Journal of Electrical and Computer Engineering Innovations
(JECEI)
Journal homepage: http://www.jecei.sru.ac.ir

Research paper

Fast and Efficient Hardware Implementation of 2D Gabor Filter for a Biologically-Inspired Visual Processing Algorithm

A. Mohammadi Anbaran[1], P. Torkzadeh[1,*], R. Ebrahimpour[2, 4], N. Bagheri[3, 5]

[1]*Electrical Engineering Department, Faculty of Engineering, Islamic Azad University, Science and Research Branch, Tehran, Iran.*

[2]*Artificial Intelligence Department, Faculty of Computer Engineering, Shahid Rajaee Teacher Training University; Tehran, Iran.*

[3]*Communication Engineering Department, Faculty of Electrical Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran.*

[4]*School of Cognitive Sciences, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran.*

[5]*School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran.*

Article Info

Article History:
Received 24 May 2020
Reviewed 27 July 2020
Revised 22 September 2020
Accepted 18 November 2020

Keywords:
Gabor filter
FPGA
Separable filter
Convolution
HMAX model

*Corresponding Author's Email Address:
p-torkzadeh@srbiau.ac.ir

Abstract

Background and Objectives: Programmable logic devices, such as Field Programmable Gate Arrays, are well-suited for implementing biologically-inspired visual processing algorithms and among those algorithms is HMAX model. This model mimics the feedforward path of object recognition in the visual cortex.

Methods: HMAX includes several layers and its most computation intensive stage could be the S1 layer which applies 64 2D Gabor filters with various scales and orientations on the input image. A Gabor filter is the product of a Gaussian window and a sinusoid function. Using the separability property in the Gabor filter in the 0° and 90° directions and assuming the isotropic filter in the 45° and 135° directions, a 2D Gabor filter converts to two more efficient 1D filters.

Results: The current paper presents a novel hardware architecture for the S1 layer of the HMAX model, in which a 1D Gabor filter is utilized twice to create a 2D filter. Using the even or odd symmetry properties in the Gabor filter coefficients reduce the required number of multipliers by about 50%. The normalization value in every input image location is also calculated simultaneously. The implementation of this architecture on the Xilinx Virtex-6 family shows a 2.83ms delay for a 128×128 pixel input image that is a 1.86X-speedup relative to the last best implementation.

Conclusion: In this study, a hardware architecture is proposed to realize the S1 layer of the HMAX model. Using the property of separability and symmetry in filter coefficients saves significant resources, especially in DSP48 blocks.

Introduction

Feature extraction in object recognition is performed in the primary visual cortex (V1) in the mammalian visual pathway. The V1 simple cells behavior modeling discovered by Hubel and Wiesel [1] is generally performed by 2 dimensional (2D) Gabor filters. A Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. A pyramid of 2D Gabor filters is a common approach for modeling classical simple cells (S1

layer) of HMAX (Hierarchical model and X) model [2], [3]. HMAX is a computational model of the ventral visual pathway found within the visual cortex, which is responsible for object recognition with an admissible performance. In S1 layer of the HMAX model, which is one of the computationally intensive stages of the model, the real part of the 2D Gabor filter is applied to the input image in four orientations and 16 scales. Applying different filters increases the robustness of the model for changes of object orientation and scale in the input image. This is performed in the S1 layer of the model, which is one of the computationally intensive stages of the model.

In addition to biologically-inspired systems, the 2D Gabor filter is employed in many other applications, such as texture classification, facial expression recognition techniques [4], edge detection [5], and iris recognition [6]. For this reason, speedup in applying a 2D Gabor filter to an image is particularly critical in real-time applications.

Many articles have been written in the last decade with the aim of acceleration of applying Gabor 2D filter to an input image. In [7], a technique has been proposed to integrate the interpolation and the convolution processes of the Gabor filter. The integration of these two processes makes the 2D Gabor filter separable along any direction.

Separating the filter in two dimensions, x and y, converts the 2D filter to two more efficient 1D filters with Gaussian and sinusoidal modulations. However, this method requires a technique for re-sampling an image by an interpolation kernel, a process which involves additional computational complexity.

To convert convolution to multiplication, some articles have used frequency domain and Fourier transforms. To obtain the final output, [8] first transforms input data and filters to Winograd or frequency domain, performs element-wise multiplication, and then applies inverse transformation. This study proposes a novel architecture for implementing fast algorithms on FPGAs.

Utilizing some mathematical techniques in mathematical relationships reduces computation overhead without any loss in accuracy. Some recent studies have computed filters as linear combinations of a smaller number of separable filters, thus greatly reducing computational complexity at no cost in terms of performance [9], [10].

In this study, a filter with rank R converts to an R separable filter or R filter with rank one. Separation in a 2D filter indicates that separation may be achieved by applying two or more 1D filters, which greatly reduces the run-time and computational resource requirements without a loss in accuracy. The idea of separating Gabor filter kernels is also introduced in [11], but the suggested method only accommodates particularly oriented Gabor filters and so is not generic or flexible enough.

Reference [12] proposes GPU acceleration of the texture feature extraction algorithm by using separable 1D Gabor filters to approximate non-separable Gabor filter kernels.

