Indonesian Journal of Electronics and Instrumentation Systems (IJEIS)
Vol.10, No.1, April 2020, pp. 87~98
ISSN (print): 2088-3714, ISSN (online): 2460-7681
DOI: 10.22146/ijeis.36517

Pengembangan True Random Number Generator berbasis Citra menggunakan Algoritme Kaotis

Dian Arief Risdianto*[1], Bambang Nurcahyo Prastowo[2]
[1]Prodi Elektronika dan Instrumentasi, DIKE, FMIPA UGM, Yogyakarta, Indonesia
[2]Departemen Ilmu Komputer dan Elektronika, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]dian.arief.r@mail.ugm.ac.id**, [2]prastowo@ugm.ac.id

Abstrak

Keamanan sebagian besar sistem kriptografi bergantung pada key generation yang tidak dapat diprediksi menggunakan Random Number Generator (RNG) yang nondeterministik. PRNG menghasilkan deret random number dengan pola berulang dalam jangka waktu tertentu dan dapat diprediksi jika kondisi awal dan algoritme diketahui. TRNG mengekstrak entropi dari sumber fisik yang tidak dapat diprediksi untuk menghasilkan bilangan acak yang nondeterministik. Namun, sebagian besar sistem tersebut memiliki biaya, kompleksitas, dan tingkat kesulitan yang relatif tinggi. Jika kamera diarahkan pada adegan yang acak, deret random number yang dihasilkan dapat diasumsikan acak pula. Namun, kerugian kamera digital sebagai sumber angka acak terletak pada pola bias yang dihasilkan. Deret raw tanpa pemrosesan lebih lanjut dapat memiliki pola noise yang tetap. Dengan menerapkan pengolahan citra digital dan juga algoritme kaotis, kamera digital dapat digunakan untuk menghasilkan true random number. Dalam penelitian ini, untuk preprocessing data citra digunakan metode algoritme floyd-steinberg. Untuk menyelesaikan masalah terdapatnya beberapa piksel hitam atau putih berurutan yang muncul di area citra yang diproses, digunakan algoritme arnold-cat map sedangkan operasi XOR digunakan untuk mengkombinasi data dan membangkitkan true random number. Pengujian statistic NIST, analisis scatter dan histrogram menunjukan penggunaan metode ini mampu menghasilkan true random number yang benar-benar acak..

Kata kunci*—TRNG, PRNG, Arnold's Map, Floyd-Steinberg*

Abstract

The security of most cryptographic systems depends on key generation using a nondeterministic RNG. PRNG generates a random numbers with repeatable patterns over a period of time and can be predicted if the initial conditions and algorithms are known. TRNG extracts entropy from physical sources to generate random numbers. However, most of these systems have relatively high cost, complexity, and difficulty levels. If the camera is directed to a random scene, the resulting random number can be assumed to be random. However, the weakness of a digital camera as a source of random numbers lies in the resulting refractive pattern. The raw data without further processing can have a fixed noise pattern. By applying digital image processing and chaotic algorithms, digital cameras can be used to generate true random numbers. In this research, for preprocessing image data used method of floyd-steinberg algorithm. To solve the problem of several consecutive black or white pixels appearing in the processed image area, the arnold-cat map algorithm is used while the XOR operation is used to combine the data and generate the true random number. NIST statistical tests, scatter and histrogram analyzes show the use of this method can produce truly random numbers.

Keywords— *TRNG, PRNG, Arnold's Map, Floyd-Steinberg*

Received June 26th,2018; Revised April 29th, 2020; Accepted April 30th, 2020

1. PENDAHULUAN

Perluasan penggunaan komunikasi digital melalui jaringan komputer, internet, dan perangkat nirkabel telah menghasilkan kebutuhan yang lebih besar untuk perlindungan transmisi informasi menggunakan kriptografi. Kriptografi memungkinkan pertukaran pesan asli antara pengirim dan penerima dengan menggunakan kunci kriptografi yang telah dihasilkan dan didistribusikan secara hati-hati untuk enkripsi dan dekripsi. Keamanan sebagian besar sistem kriptografi bergantung pada *key generation* yang tidak dapat diprediksi menggunakan *Random Number Generator* (RNG) yang nondeterministik. Oleh karena itu banyak metode telah diusulkan untuk menghasilkan keacakan seperti *pseudo random generator* (PRNG), *hardware-based random number generator* dan *true random number generator* (TRNG). PRNG menghasilkan deret *random number* berdasarkan algoritme deterministik menggunakan komputer. Deret yang dihasilkan memiliki pola berulang dalam jangka waktu tertentu dan dapat diprediksi jika kondisi awal dan algoritme diketahui.

TRNG mengekstrak entropi dari sumber fisik yang tidak dapat diprediksi untuk menghasilkan bilangan acak yang nondeterministik. Namun, dibutuhkan usaha lebih untuk mendapatkan bilangan random yang non-deterministik dan berasal dari aktivitas fisik di luar komputer. Terdapat berbagai macam metode dengan sumber keacakan berbeda diantaranya adalah RNG berbasis kuantum [1], polarisasi pada *fiber optic* [2], dan nanomagnet [3] untuk menghasilkan bilangan acak. Namun, sebagian besar sistem tersebut memiliki biaya, kompleksitas, dan tingkat kesulitan yang relatif tinggi.

