INDICON-2014

Chaos Based Image Encryption using Latin Rectangle Scrambling

Santosh Chapaneri[1], Radhika Chapaneri[2],

[1] St. Francis Institute of Technology,

[2] Thadomal Shahani College of Engineering,

University of Mumbai

13-12-2014

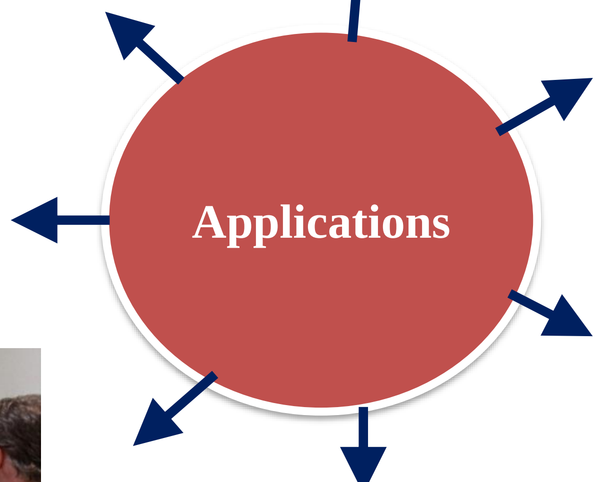
Problem: Image Encryption

Medical Imaging

Satellite Images

Military Images

Multimedia Security

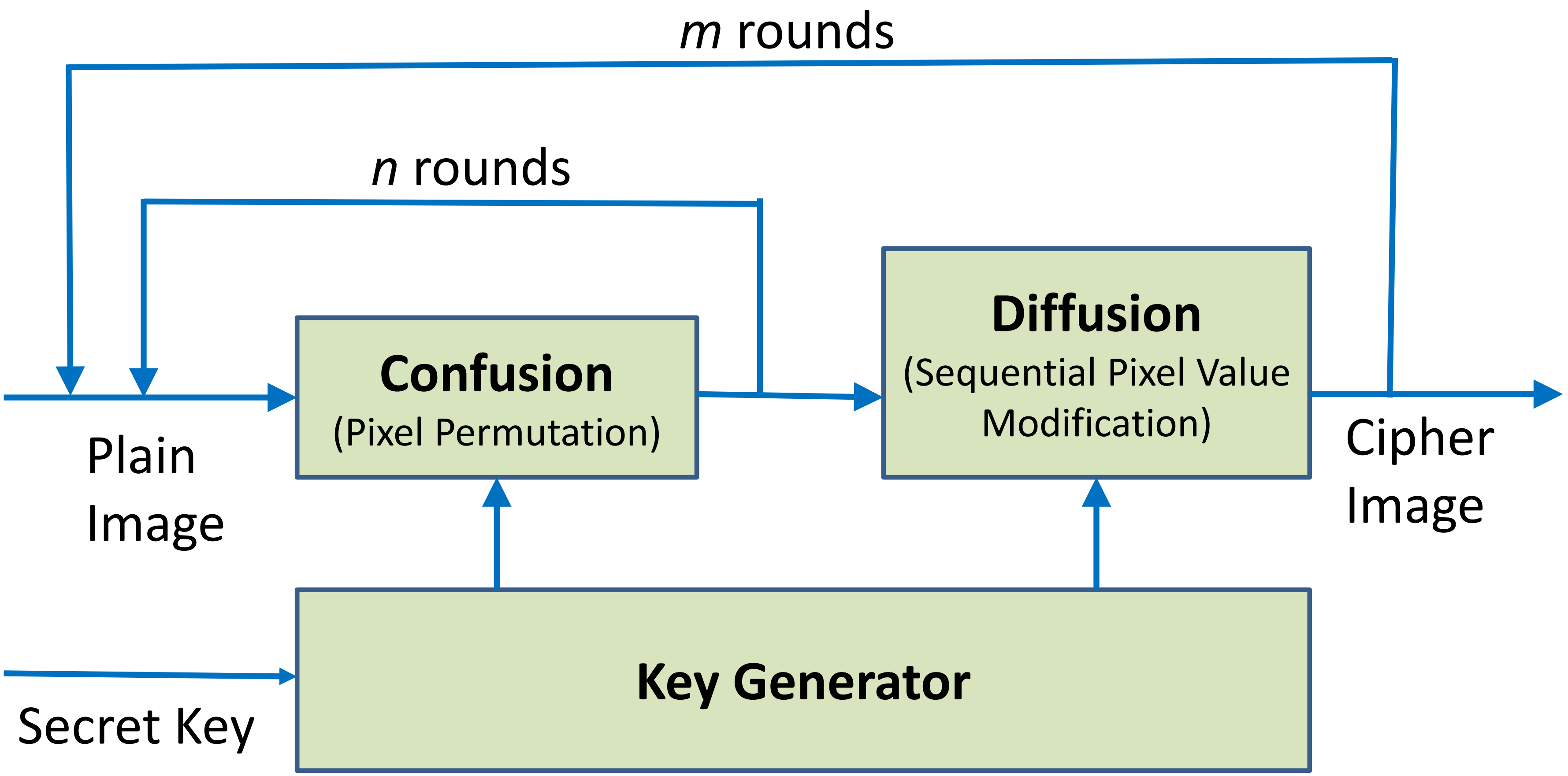


Confidential video conference

Pay-per-view TV

Aadhar card Enrolment

General Architecture

