Computational Time Analysis of Medical Images Implemented on FPGA Architecture

Dr.B.Vijayakumari
Asst. Professor, Department of ECE,
Mepco Schlenck Engineering college,
Sivakasi, India.
vijayakumari@mepcoeng.ac.in

R.B.Aswathy,
PG Scholar, Department of ECE,
Mepco Schlenck Engineering college,
Sivakasi, India.
achurabi123@gmail.com

***Abstract*— Image analysis with FPGA is being considered as a major attention in the current scenario. In this work, the computation time of enhanced bone images using Field-Programmable Gate Array (FPGA) architecture has been analyzed. Semivariogram analysis is known to be a computationally intensive algorithm that has typically seen applications in the geosciences, remote sensing areas and also in the area of medical image processing. This work mainly focuses on the reduction of computational time analysis with improved resource utilization. Complex computations are implemented using contemporary FPGA's which consists of large number of logic gates resources and RAM blocks. The proposed method enhances the input bone image using pre-processing technique with semivariogram algorithm and compares the computational time analysis of FPGA using pipelined and non-pipelined architecture and their results are validated and implemented on Spartan-6 Field-Programmable Gate Array (FPGA) platform.**

Keywords—semivariogram algorithm; FPGA architecture; pipelined and non-pipelined architecture; bone images.

I. INTRODUCTION

Texture is an important pre-attentive cue in human and machine vision and is defined as the characteristic spatial arrangements of features which can distinguish one image from other images. Early attempts to use textural features for image classification were based on second-order statistics derived from images [1]. There are two types of textures based on spatial frequency, namely, fine and coarse. Fine textures have high spatial frequencies or a high number of edges per unit area. Coarse textures have low spatial frequencies or a small number of edges per unit area. Several probability models for textures have gained wide acceptance, because of their modeling power and expressiveness [2]. Statistical properties containing the problem of texture analysis provide access to a wide range of well-established theories and methodologies in mathematical statistics to be introduced into texture modeling [3]. In particular, Markov Random Fields (MRFs) describe a texture in terms of spatial geometry and quantitative strengths of inter-pixel statistical dependency [4]. The semivariogram is a method that can be applied to two dimensional plain-projection images and is based on Markov Random Fields (MRF) [5]. Texture analysis can also be utilized to investigate bone micro architecture, and recently, applications in the area of medical imaging have been investigated [6].

Semivariogram analysis is a computationally intensive statistical algorithm that has typically seen applications in the geosciences and remote sensing areas [7, 8]. Fast computation of variogram based on the Fast Fourier Transform (FFT) algorithm as opposed to the spatial domain [7]. It can be used as a descriptor of second-order statistics within the image and hence provide a quantitative measure of texture. Stochastic parameters such as correlation length/range, nugget and sill calculated from the semivariogram plot can be used to represent spatial variations in bone images [9]. Usually the hardware implementation is desirable to optimize the performance in terms of speed, area and power dissipation. Reconfigurable platforms such as Field-Programmable Gate Arrays (FPGAs) have been investigated for implementing many image processing algorithms [10]. An FPGA consists of an array of reconfigurable logic cells and advances in technology, and development tools have made it possible to implement complex algorithms on FPGAs [11]. Because FPGA designs can be optimized for a given application, they often have superior performance in terms of speed and power dissipation over generic integrated circuit designs and microprocessor based implementations. FPGA systems have been demonstrated to have higher throughput compared with DSP and general-purpose microprocessor based system designs [12]. For this reason, FPGAs are increasingly being used in areas formerly dominated by Application Specific Integrated circuits (ASICs), such as embedded system design and digital image processing applications. The other literature studies include the following collection of information

In summary, there are a number of approaches that have been implemented to compute second-order statistics from images, but they are mostly limited to GLCMs due to their popularity. Only one of the approaches directly addresses semivariogram computation for Kriging applications with a suggested theoretical architecture. The

FPGA board used for the implementation is Xilinx Spartan-6 starter kit. In this paper, the computation time is analyzed in case of pipelined and non-pipelined architecture to get the better result compared to that of mathematical calculation.

