GPU Implementation of Fuzzy Anisotropic Diffusion

Roberto de Jesús Duarte Coello, Felipe de Jesús Solís Lugo, Alejandro Castillo Atoche
Engineering Department, Universidad Autónoma de Yucatán
Mérida, C.P. 97203, Yucatán, México

and

Jaime Ortegon Aguilar
Science Division, University of Quintana Roo,
Chetumal, C.P. 77019, Quintana Roo, México

ABSTRACT

In this paper, we present a GPU-based implementation of the Fuzzy-Anisotropic diffusion technique oriented for high-resolution multidimensional image/video techniques. The aggregation of parallel computing and the HW/SW co-design techniques are used in order to improve the time performance of the Fuzzy-Anisotropic Diffusion algorithm for image/video applications. Experimental results show the significantly increased performance efficiency both in resolution enhancement and in computational complexity reduction metrics gained with the proposed approach.

1. INTRODUCTION

Advances in sensor technology are revolutionizing the way of images are collected, managed and processed. The incorporation of latest-generation sensors to multispectral and hyperspectral systems is currently producing a near-continual stream of high-dimensional image data. Such amount of collected information is now required to be processed in (near) real-time mode for newer applications in Earth monitoring, in medical image/video fusion and enhancement, and computer vision. Also, these applications need timely responses for swift decision which depend upon real-time performance of algorithm implementation [1], [2].

Additionally, the computational complexity of the advanced high resolution multidimensional image/video techniques that employ the recent methods for enhancement, reconstruction, post-processing and classification procedures [3]-[7], is unacceptable for (near) real-time implementation. In this regard, a tremendous amount of data processing is required for different type of image processing algorithms. To provide such high computational demands under real time constraints, highly parallel processing schemes must be applied. Usually, general-purpose systems are based on multi-PC, field programmable gate arrays (FPGAs) or digital signal processing (DSP) platforms.

Therefore in this study, the aggregation of the fuzzy anisotropic diffusion and the GPU implementation techniques via the hardware/software co-design paradigm is addressed for real-time data processing. The principal innovation that distinguishes our approach from previous studies [5]-[10] is twofold: first, the conceptualization and algorithmically adaptation of the fuzzy

Anisotropic Diffusion technique for image/video data processing is employed. Because the noise and edges are both high frequency image components, most of the conventional approaches do not work well for edge-preserving smoothing of images corrupted with noise. In this work, instead of using the Laplacian filter as the edge factor in the anisotropic diffusion, fuzzy edge detectors are introduced in order to provide a more flexible and robust way to define the edges. The essential idea is to avoid blurring of the edges, with the incorporation of an edge stopping function which estimates the diffusion coefficients ensuring the smoothing process only in the interior regions without crossing the edges. Second, the algorithmic implementation using massively processors in a graphic processing unit (GPU) platform is performed. Here, parallel computing techniques are used in order to improve the time performance of the algorithm.

The analysis results of the simulations and the performance testing with the NVIDIA Tesla C2075, are indicative that our GPU-based implementation is also oriented toward video processing applications.

2. FUZZY ANISOTROPIC DIFFUSION ALGORITHM

Since Perona and Malik proposed in 1990 the anisotropic diffusion, this developed technique had been applied to different areas of image processing including edge enhancement, noise reduction and segmentation. In this subsection, we present the description of how the fuzzy and the anisotropic diffusion technique is aggregated and then, efficiently implemented in a GPU platform.

2.1 Anisotropic Diffusion

Let us consider the diffusion processing technique as the result of convolving an original image *I*o with a Gaussian kernel *G* of increasing width as follows

$$I(x,y,t) = I_0(x,y) * G(x,y,t) . \qquad (1)$$

Here, (1) acts as a low-pass filter suppressing high frequencies in the image *I*. The problem is that the image edges and noises, are both high frequency signals, and therefore, the edges are blurred by this operation.

To solve this problem, the anisotropic diffusion equation proposed by Perona and Malik is next defined as

$$\begin{aligned} I_t &= \frac{\partial I}{\partial t} = div(c(x,y,t)\nabla I) \\ &= c(x,y,t)\Delta I + \nabla c(x,y,t)\nabla I \end{aligned} \qquad (2)$$

where $c(x,y,t) = g(\|\nabla I(x,y,t)\|)$ is the diffusion coefficient, ∇I represents the gradient of the image, $g(\|\nabla I(x,y,t)\|)$ is the stopping function and the initial condition is represented as $I(x,y,0) = I_0(x,y)$.

