Synthesis and Optimization of 2D Filter Designs for Heterogeneous FPGAs

CHRISTOS-S. BOUGANIS, SUNG-BOEM PARK,
GEORGE A. CONSTANTINIDES, and PETER Y. K. CHEUNG
Imperial College London

Many image processing applications require fast convolution of an image with one or more 2D filters. Field-Programmable Gate Arrays (FPGAs) are often used to achieve this goal due to their fine grain parallelism and reconfigurability. However, the heterogeneous nature of modern reconfigurable devices is not usually considered during design optimization. This article proposes an algorithm that explores the space of possible implementation architectures of 2D filters, targeting the minimization of the required area, by optimizing the usage of the different components in a heterogeneous device. This is achieved by exploring the heterogeneous nature of modern reconfigurable devices using a Singular Value Decomposition based algorithm, which provides an efficient mapping of filter's implementation requirements to the heterogeneous components of modern FPGAs. In the case of multiple 2D filters, the proposed algorithm also exploits any redundancy that exists within each filter and between different filters in the set, leading to designs with minimized area. Experiments with real filter sets from computer vision applications demonstrate an average of up to 38% reduction in the required area.

Categories and Subject Descriptors: J.6 [**Computer-Aided Engineering**]: Computer-aided design (CAD); B.6.3 [**Logic Design**]: Design Aids; I.3.1 [**Computer Graphics**]: Hardware Architecture; I.4.3 [**Image Processing and Computer Vision**]: Enhancement

General Terms: Design Algorithms, Performance

Additional Key Words and Phrases: FPGA, 2D filter design, Singular Value Decomposition, reconfigurable logic

ACM Reference Format:
Bouganis, C.-S., Park, S.-B., Constantinides, G. A., and Cheung, P. Y. K. 2009. Synthesis and optimization of 2D filter designs for heterogeneous FPGAs. ACM Trans. Reconfig. Techn. Syst. 1, 4, Article 24 (January 2009), 28 pages. DOI = 10.1145/1462586.1462593.
http://doi.acm.org/10.1145/1462586.1462593.

This work was funded by the UK Research Council under the Basic Technology Research Programme "Reverse Engineering Human Visual Processes" GR/R87642/02 and by the EPSRC (EP/C549481/1) grant.
Authors' address: C.-S., Bouganis, S.-B., Park, G. A., Constantinides, and P. Y. K. Cheung, Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, United Kingdom; email: christos-savvas.bouganis@imperial.ac.uk.

1. INTRODUCTION

One of the fundamental operators in image processing is the two dimensional convolution. Examples can be found in edge detection and smoothing operations, where the image is convolved with a specific 2D filter, or kernel, in order to produce the desired result. Early filter sizes tended to be relatively small, for example, 3×3 pixels. In recent years, many image processing applications have appeared in the literature that require the use of much larger 2D filters. Moderate-size examples can be found in face detection/recognition applications where kernels with size of 23×23 pixels are used [Belhumeur et al. 1997], and some more extreme examples can be found in medical imaging where applications require kernels with size of up to 63×63 pixels. Moreover, image processing algorithms usually require the convolution of the input image with a set of 2D filters rather than with a single filter, which increases the computational load [Bouganis et al. 2004]. At the same time, real-time implementation is often required, making the use of hardware acceleration a necessity [Bouganis et al. 2004].

FPGAs are often used to achieve this goal due to their fine-grain parallelism and reconfigurability. Modern FPGAs are heterogeneous devices, often targeting the DSP community, and thus providing a mixture of resources that can be used in DSP applications. The two main coarse-grain cores that are usually included in the recent devices are embedded RAMs and embedded multipliers (Xilinx[1], Altera[2]). The former provides fast localized memory access, while the latter provides high-speed accurate multiplication.

In line with this direction, the proposed algorithm departs from the current methods of 2D filter implementation by providing an approach that makes explicit use of the heterogeneity of the device, targeting the reduction of design area. Furthermore, it provides a framework which allows the designers to move their designs to different points in the two dimensional design space of embedded multipliers and 4-input look-up tables (4-LUTs), keeping the variance of the error at the output of the system below a specified level. In our previous work [Bouganis et al. 2005a; 2005b] we addressed the problem of allocating in an efficient way the different resources of a modern reconfigurable device for implementation in an FPGA of a 2D convolution with a single filter. In Bouganis et al. [2005], we proposed an algorithm that extends our work to target the implementation of a set of 2D filters, by providing an approach that exploits any redundancy that exists within each filter and between different filters in the set.

This article presents in more depth the main ideas of the proposed algorithms [Bouganis et al. 2005a; 2005b; 2005], provides a more detailed cost model of the main components of the design, and presents a more efficient strategy in terms of computational complexity for searching the design space than in Bouganis et al. [2005a, 2005b, 2005]. Moreover, results are presented

[1]http://www.xilinx.com
[2]http://www.altera.com

for fully placed and routed FPGA designs regarding their area, speed, and power consumption. In summary, the novel contributions of this article are:

—The use of Singular Value Decomposition to approximate a 2D filter with a number of separable 2D filters and one low-complexity nonseparable 2D filter. This provides an efficient mapping from the 2D filter design requirements to the heterogeneous parts of modern reconfigurable devices. Furthermore, it reduces the number of high precision multipliers required for implementation.

—An extension of the Singular Value Decomposition to approximate a set of 2D filters with a number of separable 2D filters and a set of low-complexity nonseparable 2D filters. Moreover, the proposed technique exploits any existing redundancy within each filter and between different filters of the input set, leading to designs with minimized resource requirements.

—A resource allocation algorithm that maps the decomposed filter design onto a given set of heterogeneous resources on an FPGA, including embedded multipliers and LUTs, in order to minimize resource usage. The proposed method shows a significant reduction in resource usage, targeting various user requirements.

The article is organized as follows. Section 2 describes the current related work regarding implementation of 2D filter designs in hardware, and more specifically in an FPGA. Section 3 describes and formulates the objective of the paper. High level and detailed descriptions of the proposed algorithm are given in Section 4. Section 5 focuses on the evaluation of the algorithm, and Section 6 concludes the article.

2. RELATED WORK

The article focuses on the typical DSP case where designs with high throughput are required, but design latency is of secondary importance. For this reason, only pipelined techniques for the implementation of a 2D convolution filter are considered.

There are a large number of publications regarding digital filter implementations taking into account the effects of coefficient quantization on the filter's response. Siohan [1990] presents statistical bounds for the degradation of the frequency response due to coefficient quantization in the case of direct-form realization of 2D FIR filters with quadrantal or octagonal symmetry. Haseyama and Matsuura [2006] consider the problem of achieving the most accurate approximation of the filter's frequency response when the coefficients are represented using floating-point precision. A fixed number of bits are allocated to the mantissa and exponent of the coefficients. The proposed algorithm uses a genetic algorithm and simulating annealing to calculate the coefficients and it was reported that the proposed method produces better approximation of the filter's response than simulating annealing-based only methods.

Moreover, some work has been concentrated on the implementation of multiplierless FIR digital filters where the coefficients can be expressed as sums of signed power-of-two (SPT) terms. It should be noted that a mixed-integer

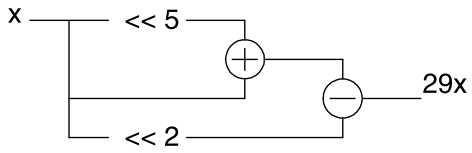


Fig. 1. Detailed diagram of a constant coefficient multiplier using canonic signed digit recoding. Coefficient 29 is recoding to [1 0 0 -1 0 1] using CSD, which leads to a reduction in the required adders.

linear programming formulation (MILP) can lead to the optimum solution, but its computational requirements prohibit this method for high-order filters [Chen et al. 1995]. Polynomial-time algorithms for designing digital filters with power-of-two coefficients can be found in [Chen et al. 1995; Li et al. 2002]. However, these works consider the problem of digital filter implementation in general and not the problem of mapping a digital filter on an FPGA device taking into account the device's specific structure.

Current design methods for implementing a 2D filter in an FPGA have two main stages. In the first stage, the filter's coefficients are approximated using finite word-length precision through optimization of an objective function. Depending on the application, the objective function can be the mean square error or the maximum absolute error, and can be applied in the spatial or frequency domain [Kodek 1980]. In the second stage, the constant coefficient multiplications are often implemented as shift/add combinations using 4-LUTs and, to further optimize the design, the coefficients are sometimes transformed using *canonic signed digit* recoding, which reduces the required logic [Koren 2002].

Canonic signed digit recoding represents the coefficients in a way such that high-speed low-area multiplication can be achieved. It is a radix-2 signed digit system using the set $\{1, 0, -1\}$. Given a number, its canonic signed digit representation has two important properties: (a) the number of nonzero digits is minimum, and (b) no two nonzero digits are adjacent. Due to the first property, canonic signed representation is widely used for implementing constant coefficient multipliers [Park and Kang 2001; Samueli 1989]. Figure 1 illustrates an example of CSD, in the case where the coefficient has the value 29. The usual implementation needs three adders, where under CSD recoding only two are required.

