ACCELERATED UNSUPERVISED FILTERING FOR THE SMOOTHING OF ROAD PAVEMENT SURFACE IMAGERY

Henrique Oliveira[1,3], *José Caeiro*[2,3], *Paulo Lobato Correia*[1]

[1]Instituto de Telecomunicações – Instituto Superior Técnico
[2]Grupo de Sistemas de Processamento de Sinal – SIPS/INESC ID
[3]Escola Superior de Tecnologia e Gestão – Instituto Politécnico de Beja

ABSTRACT

An accelerated formulation of the Unsupervised Information-theoretic Adaptive Image Filtering (UINTA) method is presented. It is based on a parallel implementation of the algorithm, using the Open Computing Language (OpenCL), while maintaining the precision and efficiency of the original method, which are briefly discussed focusing on the respective computational complexities. The experimental computational efficiency is compared with the one obtained using the standard implementation, highlighting the significant improvement of computational times achieved with the proposed one. This new implementation is tested for the smoothing of road pavement surface images, for which the original method had been previously applied, showing the clear advantage of its use.

Index Terms— Road crack detection, image filtering, density estimation, computational complexity, entropy reduction.

1. INTRODUCTION

Pavement surface imagery acquired during high speed road surveys, captured using INO's LRIS 4K model [1], present a high variance in pixels intensities. This represents a challenge when developing automatic algorithms for road pavement surfaces distress detection, with most algorithms targeting the detection of road cracks [2].

Texture smoothing filtering techniques can be applied to this kind of imagery, to reduce the variance of pixel intensities, especially in regions not revealing crack distresses, ensuring that an adequate segmentation procedure will then be able to better distinguish between crack and no-crack pixels [3]. A modern noise reduction method was proposed by Awate and Whitaker whose principles stand close to the non-local-means algorithms [4]. This kind of methods address the preservation of structure in digital images [5], which is an important characteristic when dealing with the road crack detection problem, since crack information in images may not be severely deteriorated when a smoothing technique is applied.

The authors acknowledge the support from Fundação para a Ciência e Tecnologia (FCT) and Instituto de Telecomunicações (http://www.it.pt/), under project PEst-OE/EEI/0008/.

Reducing the entropy of the intensity patterns in image regions, by applying a filtering technique like the Unsupervised Information-theoretic Adaptive Image Filtering (UINTA) [4], is a smoothing filtering strategy that can be successfully included in a road pavement surface crack detection system, as shown in [3]. Nevertheless, the resulting processing times are very high, due to the computational complexity of the algorithm, $O(|T||A_i|E^D)$, where $|T|$ is the total number of pixels of the image, $|A_i|$ is the size of a random sample used in the entropy estimation procedure, E is the neighborhood window size and D is the image dimension.

Two approaches for improving the computational times may be pursued, notably: to choose a more efficient density estimation algorithm or to implement a parallel version of the UINTA algorithm.

Following the first approach, a nonparametric density estimation, capable of dealing with the computational requirements of large images, is described in [6]. It uses a fast algorithm, in terms of dataset size and dimensionality, based on the *kd-tree* probability density estimation. It is also experimentally demonstrated in [7] that *dual-tree* methods give the best results when dealing with high dimensional multivariate nonparametric probability density estimations. Other approaches presented in the literature for improving the computational speed of kernel density estimation methods include the proposal made by Silverman, which is based on the Fast Fourier Transform (FFT) [8], while the proposal made by Elgammal *et al*. is based on the Fast Gauss Transform [9]. Recent parallel implementation proposals were suggested by Michailidis and Margaritis for GPUs with the CUDA framework [10], as well as proposed by Srinivasan *et al* [11]. From the scientific literature, no OpenCL implementation based on parallel computing was found, this framework considered a royalty free open standard for cross platform parallel programming of modern processors, allowing an heterogeneous model of computing not restricted to one brand of GPUs as CUDA.

However, the probability density estimate adopted in reference [7] is based on the entire set of image intensities, resulting on $O(n\ log(n))$ computational complexity, where n is the total number of pixels in the image sample. In UINTA, a more local and problem connected approach is followed, resulting in a lower computational complexity.

