, 2a and 2d demonstrate how the edge detector behaves when 20% of Gaussian noise is added to the input image. Fig. 2a is the edge map of the noisy image, as detected by the original Canny algorithm with the same parameters values as when processing the noise-free image, while Fig. 2b represents the edge map of the noisy image generated by the original Canny algorithm, which sets the Threshold value based on the gradient magnitude histogram of the image. Finally, Fig. 2d displays the result of applying the Canny algorithm to the image with an optimal parameter set (PS) value for Sigma and Threshold.

The importance of choosing appropriate values for Sigma and Threshold can be acknowledged by comparing the edge maps shown in Fig. 2. While using a fixed PS for processing



(a) Edge map of noise-free image

(b) Edge map of noisy image with fixed parameters

(c) Edge map of noisy image with new Threshold

(d) Edge map of noisy image with optimal parameters

FIGURE 2. Edge map of noise-free image.

the noisy image leads to a poor output, as demonstrated in Fig. 2b, Fig. 2c shows that changing the Threshold value in accordance with the input image conditions can improve the resulting edge maps. However, the best result (most similar to the edge maps obtained from the noise-free image; Fig. 2a), can be achieved using the optimal PS as displayed in Fig. 2d. Therefore, when the environment is noisy, updating the parameters with appropriate values can improve the edge detection performance.

The smoothing process is usually performed using a fixed predetermined Sigma value [14]–[19]. However, because of the direct correlation between the degree of smoothing and the Sigma value (this value itself corresponds to the standard deviation of the Gaussian filter), it may be reasonable to change the Sigma value if the input image specification changes. For example, having selected a small value for Sigma, in a high-noise environment, some noisy pixels may be incorrectly detected as edge pixels. Instead, a high value needs to be assigned to Sigma in this scenario to avoid misdiagnosing the edges, even though a number of legitimate edges are not detected. In contrast, in low-noise conditions, Sigma should take small values to provide good localization without undue loss of real edges.

B. CANNY PARAMETERS AND COMPUTATIONAL COMPLEXITY

The value assigned to Sigma affects not only the Canny algorithm's performance but also the Gaussian filter's computational complexity, which itself is proportional to the kernel size of a compute-intensive 2D convolver realizing the filter. Since the Gaussian kernel size is determined by the Sigma value, reducing the filter's computational complexity requires the selected Sigma value to be as small as possible.

Now, the main design challenge is to find the right value for Sigma, since it can lead to the best tradeoff between the Canny's computational complexity and its performance, especially in noisy environments (to reduce the computational complexity, the Sigma value must be minimized, whereas, it has to take a large value to improve the edge detection performance).

TABLE 1. Performance and computational complexity tradeoff (*: smallest Sigma value which satisfies "Performance ≥ MDP" must be selected.)

Noise \ Sigma	Low Complexity	Low Perf.	High Complexity	High Perf.
Low	Low	High	High	Low
High	Low	Low	High	High*

On the other hand, in practice, there are situations where the (image processing) application can tolerate some degree of performance degradation and, consequently, a smaller value can be assigned to Sigma to reduce the Canny's computational complexity. Having defined the MDP, short for "Minimum Desired Performance", as the lowest performance acceptable by the application, when the MDP is lower than the maximum possible performance deliverable by the algorithm, the Canny edge detector can work with a smaller Sigma value, which results in reduced computational complexity. This observation, summarized in Table 1, motivates the introduction of a new implementation of the Canny algorithm with a dynamic mechanism for assigning values to Sigma and Threshold, adaptive to input conditions and output requirements. This Canny edge detection is called "Adaptive Parameter Selection" (APS) in this paper.

C. PROPOSED ADAPTIVE PARAMETER SELECTION

The main idea of the presented APS Canny is to find the best values for Sigma and Threshold, based on the input quality (input image noise level), that can deliver output (edge maps) quality not less than the application's MDP, while keeping the algorithm's computational complexity as small as possible. Then the APS algorithm uses the noise intensity and given MDP to select the appropriate precalculated values for Sigma and Threshold. As demonstrated in Fig. 3, the proposed dynamic parameter selection approach requires some alterations to the original Canny algorithm as follows.

- **Configuration table**: A new configuration table operates in the proposed APS Canny. In each entry of this table, and for every combination of the noise intensity and MDP, a set of values for Sigma and Threshold

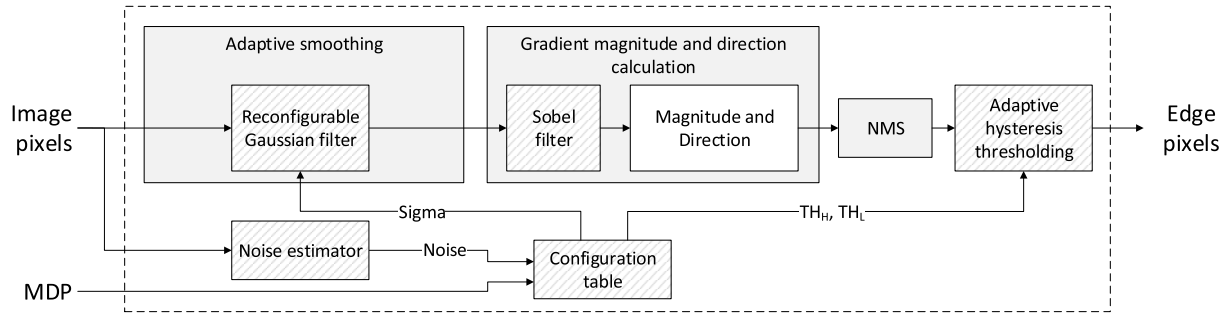


FIGURE 3. Proposed APS Canny edge detection. Image pixels, noise intensity and MDP are inputs and edge maps are output.

is stored, that can satisfy the application conditions and requirements. These values are precomputed offline (only once) based on extensive analyses using a standard benchmark image dataset. In this implementation, the noise level is evaluated by an estimator, and the MDP is either determined by the application user or precalculated using an offline profiler.

- **Noise estimator**: Based on the proposed adaptive approach, the noise intensity of the input image is roughly calculated using an estimator [20], [21] and then, along with the MDP, applied to the configuration table to select the right pair of Sigma and Threshold values.
- **Adaptive smoothing stage**: Unlike the original Canny algorithm with a simple fixed-kernel convolver, in the proposed APS implementation, to improve the noise robustness of Canny edge detection, the proposed smoothing stage now accommodates a reconfigurable 2D convolver. As shown in Fig. 3, based on the noise level estimated for the input image and MDP, a new Sigma value is fetched from the configuration table that determines the size and coefficients of the new kernel.
- **Adaptive thresholding stage**: To support the required adaptability of the proposed Canny edge detector, the traditional hysteresis thresholding stage is redesigned such that, instead of using fixed values (or even employing dynamically calculated comparison values, as presented in [8]), it can adapt itself to the circumstances, i.e. the comparators of this stage use the Threshold value retrieved from the configuration table as the edge pixel selection criterion.