Techniques to further reduce the required resources when multiplying by several coefficients can also be found in the literature [Dempster and Macleod 1995; Pasko et al. 1999; Yurdakul 2005]. This is achieved by using common partial results, which effectively reduces the required area for the multiplications. Figure 2 illustrates an example of using common partial results for multiplication with coefficients having values 20 and 21.

Another technique [Andrews 1999] exploits the potential separability of a 2D filter into sets of two 1D filters by using the Singular Value Decomposition (SVD) [Press et al. 1992] to express the original filter as a linear combination of separable filters. This technique can be lossless, where the original filter is decomposed in a linear combination of separable filters without introducing

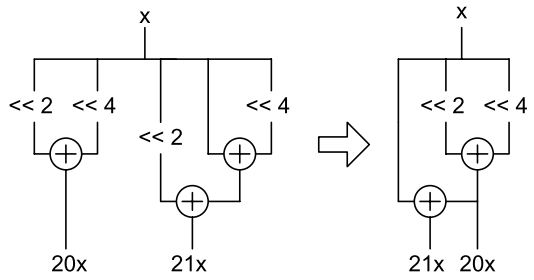


Fig. 2. Common subexpression elimination. The two products 20x and 21x can share a common subexpression which leads to a reduction in the required adders.

any error, or it can be lossy, where a certain error in the original filter approximation is allowed. Using this technique, the initial filter can be implemented as a set of 1D filters where half of them are applied to the rows of the input image, and the other half to the columns of the resulting image from the first convolution. By decomposing the 2D filter, the number of necessary multiplications may be reduced. For a 2D filter with size $m \times n$, the number of multiplications in the original form is mn. By applying the Singular Value Decomposition algorithm, each stage of the decomposition requires $m + n$ multiplications. However, the number of levels of decomposition that are required depends on the separability properties of the filter and the arithmetic accuracy for storing intermediate results. Moreover, the decomposition of the filter into separable filters is considered to be, in this work, independent to the quantization process of the coefficients.

These techniques can be combined with algorithms that minimize the area cost of a filter by representing the result of a multiplication/addition using an appropriate number of bits such that the final error at the output of the filter is bounded by a user defined value such as in Constantinides et al. [2003]. Current algorithms can be classified in three categories: those that use an analytic approach to scaling and error estimation [Constantinides et al. 2003], those that use simulation [Kum and Sung 2001], and finally those that use both techniques [Cmar et al. 1999].

In this article, we propose a novel algorithm to optimize a 2D convolution filter implementation in a heterogeneous device, given a set of constraints regarding the number of embedded multipliers and 4-LUTs. The algorithm is based on a lossy synthesis framework, where an approximation of the original 2D filter is targeted which minimizes the error at the output of the system and at the same time meets the user's constraints on resource usage. Moreover, we extend our framework to applications that require convolutions with a set of 2D filters rather than with a single 2D filter, by exploiting any redundancy that exists within each filter and between different filters in the set. The proposed method alters the structure of the original filter(s) in order to find a structure that can be mapped in a more efficient way to the targeted device. The exploration of the design space is performed at a higher level than the word-length optimization methods or methods that use common subexpressions to reduce

the area, since they do not consider altering the computational structure of the filter. The proposed technique is thus complementary to these previous approaches.

3. OBJECTIVE

Under the lossy synthesis framework, we extend the current two-dimensional design space of area-speed to include a third dimension which is the accuracy of the final result [Constantinides et al. 2001]. By allowing the user to specify the acceptable deviation from the exact result, we are able to explore the design space more effectively and provide designs that directly target the user's requirements. In the present work, we are interested in finding an algorithm that produces a hardware-friendly design for a 2D convolution realization, achieving a minimum error at its output given a bound on the available area. In the specific problem, the metric that is used to describe the accuracy of the result is the variance of the noise at the output of the system. This describes the uncertainty of the result at the output and has been adopted by many researchers from the Digital Signal Processing field [Constantinides et al. 2003]. The rest of the section focuses on the 1D case for simplicity; however, the same results are valid in the 2D case too.

From Mitra [2002] the variance of a signal at the output of a linear time invariant (LTI) system, and in our specific case of a 2D convolution, when the input signal is a white random process is given by (1), where σ_y^2 is the variance of the signal at the output of the system, σ_x^2 is the variance of the signal at the input, and $h[n]$ is the impulse response of the system.

$$\sigma_y^2 = \sigma_x^2 \sum_{n=-\infty}^{\infty} |h[n]|^2 \ . \tag{1}$$

Under the proposed framework, the impulse response of the new system $\hat{h}[n]$ can be expressed as the sum of the impulse response of the original system $h[n]$ and an error impulse response $e[n]$ as in (2).

$$\hat{h}[n] = h[n] + e[n] \ . \tag{2}$$

The new system can be decomposed into two parts, as shown in Figure 3. The first part has the original impulse response $h[n]$, where the second part has the error impulse response $e[n]$. Thus, the variance of the noise at the output of the system due to the approximation of the original impulse response is given by (3), where *SSE* denotes the sum of square errors in the filter's impulse response approximation.

$$\sigma_{\text{noise}}^2 = \sigma_x^2 \sum_{n=-\infty}^{\infty} |e[n]|^2 = \sigma_x^2 \cdot SSE \ . \tag{3}$$

In the case where the system exhibits N outputs, the objective is a function $f(\cdot)$ of the output noise variances (4), where $\sigma_{\text{noise},i}^2$ denotes the variance of the

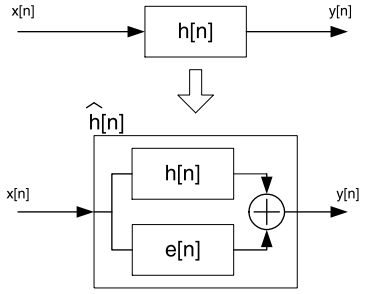


Fig. 3. The top graph shows the original system, where the second graph shows the approximated system and its decomposition to the original impulse response and to the error impulse response.

i^{th} output. In this work we have used the average function, but other functions like the max(·) function can be accommodated by the system.

$$\sigma^2_{system} = f(\sigma^2_{\text{noise},1}, \sigma^2_{\text{noise},2}, \ldots, \sigma^2_{\text{noise},N}) \,. \tag{4}$$

From (3) and (4) it can be concluded that the uncertainty at the output of the system is proportional to the sum of square error of the impulse response approximation, which can be used as a measure to assess the system's accuracy.

4. ALGORITHM DESCRIPTION

4.1 Single 2D Filter

The main idea behind the proposed method is to decompose a given 2D filter into a set of separable 2D filters, and to one nonseparable 2D filter which encodes the trailing error of the decomposition. The reason for that is made clear in the following.

A 2D filter is called separable if its impulse response $h[n_1, n_2]$ is a separable sequence, that is,

$$h[n_1, n_2] = h_1[n_1]h_2[n_2].$$

The important property is that a convolution with a separable filter can be decomposed into two one-dimensional convolutions as shown:

$$\begin{aligned} y[n_1, n_2] &= \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} x[n_1 - i, n_2 - j]h_1[i]h_2[j] \\ &= \sum_{i=-\infty}^{\infty} h_1[i] \sum_{j=-\infty}^{\infty} x[n_1 - i, n_2 - j]h_2[j] \,, \end{aligned}$$

where x and y denote the input and output images respectively.

The separable filters can potentially reduce the number of required multiplications from $m \times n$ to $m+n$ for each filter with size $m \times n$ pixels. The nonseparable parts encode the trailing error of the approximation and still require $m \times n$ multiplications. However, the coefficients are intended to need fewer bits for representation and therefore their multiplications are of low complexity. The decomposition that we are seeking should provide a ranking for the coefficients according to their impact in the overall filter approximation error. This ordering would enable the algorithm to assign the available resources in an optimum way. Moreover, the decomposition should remove any redundancy that exists within the 2D filter.

In more detail, given a 2D filter $\mathbf{F}$, the algorithm seeks a decomposition of the form given in (5), where $\hat{\mathbf{A}}$ is a linear combination of the separable filters $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_r$, and $\mathbf{E}$ is a non-separable 2D filter.

$$\mathbf{F} = \hat{\mathbf{A}} + \mathbf{E} \, . \tag{5}$$

Since the $\mathbf{A}_j$ matrices are separable, they can be decomposed into a product of vectors as $\mathbf{A}_j = \mathbf{u}_j \mathbf{v}_j^T$, where $\mathbf{u}_j$ and $\mathbf{v}_j$ are vectors. Thus, the approximation of a filter with r separable levels of decomposition is as in (6).

$$\mathbf{F} = \sum_{j=1}^{r} \lambda_j \mathbf{A}_j + \mathbf{E} = \sum_{j=1}^{r} \lambda_j \mathbf{u}_j \mathbf{v}_j^T + \mathbf{E} \, . \tag{6}$$

The nonseparable term $\mathbf{E}$ can be considered as the error term of the approximation of the filter from the separable set. This term is essential for achieving an adequate approximation of the filter using a relative small number of separable filters.