Due to the locality of UINTA, it is easy to achieve good speedups with a parallel implementation of the algorithm, instead of following approaches [7], [8] and [9]. Therefore a parallel version of the UINTA algorithm implemented with the OpenCL framework is presented, these GPU based parallel programs being considered a good low-cost solution for improving the computational times, as demonstrated in this paper for the development of automatic systems for the detection of cracks on images. These automatic crack detection systems typically demand a significant computational effort due to handling input images of large size. Therefore, efficient pre-processing procedures are needed to smooth them without significantly degrading the crack structures.

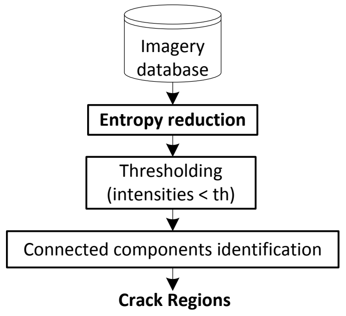
The paper organization is as follows. Section 2 presents the crack detection system architecture considered, including the image acquisition procedures and subsequent processing. Section 3 describes the theoretical and practical formulation of the proposed smoothing approach, discusses the corresponding computational complexity and explains how to achieve a low-cost fast implementation. Section 4 provides a set of experimental results and their discussion. Section 5 draws some conclusions and presents hints for future work

2. SYSTEM ARCHITECTURE

Since the proposed efficient smoothing strategy will be tested in the context of a system that detects cracks in road pavement surface images, this section describes the considered system architecture. In this paper a simple crack detection system is used, mainly to allow highlighting the importance of the smoothing step. The system architecture is illustrated in Figure 1, including the imaging system used for imagery acquisition during high speed road pavement surveys, the entropy reduction (smoothing) module and the subsequent crack detection by thresholding and processing of the resulting connected components for the identification of crack segments.

The road surface images considered were acquired by the INO's LRIS 4k system. It features a laser imaging system for road pavement surface imagery acquisition, allowing operation at the high speeds (70 km/h and higher) required for safe driving in highways [1]. The system is composed by two sets of linescan sensors combined with two laser illuminators (positioned at the left and right on the backside of the vehicle), providing a good contrast between crack and no-crack regions of the image. Each linescan set covers half road lane outputting images with dimensions of 4096×2048 pixels, with the larger image dimension being parallel to the road axis and the other dimension covering approximately 2 meters of pavement surface (half road lane). All the images have 8-bit gray level resolution, presenting pixel intensities ranging from 0 to 255. A sample pair of images simultaneously acquired by the left and right sensors are shown in Figure 2.

Using such a simple system architecture, as shown in Figure 1, is only possible if a very efficient entropy reduction module can be designed for smoothing the images of


Imagery database
Entropy reduction
Thresholding (intensities < th)
Connected components identification
Crack Regions
Fig. 1. Architecture of the proposed crack detection system.

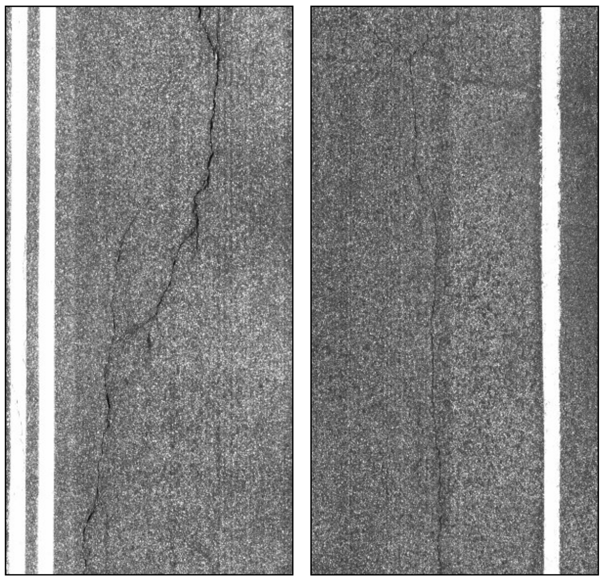


Fig. 2. Sample road pavement surface images acquired by the INO's LRIS 4k model.

road pavement surface. Since a very large amount of imaging data is collected while surveying existing road networks, a timely processing of the acquired footage is required. Therefore, it is important to have a processing system of low computational complexity, notably at the entropy reduction stage where a nonparametric multivariate probability density estimation method is applied. This allows considering a very simple segmentation approach for crack detection, such as the modified Otsu thresholding operator proposed by Wan and Wang [12].