The complexity and severity of computations in the Gabor filter's real-time applications, such as S1 layer of HMAX, has always posed a challenge in filter implementation. In recent years, researchers have been fascinated by an effective approach to overcoming this challenge: the design of hardware accelerators, which enable massive parallel processing and pipelining [13]-[15].

In addition, FPGA-based accelerators provide fast programming times and cost-effective ways for evaluating algorithms and prototyping, which eliminate fabrication time [16]. Furthermore, FPGA fabricators have developed IP Cores, such as CORDIC Cores, Memory Cores, and math calculation Cores, for instance rooting and power, which make designing easier, faster, and more optimal.

A. Our Contributions

As the main contribution of this paper, we proposed a resource efficient version of the S1 layer of the HMAX model for hardware implementation. More precisely, the proposed model has the following properties:

- The 2D Gabor filter is separable in 0° and 90° directions, and in other directions the Gabor filters becomes separable by approximating them using its isotropic (γ = 1,circular) version. In order to reduce the complexity of computation, separation and symmetry properties have been used in the isotropic version of 2D Gabor filters, which is effectively raises the filtering speed.
- This method is employed in the design of hardware architecture for accelerating the S1 layer of the HMAX model. The effect of this approximation (i.e. γ = 1) on the accuracy of the HMAX model has been investigated in [17] and it has been shown that it has no effect on its accuracy.
- Pipeline hardware architecture for the proposed S1 layer was designed and implemented on one Virtex-6 FPGA family. By utilizing the separation and symmetry property, there is a desirable reduction in hardware resources, especially in DSP48E1 and memory blocks.
- The normalization value of the results is calculated in parallel with the filter to increase the illumination invariance.

B. Paper Organization

The organization of the present paper is as follows. The second section reviews the S1 layer of a HMAX

model of a classical simple cell emulating V1 simple cell. In the third Section discusses the mathematical relations of the separability and symmetric coefficient of a Gabor filter in special orientations. The next section presents the overall hardware architecture of the S1 layer of the HMAX model according to the contents of the previous section. The next section presents the simulations of the modified model.

The following section provides the results of the proposed architecture and a comparison with a state-of-the-art implementation. Finally, the conclusion of the work is presented.

S1 Layer of HMAX

The S1 layer is the lowest layer of the HMAX model and receives a gray value image as its input. This input image is then applied to a set of Gabor filters as an edge detector filter. The Gabor filters fit very well with the receptive field weight functions found in the simple cells of the primary visual cortex. The following equations describe the real part of the two-dimensional Gabor filter [3]:

$$G(x,y) = e^{-\frac{X^2+\gamma^2Y^2}{2\sigma^2}}.cos\left(\frac{2\pi}{\lambda}X\right)$$
$$X = x\,cos\,\theta + y\,sin\,\theta \quad (1)$$
$$Y = -x\,sin\,\theta + y\,cos\,\theta$$

x and y are determined by the filter size, which spans a range of sizes from 7×7 to 37×37 pixels in the steps of the two pixels. Table 1 of [2] presents all filter parameters, i.e., the aspect ratio, γ=0.3, orientation θ (0°, 45°, 90°, and 135°), effective width σ, and wavelength λ. Thus, the complete pyramid consists of 4×16=64 filters, leading to 64 different S1 receptive field types (four orientations and 16 scales). Figure 1 shows the 2D Gabor filter bank.

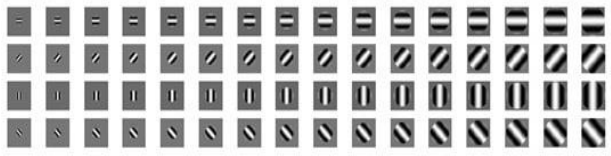


Fig. 1: The 2D Gabor filter bank with four orientations and 16 scales [17].

In the S1 layer of the HMAX model, the 64 Gabor filters should be applied to the input image and then the results should be normalized for illumination invariance. Normalization values are obtained by calculating the root sum square values of the input image pixels at each location where the Gabor filter is applied. Figure 2 provides the pseudo code for computing the S1 layer response and Fig. 3 shows the result of the S1 layer in various orientations and scales of the Gabor filter.

```
For each scale, s = 7 : 2 : 37
    For each orientation, Θ = 0 : 45 : 135
        Extract s×s Gabor Filter in Θ orientation from
memory
        Computing the 2D convolution between the input
Image and the Gabor Filter
        Computing the Normalization value of the Image in
the s×s region
    end
end
result <= 2D convolution / Normalization value
```

Fig. 2: Pseudo code of the S1 layer.