Given an input image $\mathbf{X}$ and a 2D filter $\mathbf{F}$, the resulting image of the convolution is given by $\mathbf{Y} = \mathbf{X} \circledast \mathbf{F}$, where $\circledast$ denotes convolution. Using the filter decomposition in (6), the resulting image is given by:

$$\mathbf{Y} = \mathbf{X} \circledast \left(\sum_{j=1}^{r} \lambda_j \mathbf{A}_j + \mathbf{E} \right) = \sum_{j=1}^{r} \lambda_j (\mathbf{X} \circledast \mathbf{u}_j) \circledast \mathbf{v}_j^T + \mathbf{X} \circledast \mathbf{E} \, . \tag{7}$$

A top-level diagram of such design is illustrated in Figure 4. The figure shows a design containing two decomposition levels, $r = 2$. A detailed view of one of the separable stages is illustrated in Figure 5.

For a given 2D filter there is an infinite number of possible decompositions described by (6). We are seeking the decomposition that has the minimum number of separable parts r for a given mean square error approximation, which corresponds to minimizing the number of required multipliers. Furthermore, the initial levels of the decomposition should have more impact on the mean square error approximation of the filters than the later ones, a property that provides a ranking for the coefficients and is exploited by the proposed algorithm.

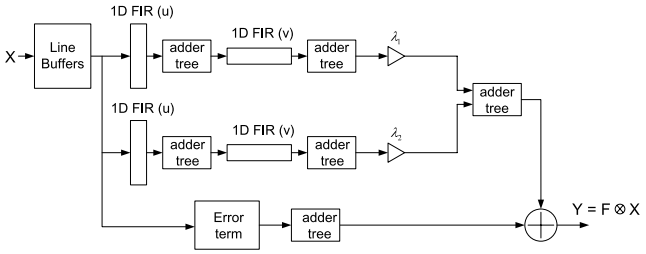


Fig. 4. Top-level diagram of the proposed decomposition for the case of two separable stages. The diagram illustrates the major components of the decomposition and their connections. A raster-scan format is assumed for the input image.

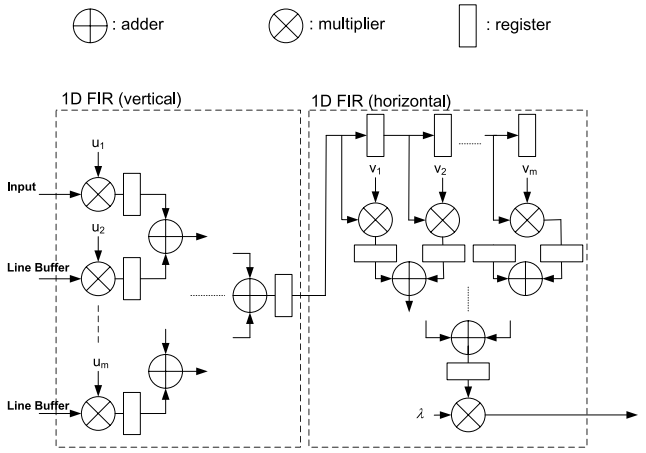


Fig. 5. Detailed diagram of a decomposition stage.

The decomposition that we are seeking is given by the Singular Value Decomposition algorithm, which decomposes a matrix into a linear combination of the fewest possible separable matrices [Press et al. 1992]. By applying the Singular Value Decomposition algorithm, the initial filter $\mathbf{F}$ is decomposed into a set of separable filters $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_L$:

$$\begin{aligned} \mathbf{F} &= \mathbf{U}\Lambda\mathbf{V}^T \\ &= \sum_{i=1}^{L} \lambda_i \mathbf{u}_i {\mathbf{v}_i}^T \\ &= \sum_{i=1}^{L} \mathbf{A}_i \ , \end{aligned} \tag{8}$$

where $\mathbf{U}$ and $\mathbf{V}^T$ are orthogonal matrices and Λ is a diagonal matrix containing the eigenvalues λ_i. The eigenvalues are sorted in descending order, thus $\lambda_1 \geq$

$\lambda_2 \geq ... \geq \lambda_L$. $\mathbf{u}_i$ and $\mathbf{v}_i$ correspond to the i^{th} columns of the $\mathbf{U}$ and $\mathbf{V}$ matrices respectively.

Without taking into account the quantization effects of the coefficients into the filter approximation, the mean square error that is achieved by including the first D decomposition levels is given by:

$$\begin{aligned} Err_D &= \frac{1}{m^2}||\mathbf{F} - \hat{\mathbf{F}}||^2 \\ &= \frac{1}{m^2}\left|\left|\sum_{i=D+1}^{L} \lambda_i \mathbf{u}_i {\mathbf{v}_i}^T\right|\right|^2 \\ &= \frac{1}{m^2} \\ &\quad \mathrm{tr}\left\{\left(\sum_{i=D+1}^{L} \lambda_i \mathbf{u}_i {\mathbf{v}_i}^T\right)\left(\sum_{i=D+1}^{L} \lambda_i \mathbf{u}_i {\mathbf{v}_i}^T\right)^T\right\} \\ &= \frac{1}{m^2}\sum_{i=D+1}^{L} {\lambda_i}^2 , \end{aligned} \tag{9}$$

where $||\cdot||$ denotes the Frobenius norm, tr$\{\cdot\}$ denotes the trace of a matrix and λ_i denotes the eigenvalue that corresponds to the i^{th} decomposition level. This provides a lower bound for the mean square error in the approximation when only the first D levels of the decomposition are considered without taking into account the quantization error of the coefficients.

4.2 Multiple 2D Filters

The algorithm can be extended to the case where a set of 2D filters need to be convolved with an input image. In this case, the main idea is to decompose a given set of 2D filters into one set of filters, which are common for all the given filters, and into a second set of filters which are specific for each filter. The common set of filters should capture the common properties of the given set, while the second set should capture their individual properties. Similar to the single filter case, out of all possible decompositions we are interested in the case where the common set of filters consists of separable filters.

Given a set of 2D filters $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_k$, the algorithm seeks a decomposition of the form given in (10), where $\hat{\mathbf{A}}_i$ is a linear combination of the separable filters $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_r$, and $\mathbf{E}_i$ is a nonseparable 2D filter.

$$\mathbf{F}_i = \hat{\mathbf{A}}_i + \mathbf{E}_i \ . \tag{10}$$

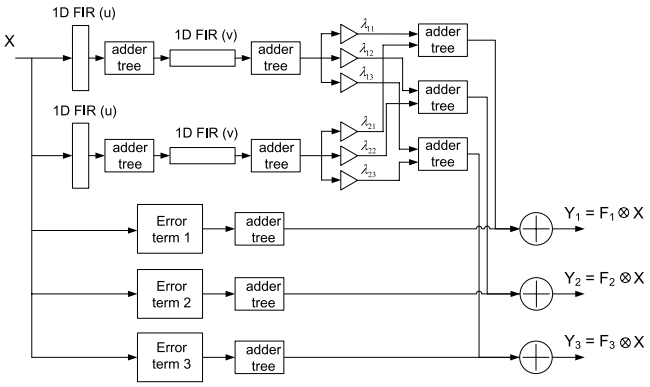


Fig. 6. Top-level diagram of the proposed decomposition for the case of three filters using two separable stages. The diagram illustrates the major components of the decomposition and their connections. The necessary line buffers are omitted for reasons of clarity.

Similar to (6), the approximation of each filter with r separable levels of decomposition is given in (11), where the resulting images of the convolutions with an input image $\mathbf{X}$ are given by (12).

$$\mathbf{F}_i = \sum_{j=1}^{r} \lambda_{ij}\mathbf{A}_j + \mathbf{E}_i = \sum_{j=1}^{r} \lambda_{ij}\mathbf{u}_j\mathbf{v}_j^T + \mathbf{E}_i \tag{11}$$

$$\mathbf{Y}_i = \mathbf{X} \circledast \left(\sum_{j=1}^{r} \lambda_{ij}\mathbf{A}_j + \mathbf{E}_i \right) = \sum_{j=1}^{r} \lambda_{ij}(\mathbf{X} \circledast \mathbf{u}_j) \circledast \mathbf{v}_j^T + \mathbf{X} \circledast \mathbf{E}_i \ . \tag{12}$$

A top-level diagram of such design is illustrated in Figure 6. The figure shows a design regarding three filters, $k = 3$, and two decomposition levels, $r = 2$. The necessary line buffers are omitted for reasons of clarity. It should be noted that the introduction of a common set of filters does not affect the throughput of the design. The output of each filter in the original set is produced by applying a filter specific linear transformation to the outputs of the common set of filters and adding to that the output of a filter which is specific for each filter in the original set ($\mathbf{E}_i$).

Similar to the single filter case, there is an infinite number of possible decompositions described by (10). In our case, we seek an extension of the SVD algorithm to many filters.