Ada banyak metode pembangkitan atau TRNG, penelitian tentang topik ini masih terus dilakukan. Penelitian *random number generator* juga dilakukan menggunakan kamera [4], peralatan yang mudah ditemui, untuk memotret fenomena makroskopik sebagai sumber keacakan. Jika kamera diarahkan pada adegan yang acak, deret random number yang dihasilkan dapat diasumsikan acak pula. Namun, kerugian kamera digital sebagai sumber angka acak terletak pada pola bias yang dihasilkan. Deret *raw* tanpa pemrosesan lebih lanjut dapat memiliki pola *noise* yang tetap. Untuk dapat memenuhi unsur ketidakpastian, fenomena tersebut harus dapat dijustifikasi oleh sistem dinamika yang tidak stabil dan teori khaos (*chaos*). Khaos didefinisikan sebagai perilaku non-prediktif dari sistem dinamis nonlinear, yang sangat sensitif terhadap kondisi awal [5]–[8]. Dengan menerapkan pengolahan citra digital dan juga algoritme kaotis, kamera digital dapat digunakan untuk menghasilkan *true random number*. Dalam penelitian ini, untuk *preprocessing* data citra digunakan metode algoritme *floyd-steinberg* [9], [10]. Untuk menyelesaikan masalah terdapatnya beberapa piksel hitam atau putih berurutan yang muncul di area citra yang diproses, digunakan algoritme *arnold-cat map*[11]–[13]., sedangkan operasi *XOR* digunakan untuk mengkombinasi data dan membangkitkan *true random number*.

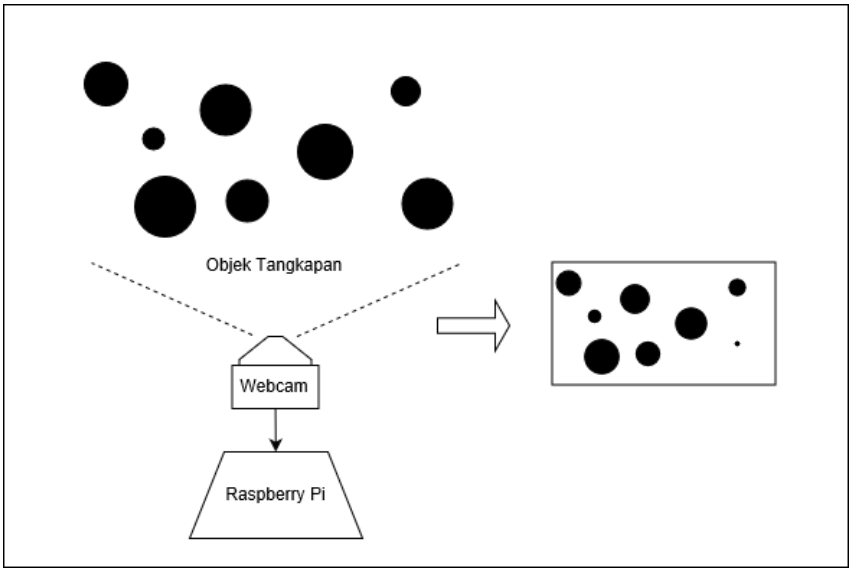
2. METODE PENELITIAN

2.1 Rancangan Sistem

Sistem *True Random Number Generator* yang dibangun pada penelitian ini menggunakan masukan citra tangkapan webcam sebagai sumber keacakan. Objek tangkapan citra yang digunakan dalam penelitian ini adalah adegan acak, atau dengan kata lain, objek sembarang selama objek tersebut memiliki pencahayaan yang cukup atau dapat ditangkap oleh kamera. Dalam penelitian ini juga tidak dilakukan perlakuan khusus/pengkondisian terhadap objek. Gambar 1 menunjukkan konfigurasi pengambilan data. Objek dipotret secara kontinyu oleh webcam. Selain itu, citra yang digunakan adalah berukuran 480×480. Konfigurasi setelan ukuran ini mempertimbangkan pemrosesan setelah ini. Jika citra yang ditangkap lebih besar dari

480×480, citra akan dilakukan *cropping* untuk digunakan bagian tengahnya saja (ukuran 480×480). Data citra tersebut kemudian diproses oleh Raspberry Pi dengan menerapkan algoritme kaotis. Algoritme kaotis yang diterapkan pada citra tersebut adalah *arnold-cat map*.

Secara garis besar, diagram blok sistem bisa dilihat pada Gambar 2. Citra tangkapan webcam akan dilakukan *pre-processing* terlebih dahulu. Setelah itu dilakukan penerapan algoritme kaotis dengan *arnold cat map*. Dilanjutkan dengan pembangkitan untuk menghasilkan *true random number*.