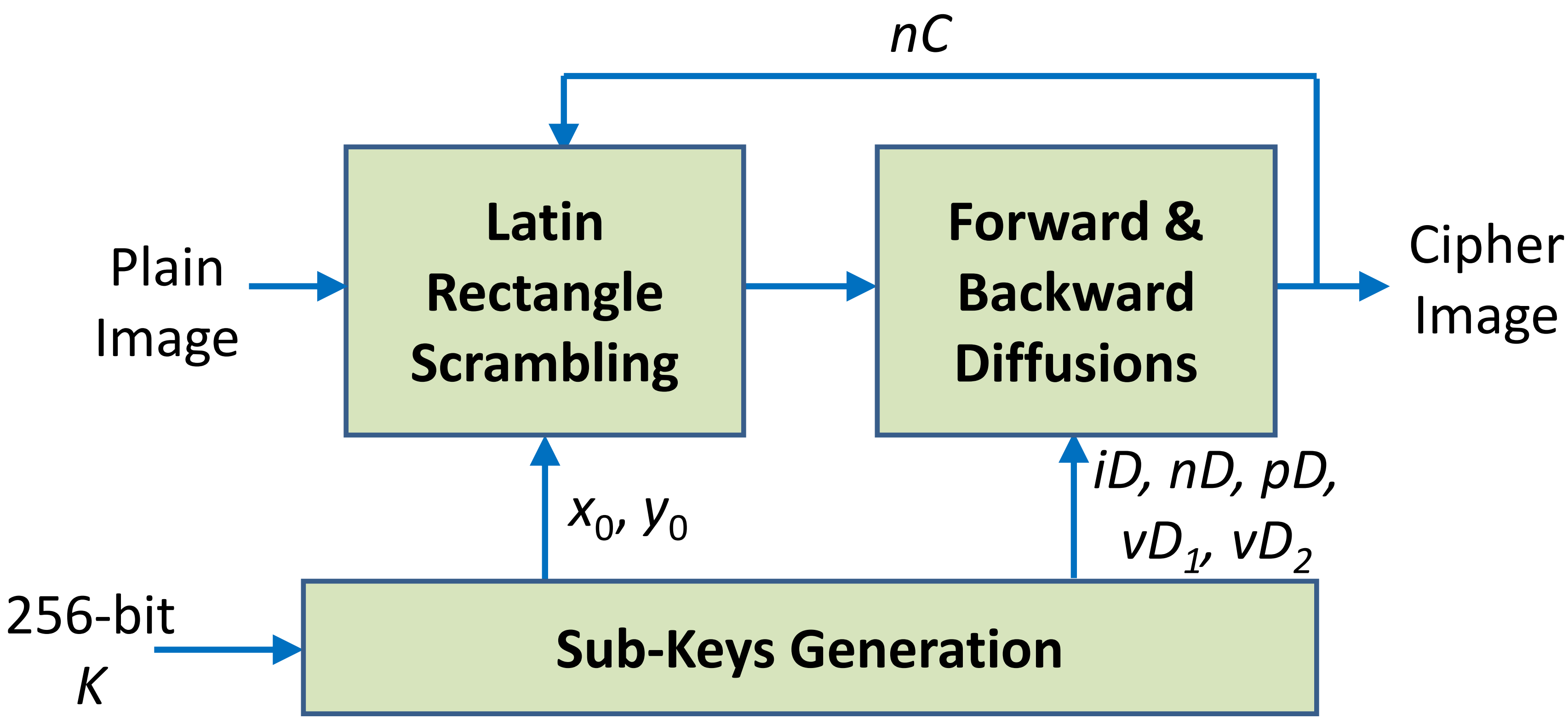


Related Work

Chaos: **Deterministic** model of **dynamical** system that yields **unpredictable** behavior

Chaos Based Solutions	Non-Chaos Based Solutions
Logistic Map [1]	Sudoku Matrix approach [4]
Baker Map [2]	Latin Square Image Cipher [5]
Arnold Map [3]	Selective Encryption [6]

Proposed Architecture



Latin Square (LS) / Latin Rectangle (LR)

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{bmatrix} \begin{bmatrix} \alpha & \beta & \delta \\ \delta & \alpha & \beta \\ \beta & \delta & \alpha \end{bmatrix} \qquad \begin{bmatrix} 1 & 3 & 2 & 4 & 5 \\ 2 & 1 & 4 & 3 & 5 \\ 5 & 2 & 1 & 4 & 3 \end{bmatrix} \begin{bmatrix} \alpha & \beta & \mu & \delta \\ \beta & \alpha & \delta & \mu \end{bmatrix}$$

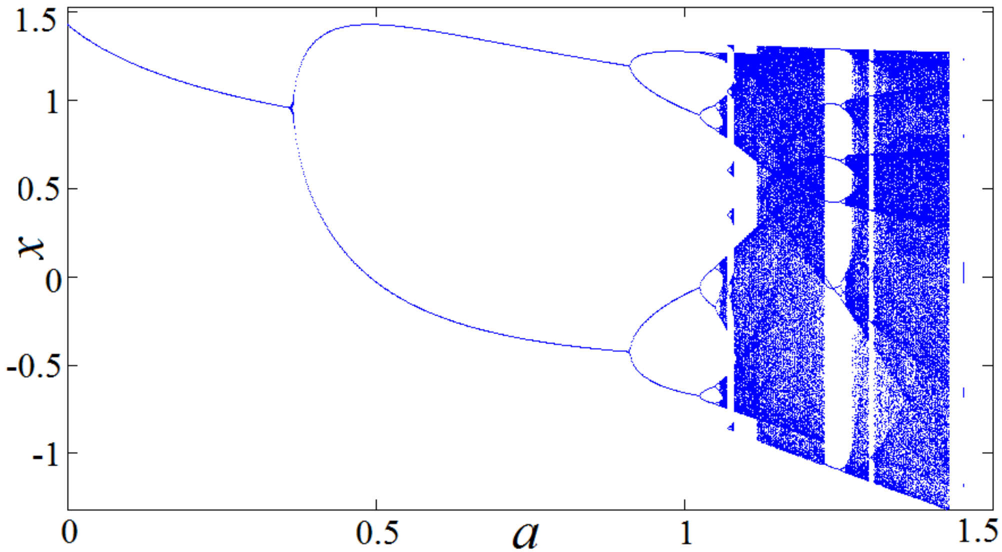
- *Latin Square* = *P* x *P* array such that each row and column contains every symbol from {1, 2, …, *P*} exactly once

- *Latin Rectangle* = *P* x *Q* array with elements occurring at most once in each row and column