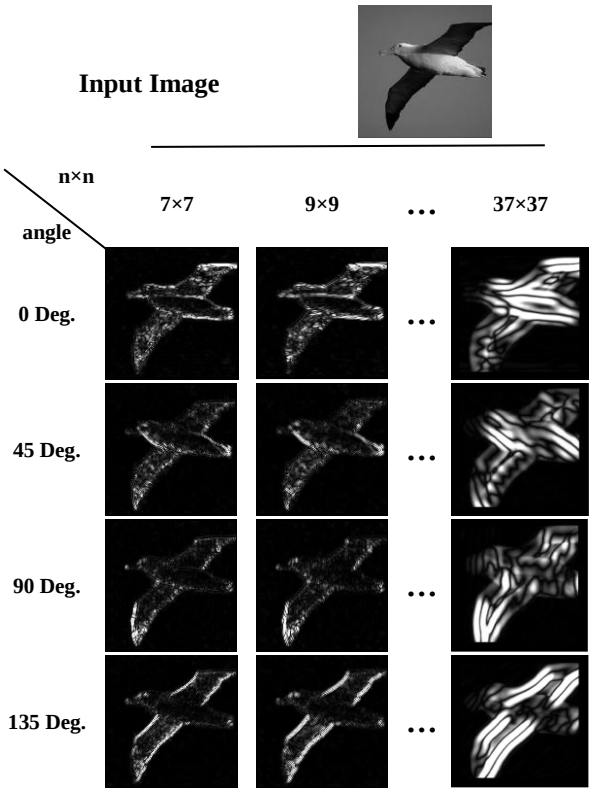


Fig. 3: Applying Gabor filters in 4 orientations and in 16 scales.

Mathematical Relations

The conventional way to apply the Gabor filters in the S1 layer is convolution in the spatial domain. The complexity of convolution depends directly on M×N and P, where P is the width and height of the filter and M and N are the width and height of the image, respectively.

The complexity of calculating the filter response for one location is P^2 and for the entire image is MNP^2.

A filter is called separable if it can be expressed as the multiplication of two column and row vectors. Therefore, the convolution in the separable filters can be performed by two one-dimensional filters.

Consequently, the computational complexity decreases to 2PMN or P/2 times.

$$G(x,y) = G_1(x) * G_2(y)$$

$$\begin{aligned} Out(x,y) &= G(x,y) * In(x,y) \\ &= [G_1(x) * G_2(y)] * In(x,y) \\ &= G_1(x) * [G_2(y) * In(x,y)] \end{aligned} \quad (2)$$

where the operator * denote to the convolution, G(x,y) is the 2D Gabor filter and G_1 and G_2 are 1D Gabor filters by column and row representation. Out(x,y) and In(x,y) are the input image and the output of the filter, respectively.

Solving (1 in 0° and 90° shows that the Gabor filter is separable in these orientations.

$$\begin{aligned} G_{\theta=0}(x,y) &= e^{-\frac{x^2+\gamma^2y^2}{2\sigma^2}}.\cos\left(\frac{2\pi}{\lambda}x\right) \\ &= e^{-\frac{\gamma^2y^2}{2\sigma^2}}.\left[e^{-\frac{x^2}{2\sigma^2}}\cos\left(\frac{2\pi}{\lambda}x\right)\right] \\ &= {E_2}^T(y) * E_1(x) \end{aligned} \quad (3)$$

$$\begin{aligned} G_{\theta=90}(x,y) &= e^{-\frac{y^2+\gamma^2x^2}{2\sigma^2}}.\cos\left(\frac{2\pi}{\lambda}y\right) \\ &= \left[e^{-\frac{y^2}{2\sigma^2}}\cos\left(\frac{2\pi}{\lambda}y\right)\right].e^{-\frac{\gamma^2x^2}{2\sigma^2}} \\ &= {E_1}^T(y) * E_2(x) \end{aligned}$$

in which:

$$\begin{aligned} E_1(x) &= e^{-\frac{x^2}{2\sigma^2}}\cos\left(\frac{2\pi}{\lambda}x\right) \\ E_2(y) &= e^{-\frac{\gamma^2y}{2\sigma^2}} \end{aligned} \quad (4)$$

where ${E_1}^T(y)$, ${E_2}^T(y)$ and $E_1(x)$, $E_2(x)$ are column and row vectors, respectively, and the * sign designates the convolution of the two column and row vectors.

In any orientation Θ other than 0° and 90°:

$$G_\theta(x,y) = e^{\left(-\frac{(x\cos\theta+y\sin\theta)^2+\gamma^2(-x\sin\theta+y\cos\theta)^2}{2\sigma^2}\right)}\cos\left(\frac{2\pi}{\lambda}(x\cos\theta + y\sin\theta)\right) \quad (5)$$

In order to reduce the complexity of computation and make the Gabor filters separable, we approximate them using its isotropic (γ=1, circular) version. The effect of this approximation on the accuracy of the HMAX model is investigated in [17] and substituting γ=1 does not reduce accuracy. Therefore:

$$\begin{aligned} &G_\theta(x,y) \\ &= e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}\left[\cos\left(\frac{2\pi}{\lambda}x\cos\theta\right)\cos\left(\frac{2\pi}{\lambda}y\sin\theta\right)\right. \\ &\left.- sin\left(\frac{2\pi}{\lambda}x\cos\theta\right)\sin\left(\frac{2\pi}{\lambda}y\sin\theta\right)\right] \end{aligned} \quad (6)$$

$$\begin{aligned} &= e^{\left(-\frac{x^2}{2\sigma^2}\right)}cos\left(\frac{2\pi}{\lambda}x\cos\theta\right)e^{\left(-\frac{y^2}{2\sigma^2}\right)}cos\left(\frac{2\pi}{\lambda}y\sin\theta\right) \\ &- e^{\left(-\frac{x^2}{2\sigma^2}\right)}sin\left(\frac{2\pi}{\lambda}x\cos\theta\right)e^{\left(-\frac{y^2}{2\sigma^2}\right)}sin\left(\frac{2\pi}{\lambda}y\sin\theta\right) \\ &= E_3^T(x).E_4^T(y) - O_1^T(x).O_2^T(y) \end{aligned}$$