4.3 Rank-1 Decomposition Algorithm

Under our framework, the decomposition problem of multiple 2D filters can be formulated as follows. Given a set of $m \times n$ matrices $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_k$,

Algorithm: Decompose a set of $m \times n$ 2D matrices $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_k$ into rank-1 matrices $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_r$
FOR $j = 1 : r$
  Calculate eigenvector $\mathbf{u}$ that corresponds to the largest eigenvalue of the $n \times n$ matrix $\sum_{i=1}^{k} \mathbf{F}_i \mathbf{F}_i^T$
  Form $m \times k$ matrix $\hat{\mathbf{F}} = [\mathbf{F}_1^T \mathbf{u}, \mathbf{F}_2^T \mathbf{u}, \ldots, \mathbf{F}_k^T \mathbf{u}]$
  Calculate the eigenvector $\mathbf{v}$ that corresponds to the largest eigenvalue of the $m \times m$ matrix $\hat{\mathbf{F}}\hat{\mathbf{F}}^T$.
  Repeat
    Form the $k \times n$ matrix $\hat{\mathbf{F}} = [\mathbf{F}_1 \mathbf{v}, \mathbf{F}_2 \mathbf{v}, \ldots, \mathbf{F}_k \mathbf{v}]^T$
    Calculate the eigenvector $\mathbf{u}$ that corresponds to the largest eigenvalue of the $n \times n$ matrix $\hat{\mathbf{F}}^T \hat{\mathbf{F}}$.
    Form the $m \times k$ matrix $\hat{\mathbf{F}} = [\mathbf{F}_1^T \mathbf{u}, \mathbf{F}_2^T \mathbf{u}, \ldots, \mathbf{F}_k^T \mathbf{u}]^T$
    Calculate the eigenvector $\mathbf{v}$ that corresponds to the largest eigenvalue of the $m \times m$ matrix $\hat{\mathbf{F}}\hat{\mathbf{F}}^T$.
  until $\mathbf{u}$ and $\mathbf{v}$ vectors change less than a pre-specified value that is set by the user.
  $\mathbf{A}_j = \mathbf{u}\mathbf{v}^T$.
  $\lambda_{ij} = \mathbf{v}^T \mathbf{F}_i^T \mathbf{u}$.
  Replace $\mathbf{F}_i$ with $\mathbf{F}_i - \lambda_{ij} \mathbf{u}\mathbf{v}^T$.
END

Fig. 7. Outline of the rank-1 decomposition algorithm.

find the minimum set of rank-1 matrices $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_r$, such that linear combinations of them can span the matrices $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_k$ *i.e.*

$$\mathbf{F}_i = \sum_{j=1}^{r} \lambda_{ij} \mathbf{A}_j. \tag{13}$$

A matrix $\mathbf{A}$ is a rank-1 matrix, separable, if it can be decomposed as the product of two vectors, that is, $\mathbf{A} = \mathbf{u}\mathbf{v}^T$. The definition of the problem is the extension of a Singular Value Decomposition for one matrix to a collection of matrices. The special case where $k = 2$ is well researched [Ja'Ja 1978]. The general case where $k > 2$ has been shown to be an NP-complete problem [Hastad 1990]. The proposed framework is based on the decomposition proposed in Shashua and Levin [2001]. It is an iterative algorithm which is based on the Singular Value Decomposition algorithm. It is designed such that for the case of a single matrix ($k = 1$) the same decomposition as the Singular Value Decomposition algorithm is produced. This provides a unified framework for the case of single 2D filter and the case of multiple 2D filters.

The rank-1 decomposition algorithm is given in Figure 7. It should be noted that the rank-1 decomposition algorithm is a greedy algorithm since previous selections of the rank-1 elements are not reevaluated. Thus, the process is not guaranteed to converge to a global minimum. However, the convergence of the algorithm to a local minimum is guaranteed. Following Shashua and Levin [2001], let R_i^r be the residual of the i^{th} matrix after approximating the input set using r rank-1 matrices. It can be proven that (14) holds, where $M = \min(m, k)$, $N = \min(mk, n)$, and $||.||_F$ denotes the Frobenius norm. k denotes the number of

filters F_i in the input set, and m and n denote the number of rows and columns of the filters respectively. It should be noted that this provides only an upper bound on the convergence rate of the algorithm. Experiments have shown that the convergence rate is much faster. For more details and proof of convergence the reader is referred to Shashua and Levin [2001].

$$\sum_{i=1}^{k} ||R_i^r||_F^2 \leq \left(1 - \frac{1}{NM}\right)^r \sum_{i=1}^{k} ||F_i||_F^2 \,. \tag{14}$$

4.4 Detailed Description of the Algorithm

This section gives a detailed description of the proposed algorithm under the unified framework. Given a set of $m \times n$ filters $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_k$, and a set of constraints regarding the available number of embedded multipliers and slices, the algorithm finds a design that minimizes the variance at the output of the system and meets the constraints on resource requirements. The algorithm can be divided into two main stages: the *slice allocation stage* and the *multiplier allocation stage*.

4.4.1 *Slice Allocation Stage.* In the slice allocation stage, the algorithm decomposes the input filters using the rank-1 decomposition algorithm and implements all the constant coefficient multiplications using only slices. In the case of infinite precision, the matrices in (11) can be defined by one call of the rank-1 decomposition algorithm. However, due to the coefficient quantization in a hardware implementation, quantization error is inserted at each level of the decomposition. The algorithm reduces the effect of the quantization error by propagating the error of each decomposition level to the next one.

The algorithm first determines the separable filter $\mathbf{A}_1 = \mathbf{u}_1\mathbf{v}_1^T$ for the first level of the decomposition and the corresponding λ_{i1} for $i = 1, \ldots, k$ using the rank-1 decomposition algorithm. The algorithm quantizes the coefficients that correspond to the vectors $\mathbf{u}_1$ and $\mathbf{v}_1$ by using canonical signed digit recoding and using one nonzero bit per coefficient. Because the coefficients for this level are quantized, the λ_{i1} terms are reevaluated to correct for the quantization effects. The new λ_{i1} are calculated using the following system of linear equations [Strang 1998]:

$$\mathbf{F}_i\mathbf{v}_1 = \lambda_{i1}\mathbf{u}_1 \tag{15}$$

$$\mathbf{F}_i{}^T\mathbf{u}_1 = \lambda_{i1}\mathbf{v}_1 \tag{16}$$

where $\mathbf{u}_1$ and $\mathbf{v}_1$ have been quantized. The new λ_{i1} are quantized using the same quantization method as before.

The algorithm updates the input filters $\mathbf{F}_i$ as $\mathbf{F}_i \leftarrow \mathbf{F}_i - \lambda_{i1}\mathbf{u}_1\mathbf{v}_1^T$, and repeats the above process until it estimates r levels of decomposition. The final error terms $\mathbf{E}_i$ are formed by the resulting filters $\mathbf{F}_i$ where all the coefficients are quantized as before.

The Precision Equalizing algorithm has been developed for allocating the available slices to the coefficients in an optimum way. The main idea behind the algorithm is to equalize the precision of the coefficients that belong to the same 1D FIR filter. The motivation behind this algorithm is that the error that

Algorithm: Precision Equalizing Algorithm
FOR $k = 1$ to m (where m is the number of 1D FIR filters
    Find the current precision of each coefficient
    Recalculate the quantization error of the coefficient
    Recalculate the eigenvalue of the corresponding level and reset its HW
    Recalculate the coefficients of lower decomposition levels using the rank-1 algorithm and reset their HW to the default values
    Calculate the filter error of the new filter configuration and store it
END
Find the configuration with the smallest variance.

Fig. 8. Summary of the Precision Equalizing Algorithm.

is injected into the system by the quantization of each coefficient in the same 1D filter is propagated to the output of the system through the same transfer function. Given that the variance of the noise at the output of the system due the quantization of each coefficient is proportional to the variance of the signal at the input of the coefficient multiplier, which is the same for the coefficients that belong to the same 1D filter, a sensible choice would be for the coefficients of the same 1D filter to have the same precision. It should be noted that in the Precision Equalizing Algorithm only one coefficient for each 1D FIR filter is considered for optimization at each iteration, leading to solutions that are computational efficient. Figure 8 gives an outline of the Precision Equalizing Algorithm.

Regarding the complexity of the algorithm, for the case of k $m \times n$ filters and for one iteration, the Precision Equalizing Algorithm has a complexity proportional to $3r + mnk$, where r is the number of decomposition levels under investigation.

The first stage terminates when all the available slices have been used by the algorithm.

4.4.2 *Embedded Multiplier Allocation Stage.* In this stage, the algorithm determines the coefficients that will be placed to embedded multipliers. The coefficients that have the largest cost in terms of slices in the current design and reduce the filters' approximation error when they are allocated to embedded multipliers, are selected. The second condition is necessary due to the limited precision of the embedded multipliers (e.g., 18 bits in Xilinx Virtex II devices), which in some cases may restrict the approximation of the multiplication and consequently of the final result. Because of the above condition, in the case where the user requires a design with high accuracy, not all the available embedded multipliers specified by the user are always allocated.

Since the proposed algorithm is based on the Singular Value Decomposition algorithm, there exists an ordering of the coefficients regarding the impact of their approximation to the total approximation error. That is, coefficients from the initial decomposition levels have larger impact than coefficients of the

later levels. This feature is explored by the algorithm to efficiently allocate the embedded multipliers to coefficients that belong to the initial decomposition levels and at the same time their placement on an embedded multiplier would save a considerable amount of resources.

