On the Design of Approximate Sobel Filter

Alain Aoun*, Mahmoud Masadeh§ and Sofiène Tahar*
{a_alain, tahar}@ece.concordia.ca*, mahmoud.s@yu.edu.jo§
*Department of Electrical and Computer Engineering, Concordia University, Montreal, Quebec, Canada
§Computer Engineering Department, Yarmouk University, Irbid, Jordan

***Abstract*—Approximate computing (AC) is an emerging computing paradigm for energy efficiency. Typically, AC is implemented at the primary arithmetic level, e.g., addition, multiplication, and division, and its performance is evaluated by integration within an application. However, the achieved design efficiency may not be satisfactory. Therefore, for a specific approximate application, we need to study the most suitable settings of its basic approximate component. In this paper, we investigate several approximate designs of the Sobel filter, which is used for image edge detection. We consider different target designs, e.g., for 25% area reduction, we determine the various types of the used full adders and the number of components for each type. For an approximate Sobel filter with 15% to 55% area and power reduction compared to the exact design, we determine the settings for each target design. The obtained Sobel designs are evaluated for different benchmark images, i.e., Cameraman, Lena, and Bikesgray, and show a highly acceptable quality for edge detection. The average multiscale structural similarity (MS-SSIM) index for all evaluated designs on the three benchmark images was 0.73.**

***Keywords*-Approximate Computing, Approximate Arithmetic, Sobel Filter, Digital Circuits, Energy Efficiency, Edge Detection**

I. Introduction

Edges in images are areas with a jump in intensity between adjacent pixels. Edge detection diminishes the amount of data and eliminates nonessential information while keeping the essential structural properties of an image. Existing techniques of edge detection belong to two main categories: 1) *Gradient*: which detects edges by finding the minimum and the maximum in the first derivative of the image, such as the Sobel filter [1]; and 2) *Laplacian*: which searches for zero crossings in the second derivative of the image, such as Gaussian filter [2].

An edge represents a spot in the image with an impulsive transition in the coloring level of the pixels. Also, an edge describes a transition between objects or a transition between an object and the background. Thus, edges naturally obtain visual attraction from humans. However, images include noise, which also causes sudden changes in pixel weights. The process of edge detection includes three main steps: (*i*) *noise reduction*: reduce the noise as much as possible while preserving edges; (*ii*) *edge enhancement*: use a high pass filter to emphasize edges and dilute other pixels; and (*iii*) *edge localization*: find the maxima of output from the previous filter which expresses possible edges and removes noise-related spurious edges.

The Sobel filter is frequently used for edge detection in image processing applications and for obtaining an image emphasizing edges [1]. It works by calculating the gradient of image intensity at each pixel within the image. It finds the direction of the largest increase from light to dark and the rate of change in that direction. The result shows how suddenly or smoothly the image changes at each pixel, and therefore how likely that pixel represents an edge. It uses two 3 x 3 kernels/masks matrices, given as M_x and M_y as follows:

$$M_x = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad M_y = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

One for changes in the horizontal direction (M_x), and one for changes in the vertical direction (M_y). The two kernels, with small integer values, are convolved with the original image to calculate the approximations of the derivatives. Thus, it is reasonable in terms of computations. The Sobel filter is used in autonomous systems with a sensor camera for frame preprocessing. Thus, it is used in real-time for feature extraction to avoid a collision.

In the rest of this paper we will discuss related work in Section II and the proposed approximate Sobel filter designs in Section III. We dedicate Section IV to analyze the proposed approximate designs in terms of resource usage and quality. We conclude this work in Section V.

II. Related Work

In [3], the authors presented various approximations of the Sobel filter using truncation techniques. The output of the Sobel filter, when the data size is reduced by one bit, is matched with the exact output. However, when the data size is reduced by two bits, the result provokes a considerable quality loss. On the other hand, when the data size is reduced by three bits, the output of the filter is hardly recognizable. The authors of [4] explored workload-aware approximate computing. which they applied on the Sobel filter. The used design of the Sobel filter includes 18 multipliers and 17 adders, each can have different settings. Thus, they revealed that even under the same approximation configuration, different input workloads will produce various output qualities.