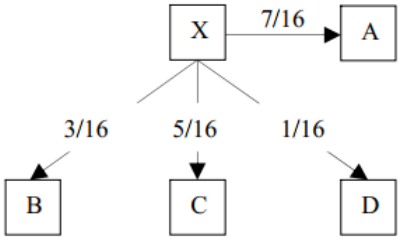


Gambar 1 Konfigurasi Pengambilan Data

Gambar 2 Diagram blok sistem secara umum

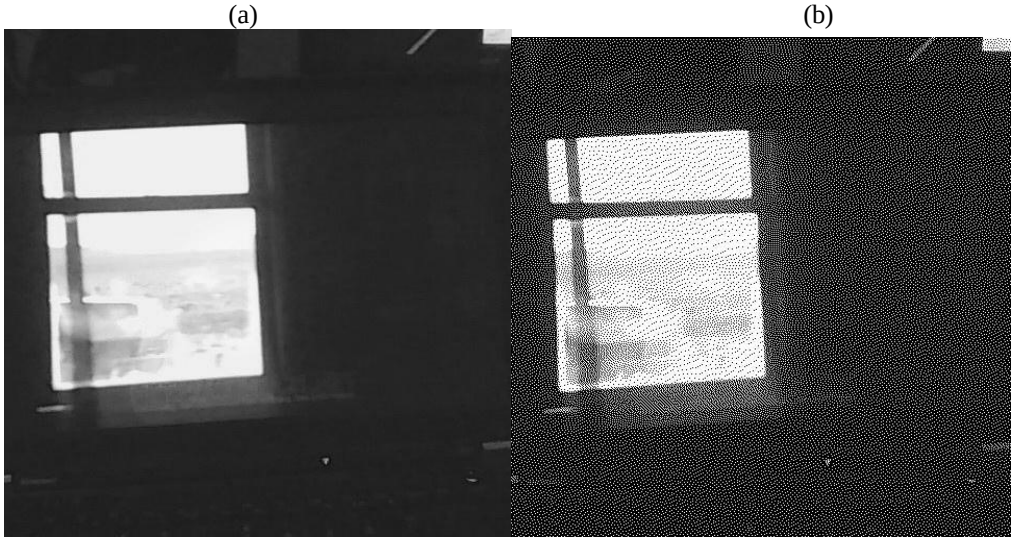
2.2 Tahap Pre-processing

Citra tangkapan webcam akan masuk ke tahap *pre-processing*, pada tahap ini ini citra digital akan diubah menjadi citra biner yang dapat dijadikan sebagai sumber keacakan dari TRNG. Transformasi biner dilakukan dengan menggunakan metode *error diffusion dithering*. Secara umum, *dither* merupakan metode untuk menambahkan *noise* buatan. Alasan menggunakan *diffusion error dithering* ini adalah, metode ini, dibandingkan dengan cara *threshold*, dapat menyimpan perbedaan citra yang memiliki kemiripan. Ini dapat secara efektif memastikan bahwa angka-angka yang dihasilkan dari citra biner ini adalah benar-benar acak. Dalam hal ini digunakan algoritme *floyd-steinberg* yang menggunakan distribusi kuantisasi error yang tidak merata ke piksel di sekitarnya. Gambar 3 menunjukkan proses dari algoritme Floyd-Steinberg. Error tersebar ke piksel kanan dan bawah. Dimana X mewakili piksel saat ini, dan A, B, C dan D mewakili piksel tetangga yang menerima kesalahan 7/16, 3/16, 5/16 dan 1/16 [14]. Dalam penelitian ini, citra dipindai dari kiri ke kanan dari urutan atas ke bawah. Citra *grayscale* asli dan hasil citra dithering menggunakan *floyd-steinberg* ditunjukkan pada Gambar 4.



Gambar 3 *Floyd-Steinberg Error-Diffusion* [14]

(a) (b)



Gambar 4 Jalannya transformasi biner: (a) citra grayscale ; (b) citra biner yang dihasilkan oleh *dithering*

2.2 Algoritme Arnold Cat-Map

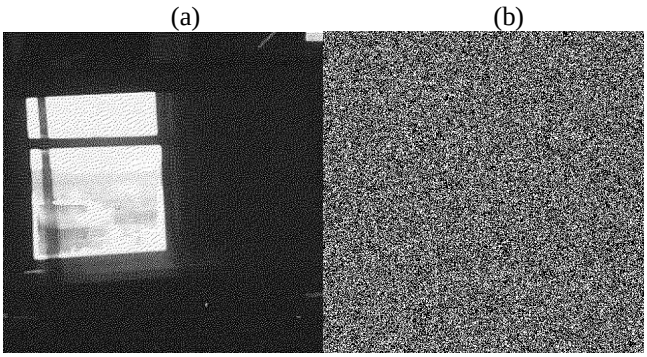
Sangat mungkin ada sejumlah kesamaan di antara citra yang berbeda yang berasal dari adegan yang sama. Oleh karena itu, setelah transformasi biner, citra harus dilakukan pemrosesan lagi dengan mengacaknya. Jika tidak, *random number* yang dihasilkan mungkin juga memiliki tingkat kesamaan tertentu. Selain itu, citra biner yang dihasilkan, secara umum, memiliki piksel hitam berurutan atau piksel putih di beberapa area, yang dapat menurunkan keacakan *random number* yang dihasilkan [15]. Untuk mengatasi masalah ini, digunakan metode berbasis kaotis. Metode ini dapat membuat output sangat sensitif terhadap input. Algoritme kaotis yang digunakan adalah *arnold-cat map*.

Arnold-cat map, juga dikenal sebagai transformasi *Arnold* diusulkan oleh V.I. Arnold dalam penelitian teori ergodik. Transformasi Arnold Cat Map adalah fungsi chaos 2D yang mentransformasikan koordinat (x, y) di dalam citra ke koordinat baru di dalam citra yang sama. Hal ini dapat digunakan untuk mengacak citra digital berukuran N × N seperti dijelaskan pada persamaan (1).