The following section describes the proposed flexible, table-based APS Canny edge detection algorithm in detail while its computational complexity and hardware implementation are discussed in Section V.

D. CONFIGURATION TABLE

For each input image, the configuration table provides an appropriate PS for each combination of the image's estimated noise intensity and application's MDP. Using the selected PS (stored as a table entry) with the minimal calculation effort (computational complexity), the proposed APS Canny can provide the desired noise robustness, represented in the best possible edge map.

TABLE 2. Ranges of sigma and threshold values.

Parameter	From	To	Step	Total number
Sigma	0.1	2.7	0.2	14
TH_H	0.01	0.99	0.02	50
TH_L	0.004	0.369	0.008	50

To construct the configuration table, an enumeration method is used in this work to investigate the impact of Sigma and Threshold values on the performance of the Canny edge detection algorithm (output edge map quality) under a wide range of processing conditions. The study carried out in this research takes a wide range of Sigma and Threshold parameters (50 different values for each) into account, where $TH_H \in [0.01, 0.99]$, step 0.02 and $TH_L \in [0.004, 0.396]$, step 0.008 ($TH_L = 0.4 \times TH_H$ is a common assumption in most Canny implementations [22]).

Unlike most fixed-parameter Canny realizations, where the Sigma value is typically set to a fixed value of about 1 [8], [9], in the current study, $Sigma \in [0.1, 2.7]$. Observations made in this research reveal that while the selected range is large enough, any further increase in the Sigma value beyond 2.7 can severely reduce the edge detection performance. The selected ranges for the parameters, shown in Table 2, result in a total of 700 PS combinations which are investigated in the presented research to find the most appropriate PSs that lead to the best achievable edge maps (highest acceptable performance according to the MDP) with the smallest Sigma values (corresponding to the least computational complexity).

To evaluate the similarity of edge maps produced by different implementations of the Canny algorithm, P_{co}, i.e., the percentage of correctly detected edge pixels [23], is widely used as the quality metric in the existing literature [8], [9], [24]. According to this definition of P_{co}, it can be calculated as

$$P_{co} = C_{NC}/C_{OC} \tag{1}$$

where C_{NC} is the number of true edge pixels in a noisy image identified by the "New Canny", while C_{OC} is the total number of edge pixels of the clean image recognized by the

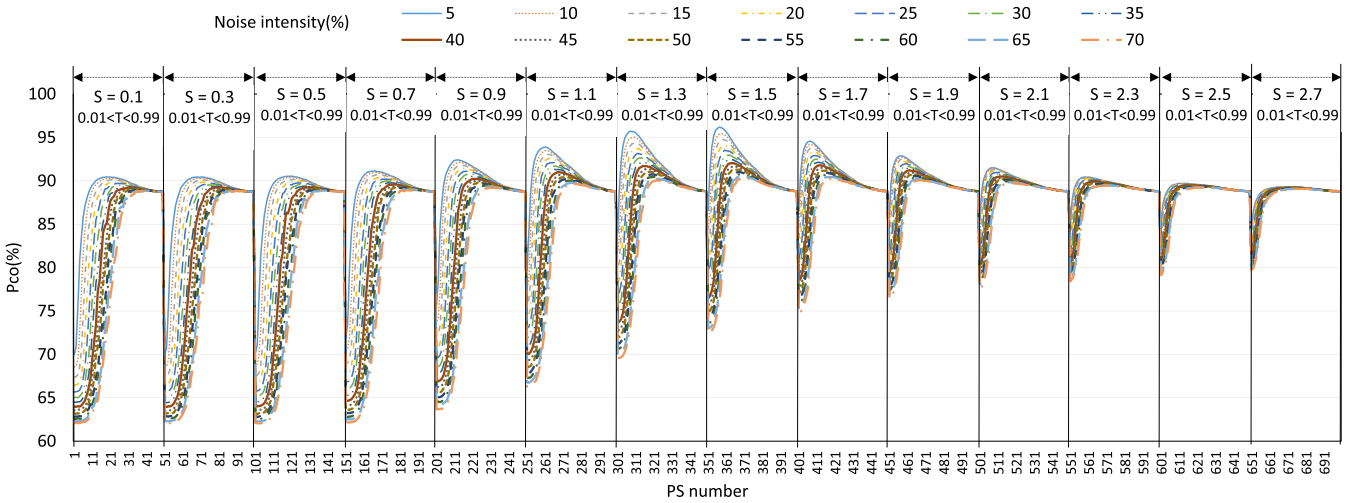


FIGURE 4. Performance comparison of 672 PSs for different noise intensities, where *S* and *T* represent Sigma and Threshold values, respectively.

TABLE 3. PS specification in terms of sigma and threshold values.

Parameter	PS Number			
	1-50	51-100	···	651-700
Sigma	0.1	0.3	···	2.7
TH_H	0.01, 0.03, 0.05, ···, 0.99	0.01, 0.03, 0.05, ···, 0.99	···	0.01, 0.03, 0.05, ···, 0.99

"Original Canny". Here, true edge pixels are those identified by both Original and New Canny algorithms.

In this research, to find the best PSs that suit the circumstances, the following steps are taken.

1) A set of eight different images from the Standard Test Image (STI) database [25] are selected and analyzed.
2) For each image, 14 noisy versions with noise intensity ranging from 5% to 70% (step 5%) are generated.
3) With 700 different PSs, the Canny algorithm is applied to every noisy version of each selected STI image, and consequently, $700 \times 14 \times 8 = 78400$ different P_{co}s are calculated.
4) The obtained P_{co}s are averaged over each noise intensity (from eight images), resulting in $700 \times 14 = 9800$ averaged P_{co}s. The average P_{co} for each combination of the noise intensity and MDP can be calculated using Equation 2, where N and I represent the amount of additive noise and image (pixels) respectively.

$$Average\ P_{\text{co}}(N, PS) = \frac{\sum_{i=1}^{8} C_{\text{NC}}(N, PS, I(i))}{8C_{\text{OC}}} \quad (2)$$

5) To discern a pattern from this large data set, as demonstrated in Fig. 4, the obtained averaged P_{co}s are plotted as separate curves, for 14 PS intervals with 14 curves per interval differentiated by the 14 Sigma values from Table 3.

Fig. 4 shows how applying different PSs affects the performance of the Canny algorithm. In each of the 14 regions, the performance sharply improves when the Threshold value increases. However, after a certain point, the P_{co} gradually starts declining in spite of the continuous rise in the TH_H. The overall trend of Fig. 4 also reveals a similar relationship between the P_{co} and Sigma value. This indicates that to improve the edge detection performance, tuning up not only the Threshold, but also the Sigma value should be taken into account under the circumstances.

