FIELA: A Fast Image Encryption with Lorenz Attractor using Hybrid Computing

P Kranthi Kumar*, B V Nagendra Prasad*, Gelli MBSS Kumar, V. Chandrasekaran, P.K.Baruah
Sri Sathya Sai Institute of Higher Learning, Prasanthi Nilayam,India.
{kranthipls, nagendraprasad.bv, suneel.gmbs}@gmail.com,{vchandrasekaran,pkbaruah}@sssihl.edu.in

***Abstract*—In the past few years, the transmission of digital images across the world has increased. Images such as military images, personal photos are transmitted which are not intended for all. It is essential to secure these images from unauthorized access and modifications. Chaos theory is a scientific discipline that deals with non-linear dynamical systems that are effectively impossible to predict or control. Lorenz attractor is one such chaotic dissipative flow used for image encryption through the generation of confusion matrix. In the encryption method used, image is considered as a cube. To generate the confusion matrix, Lorenz attractor is applied on every coordinate of the cube [1]. We have proposed a fast encryption scheme called FIELA, in which novel improvements to the bin creation and sorting scheme have been implemented to harness the power of parallel computing resources like GPUs and multi-cores.**

***Index Terms*—Lorenz Attractor, Confusion Matrix, GPU, SpeedUp.**

I. Introduction

Digital images such as personal photos, military images which are not supposed to be exposed to others are transmitted across the world. Using encryption algorithms, these images need to be transmitted securely.

The special features of images like redundancy of data, strong correlation among adjacent pixels, less sensitive compared to text data makes it difficult to apply simple encryption algorithms. For real time image encryption, the algorithms which take lesser time without compromising on security are preferred. An encryption algorithm with higher degree of security features is of no practical use if it is slow.

In case of images, cryptography needs unique requirements like confusion, diffusion and dependence on secret keys. To meet these requirements, chaos theory can be used. Given the parameters, chaotic systems are seemingly random but they are predictable and reproducible with strong mathematical formulation. The required properties of cryptography are readily satisfied by chaotic systems. One such chaotic flow is Lorenz attractor which is an autonomous dissipative flow. Lorenz attractor [2] is governed by the equations

$$\frac{dx}{dt} = \sigma(y - x); \frac{dy}{dt} = rx - xz - y; \frac{dz}{dt} = xy - bz \quad (1)$$

where x, y, z, t are variables which take real values, and parameters σ, r, b are positive real constants. σ is called the Prandtl number, and b is called the Rayleigh Number. The dynamical orbit of Lorenz attractor will be chaotic for σ = 10.0 and b = 8/3 and r >24.74[2].

The Lorenz equations cannot be solved analytically by integration. Instead, a numerical approximation technique must be used. So, Runge-Kutta method of 4th order is employed. This Lorenz attractor is applied multiple times in the encryption process. It makes the whole process compute intensive. Fortunately GPUs which have many compute cores can be used for this process to make this computation faster.

The paper is organized as follows. Section II illustrates the encryption method used. Section III contains the parallelized version of the algorithm. Section IV presents the experimental setup and Section V contains the results and analysis of the algorithm.

II. Encryption Method Used

The overview of the algorithm used has been provided in the form a Figure 1. In this method, the given input 2D image is represented as a cube which is composed of bits arranged in three dimensional space [3]. At each pixel location, the Z-axis consists of 8 bit binary representation of the intensity value at that pixel location.

This way of looking at the image leads to the definition of a coordinate. A coordinate can be defined as a position of an intensity bit in the 3D structure of the image. So, for an image of size 512x512, the 3D view gives 512 x 512 x 8 coordinates. Encryption can be achieved by the

*student author

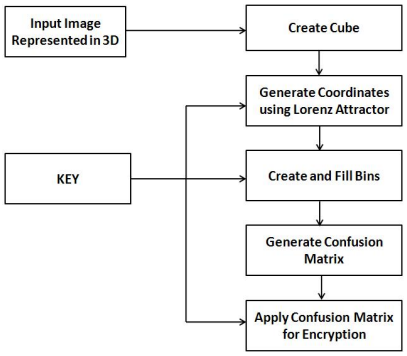


Fig. 1: Flow Chart of the Encryption Method

permutation of the bits and permutation in turn can be achieved using Lorenz attractor.

