# *Design Criteria for Real-time Processing of HW Gabor Filters in Visual Search*

G.D. Licciardo[1], C. Cappetta[2], L. Di Benedetto[3]
Dept. of Industrial Engineering (D.I.In.)
University of Salerno
Fisciano, Italy
gdlicciardo@unisa.it [1], ccappetta@unisa.it [2], ldibenedetto@unisa.it[3]

***Abstract*— Gabor filters gained a great importance in multimedia processing and visual search applications, thanks to the good spatial frequency and position selectivity, despite of their heavy computational complexity. Further, the large number of parameters to be imposed sets a number of trade-offs between accuracy and complexity making the use of Gabor filters very challenging. In this work, a number of criteria are exploited for a careful choice of the parameters, in order to allow implementing accurate two-dimensional filters, with a computational complexity that can be adapted to target platforms with different capabilities. In order to show this, three hardware designs of a Gabor filter-based edge detection system are derived, implementing two and four orientations, all capable of real-time processing with different Area-Delay-Power performances and accuracies. The derived designs have been implemented on a FPGA-based ASIC prototyping system and synthesized in 90nm CMOS std_cells, returning a maximum operating frequency of 179 *MHz* and 350 *MHz*, respectively. Therefore, the proposed filters achieve state-of-the-art performances with the best throughput of 86 and 168 Full-HD (1920x1080 pixels) frames-per-second, for FPGA and std_cell implementations, respectively.**

***Index Terms*— Gabor filters, Visual Search, Edge detection, Field Programmable Gate Arrays.***

## I. INTRODUCTION

Gabor filters (GF) form a class of linear, band-pass filters minimizing the joint uncertainty for frequency and spatial position [1]. As demonstrated in [2], this family of filters well describes the behavior of mammalian visual cortex cells, making them very attractive for computer vision [3]. Further, GF produce robust features in the detection of edges and corners from images [4], where the selection of specific angles in $[0, \pi]$ *rad* permits good selectivity along preferential orientations. Therefore, GF play a key role in edge and corner detection, segmentation, gait analysis and Visual Search (VS) [4]-[7]. In modern electronic systems, they usually compose the initial stage of processing nodes in specific neural and Internet of Things (IoT) distributed sensor networks [8]. However, the high computational complexity of such filters compels to hardware (HW) implementations if real-time operations are required, while ensuring a good accuracy of the results and low-power consumption. To the best of our knowledge, however, the recent literature presents very few HW oriented designs employing GF. In [9], [10] real-time operations are achieved by using reduced 3x3 Gabor-like kernels and one orientation only to implement a Cellular Neural Network (CNN). The implementation in [11] is strictly platform-dependent since the good performances are due to the use of high-frequency overclocked Digital Signal Processors (DSPs) embedded in modern Field Programmable Gate Arrays (FPGAs).

Performances of the GF and their computational complexity give raise to a number of trade-offs, which are regulated by the high number of parameters defining the kernels and the number of orientations. Gabor-like solutions usually approximate GF by significantly reducing accuracy in favor of acceptable performances. In this work, it is shown that a careful choice of the parameters allows implementing accurate two-dimensional (2D) GF, with a computational complexity that can be adapted also to resource constrained target platforms [12], with a marginal loss of accuracy. In order to show this, three different HW designs of a GF-based edge detection system are derived, implementing two and four orientations, all capable of real-time processing with different Area-Delay-Power (ADP) performances and accuracies. The derived Application Specific Image Processors (ASIPs) are targeted to both FPGA and Application Specific Integrated Circuit (ASIC), using CMOS 90nm standard cells (std_cell). The best-case maximum path delay is 5.6 *ns* on a Virtex 7 based ASIC prototyping board [13], which allows processing a Full-HD frame in 11.6 *ms*. Quite halved delays are obtained from std_cell implementation.