3. PROPOSED FAST UNSUPERVISED FILTERING

The proposed fast probability density estimation filtering approach uses a parallel version of the unsupervised information-theoretic adaptive image filtering [4].

Modern Graphic Processing Units (GPUs) are very efficient at manipulating images and their highly parallel structure makes them more effective than general purpose Computer Processing Unit (CPU), for algorithms where processing of large blocks of data can be done in parallel. The OpenCL is a framework for writing programs consisting of heterogeneous processors: CPUs, GPUs, Digital Signal Processors, etc. OpenCL has an Application Program Interface (API) that is used to define and control the processor platforms and also a C-based language for writing functions that execute on the processing devices (kernels).

The entropy estimation in UINTA is based on a classical multivariate Gaussian kernel density estimation. This leads to O(n^2) computational complexities, where n is the total number of pixels in the image sample. The UINTA algorithm authors state that their method is computationally inefficient and therefore limit the size of the samples that can be used to a maximum of $|A_i| = 1000$ of points, defined on a discrete Cartesian grid. Even for such small samples, the procedure is very slow. These are clearly limiting factors for the direct application of the UINTA methodology to smooth pavement surface images. A system for the automatic detection of cracks based on UINTA has been proposed in [3], where only a subset of carefully selected image blocks are pre-processed using this smoothing technique, rather than the whole image.

Despite the computational limitations mentioned by researchers, the UINTA method presents good experimental results and is theoretically sound. Central to the UINTA filter is the computation of an estimate of the entropy given by:

$$h(\tilde{Z}) = -E_P[\log P(\tilde{Z})]. \tag{1}$$

The entropy is estimated over the stationary random vector $\tilde{Z} = (\tilde{X}, \tilde{Y})$, representing image regions. The full set of image pixels is represented by $\{t\}$. The subjacent random process is considered to be stationary and ergodic. The original image is represented by $X(t)$, the corresponding set of neighborhood intensities by $Y(t)$ and regions by $Z(t)$. The corresponding random variables for the observed degraded image are represented by $\tilde{X}$,$\tilde{Y}$ and $\tilde{Z}$.

For a stationary ergodic process, the entropy of the image may be approximated by the average of estimated local entropies:

$$h(\tilde{Z}) = -E_P[\log P(\tilde{Z})] \approx \frac{1}{|T|}\sum_{t_i \in \{t\}} \log[P(\tilde{z}_i)], \tag{2}$$

where $|T|$ is the number of points in the image and z_i is the vector of intensities associated to each point t_i. The entropy minimization procedure of UINTA, which is based on the gradient descent technique, can be written as:

$$\hat{x}^{m+1} = \hat{x}^m - \lambda \frac{\partial h}{\partial \hat{x}^m}, \tag{3}$$

where x is the image intensity for each point in the image and the derivative of the entropy is given by (4), where C is the covariance matrix and G_n represents the computed value from the pdf estimate.

$$\frac{\partial h}{\partial \hat{x}_i} \approx \frac{1}{|T|} \times \frac{\partial \log[P(\tilde{z}_i)]}{\partial \hat{x}_i} =$$
$$-\frac{1}{|T|}\frac{\partial \hat{z}_i}{\partial \hat{x}_i} \sum_{t_j \in A_i} \frac{G_n(\hat{z}_i - \hat{z}_j)}{\sum_{t_k \in A_i} G_n(\hat{z}_i - \hat{z}_k)} C^{-1}(\hat{z}_i - \hat{z}_j). \tag{4}$$