The ordinary differential equations of Lorenz attractor Equation 1 are solved using Runge-Kutta method of 4th order. This method poses computational problems for higher coordinate values. So, these coordinates were normalized for computational purposes. Then Lorenz attractor is applied on each of the normalized coordinates. The secret key determines the number of iterations the Lorenz attractor should be applied on each coordinate. These coordinates after the application of Lorenz attractor are called chaotic coordinates $(\text{LNormC})_i$.

A bin is defined by its minimum and maximum in (x,y,z) coordinate range. According to the $(\text{LNormC})_i$ coordinates, the coordinates of the cube are pushed into bins. These bins were later sorted lexicographically. Eventually, the coordinates of the cube were also sorted accordingly. These coordinates form the confusion matrix. Using this confusion matrix, the values at the coordinates of the original image were permuted at the bit level. The confusion matrix is applied on the image multiple times based on the key.

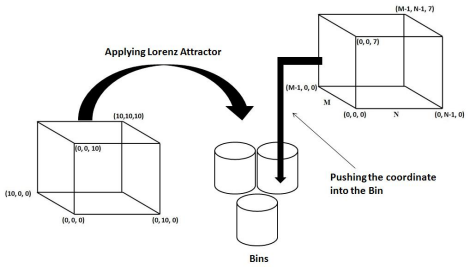


Fig. 2: Filling the Bins

So, the confusion matrix can be generated independent of the intensity values of the image. The only parameters required for the generation of confusion matrix are image size and the secret key. For the decryption process, the confusion matrix can be generated and applied on the encrypted image. This gives back the original image.

FIELA A New approach in the Application of Lorenz Attractor: The sequential code of the original algorithm[1] was profiled using GPROF. We observed that the Create and Fill bins step was consuming 54.89 % of the total time. This part of the algorithm is not suitable for parallel environment because bins should be filled sequentially. It might happen that two coordinates A,B enter the same bin. The order in which A,B enter the bin decides the final ordering of coordinates in confusion matrix. If this is done in parallel, there might be a possibility that A,B enter the same bin in a different order each time. In which case, the confusion matrix generated by parallel algorithm need not be same always. So, if the same confusion matrix is not generated, the decrypted image will not be the same as original image.

So, we modified the way bins are created so that each bin contains at most one coordinate without compromising on the accuracy of the original algorithm [1]. This modification not only makes the application parallelizable but also reduces the time taken on a single core. This new modified approach is termed as FIELA: A fast image encryption using Lorenz Attractor. The Create and Fill bins in the FIELA takes negligible amount of time.

III. Efficient Parallel Algorithm for FIELA

The computation of Lorenz attractor on the coordinates of the FIELA code took 76% of the total time. We observed from the code that Lorenz attractor is applied on each coordinate independently. Since there is no dependency, the Lorenz attractor can be applied in parallel on each coordinate. The next part of the algorithm which takes more time is the encryption process. It takes 9% of the total time. This encryption is an iterative process in which each iteration depends on the previous one. So, GPUs are not suitable for this part. Within each iteration, the operations are independent. So, loop level parallelism can be exploited using multi cores. Hence, the algorithm is divided into two parts. The first part is the application of the Lorenz attractor on coordinates which can be done on GPUs and the second is the encryption process on multi cores. This way the power of hybrid computing is utilized.

The observation mentioned in the above paragraph led to the implementation of highly compute intensive Lorenz attractor on GPUs. The work among the GPU threads is divided as follows.