Using the curves in Fig. 4, the best PS can be selected based on the noise intensity of the input image and MDP, determined by the application requirements. Investigating this figure reveals that in many situations, there is more than one PS that can satisfy the requested MDP for a certain noise intensity. However, to keep the convolution kernel as small as possible, and consequently to decrease the algorithm's computational complexity, only the PS with the smallest Sigma value should be selected.

The procedure presented in Algorithm 1 describes the proposed PS selection process more clearly, with an example presented in Fig. 5 that reveals details of the portion of Fig. 4 corresponding to a noise intensity of 10%. According to the information presented in Fig. 5, for instance, with $MDP = 94\%$ as the desired performance, although there are 26 candidates for the PS (marked as "+") with $Sigma = [1.3, 1.7]$, only PS number 310 with the lowest Sigma value ($Sigma = 1.3$ and $TH_H = 0.19$) should be selected for this specific circumstance. Other PSs are chosen in the same manner and the configuration table is created accordingly, as shown in Table 4. The information presented in this table

TABLE 4. Configuration table with Sigma and Threshold values represented by "S" and "T", respectively. Gray cells correspond to cases with maximum achievable performance.

	Noise intensity (%)																											
	5		10		15		20		25		30		35		40		45		50		55		60		65		70	
MDP (%)	S	T	S	T	S	T	S	T	S	T	S	T	S	T	S	T	S	T	S	T	S	T	S	T	S	T	S	T
94	1.3	0.15	1.3	0.19	1.3	0.21	1.5	0.19	1.5	0.19	1.5	0.25	1.5	0.25	1.5	0.29	1.5	0.29	1.5	0.31	1.5	0.33	1.5	0.37	1.5	0.41	1.5	0.39
93	1.3	0.21	1.3	0.25	1.3	0.27	1.3	0.23	1.3	0.23	1.5	0.25	1.5	0.25	1.5	0.29	1.5	0.29	1.5	0.31	1.5	0.33	1.5	0.37	1.5	0.41	1.5	0.39
92	1.1	0.21	1.1	0.23	1.1	0.23	1.1	0.23	1.1	0.29	1.3	0.25	1.3	0.29	1.5	0.29	1.5	0.29	1.5	0.31	1.5	0.33	1.5	0.37	1.5	0.41	1.5	0.39
91	0.9	0.25	1.1	0.27	1.1	0.27	1.1	0.29	1.1	0.29	1.1	0.31	1.1	0.33	1.1	0.35	1.3	0.35	1.3	0.37	1.5	0.33	1.5	0.37	1.5	0.41	1.5	0.39

Algorithm 1 PS Selection

```
for MDP = 91 to 94 do
  for N = 5% to 70% do
    for PS number = 1 to 700 do
      if Average P_co(N, PS) > MDP then
        Insert PS into PS candidates set
      end if
    end for
    if PS candidates set = ∅ then
      Selected PS(N, MDP) = PS resulting MAP
    else
      Selected PS(N, MDP) = PS with minimum Sigma
    end if
  end for
end for
```

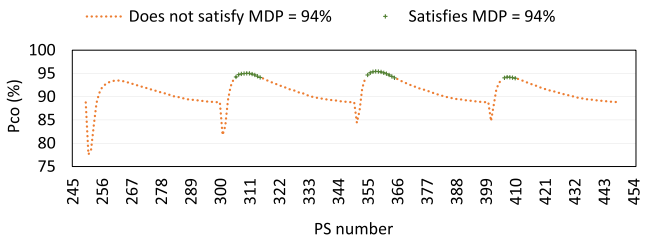
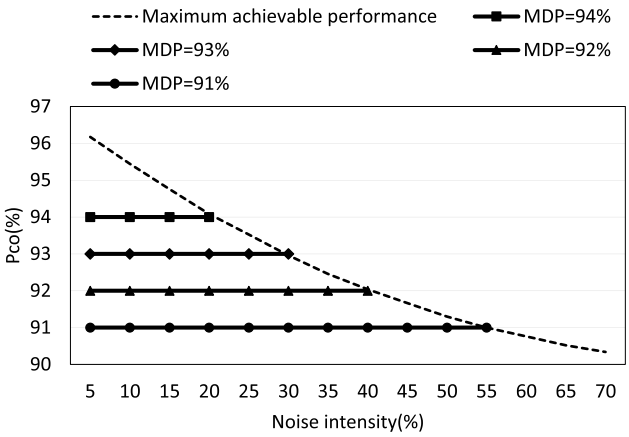


FIGURE 5. Delivered performance using different PSs for noise intensity of 10%, where 26 PSs marked as "+" can satisfy *MDP* = 94%.

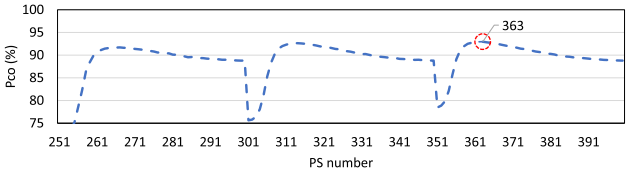
is used to select appropriate values for Sigma and Threshold for each combination of the image noise intensity and MDP.

It is clear that with increased noise, the quality of extracted edge maps decreases and the algorithm's performance declines. In other words, in environments with too much noise, the maximum achievable performance (MAP) cannot exceed a certain limit and, therefore, cannot reach the requested MDP. Such situations, shown in the cells shaded gray in Table 4, can only be handled by a modified version of the aforementioned PS selection method, as discussed below.

To make the steps of the modified PS selection process clear, first the Canny parameters (Sigma and Threshold) are eliminated from Fig. 4 converting the figure into a summary outline as displayed in Fig. 6a. Now this new figure, which concentrates only on the relationship between the input image's noise intensity and MDP, simply represents the highest performance which can be delivered by the Canny edge detection algorithm under certain noise levels.



(a) Maximum achievable performance for different noise intensities



(b) Spotted peak deliverable performance for noise intensity of 25%

FIGURE 6. Maximum achievable performance for different noise intensities.

1) For the leftmost gray-shaded cell in the top row of Table 4, with a noise intensity of 25% and $MDP = 94\%$, investigating Fig. 6a reveals that there is no PS that can possibly achieve the requested output quality (MDP) and therefore this selection procedure needs to find a PS that can deliver the MAP (the closest deliverable performance) instead.
2) By focusing only on the curves representing the noise intensity of 25%, all 14 intervals of Fig. 4 are searched for a peak deliverable performance, which occurs right at PS number 363 with $Sigma = 1.5$ and $TH_H = 0.19$ for $MAP \approx 93\%$, as depicted in Fig. 6b. Now this PS can be stored in the configuration table as shown in leftmost gray-shaded cell in the top row of Table 4.
3) Once the PSs of the other gray-shaded entries in the first row of Table 4 are determined through steps 1 and 2, the rest of the modified PS selection process is to copy the content of an upper cell into the lower gray-shaded cell, if there is any.