The proposed pseudo code for the kernel is presented in Figure 3.

```
ParallelUinta(sourceImage, outputImage, Fi_row, Fi_col,
       nrows, ncols)
  Ni = 1000
  W = 4
  sigma = 3.0
  lambda = 0.2 * sigma * sigma / 9.0
  ti_col = get_global_id(0)
  ti_row = get_global_id(1)
  xi = read_imagef(sourceImage, ti_row, ti_col)
  FOR k = 0; k < Ni; k++
    sum = 0.0
    Ai_col = Fi_col[k] + ti_col
    Ai_row = Fi_row[k]      + ti_row
    FOR  u = -W; u <= W; u++
      FOR v=-W; v <= W; v++
        x_uv = read_imagef(sourceImage, ti_row+u, ti_col+v)
        ai_uv= read_imagef(sourceImage, Ai_row+u, Ai_col+v)
        a = x_uv - ai_uv
        sum += a * a
      END FOR
    END FOR
    xk = read_imagef(sourceImage, Ai_row, Ai_col)
    p = exp(sum / (-0.5 * sigma) )
    sum_k += p
    sum_j += p * (xi - xk)
  END FOR
  r = sum_j / (sigma * sum_k)
  y = xi - lambda * r
  write_imagef(outputImage, ti_col, ti_row, y)
END
```

Fig. 3. UINTA pseudo code based on parallel processing.

The kernel is applied to each pixel of the *sourceImage* and the updated value of the pixel, after the application of the parallel UINTA kernel, is stored in the *outputImage*. The coordinates of the image are referenced by *(ti_col, ti_row)*. The *read_image()* and *write_image()* functions are provided by OpenCL with appropriate sampling procedures.

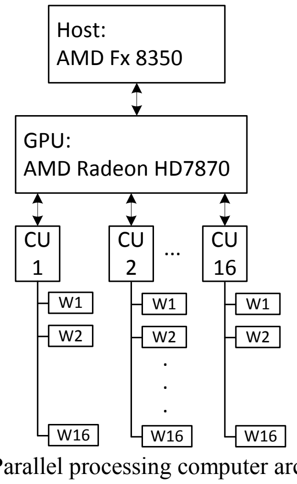


Fig. 4. Parallel processing computer architecture.

The *(Fi_row, Fi_col)* represent a previously generated random Gaussian distributed vector sample of *Ni* points, centered on *(0,0)* coordinate and is used to form the *(Ai_row, Ai_col)* vector of coordinates that will point to the local image intensities used to estimate the pdf, as proposed in [4].