in which:

$$\begin{aligned} E_3(x) &= e^{\left(-\frac{x^2}{2\sigma^2}\right)}cos\left(\frac{2\pi}{\lambda}x\,cos\,\theta\right) \\ E_4(y) &= e^{\left(-\frac{y^2}{2\sigma^2}\right)}cos\left(\frac{2\pi}{\lambda}y\,sin\,\theta\right) \\ O_1(x) &= e^{\left(-\frac{x^2}{2\sigma^2}\right)}sin\left(\frac{2\pi}{\lambda}x\,cos\,\theta\right) \\ O_2(y) &= e^{\left(-\frac{y^2}{2\sigma^2}\right)}sin\left(\frac{2\pi}{\lambda}y\,sin\,\theta\right) \end{aligned} \quad (7)$$

This signifies that the 2D Gabor filter is equal to the minus of two separable filters in each arbitrary orientation.

Especially in 45° and 135° orientations $E_3 = E_4$ and $O_1=O_2$ due to $\sin\theta = \cos\theta$:

$$\begin{aligned} &G_{\theta=45}(x,y) \\ &= \left[e^{-\frac{x^2}{2\sigma^2}}\cos\left(\frac{\sqrt{2}\pi x}{\lambda}\right)\right].\left[e^{-\frac{y^2}{2\sigma^2}}\cos\left(\frac{\sqrt{2}\pi y}{\lambda}\right)\right] \\ &- \left[e^{-\frac{x^2}{2\sigma^2}}\sin\left(\frac{\sqrt{2}\pi x}{\lambda}\right)\right].\left[e^{-\frac{y^2}{2\sigma^2}}\sin\left(\frac{\sqrt{2}\pi y}{\lambda}\right)\right] \end{aligned} \quad (8)$$

$$\begin{aligned} &G_{\theta=135}(x,y) \\ &= \left[e^{-\frac{x^2}{2\sigma^2}}\cos\left(\frac{\sqrt{2}\pi x}{\lambda}\right)\right].\left[e^{-\frac{y^2}{2\sigma^2}}\cos\left(\frac{\sqrt{2}\pi y}{\lambda}\right)\right] \\ &+ \left[e^{-\frac{x^2}{2\sigma^2}}\sin\left(\frac{\sqrt{2}\pi x}{\lambda}\right)\right].\left[e^{-\frac{y^2}{2\sigma^2}}\sin\left(\frac{\sqrt{2}\pi y}{\lambda}\right)\right] \end{aligned} \quad (9)$$

Then:

$$G_{\theta=45}(x,y) = {E_3}^{T}(x) * E_3(y) - {O_1}^{T}(x) * O_1(y)$$

$$G_{\theta=135}(x,y) = {E_3}^{T}(x) * E_3(y) + {O_1}^{T}(x) * O_1(y) \quad (10)$$

in witch:

$$E_3(x) = e^{-\frac{x^2}{2\sigma^2}} \cos\left(\frac{\sqrt{2}\pi x}{\lambda}\right)$$

$$O_1(x) = e^{-\frac{x^2}{2\sigma^2}} \sin\left(\frac{\sqrt{2}\pi x}{\lambda}\right) \quad (11)$$

Therefore, in these cases, the 2D Gabor filter transforms to plus or minus of two separate 1D filter. In addition, each 1D filter can be halved by using the even or odd symmetry properties in the functions of (10). This will reduce the required number of multipliers by about 50%.

The use of separability and symmetry properties improves storage capacity of Gabor filter coefficients. For example, storing a 37×37 filter requires 1,699 memory locations. However, in the proposed design, 19+19=38 memory locations are utilized. For a total of 64 filters in S1 layer of HMAX, instead of $\sum_{k=7,9,\ldots 37} 4k^2 = 145{,}664$ memory locations, $\sum_{k=7,9,\ldots 37} 4 * 2 * (k+1)/2 = 5{,}888$ are utilized which k is the filters dimension. This represents a 95.8% reduction in the memory requirement for storing Gabor filter coefficients.

Hardware Architecture of the Accelerator

This section describes the hardware accelerator architecture of the HMAX model's S1 layer. This design is the result of an accurate study of the S1 layer structure and the arrangement of the steps involved in this layer. The design is part of the HMAX model and is ultimately utilized with this model.

A gray scale image as the accelerator input is stored in an internal memory. In addition, the Gabor filter coefficients are pre-calculated with the functions presented in the previous section and stored in the FPGA internal memory.

Figure 3 presents the block diagram of the entire S1 layer architecture. The design has two paths. In the first path, the input image data is applied to the 1D filter and the outputs are stored in an intermediate RAM. In the second path, instead of the input image, the stored values of the first path are applied to the 1D filter with a vertical sweep. Since the input image is required to apply other filters, the memory allocated for the input image cannot be used for intermediate values. Except for input image RAM and normalization values calculator (the normalization value is the same for all filters), this architecture has been replicated four times (for orientations 0°, 45°, 90° and 135°). The results of applying 0° and 90° Gabor filter are directly divided by the normalization value, but according to (10), the results of applying E_3 and O_1 filters must be + and – for Gabor filters 135° and 45°, respectively. The normalization value is calculated with the processing of the filter, simultaneously.