The approach presented in this paper aims to make the Canny edge detection algorithm more noise robust with the least computational complexity. The impact of the proposed

APS approach on the robustness of Canny is covered in Section IV and its effect on the implementation performance such as the execution time are presented in Section V.

IV. APS AND PERFORMANCE

Based on the input image noise intensity and the requested MDP, the configuration table presented in Section III adaptively selects values for the Canny's Sigma and Threshold parameters. In this section, the implication of the proposed APS approach for the Canny algorithm's performance is studied. In this section, the edge maps obtained from the original Canny for the noise-free images are used as the basis for comparison.

As mentioned in Section I, to improve the output quality, in the variable-threshold Canny approaches, the Threshold value is calculated on-the-fly based on the input image's condition. These implementations of Canny edge detection are expected to result in a higher performance level than their fixed-parameter counterparts. However, since they ignore Sigma, environmental conditions such as the noise intensity can negatively affect their output quality. In the following sub-section, these two categories of Canny edge detection implementations are evaluated against the proposed APS Canny, which adaptively selects appropriate values for both Threshold and Sigma.

A. QUANTITATIVE EVALUATION

Exhaustive quantitative studies are carried out on a subset of 100 images selected from the Berkeley database [26] using the PSs listed in the configuration table (Table 4). This analysis validates the flexibility of the proposed APS Canny edge detector to adapt to the varying conditions of the input images and to provide an extracted edge map whose quality is as close as possible to that of the reference edge map.

Fig. 7 illustrates the P_{co} of the reference edge map and the edge map obtained using the new APS method for different Gaussian noise intensities from 5% to 70%, alongside Table 5 listing the standard deviations related to the APS case. With a different marker for each P_{co} that corresponds to an MDP, every point on the graph demonstrates the average of the P_{co}s for the selected 100 standard images. The test results presented in this figure can be divided into two categories:

1) Cases in which the achieved performance is not less than the MDP. For example, for $MDP = 94\%$, with noise intensities from 5% to 15%, the delivered performance is higher than the MDP.
2) Cases in which the achieved performance is lower than the MDP. For example, for $MDP = 91\%$, with noise intensities higher than 20%, the provided performance is lower than the MDP.

In the first category, the proposed APS approach works perfectly, since in all cases the performance provided by the adaptive Canny edge detection algorithm satisfies the MDP. In the second category, due to the high noise intensity, the MDP cannot be met by either the APS, variable-threshold or fixed-parameter Canny edge detectors. However, in these cases, the output performance of the proposed APS method is always higher than that of the other two approaches. Similar investigations performed for the same images affected by S&P noise lead to similar results, as shown in Fig. 8. From the results presented in Figs. 7 and 8, it is apparent that using the proposed APS Canny algorithm, the requested MDP can be delivered in different circumstances.

As in most other cases, the state-of-the-art Canny edge detectors reported in [8], [9] and [24] do not provide

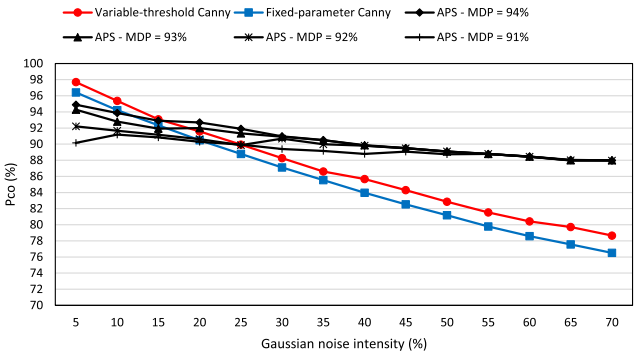


FIGURE 7. Edge map qualities from APS, variable-threshold and fixed-parameter methods for different Gaussian noise intensities.

TABLE 5. Standard deviation of P_{co} reported in Fig. 7 for proposed APS Canny edge detection approach.

Noise intensity (%)	MDP (%)			
	94	93	92	91
5	2.24	2.83	3.00	3.85
10	2.70	3.25	3.26	3.55
15	3.04	3.54	3.51	3.70
20	3.10	3.41	3.80	3.95
25	3.40	3.66	4.16	4.12
30	3.79	3.82	3.91	4.38
35	3.97	3.98	4.23	4.51
40	4.27	4.24	4.26	4.64
45	4.34	4.33	4.37	4.55
50	4.50	4.48	4.49	4.68
55	4.59	4.59	4.58	4.58
60	4.66	4.62	4.65	4.63
65	4.55	4.40	4.45	4.46
70	4.69	4.63	4.60	4.64

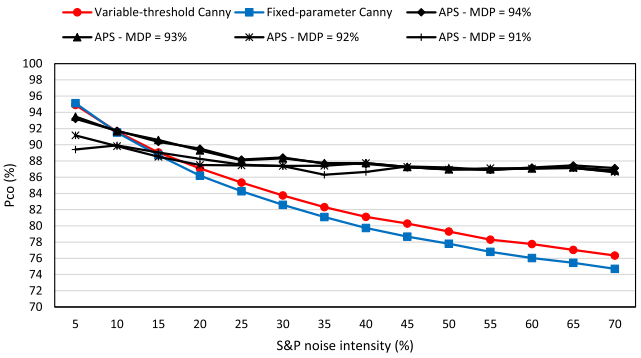
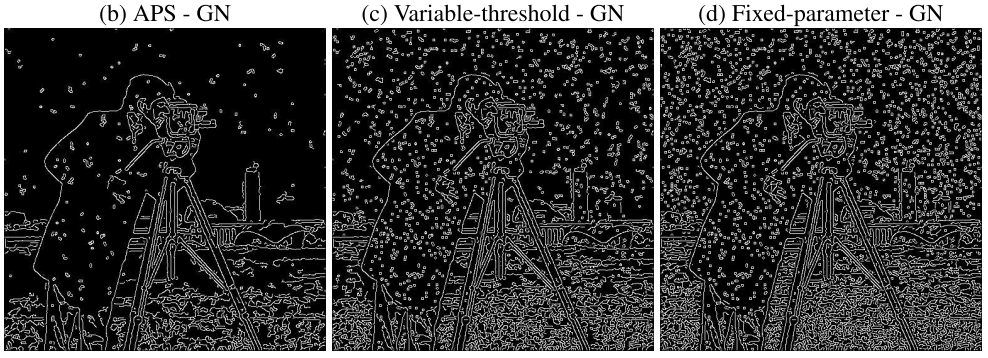
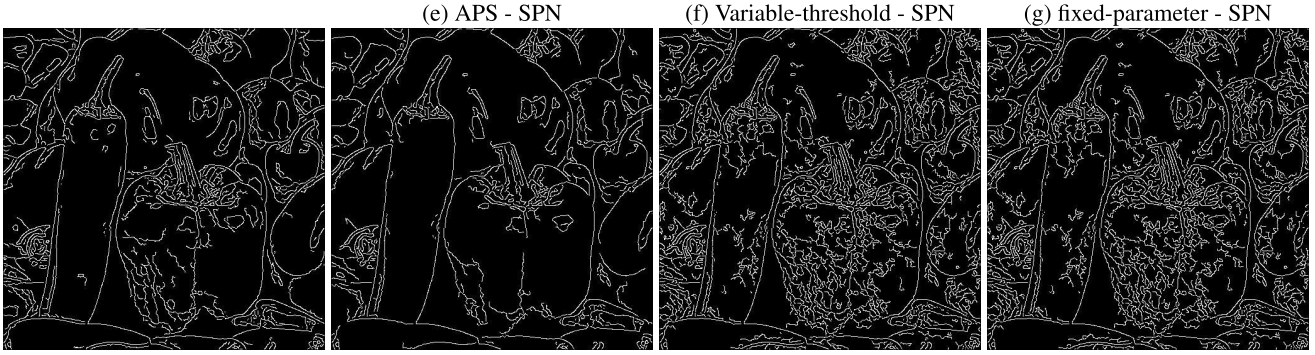