II. SEMIVARIOGRAM ALGORITHM

The semivariogram displays the average change of a property and relation between a pair of pixels with changing lag (the distance between a pair of pixels). The semi variance typically increases with lag distance and converges to the sum of the nugget variance and the sill variance when the separation distance (h) approaches infinity. The variogram describes the spatial dependence of data and gives the range of spatial correlation, within which the values are correlated with each other and they are independent in nature.

The physical distance between the two locations (x_i, x_i+h) in an image is known as the lag distance or lag and is denoted as h. The semivariogram, denoted as (h), displays the average change of a property with changing lags, and characteristics are specific to the application domain under consideration. The experimental variogram is calculated by averaging one half the differences squared of the pixel values $z(x_i)$ over all pairs of observations with the specified lag distance and orientation. The relation between a pair of pixels that are lag distance h apart can be given by the average variance of the difference between all such pairs and is expressed as follows:

$$\gamma(h) = \frac{1}{2m}\sum_{i=1}^{m}[z(x_i) - z(x_i + h)]^2 \qquad (1)$$

Where m is the total number of pixel pairs used in the computation of the sum. The value of m is governed by the geometrical properties of the region of interest in the image used for the statistical analysis and the value of h. Given the same region, larger values of h will result in lower values of m since the dipole connecting the pixels in the pair may not lie within the region of interest for certain pixels.

There are few constraints for a semivariogram model to be able to represent typical measured data. These include:

a. a monotonic increase with increasing lag distance from the ordinate
b. an asymptotic maximum or ‘‘sill’’
c. a positive intercept on the ordinate or ‘‘nugget’’
d. periodic fluctuation or a ‘‘hole’’ and anisotropy.

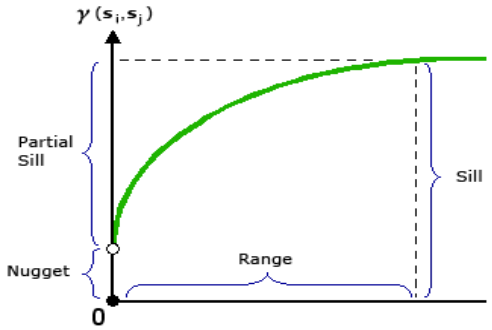


Fig. 1 Graph representing semi-variogram model

The general diagram representing the semivariogram model with nugget, sill and range as stochastic parameter is shown in figure (1). The distance where the model first flattens out is known as the range or correlation length. Sample locations separated by distances closer than the range are spatially auto-correlated. It also describes the degree of smoothness or roughness in an image. Large correlation length implies a smooth variation, whereas a small correlation length implies variations of Bone Mineral Density (BMD) map over spatial domain. The value on the y-axis is called the sill. It can be used to refer to the ‘‘amplitude’’ of certain component of the semivariogram. The sill of the variogram represents the variance of the BMD map [13].

According to theory, the semivariogram value at the zero separation distance (lag = 0) should be zero. However, at an infinitesimally small separation distance, the semivariogram often exhibits a nugget effect, which is some value >0. The nugget represents variability at distances smaller than the typical sample spacing. Variation at micro scales smaller than the sampling distances will appear as part of the nugget effect. It represents the sum of noise in a image and measurement errors of calculations. More commonly, the variogram values are computed for each distance value by averaging the values obtained for different angles from 0^0 to 360^0 for the same distance value. Such a variogram is considered to be isotropic in nature and is used for bone image analysis. Each difference value will have a distance value associated with it equal to the Euclidian distance between the pixels (lag distance). Based on the lag distance information, the difference value can then be assigned for computation of a particular variogram for that lag distance. In practice, this can be implemented as an accumulator to achieve the summation of Eqn. (1). A count of the number of pixel pairs for each lag distance corresponding to m in the equation would have to be maintained and would vary based on the lag distance.