Latin Rectangle Scrambling

- Using a discrete 2D Hénon chaotic map:

$$x_{n+1} = 1 - ax_n^2 + y_n$$
$$y_{n+1} = bx_n$$



Bifurcation Diagram

Chaotic when
a = 1.4, b = 0.3

Lyapunov Exponent = 0.4241

Latin Rectangle Scrambling

Algorithm I: Construction of Latin Square	
Input:	x_0, y_0 – initial conditions H, W – height and width of image to be scrambled
Out:	LS – Latin square of order max(H, W)
1.	$M = \max(H, W)$; $a = 1.4$; $b = 0.3$; $N_0 = 1000$
2.	for $n \leftarrow 1$ to $\mathbf{N_0+M}$ do $X(n), Y(n) \leftarrow HénonMap(x_0, y_0)$
3.	$X \leftarrow X(N_0+1: N_0+M)$; $Y \leftarrow Y(N_0+1: N_0+M)$
4.	$\boldsymbol{X \leftarrow 10^6 X - floor(10^6 X); Y \leftarrow 10^6 Y - floor(10^6 Y)}$
5.	$Q \leftarrow Sort(X)$; $F \leftarrow Sort(Y)$
6.	for $r \leftarrow 1$ to M do
7.	$LS(r, :) \leftarrow Q >> F(i)$
8.	end

Latin Rectangle Scrambling

Algorithm II: Latin Rectangle Scrambling

Input: I – plain image to be scrambled
LS – Latin square of order $M = \max(H, W)$

Out: SI – Scrambled image

1. If $\boldsymbol{H == W}$
2. for $i \leftarrow 1$ to H do $T(i,:) = I(i, LS(i,:)')$ % *Row*
3. for $j \leftarrow 1$ to W do $SI(:,j) = T(LS(:,j)', j)$ % *Column*
4. else if $\boldsymbol{H > W}$
5. for $j \leftarrow 1$ to W do $T(:,j) = I(LS(:,j)', j)$ % ***Column***
6. $LR \leftarrow \text{zeros}(H, W)$
7. for $i \leftarrow 1$ to M do
8. $r = LS(i,:)$; $\boldsymbol{r(r > W) = [\ ]}$; $LR(i,:) = r$; % ***LR***
9. for $i \leftarrow 1$ to H do $SI(i,:) = T(i, LR(i,:)')$ % ***Row***

Latin Rectangle Scrambling

10	3	50	130
4	15	77	137
15	54	102	125
75	113	113	112
146	126	97	110
168	115	89	104

(a)

4	5	6	2	1	3
5	6	1	3	2	4
6	1	2	4	3	5
3	4	5	1	6	2
1	2	3	5	4	6
2	3	4	6	5	1

(b)

4	5	6	2	1	3
5	6	1	3	2	4
6	1	2	4	3	5
3	4	5	1	6	2
1	2	3	5	4	6
2	3	4	6	5	1

(c)

75	126	89	137
146	115	50	125
168	3	77	112
15	113	97	130
10	15	102	110
4	54	113	104

(d)

4	2	1	3
1	3	2	4
1	2	4	3
3	4	1	2
1	2	3	4
2	3	4	1

(e)

137	126	75	89
146	50	115	125
168	3	112	77
97	130	15	113
10	15	102	110
54	113	104	4

(f)