Finally, the algorithm allocates the slices that have been freed, due to the previous allocation of the embedded multipliers, to the rest of the design in order to achieve an even lower approximation error. This stage is similar to the slice allocation stage except that the coefficients that have been allocated to embedded multipliers are not considered for optimization.

It should be noted that the above decomposition is realized for a specific number of decomposition stages. In order to explore the design space for different decomposition levels, the proposed algorithm exhaustively searches a predefined set of separable levels, and returns the designs that achieve the lowest variance at the output of the system, for a specific area constraint.

4.5 Cost Model

The aim of the resource allocation algorithm is to find a filter decomposition that achieves the minimum error variance at the outputs of the system given constraints on the available resources. For this reason an accurate and relatively fast model for the resource usage is required.

A cost model has been constructed for a Xilinx Virtex-II device[3] to predict the cost of the different components that are used by the algorithm like constant coefficient multipliers and adder trees.

In a Xilinx FPGA, the smallest logic element unit is a "slice." It consists of two 4-LUTs, two storage elements (flip-flops) and some other components like carry logic, logic gates and multiplexers [Xil]. In the current cost model, a slice is further decomposed into two half-slices, each consisting of a single 4-LUT and a flip-flop. The model tracks the used LUT and flip-flop components of each half-slice in order to estimate the required design area. A half-slice does not need to be fully utilized. Either the flip-flop or the LUT can be used by the design. A LUT connected to a flip-flop counts as a half-slice. A flip-flop connected to two flip-flops on either side, as in delay element, counts as a half-slice, with its LUT component not being used. Similarly, a LUT having LUTs at its input and output, as in combinatorial block, is also considered as a half-slice with its flip-flip not being used.

The current model predicts a resource usage within 3% of the actual cost when the component is synthesized and placed on the device. The cost estimate of individual components is then aggregated to obtain the cost estimate of the final decomposition. Experiments show that the worst case system-level estimate deviates by 22% of the actual area, with a mean value of 12%. This is because the synthesis and place and route tools optimize further the design beyond the scope of the current cost model, which has to be relatively fast, and tracks only the LUT and flip-flop components of a slice. It should be noted that

[3]http://www.xilinx.com/support/documentation/user_guides/ug002.pdf

different cost models for different devices can be easily accommodated by the system.

4.6 Frequency vs. Area

Another dimension in hardware design is the maximum achieved frequency of the design. Due to the nature of the 2D convolution, the design can be pipelined very deeply. In the current framework, the algorithm can be tuned for a specific frequency using the minimum required area.

The framework uses delay models for all the components and inserts extra registers whenever a particular component is likely to exceed the predefined time delay. It should be noted that the delay estimation is only an approximation since the routing delay, which depends on the relative placement of the components, has not been modeled. Simulations show that for a 9×9 Gabor filter targeting to a specific noise variance at the output of the system, a change in the clock rate from 90MHz to 57MHz leads to an area saving of 26%. More results regarding the frequency versus area trade-off are illustrated in the performance evaluation section.

4.7 Adder Trees

A component with a large contribution to the final cost of the design is the adder tree that is required after each 1D or 2D FIR stage. The proposed algorithm minimizes the cost of the adder tree by clustering together intermediate results that have similar precisions. This leads to the construction of more "compact" adder trees. This property has also been exploited by the Precision Equalizing Algorithm.

4.8 Area Reduction vs. Separability

The performance of the proposed algorithm is also a function of the separability properties of the 2D filters. The more separable the filters are, the more reduction in area can be achieved. Considering a single 2D filter, its rank gives only a lower bound for the separable levels that are needed in order to approximate the filter, due to the quantization of the filter coefficients. The proposed algorithm exhaustively searches a predefined set of separable levels, and returns the designs that achieve the lowest variance at the output of the system, for a specific area constraint. Thus, in the worst case scenario, where a 2D filter has a full rank, the proposed algorithm will return a design that is no worse than a design obtained using current techniques.

5. PERFORMANCE EVALUATION

The performance of the proposed algorithm is compared to a direct pipelined implementation of a 2D convolution using Canonic Signed Digit recoding for the constant coefficient multipliers. In the case of multiple 2D filters, an equivalent number of 2D convolution designs are implemented and their combined performance is compared against the proposed architecture.

The main target of the proposed algorithm is the computer vision field. Thus, the filters that are used to assess the performance of the algorithm

Table I. Test Sets

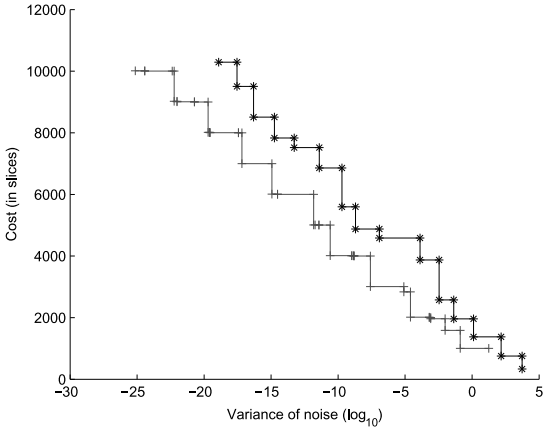
Tests 1 and 2 are used to assess the performance of the algorithm in the case of a single filter, while Test Cases 3 to 6 are used for multiple filter evaluation. Tests 1 to 4 correspond to common convolution operators in the computer vision field, while Test Cases 5 and 6 correspond to specific applications drawn from the computer vision field.

Test Number	Description
1	One 9×9 Gabor filter $F(x, y) = \alpha \sin\theta e^{-\rho^2\left(\frac{\alpha}{\sigma}\right)^2}$, $\rho^2 = x^2 + y^2$, $\theta = \alpha x$, $\alpha = 4$, $\sigma = 6$
2	One 9×9 Laplacian of Gaussian filter $LoG(x, y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$, $\sigma = 1.4$
3	Four 9×9 Gabor filters $F_i(x, y) = \alpha \sin(\theta_i) e^{-\rho^2\left(\frac{\alpha}{\sigma}\right)^2}$, $\rho^2 = x^2 + y^2$, $\theta_i \in \left\{\alpha x, \alpha\sqrt{2}(x + y), \alpha y, \alpha\sqrt{2}(y - x)\right\}$ $\alpha = 4$, $\sigma = 6$
4	Four 15×15 Gabor filters $F_i(x, y) = \alpha \sin(\theta_i) e^{-\rho^2\left(\frac{\alpha}{\sigma}\right)^2}$, $\rho^2 = x^2 + y^2$, $\theta_i \in \left\{\alpha x, \alpha\sqrt{2}(x + y), \alpha y, \alpha\sqrt{2}(y - x)\right\}$ $\alpha = 6$, $\sigma = 20$
5	Four 15×15 filters from a steerable pyramid for edge detection [Bouganis et al. 2004]
6	Six 15×15 filters describing inhibition neural connections [Li 2003]

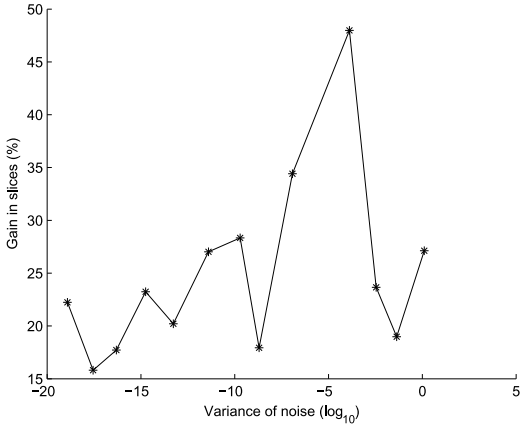
are drawn from real computer vision applications. The test sets are shown in Table I. Tests 1 and 2 evaluate the performance of the algorithm in the case of a single filter. The Gabor and the Laplacian of Gaussian filters have been selected because of their extensive use in computer vision applications. A convolution with a Gabor filter yields images which are locally normalized in intensity and decomposed in terms of spatial frequency and orientation [Gong et al. 2000]. The Laplacian of Gaussian filter is used mainly for edge detection. The rest of the test set evaluate the performance of the algorithm in the case of multiple filters. Tests 3 and 4 focus on Gabor filters, where tests 5 and 6 focus on filters that are used in specific applications. Test set 5 is used in Bouganis et al. [2004], where the authors use the filters to decompose an image into scale and orientation selective channels for scene analysis. Test set 6 refers to a set of six filters that are used in the case of pre-attentive segmentation in Primary Visual Cortex [Li 2003]. The selected test sets for performance evaluation comprise a representative set of filters that vary from commonly used filters to real application-specific filters.

