Microprocessors and Microsystems 45 (2016) 1–9

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

A quality-aware Energy-scalable Gaussian Smoothing Filter for image processing applications

Bharat Garg*, G.K. Sharma

ABV-Indian Institute of Information Technology and Management Gwalior, Gwalior 474015, Madhya Pradesh, India

ARTICLE INFO

Article history:
Received 17 July 2015
Revised 14 December 2015
Accepted 9 February 2016
Available online 8 March 2016

Keywords:
Energy-efficiency
Approximate designs
Gaussian smoothing filter
Edge-detection
Quality-energy tradeoff

ABSTRACT

Energy-efficient design is the prime requirement for modern portable devices as these devices employ compute intensive image/video processing cores which produces output for human consumption. The limited perception of human sense can be exploited to improve energy-efficacy via approximate designs. In this paper, a novel quality-aware Energy-Scalable Gaussian Smoothing Filter (ES-GSF) is proposed that significantly reduces energy requirement at the cost of slightly reduced quality. The energy scalability within ES-GSF is achieved by exploiting the relative significance of kernel coefficients existing on different boundaries. The ES-GSF computes significant boundaries for the given energy budget. Simulation results show that ES-GSF consumes 30.46% reduced energy with graceful quality degradation over the well-known existing architectures. Further, the ES-GSF can scale energy up to 65.05% when switched from high quality mode to energy-efficient mode. The efficacy of the proposed filter is demonstrated in edge detection where ES-GSF embedded edge detectors consume 29.9% less energy over the well-known existing architectures.

1. Introduction

In recent years, there has been an exponential usage of the battery operated portable devices that exhibit different functionalities to attract large number of consumers. The complexity of these devices is also increasing exponentially which results in poor energy-efficiency and reliability. Fortunately, the advancement in the VLSI technology has allowed us to fabricate billions of transistors on the same chip to implement desired functionality. However, the energy-efficiency is still a serious concern for these high density portable devices as user cannot tolerate rapid discharge of battery. Thus, energy-efficient designs are the critical requirement for these devices as they not only increase battery life-time but also increase the reliability. Since most of these portable devices employ multimedia processing that produces output for human consumption, it allows approximate computation due to limited human perception. Therefore, approximate design has become a new paradigm for energy-efficient designs for the multimedia applications. The approximate designs [1] exhibit speed, power, area and accuracy (SPAA) tradeoff. Significant efforts have been devoted to achieve energy efficient designs/architectures [2–4] using the concept of approximate computing. These designs improve power, area and delay metrics significantly over the conventional designs at the cost of minor loss of accuracy.

In most of the multimedia applications, image processing tasks are applied to real word images where noise may introduce during acquisition, compression or transmission stage. In order to eliminate the introduced error, noise reduction tasks are commonly employed which involve averaging pixels value residing in a local area and generating a blurred or smoothed image. Gaussian smoothing filter (GSF) is commonly utilized in various image/video processing applications such as edge detection, image blurring, mosaicing etc. and critically determines the performance of these applications. The well-known edge detection algorithms such as Canny [5] and Marr and Hildreth [6] employ 2D-GSF. The 2D-GSF is also useful in many other applications such as texture segmentation [7], tone mapping of high dynamic range images [8], image blurring [9], and image mosaicing [10]. The various existing GSF architectures provide tradeoff between performance and the quality, and are selected according to the requirement of the application [11–13]. However, in real world applications, requirements are changing which demand different performance under different conditions that can be achieved by approximate designs. In these designs, higher energy-efficiency/performance is achieved at the cost of acceptable loss in accuracy/quality of the results. This work presents a novel Energy-Scalable Gaussian Smoothing Filter (ES-GSF) that provides energy-quality tradeoff. This energy-quality tradeoff is achieved by varying number of pixels based on their

* Corresponding author. Tel.: +91 8103283400.
E-mail addresses: bharat@iiitm.ac.in, bgarg.iiitm@gmail.com (B. Garg), gksharmabharat@iiitm.ac.in (G.K. Sharma).

http://dx.doi.org/10.1016/j.micpro.2016.02.012
0141-9331/

significance. The objective of this work is to develop a reconfigurable architecture that can configure diameter of the bell-shaped kernel according to given energy/quality constraint. The architecture proposed in this paper provides significantly improved design metrics over the existing architectures with graceful quality degradation. The major contributions of this paper are as follows:

- We introduce an approximate Gaussian smoothing filter mask that reduces implementation complexity significantly.
- We propose an energy-scalable Gaussian smoothing algorithm that utilizes a novel concept of significance of boundaries which is based on the weighted coefficients.
- An energy-scalable Gaussian smoothing filter architecture is proposed that provides quality-energy tradeoff.
- The efficacy of the proposed filter is evaluated by implementing edge detectors embedded with proposed ES-GSF and well-known GSFs and simulating with benchmark images.