In order to evaluation 10 designs of approximate adders, a Sobel filter application is utilized in [5]. The authors reported that an approximate adder logic with the appropriate approximate bits can be applied to an edge-detecting filter and has a satisfactory error and image quality. The authors of [6] proposed Learned Approximate Computing (LAC), which concentrates on optimizing the application kernels instead of optimizing the hardware approximations. LAC was evaluated based on various applications including 3x3 Sobel filter for edge detection. The authors of [7] extended a RISC-V processor by adding a variable bit-width memory unit beside the existing variable bit-width arithmetic units. The influence

978-1-6654-9324-6/22/$31.00 ©2022 IEEE

of both variable bit-width arithmetic and memory units on the output accuracy and the energy consumption is evaluated on the Sobel filter. We notice that various techniques of approximate computing are utilizing the Sobel filter at the application level because image processing applications are error-resilient and suitable for approximation.

Most of the related work investigated approximation techniques or approximate *arithmetic units*, e.g., adders, multipliers, squaring, and square root units. Thereafter, they used the Sobel filter as an application in order to evaluate the overall performance, i.e., area, power, delay, energy and quality, of the proposed design(s). However, given the importance of the Sobel filter in image processing, finding the optimal approximated version is necessary. Therefore, in this paper, we propose to investigate the most suitable design settings of its basic approximate components.

III. Proposed Methodology

Approximate computing relies on the principle of significance-driven approximation [8]. Thus, it is critical to distinguish the approximable parts with their approximation settings. For instance, a hardware that approximates the most significant bits (MSBs) will result in a poor quality. In the work we propose in this paper, we aim to limit the effect of approximation on the MSBs while maximizing the benefits obtained from an approximate hardware. We search for the designs that yield minimal quality loss while offering maximum reduction in resource usage, i.e., area and power.

The arithmetic operations performed by the Sobel filter are multiplication, addition, and square root. The previous work is utilizing the Sobel filter as an application to evaluate their proposed approximate components, i.e., adders and multipliers. Thus, the approximate filter will have a reduced area, power, delay, or energy compared to the exact design. In this paper, we consider the Sobel filter as a target approximable application. We introduce approximation into multiple operations, each with different configurations, in order to generate an approximate filter with a specified design metric, e.g., a design with 30% less power consumption. Thus, we propose to design an approximate filter by approximating these operations with a specific configuration. Accordingly, a suitable design will be generated for a specified reduction in a given metric. In this work, we target the reduction of area and power with condensing goals of 15%, 25%, 35%, 45% and 55% for each metric. The reductions are achieved using the method proposed in [9] which solves for *position independent* replacement, and the two equations proposed there are used in this work, which are:

$$M_T = \sum_{i=1}^{n} Q_i \times M_i \tag{1}$$

$$Q_T = \sum_{i=1}^{n} Q_i \tag{2}$$

where n, M_T, Q_i and Q_T are the types of basic cells, total resource usage, the quantity of a given unit, e.g., exact full-adder, and the total quantity, respectively. Moreover, M_i is the correspondent hardware usage, e.g., area or power usage, of a given unit. The concept of *position independent* replacement is applied in this paper to generate various approximate Sobel filters. The goal is achieved by replacing the exact basic blocks, i.e., conventional exact full-adders (FAs), with their approximate counterparts proposed in [10] which are known as AMA1, AMA2, AMA3, AMA4, and AMA5. Those FAs showed superiority over other FAs as shown in [11]. In this direction, as a first step, we perform a synthesis to determine the hardware usage, i.e., finding M_i, of the basic cells in our library, i.e., exact and approximate FAs. This step is required to solve Eq. (1). The synthesis is performed using *Synopsys Design Vision* [12] and TSMC 0.18 μm CMOS technology [13]. The results of the synthesis are summarized in Table I.