Now, the discrete version of the anisotropic diffusion equation of (2) is represented as

$$\begin{aligned} I_S^{t+1} = \ & I_S^t \ + \\ & \lambda \Bigg[\frac{1}{d_x^{\,2}} \cdot c_N \cdot D_N(I) + \frac{1}{d_x^{\,2}} \cdot c_S \cdot D_S(I) + \\ & \frac{1}{d_y^{\,2}} \cdot c_E \cdot D_E(I) + \frac{1}{d_y^{\,2}} \cdot c_W \cdot D_W(I) + \\ & \frac{1}{d_d^{\,2}} \cdot c_{NE} \cdot D_{NE}(I) + \frac{1}{d_d^{\,2}} \cdot c_{NW} \cdot D_{NW}(I) + \\ & \frac{1}{d_d^{\,2}} \cdot c_{SE} \cdot D_{SE}(I) + \frac{1}{d_d^{\,2}} \cdot c_{SW} \cdot D_{SW}(I) \Bigg]_{i,j}^{t} \end{aligned} \qquad (3)$$

where $c_N, c_S, c_E, c_W, c_{NE}, c_{NW}, c_{SE}, c_{SW}$ represents the conduction coefficients, d_x, d_y, d_d are the distance between the pixels, and $D_N, D_S, D_E, D_W, D_{NE}, D_{SE}, D_{NW}, D_{SW}$ indicates nearest-neighbor differences for the corresponding direction.

In this paper, the conduction coefficients are calculated as follows

$$c_x = \frac{1}{1+\left(\frac{D_x}{K}\right)^2}. \tag{4}$$

The parameter K in (4) is chosen according to the noise level and the edge strength. In any image/video system, the noise level is not known a priori with certainty. Furthermore, because of inherent noise, the calculated edge strength based on gradient is not a true reflection of the edge strength. Therefore, the ambiguity of choosing a suitable value for parameter K, and thus, the uncertainty in the diffusion coefficient justifies the use of fuzzy set theory in such situations.

2.2 Fuzzy Anisotropic Diffusion

Fuzzy logic systems (FLS) is a rule-based theory in which an input is first fuzzifier (converted from a crisp number to a fuzzy set) and subsequently processed by an inference engine that retrieves knowledge in the form of fuzzy rules contained in a rule-base. The fuzzy sets computed by the fuzzy inference as the output of each rule are then composed and defuzzified (converted from a fuzzy set to a crisp number).

In this study, we propose to change each nearest-neighbor differences D_x that define de edge factor (algorithmically same as Laplacian Filter in each direction) in the traditional Perona-Malik algorithm with a FLS approach that calculates the edges in order to avoid the noise of the image. The following fuzzy rules are defined

Fuzzy Rule 1:

IF D_{NW} is *Zero* AND D_N is *Zero* AND D_{NE} is *Zero* THEN is *Black* ELSE is *White*

Fuzzy Rule 2:

IF D_{SW} is *Zero* AND D_S is *Zero* AND D_{SE} is *Zero* THEN is *Black* ELSE is *White*

Fuzzy Rule 3:

IF D_{NE} is *Zero* AND D_E is *Zero* AND D_{SE} is *Zero* THEN is *Black* ELSE is *White*

Fuzzy Rule 4:

IF D_{NW} is *Zero* AND D_W is *Zero* AND D_{SW} is *Zero* THEN is *Black* ELSE is *White*

Fuzzy Rule 5:

IF D_N is *Zero* AND D_{NE} is *Zero* AND D_E is *Zero* THEN is *Black* ELSE is *White*

Fuzzy Rule 6:

IF D_N is *Zero* AND D_{NW} is *Zero* AND D_W is *Zero* THEN is *Black* ELSE is *White*

Fuzzy Rule 7:

IF D_E is *Zero* AND D_{SE} is *Zero* AND D_S is *Zero* THEN is *Black* ELSE is *White*

Fuzzy Rule 8:

IF D_W is *Zero* AND D_{SW} is *Zero* AND D_S is *Zero* THEN is *Black* ELSE is *White*