## II. IMPLEMENTED KERNEL AND DESIGN CONSTRAINS

2D GF are obtained by multiplying a generic 2D Gaussian and a complex exponential [4]:

$$\psi(x, y, f_0, \theta) = \frac{f_0^2}{\pi\gamma\eta} e^{-\left\{\frac{f_0^2}{\gamma^2}x'^2 + \frac{f_0^2}{\eta^2}y'^2\right\}} e^{j2\pi f_0 x'} \tag{1}$$

where, $f_0$ is the central frequency, $\theta$ is the rotation angle of the filter, $\gamma$ and $\eta$ are the variances of the Gaussian projections along the major and minor rotated axis, $x'$ and $y'$. In turn, $x'$ and $y'$ are the rotated coordinates, defined as $\begin{cases} x' = x\cos\theta + y\sin\theta \\ y' = -x\sin\theta + y\cos\theta \end{cases}$, where $x$ and $y$ represent the coordinates of the reference system. Features obtained from GF benefit from the properties of GF in conjunction with directional and frequency selectivity introduced by the periodic component [4]. Therefore, the obtained features are stable against affine transformations, frequency scaling and resolution changes. Consequently, Gabor features are prone to reject noise and object deformations. As usually done in VS applications, the scale invariance property is fully exploited by multiscale designs [5], [14]. Furthermore, the accuracy of GF can be improved by considering at least two orientations for each scale, each one giving rise to a different set of kernel coefficients. Edges are generally identified by a max pooling or sum pooling of the outputs from all the orientations and scales [5], [15]. Moreover, only the odd imaginary part of GF will be processed, since it represents a good approximation of the second derivative of its step edge response [5], [15], [16]. Therefore, an accurate implementation of GF as edge detector can be approximated as:

$$G(x,y,f_0,\theta)=\frac{f_0^2}{\pi\gamma\eta}e^{-\left\{\frac{f_0^2}{\gamma^2}x'^2+\frac{f_0^2}{\eta^2}y'^2\right\}}\sin(2\pi f_0 x') \qquad (2)$$

with a consequent reduction in complexity and in the number of implemented resources. In the following, the design parameters determining the behavior of the filters and the way they are selected in the proposed implementations to limit the computational complexity are presented.

The central frequency, $f_0$, must be lower than 0.5 to satisfy the Nyquist criterion. However, a more effective range is restricted to [0.1, 0.2], since lower frequencies fail to follow sharp edges, causing excessive blurring during the image processing, while $f_0$>0.2 would require an unwanted oversampling of the kernel with respect to the resolution grid of the input image [6]. In our designs, $f_0$=0.2 has been chosen to satisfy the above constrains while minimizing the blurring effect.

The $\gamma$ and $\eta$ parameters correspond to the standard deviations of the Gaussian envelope along *x'* and the Gaussian projection of the GF along *y'*, respectively. Although precise constrains do not exist to select their values, in order to avoid blurring problems on one hand and an excessive sensitivity of the GF to small variations into the images on the other hand, their values must be related to the kernel dimensions, $N_S$. These relations are generally determined by the subjective observation of the results, so that they can change depending on the quality of the input set. In our designs, however, $\gamma$=1 has been imposed to include at least one period of the periodic component in the Gaussian envelope, to ensure the presence of the odd function needed for edge detection [16]. The Gaussian shape has been adequately approximated by imposing the minimum filter dimensions to 6$\gamma$+1 [17], giving rise to a filter having dimensions 7x7. In our design, this filter has been combined to a 9x9 one to implement a multiscale architecture improving the accuracy and stability of the results. In turn, $\eta$=15 has been imposed in order to simplify (2) in a HW friendly fashion as explained in the next section. Results have been verified providing a good perceptional quality, as shown in Fig. 1.

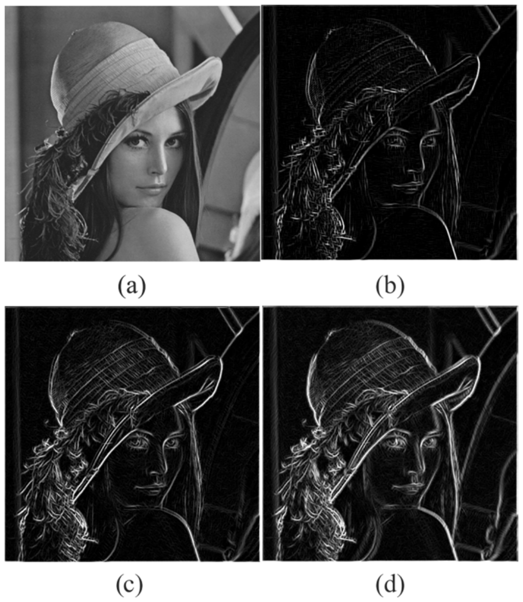

Fig. 1: a) Original *Lena* image and results for $f_0=0.2$, $\gamma=1$, $\eta=15$ considering 7x7 GF and (b) two, (c) four and (d) eight orientations.