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix} \bmod N \quad (1)$$

dimana $(x_n, y_n) \in [0, N-1] \times [0, N-1]$

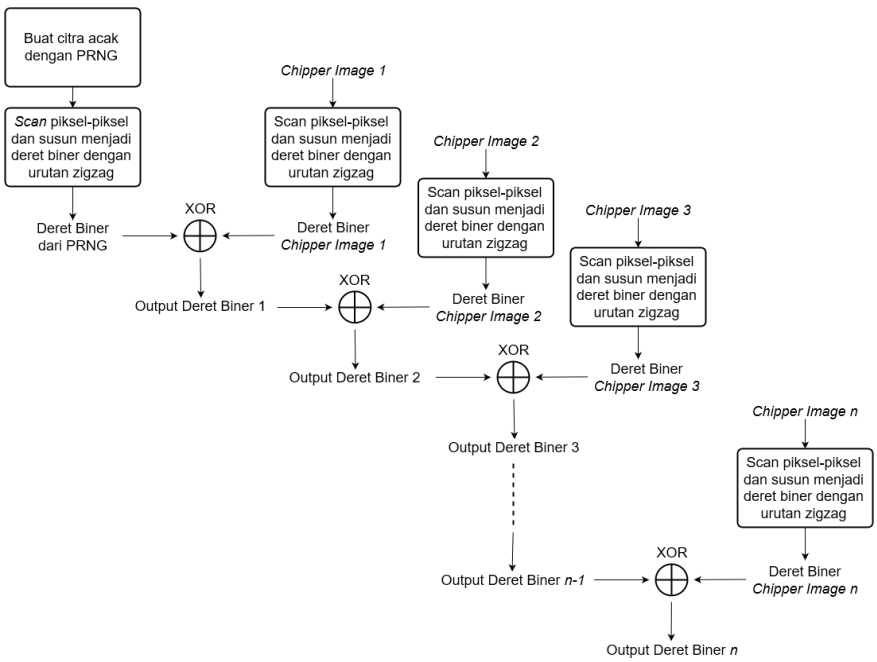
adalah koordinat piksel citra asli; N adalah tinggi atau lebar citra yang diproses; (x_{n+1}, y_{n+1}) adalah koordinat citra yang diacak. Transformasi mengubah posisi piksel, dan jika itu dilakukan beberapa kali, gambar yang tidak teratur dapat dihasilkan[16]. Pada penelitian ini, iterasi dilakukan sebanyak tujuh kali dan gambar dapat diacak secara efisien. Hasil dari penerapan algoritme ini ditunjukkan pada Gambar 5.



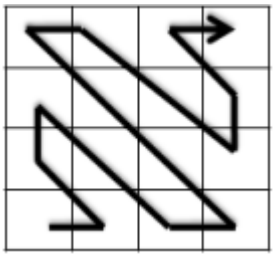
Gambar 5 (a) citra biner pada Gambar 3; (b) chipper image setelah penerapan *Arnold cat map*

2.2 Pembangkitan True Random Number

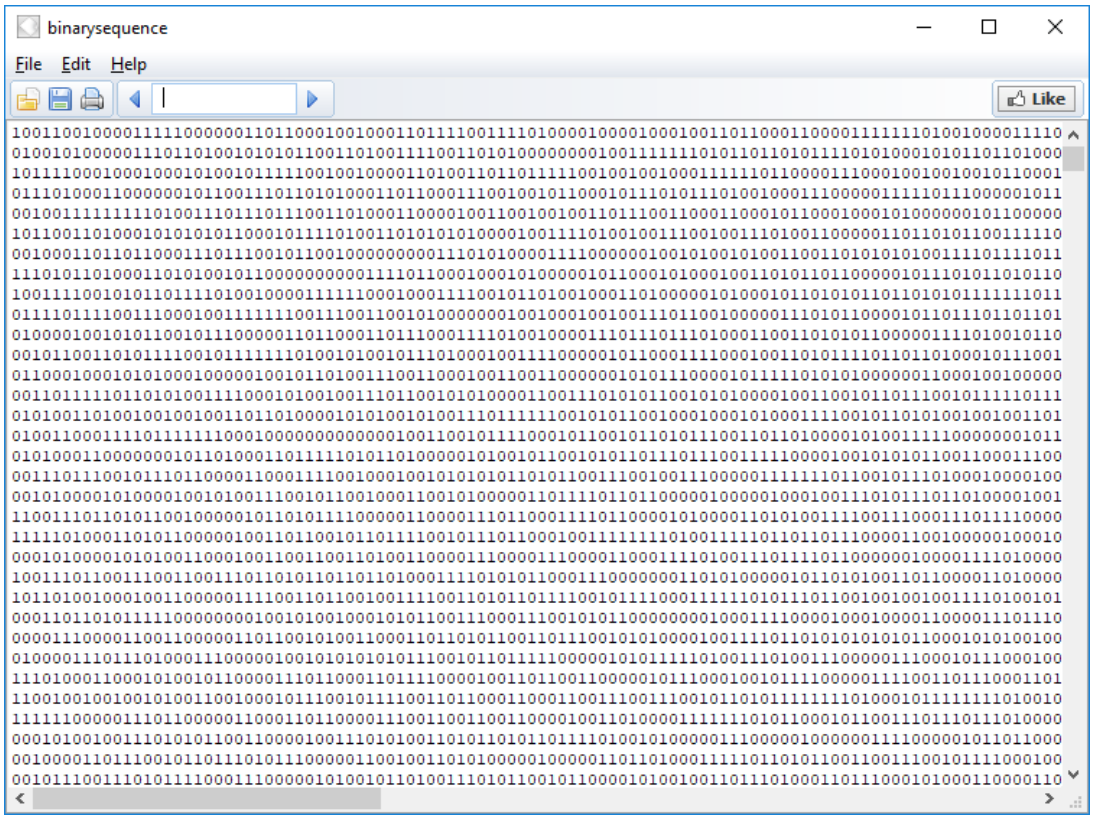
Pembangkitan true random number dilakukan dengan menggunakan metode operasi XOR. Masukan citra yang digunakan adalah *chipper image* hasil dari penerapan *arnold-cat map*. Namun, pada tahap awal pembangkitan, diperlukan masukan citra acak hasil dari PRNG untuk memastikan tidak adanya bias. Citra acak tersebut memiliki ukuran yang sama dengan *chipper image* hasil *arnold-cat map* untuk kemudian diterapkan operasi XOR antara keduanya. Sebelum dilakukan operasi XOR, piksel-piksel pada kedua citra dipindai untuk kemudian disusun dengan pola *continuous diagonal* dalam mendapatkan deret biner untuk meningkatkan keacakan. Secara umum, metode pembangkitan random number yang digunakan ditunjukkan pada gambar 6. Pola *continuous diagonal* untuk menyusun deret biner ditunjukkan pada gambar 7. Hasil dari *bitwise* XOR tersebut adalah berupa deret *random binary number* yang merupakan *output* hasil dari sistem ini. Untuk membangkitkan deret selanjutnya, dilakukan operasi XOR lagi antara deret biner dari *chipper image* yang lain dengan deret yang dihasilkan pada proses sebelumnya, begitu seterusnya. *True random number* yang dihasilkan pada masing-masing output adalah 230.400 *bits* sehingga hanya dengan lima tangkapan citra saja dapat menghasilkan lebih dari 1Mbits *random number*. Hasil deret biner yang dihasilkan seperti ditunjukkan pada Gambar 8.