- For each thread, only one coordinate is assigned. So, the number of threads created depends on the image size.
- Each thread normalizes the coordinate and applies Lorenz attractor on it. At the end of the computation of the thread, chaotic coordinates are obtained.

Creation of bins can be done in parallel. Each thread creates a bin and assigns the coordinate belonging to that thread to the bin. This bin is mapped to the integer value of the chaotic coordinates. Once all the bins are created, the next step is to sort the bins. In order to sort the bins, CUDA THRUST library is used. THRUST provides a high-level interface for GPU programming with functionalities like sort, scan, transform, and reduction operations [4]. The sorted bins gives the confusion matrix which is used in the encryption process.

IV. Experimental Setup

The experimental platform is the NCSA Forge supercomputer at University of Illinois. It contains NVIDIA Fermi M2070 Accelerator Units. There are 448 CUDA Cores with 1.03 teraflops single-precision performance and 515 gigaflops double-precision performance.

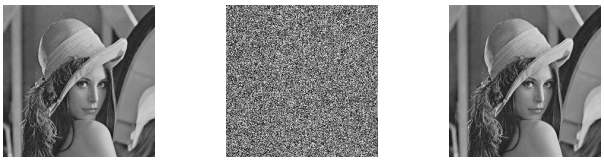
Operations on the image were performed on OpenCV platform. The C++ code was compiled using gcc compiler with O3 option which performs optimizations like loop unrolling, SIMD, dead code elimination etc. GPROF was used for profiling the sequential code. The CUDA code was compiled using nvcc compiler with compute capability 2.0. To know the number of registers used by each thread, an option ptxas-options=-v was used. To utilize the debugging mode in GPU, the CUDA code was compiled using options -G -g. OpenMP was used for the implementation on multi cores. The OpenMP code was executed on 16 cores. RDTSC timers were used for timing all the non-GPU codes. CUDA events are used for timing the CUDA code. Warming up time for the GPU was not considered.

V. Results and Analysis

A. Quality Test

The 512x512 Lena image was chosen for the quality test. The results of modified algorithm are shown in Figure 3.

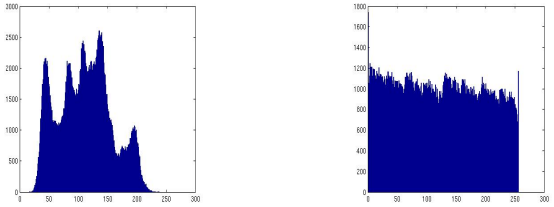
The quality of results are discussed below. In the original algorithm statistical analysis was used to analyze the security of the encrypted image. The different types of analysis that were looked into are 1) Histogram Analysis 2) Information Entropy 3) Correlation of adjacent pixels 4) Key space Analysis. So, the statistical analysis was performed to the modified algorithm as it was done for original algorithm. The results of FIELA are similar with the original algorithm. This tells us that our FIELA does not compromise on the accuracy of results.



Original Lena Encrypted Lena Decrypted Lena

Fig. 3: Encryption and Decryption using the FIELA (Key used is 01234567890123456789)

1) Histogram Analysis: This test is used to illustrate superior confusion property of encrypted image. The histograms of the original 512x512 gray scale Lena Image and the encrypted Lena Image are shown in the Figure 4. We can see that the histogram of the original image is very different from the histogram of the encrypted image. In addition to that the encrypted image's histogram is fairly uniform when compared to the histogram of original image. Hence the encrypted image does not reveal any kind of information to attacker during the transmission of the image.



Original Image Encrypted Image

Fig. 4: Histograms of Original and Encrypted Lena Images

2) Information Entropy: Information entropy for a 512x512 image of gray scale Lena image was calculated. The entropy of the encrypted image using the original algorithm is 7.9930. The entropy obtained for encrypted image using FIELA is 7.9934 which is close to theoretical value 8. This means that encryption process is secure against the entropy attack.