TABLE I
PSNR AND SSIM VALUES OBTAINED COMPARING THE 2OR AND 4OR RESULTS WITH THE 8OR CASE

| Image | 2Or | | 4Or | |
|---|---|---|---|---|
| | *PSNR [dB]* | *SSIM* | *PSNR [dB]* | *SSIM* |
| airplane | 21.90 | 0.72 | 25.16 | 0.87 |
| baboon | 20.11 | 0.58 | 25.47 | 0.86 |
| boat | 23.40 | 0.68 | 26.78 | 0.84 |
| Lena | 23.42 | 0.66 | 27.56 | 0.88 |
| peppers | 23.49 | 0.68 | 26.00 | 0.84 |
| Average | 22.46 | 0.66 | 26.19 | 0.86 |

The number of orientations is a serious trade-off between the accuracy of the results and the computational complexity of GF. With more orientations, the impossibility to separate GF makes the number of operators quadratically dependent from the filter dimensions, $N_S$, thus making very difficult the implementation of HW designs with more than four orientations (*4Or*). As an example, a raw estimation of the area required for an eight orientations (*8Or*) design with $f_0=0.2$, $\gamma=1$, $\eta=15$ and 7x7 and 9x9 scales returns more than 7 $mm^2$ in CMOS 90*nm* and about 240k LUTs in FPGA. In both cases, the dimensions are unacceptable for a usable implementation. However, in order to show that suitable results can be obtained from *2Or* and *4Or*, in Table I these are compared with results from *8Or* in terms of Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM). The *2Or* reduces the SSIM to 0.66, which could result insufficient for some applications but highly desirable for resource-constrained devices, while *4Or* presents a SSIM=0.86 and a more than acceptable PSNR value for the large part of applications.

## III. NUMERICAL REMARKS

Our HW friendly design implements a Fixed Point (FI) coding of the final and intermediate results to reduce the complexity of the mapped arithmetic operators with respect to the IEEE-754 32 bits floating-point (FP32) format. In order to prevent the loss of accuracy arising from this choice, starting from the considerations in [14] for a 2D Gaussian filter, a modeling of the worst case scenario has been developed, allowing to choose the minimum acceptable codelength for the final results. The maximum errors of the Gabor coefficients are reported in Table II as a function of their codelenght. The first two bits represent the sign and the integer part respectively, while the length of the fractional part varies. By imposing a 16 *bits* code (FI16), the error is bounded to about 3x10$^{-5}$, namely to the Least Significant Bit (LSB). To evaluate the error

TABLE II
MAXIMUM ERROR ON THE GABOR FILTER COEFFICIENTS VARYING THE FI CODELENGTH FOR THE CHOSEN SET OF PARAMETERS

| Number of Bits | Maximum Error |
|---|---|
| 14 | 1.04E-04 |
| 15 | 5.95E-05 |
| **16** | **2.98E-05** |
| 17 | 1.27E-05 |
| 18 | 7.58E-06 |

propagation up to the final result, the worst case for each orientation has been calculated by summing only the positive kernel coefficients multiplied by the maximum input value (255). Results are reported in Table III. The shortest code of the final result for which the error is lower than the LSB is given by 12 *bits*, where 8 and 4 *bits* are considered for the integer and fractional part, respectively. Only positive results are considered.

## IV. ARCHITECTURE DESIGN

In this work, a *2Or* and a *4Or* designs have been implemented, working on $\theta_{Two} = \{0, \pi/2\}$ and $\theta_{Four} = \{0, \pi/4, \pi/2, 3\pi/4\}$ orientation sets, respectively. A third design (*4Or*-Shared) has been derived from the *4Or* by implementing a resource sharing architecture to obtain a better accuracy-area trade-off. The general architecture is schematized in Fig. 2.