5.1 Area Gain Evaluation

Tests 1 and 2 are used to assess the performance of the proposed algorithm for the case of a single filter. Figure 9(a) shows the achieved variance of the error at the output of the filter as a function of the area, for the proposed and



(a)



(b)

Fig. 9. (a) Achieved variance of the noise at the output of the design versus the area usage of the proposed design (+) and the reference design (*) for test case 1. (b) illustrates the percentage gain in slices of the proposed framework for different values of the variance of the noise.

the reference algorithms. It can be concluded that in all cases the proposed algorithm leads to designs that use less area than the reference algorithm, for the same variance of the error at the output. Figure 9(b) illustrates the relative reduction in area achieved by the proposed algorithm for different values of the noise variance. The oscillation in the figure is due to the discrete nature of the design space exploration. An average reduction of 24.95% and 12.28% is achieved for the test case 1 and 2 respectively. Alternative, the proposed methodology produces designs with up to 50dB improvement in the signal to noise ratio requiring the same area in the device with designs that are derived from the reference algorithm. Moreover, test case 1 was used for evaluation of the performance of the algorithm when embedded multipliers are available. 30 embedded multipliers of 18×18 bits are made available in the algorithm.

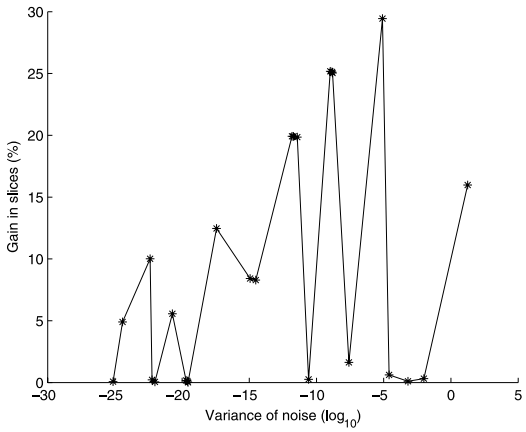
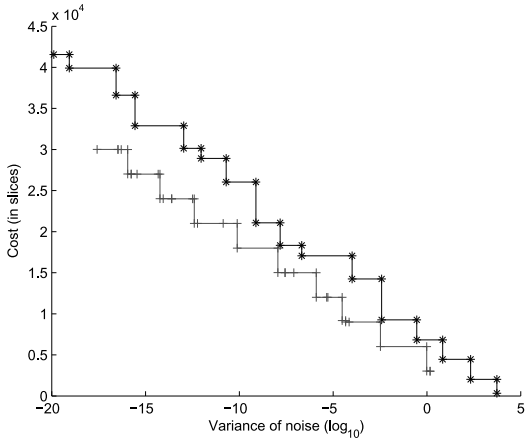


Fig. 10. Percentage gain in slices of the proposed framework for test case 1 for different values of the variance of the noise between a design that uses 30 embedded multipliers and a design that uses none.

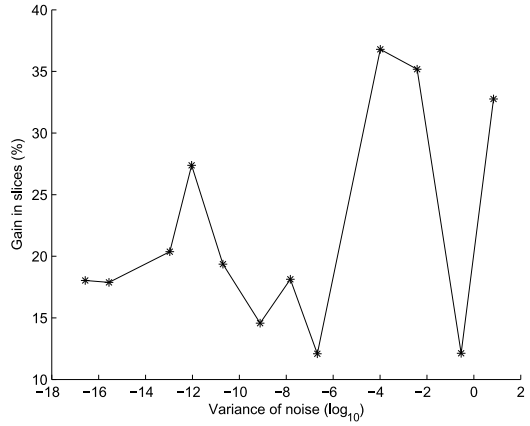
The relative percentage reduction achieved by the algorithm between designs that use the embedded multipliers and designs that realized without any embedded multiplier is illustrated in Figure 10 for different values of the noise variance. The oscillation in the figure is due to the discrete nature of the design space exploration. It can be concluded that the algorithm explores efficiently the available embedded multipliers leading to significant gains in the required area. The average gain in slices achieved by the algorithm is 9.35%. It should be noted that the impact of the embedded multipliers is more evident in the larger variances of the noise, since a LUT based constant coefficient multiplier using Canonic Signed Digit recoding leads to more accurate results after a certain word-length of the constant coefficient. Thus, designs targeting very small variance of the noise at the output are implemented more efficiently using LUT based constant coefficient multipliers.

Tests 3 and 4 evaluate the performance of the algorithm in the case of multiple filters. Test case 3 evaluates the algorithm for a case of four 9×9 filters, where test case 4 uses four 15×15 filters. Figure 11(a) illustrates the achieved mean value of the variance of the error at the output of the system for a given area constraint for the proposed and the reference algorithms. It can be concluded that in all the cases, the proposed algorithm explores the redundancy in the filters and produces superior results than the reference algorithm. Figure 11(b) illustrates the relative percentage reduction in area achieved by the proposed algorithm for different values of the noise variance. The achieved average gain in slices is 20.06% and 41.87% for test case 3 and 4 respectively. The above results show that in the case of large filters the reduction achieved by the proposed algorithm is more evident, since the existing redundancy is larger.

Finally, the proposed algorithm is tested using kernels from specific computer vision applications. Test cases 5 and 6 ensemble two of these cases.



(a)



(b)

Fig. 11. (a) Achieved variance of the noise at the output of the design versus the area usage of the proposed design (+) and the reference design (*) for test case 3. (b) illustrates the percentage gain in slices of the proposed framework for different values of the variance of the noise.

Figure 12 shows the performance of the proposed algorithm for the two cases. In both cases, the proposed algorithm outperforms the reference algorithm, regardless of the area constraint. The mean relative percentage gain in slices is 38.01% for test case 5 and 13.57% for test case 6.

These experiments demonstrate the power of the proposed algorithm to exploit the redundancy inside a convolution filter, and between different filters in order to produce better designs in terms of area and accuracy. Moreover, it exploits the heterogeneous structure of modern FPGAs regarding the embedded multipliers by producing an order for the coefficients regarding their impact to the final approximation error and allocating appropriately the embedded multipliers.

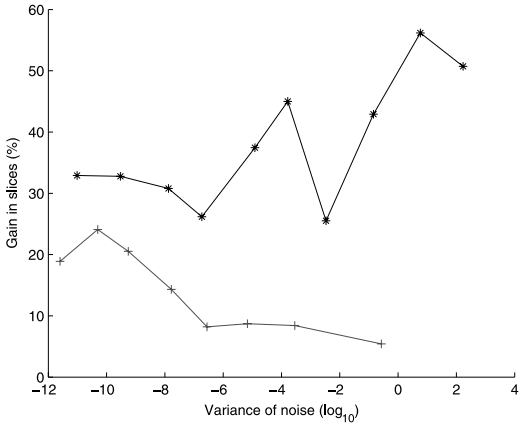


Fig. 12. Percentage gain in slices of the proposed framework for different values of the variance of the noise for test cases 5 (*) and 6 (+).

5.2 Algorithmic Considerations

As has been described in Section 4.4, the proposed methodology performs a search over the number of decomposition levels in order to find the one that produces the minimum variance of the error at the output of the system for a given set of resources. In the case where infinite precision arithmetic was used and under the assumption that all multiplication blocks require the same area for implementation, the above search strategy would be unnecessary since the area and the error of the system are monotonic functions of the number of the decomposition levels. However, due to the quantization effects and the varying area costs of the multiplication blocks, the area and the error of the system are not monotonic functions of the number of the decomposition levels anymore. Thus, the algorithm has to explore the design space by iterating through the decomposition level space when the optimum design in terms of area is required for a given set of resources.

Figure 13 illustrates the area usage of the design versus the variance of the noise at the output of the system for test case 1. The graph is annotated with the different decomposition levels that are used at each design point. This demonstrates the necessity to iterate through the decomposition level space in order to find the optimal decomposition for a specific set of resources.

5.3 Cost Model Accuracy

The accuracy of the cost model is important for the validity of the results. The cost model has been developed separately for each component in the filter design, without exploiting any further optimization when different components are combined together. Thus, it is expected that the estimated cost of the design to deviate from the actual cost. Test 1 and test 3 are used to assess the accuracy of the estimated cost model. Figure 14 illustrates the distribution of the deviation between the predicted cost of the design using the proposed model and the the actual cost of the design reported by the Xilinx back-end

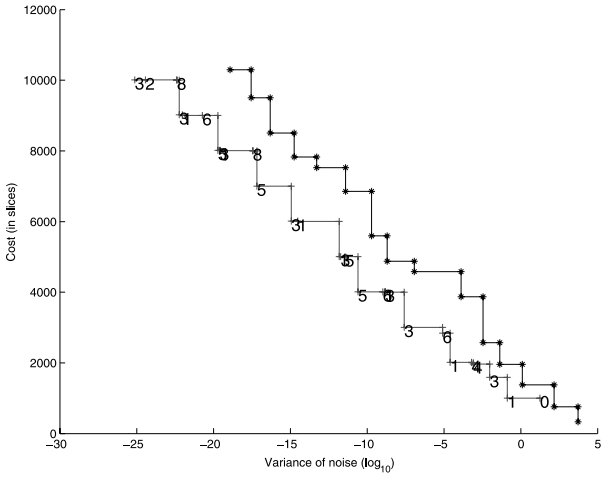


Fig. 13. Achieved variance of the noise at the output of the design versus the area usage of the proposed design (+) and the reference design (*) for test case 1. The numbers on the graph refer to the number of decomposition levels that are used in each design point.