FIGURE 8. Edge map qualities from APS, variable-threshold and fixed-parameter methods for different S&P noise intensities.

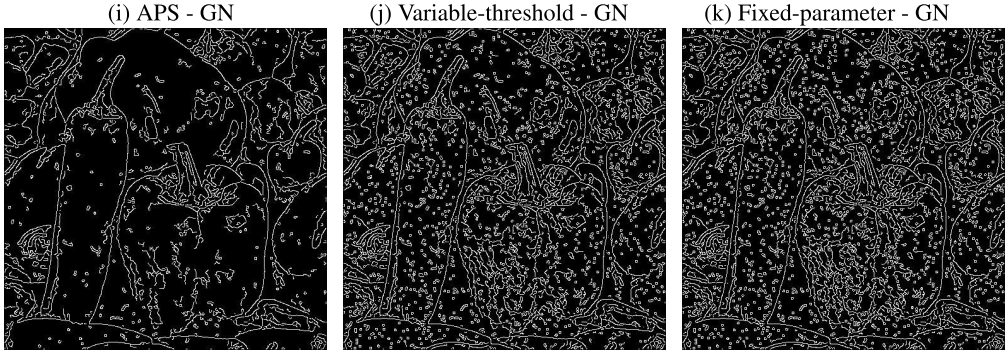
(a) Original Canny - no noise (b) APS - GN (c) Variable-threshold - GN (d) Fixed-parameter - GN



(e) APS - SPN (f) Variable-threshold - SPN (g) fixed-parameter - SPN



(h) Original Canny - no noise (i) APS - GN (j) Variable-threshold - GN (k) Fixed-parameter - GN



(l) APS - SPN (m) Variable-threshold - SPN (n) Fixed-parameter - SPN

FIGURE 9. **Comparison of edge maps of "cameraman" and "peppers" images, generated by different Canny edge detection algorithms with 20% noise level considered for both Gaussian and S&P noise models. GN: Gaussian noise, SPN: S&P noise.**

evaluation results for a variety of noise intensities. Hence, there are not many performance measures available to be shown in Figs. 7 and 8, but just one P_{co} for noise intensity of 40% per design. They are reported in Table 6 alongside that of the proposed Canny edge detection approach, revealing that the APS method can deliver the best result.

B. VISUAL EXAMPLES

In order to visually evaluate the impact of the proposed adaptive Canny, Fig. 9 is constructed. It presents the edge maps obtained from the fixed-parameter, variable-threshold and APS Canny edge detection algorithms for the well-known ''cameraman'' and ''peppers'' images with 20% Gaussian or

TABLE 6. Comparing performance of proposed APS approach with state-of-the-art for noise intensity of 40%.

Method	[8]	[9]	[24]	APS
P_{co} (%)	64.9	76.6	60.5	88

S&P additive noise. The edge maps by the original Canny algorithm for noise-free versions of the images are also shown in this figure as references for comparison.

As noted earlier, in the proposed APS approach, the Sigma and Threshold values are selected from the configuration table. Since the environmental conditions are taken into account in the table-based parameter selection process, the resulting edge maps shown in Figs. 9b, 9e (Figs. 9i and 9l) are comparable to that of the reference edge map presented in Fig. 9a (Fig. 9h). From Figs. 9c, 9f, 9j and 9m, it can be concluded that the new APS Canny algorithm is more robust to both Gaussian and S&P additive noise than the variable-threshold method (which is actually based on the original Canny algorithm). Since in the latter, the Threshold value is updated according to the noise intensity, the edge detection performance for the noisy images is found to be slightly better than the approach that works with fixed parameters. As shown in Figs. 9d, 9g, 9k and 9n, in the presence of noise, due to applying same Sigma and Threshold values previously used for detecting edges of the noise-free images, the fixed-parameter method delivers a poor performance compared to that of the proposed APS implementation of Canny edge detection.

V. HARDWARE IMPLEMENTATION

This section presents hardware realization of the proposed APS Canny edge detection scheme, based on the block diagram shown in Fig. 3, followed by evaluation results. As apparent from the figure, in respect to the conventional Canny's structure depicted in Fig. 1, the new implementation consists of a combination of new and modified functional modules, namely:

- the noise estimator (new),
- adaptive smoothing module (modified),
- configuration table (new) and
- adaptive thresholding (modified),

as well as the gradient magnitude and direction calculation, and NMS units implemented based on the approaches developed in [24].

A. NOISE ESTIMATOR

The proposed hardware implementation accommodates a noise estimation module [21] in the smoothing stage. Having received a window of image pixels, the estimator approximately calculates the noise intensity of the pixel block as the ratio of the number of corrupted pixels to the total number of input pixels. In the proposed design, the noise estimator encodes the result onto a 4-bit number representing any of the quantized noise intensities (14 of them) in $\{5\%, 10\%, \cdots, 65\%, 70\%\}$.

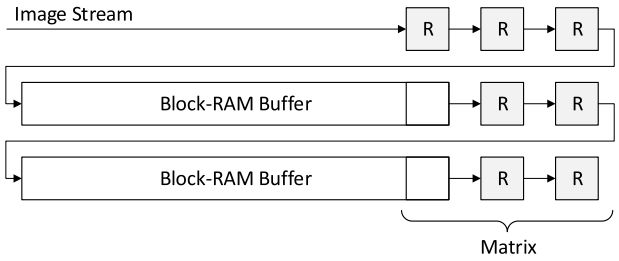


FIGURE 10. Sliding window used in APS convolver.