Gambar 6 Pembangkitan Random Number



Gambar 7 Pola pemindaian *continuous diagonal*



Gambar 8 Deret biner yang dihasilkan

3. HASIL DAN PEMBAHASAN

Dari hasil implementasi dilakukan pengujian yang dilakukan untuk mengetahui kehandalan sistem dalam pembangkitkan *random number*. Pengujian dilakukan secara visual dan dengan uji statistik. Untuk mendapatkan data uji yang cukup, pertama, diambil beberapa buah citra. Kemudian citra-citra tersebut ditangani dengan metode pemrosesan seperti pada Bagian 2, sementara itu, *random number* yang dihasilkan dari citra-citra ini disimpan dalam beberapa file berbeda dengan masing-masing berukuran 1 Mbits.

3.1 Statistical Test

Uji statistik yang digunakan dalam penelitian ini adalah uji statistik NIST. Uji statistik NIST ini digunakan untuk menguji keacakan dari bit yang dihasilkan. Untuk setiap pengujian, ada nilai hasil yang disebut *p_value* ($0 \leq p_{value} \leq 1$) dihitung dari deret biner acak. Jika *p_value* lebih besar dari ambang batas yang ditentukan sebelumnya (secara *default* $\alpha = 0{,}01$) yang juga disebut tingkat signifikansi, artinya bahwa deret acak ini dikategorikan acak. Jika tidak, deret ini

dikategorikan tidak acak. Selain itu, jika *p_value* lebih mendekati 1, dapat dikatakan bahwa deret ini adalah deret acak yang lebih baik.

Dari Tabel 1, dapat diamati bahwa semua hasil deret biner dari metode yang telah diusulkan ini berhasil melewati 15 uji statistik NIST, sedangkan baik deret biner untuk citra awal tanpa pemrosesan maupun setelah diterapkan *preprocessing* dan *arnold cat map* saja gagal memenuhi uji statistik NIST. Hal ini membuktikan penggunaan metode dalam penelitian ini telah mampu menghasilkan *random number* yang benar-benar acak.

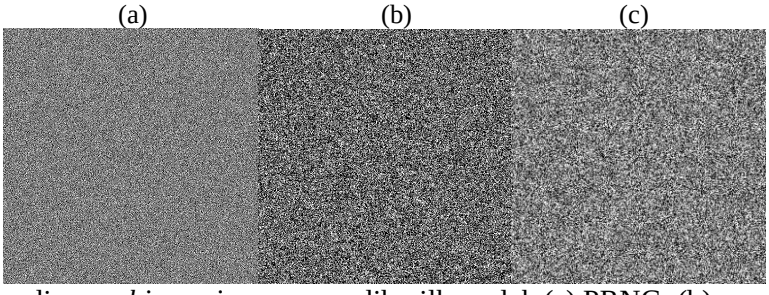
Pada tabel 1, masing-masing 15 uji statistik NIST adalah sebagai berikut: *frequency test* (FT), *test for frequency within a block* (FBT), *runs test* (RT), *test for the longest run of ones in a block* (LROBT), *aperiodic template matching test* (ATMT), *periodic template matching test* (PTMT), *Maurer's universal statistical test* (MUST), *serial test* (ST), *approximate entropy test* (AET), *cumulative sum test* (CST), *random excursions test* (RET), *random excursions variant test* (REVT), *random binary matrix rank test* (RBMRT), dan *linear complexity test* (LCT).

Tabel 1 Hasil Pengujian NIST

Pengujian NIST	**P-Value pada-**			
	Deret Citra awal	**Deret dengan *preprocessing* dan *arnold cat map***	**Deret setelah bitwise XOR tanpa *pseudo random***	**Deret setelah Bitwise XOR *pseudo-random* dan *chipper image***
FT	0	0.239595	0	0.509010
FBT	0	0	0	0.600409
RT	0	0	0	0.605265
LROBT	0	0	0.052228	0.403431
ATMT	0	0	0	0.733302
PTMT	0	0	0.000854	0.638169
MUST	0	0	0.038552	0.700650
ST	0	0	0.008299	0.250681
AET	0	0	0	0.521283
CST	1	0	0	0.633348
RET	0.761365	0.699986	0.582011	0.600468
REVT	0.823063	0.617075	0.573844	0.706734
RBMRT	0.068972	0.150352	0.927260	0.429680
LCT	0.462324	0.558453	0.333067	0.550044

3.2 Chipper-image Inspection

Tes visual merupakan cara mudah dan cepat bagi manusia untuk memeriksa karakteristik gambar dan parameter. Dalam tes ini diamati *cipher-images* yang dihasilkan oleh *Pseudo Random Number Generator* (PRNG), *Arnold Cat Map*, dan operasi XOR antara *chipper image* PRNG dan *Arnold Cat Map*. Gambar 9 menunjukkan perbandingan *chipper-image* yang dihasilkan.

