Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi
International Journal of Engineering Research and Development

UMAGD, (2023) 15(2), 463-471.

10.29137/umagd.1165147

Cilt/Volume:15 Sayı/Issue:2 Haziran/June 2023

Araştırma Makalesi / Research Article

A Parallel Architecture for Improving the Performance of the Kriging Algorithm

Özgür Tamer[1*], A.Esat Genç[2],

[1]Elektrik Elektronik Mühendisliği Bölümü Dokuz Eylül Üniversitesi İzmir, TÜRKİYE

Başvuru/Received: *22/08/2023* ***Kabul / Accepted***: *12/03/2023* ***Çevrimiçi Basım / Published Online:*** *06/06/2023*

Son Versiyon/Final Version: *xx/xx/xxxx*

Abstract

Estimating missing data values by using interpolation algorithms is a well-known technique. Kriging is an optimized interpolation method based on regression against evaluated values from the surrounding observation points, weighted according to spatially varying values according to the covariance between these observation points. It has been widely used for estimating the missing geological data of the areas based on the measurements in close proximity. In this work we use the Kriging to recover the missing pixels of digital images. Even though Kriging is considered as successful on estimating the missing pixels, the algorithm has a high operation load, causing delays especially for live streaming videos. In this paper we propose a parallel architecture to improve the performance and reduce the operation time of the Kriging Algorithm for estimating the missing pixels. The proposed method can be applied on Field Programmable Gate Arrays (FPGA) and considerable performance improvement have been achieved depending on the number of logic blocks available inside the FPGA

Key Words

"Kriging algorithm, parallel architectures, interpolation, image reconstruction, FPGA"

***Corresponding Author:** ozgur.tamer@deu.edu.tr*

1. Introduction

Interpolation algorithms are used to construct new data points or to estimate missing data points by weighting the values of discrete set of priorly known data points (Vaseghi, 2012). Interpolation is an important research topic in the fields where large data sets in the form of vectors or matrices are used, like Geostatistics, Geography Information System (GIS), Image Processing, Motor Control and many other fields (Vaseghi, 2012). There are many interpolation methods such as Linear Interpolation, Inverse Distance to a Power, Local Polynomial Interpolation, Trilinear Interpolation (Vaseghi, 2012) and Kriging (He et al., 2011).
The Kriging algorithm is considered as the optimal interpolation method for minimizing the possible interpolation errors (He et al., 2011) and it is generally preferred to estimate the geological characteristics of the area based on the measured data from the nearby locations (Johnston et al., 2003), (Bohling, 2005), (Li & Dong, 2011), (Bonaventura et al., 2005). In these studies, geostatistical data is predicted by evaluating the weighted sum of the surrounding observed values to derive a prediction for each point. Different methods for determining the weights are proposed in various works (Bohling, 2005).
Interpolation in image processing is used to reconstruct the pixels lost in the sampling process by smoothing the data samples with an interpolation function (Miklós, 2004). Even the construction of a raw digital image is based on interpolating of different color values acquired from of different colors which is called Bayer filtering (Bayer, 1976).
The quality of the resulting image depends on the preferred interpolation technique. There are two basic categories used in image interpolation; deterministic techniques and statistical techniques (Miklós, 2004). Deterministic techniques uses a fixed weighting scheme between data points while statistical techniques depend on approximating the weights based on the statistical properties of the data. Even though statistical methods promise a lower estimation error they increase the computation time (Miklós, 2004).