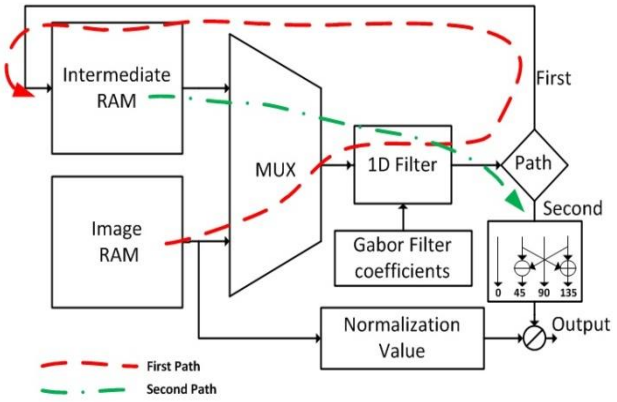


Fig. 4: The block diagram of the S1 layer architecture.

Figure 5 shows the architecture of a 1D filter. At each clock pulse, one pixel from a row of the input image is entered into a filter. Therefore, in the 37×37 filter, after 37 clock pulses, the data is available and the accelerator starts to work.

By exploiting the even or odd symmetry in the coefficients of the 1D filters, the pixel values of the input image are added or subtracted and then applied to the filter coefficients. Finally, the outputs of 19 multipliers, resulting of 37×37 filter, are added together using a pipelined version of the adder tree. Using multi-stage pipeline structure, the 1D filter output is prepared in one position of the input image in one clock pulse.

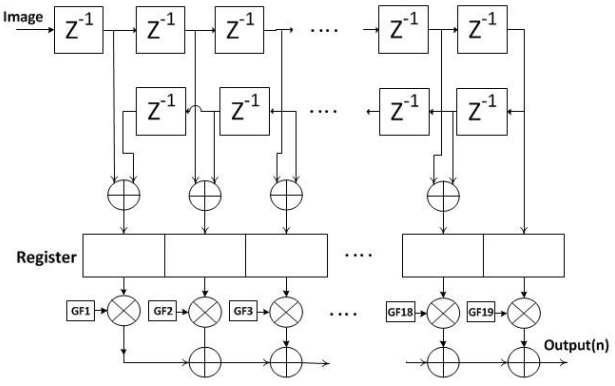


Fig. 5: Architecture of a 1D filter (with even symmetry). Z^{-1} represents a one-sample delay and GF is a Gabor Filter coefficient.

Figure 6 shows the architecture of the normalization values calculator. Initially, the image values are squared and entered into a shift register. A pipeline adder collects the register values and the results of the first path are stored in an intermediate RAM.

In the second path, the stored values are vertically swept and entered into the shift register. The square root of the results are used as normalization value as shown in Fig. 5.

The "Image RAM" shown in Fig. 6 and Fig. 4 is the same.

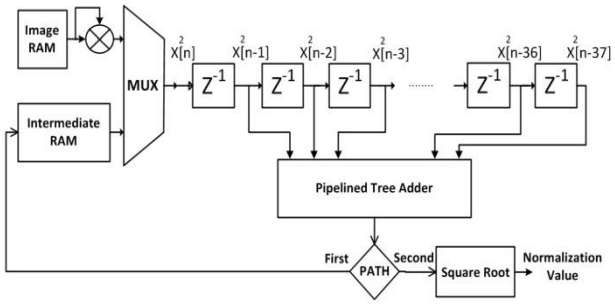


Fig. 6: Architecture of the normalization values calculator.

With the employment of the Xilinx IP cores available in ISE Design Suite software, the design of the circuit is facilitated and its square root, division, multiplication, and RAM components become more efficient. In order to more parallel processing in the hardware, four 1D filters with orientations of 0°, 45°, 90° and 135° have been instantiated to generate the results at four angles simultaneously.

Simulation

The present study investigates three different simulations on various images from well-known datasets, such as Caltech 101 [18]. The first one is the S1 layer from the standard HMAX model written by Serre et al. [2], whose source code is available online. This is a simulation in MATLAB and serves as a basis for comparing our extension code results.

The second simulation is the modified code in MATLAB, which converts a 2D filter to two 1D filter by use of the separability and symmetry properties of isotropic Gabor filters (γ=1). As expected, the result of this simulation is exactly equivalent to the first simulation or the standard model, due to the mathematical proof given in the third section.

With the usage of ModelSim software, the VHDL code of the designed architecture is simulated as the third simulation.

This simulation tests the functional operation and timing characteristics of the circuit (Fig. 7). The functional simulation is useful for checking the fundamental correctness of the designed circuit and the accuracy of the results due to the bit width limitations.

Hardware implementation of circuit with Xilinx ISE determines the timing and resource consumption. The timing simulation evaluates the speed performance by considering the latency of wires and logic components and tests the mathematical operation of the circuit. The target platform for simulation and synthesis is a mid-range commercial Xilinx FPGA, i.e. XC6VLX240T of the Virtex-6 family.

Implementation Results and Comparisons

By considering accuracy, efficiency, as well as resource and power consumption, this section evaluates the accelerator proposed. Implementation is performed on a Xilinx XC6VLX240T device, which hosts approximately 150,000 LUTs, a 14.5 Mb built-in RAM, and 768 DSP48. The design is written in VHDL language and synthesized and implemented with ISE 14.7 software.