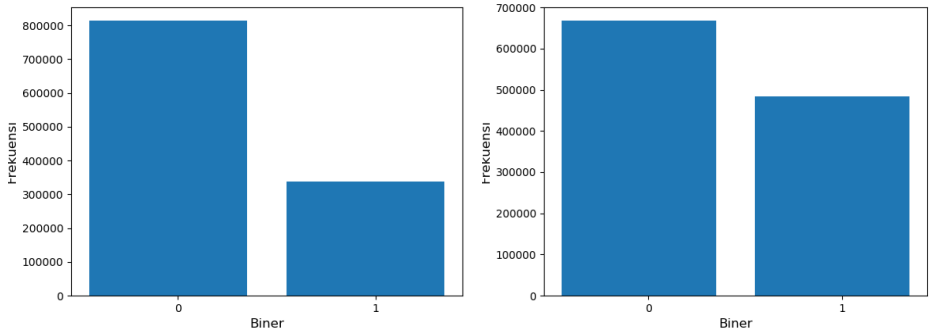


Gambar 9 Perbandingan *chipper-image* yang dihasilkan oleh (a) PRNG; (b) *arnold cat map*; (c) operasi XOR antara chipper image PRNG dan *arnold cat map*

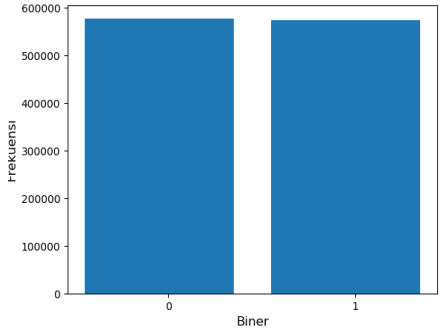
3.2 Uniformity Test

Random Number Generator yang dibangun ini tidak boleh bias, artinya sistem ini harus menghasilkan *random number* dengan probabilitas yang sama, atau dengan kata lain, angka biner yang muncul harus memiliki frekuensi penampilan yang sama. Pengujian ini dilakukan melalui fungsi plotting histogram. Dari hasil pengujian, *random number* yang dihasilkan memperlihatkan hasil dengan distribusi yang merata. Gambar 10 menunjukkan histogram *random number*.

(a) (b)



(c)



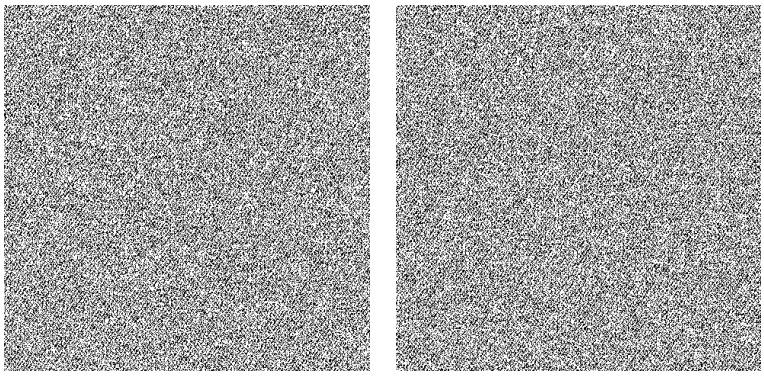
Gambar 10 Histogram *random number* (a) citra awal; (b) setelah *preprocessing* dan *arnold cat map* (c) hasil akhir *random number* setelah operasi XOR

3.2 Image hashing analysis

Dalam tes ini dilakukan *hashing* pada citra untuk diamati *average hash* pada citra yang mirip. Analisis ini dilakukan dengan menggunakan fungsi *average hash* pada Python. Hasil dari *hashing* pada dua buah citra asli yang sangat mirip dapat dilihat pada Gambar 11. *Hashing* yang dilakukan pada citra asli dapat diamati bahwa hasilnya adalah identik. Hal ini berarti sistem tidak dapat mengenali perbedaan pada kedua citra yang diamati. Namun, *hashing* yang dilakukan pada citra tersebut setelah dilakukan pemrosesan dengan *floyd-steinberg* dan *arnold-cat map* memperlihatkan *average hash* yang berbeda. Hal ini membuktikan bahwa algoritme kaotis pada sistem yang dibangun ini berhasil diterapkan karena keluaran sangat sensitif pada kondisi awal dengan perubahaan yang sangat kecil. Gambar 12 menunjukkan *average hash* pada dua citra yang mirip tersebut setelah dilakukan pemrosesan dengan *floyd-steinberg* dan *arnold cat map*.

1F1F070080F87EF8 1F1F070080F87EF8

Gambar 11 Perbandingan a*verage hash* yang dihasilkan oleh dua citra asli yang mirip dengan objek yang sama