(a) Plain Image (6 x 4),

(b) Latin Square (6 x 6),

(c) Partial Latin Square (6 x 4),

(d) Column Scrambling with (c),

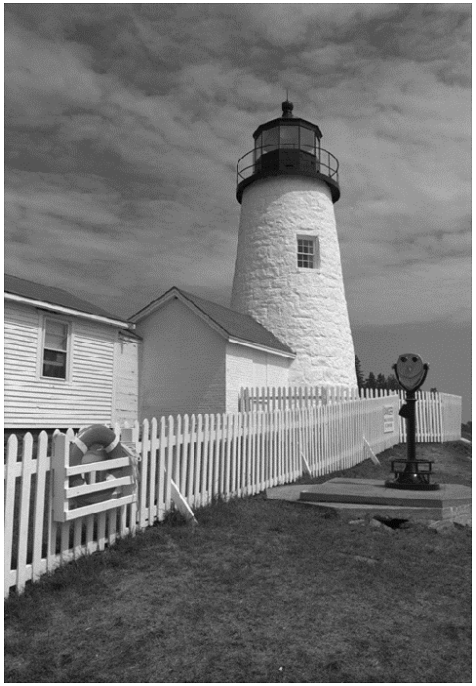
(e) Latin Rectangle (6 x 4),

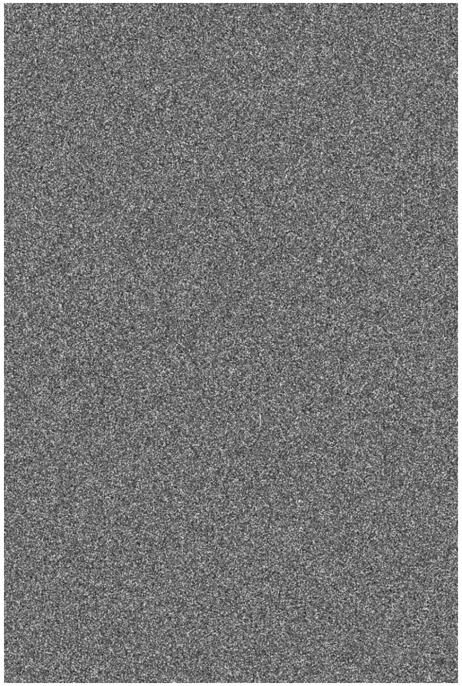
(f) Row Scrambling with (e)

INDICON-2014

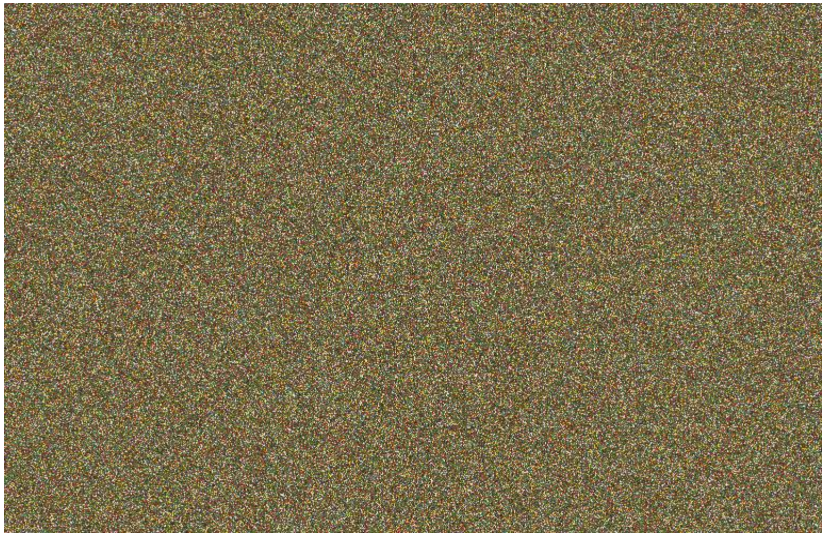
Latin Rectangle Scrambling

720 x 480





512 x 768



Pixel Diffusion

- Update pixel values to avoid differential cryptanalytic attack
- Typically done once per iteration => more iterations needed to achieve better security
- Proposed: perform **Forward** as well as **Backward** diffusions in single iteration => less iterations required

Bi-directional Pixel Diffusion

Plain Image

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	20
21	22	23	24

LR Scrambled Image

12	6	13	3
22	5	4	19
20	11	14	21
1	24	15	18
23	9	2	8
7	10	16	17

Forward Diffusion

12	6	13	3
22	5	4	19
20	11	14	21
1	55	32	78
167	174	176	136
143	249	233	216

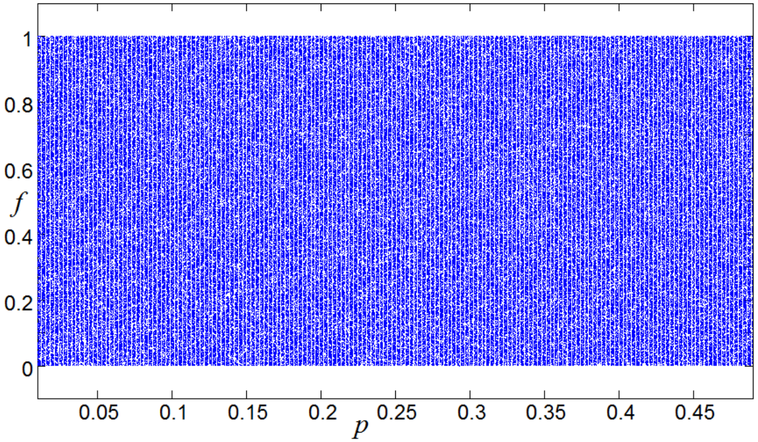
Backward Diffusion

223	216	119	210
203	220	118	98
197	201	9	26
250	37	41	139
249	80	159	146
211	66	100	187