The computational complexity is still O (n^2) where n is the number of pixels in the image region, but multiple passes through the image for each lag distance are avoided, and parallel implementation is now possible. Table 1 shows

the computational complexity of the semivariogram algorithm for the isotropic computation case. Calculation of the variogram value includes the computation of the coordinate distances and gray level differences between all the pixel pairs, and summation of the squares of differences. Thus P = M x N is the number of pixels in the image, and there are P (P – 1)/2 unique pixel pairs

TABLE 1 performance measure for P=M x N images

Operation	Computational formula for P = M x N images
Unique pixel pairs	P (P-1) /2
Distance coordinate additions/subtractions	3 x (P (P-1) /2)
Euclidian distance squares/roots	3 x (P (P-1) /2)
Differences between pixel values	P (P-1) /2
Squares of differences	P (P-1) /2
Sum of squared differences	P (P-1) /2
Total number of additions	5 x (P (P-1) /2)
Total number of multiplications	2 x (P (P-1))

Each Euclidian distance calculation has 3 x (P (P - 1)/2) additions and 3 x (P (P - 1)/2) multiplications (squares and square roots are counted as one multiply operation). The calculation of the differences between the pixel values in each pair involves P (P - 1)/2 subtractions and P (P – 1)/2 multiplications. Calculation of the final variogram involves P (P - 1)/2 additions. Hence, the variogram value can be computed with 5 x (P (P-1) /2) sums and 2 x (P (P-1)) multiplications each. The maximum value of the lag distance h can be computed to be $\sqrt{(M}$. The computational complexity of the variogram calculation can be shown to be O (P^2), where P = M x N is the number of pixels in the image region. Large number of computations values makes the semivariogram calculation extremely time intensive. It is worthwhile investigating method to speed up the computation in order to make the semivariogram a practical technique for clinical usage. For this reason, FPGA architectures have been proposed to reduce the computational time for calculating the variogram values.

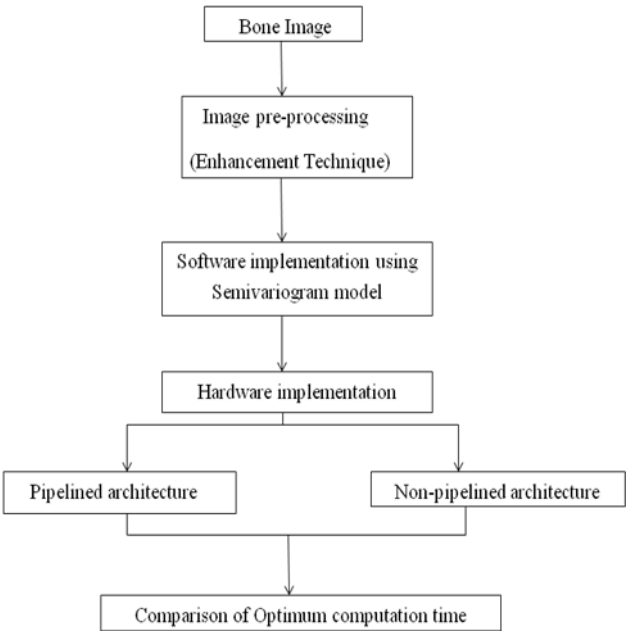


Fig. 2 Basic flow diagram

III. MATLAB SIMULATION

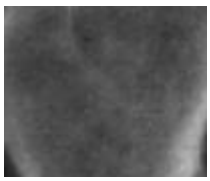
The proposed method involves MATLAB simulation with FPGA implementation under Spartan-6 FPGA kit. The bone images are given as input and the input image is enhanced and the semivariogram technique is applied. The input bone images are taken from different sources as X-RAY images, DXA bone images and also the region of interest from DXA image. The image enhancement consists of three techniques namely contrast enhancement, brightness adjustment, and the sharpening operation to get better resolution by interchanging the techniques order. The image processing has been done in two different mapping like,