TABLE I: Synthesis of Basic Arithmetic Units

Design	Power (μW)				Area (μm^2)	Delay (ns)	
	Internal	Switching	Leakage	Total	-	Sum	Carry
Exact	23.785	7.1674	3.854E-03	30.96	93.5	0.91	0.7
AMA1	13.4	6.93	2.99E-03	20.4	77.24	0.58	0.25
AMA2	8.39	4.42	1.52E-03	12.8	48.78	0.25	0.36
AMA3	3.6	2.69	1.33E-03	6.29	24.39	0.20	0.26
AMA4	4.31	3.76	1.5E-03	7.97	32.52	0.21	0.1
AMA5	1.8	1.31	3.34E-04	3.1	16.26	0.05	0.05

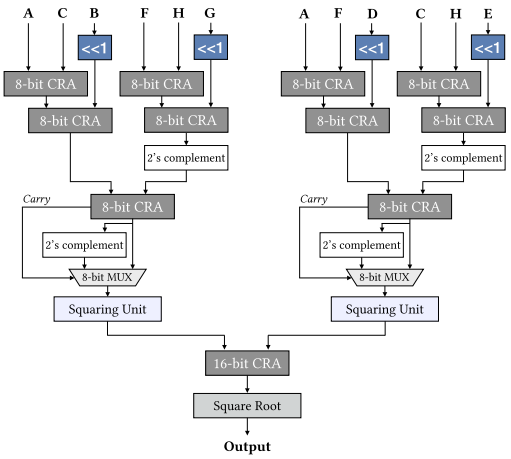


Fig. 1: Hardware Implementation of the Sobel Filter

Thereafter, we determine the Q_T for the hardware implementation of the Sobel in the direction of solving Eq. (2). As shown in Fig. 1, the Sobel circuit accepts eight inputs, i.e., A, B, C, D, E, F, G and H, representing the adjacent pixels of the targeted pixel. Moreover, the Sobel circuit consists of twelve 8-bit Carry-Ripple Adder (CRA) units, two 8-bit 2's complement units, two 8-bit squaring (SQR) units, one 16-bit CRA unit and one 16-bit square root unit. In this work, the square root unit is exact. The 2's complement can be achieved using inverters and 8-bit CRA. Hence, we count the total number of 8-bit CRA units to be equal to 14. Furthermore, the number of FAs is eight, sixteen and thirty-one in the 8-bit CRA, 16-bit CRA and SQR units, respectively. Subsequently, we determine the quantities of FAs in the Sobel hardware as $Q_T = 190$.

Based on the synthesis shown in Table I and the Q_T found, we solve Eqs. (1) and (2) for various reduction targets to determine the number of exact and approximate FAs in each unit. Eqs. (1) and (2) are solved to find the designs that achieve the chosen reductions in terms of area and power, simultaneously. For instance, for a 15% reduction goal, we aim

TABLE II: Quantities of FAs that can Achieve a Given Reduction

Target Reduction	Unit	AMA1		AMA2		AMA3		AMA4		AMA5	
		Exact FAs	AC FAs	Exact FAs	AC FAs	Exact FAs	AC FAs	Exact FAs	AC FAs	Exact FAs	AC FAs
15%	8-bit CRA	1	7	5	3	6	2	6	2	7	1
	16-bit CRA	2	14	11	5	13	3	12	4	13	3
	SQR Unit	4	27	21	10	25	6	24	7	25	6
25%	8-bit CRA	-4	12	4	4	5	3	5	3	6	2
	16-bit CRA	-7	23	8	8	11	5	10	6	11	5
	SQR Unit	-14	45	15	16	21	10	19	12	22	9
35%	8-bit CRA	-8	16	2	6	4	4	4	4	5	3
	16-bit CRA	-16	32	4	12	8	8	7	9	9	7
	SQR Unit	-31	62	8	23	16	15	14	17	18	13
45%	8-bit CRA	-13	21	0	8	3	5	2	6	4	4
	16-bit CRA	-25	41	1	15	6	10	5	11	7	9
	SQR Unit	-49	80	2	29	12	19	10	21	14	17
55%	8-bit CRA	-17	25	-1	9	2	6	1	7	3	5
	16-bit CRA	-35	51	-2	18	4	12	3	13	5	11
	SQR Unit	-67	98	-5	36	8	23	5	26	10	21

to achieve a reduction of at least 15% in both area and power. If the power reduction requires more approximation to meet the reduction goal than the requirement for area reduction, then the requirement for power is considered. As such, the target reduction for the two metrics, i.e., area and power, are achieved. In this work, only homogeneous approximate designs are studied, i.e., only one type of approximate basic cell is used in a given design. Thus, for Eqs. (1) and (2), we determine $n = 2$.