Kriging attempts to minimize the error variance by calculating every possible distance weighting function based on the statistical properties of the measurement values (He et al., 2011). These calculations increase the computational complexity and therefore the computational load of the algorithm. There exists various studies of using the Kriging algorithm about reconstruction of an image that have missing pixels. Panagiotopoulou and Anastassopoulos propose a method to interpolate a high resolution image from a low resolution image by using the Kriging technique (Panagiotopoulou & Anastassopoulos, 2007). They apply the Kriging algorithm to subpixel shifted and aliased low-resolution images to reconstruct the missing pixels of a high-resolution image. Their results show significant improvement when compared with alternative interpolation methods. Wielgosz et al. proposed and FPGA based architecture to reduce the computation time of the algorithm (Maciej Wielgosz Mauritz Panggabean, 2013). In their approach, the missing pixels are fixed and the weights are predetermined based on the statistical data of the previous images. Therefore it is neither a deterministic nor a statistical method, it is a hybrid approach. They also reduce time by performing several operations concurrently. Co-kriging grid-point interpolation is used by Chernetskiy, Tao, and Muller to generate Digital Elevation Model (DEM) and determine height uncertainties for each location (Chernetskiy et al., 2019). Their results show improvement in satellite images even for urban area satellite images. Li and Dong proposed an interpolation algorithm based on Kriging Interpolation and running on Graphics Processing Unit (GPU) cores (Li & Dong, 2011). They first use Kriging to interpolate the missing data values to build stratum surfaces and build a three-dimensional geological model. In their work, Kriging algorithm runs on a GPU with CUDA® (Compute Unified Device Architecture) cores. CUDA® is a graphics-processing unit, which can also be used as a general-purpose parallel computing architecture proposed by NVIDIA® Corporation. Their results show speedup values greater than 100 for some grid counts. The work presented by Yfantis and Au shows a Kriging based image compression technique (Hagstrom & Yfantis, 1994). They use Kriging interpolator to estimate the pixels not included in the image. Their results are promising when compared with other compression techniques. In the work presented by Han and friends, Kriging is used to pretreat of an image to be inpainted by the proposed improved Criminisi based inpainting algorithm (Han et al., 2021). They first degrade the image to a lower resolution, then use Kriging to improve the quality called "pretreatment". Their results are superior when compared with traditional Criminisi based inpainting algorithm. This work is similar to the work presented in this paper since intentional degrading is performed on the image. A salt and pepper noise reduction algorithm based on Kriging interpolation is presented by Varatharajan and friends called the Adaptive Decision based Kriging interpolation filter (ADKIF) (Varatharajan et al., 2018). The proposed method is a decision based one using a 3×3 kernel and requires at least 3 non noisy pixels in the current processing window, if this condition is not satisfied window size is increased. Presented results show comparable performance with similar methods for most images and superior in some of the images. An FPGA based semivariogram computation methodology is presented by Lagadapati, Shirvaikar and Dong (Lagadapati et al., 2015). Even though this work is not directly focused on Kriging, it presents a work based on FPGA implementation, which shows similarity with this paper. Their aim is to speed up the semivariogram value calculation by concurrent operating modules dedicated to the constituent computational tasks. Presented results show a significant improvement in speed for evaluating the semivariogram value using the proposed architecture.
The main aim of this work is to interpolate the missing image pixels which are dropped on purpose and in a predefined order for compression of images. A parallel architecture is proposed to reduce the processing load of the algorithm to avoid delays especially in video streams (Rønningen et al., 2011). Initial results of this work has been presented by Guvendik et. al. (Güvendik et al., 2012)
General definition of the Kriging algorithm will be introduced in the second section of this paper, which is going to be followed by the presentation of the proposed parallel architecture. At the fourth section, results of this application will be presented. Finally, the results will be concluded.