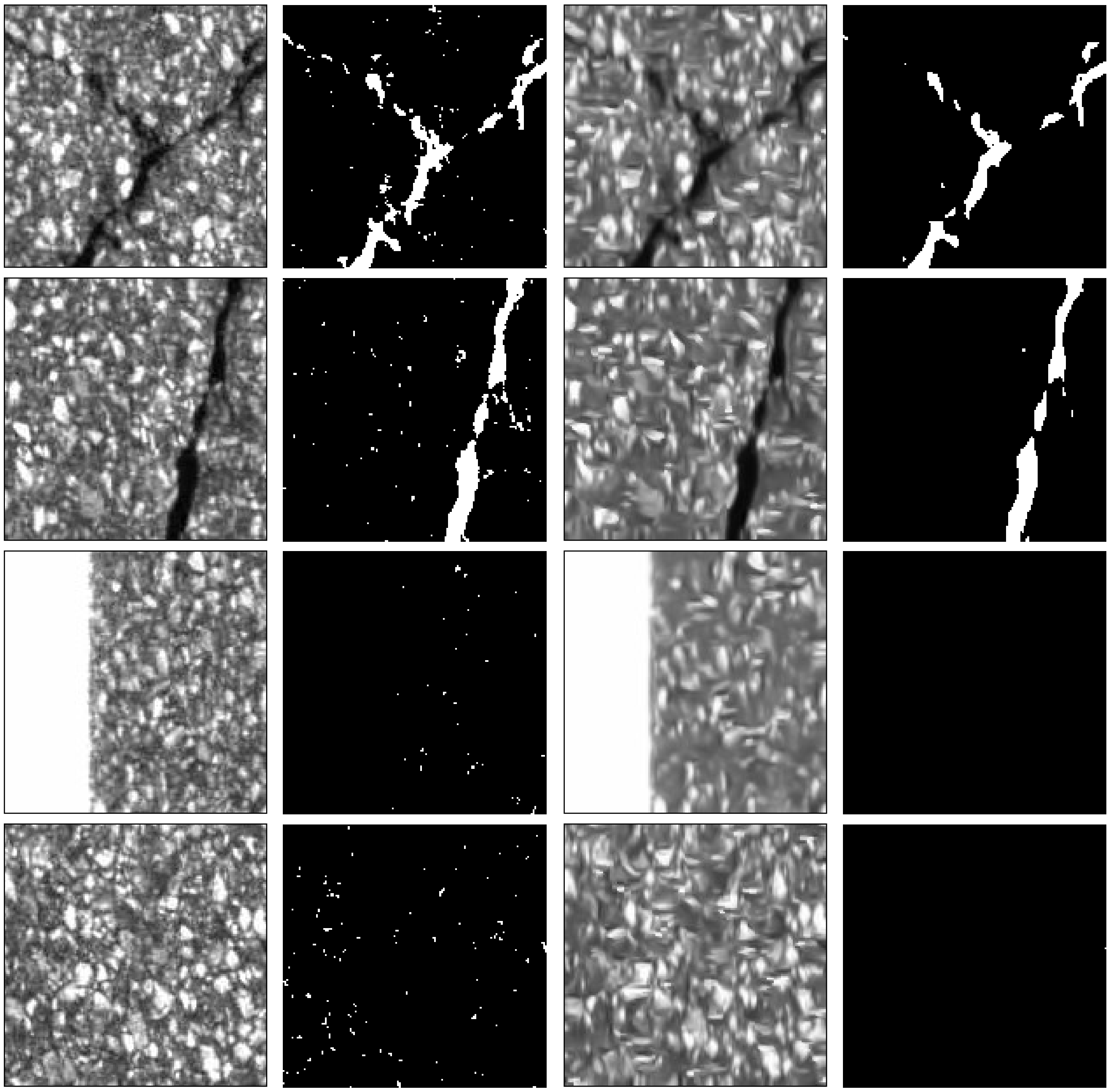


Fig. 5. Experimental sample results: original sample regions taken from original images, two showing: cracks, a white lane marking and without cracks (left column, from top to bottom); the respective segmentation results using *th* = 60, computed by means of the modified Otsu algorithm (2nd column); smoothed images using the proposed fast probability density estimation filtering (3rd column); the respective segmentation results (right column), again using *th* = 60 to better highlight the effects of smoothing, i.e. less number of no-crack regions.

In the present proposal, the parallel version of the algorithm was programmed using version 1.2 of the OpenCL framework standard. An AMD Radeon HD 7870 OC GPU was used for the implementation of the algorithm. The HD 7870 GPU has 20 CUs (Compute Units). Each CU can execute a different kernel program.

A maximum of 256 processing threads, with simultaneous concurrent execution are allowed, and they may be assigned to any subset of CUs. Due to the geometry of the images, with dimensions that are powers of 2, and the nature of the parallel version of the algorithm, 16 CUs were chosen for the parallel algorithm. The corresponding parallel processing architecture is shown in Figure 4. The expected speedup is proportional to the number of work-items (WIs), thus leading to a significantly reduction in processing time when compared to a single core based computation. The adopted number of WIs is the maximum allowed (256) for parallel execution on the HD 7870 GPU, corresponding to 16 CUs times 16 WIs. The same number of iterations as suggested in [4], i.e. 10 iterations per image, is adopted.

After the entropy reduction, a simple segmentation by thresholding can be performed using the intensity *th* computed according to proposal made in [12].

4. EXPERIMENTAL RESULTS

Experimental results are presented based on imagery acquired by INO's LRIS 4K model [1], taken during an experimental road pavement survey over Canadian roads.

The proposed implementation of UINTA was developed using the C programming language on the Linux OS and version 4.7.3 of the GNU C compiler, running on a single core of the AMD FX 8350 CPU. Typical processing times achieved are shown in Table 1.