Bi-directional Pixel Diffusion

- Using 1D Piecewise Linear Chaotic Map (PWLCM):

$$f(k) = F\left(f(k-1), p\right) = \begin{cases} f(k-1)/p & \text{if } f(k-1) \in [0, p); \\ \left(f(k-1)-p\right)/\left(0.5-p\right) & \text{if } f(k-1) \in [p, 0.5]; \\ F\left(1-f(k-1), p\right) & \text{if } f(k-1) \in (0.5, 1] \end{cases}$$



Discrete 1D PWLCM:

$$\phi(k) = \mathrm{mod}\left(floor\left(\left(\left(f(k)+1\right)/2\right)\times 10^{8}\right), L\right)$$

Bifurcation Diagram of PWLCM:

Chaotic in the range of control parameter $p \in (0, 0.5)$

Bi-directional Pixel Diffusion

Algorithm III: Bi-directional Pixel Diffusion	
Input:	*SI* – Latin scrambled image
	iD, nD, pD, vD_1, vD_2 – secret keys
Out:	*CI* – Cipher (encrypted image)
1.	$IS = H \times W$
2.	for $k \leftarrow 1$ to IS do $f(k) \leftarrow PWLCM(iD, nD, pD)$
3.	for $k \leftarrow 1$ to IS do $f(k) \leftarrow Digitize(f(k))$ % ***Discrete map***
4.	$SI \leftarrow Vectorize(SI)$ % *1-D vector*
5.	$TI \leftarrow ForwardDiffusion(SI, vD_1)$ % ***Forward Diff.***
6.	$CI \leftarrow BackwardDiffusion(TI, vD_2)$ % ***Backward Diff.***
7.	$CI \leftarrow Reshape(CI, H, W)$

$$TI(1) = \phi(1) \oplus \{[SI(1) + \phi(1)] \bmod N\} \oplus vD_1$$

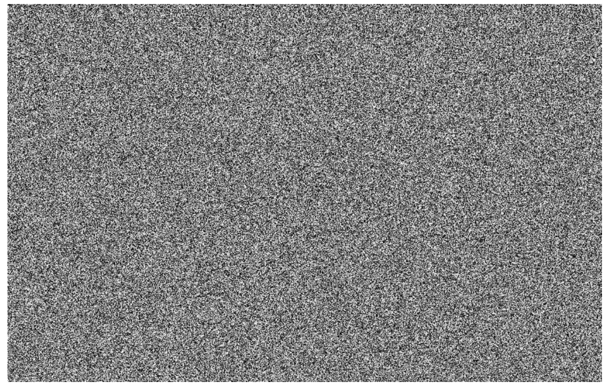
$$TI(k) = \phi(k) \oplus \{[SI(k) + \phi(k)] \bmod N\} \oplus TI(k-1)$$

$$CI(IS) = \phi(IS) \oplus \{[TI(IS) + \phi(IS)] \bmod N\} \oplus vD_2$$

$$CI(k) = \phi(k) \oplus \{[TI(k) + \phi(k)] \bmod N\} \oplus CI(k+1)$$

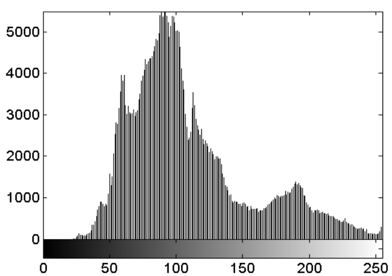
Results: Visual, Entropy, Correlation

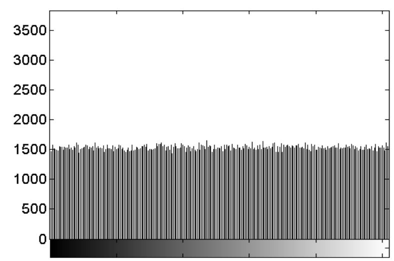
Plain Image (P)