### A. Gabor Coefficients Memories

Coefficients of GF are stored in devoted memories. Their dimensions can be reduced by observing that, for the symmetry of (2), $G_{i,j}^{\theta} = G_{j,i}^{\theta+\pi/2}$ for $i, j \in I$ and $\theta \in [0, \pi/2]$, where $G_{i,j}^{\theta} = G(i, j, f_0, \theta)$. Further, for $\theta = 0$ and $\theta = \pi/2$, all the rows and the columns coincide, respectively. Therefore, only $N_S$ coefficients must be stored. In turn, for $\theta = \pi/4$ and $\theta = 3\pi/4$, the symmetry develops along the minor and major diagonal of the kernel matrixes, respectively and only $(2N_S - 1)$ coefficients must be stored. Therefore, considering the FI16 coding two memories of 112 *bits* and 208 *bits* must be instantiated when $N_S$=7; while for $N_S$=9, they increase to 144

TABLE III
MAXIMUM ERROR ON THE FILTERING RESULTS FOR THE CHOSEN CODELENGTH OF THE GABOR COEFFICIENTS

| Number of Bits | Maximum Error | LSB |
|---|---|---|
| 10 | 1.05E-01 | 2.50E-01 |
| 11 | 4.95E-02 | 1.25E-01 |
| **12** | **3.57E-02** | **6.25E-02** |
| 13 | 3.57E-02 | 3.13E-02 |
| 14 | 2.00E-02 | 1.56E-02 |

*bits* and 272 *bits*. Fig. 3(a) reports the memory requirements for different scales, varying the number of orientations.

### B. Memory Module

The 8 *bits* Luma components of the incoming pixels are acquired in raster scan order from an image source (e.g. image sensor) and stored in a Memory Module (MM), arranged to operate like a long FIFO, having overall dimensions $W$x$N_S^{max}$, being $W$ the width of the input image and $N_S^{max}$ the dimension of the greatest kernel. For its implementation, $N_S^{max}$ dual-port RAMs have been instantiated, one for each row, having dimensions 1x($W$-$N_S^{max}$), while a bank of $N_S^{max}$x$N_S^{max}$ registers terminates the rows to make the pixels available in parallel to the Arithmetic Unit [17], [18].

### C. Arithmetic Unit and Max Pooling Unit

The Arithmetic Unit (AU) takes the data coming from the Gabor Coefficients Memories and the MM and process the filtering operations for each orientation. This unit exploits the separability property of (2) along the directions $\theta_{Two}$ as:

$$\begin{cases} G_0 = G(x, y, f_0, 0) = \dfrac{f_0^2}{\pi\gamma\eta}\sin(2\pi f_0 x)\, e^{-\frac{f_0^2}{\gamma^2}x^2} \times e^{-\frac{f_0^2}{\eta^2}y^2} = \\ \quad = G_0^H \times G_0^V \\ G_{\pi/2} = G\left(x, y, f_0, \dfrac{\pi}{2}\right) = \dfrac{f_0^2}{\pi\gamma\eta}\sin(2\pi f_0 y)\, e^{-\frac{f_0^2}{\gamma^2}y^2} \times e^{-\frac{f_0^2}{\eta^2}x^2} = \\ \quad = G_{\pi/2}^V \times G_{\pi/2}^H \end{cases} \tag{3}$$