Table II shows the quantities of approximate and exact FAs in each of the 3 units, i.e., 8-bit CRA, 16-bit CRA and SQR unit, for a given target reduction when using a given approximate FA. The replacement of FAs is performed such as the most significant bits (MSBs) are computed using exact FAs. This replacement policy is set to limit the error in the MSBs, i.e., large error distance. From Table II we can notice that under the given reduction targets, solving Eqs. (1) and (2) resulted in a negative quantity of FAs, i.e. quantities highlighted in red. Thus, the target reduction cannot be achieved when using this type of approximate basic cell, i.e., approximate FA. From Table II, we determine 5 designs that can achieve a reduction of 15%, 4 designs that can achieve each a reduction of 25%, 35% and 45%, while only 3 designs can achieve a reduction of 55%, i.e., designs with quantities shown in black.

IV. Design Analysis

In this section, we discuss the hardware resource usage and the output quality of the various approximate Sobel filter implementations that are proposed in this paper. We compare the proposed approximate designs with the exact design in the direction of determining their effectiveness. The proposed implementations of the approximate Sobel filter are modeled in VHDL and Matlab [14] for synthesis and quality analysis, respectively.

A. Resource Usage

The synthesis of proposed approximate Sobel filters is performed using *Synopsys Design Vision* [12] and TSMC 0.18 μm CMOS technology [13]. The synthesis results are summarized in Table III. The average area reduction, i.e., actual reduction, is 14.49%, 23.09%, 33.07%, 43.18%, 53.21% for a target reduction of 15%, 25%, 35%, 45% and 55%, respectively. On the other hand, the average power reduction is 31.45%, 35.70%, 53.09%, 69.03% and 72.14% for a target reduction of 15%, 25%, 35%, 45% and 55%, respectively. The achieved area reduction is near the set target while the reduction of power exceeds the set goal because the approximate FAs offer greater savings in power compared to the area. Hence, more approximate FAs are required to meet a given target reduction in area than it is required for power. Moreover, the achieved area reduction did not meet the targeted reduction as the square root unit is not approximated and hence not considered in the total area, i.e., M_T, in Eq. (1). Subsequently, Eqs. (1) and (2) were applied to a portion of the circuit, i.e., CRA and SQR units. On the other hand, the power-area-product is always achieved with a minimum actual reduction of 16.41%, 39.61%, 61.02%, 72.70% and 82.40% for a target reduction of 15%, 25%, 35%, 45% and 55%, respectively.

TABLE III: Synthesis Results of the 20 Approximate and the Exact Sobel Designs

Target Reduction	Type of AC FA	Area (μm^2)	Power (mW)	Delay (ns)
15%	AMA1	16,880.29	6.44	13.81
	AMA2	16,583.57	15.10	16.26
	AMA3	16,607.96	22.32	18.12
	AMA4	16,774.64	18.34	15.32
	AMA5	16,868.19	24.56	17.00
25%	AMA2	15,286.63	11.53	15.44
	AMA3	14,949.17	18.85	18.68
	AMA4	15,189.03	14.51	14.55
	AMA5	14,810.99	20.21	16.45
35%	AMA2	13,229.42	5.72	15.07
	AMA3	13,083.04	14.63	18.57
	AMA4	13,542.43	11.78	14.23
	AMA5	12,566.77	15.37	14.78
45%	AMA2	11,529.99	1.80	6.43
	AMA3	11,424.26	11.84	19.05
	AMA4	11,225.00	6.75	12.6
	AMA5	10,322.56	10.97	12.99
55%	AMA3	9,765.47	8.93	18.80
	AMA4	9,639.39	4.34	11.27
	AMA5	8,078.34	7.88	11.79
Exact Hardware	—	19,579.95	25.312	17.31