2. Methodology

Kriging is an optimized interpolation method based on regression against evaluated values from the surrounding observation points, weighted according to spatially varying values according to the covariance between these observation points [4].
It is usually used in Geostatistic calculations like elevation, material density of an area by taking the data from surrounding points and estimate a data value for unknown location [3]. The algorithm can also be used in image processing applications to estimate unknown pixel values [10], [12].
The aim of the Kriging algorithm is to find the summation of the surrounding residual values of data points and obtain the corresponding weights of the points. As presented in Fig 1. Z(u) is the point with unknown value, λ is the Kriging weight for each corresponding measurement value given by Z(1) to Z(6) The cumulative sum of these weighted values gives the estimated data.

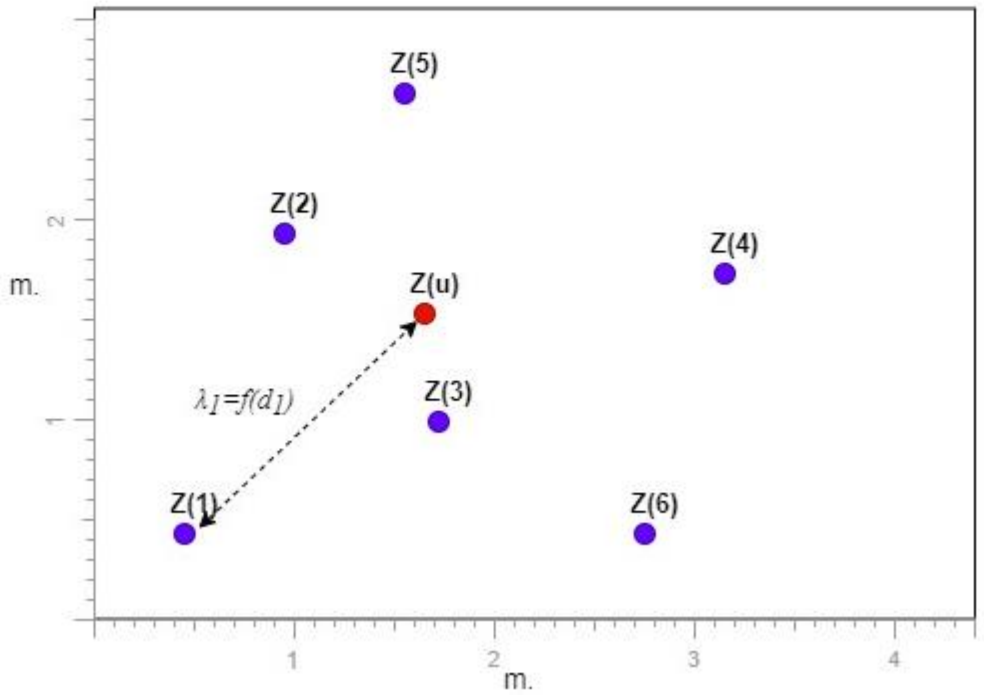


Figure 1. Geographic data points with their coordinates**.**

We define Z(u) as a random field with a mean value m(u). Therefore, the residual component of Z(u) is given by;

$$R(\mathbf{u}) = Z(\mathbf{u}) - m(\mathbf{u}) \tag{1}$$

Kriging algorithm estimates the value at point u by weighting the sum of the measured values at surrounding locations.
The general expression for the Kriging algorithm is given by;

$$Z^{*}(\mathbf{u}) - m(\mathbf{u}) = \sum \lambda_{\alpha}[Z(\mathbf{u}_{\alpha}) - m(\mathbf{u}_{\alpha})] \tag{2}$$

where **u**, **u**$_\alpha$ denote the position vectors for the location to be estimated and one of the data of a location of close proximity, indexed by α. In equation (2) n(**u**) denotes the number of locations with observed data in close proximity used for estimation of Z^*(**u**) and m(**u**), m(**u**$_\alpha$) are the expected values of Z(**u**$_\alpha$) and Z(**u**) while λ_α is the Kriging weight assigned to Z(**u**$_\alpha$) for estimating the value at location **u**.
The weights differ for different locations based on axial distances.
The goal is always to determine Kriging weights (λ) that minimizes the estimation error;

$$\sigma_E^2(\mathbf{u}) = \mathrm{Var}\,\{Z^{*}(\mathbf{u}) - Z(\mathbf{u})\} \tag{3}$$

The assumption for data points are given by

$$Z(\mathbf{u}) = R(\mathbf{u}) + m(\mathbf{u}) \tag{4}$$

$$\mathrm{E}\{R(\mathbf{u})\} = 0 \tag{5}$$

Where R(u) is a random field with stationary covariance and a mean of 0,
The unbiasedness condition for minimizing the estimation error variance is given by;

$$\begin{gathered} Z(\mathbf{u}) = R(\mathbf{u}) + m(\mathbf{u}) \\ \mathrm{E}\{R(\mathbf{u})\} = 0 \\ \mathrm{E}\{Z^*(\mathbf{u}) - Z(\mathbf{u})\} = 0 \end{gathered} \tag{6}$$

Under these conditions the covariance between all data points must be calculated as given in (8).

$$\mathrm{Cov}\,\{R(\mathbf{u})R(\mathbf{u}+\mathbf{h})\} = \mathrm{E}\{R(\mathbf{u}) \cdot R(\mathbf{u}+\mathbf{h})\} = C_R(\mathbf{h}) \tag{7}$$

2.1 Simple Kriging

Data point estimations for simple Kriging is given by;

$$Z_{SK}^*(\mathbf{u}) = m + \sum_{\alpha=1}^{m(\mathbf{u})} \lambda_{\alpha}^{SK}(\mathbf{u})[Z(\mathbf{u}_\alpha) - m] \tag{8}$$

This is similar with the general formula. The specialty of the Simple Kriging method is the constant mean “m” as seen from(10).

$$\begin{gathered} \mathrm{E}[Z_{SK}^*(\mathbf{u})] = m = \mathrm{E}[Z(\mathbf{u})] \\ \mathrm{E}[Z(\mathbf{u}_\alpha) - m] = 0 \end{gathered} \tag{9}$$

It is obvious that the error expression of residuals can be obtained as;

$$\begin{gathered} Z_{SK}^*(\mathbf{u}) - Z(\mathbf{u}) = \\ \sum_{\alpha=1}^{n(u)} \lambda_{\alpha}^{SK}(\mathbf{u})R(\mathbf{u}_\alpha) - R(\mathbf{u}) = \\ R_{SK}^*(\mathbf{u}) - R(\mathbf{u}) \end{gathered} \tag{10}$$

Error of the variance is evaluated by the linear combination of random variables as presented in (13).