1. Enhance contrast operation, Brightness adjustment, Sharpen operation (EBS)
2. Sharpen operation, Brightness adjustment, Enhance contrast operation (SBE)

The input dataset images are,

(a) (b)

(c) (d)



(e)

Fig. 3 Input images

The figure 3 represents the input bone images. From the Fig. 3 (a),(b),(c) are X-RAY images (d) indicates the DXA image and (e) indicates the region o f interest from the DXA image

Contrast operation:

Contrast adjustment remaps image intensity values to the full display range of the data type. An image with good contrast has sharp differences between black and white.

Brightness operation:

Brighten(beta) shifts the intensities of all colors in the current color map in the same direction. The colors gets bright when beta is between 0 and 1, and they darken when beta value is between -1 and 0. The magnitude of the change is proportional to the magnitude of beta.

Sharpen operation:

The luma intensity transitions should be more acute to sharpen the color image, while preserving the color information of the image. The word luma represents the brightness of an image. A low pass filter is applied to the luma portion in order to blur the image.

1. *EBS*

The input image is initially enhanced using contrast technique and the output is further processed to adjust the brightness and the final step involved is to sharpen the region required to get the better resolution.

2. *SBE*

The input image is initially enhanced using sharpening operation and output is further processed to adjust the brightness and the final step involved is to contrast the region required to get the better resolution.

From the above two pre-processing technique better resolution is obtained through EBS and their results are shown in figure (6). Using MATLAB, the pre-processing step as well as the software implementation of semivariogram algorithm has been performed.

IV. FPGA IMPLEMENTATION

Fast computations of semivariogram were designed using the FPGA architecture. Two architectures were designed and implemented in succession with the FPGA architecture as pipelined and non-pipelined architecture. Better results are obtained using the pipelined architecture. The pipelined architecture is constructed from non-pipelined architecture by just inserting the register between them. Both architecture were programmed using Verilog and their results are validated

Non-pipelined architecture:

The input pixel values are stored and their values are selected from the data register file as z (x_i) and z (x_i + h), where h is the lag distance. The value i is known to be a number of iteration that ranges from 1 to m, where m is the number of pixel values. The selected pixel value then performs 2's complement and squaring operation. The single iteration is known to be the combination of 2's complement and squaring and the consequent m number of iterations are summed. In case, this is applicable only for the small matrices whereas for large matrices the storage of intensity value become complex. The architecture representing the non pipelined structure is shown in figure (4). The summation result is then multiplied with the (1/ (2 x number of pixels)) to get the result of non-pipelined architecture.

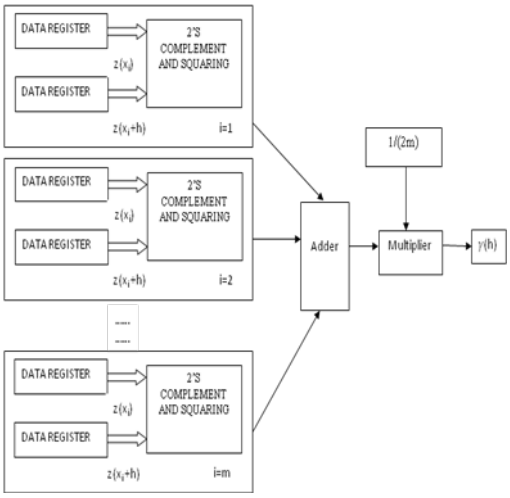
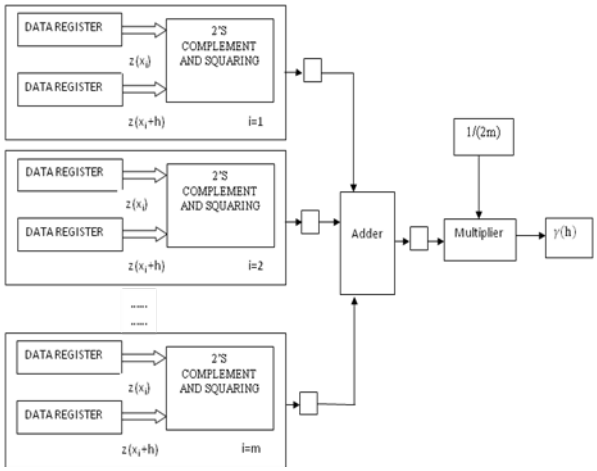