B. Quality Analysis

The 20 proposed designs were tested for quality analysis using three benchmark images, i.e., “bikesgray”, “cameraman”

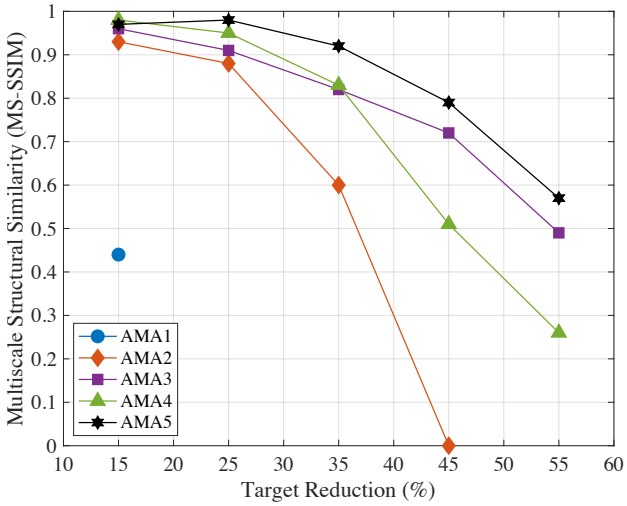
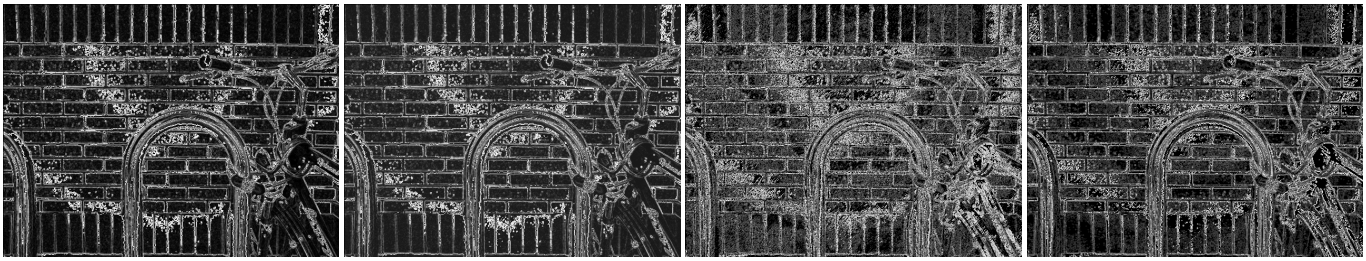


Fig. 2: Target Reduction versus Average Output Quality for the 20 Proposed Designs

and "lena". The quality of the resulting images when using various designs are analyzed objectively and subjectively. The objective assessment is measured using the multiscale structural similarity (MS-SSIM) index [15] which measures the quality of the input image compared to the reference image. The value of MS-SSIM ranges from 0 to 1, where a higher value indicates a better output quality. Fig. 2 shows the average MS-SSIM for the three resulting images when using various proposed designs. From Fig. 2 we can notice a trend of reduced quality as the target reduction increases. For instance, the designs approximated using AMA2 achieves an average MS-SSIM of 0.93 and 0.005 for a target reduction of 15% and 45%, respectively. However, not all design result in deteriorated quality for a larger target reduction. For instance, the MS-SSIM of the AMA5 based design increased from 0.97 to 0.98 when the target reduction was increased from 15% to 25%. Additionally, we note that a simpler hardware might deliver better quality. For instance, AMA5 is a simpler approximate FA compared to AMA4. Nonetheless, for a target reduction of 55%, AMA5 offers a superior output quality while delivering greater savings in the area and competitive power and delay values. Also, for a target reduction of 15%, the approximate Sobel based on AMA1 achieved a low quality, while AMA1 requires the highest resource compared to the other four approximate FAs. Finally, as shown in Fig. 2, we determine the approximate Sobel filter based on AMA4 with 15% target reduction and AMA5 with 25%, 35%, 45% and 55% as the Pareto optimal since they offer the best quality in an objective quality assessment, i.e., using MS-SSIM, for a given target reduction.