$$\begin{aligned} \sigma_E^2(\mathbf{u}) &= Var\{R_{SK}^*(\mathbf{u})\} + Var\{R_{SK}(\mathbf{u})\} - 2Cov\{R_{SK}^*(\mathbf{u}), R_{SK}(\mathbf{u})\} \\ &= \sum_{\alpha=1}^{n(\mathbf{u})} \sum_{\beta=1}^{n(\mathbf{u})} \lambda_{\alpha}^{SK}(\mathbf{u})\,\lambda_{\beta}^{SK}(\mathbf{u})C_R(\mathbf{u}_\alpha - \mathbf{u}_\beta) + C_R(0) - 2\sum_{\alpha=1}^{n(\mathbf{u})} \lambda_{\alpha}^{SK}(\mathbf{u})C_R(\mathbf{u}_\alpha - \mathbf{u}) \end{aligned} \tag{11}$$

The derivative of the equation in (13) is taken with respect to the Kriging weights and each derivative is set to zero to minimize the error variance. As a result, we get the following system of equations (14);

$$\begin{gathered} \sum_{\beta=1}^{n(u)} \lambda_{\alpha}^{SK}(\mathbf{u})C_R(\mathbf{u}_\alpha - \mathbf{u}_\beta) = C_R(\mathbf{u}_\alpha - \mathbf{u}) \\ \alpha = 1, \ldots, n(\mathbf{u}) \end{gathered} \tag{12}$$

In vector notation these equations become;

$$\begin{gathered} \mathbf{K}\lambda_{SK}(\mathbf{u}) = \mathbf{k} \\ \lambda_{SK} = \mathbf{K}^{-1}\mathbf{k} \end{gathered} \tag{13}$$

Where **K** is covariance matrix between all pixels; **k** is the covariance vector between data pixels and estimation pixel; **λ** is the Kriging weight vector.

Kriging has three main types; Simple Kriging, Ordinary Kriging and Kriging with a trend. In this paper Simple Kriging is employed.

3. Parallel Kriging Architecture

Kriging interpolation has great performance on reconstructing a dataset from small amount of information as mentioned previously. However, computational load of the algorithm is very high due to the computation procedure of the weights. A parallel architecture is introduced in this section for improving the performance of the algorithm.

In case of randomly missing pixels we need to calculate Kriging weights for each estimation process. However, the proposed architecture assumes that, the pixels to be estimated are deleted on purpose for compression of streaming videos. Therefore, the locations of the missing pixels is predetermined by the compression algorithm as can be seen in Fig. 2. Each number in Fig. 2 (a) corresponds to a pixel of an image. As can be observed image is divided into 3x3 sub-images and all the pixels of the sub-image are numbered from 1 to 9. In order to compress the image, pixels marked with one or more numbers can be deleted before it is transmitted (Panggabean et al., 2011). In Fig. 2 (b) pixels numbered with 2 and 7 are deleted before transmission. Since the pixels are deleted have fixed positions, corresponding distance with the available pixels are also predetermined. Therefore, we can pre-calculate the corresponding Kriging weights according to the distances and store them.

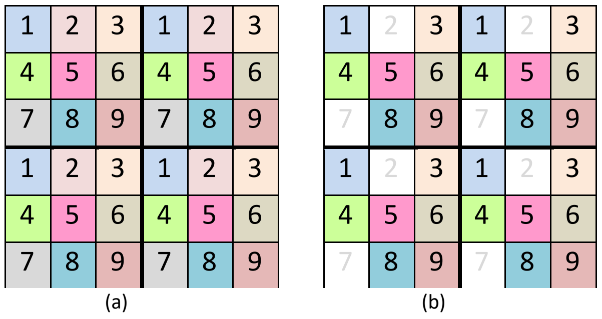


Figure 2. Image with pixels marked with numbers for planned deleting

The proposed architecture has distributed identical processing elements (PE) as shown in Fig. 3. Each PE is responsible for interpolating the unknown pixel by using the available pixels.
Each PE, as shown in Fig. 4, contains a look-up table (LUT) to keep the values of Kriging weights and the distance calculator which finds the distance between available pixel and the estimated pixel by using locations of them. The RAM block acquires and stores the available pixels' value until the end of each estimation process to prevent flow delays.
Each PE stores the values of the observation points pixels and calculates the corresponding pixel to be estimated by weighting it with appropriate LUT value. We evaluate the estimation values, by adding the outputs of each processing element as shown in Figure 3.