Fig. 4 Non-pipelined architecture

Pipelined architecture:

In general the pipelined architecture varies from non-pipelined architecture only by the addition of registers. Pipelining is a digital design technique that allows the designer to avoid data dependencies and increase the level of parallelism in an algorithmic hardware implementation. The data dependence is known for their functional equivalence, but the required circuit is divided into a chain of independent stages. All stages in the chain run in parallel on the same clock cycle. The only difference is the source of data for each stage.



☐ Indicates the register

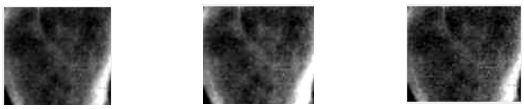
Fig. 5 Pipelined architecture

Each stage receives its data values from the result computed by the preceding stage. The pipelined architecture is shown in figure (5). The above architectures are implemented in 5 x 5 and their device utilization is illustrated in next section.

V. Results and Discussion

The results for pre-processing and software implementation using MATLAB and hardware implementation using pipelined and non-pipelined architecture are discussed in this section. From the two pre-processing techniques better resolution is achieved through EBS technique when compared to that of other technique named, SBE. The software implementation of semivariogram is done using MATLAB and their corresponding graph is plotted. Similarly the hardware implementation of semivariogram is done through non-pipelined and pipelined architecture and their computational value along with the device utilization summary for 5 x 5 image pixels are compared

a) EBS Enhancement Technique



(a) (b) (c)

Fig. 6 Pre-processing step done using EBS technique

From the figure, the (a) indicates the contrast enhanced input image where the considered input image is the region of interest from DXA (b) indicates the brightened image and (c) indicates the sharpened image

b) Software Simulation

The semivariogram algorithm is validated for 5 x 5 image using MATLAB. The X-axis represents the lag distance, h and the Y-axis represents the semivariogram

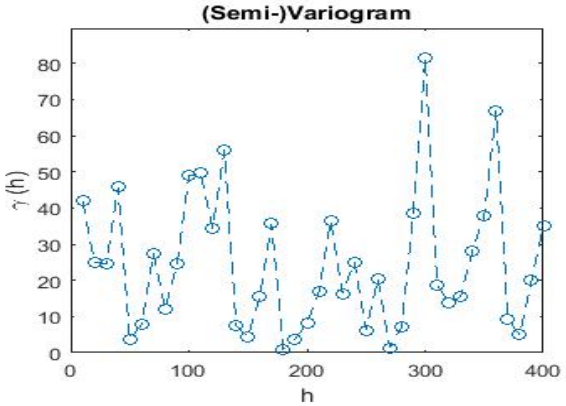


Fig. 7 Semivariogram using MATLAB

From the figure, the semivariogram value (h) value increases as the lag distance value h, is said to be increased. It is also stated that as the image pixel value increases the semivariogram value (h) also increases rapidly.

c) Hardware Implementation

The simulation result for 5 x 5 images is shown in Fig8 and the device utilization summary of pipelined and non-pipelined architecture is compared for the 5 x 5 image and is validated as follows