Analyzing the imaging subjectively, i.e., visually, we notice that the resulting edge detection shown in Fig. 3b offers a very similar result to the exact design. However, the MS-SSIM of the image shown in Fig. 3b is 0.65 only. Furthermore, the resulting edge detection in Figs. 3c and 3d have a difference in MS-SSIM of 0.01. Nonetheless, when analyzing the images subjectively, we can notice the edge detection in Fig. 3d, i.e., using AMA5 with a target reduction of 55%, offers a closer result to the exact than the resulting image shown in Fig. 3c. From Figs. 4a and 4b, we notice that the approximate model detected the body of the "cameraman" with less noise. However, the approximate design resulted in more ground

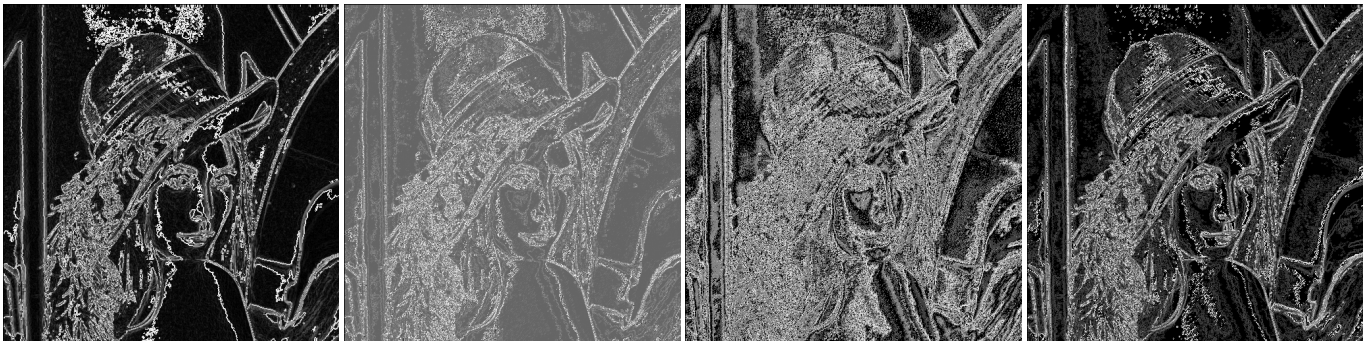


(a) (b) (c) (d)

Fig. 3: Edge Detection of "Bikesgray" Using the (a) Exact, (b) AMA3; Target = 25%, (c) AMA4; Target = 45% and (d) AMA5; Target = 55%, Implementation of the Sobel Filters

(a) (b) (c) (d)

Fig. 4: Edge Detection of "Cameraman" Using the (a) Exact, (b) AMA1; Target = 15%, (c) AMA3; Target = 45% and (d) AMA5; Target = 55%, Implementation of the Sobel Filters



(a) (b) (c) (d)

Fig. 5: Edge Detection of "Lena" Using the (a) Exact, (b) AMA3; Target = 55%, (c) AMA4; Target = 55% and (d) AMA5; Target = 55%, Implementation of the Sobel Filters

noise compared to the exact. Also, we notice that the resulting edge detection shown in Fig. 4c detected the edges better than the two preceding designs, i.e., less noise on the body of the cameraman and minimal noise on the ground. The MS-SSIM of Figs. 4c and 4d have a difference of 0.06. However, when analyzed subjectively, as shown Fig. 4c, we notice that the edge detection when using the approximate Sobel filter based on AMA3 and a target reduction of 15% offers a greater quality as the human can perceive the information it carries. From Fig. 5 of Lena, we notice that for low MS-SSIM, the results shown in Figs. 5b and 5d are still consumable, i.e., can be perceived visually. On the other hand, the result shown in Fig. 5c cannot be recognized. Finally, we notice from Fig. 5 that for the same target reduction, using different basic approximate units generates diverse output quality.