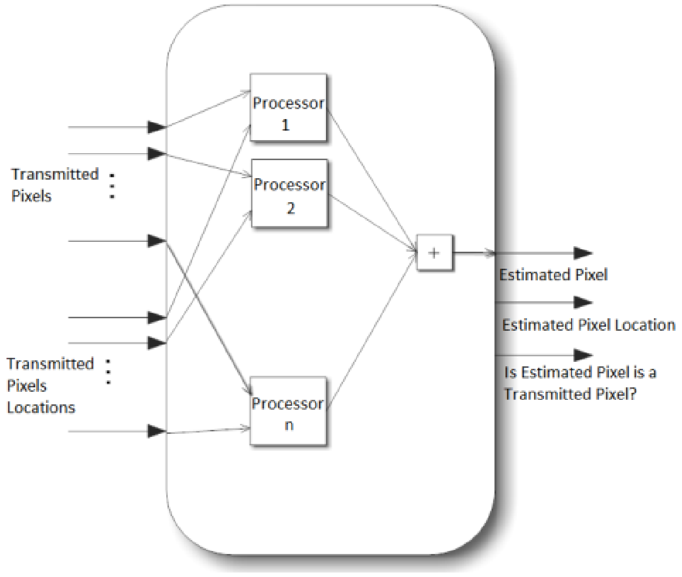


Figure 3. Block diagram of the parallel architecture

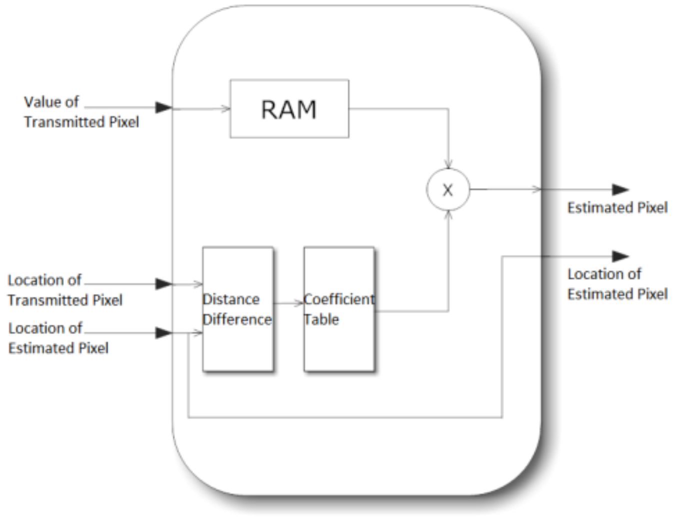
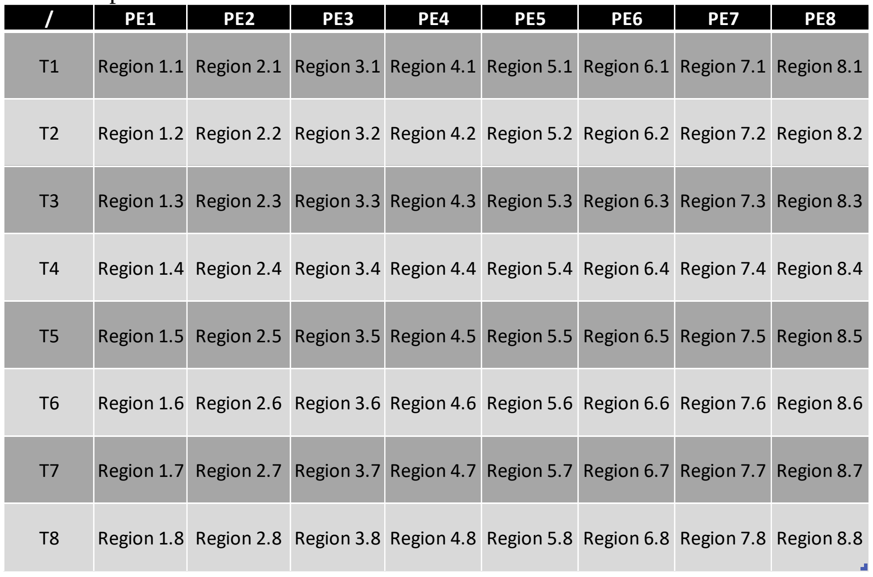

Value of Transmitted Pixel
RAM
X
Estimated Pixel
Location of Estimated Pixel
Location of Transmitted Pixel
Location of Estimated Pixel
Distance Difference
Coefficient Table

Figure 4. Block diagram of a single processing element

The image is divided into regions as presented in Figure 5. Each PE (denotes as PEX on the most top row) is responsible of processing different regions in its dedicated column. At each time slot each PE processes a region then switches to the next region in the same column therefore regions of each row is processed concurrently. The concurrent operation of the PE's reduces the processing time at a similar ratio since all of the operation is considered as parallel operation. Only reconstruction of the image after all of the pixels are evaluated is considered as the serial part which has minor effect on the total time.