The input image is assumed to be 128×128 pixels and converts to a gray scale with an integer number between 0 and 255 (8 bits) assigned to every pixel. The input image is stored in an internal memory. The output is expressed with four 16-bit integer numbers in conjunction with the 14-bit address of the image pixel and a signal for output validation (Fig. 7). The output is not stored in the internal memory due to its consumption of excessive storage space.

For each of the Gabor filter coefficients, a 16-bit fixed point number was assigned; 8 bits for integer part and 8 bits for fractional part.

Fixed point expression was selected since this expression uses fewer physical resources and fewer clock cycles than do floating point numbers. At the end of the first and second path, 8 bits from the low significant of the result numbers are removed. So the number stored in the intermediate RAM and the result of the second path will be integer. The bit width of intermediate values is adjusted such that it never overflows in the worst-case scenario.

Multiplication, division, RAM, and square root components are generated by Core Generator software. A state machine controls the entire circuit via multiple counters for counting the rows and columns of the input image.

Implementation of the total circuit indicates that the design can run at a minimum period of 5.38nsec or a maximum frequency of 185 MHz.

Since the processing of a single line of input images with the pipeline structure takes about N clock cycles, processing the first path of filter in four orientations requires about MN clocks, where M and N are the width and height of the input image, respectively.

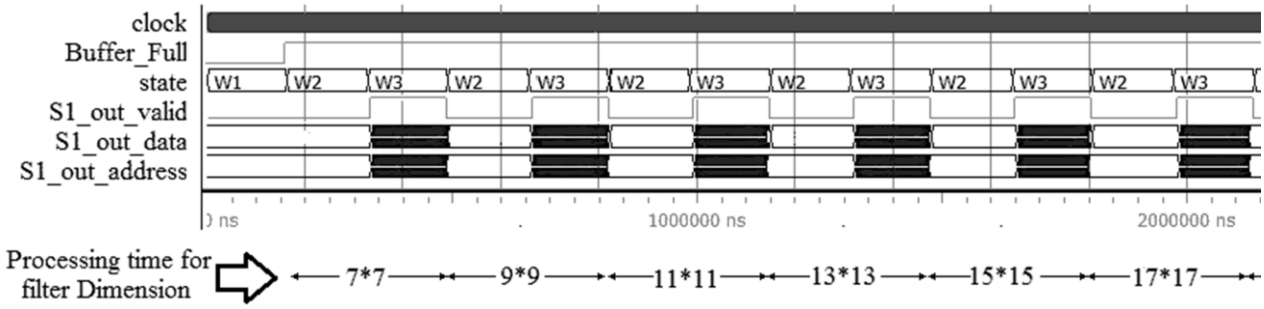


Fig. 7: Simulation of the S1 layer with ModelSim (states: W1: load image, W2: first path of filter, W3: second path of filter).

Therefore, process of first and second path of the filter and produce the final result take about 2MN clocks. Thus applying 16 filters take about:

$$Del = 16 \times 2MN \times 5.38 \text{ nsec} \quad (12)$$

Del is the total delay of our hardware for each input image with M×N pixels. This time is 2.82msec for 128×128 pixel image. Simulation of the circuit shows 2.84msec delays, resulting in a maximum throughput of 354 frames per second (fps). Consequently, the average processing time for one filter is 2.84ms/64=44µs. Obviously, this circuit is one layer of the HMAX model and its throughput should be comparable by the throughput of the rest of the circuit. In this calculation, the loading time of the input image is eliminated due to the pipeline structure of the circuit.

Table 1 presents the total hardware resources consumed by the implemented circuit on a XC6VLX240T device. The limiting factor in further parallel processing is the amount of memory available. Table 1 is for a 128×128 pixel image. The major impact of increasing the input image's size is on the RAM used as it is directly related to the image's dimensions. Therefore, if the image is enlarged to 256×256, then the memory consumption is quadrupled. Regardless of the amount of memory required, one of the advantages of the proposed architecture is that increasing the size of the input image has no effect on the number of multipliers required. Number of multipliers is affected by the largest filter dimension or 37. It should be noted that not utilizing the separability and symmetry property, we will require 37×37×4=5,476 multiplications.

The estimated dynamic power consumption is 1.02W and the quiescent power consumption is 2.97W due to leakage by use of Xilinx XPower Analyzer software.

For comparison of circuit speeds, Table 2 provides three different implementations of the HMAX model's S1 layer released in recent years. In addition to the standard HMAX model which is discussed in the present paper, there is an extension model developed by Mutch et al. [3]. In this model, Gabor filters with same sizes (11×11) are used for all scales, applying them to scaled versions of the image. The advantage of this model is in reducing computational complexity, so it has been used in many implementations in recent years.

Table 1: Hardware resource usage

Resource	Used	Available	Percent
DSP48E1	77	768	10%
RAMB36E1	57	416	13%
RAMB18E1	5	832	1%
Slice Reg.	15,598	301,440	5%
LUTs	9,619	150,720	6%

Reference [19] implements the whole Mutch HMAX model in the embedded Power PC, applies four 11×11 Gabor filter to 10 images pyramid in the S1 layer, and reports a 511ms delay. The hardware implementation of this study achieved a maximum delay of 260msec. In the hardware proposed in [14], implements the application of an 11×11 Gabor filter, in four directions, to a 12 images pyramid, from 256×256 to 38×38 image dimensions, and reports a hardware delay of 56.3 msec. In an article similar to the current work, the S1 layer of the standard HMAX model with a 128×128 image is implemented on programmable hardware, achieving a favorable Throughput of 190 fps. with a separable Gabor filter [13].