3) Correlation: Correlation of the adjacent pixels in the encrypted image is calculated to see how they are related. A total number of 4096 adjacent pairs of pixels in three directions: horizontal, vertical, diagonal are selected randomly. The correlation coefficient of adjacent pixels for both original image and the encrypted image

using FIELA are listed in the Table I. The correlation coefficient for the original image is close to 1, so they are highly correlated. For the encrypted image, adjacent pixels are highly uncorrelated as their correlation coefficient is close to 0.

TABLE I: Correlation coefficients of two adjacent pixels in original and encrypted image

Direction	Original Image	Encrypted Image
Horizontal	0.9681	0.0219
Vertical	0.9821	0.0230
Diagonal	0.9819	0.0208

4) Key Space Analysis: This test shows us what happens when the encrypted image is decrypted with a different key. The below Figure 5 consists of Lena Image encrypted with key 01234567890123456789 and the image decrypted using a different key that is 01234567890123456788. We can see that even for a slight change in the key the decrypted image is totally different from the original image.

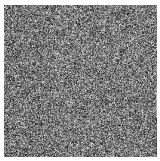


Image Encrypted with key 01234567890123456789

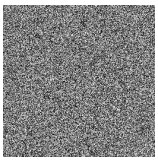


Image Decrypted with key 01234567890123456788

Fig. 5: Key Sensitivity

B. Experimental Analysis

The original and FIELA algorithms were tested on gray scale images of size 100x100, 256x256, 512x512. Table II contains the timings of the application of Lorenz attractor and encryption for both original algorithm and FIELA. The application of Lorenz attractor is same for both original and FIELA algorithms. Only the encryption process differs for both of them. We observe that as the image size increases, the speed up of FIELA over the original algorithm increases drastically.

The timings of Multi core and GPU implementations of FIELA algorithm are compared with sequential FIELA in the further results. Table III contains the execution timings on multi-core platform and the speedup over singe-core.

TABLE II: Single core timings (in seconds)

Image size	Appln of Lorenz attr	Encryption	
		Original	FIELA
100x100	20.47	9.36	0.43
256x256	136.27	396.81	4.90
512x512	531.56	6686.92	20.56

TABLE III: Multi core timings and Speed Up over single core

Image size	Appln of Lorenz Attr		Encryption	
	Time (in s)	Speed Up	Time (in s)	Speed Up
100x100	17.063	1.19	0.15	2.81
256x256	114.45	1.19	2.54	1.93
512x512	467.25	1.13	13.54	1.51

The results of the GPU implementation of application of Lorenz attractor are discussed below. Using the compiler option –ptxas-options=-v for the GPU code provided the information about the total number of registers used per thread as 48. Feeding this value in the CUDA occupancy calculator [5] gives us the best block size with maximum thread occupancy. The best block size for this algorithm is 320 with 42% thread occupancy. With this block size, the number of active thread blocks per multi processor are 2. The graph showing the active warps for the selected block size is given in Fig 6. So, the total number of threads active per multi processor are 640. The number of such multi processors in a GPU are 16. So, the expected speedup is 16x640 = 8960x, however the speed up achieved for a 512x512 image is 1508x over the sequential FIELA. The timings of GPU and multi-core versions of algorithm were compared. Table IV contains the execution timing on GPU and the speedup over the multi core platform.

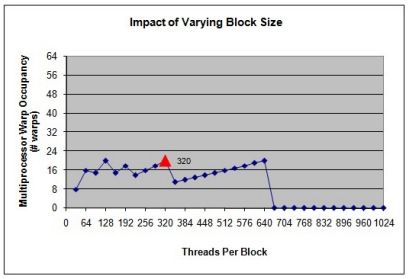


Fig. 6: Graph from Occupancy Calculator

In order to improve the thread occupancy we tried to minimize the number of registers used by each thread.