/	PE1	PE2	PE3	PE4	PE5	PE6	PE7	PE8
T1	Region 1.1	Region 2.1	Region 3.1	Region 4.1	Region 5.1	Region 6.1	Region 7.1	Region 8.1
T2	Region 1.2	Region 2.2	Region 3.2	Region 4.2	Region 5.2	Region 6.2	Region 7.2	Region 8.2
T3	Region 1.3	Region 2.3	Region 3.3	Region 4.3	Region 5.3	Region 6.3	Region 7.3	Region 8.3
T4	Region 1.4	Region 2.4	Region 3.4	Region 4.4	Region 5.4	Region 6.4	Region 7.4	Region 8.4
T5	Region 1.5	Region 2.5	Region 3.5	Region 4.5	Region 5.5	Region 6.5	Region 7.5	Region 8.5
T6	Region 1.6	Region 2.6	Region 3.6	Region 4.6	Region 5.6	Region 6.6	Region 7.6	Region 8.6
T7	Region 1.7	Region 2.7	Region 3.7	Region 4.7	Region 5.7	Region 6.7	Region 7.7	Region 8.7
T8	Region 1.8	Region 2.8	Region 3.8	Region 4.8	Region 5.8	Region 6.8	Region 7.8	Region 8.8

Figure 5. Working regions of PEs and time slots

4. Results

First, simulation time of the algorithm by using one PE is measured. The algorithm, which uses necessary operations including row/column selection, subtraction, addition, multiplication, division, absolute value needs 3180 cycles for each pixel estimation [3]. If the resolution of the image is considered as 512x512, total cycles to get result will be 834,404,352. 250 MHz clock frequency which is the clock frequency of Xilinx Virtex 6 FPGA is preferred for calculation of necessary operation time. Xilinx Virtex 6 FPGA is preferred since the development board available for use in our experiments are equipped with these FPGA's. Correspondingly, the total time to get result for a single PE will be 3.337 seconds.
If the 512x512 resolution image is divided into N processing elements by using the parallel architecture for the Kriging algorithm, total time can be calculated as;

$$t = s/n \tag{23}$$

Here, t is the total operation time, s is the single PE operation time and n is parallel PE number. Consequently, total reconstruction time of the 512x512 resolution MALE image by using (14.) with 8 PEs will be 0.417 seconds. Operation times for different number of parallel PEs are presented in Table-2.

A 512x512 pixel MALE image, which is presented in Fig. 6 is used to test the parallelized Kriging algorithm. The parallel architecture is developed using Simulink HDL libraries and simulated using the same environment [7]. The developed architecture has 32 concurrent processing elements.

Table 2. Number of Parallel PEs and total calculation time

Number of Parallel PEs	Process Times (sec.)
1	3.337
2	1.668
3	1.112
4	0.834
8	0.417
16	0.208
32	0.104

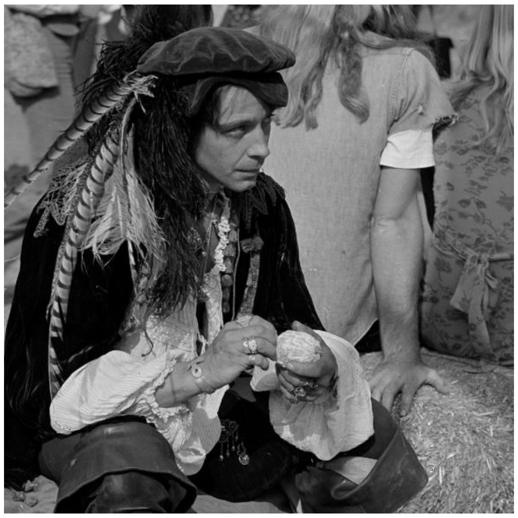


Figure 6. Uncompressed image

Figure 7 presents the reconstructed images. The first image is reconstructed from %25 of the original image while and the second image contains only %11.11 of it. Here, neighborhood of nearest 8 pixels is used to calculate the Kriging weights. Exponential semivariance model, which is shown in Figure 3, tells, if the distance (h) is increasing from an observation point towards the estimation points, the Kriging weights of this observation point for these estimation points are decreasing. Thus, using the neighborhood of nearest 8 pixels is considered as satisfactory for accurate reconstruction.