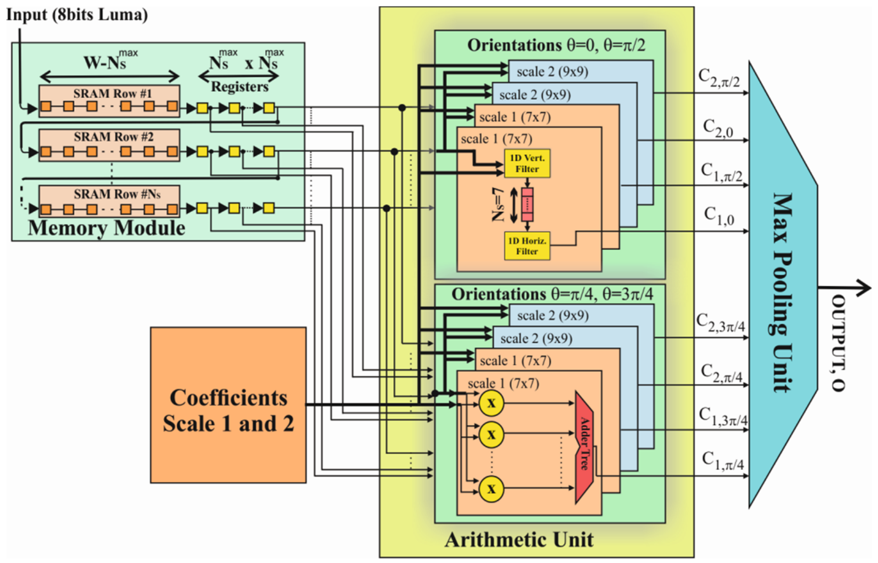

Fig. 2: General block scheme of the proposed designs.

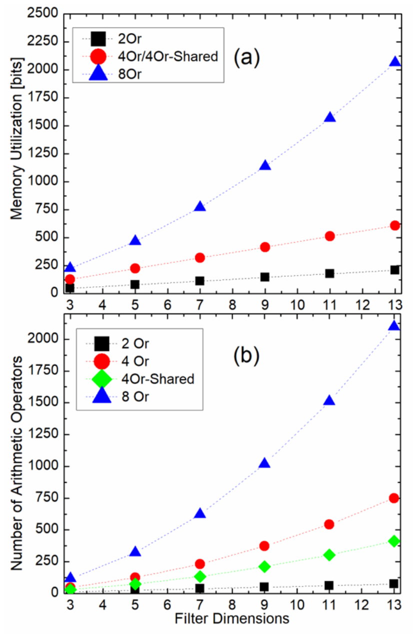

Fig. 3: (a) Memory utilization in *bits* and (b) Number of the total arithmetic units as sum of adders and multipliers, varying the filter dimensions.

reducing the complexity along these directions from O($N_s{}^2$) to O($N_s$). Furthermore, the choice $\eta = 15$ simplifies (3) as:

$$\begin{cases} G_0 = G_0^V \times G_0^H = I^T \times G_0^H \\ G_{\pi/2} = G_{\pi/2}^V \times G_{\pi/2}^H = G_{\pi/2}^V \times I \end{cases} \quad (4)$$

where $I = [1,1,...,1]$. On the contrary, the impossibility to separate the kernel along the remaining orientations is the main reason of the increment of arithmetic components. A more general analysis is shown in Fig. 3(b) where the number of total adders and multipliers are reported as a function of the filter dimensions and of the number of orientations. The *4Or-Shared* design presents a reduced arithmetic unit, obtained by using one shared structure to calculate both the $\pi/4$ *rad* and the $3\pi/4$ *rad* orientations. Therefore the 4 sub-modules depicted in Fig. 2 (Orientations $\theta = \pi/4$, $\theta = 3\pi/4$) reduce to 2. This implementation reduces the number of adders and multipliers by 41% and 44.5% respectively, with respect to the *4Or* design. The cost to pay is the increase of the latency, since the *4Or-Shared* design needs two clock cycles to filter a pixel along all the orientations for each scale. The output Max Pooling Unit (MPU) simply calculates

TABLE IV
SYNTHESIS OF THE PROPOSED DESIGNS IN STD_CELLS

| | ***Std_cells*** | | |
|---|---|---|---|
| | *2Or* | *4Or* | *4Or-Shared* |
| Platform | 90nm | 90nm | 90nm |
| Area [µm$^2$] | 1002929 | 2414128 | 1801557 |
| Delay[ns] | 2.859 | 3.374 | 6.830 |
| Power[a] [mW] | 14.013 | 76.453 | 27.135 |
| fps[b] | 168 | 142 | 70 |

[a.] Normalized at 100MHz [b.] Full-HD frames

$$O = Max\left\{C_{1,0}, C_{1,\pi/2}, C_{2,0}, C_{2,\pi/2}, C_{1,\pi/4}, C_{1,3\pi/4}, C_{2,\pi/4}, C_{2,3\pi/4}\right\},$$

namely the greatest value coming from the AU for each pixel as a feature candidate to compose the final output.