The proposed fuzzy anisotropic diffusion approach is next defined as

$$\begin{aligned} I_S^{t+1} = I_S^t + \lambda \Bigg[& \frac{1}{d_x{}^2}\cdot c_N \cdot \Upsilon_N(I) + \frac{1}{d_x{}^2}\cdot c_S \cdot \Upsilon_S(I) \\ & + \frac{1}{d_y{}^2}\cdot c_E \cdot \Upsilon_E(I) + \frac{1}{d_y{}^2}\cdot c_W \cdot \Upsilon_W(I) + \\ & \frac{1}{d_d{}^2}\cdot c_{NE} \cdot \Upsilon_{NE}(I) + \frac{1}{d_d{}^2}\cdot c_{NW} \cdot \Upsilon_{NW}(I) \\ & + \frac{1}{d_d{}^2}\cdot c_{SE} \cdot \Upsilon_{SE}(I) + \frac{1}{d_d{}^2}\cdot c_{SW} \cdot \Upsilon_{SW}(I) \Bigg]_{i,j}^{t} \end{aligned} \tag{5}$$

where Υ_x represents the edge in each direction using a centroid deffuzifier. Now, we are ready to implement the Fuzzy Anisotrpic Diffusion in the GPU platform.

3. GPU IMPLEMENTATION OF THE FUZZY-ANISOTROPIC DIFFUSION

The GPU implementation of the algorithm was done taking advantage of the multicore architecture of the GPU and textures for high-speed memory access. For the implementation, it was used a triangular membership function [3], to define the fuzzy sets the where the values to define the size of the triangle are the following, WHITE = 147, BLACK = 107 and ZERO = 5.

First, the fuzzy sets are initialized using the triangular membership function. The fuzzy sets are allocated in the GPU memory, and set using functions from NPP library. The NPP library has optimized functions that work on 1-D and 2-D signals. The image is allocated in texture memory, since it will be read-only, and texture fetch is cached, each multiprocessor has 8K cache memory. The enhancement is performed out of place, i.e., the input and output image are separated.

Since the objective of the implementation is to enhance large images, it is not likely to have a kernel thread for each pixel; hence there are as many threads as columns in the image (to take advantage of coalescence) or the maximum number supported by the device. Each thread apply the fuzzy rules, deffuzifies the resulting fuzzy set and compute the resulting value for the fuzzy anisotropic diffusion.

The fuzzy rules are implemented in the form of minimum function for the conditional part of the rule. The resulting fuzzy sets *Black* and *White* are implemented also as minimum functions. Afterwards, the fuzzy sets are defuzzified computing the centroid of them. Finally, the resulting scalar is used on the fuzzy anisotropic diffusion equations. The whole process is summarized in Fig. 1.

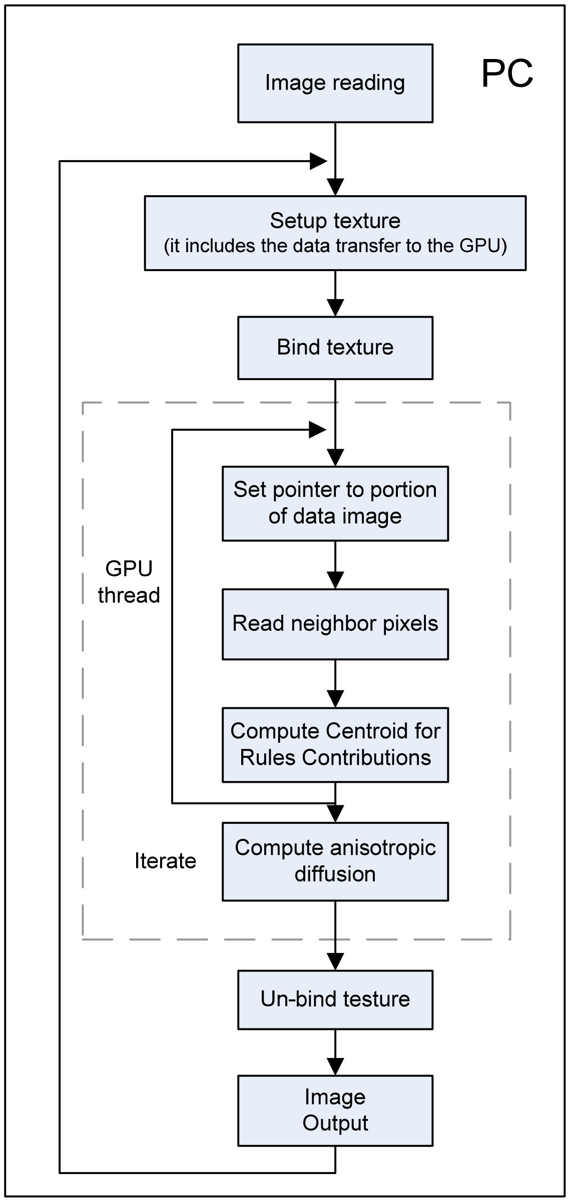
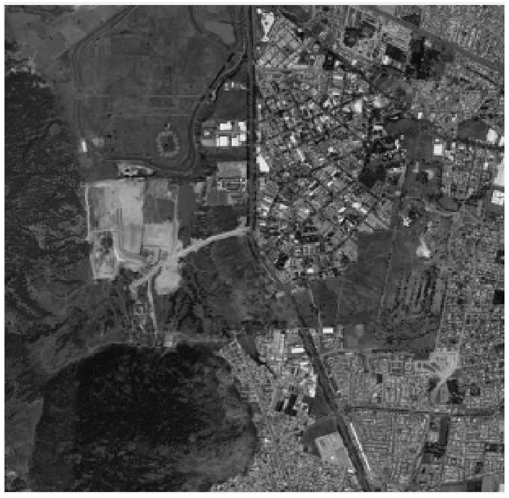