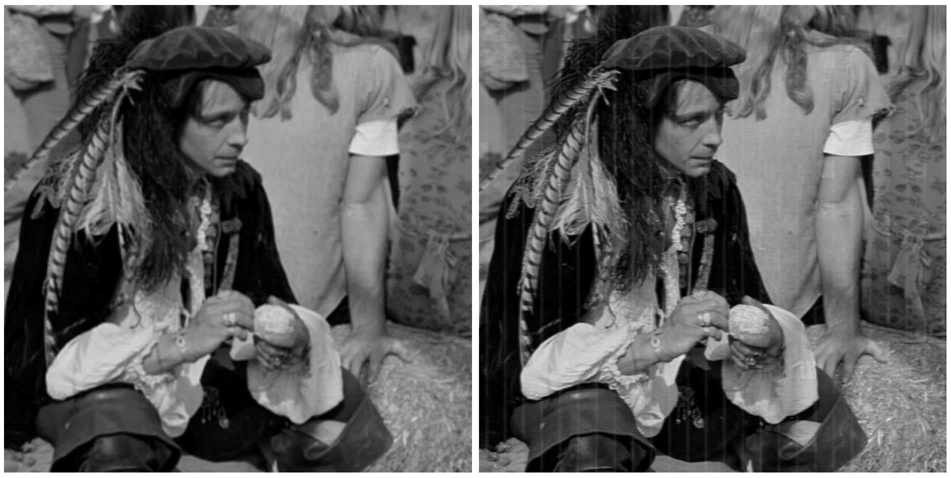


Figure 7. Reconstructed images from 11% (left) of uncompressed image and 25% (right) of uncompressed image

Increasing or decreasing the size of the image also effects processing time. Table 3 presents processing time in seconds for different image resolutions vs number of processing elements. As can be observed increasing the image resolution dramatically increases processing times and more PEs are needed for reconstructing the image in desired timing limits. Numbers in parenthesis next to time in every row denotes the number of PEs used in the simulation.

Table 3. Effect of the Resolution of the image

Resolution	Time (sec)	Time(8)	Time(16)	Time (32)	Time(64)	Time(128)	Time (256)
512x512	3,33447168	0,416809	0,2084045	0,1042022	0,0521011	0,026051	0,013025
1024x1024	13,3378867	1,667236	0,8336179	0,416809	0,2084045	0,104202	0,052101
Full HD	26,376192	3,297024	1,648512	0,824256	0,412128	0,206064	0,103032
4K	105,504768	13,1881	6,594048	3,297024	1,648512	0,824256	0,412128

In Table 4 comparison of the proposed work with results of other similar techniques is presente. In the work presented by Hudson et al. an implementation of an engine on the Xilinx XC6264 image interpolation is performed which utilizes 2-5-2 splines (Hudson et al., 1998). In the work presented by Fahmy bilinear interpolation hardware interpolating a stream of sub-pixel addresses in a single-cycle delay pipeline is propoed (Fahmy, 2008). A Hardware Architecture for Bicubic Interpolation (HABI) is proposed by Nuño-Maganda and friends (Nuño-Maganda & Arias-Estrada, 2005). The proposed architecture is integrated by three main blocks: for generating the interpolation coefficients. Finally a highly tunable motion estimation architecture is proposed in the work presented by Bournias and friends implementing the Horn and Schunck algorithm in FPGAs (Bournias et al., 2021). As can be observed the proposed algorithm needs more operating time with respect to the methods in the literature. However it is more flexible since controlled erasing of the pixels can be increased to save bandwidth.

Table 4. Comparison with other work

	FPGA Frequency (MHz)	Initial Image	Reconstructed Image	Duration (ms)
Hudson et.al. (Hudson et al., 1998)	30	128x128	512x512	4.91
Fahmy (Fahmy, 2008)	250	128x128	512x512	1
Nuño-Maganda (Nuño-Maganda & Arias-Estrada, 2005)	100	640 x 480	884x660	3.5
Bournias	800	256x256	1024 x 1024	0.5
This work	250	341x341	512x512	13 (256 PEs)

5. Conclusion

In this paper, we propose a parallel processing architecture to improve the performance of the Kriging algorithm. Kriging algorithm helps to recover the effects of data clustering by assigning each points within a cluster different amount of weight. The parallel architecture enables concurrent operation and therefore reduction in processing time without effecting the output image or video and therefore, improving the overall performance of the algorithm. Simulation results show that an FPGA with enough number of processing elements processing time can be reduced to 0.1 seconds, which is adequate for even videos with 60 frames per second. As a future work, it is planned to develop a video compression algorithm for real time streaming videos by intentionally erasing a considerable amount of pixels and reconstructing them at the destination by using the proposed approach.

References

Bayer, B. E. (1976). *Color Imaging Array* (Patent No. US3971065A).

Bohling, G. (2005). Kriging. *Kansas Geological Survey, October*, 1–20. https://doi.org/10.2104/ag050010

Bonaventura, L., Castruccio, S., Laboratorio, M. O. X., Matematica, D., & Milano, P. (2005). *Random notes on kriging : an introduction to geostatistical interpolation for environmental applications*.