V. Conclusion

In this work, we have proposed several approximate Sobel filter hardware designs. The approximate versions are generated by replacing the exact full-adders (FAs) with their approximate counterparts. The used approximate FAs in this work are widely known and have been proposed in [10]. The replacement policy adopted in this paper consists of homogeneous replacement, i.e., single type of approximate FA at a time, and replacing FAs that compute the least significant bits (LSBs). The replacement of FAs that compute the LSBs is adopted to reduce the error in the most significant bits (MSBs) and thus reduce the magnitude of the error. The number of FAs replaced in a circuit is chosen based on a target reduction in area and power. The chosen targets reduction in this paper are 15%, 25%, 35%, 45% and 55%. The effective power reduction exceeded the set target while the effective reduction of area usage was near the chosen target. The quality of the edge detection using the approximate Sobel filter varied among the designs with an average MS-SSIM of all evaluated images of 0.73. Moreover, when analyzing the quality objectively, i.e., based on quantitative error metric, some images indicate a low quality while when analyzed subjectively, i.e., visually, the quality is deemed acceptable as it can be perceived by the human brain. As a future work, we will investigate the implementation of an approximate Sobel filter with heterogeneous block replacement, i.e., more than one approximate FA is used at a time. Moreover, we aim to improve the library of approximate FAs to produce a larger design space and potentially find superior designs.

References

[1] I. Sobel and G. Feldman, "A 3×3 isotropic gradient operator for image processing," *Pattern Classification and Scene Analysis*, pp. 271–272, 01 1973.

[2] M. Basu, "Gaussian-based edge-detection methods-a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 32, no. 3, pp. 252–260, 2002.

[3] G. Rodrigues, F. Lima Kastensmidt, and A. Bosio, "Survey on approximate computing and its intrinsic fault tolerance," *Electronics*, vol. 9, no. 4, p. 557, 2020.

[4] D. Ma, R. Thapa, X. Wang, X. Jiao, and C. Hao, "Workload-aware approximate computing configuration," in *Design, Automation & Test in Europe*, 2021, pp. 920–925.

[5] Y. Chung and Y. Kim, "Comparison of approximate computing with sobel edge detection," *IEIE Transactions on Smart Processing & Computing*, vol. 10, no. 4, pp. 355–361, 2021.

[6] V. Gupta, T. Li, and P. Gupta, "Lac: Learned approximate computing," in *Design, Automation & Test in Europe*. IEEE, 2022, pp. 1169–1172.

[7] G. Ndour, T. T. Jost, A. Molnos, Y. Durand, and A. Tisserand, "Evaluation of variable bit-width units in a RISC-V processor for approximate computing," in *International Conference on Computing Frontiers*, 2019, pp. 344–349.

[8] M. Masadeh, O. Hasan, and S. Tahar, "Input-conscious approximate multiply-accumulate (MAC) unit for energy-efficiency," *IEEE Access*, vol. 7, pp. 147 129–147 142, 2019.

[9] A. Aoun, "On the improving of approximate computing quality assurance," Master's thesis, Concordia University, April 2021, unpublished. [Online]. Available: https://spectrum.library.concordia.ca/id/eprint/988392/

[10] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 1, pp. 124–137, 2012.

[11] M. Masadeh, O. Hasan, and S. Tahar, "Comparative study of approximate multipliers," in *ACM Great Lakes Symposium on VLSI*, 2018, pp. 415–418.

[12] Synopsys, "Design compiler graphical," last Accessed October 16, 2022. [Online]. Available: https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/design-compiler-graphical.html

[13] CMC Microsystems, "TSMC 18µm CMOS Process Technology," last Accessed October 16, 2022. [Online]. Available: https://www.cmc.ca/tsmc-0-18-um-cmos/

[14] Mathworks, "What is MATLAB," last accessed October 16, 2022. [Online]. Available: https://www.mathworks.com/discovery/what-is-matlab.html

[15] Z. Wang, E. Simoncelli, and A. Bovik, "Multiscale structural similarity for image quality assessment," in *Asilomar Conference on Signals, Systems & Computers*, vol. 2, 2003, pp. 1398–1402 Vol.2.