The rest of the paper is organized as follows: The principle of Gaussian smoothing and various energy efficient GSF architectures are discussed in Section 2, whereas proposed approximate Gaussian kernel and its architecture is given in Section 3. Section 4 provides details of proposed ES-GSF algorithm and architecture. The simulation environment, comparative analysis of various GSFs and their efficacy in the real world applications are given in Section 5. Finally, conclusion is given Section 6.

2. Related work

Gaussian filter is a special weighted averaging filter and is employed in several image processing tasks such as image blurring, image segmentation, and edge detection. The performance of the Gaussian filter determines the performance of these image processing tasks. This section first explores principle of GSF and then reviews different energy efficient architectures available in the literature.

2.1. Principle of Gaussian smoothing

Gaussian smoothing is the most commonly used image smoothing technique. A 2D Gaussian function with zero mean (μ) and constant standard deviation (σ) is expressed by:

$$g(x, y) = exp(-(x^2 + y^2)/2\sigma^2) \quad (1)$$

where, x and y are two variables. The standard deviation plays an important role in the Gaussian function behaviour. The values distributed between $\pm\sigma$, $\pm 2\sigma$, and $\pm 3\sigma$ account 68%, 95%, and 99.7% of the set, respectively. The Gaussian distribution convolves the image with the 2D Gaussian distribution function, which is a point-spread function. In order to approximate the Gaussian function in discrete time, we need to have infinitely large convolution kernel, which is impossible practically. But for approximation, we can neglect the values beyond $\pm 3\sigma$, where the distribution has almost reached zero.

In general, Gaussian filter is a non-uniform low-pass filter. The values of kernel coefficients have inverse relation with distance i.e., they decrease with increase in distance from the centre, which has the highest value. The amount of blurring depends on the width of the peak. Larger the value of sigma, wider the peak. To maintain the Gaussian nature of the filter we must increase the kernel size while increasing sigma. The kernel is symmetric about the centre with no directional bias and its coefficients depend on the value of sigma. Although, the kernel allows fast computations because of its separable nature, it does not preserve the brightness of the image.

In order to efficiently implement the Gaussian function, it must be approximated by fixing number of coefficients commonly called kernel or mask. Larger value of σ is considered for better smoothing, whereas large kernel size is needed to accurately represent the function. A 5 × 5 Gaussian kernel can be obtained by evaluating Eq. (1) for each value of the variables x and y from $[-2, 2]$. For example, a 5 × 5 Gaussian kernel with $\sigma = 1$ is obtained by varying the value of variables x and y from -2 to 2 as shown below.

$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix} \quad (2)$$

In order to evaluate the smoothed value of a pixel, a 5 × 5 image sub-matrix is extracted with pixel to be smoothed in centre position and the kernel is multiplied with this image sub-matrix. Further, sum of all coefficients of the kernel must be one i.e. kernel must be normalized by the sum of all coefficients to maintain the intensity level of the smoothed image. Considering more neighbour pixels i.e. increasing the size of the kernel to find out the average value of the noisy pixel provides better smoothing but increases the complexity of the design. All existing image smoothing filter architectures are designed with fixed kernel sizes and thus provide fixed energy and quality parameters.

2.2. Energy-efficient GSF architectures

Very less research has been devoted to derive an energy-efficient Gaussian smoothing filter. The conventional approaches such as voltage over scaling (VOS) and truncation of few least significant bits fail to provide sufficient energy savings and introduce significant errors. As the accurate Gaussian kernel contains floating point coefficients that require large silicon area, Khorbotly and Hassan, [11] proposed a kernel with rounded fixed-point approximate coefficients. This kernel provides area efficient smoothing and improves the performance at the cost of introduced error. Hsiao et al. [12] proposed a 5 × 5 kernel based 2D-GSF that contains coefficients in terms of power-of-two. Although, the implementation area of the mask is small as the coefficients are in power-of-two form, it does not show energy scalability. A simplified approximate GSF is proposed in [13] by exploiting nearest-pixels approximation and rounding-off kernel coefficients. Although, this architecture provides high energy-efficiency and acceptable output image quality simultaneously, applications with varying energy budget declines its use as it provides output with fixed energy. None of the existing filter architectures provide flexibility to select desired quality. The next section introduces a variable size Gaussian smoothing filter that provides energy-scalability at the cost of reduced quality, thus exhibiting an energy-quality tradeoff.

3. Proposed mask and concept of significance of boundaries

This section begins with introduction of proposed approximate kernel and discussion on relative significance between the kernel coefficients using boundary concept. The quantitative analysis on the non-significant coefficients elimination and corresponding complexity reduction is discussed in latter part of this section.

3.1. Proposed approximate kernel

The Gaussian smoothing expression is given by the Eq. (1), and corresponding 5 × 5 kernel (with $\sigma = 1$) is given by Eq. (2). The accurate implementation of the kernel demands floating point arithmetic which is complex and costly. To reduce implementation complexity, each coefficient in the kernel is approximated by sum of power of two (SOPOT) as given by Eq. (3). The SOPOT coefficient in the approximate kernel can be efficiently multiplied using