## V. Synthesis and Results

The designs have been targeted to TSMC CMOS 90nm std_cells and to a Xilinx Virtex 7 XC7V2000tflg1925-1, as part of the proFPGA DUO ASIC prototyping board obtaining the results reported in Tables IV and V, respectively. The proposed *2Or* design exhibits a delay time reduction of 16.8% with respect to the 3x3 Gabor-like simplified filter in [9], although, it maps 71.3% and 85.9% less LUTs and FFs respectively. The *4Or* design uses 92.8% more LUTs and 28.7% more DSP units than [9]. The *4Or-Shared* design, having the same accuracy of 4*Or*, saves the 27.7% DSPs and 41.8% FFs with respect to [9], at the cost of a 58.9% overhead in terms of occupied LUTs, which has the same accuracy of *4Or*. The performances in terms of frames-per-second (fps) show that the proposed designs achieve real-time performances, even in the case of the slowest implemented architecture, achieving a worst-case throughput of 48 fps.

## VI. Conclusions and Future Works

The work proposes three different designs to obtain an HW accelerator for edge detection and VS applications in real-time using GF. The architectures match different ADP constraints. The proposed ASIPs achieve state-of-the-art performances both on FPGA and std_cell implementations thanks to the developed HW friendly implementation. Future developments on the proposed architectures will regard the possibility of Distributed Arithmetic techniques to simplify the arithmetic circuitries [18], [19], in order to implement more orientations or more scales, or to target the designs on more resource-constrained devices. Moreover, the application of approximate computing [20] or the use of CORDIC algorithm [21] could improve the proposed designs, in terms of resource usage and power consumption. These techniques could also be used to implement Neural Network architectures [8], achieving adaptive filtering according to the incoming pixels.

TABLE V
SYNTHESIS OF THE PROPOSED DESIGNS ON FPGA PLATFORM

| | **FPGA** | | | | | | |
|---|---|---|---|---|---|---|---|
| | *With DSPs* | | | | *Without DSPs* | | |
| | *2Or* | *4Or* | *4Or-Shared* | *Cesur [9]* | *2Or* | *4Or* | *4Or-Shared* |
| Target platform | Virtex 7 | Virtex 7 | Virtex 7 | Stratix 4 | Virtex 7 | Virtex 7 | Virtex 7 |
| LUTs | 4022 | 27042 | 22281 | 14025 | 8233 | 59930 | 41265 |
| FFs | 2864 | 16220 | 11833 | 20321 | 3440 | 21812 | 14912 |
| DSPs | 32 | 260 | 146 | 202 | -- | -- | -- |
| Path Delay[ns] | 5.600 | 6.400 | 10.000 | 6.734 | 5.600 | 6.400 | 10.000 |
| Power[a] [W] | 0.775 | 1.629 | 1.748 | -- | 0.821 | 1.898 | 2.327 |
| fps[b] | 86 | 75 | 48 | 71 | 86 | 75 | 48 |

[a.] Normalized at 100MHz [b.] Full-HD frames

## REFERENCES

[1] D. Gabor, "Theory of Communication. Part 1: The Analysis of Information", *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, Vol. 93, No. 26, pp. 429 – 441, Nov. 1946.

[2] J. G. Daugman, "Uncertainty Relation for Resolution in Space, Spatial Frequency, and Orientation Optimized by Two-Dimensional Visual Cortical Filters", *Journal of Optical Society of America A*, Vol. 2, No. 7, pp. 1160 – 1169, Jul. 1985.

[3] J.M. Guo, H. Prasetyo and K. Wong, "Vehicle Verification Using Gabor Filter Magnitude with Gamma Distribution Modeling", *IEEE Signal Processing Letters*, Vol. 21, No. 5, pp. 600 – 604, Mar. 2014.

[4] J.-K. Kamarainen, V. Kyrki and H. Kalviainen, "Invariance Properties of Gabor Filter-Based Features – Overview and Applications", *Image Processing, IEEE Trans. on*, Vol. 15, No. 5, pp. 1088-1099, Apr. 2006.

[5] W. Jiang, K.-M. Lam and T. Z. Shen, "Efficient Edge Detection Using Simplified Gabor Wavelets", *Man, and Cybernetics, Part B (Cybernetics), IEEE Transactions on Systems*, Vol. 39, No. 4, pp. 1036 – 1047, Aug. 2009.