Parallel processing times show a speedup of approximately 75 times when compared to the original single core UINTA version (see Table 1, for the comparison between single core and parallel processing times based on 16 CUs), referred to the FX 8350 CPU and the HD 7870 GPU. The execution times for the original and parallel versions are linearly proportional to the number of images pixels. Further improvements on computation times are foreseeable if a more complex kernel density estimation is pursued, aligned with references [10] and [11], but not very significant since parallel UINTA leads to few computations per pixel as shown in the pseudo code in Figure 3. These small improvements are related to the improvement of the inner *for* cycles in the kernel represented in Figure 3, by taking advantage of loop unrolling. Michailidis and Margaritis state in [10] that the performance gains of the two CUDA based optimized implementations of the kernel density estimation algorithm versus the naïve implementation are small. This naïve implementation is of the same kind as proposed in this paper, but with more steps, since UINTA has a gradient descent procedure that leads to some calculus simplifications. The computational efficiency achieved makes it possible the use of this entropy reduction approach for the smoothing of pavement surface images using a low-cost hardware platform.

Image size	**Processing times (sec.)**		
	Single	**16 CUs / 256 WIs**	**20 CUs / 240 WIs**
256×256	130	1.71	1.98
512×512	510	6.77	7.64
1024×1024	2079	26.71	30.22
2048×2048	8208	107.26	123.25
4096×4096	35016	467.75	544.46

Table 1. Computational times as a function of sample image size.

Sample results obtained by the segmentation module after smoothing using the proposed parallel implementation of UINTA are presented in the right column of Figure 5, which may be compared to those obtained using nonsmoothed images (shown on the 2nd column). Crack regions are shown as white regions while the remaining image regions do not contain crack pixels. These results provide a good match when visually evaluated by a road expert.

5. CONCLUSIONS AND FUTURE WORK

In this paper, a parallel implementation of UINTA filtering method that leads to a considerable computational time improvement (approximately 75 times), is proposed. The new implementation obtains exactly the same results of the original UINTA, while considerably improving the computational time required to apply the algorithm.

The parallelization of other smoothing methods using the same hardware devices is part of the planned future work.

REFERENCES

[1] INO, [Online], http://www.ino.ca/en/examples/laser-road-imaging-system-(lris)/, February 2014.

[2] H. Oliveira and P. Correia, "Automatic Road Crack Detection and Characterization," *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*, vol. 14, no. 1, pp. 155-168, March 2013.

[3] H. Oliveira, J. Caeiro and P.L. Correira, "Improved Road Crack Detection Based on One-class Parzen Density Estimation and Entropy Reduction," in *Proceedings of IEEE International Conference on Image Processing – ICIP2011*, Hong Kong 2010.

[4] S. Awate and R. Whitaker, "Unsupervised, Information-Theoretic, Adaptive Image Filtering for Image Restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 28, no. 3, pp. 364-376, March 2006.

[5] A. Buades, B. Coll, J.M. Morel, "A review of image denoising algorithms, with a new one," *in SIAM Journal on Multiscale Modeling and Simulation*, vol. 4, no. 2, pp. 490-530, 2005.

[6] A. Gray and A. Moore, "Nonparametric Density Estimation: Toward Computational Tractability," *in Proceedings of SIAM International Conference on Data Mining*, San Francisco, USA, May 2003.

[7] D. Lang, M. Klaas and N. Freitas, "Empirical Testing of Fast Kernel Density Estimation Algorithms, " *University of British Columbia, Department of Computer Science*, Technical Report TR-2005-03, 2003.

[8] B. Silverman, "Algorithm AS 176: Kernel density estimation using the fast fourier transform," *Applied Statistics*, vol. 31, pp. 93-99, 1982.

[9] A. Elgammal, "R. Duraiswami, L.S. Davis, "Efficient kernel density estimation using the fast gauss transform with applications to color modelling and tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1499-1504, 2003.

[10] P.D. Michailidis, K.G. Margaritis, "Accelerating kernel density estimation on the GPU using the CUDA framework," *Applied Mathematical Sciences*, vol. 7, no. 30, pp. 1447-1476, 2013.

[11] B.V. Srinivasan, Q. Hu, R. Duraiswami, "GPUML: Graphical processors for speeding up kernel machines," *Workshop on High Performance Analytics-Algorithms, Implementations and Applications*, 2010.

[12] Y. Wan and J. Wang, "A Modified Otsu Image Segmentation Method based on the Rayleigh Distribution," in *Proceedings of 3th IEEE International Conference on Computer Science and Information Technology*, July 2010.