PC
Image reading
Setup texture
(it includes the data transfer to the GPU)
Bind texture
GPU thread
Set pointer to portion of data image
Read neighbor pixels
Compute Centroid for Rules Contributions
Iterate
Compute anisotropic diffusion
Un-bind testure
Image Output

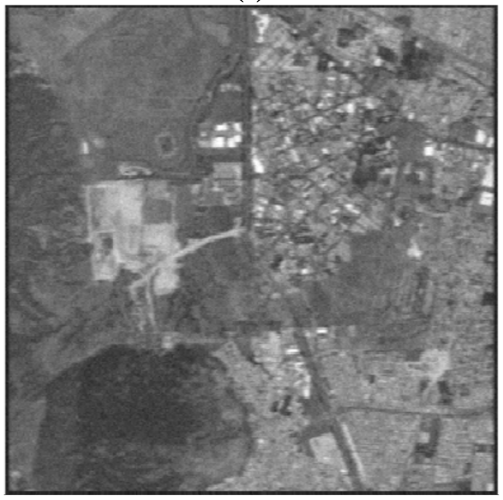
Fig. 1.HW/SW co-design of the Fuzzy Anisotropic Diffusion algorithm.

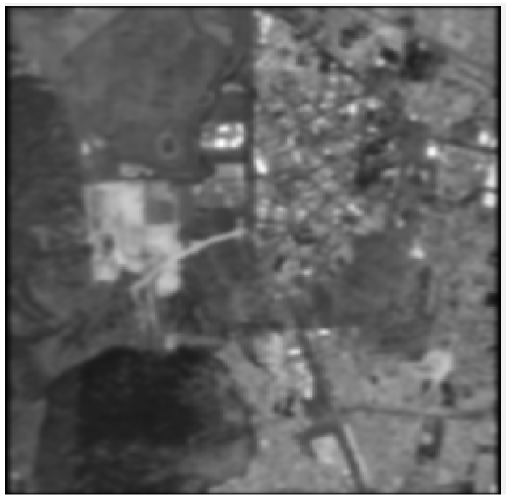
It is worth to mention, the result of the fuzzy anisotropic diffusion algorithm implementation has to be normalized to the range [0,255]. This is employed using functions from NPP library.

4. SIMULATIONS AND PERFORMANCE ANALYSIS

In order to demonstrate the enhanced imaging with our proposed approach towards video processing, we have conducted some initial simulation experiments using a real-world satellite scene. The tested scene is shown in Fig. 2(a). In the reported simulations, we considered the case of white observation noise [2] with SNR of 15 dB as shown in Fig. 2(b). Fig. 2(c) presents the image enhancement using the anisotropic diffusion method and figures 2(d) shows the image enhancement with the proposed fuzzy anisotropic diffusion method.


(a)


(b)


(c)

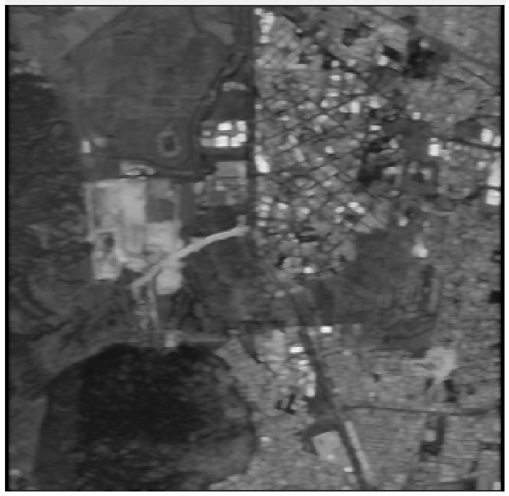

(d)

Fig. 2. Implementation results: (a) original test scene; (b) degraded scene image formed applying the Matched Space Filter method; (c) image reconstructed applying the Anisotropic Diffusion algorithm (3.57 dB); (d) image reconstructed applying the Fuzzy Anisotropic Diffusion algorithm (6.21 dB).