[6] W.-C. Zhang, F.-P. Wang, T. L. Zhu and Z.-F. Zhou, "Corner Detection using Gabor Filters", *IET Image Processing*, Vol. 8, No. 11, pp. 639 – 646, Nov. 2014.

[7] H. Hu, "Enhanced Gabor Feature Based Classification Using a Regularized Locally Tensor Discriminant Model for Multiview Gait Recognition", *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 22, No. 7, pp. 1274 – 1286, Jul. 2013.

[8] B. Kwolek , "Face Detection Using Convolutional Neural Networks and Gabor Filters", *Artificial Neural Networks: Biological Inspirations – ICANN 2005*, pp. 551 – 556, Springer, Berlin Heidelberg, D, Sep. 2005.

[9] E. Cesur, N. Yildiz and V. Tavsanoglu, "An Improved FPGA Implementation of CNN Gabor-Type Filters", *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, Rio de Janeiro, BR, Jul. 2011.

[10] E. Cesur, N. Yildiz and V. Tavsanoglu, "On an Improved FPGA Implementation of CNN-Based Gabor-Type Filters", *Circuits and Systems II: Express Briefs, IEEE Transactions on*, Vol. 59, No. 11, pp. 815 – 819, Oct. 2012.

[11] Y. C. P. Cho, N. Chandramoorthy, K. M. Irick and V. Narayanan, "Multiresolution Gabor Feature Extraction for Real Time Applications", *Signal Processing Systems (SiPS), 2012 IEEE Workshop on*, Quebec City, CDN, Oct. 2013.

[12] G. D. Licciardo and M. Costagliola, "An H.264 Encoder for Real Time Video Processing Designed for SPEAr Customizable System-on-Chip Family," *2007 IEEE International Conference on Signal Processing and Communications*, Dubai, 2007, pp. 824-827.

[13] *Virtex - 7 Family, DS183 (v1.23), Xilinx, San Jose, CA, USA, Jun. 23, 2015.*

[14] G.D. Licciardo, T. Boesch, D. Pau and L. Di Benedetto , "Frame Buffer-less Stream Processor for Accurate Real-Time Interest Point Detection", *Integration, the VLSI Journal*, Vol. 54, pp. 10 – 23, Jun. 2016.

[15] F. Pellegrino, W. Vanzella and V. Torre, "Edge Detection Revisited", *Systems, Man, and Cybernetics – Part B: Cybernetics, IEEE Trans. on,* Vol. 34, No. 3, pp. 1500–1518, Jun. 2004.

[16] R. Mehrotra, K. R. Namuduri and R. Ranganathan, "Gabor Filter-based Edge Detection", *Pattern Recognition,* Vol. 25, No. 12, pp. 1479–1494, Dec. 1992.

[17] G. D. Licciardo, A. D'Arienzo and A. Rubino, "Stream processor for real - time inverse Tone Mapping of Full - HD images", in *IEEE Trans. on VLSI Systems,* Vol.23, No. 11, pp. 2531–2539, Nov. 2015.

[18] *G. D. Licciardo, C. Cappetta, L. Di Benedetto, A. Rubino and R. Liguori, "Multiplier-Less Stream Processor for 2D Filtering in Visual Search Applications," in Circuits and Systems for Video Technology, IEEE Trans. on, vol. 28, no. 1, pp. 267-272, Jan. 2018.*

*[19] G.D. Licciardo, C. Cappetta, L. Di Benedetto and M. Vigliar, "Weighted Partitioning for Fast Multiplier-less Multiple Constant Convolution Circuit", Circuits and Systems II: Express Briefs, IEEE Trans. on, Vol. 64, No. 1, pp. 66 – 70, Jan. 2017.*

[20] *S. Venkatachalam and S.-B. Ko, "Design of Power and Area Efficient Approximate Multipliers", Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, Vol. 25, No. 5, pp. 1782 – 1786, Jan. 2017.*

[21] *K. Radhakrishnan, Devi Sri, R. Dhanalakshmi, M. Elakkiya and G. Gayathiri, "Design and implementation of high speed Gabor filter with variable thresholding process for disease detection", Sensing, Signal Processing and Security (ICSSS), 2017 Third International Conference on, Chennai, IND, May 2017.*