In a comparable work, two 7×7 and 9×9 Gabor filters are applied to an input image in four directions [20]. As reported, 75 and 143 fps could be processed in their FPGA and std_cell implementations for full-HD (1920×1080) images, respectively. Even though the image dimensions are about 126 times larger than those of the present study's implementation, only the two smallest filters, out of the 16, are used in this implementation. In the present work, there are 16 scales filter from 7×7 to 37×37 which include 36,416 multiply accumulate while in 7×7 and 9×9 filters 130 multiply accumulate are required.

Table 2: Speed comparisons among various implementations

	Image Size (pixel)	Filter Size	Throughput (fps)	Description
Debole [19]	140×140 ~ 30×30 (10*)	11×11	3.85	Mutch HMAX model, 11 orientation
Masshri [14]	256×256 ~ 38×38 (12*)	11×11	17.8	Mutch HMAX model
Orchard [13]	128×128	7×7 ~ 37×37 (16*)	190	standard HMAX
Licciardo [20]	1920×1080	7×7 and 9×9	75/143	FPGA/Std_cell implementation
This paper	128×128	7×7 ~ 37×37 (16*)	353	standard HMAX

* Number of images in the pyramid

Therefore, the complexity ratio in these two works in filters will be 36,416/130=280, which is much larger than image dimensions complexity ratio (126). By other calculation, according to (12), delay of our circuit for 1920×1080 pixel images is about 357msec. Since the difference in complexity of the filters used in the two implementations is 280 times, therefore, it can be concluded that if our circuit hardware resources are used for [20] implementation, the circuit delay will be 357/280=1.34msec which is 9.9X speedup.

Comparisons of [14], [19], [20] are unfair due to their differences in, for example, the model, filter size, input image size, and implementation platform. Despite this, Table 2 presents these implementations since they are the most recently designed for increasing the response speed.

The design that is most resembling to that of present article is [13], which is an architecture about 1.86X slower than the current work's on the identical hardware platform.

Conclusion

In this study, we proposed a new method for applying an isotropic 2D Gabor filter to an input image. With the conversion of one 2D filter to two 1D filters (the separability property), the computational complexity is reduced. Besides, to determine the efficiency and resource requirements of the proposed method, pipeline structure of the S1 layer of the HMAX model has been implemented, which consists of 64 2D Gabor filter in different scales and orientation as well as normalization value calculator circuit.

The architecture implemented on a fast and mid-range commercial FPGA platform, i.e. XC6VLX240T, for which the achieved throughput was 353 fps. Using this method saves FPGA resources, specially the number of DSP48E1 and RAM blocks, as well as reducing hardware delays.

The hardware design of the S1 layer of the standard HMAX model is performed with the assistance of Xilinx IP cores.

The proposed S1 layer architecture accelerates by more than 1.86X compared to the most similar work.

Using two one-dimensional filters and using a pipeline structure between the two filters can be introduced as a future work.

Author Contributions

A. Mohammadi Anbaran and N. Bagheri carried out the hardware design, simulation and implementation of the circuit and wrote the manuscript. P. Torkzadeh and R. Ebrahimpour carried out the data analysis and edit the manuscript. A. Mohammadi Anbaran collected the data and interpreted the results.

Acknowledgment

We thank the editor and all anonymous reviewers.

Conflict of Interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

Abbreviations

HMAX	Hierarchical model and X
FPGA	Field Programmable Gate Arrays
2D	2 Dimension
GF	Gabor Filter
CORDIC	COordinate Rotation DIgital Computer

References

[1] D.H. Hubel, T.N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," J. Physiol., 148(3): 574-591, 1959.

[2] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, T. Poggio, "Robust object recognition with cortex-like mechanisms," IEEE Trans. Pattern Anal. Mach. Intell., 29(3): 411-426, 2007.

[3] J. Mutch, D.G. Lowe, "Multiclass object recognition with sparse, localized features," in Proc. Computer Vision and Pattern Recognition conf. (CVPR'06): 11-18, 2006.

[4] S. Rajan, P. Chenniappan, S. Devaraj, N. Madian, "Facial expression recognition techniques: a comprehensive survey," IET Image Proc., 13(7): 1031-1040, 2019.

[5] R. Mehrotra, K.R. Namuduri, N. Ranganathan, "Gabor filter-based edge detection," Pattern Recognit., 25(12): 1479-1494, 1992.

[6] S. Liu, Y. Liu, X. Zhu, Z. Liu, G. Huo, T. Ding, et al., "Gabor filtering and adaptive optimization neural network for iris double recognition," in proc. Chinese Conf. on Biometric Recognition: 441-449, 2018.

[7] G. Amayeh, A. Tavakkoli, G. Bebis, "Accurate and efficient computation of gabor features in real-time applications," in proc. Int. Symposium on Visual Computing: 243-252, 2009.

[8] Y. Liang, L. Lu, Q. Xiao, S. Yan, "Evaluating fast algorithms for convolutional neural networks on FPGAs," IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 39(4): 857-870, 2020.