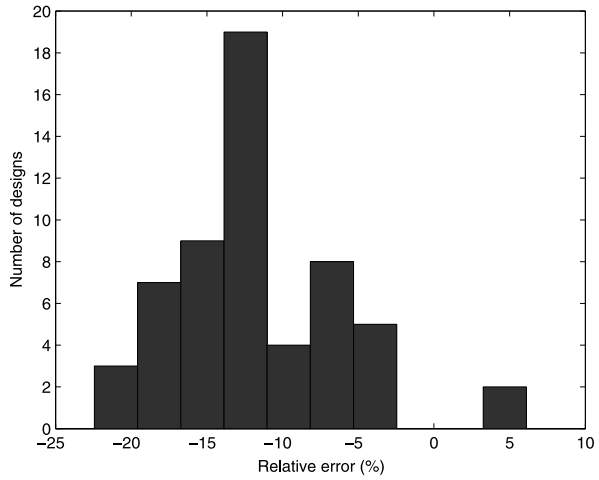


Fig. 14. Distribution of the error in the area prediction for a design. Relative Error $= \frac{\text{Actual area} - \text{Predicted area}}{\text{Actual area}} * 100\%$.

tools. The target device is a XC2V6000 FPGA, and 57 designs were used. 32 designs from test case 1 (single filter) and 25 designs from test case 3 (multiple filters). It can be concluded that the model usually overestimates the actual area of a design by a mean relative error value of 12%. The framework's overestimation in the area of the design is expected due to the extra optimizations that are applied by the back-end mapping tools. However, it should be noted

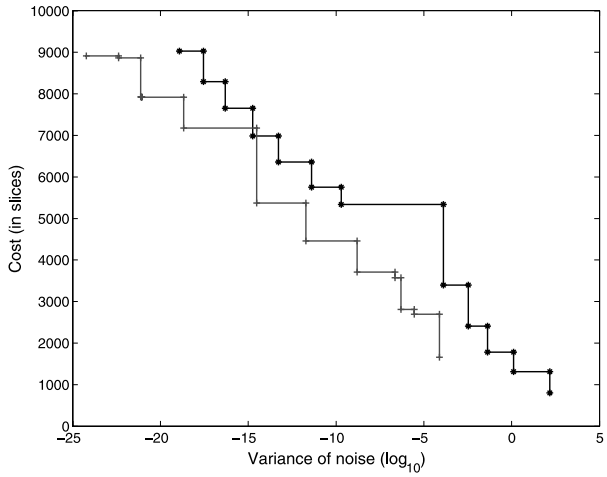


Fig. 15. Achieved variance of the noise at the output of the system versus the area usage of the proposed design (+) and the reference design (*) for test case 1. The designs have been synthesized and placed and routed using the Xilinx tools ISE 8.1. The target device is a Xilinx device XC2V6000-4.

that the distribution of the deviation is narrow, implying certain confidence on the decisions of the algorithm during the optimization and its results.

Using the proposed algorithm, the obtained designs for test case 1 were synthesized and placed and routed on a Xilinx XC2V6000-4 device. Also, the designs obtained by the reference algorithm were synthesized for the same target device. Figure 15 shows the achieved area usage of the designs versus the variance of the noise at the output of the system. In all the cases, the proposed algorithm produces designs that require less area than the reference algorithm for the same variance of the error at the output of the system. The different shape of the plots between Figure 15 and Figure 9(a) is explained by the use of Pareto curve.

5.4 Frequency vs. Area of the Design

Depending on the user's requirements, the proposed framework has the option to trade area for frequency and vice versa. This is achieved by pipelining the design to different depths. The proposed framework tracks all the components in the design, and inserts extra pipeline stages in those that are likely to exceed the user's target clock period. The delay estimation is an approximation of the actual delay and routing delay between the components has not be modeled since it depends on the relative placement of the components in the device. Figure 16 illustrates the frequency versus area trade-off for test case 1, when the variance of the noise at the output of the system is fixed to $\sigma^2_{noise} = 10^{-7}$. The target device is a Xilinx device XC2V6000-4, and the Xilinx tools ISE 8.1 were used for synthesis, placement, and routing. The graph demonstrates the ability of the framework to explore the area-speed design space and to produce a design that targets the user requirements.

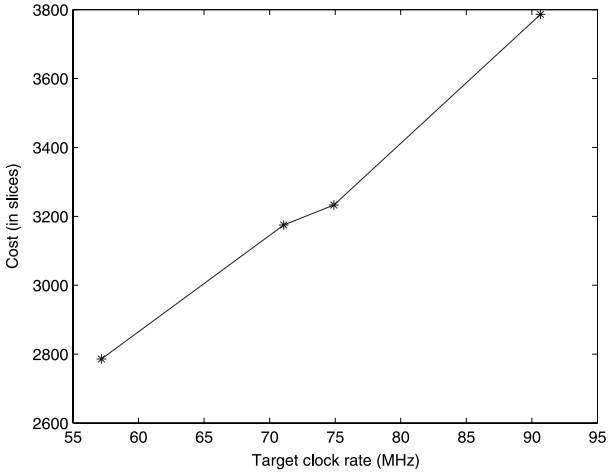


Fig. 16. Speed-area trade-off for test case 1. The noise variance at the output is fixed to $\sigma^2_{noise} = 10^{-7}$. The target device is a Xilinx device XC2V6000-4, and the Xilinx tools ISE 8.1 were used for synthesis, placement, and routing.

5.5 Power Issues

The power consumption of the designs derived by the proposed methodology is evaluated and is compared against designs that are derived by the reference algorithm. Test case 1 is used for the evaluation and comparison, the target device is a XC2V6000-4 from Xilinx, the back-end tools from Xilinx are used for placement and routing, and the XPower tool from Xilinx is used for the estimation of the total power consumption. Post place and route information is used to fully capture glitching effects, and the stimuli of each design is a 256×256 image of Lenna, a widely used standard test image in image processing community. Figure 17 illustrates the total power consumption of designs derived by the proposed methodology and the reference algorithm. The results show that the proposed methodology produces designs with up to 40dB improvement in the signal to noise ratio consuming the same power with designs that are derived from the reference algorithm. On average, a 53% reduction on the consumed power is achieved assuming designs that target uniformly placed noise variance at the output of the system. This makes the proposed methodology a better choice for DSP designers that target low-power applications.

5.6 Targeting an Altera Device

A high-level area model for a Stratix III device from Altera was developed and incorporated into the proposed framework. The high-level area model has a similar structure to the Xilinx's model but modifications have been made to address the specific architecture of the new device. Experiments have demonstrated a maximum relative error of 15% in the area prediction, which outperforms the corresponding Xilinx high-level area model. Figure 18 shows

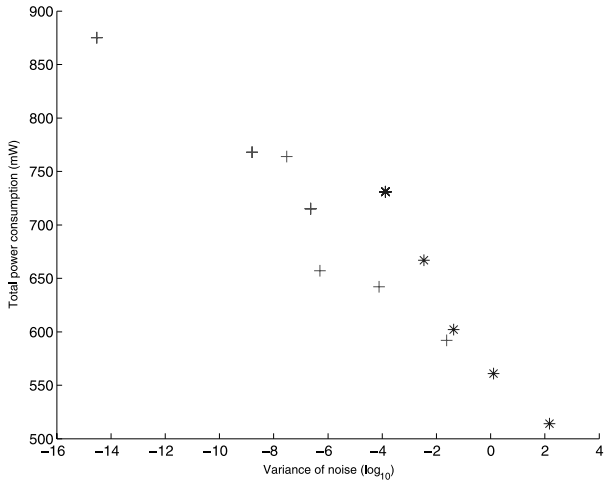


Fig. 17. Total power consumption for test case 1. (+) indicates designs derived using the proposed methodology, where (*) indicates designs derived using the reference algorithm. On average, a 53% reduction on the consumed power is achieved assuming designs that target uniformly placed noise variance at the output of the system. The target device is a Xilinx device XC2V6000-4, and the Xilinx tools ISE 8.1 were used for synthesis, placement, routing, and power estimation.