Cipher Image (C)

Decrypted Image (MSE = 0)

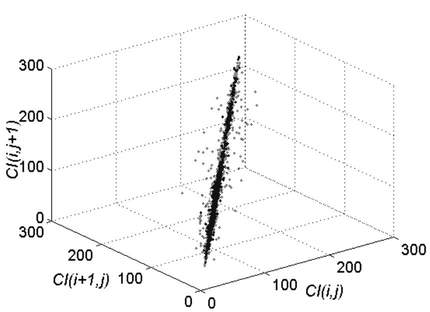


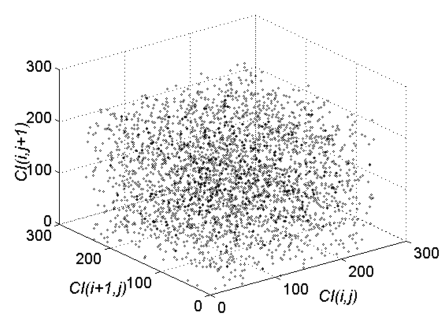


<= Histograms / Entropy

$HG(P)$ = 7.2859, $HG(C)$ = **7.9995**

$HL(P)$ = 5.5016, $HL(C)$ = **7.9109**





<= Correlation Scatter Plots

$\rho(P)$ = 0.9937, $\rho(C)$ = **0.0033**

Results: Diffusion Property Analysis

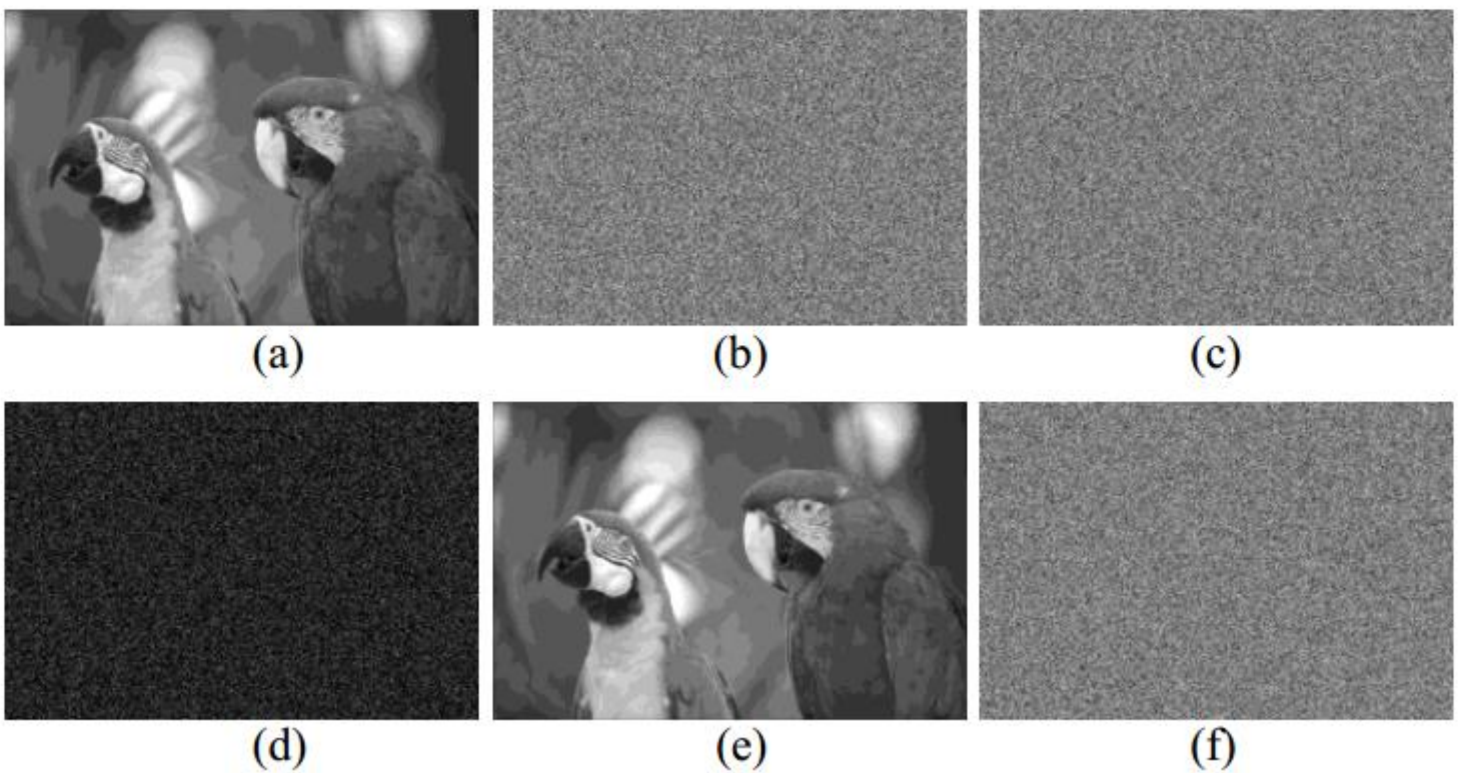
Encryption Method	GDD	NPCR%	UACI%
Ideally	1	> 99.609	> 33.464
Borujeni et al. [7]	0.9173	95.5293	33.4398
Behnia et al. [8]	0.8147	42.8195	32.4671
Proposed	**0.9951**	**99.6442**	**33.5061**