B. ADAPTIVE SMOOTHING MODULE

As discussed in Section III, based on the estimated noise and requested MDP, a new Sigma value is fetched from the configuration table that matches a specific convolution kernel (each PS corresponds to a particular convolution kernel). This means that the proposed APS Canny edge detector requires a set of predefined 2D convolution kernels (i.e. predefined sizes and coefficients), supported by a Gaussian filter implementation capable of working flexibly with each of those kernels.

Table 4 indicates that based on the experiments setup (MDP and noise intensity ranges), only four (out of 14) distinct Sigma values (i.e. 0.9, 1.1, 1.3 and 1.5) are enough to satisfy the requirements of the proposed APS Canny implementation.

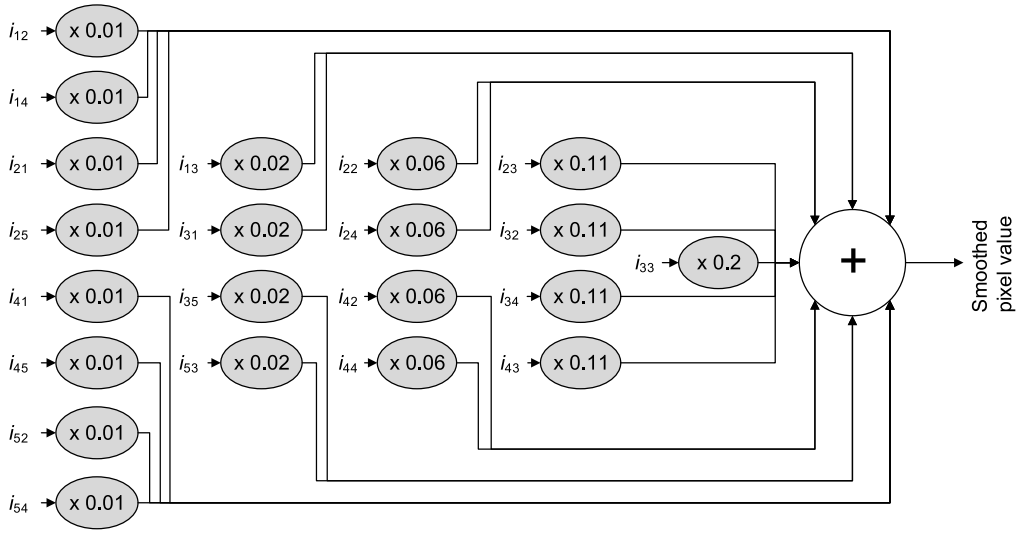
As previously noted in Section I, Sigma is actually the standard deviation of the Gaussian distribution, where, based on the three-sigma rule of thumb, 99.7% of the data spans within three times the standard deviations (Sigma) of the mean [27]. Therefore, the convolution kernel size (an odd integer) has to be determined such that the coefficients of the kernel matrix conservatively fall inside the range of $mean \pm 3Sigma$, resulting in two 5×5 kernels corresponding to $Sigma = 0.9$ and 1.1, and two 7×7 kernels for $Sigma = 1.3$ and 1.5.

Given the number of convolution kernels and their associated sizes, to implement the flexible Gaussian filter, the proposed adaptive smoothing module requires 148 predefined coefficients in total, each equivalent to a constant multiplier when implemented in hardware. However, studying the four kernels reveals that while coefficients equal to zero can be simply eliminated, a number of the coefficients are commonly used by the kernels. Hence, some of the multipliers can be time-shared among different Sigma values, resulting in only 104 independent constant multiplication units.

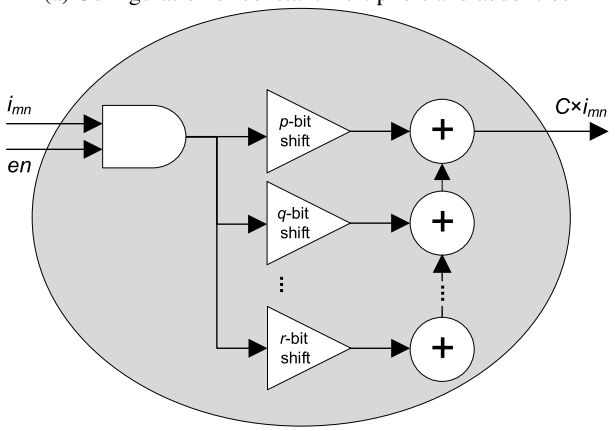
Constant multiplication can be realized using different approaches [28], however, in the proposed APS Canny edge detector, it is implemented through a series of shift-add operations based on the following steps:

1) Each constant coefficient is decomposed into a set of small powers of two to shift the multiplicand to the left and concurrently produce all required partial products.
2) The partial products are then summed up using a tree adder to deliver the final result.

Based on Table 7, which lists the coefficients of the 5×5 kernel matrix corresponding to $Sigma = 0.9$, and by using



(a) Configuration of constant multipliers and adder tree



(b) Implementation of shift-add module, where constant value C is decomposed into a set of wired left shits.

FIGURE 11. Configuration of constant multipliers and adder tree.

TABLE 7. 5 × 5 kernel matrix corresponding to *Sigma* = 0.9.

$$\begin{bmatrix} 0.00 & 0.01 & 0.02 & 0.01 & 0.00 \\ 0.01 & 0.06 & 0.11 & 0.06 & 0.01 \\ 0.02 & 0.11 & 0.20 & 0.11 & 0.02 \\ 0.01 & 0.06 & 0.11 & 0.06 & 0.01 \\ 0.00 & 0.01 & 0.02 & 0.01 & 0.00 \end{bmatrix}$$

the structure demonstrated in Fig. 11, the following example briefly illustrates hardware realization of the smoothing process via 2D convolution, and the steps to be taken in the proposed APS Canny algorithm.

1) Once $Sigma = 0.9$ is selected, the proposed reconfiguration mechanism specifies the corresponding predefined kernel matrix, presented in Table 7, by enabling a specific combination of 21 constant multipliers, as displayed in Fig. 11a (by applying '1' to the AND gate's input labeled "en", as demonstrated in Fig. 11b), and disabling the rest of the constant multipliers.
2) Now the kernel matrix can be multiplied into a same-sized window of image pixels (with the target pixel in the middle of the window) using a network of 8-bit wide (pixel-sized) constant multipliers, followed by an adder architecture accumulating the contributing partial products generated by the shift-add modules (Fig. 11b) to produce a smoothed value for the input pixel. The accumulator is constructed as a fixed 49-operand parallel tree adder (corresponding to the largest kernel size used in the proposed APS Canny edge detector) and hence, receives outputs of all constant multipliers at once. Clearly, once the Sigma value changes (due to any alteration in the processing circumstances), another convolution kernel matrix with its own set of coefficients, implemented through a different set of enabled constant multipliers, may be needed.