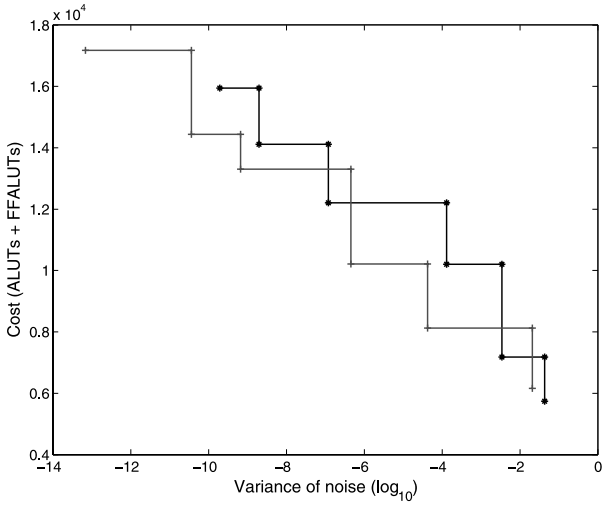


Fig. 18. Achieved variance of the noise at the output of the system versus the area usage of the proposed design (+) and the reference design (*) for test case 1. The designs have been synthesized and placed and routed using the Altera tools Quartus II (v7.1). The target device is a Stratix III (EP3SE50). Please note that the two points (+) that correspond to the proposed design and are above the reference line are due to the Pareto curve and do not correspond to actual designs.

the achieved area usage of the designs versus the variance of the noise at the output of the system for test case 1. The obtained results correspond to synthesized and placed and routed designs on an Altera Stratix III device (EP3SE50). In all the cases, the proposed framework produces designs that require less

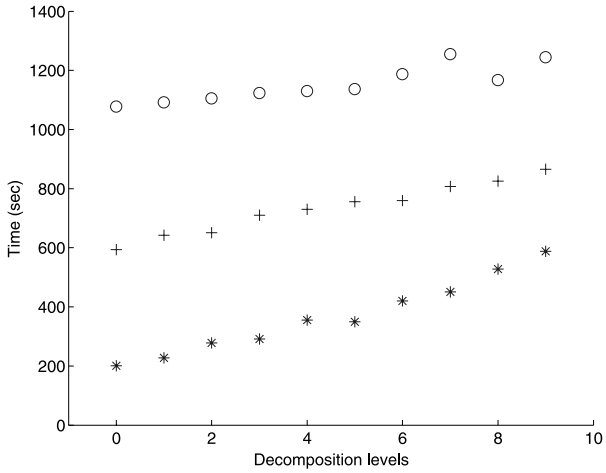


Fig. 19. Time required for exploration of different decomposition levels for test cases 1 (*), 3 (+), and 4 (o). The algorithm was implemented in MATLAB running in an Intel Core 2 CPU at 1.86GHz.

area than the reference algorithm for the same variance of the error at the output of the system.

5.7 Computational Requirements

The required time for exploration of the design space using the proposed algorithm depends on the set of the input filters and the set of targeted resources. Figure 19 illustrates the required run-time of the algorithm for Test cases 1, 3, and 4 targeting different number of decomposition levels. A zero decomposition level implies that only the error term is used in the approximation. In all the cases the target resource is 30,000 slices. The total times for test case 1, 3, and 4 were approximately 1, 2, and 3.2 hours respectively. The algorithm was implemented in MATLAB running on an Intel Core 2 CPU at 1.86GHz PC.

6. CONCLUSION

This article presents a novel 2D filter design methodology for heterogeneous FPGA devices. The methodology explores the computational structure of a 2D filter according to the different types of available resources in the device. It employs the use of a Singular Value Decomposition based algorithm which produces a decomposition of the filter that maps directly to the different components of modern reconfigurable devices. Furthermore, it extends the work to designs with multiple filters by exploiting any redundancy that exists within each filter and between different filters in the set. Experiments with filter sets from computer vision applications demonstrated a reduction of 13.57% and 38% on average in the required area. Future work will involve the use of word-length optimization techniques [Constantinides et al. 2003] and the use of current techniques for high-speed multiplication [Dempster and Macleod 1995] to further enhance the algorithm's performance. Moreover, due to the

sum of squares error criterion that is used, the proposed method is more appropriate for channel equalization applications. However, a different criterion like a min/max approximation of the filter's frequency response can be accommodated by slightly modifying the proposed algorithm.

REFERENCES

ANDREWS, M. S. 1999. Architectures for generalized 2d fir filtering using separable filter structures. In *Proceedings of the Acoustics, Speech, and Signal Processing IEEE International Conference*. Vol. 4. 2215–2218.

BELHUMEUR, P., HESPANHA, J., AND KRIEGMAN, D. 1997. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Trans. Patt. Anal. Mach. Intell. 19*, 7, 711–720.

BOUGANIS, C.-S., CHEUNG, P. Y. K., AND CONSTANTINIDES, G. A. 2005. Heterogeneity Exploration for Multiple 2D Filter Designs. In *Proceedings of the Conference on Field Programmable Logic and Applications*. 263–268.

BOUGANIS, C.-S., CHEUNG, P. Y. K., NG, J., AND BHARATH, A. A. 2004. A Steerable Complex Wavelet Construction and its Implementation on FPGA. In *Proceedings of the Conference on Field Programmable Logic and Applications*. 394–403.

BOUGANIS, C.-S., CONSTANTINIDES, G. A., AND CHEUNG, P. Y. K. 2005a. A Novel 2D Filter Design Methodology. In *Proceedings of the International Symposium in Circuits and Systems*. 532–535.

BOUGANIS, C.-S., CONSTANTINIDES, G. A., AND CHEUNG, P. Y. K. 2005b. A Novel 2D Filter Design Methodology For Heterogeneous Devices. In *Proceedings of the Conference on Field-Programmable Custom Computing Machines*. 1–10.

CHEN, C.-L., KHOO, K.-Y., AND A.N. WILLSON,JR. 1995. An improved polynomial-time algorithm for designing digital filters with power-of-two coefficients. In *Proceedings of the IEEE International Symposium on Circuits and Systems*. Vol. 1. 223–226.

CMAR, R., RIJNDERS, L., SCHAUMONT, P., VERNALDE, S., AND BOLSENS, I. 1999. A methodology and design environment for DSP ASIC fixed point refinement. In *Proceedings of the Design, Automation, and Test in Europe*. 271–276.

CONSTANTINIDES, G. A., CHEUNG, P., AND LUK, W. 2001. The Multiple Wordlength Paradigm. In *Proceedings of the Conference on Field-Programmable Custom Computing Machines*. 51–60.

CONSTANTINIDES, G. A., CHEUNG, P. Y. K., AND LUK, W. 2003. Wordlength Optimization for Linear Digital Signal Processing. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst. 22*, 10.

DEMPSTER, A. AND MACLEOD, M. D. 1995. Use of minimum-adder multiplier blocks in FIR digital filters. *IEEE Trans. Circ. Syst. II 42*, 569–577.

GONG, S., MCKENNA, S., AND PSARROU, A. 2000. *Dynamic Vision: From Images to Face Recognition* 1st Ed. Imperial College Press.

HASEYAMA, M. AND MATSUURA, D. 2006. A filter coefficient quantization method with genetic algorithm, including simulated annealing. *IEEE Signal Process. Lett. 13*, 4, 189–192.

HASTAD, J. 1990. Tensor rank is NP-complete. *J. Algo. 11*, 4, 644–654.

JA'JA, J. 1978. Optimal evaluation of pairs of bilinear forms. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing*. 173–182.

KODEK, D. 1980. Design of Optimal Finite Wordlength FIR Digital Filters Using Integer Linear Programming Techniques. *IEEE Trans. Acoust. Speech Signal Process. 28*, 304–308.

KOREN, I. 2002. *Computer Arithmetic Algorithms* 2nd Ed. Prentice-Hall Inc.

KUM, K.-I. AND SUNG, W. 2001. Combined word-length optimization and high-level synthesis of digital signal processing systems. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst. 20*, 8, 921–930.

LI, D., LIM, Y. C., LIAN, Y., AND SONG, J. 2002. A polynomial-time algorithm for designing fir filters with power-of-two coefficients. *IEEE Trans. Acoust. Speech Signal Process. 50*, 8, 1935–1941.

LI, Z. 2003. V1 mechanisms and some figure-ground and border effects. *J. Physiol. Paris 97*, 503–515.

MITRA, S. K. 2002. *Digital Signal Processing: A Computer-Based Approach* 2nd Ed. McGraw-Hill Higher Education.

PARK, I.-C. AND KANG, H.-J. 2001. Digital filter synthesis based on minimal signed digit representation. In *Proceedings of the Annual ACM IEEE Design Automation Conference*. 468–473.

PASKO, R., SCHAUMONT, P., DERUDDER, V., VERNALDE, S., AND DURACKOVA, D. 1999. A new algorithm for elimination of common subexpressions. *IEEE Trans. Comput.-Aid. Des. Integra. Circ. Syst. 18*, 1, 58–68.

PRESS, W., TEUKOLSKY, S., VETTERLING, W., AND FLANNERY, B. 1992. *Numerical Recipes in C*. Cambridge University Press.

SAMUELI, H. 1989. An improved search algorithm for the design of multiplierless fir filters with powers-of-two coefficients. *IEEE Trans. Circ. Syst. 36*, 7, 1044–1047.

SHASHUA, A. AND LEVIN, A. 2001. Linear image coding for regression and classification using the tensor-rank principle. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. Vol. I. IEEE, 42–49.

SIOHAN, P. 1990. An analysis of coefficient inaccuracy for 2-d fir direct form digital filters. *IEEE Trans. Circ. Syst. 37*, 10, 1308–1313.

STRANG, G. 1998. *Introduction to Linear Algebra*, 3rd ed. Wellesley-Cambridge Press.

YURDAKUL, A. 2005. Multiplierless implementation of 2-d fir filters. *Integration VLSI J. 38*, 4, 597–613.

Received November 2007; revised February 2008, August 2008; accepted September 2008