In the performance analysis, we implement the proposed algorithm in the CPU quad-core Intel Xeon E5603 at 1.6GHz and also, in the NVIDIA Tesla C2075 GPU platform. In Table 1, it is presented the results of such comparative analysis.

In this regard, we have compared the time performance of our algorithm against the MATLAB reference implementation. These routines have been reported to be highly optimized, and are comparable with other similar applications (see [5]-[10]).

Image size (pixels)	Time processing (seconds)	
	CPU	GPU
512 x 512	20.15	0.25
1024 x 1024	79.44	0.84

Table 1. Comparative performance analysis of the proposed fuzzy anisotropic diffusion algorithm.

Having analyzed Table 1, one can deduce that the processing time required for implementing the fuzzy anisotropic diffusion algorithm using CUDA has been drastically reduced with the GPU. Particularly, the GPU implementation of the fuzzy anisotropic diffusion algorithm takes only 0.84 seconds for the 1024 x 1024 pixels image enhanced in contrast to 79.44 seconds required with the MATLAB implementation. Thus, the processing time of the proposed GPU-based algorithm implementation is approximately 140 times less than the corresponding processing time achievable with the conventional CPU implementation.

5. CONCLUDING REMARKS

The principal result of this undertaken study relates to the GPU-based implementation of the Fuzzy-Anisotropic diffusion technique oriented for high-resolution multidimensional image/video techniques. We have examined that pursuing the proposed HW/SW co-design, the sub-tasks of the proposed Fuzzy-Anisotropic diffusion technique can be algorithmically "properly adapted" in computationally efficient parallel representation towards video processing applications. The efficiency of the co-design approach was verified with a real test-case scenario in which the time performance of the algorithm was substantially reduced up to 140 orders.

ACKNOWLEDGMENTS

This study was supported by Consejo Nacional de Ciencia y Tecnología (México) under the grant CB-2010-01-158136.

REFERENCES

[1] J.R. Jensen, Introductory digital image processing: a remote sensing perspective, Prentice-Hall, U.S.A., 2005.

[2] H.H Barrett and K.J. Myers, Foundations of Image Science, New York: Willey, 2004.

[3] R. C. Gonzalez, R. E. Woods, Digital Image Processing, 3rd Edition, Prentice Hall, 2008.

[4] J. Song, H. R. Tizhoosh, "Fuzzy Anisotropic Diffusion A Rule Based Approach", 7thW. Multiconference on Systemics, Cyebernetics and Informatics, Jul. 27-30, Orlando, USA, pp. 241-246,2003.

[5] Y.V. Shkvarko, "Unifying experiment design and convex regularization techniques for enhanced imaging with uncertain remote sensing data," IEEE Trans. Geosci. Remote Sens., vol. 48, No. 1, pp. 82–111, 2010.

[6] Y.V. Shkvarko, Unifying regularization and Bayesian estimation methods for enhanced imaging with remotely sensed data-Part II: Implementation and performance issues, IEEE Trans. Geoscience and Remote Sensing, vol. 42, No. 5, pp 932-940, 2004.

[7] M.S. Greco and F. Gini, "Statistical analysis of high-resolution SAR ground clutter data", IEEE Trans. Geosicence and Remote Sensing, vol. 45, No.3, pp. 556-575, 2007.

[8] A. Castillo Atoche, D. Torres, Y. V. Shkvarko, "Descriptive Regularization-Based Hardware/Software Co-Design for Real-Time Enhanced Imaging in Uncertain Remote Sensing Environment", EURASIP Journal on Advances in Signal Processing, Vol. 2010, 31 pages, 2010.

[9] A. Castillo Atoche, D. Torres, Y. V. Shvarko, "Towards Real Time Implementation of Reconstructive Signal Processing Algorithms Using Systolic Arrays Coprocessors", ELSEVIER Journal of System Architecture, Vol. 56, Issue 8, pp. 327-339, 2010.

[10] A. Castillo Atoche, J. Ortegon Aguilar, J. Vazquez Castillo, "A Multi-Processor System on Chip Architecture for Real Time Remote Sensing Data Processing", IEEE International Geoscience and Remote Sensing symposium (IGARSS'11), Vancouver BC., Canada, July 2011.