C. CONFIGURATION TABLE

In this project, the configuration table is realized as a linear look-up table to store the Sigma and Threshold (in fact, TH_H

TABLE 8. FPGA implementation results; APS versus conventional.

Design	FPGA Device	REGs	LUTs	Memory (Kb)	Max Freq. (MHz)	Exec. Time (ms)	Size
[16]	Virtex E	N/A	N/A	N/A	16	16.8	5×5
[17]	Spartan 6	N/A	N/A	192	201	2.64	5×5
[8]	Virtex V	5080	10312	217	250	4.90	3×3
[9]	Virtex V	3175	8250	278	272	4.29	3×3
[24]	Virtex V	2608	3808	181	246	1.30	3×3
APS	Virtex V	6648	10344	144	263	0.99	variable
APS	Virtex 7	6648	10344	144	355	0.74	variable

and TH_L) values as listed in Table 4, in 56 words. Each word of this table consists of:

- One 104-bit field to enable/disable the right combination of constant multipliers forming the convolution kernel matrix corresponding to either of the four Sigma values.
- Two 16-bit fields, to store two binary fixed-point fractions both in the format of $0.x_{15}x_{14}, \cdots, x_1x_0$, representing 50 different values each for TH_H and TH_L as listed in Table 2.

Entries of this look-up table are addressable by a 6-bit number consists of a 4-bit code representative of the noise intensity (provided by the noise estimator) and the input MDP, encoded as a 2-bit number.

D. ADAPTIVE THRESHOLDING

In a vast number of hardware implementations of Canny edge detection, where Threshold takes only a constant value, the thresholding module is implemented as simple as a single comparator. The adaptive thresholding stage of the proposed APS approach takes advantage of basic comparators except the comparison values (TH_H and TH_L) are not constant but delivered by the configuration table based on the given MDP and estimated noise intensity, as two 16-bit fixed-point numbers as displayed in Table 2.

E. FPGA IMPLEMENTATION

The APS Canny edge detection scheme presented in this work can be implemented on any hardware platform (FPGA and ASIC) with either no or slight modification. However, since recent realizations of Canny edge detection (especially those that claim to satisfy the requirements of real-time video/image processing applications) are mainly implemented on FPGA devices, for fair comparison, the proposed adaptive Canny algorithm is modeled in structural/behavioral Verilog and synthesized on both Xilinx Virtex-V (xc5vsx240t) [29] and Virtex-7 (xc7vx690t) [30] FPGA devices using the ISE v14.7 and Vivado v18.3 design suites respectively. In the remainder of this section, the main comparison results are presented and discussed.

Table 8 lists the execution time (latency of processing one frame), maximum frequency, and resource utilization reported for a number of recent state-of-the-art Canny FPGA implementations against those of the FPGA realization of the proposed APS Canny edge algorithm. As the table displays, with 263 MHz clock frequency for Virtex-V and 355 MHz for Virtex-7 device, the APS edge detector processes a 512×512 image with high noise robustness in 0.99 ms and 0.74 ms respectively, which are 24% and 43% lower than that of the best existing implementation reported in [24].

As shown in Table 8, since the smoothing stage of the APS approach is implemented using a set of constant multipliers and intense pipelining, the resource utilization of the proposed scheme is higher than that of some existing implementations. However, this extra resource usage does not make a considerable impact since it only consumes a small portion of the hardware components available in today's FPGA devices anyhow (4% of Slice Registers and 6% of Slice LUTs of Virtex-V xc5vsx240t FPGA). Table 8 also shows that the proposed APS edge detector requires 144 Kb of BRAM-FIFO in the image interface module [31] for buffering a $k \times k$ sliding window of the input image, which is lower than those of other designs listed in the table. Similar to conventional implementations of 2D convolution, in the proposed APS Canny architecture, the sliding window mechanism moves gradually over the image providing each time, a $k \times k$ matrix with the target pixel in the middle surrounded by $k^2 - 1$ neighboring pixels. This is demonstrated briefly in Fig 10.

It is worth mentioning that the conventional implementations of Canny edge detection use a 3×3 Gaussian filter which is equivalent to taking a small and fixed value for Sigma in all circumstances, which leads to high sensitivity to noise and, consequently, to poor noise robustness. Given the fact that the computational complexity of a 2D convolver with a kernel size of $k \times k$ is in $O(k^2)$, increasing the Gaussian filter's size in the previous designs incurs a dramatic increase in their computational complexity and, consequently, their execution time.

However, in the proposed APS Canny edge detector, due to the parallel architecture of the convolver's multiplication module, any increase in the Gaussian kernel size only reconfigures the way the concurrent constant multipliers are connected to the tree adder, and therefore, has no effect on the execution time of the smoothing stage.

VI. CONCLUSION

Canny algorithm is a popular edge detection method which is widely used in machine vision and image processing applications as a pre-processing step. Its performance and computational complexity depend significantly on appropriate

selection of values for the algorithm's parameters, Sigma and Threshold. In the literature, the parameters' values are either precalculated, which incur a large computational complexity, or given fixed values (regardless of the circumstances), which results in reduced complexity but degraded performance of the edge detector. The main objective of this work is to determine optimal values for the algorithm's parameters considering the additive noise intensity of the input image and the minimum desired performance requested by the application. The experimental results show that the proposed adaptive parameter selection method is noise robust since it maintains the edge detection performance within the acceptable range under any circumstances. The hardware evaluation results also confirm that the edge detection's execution time is lower than that of the state-of-the-art implementations.

REFERENCES

[1] J.-L. Chen, C.-H. Lin, Y.-C. Du, and C.-H. Huang, "Combining fractional-order edge detection and chaos synchronisation classifier for fingerprint identification," *IET Image Process.*, vol. 8, no. 6, pp. 354–362, Jun. 2014.

[2] P. Gupta and V. Gaidhane, "A new approach for flame image edges detection," in *Proc. Int. Conf. Recent Adv. Innov. Eng. (ICRAIE)*, May 2014, pp. 1–6.

[3] J. He, S. Zhang, M. Yang, Y. Shan, and T. Huang, "Bi-directional cascade network for perceptual edge detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3828–3837.

[4] X. Ma, S. Liu, S. Hu, P. Geng, M. Liu, and J. Zhao, "SAR image edge detection via sparse representation," *Soft Comput.*, vol. 22, no. 8, pp. 2507–2515, Feb. 2017.