TABLE IV: GPU timings (in seconds) and Speed Up

Image size	Appln of Lorenz Attr	Speed Up over	
		multi core	single core
100x100	0.0155	1100.83	1317.4
256x256	0.0917	1248.06	1484.62
512x512	0.3525	1325.54	1507.96

This was done by reusing the previously allocated registers. Using this optimization, the number of registers used per thread are decreased to 39. The best block size for this number of registers is 384. The number of blocks that are active per multi processor are 2. So, the total number of threads active per multiprocessor are 786. By doing this, the thread occupancy increased to 50 percent. The improved GPU results with block size 384 are in Table V. The graph which illustrating the speed up of multi core and GPU over the CPU is shown in the figure 8. In the graph, the results were plotted on a logarithmic scale.

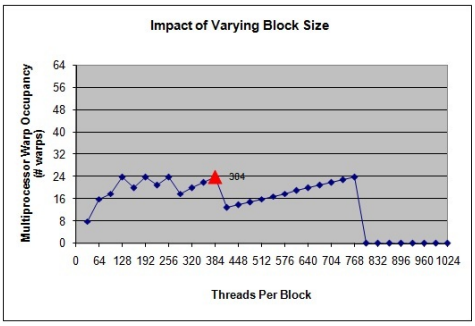


Fig. 7: Graph from Occupancy Calculator

TABLE V: GPU timings for Optimized code (in s)

Image size	Appln of Lorenz Attr	Speed Up over multi core
100x100	0.0138	1236.47
256x256	0.0850	1346.44
512x512	0.3346	1396.45

VI. Conclusion and Future Work

In this paper, we have presented a novel encryption scheme called FIELA, which is an improved version of original algorithm proposed in [1] to exploit parallelism without compromising on security aspects. Statistical Analysis was performed on image encryptions via serial

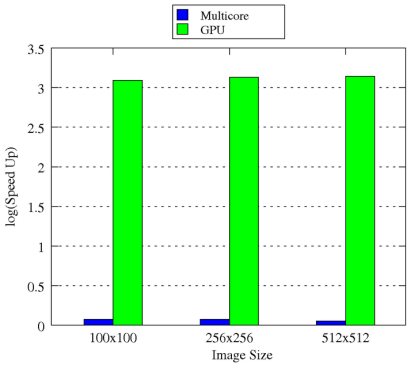


Fig. 8: Speed Up Over single core

mode FIELA. We observed that our serial mode algorithm itself outperformed the original serial algorithm with speed up of 325x for a 512x512 image. This algorithm can therefore be used in real time image encryption applications where speed is the key factor. Experimental results of the parallelized FIELA using GPUs show speed up over the serial FIELA algorithm by 1396x for a 512x512 image.

In this work, the parallelization was done using only one single GPU. This can be done on multiple GPUs where the computation can be divided across all the GPUs for the creation of confusion matrix which potentially improves performance. CUDA compute capability 3.0 can be used which increases the thread occupancy to 75% due to the increase in the register file size.

Acknowledgment

We dedicate this work to Bhagawan Sri Sathya Sai Baba, Founder Chancellor of Sri Sathya Sai Institute of Higher Learning. This work was partially supported by a nVIDIA, Pune grant under Professor Partnership Program and the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number OCI-1053575.

References

[1] Gelli MBSS Kumar & V. Chandrasekharan. A novel image encryption scheme using lorenz attractor. *4th IEEE Conference on Industrial Electronics and Applications Xi an, China*, 2009.

[2] J.C.Sprott. Chaos and time-series analysis. *OXFORD University Press*, 2003.

[3] Sai Charan Koduru and V Chandrasekharan. Integrated confusion-diffusion mechanisms for chaos based image encryption. *IEEE 8th International Conference on Computer and Information Technology Workshops.*, 2008.

[4] Jared Hoberock and Nathan Bell. Thrust: A parallel template library. 2010.

[5] http://developer.download.nvidia.com/compute/devzone/docs/html/c/tools/cuda_occupancy_calculator.xls.