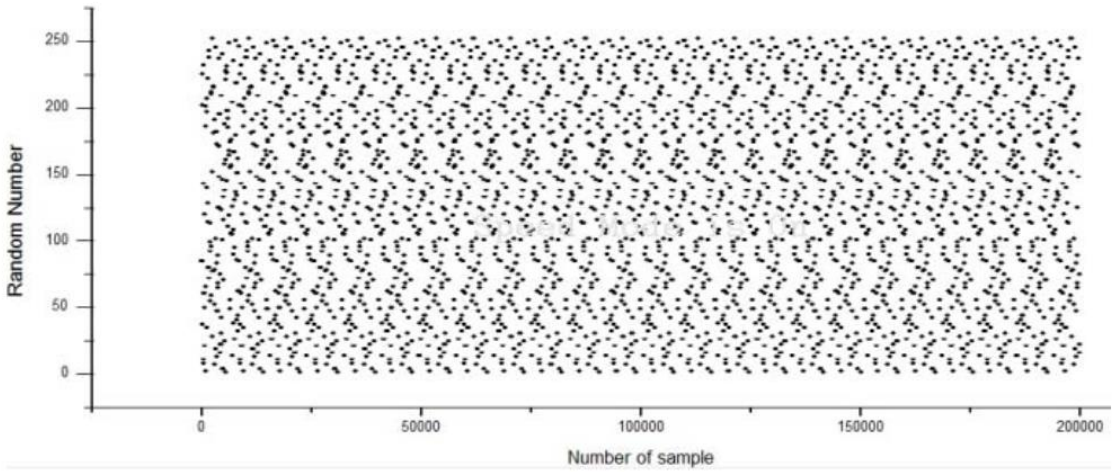


C52173F4C7F0A7F5 FEE910225BBF16F7

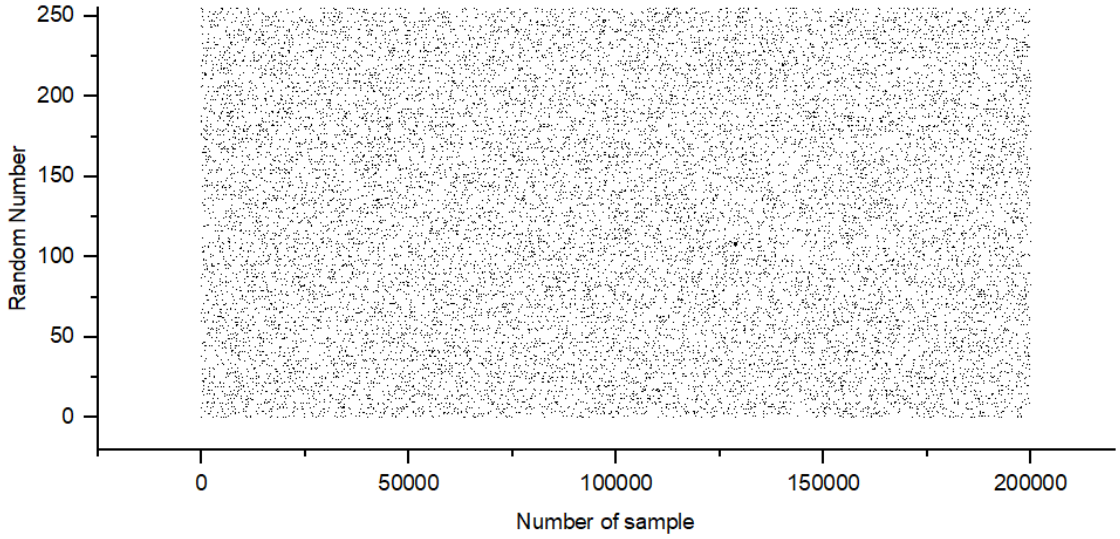
Gambar 12 A*verage hash* yang dihasilkan oleh dua citra yang mirip dengan objek yang sama setelah dilakukan pemrosesan

3.3 Scatter Analysis

Analisis pola keacakan dengan *scatter analysis* digunakan untuk membuat perbandingan data yang dihasilkan oleh PRNG dan TRNG yang dirancang. Seperti ditunjukkan pada Gambar 11, plot *scatter* PRNG memiliki pola yang sama dalam periode tertentu. Pola ini menjadi kelemahan *pseudorandom* dan bisa menjadi rentan bagi penyerang jika data PRNG digunakan pada bidang kriptografi. Gambar 12 menunjukkan plot *scatter* dari TRNG yang menghasilkan *random number* yang acak. Analisis scatter ini dilakukan dengan menggunakan software Origin™. Hasil ini bisa menjadi bukti bahwa data yang dihasilkan oleh TRNG ini benar-benar acak.



Gambar 11 Analisis scater dari *pseudorandom number*[1] [1]



Gambar 12 Analisis scater dari *true random number yang dibangun*

4. KESIMPULAN

Sistem *True Random Number Generator* berbasis citra menggunakan algoritme kaotis diusulkan dalam makalah ini. Sistem ini mampu menghasilkan deret *random number* sebanyak 230.400-bit dalam sekali proses. Uji keacakan diterapkan pada output TRNG ini dengan menggunakan uji statistik NIST. Uji statistik ini digunakan untuk mendapatkan nilai *p_value* yang merepresentasikan keacakan *random number*. Deret biner akan menjadi acak sempurna jika *p_value* sama dengan 1, dan *p_value* nol menunjukkan bahwa *random number* sepenuhnya tidak acak. Hasil tes menunjukkan bahwa semua nilai *p_value* lebih dari 0,01 yang menunjukkan deret output dari desain TRNG ini acak. Analisis *scatter* dan histogram juga dilakuan untuk membandingan pola data *random number* yang dihasilkan oleh TRNG ini dan *pseudorandom*. Analisis *scatter* dari *pseudorandom* menunjukkan adanya pola dalam waktu tertentu yang tidak ditemui di TRNG ini. Analisis histogram menunjukkan bahwa random number yang dihasilkan juga memiliki probabilitas yang sama atau tidak bias.