[5] M. Sharifi, M. Fathy, and M. T. Mahmoudi, "A classified and comparative study of edge detection algorithms," in *Proc. Int. Conf. Inf. Technol., Coding Comput.*, Apr. 2002, pp. 117–120.

[6] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[7] W. He and K. Yuan, "An improved canny edge detector and its realization on FPGA," in *Proc. 7th World Congr. Intell. Control Autom.*, Jun. 2008, pp. 6561–6564.

[8] Q. Xu, S. Varadarajan, C. Chakrabarti, and L. J. Karam, "A distributed canny edge detector: Algorithm and FPGA implementation," *IEEE Trans. Image Process.*, vol. 23, no. 7, pp. 2944–2960, Jul. 2014.

[9] D. Sangeetha and P. Deepa, "FPGA implementation of cost-effective robust canny edge detection algorithm," *J. Real-Time Image Process.*, vol. 16, no. 4, pp. 957–970, Mar. 2016.

[10] Y. Luo and R. Duraiswami, "Canny edge detection on NVIDIA CUDA," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2008, pp. 1–8.

[11] A. Alaghi, C. Li, and J. P. Hayes, "Stochastic circuits for real-time image-processing applications," in *Proc. 50th Annu. Des. Autom. Conf. (DAC)*, May 2013, pp. 1–6.

[12] P. R. Possa, S. A. Mahmoudi, N. Harb, C. Valderrama, and P. Manneback, "A multi-resolution FPGA-based architecture for real-time edge and corner detection," *IEEE Trans. Comput.*, vol. 63, no. 10, pp. 2376–2388, Oct. 2014.

[13] S. Kumar, "Canny edge detection implementation on TMS320C64×/64×+ using VLIB," Texas Instruments, Dallas, TX, USA, Tech. Rep. SPRAB78, 2009.

[14] T. L. Ben Cheikh, G. Beltrame, G. Nicolescu, F. Cheriet, and S. Tahar, "Parallelization strategies of the canny edge detector for multi-core CPUs and many-core GPUs," in *Proc. 10th IEEE Int. NEWCAS Conf.*, Jun. 2012, pp. 49–52.

[15] V. Kritchallo, B. Braithwaite, E. Vermij, K. Bertels, and Z. Al-Ars, "Balancing high-performance parallelization and accuracy in Canny edge detector," in *Architecture of Computing Systems—ARCS 2016* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2016.

[16] D. Venkateshwar Rao and M. Venkatesan, "An efficient reconfigurable architecture and implementation of edge detection algorithm using Handle-C," in *Proc. Int. Conf. Inf. Technol., Coding Comput. (ITCC)*, 2004, pp. 2–3.

[17] C. Gentsos, C.-L. Sotiropoulou, S. Nikolaidis, and N. Vassiliadis, "Real-time canny edge detection parallel implementation for FPGAs," in *Proc. 17th IEEE Int. Conf. Electron., Circuits Syst.*, Dec. 2010, pp. 499–502.

[18] X. Li, J. Jiang, and Q. Fan, "An improved real-time hardware architecture for canny edge detection based on FPGA," in *Proc. 3rd Int. Conf. Intell. Control Inf. Process.*, Jul. 2012, pp. 445–449.

[19] L. Rao, B. Zhang, and J. Zhao, "Hardware implementation of reconfigurable 1D convolution," *J. Signal Process. Syst.*, vol. 82, no. 1, pp. 1–16, Jan. 2015.

[20] T. Matsubara, V. G. Moshnyaga, and K. Hashimoto, "A FPGA implementation of low-complexity noise removal," in *Proc. 17th IEEE Int. Conf. Electron., Circuits Syst.*, Dec. 2010, pp. 255–258.

[21] P. Taghinia Jelodari, M. Parsa Kordasiabi, S. Sheikhaei, and B. Forouzandeh, "FPGA implementation of an adaptive window size image impulse noise suppression system," *J. Real-Time Image Process.*, vol. 16, no. 6, pp. 2015–2026, Jul. 2017.

[22] A. Gagalowicz and W. Philips, Eds., *Computer Vision/Computer Graphics Collaboration Technique*. Berlin, Germany: Springer, 2009.

[23] L. H. A. Lourenco, D. Weingaertner, and E. Todt, "Efficient implementation of canny edge detection filter for ITK using CUDA," in *Proc. 13th Symp. Comput. Syst.*, Oct. 2012, pp. 33–40.

[24] J. Lee, H. Tang, and J. Park, "Energy efficient canny edge detector for advanced mobile vision applications," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 28, no. 4, pp. 1037–1046, Apr. 2018.

[25] R. C. Gonzalez, R. E. Woods, and S. L. Eddins. (2019). *Standard Test Images a set of Images Found Frequently in the Literature*. [Online]. Available: http://www.imageprocessingplace.com/root_files_V3/image_databases.htm

[26] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.

[27] E. Grafarend and J. Awange, *Linear Nonlinear Models*. Cham, Switzerland: Springer, 2012.

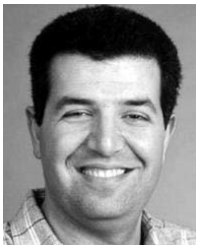
[28] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2010.

[29] *FPGA Data Sheet*, Xilinx Semiconductor, Bengaluru, Karnataka, Jun. 2016.

[30] *FPGA Data Sheet*, Xilinx Semiconductor, Bengaluru, Karnataka, Mar. 2019.

[31] C. Johnston, K. Gribbon, and D. Bailey, "Implementing image processing algorithms on FPGAs," in *Proc. 11th Electron. New Zealand Conf. (ENZCon)*, Palmerston North, the North Island, 2004, pp. 118–123.

MAHDI KALBASI received the B.Eng. degree in computer engineering from the University of Isfahan, Isfahan, Iran, the M.Sc. degree in computer architecture from the Isfahan University of Technology, Isfahan, and the Ph.D. degree in computer architecture from the University of Isfahan, in 2009, 2012 and 2018, respectively. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Golpayegan University of Technology, Isfahan. His research interests are hardware accelerators, convolutional neural networks, and reconfigurable computing.



HOOMAN NIKMEHR received the B.Sc. degree in electronic engineering and the M.Sc. degree in computer architecture engineering from the University of Tehran, Tehran, Iran, in 1992 and 1997, respectively, and the Ph.D. degree in computer engineering from The University of Adelaide, Adelaide, Australia, in 2005. He was an Assistant Professor with the Department of Computer Architecture, University of Isfahan, Isfahan, Iran, until 2018. In 2018, he joined the School of Electrical and Electronic Engineering, The University of Adelaide, as a Senior Research Fellow and then a Lecturer. His current research interests include VLSI, digital arithmetic, computer architecture, reconfigurable hardware design, and low-power design.