[9] A. Sironi, B. Tekin, R. Rigamonti, V. Lepetit, P. Fua, "Learning separable filters," IEEE Trans. Pattern Anal. Mach. Intell., 37(1): 94-106, 2014.

[10] J. Khosravi, M. Shams Esfandabadi, R. Ebrahimpour, "Image registration based on sum of square difference cost function," J. Electr. Comput. Eng. Innovations, 6(2): 263-271, 2018.

[11] V. Areekul, U. Watchareeruetai, S. Tantaratana, "Fast separable gabor filter for fingerprint enhancement," in Proc. Int. Conf. on Biometric Authentication: 403-409, 2004.

[12] W.-M. Pang, K.-S. Choi, J. Qin, "Fast Gabor texture feature extraction with separable filters using GPU," J. Real-Time Image Proc., 1(12): 5-13, 2016.

[13] G. Orchard, J.G. Martin, R.J. Vogelstein, R. Etienne-Cummings, "Fast neuromimetic object recognition using FPGA outperforms GPU implementations," IEEE Trans. Neural Networks Learn. Syst., 24(8): 1239-1252, 2013.

[14] A. Al Maashri, M. Cotter, N. Chandramoorthy, M. DeBole, C.-L. Yu, V. Narayanan, et al., "Hardware acceleration for neuromorphic vision algorithms," J. Signal Process. Syst., 70(2): 163-175, 2013.

[15] S. Moini, B. Alizadeh, M. Emad, R. Ebrahimpour, "A resource-limited hardware accelerator for convolutional neural networks in embedded vision applications," IEEE Trans. Circuits Syst. II Express Briefs, 64(10): 1217-1221, 2017.

[16] F. Abdi, P. Amiri, M.H. Refan, "Low computational complexity and high computational speed in leading DCD ERLS algorithm," J. Electr. Comput. Eng. Innovations, 7(1): 19-26, 2019.

[17] S. Chikkerur, T. Poggio, "Approximations in the hmax model," Computer Science and Artificial Intelligence Laboratory, Tech. Rep. MIT-CSAIL-TR-2011-021, 2011.

[18] L. Fei-Fei, R. Fergus, P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," Comput. Vision Image Understanding, 106(1): 59-70, 2007.

[19] M. DeBole, Y. Xiao, C.-L. Yu, A. Al Maashri, M. Cotter, C. Chakrabarti, et al., "FPGA-accelerator system for computing biologically inspired feature extraction models," in proc. 45th Asilomar Conference on Signals, Systems and Computers (ASILOMAR): 751-755, 2011.

[20] G.D. Licciardo, C. Cappetta, L. Di Benedetto, "Design of a Gabor filter HW accelerator for applications in medical imaging," IEEE Trans. Compon. Packag. Manuf. Technol., 8(7): 1187-1194, 2018.

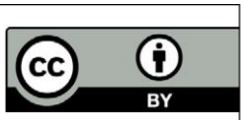
Biographies

Alireza Mohammadi Anbaran received a B.Sc. and a M.Sc. degree in electronic engineering from Shahrood University of Technology and Iran University of Science and Technology, in 2000 and 2003, respectively. He is currently pursuing the Ph.D. degree in the Faculty of Electrical and Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. His research interest includes digital circuit implementation, object recognition, digital signal processing and ASIP design.

Pooya Torkzadeh was born in Isfahan, in 1980. He received the B.Sc. degree from Isfahan University of Technology (IUT), Iran, in 2002 and the M.Sc. and Ph.D. degrees from the Sharif University of Technology (SUT), Iran, in 2004 and 2011 respectively both in electrical engineering. He is currently faculty member of Islamic Azad University, Science and Research Branch. Since 2003 he has joint to Sharif Integrated Circuit and System Group (SICAS) working on continuous/discrete time sigma-delta modulators with low power consumption for low power appliances. He has received many patents in the field of sigma-delta modulator designing and optimizing. He is the author and coauthor of more than 25 published international journal and conference papers on analog and digital integrated circuits. His research interests include ADC signal converters, low power phase locked loops (PLLs) with ultra-low phase noise amount for broad-band applications.

Reza Ebrahimpour is a professor at the department of computer engineering, Shahid Rajaee Teacher Training University, Tehran, Iran. He obtained Ph.D. degree in the field of Cognitive Neuroscience from the SCS, IPM, Tehran, Iran in July 2007. Dr. Ebrahimpour is the author or co-author of more than 100 international journal and conference publications in his research areas, which include Cognitive and Systems Neuroscience, Human and Machine Vision, Decision Making and Object Recognition.

Nasour Bagheri is an associate professor at the Electrical Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran. His research interests include cryptology, digital designing and hardware implementation. A record of his publication is available at Google Scholar: https://scholar.google.com/citations?user=D32llx44AAAAJ&hl=en.



How to cite this paper:
A. Mohammadi Anbaran, P. Torkzadeh, R. Ebrahimpour, N. Bagher, "Fast and efficient hardware implementation of 2D Gabor filter for a biologically-inspired visual processing algorithm," J. Electr. Comput. Eng. Innovations, 9(1): 93-102, 2021.

DOI: 10.22061/JECEI.2020.7548.404

URL: http://jecei.sru.ac.ir/article_1490.html