Bournias, I., Chotin, R., & Lacassagne, L. (2021). FPGA acceleration of the hom and schunck hierarchical algorithm. *Proceedings - IEEE International Symposium on Circuits and Systems*, *2021-May*, 0–4. https://doi.org/10.1109/ISCAS51556.2021.9401068

Chernetskiy, M., Tao, Y., & Muller, J.-P. (2019). 3D stereo reconstruction: high resolution satellite video. *Https://Doi.Org/10.1117/12.2533226*, *11155*, 582–593. https://doi.org/10.1117/12.2533226

Fahmy, S. A. (2008). Generalised parallel bilinear interpolation architecture for vision systems. *Proceedings - 2008 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2008*, 331–336. https://doi.org/10.1109/ReConFig.2008.15

Güvendik, C., Esat Genç, A., Tamer, Ö., & Nil, M. (2012). Improving the performance of Kriging based interpolation application with parallel processors | Kriging temelli aradeğerleme uygulamasinda paralel işlemciler ile başariminin iyileş tirilmesi. *2012 20th Signal Processing and Communications Applications Conference, SIU 2012, Proceedings*. https://doi.org/10.1109/SIU.2012.6204645

Hagstrom, B. L., & Yfantis, E. A. (1994). Performance of Multi-Bus, Multi-Memory Systems Using Variable Miss Ratio. *Int. Conf. on Computing and Information*, 831–846.

Han, R., Liu, X., Liao, S., Li, Y., Qi, Z., Fu, S., Li, Y., & Han, H. (2021). Adaptive image inpainting algorithm based on sample block by kriging pretreatment and facet model. *Https://Doi.Org/10.1117/1.JEI.30.4.043021*, *30*(4), 043021. https://doi.org/10.1117/1.JEI.30.4.043021

He, F., Fang, J., & Zou, W. (2011). An effective method for interpolation. *Proceedings - 2011 19th International Conference on Geoinformatics, Geoinformatics 2011*. https://doi.org/10.1109/GeoInformatics.2011.5980762

Hudson, R. D., Lehn, D. I., & Athanas, P. M. (1998). A run-time reconfigurable engine for image interpolation. *Proceedings - IEEE Symposium on FPGAs for Custom Computing Machines, FCCM 1998*, *1998-April*, 88–95. https://doi.org/10.1109/FPGA.1998.707886

Johnston, K., Ver Hoef, J. M., Krivoruchko, K., & Lucas, N. (2003). The principles of geostatistical analysis. *Using ArcGIS Geostatistical Analyst*, 49–80.

Lagadapati, Y., Shirvaikar, M., & Dong, X. (2015). Fast semivariogram computation using FPGA architectures. *Https://Doi.Org/10.1117/12.2077851*, *9400*, 40–49. https://doi.org/10.1117/12.2077851

Li, M., & Dong, L. (2011). Visualization three-dimensional geological modeling using CUDA. *Proceedings - 6th International Conference on Image and Graphics, ICIG 2011*, 852–857. https://doi.org/10.1109/ICIG.2011.94

Maciej Wielgosz Mauritz Panggabean, L. A. R. (2013). FPGA Architecture for Kriging Image Interpolation. *International Journal of Advanced Computer Science and Applications(IJACSA)*, *4*(12), 193–201. http://ijacsa.thesai.org/

Miklós, P. (2004). Image interpolation techniques. *2nd Siberian-Hungarian Joint Symposium On Intelligent Systems. 2004.*, 1–6.

Nuño-Maganda, M. A., & Arias-Estrada, M. O. (2005). Real-time FPGA-based architecture for bicubic interpolation: An application for digital image scaling. *Proceedings - ReConFig 2005: 2005 International Conference on Reconfigurable Computing and FPGAs*, *2005*(1). https://doi.org/10.1109/RECONFIG.2005.34

Panagiotopoulou, A., & Anastassopoulos, V. (2007). Super-resolution image reconstruction employing Kriging interpolation technique. *2007 IWSSIP and EC-SIPMCS - Proc. 2007 14th Int. Workshop on Systems, Signals and Image Processing, and 6th EURASIP Conf. Focused on Speech and Image Processing, Multimedia Communications and Services*, 144–147. https://doi.org/10.1109/IWSSIP.2007.4381174

Panggabean, M., Tamer, O., & Rønningen, L. A. (2011). Parallel image transmission and compression using windowed kriging interpolation. *2010 IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2010*. https://doi.org/10.1109/ISSPIT.2010.5711801

Rønningen, L. A., Panggabean, M., & Tamer, O. (2011). Toward futuristic near-natural collaborations on Distributed Multimedia Plays architecture. *2010 IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2010*. https://doi.org/10.1109/ISSPIT.2010.5711738

Varatharajan, R., Vasanth, K., Gunasekaran, M., Priyan, M., & Gao, X. Z. (2018). An adaptive decision based kriging interpolation algorithm for the removal of high density salt and pepper noise in images. *Computers & Electrical Engineering*, *70*, 447–461. https://doi.org/10.1016/J.COMPELECENG.2017.05.035

Vaseghi, S. V. (2012). Interpolation. In *Advanced Digital Signal Processing and Noise Reduction* (Vol. 33, pp. 3–8). https://doi.org/10.1002/0470841621.ch10