5. SARAN

Pada penelitian ini terdapat beberapa hal yang perlu disempurnakan. Sangat disarankan untuk dilakukan penelitian lebih lanjut untuk memperbaiki sistem ini, baik melakukan pencarian metode *preprocessing* dan *postprocessing* yang lebih komprehensif sehingga tingkat keacakan dapat ditingkatkan maupun jumlah *random bit* yang lebih banyak. Selain itu, pengujian tambahan juga bisa dilakukan menggunakan *diehard test suites*, *correlation test*, dan sebagainya.

DAFTAR PUSTAKA

[1] M. Siswanto and B. Rudiyanto, "Designing of Quantum Random Number Generator (QRNG) for Security Application," in *International Conference on Science in Information Technology (ICSITech)*, 2017, pp. 273–277 [Online]. Available: https://ieeexplore.ieee.org/document/8257124/. [Accessed: 16-June-2018]

[2] J. Morosi, M. Guasoni, A. Akrout, and J. Fatome, "Random Bit Generation through Polarization Chaos in Nonlinear Optical Fibers," in *Conference on Lasers and Electro-Optics Europe & European Quantum Electronics Conference (CLEO/Europe-EQEC)*, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8086464/. [Accessed: 18-June-2018]

[3] L. Gnoli, M. Bollo, M. Vacca, M. Graziano, and G. Di Natale, "True random number generator based on nanomagnets," in IEEE Nanotechnology Materials and Devices Conference (NMDC), 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7777089/. [Accessed: 18-June-2018]

[4] R. Li, "A True Random Number Generator algorithm from digital camera image noise for varying lighting conditions," in *Proceedings of the IEEE SoutheastCon*, 2015 [Online]. Available: https://ieeexplore.ieee.org/document/7132901/. [Accessed: 23-Dec-2017]

[5] I. Cicek and G. Dundar, "A chaos based integrated jitter booster circuit for true random number generators," in *European Conference on Circuit Theory and Design (ECCTD)*, 2013 [Online]. Available: https://ieeexplore.ieee.org/document/6662257/. [Accessed: 23-June-2017]

[6] I. Cicek, A. Pusane, and G. Dundar, "A novel dual entropy core true random number generator," in *International Conference on Electrical and Electronics Engineering (ELECO)*, 2013, pp. 332–335. [Online]. Available: https://ieeexplore.ieee.org/document/6713856/. [Accessed: 23-June-2018]

[7] I. Cicek, A. E. Pusane, and G. Dundar, "An Integrated Dual Entropy Core True Random Number Generator," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 64, no. 3, pp. 329–333, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7469345/. [Accessed: 23-June-2017]

[8] İ. Koyuncu, A. T. Özcerit, İ. Pehlivan, and E. Avaroğlu, "Design and implementation of chaos based true random number generator on FPGA," in *22nd Signal Processing and Communications Applications Conference (SIU)*, 2014, pp. 236–239. [Online]. Available: https://ieeexplore.ieee.org/document/6830209/. [Accessed: 23-May-2018]

[9] P. Kumar, S. Agarwal, and S. Shivani, "Halftone Visual Cryptography with Pixel Expansion," *Int. J. Inf. Comput. Technol.*, vol. 4, no. 14, pp. 1419–1427, 2014. [Online]. Available: https://www.ripublication.com/irph/ijict_spl/ijictv4n14spl_10.pdf. [Accessed: 25-May-2018]

[10] P. M. Nikate and I. I. Mujavar, "Performance Evaluation of Floyd Steinberg Halftoning and Jarvis Haltoning Algorithms in Visual Cryptography," *Int. J. Innov. Eng. Technol.*,

vol. 5, no. 1, pp. 336–342, 2015 [Online]. Available: ijiet.com/wp-content/uploads/2015/03/46.pdf. [Accessed: 26-May-2018]

[11] L. Skanderova and A. Řehoř, "Comparison of Pseudorandom Numbers Generators and Chaotic Numbers Generators used in Differential Evolution," in *Nostradamus 2014: Prediction, Modeling and Analysis of Complex Systems*, 2014, pp. 111–121. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-07401-6_11. [Accessed: 30-May-2018]

[12] Y. Wu, Z. Hua, and Y. Zhou, "n-Dimensional Discrete Cat Map Generation Using Laplace Expansions," *IEEE Trans. Cybern.*, vol. 46, no. 11, pp. 2622–2633, 2016 [Online]. Available: https://ieeexplore.ieee.org/document/7302020/. [Accessed: 28-May-2018]

[13] E. Avaroglu, "Pseudorandom number generator based on Arnold cat map and statistical analysis," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 25, no. 1, pp. 633–643, 2017 [Online]. Available: http://journals.tubitak.gov.tr/elektrik/issues/elk-17-25-1/elk-25-1-48-1507-253.pdf. [Accessed: 1-June-2018]

[14] P. Singh, B. Raman, and M. Misra, "A (n, n) threshold non-expansible XOR based visual cryptography with unique meaningful shares," *Signal Processing*, vol. 142, pp. 301–319, 2018 [Online]. Available: https://annals-csis.org/Volume_10/drp/47.html. [Accessed: 2-June-2018]

[15] L. Zhao, X. Liao, D. Xiao, T. Xiang, Q. Zhou, and S. Duan, "True random number generation from mobile telephone photo based on chaotic cryptography," *Chaos, Solitons & Fractals*, vol. 42, no. 3, pp. 1692–1699, 2009 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960077909001866. [Accessed: 3-Desember-2017]

[16] B. Mondal, N. Biswas, and T. Mandal, "A Comparative study on Cryptographic Image Scrambling," in *Proceedings of the Second International Conference on Research in Intelligent and Computing in Engineering*, 2017, vol. 10, pp. 261–268 [Online]. Available: https://annals-csis.org/Volume_10/drp/47.html. [Accessed: 20-June-2018]