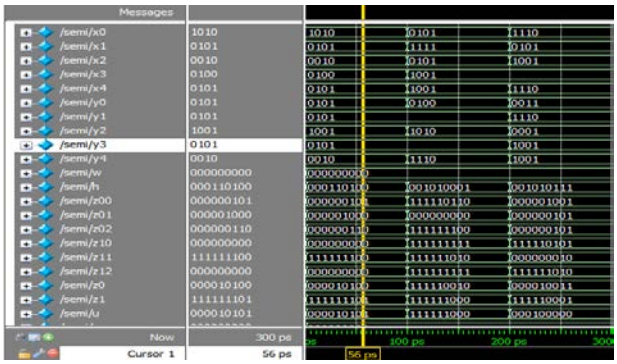


Fig 8 Simulation output

From the Fig 8, the simulation result of 5 x 5 image is shown. Where h is said to the result of semivariogram and X and Y indicates the input value. From the table 2 the number of slices and gate delay in pipelined architecture is said to less when compared to that of the non-pipelined architecture where as the memory utilization is more in pipelined compared to the non-pipelined architecture.

TABLE 2 Comparison for 5 x 5 image pixel

Parameters	Pipelined	Non- pipelined
Number of slices	52	129
Number of slice flip flops	72	116
Gate delay	13.450 ns	17.576 ns
Total memory usage	196580	186276

VI. Conclusion

Thus the device utilization for 5 x 5 image using pipelined and non-pipelined architecture is compared. The reduced computational time is obtained in case of small matrices whereas considering the large matrices, it is a challenging task. To overcome this, image dimensionality reduction can be carried out Further the efficiency of pipelined architecture can be improved by considering the floating point number to provide the precision value up to two decimal values.

VII. Acknowledgement

The authors wish to thank the Management and Principal of Mepco Schlenk Engineering College, for their support in carrying out this research work.

VIII. Reference

1. Haralick, R.M., Shanmugam, K., Dinstein, I.H.: Textural features for image classification. IEEE Trans. Syst. Man Cybern. 3(6), 610–621 (1973). doi:10.1109/TSMC.1973.4309314
2. Cootes, T., Taylor, C.: Anatomical statistical models and their role in feature extraction. Br. J. Radiol. (2014). doi:10.1259/bjr/20343922
3. Dong, X.N., Wang, X.: Biomechanical properties and micro architecture parameters of trabecular bone are correlated with stochastic measures of 2D projection images. Bone 56(2), 327–336 (2013). doi:10.1016/j.bone.2013.05.023
4. Marcotte,D.: Fast variogram computation with FFT. Comput.Geosci. 22(10), 1175–1186 (1996). doi:10.1016/S0098-3004(96)00026-X
5. Bohling, G.: Introduction to geostatistics and variogram analysis. Kansas Geol. Surv. 2 (2005)
6. Dong, X.N., et al.: Random field assessment of inhomogeneous bone mineral density from DXA scans can enhance the differentiation between postmenopausal women with and without hip fractures. J. Biomech. 48(6), 1043–1051 (2015)
7. Nelson.E, Implementation of Image Processing Algorithms on FPGA Hardware. Vanderbilt University, Nashville, TN, USA (2000)
8. Gothandaraman, A., Peterson, G.D., Warren, G.L., Hinde, R.J., Harrison, R.J.: A Pipelined and Parallel Architecture for Quantum Monte Carlo Simulations on FPGAs. VLSI Design 2010, 8 (2010). doi:10.1155/2010/946486
9. Rose, S.: Development of Parallel Image Processing Architecture in VHDL. University of West Australia, Perth, Australia (2012)
10. Pedroni, V.A.: Circuit Design and Simulation with VHDL. TheMIT Press, Cambridge, MA, USA (2010)
11. Wielgosz, M., Panggabean, M., Rønningen, L.A.: FPGA architecture for Kriging Image Interpolation. Int. J. Adv. Comput. Sci. Appl. (IJACSA). 4(12), 193–201 (2013)
12. Haralick, R.M.: Statistical and structural approaches to texture.Proc. IEEE 67(5), 786–804 (1979).
13. Mukul Shirvaikar, Yamuna Lagadapati, Xuanliang N.Dong.:Semivariogram analysis of bone images implemented on FPGA architectures DOI10.1007/s11554-016-0611-1.