GDD = Gray Difference Degree (to evaluate scrambling)
NPCR = Number of Pixels Change Rate (to evaluate robustness to attacks)
UACI = Unified Average Change Intensity (robustness to attacks)

Results: Key Sensitivity Analysis

$\boldsymbol{K}^1$=79DE299EAF4E68FD4D6047B1318E87C21A165372E293C3D1ECAF635DE481ACC**0**

$\boldsymbol{K}^2$=79DE299EAF4E68FD4D6047B1318E87C21A165372E293C3D1ECAF635DE481ACC**1**



(a) Plain image P

(b) $C^1 = E(P, K^1)$

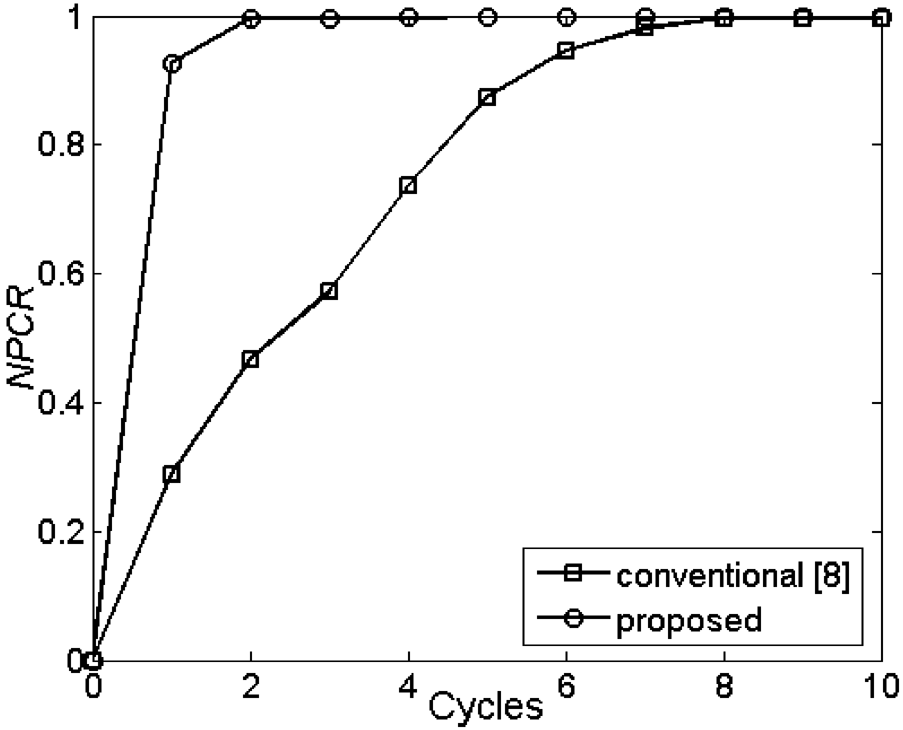
(c) $C^2 = E(P, K^2)$

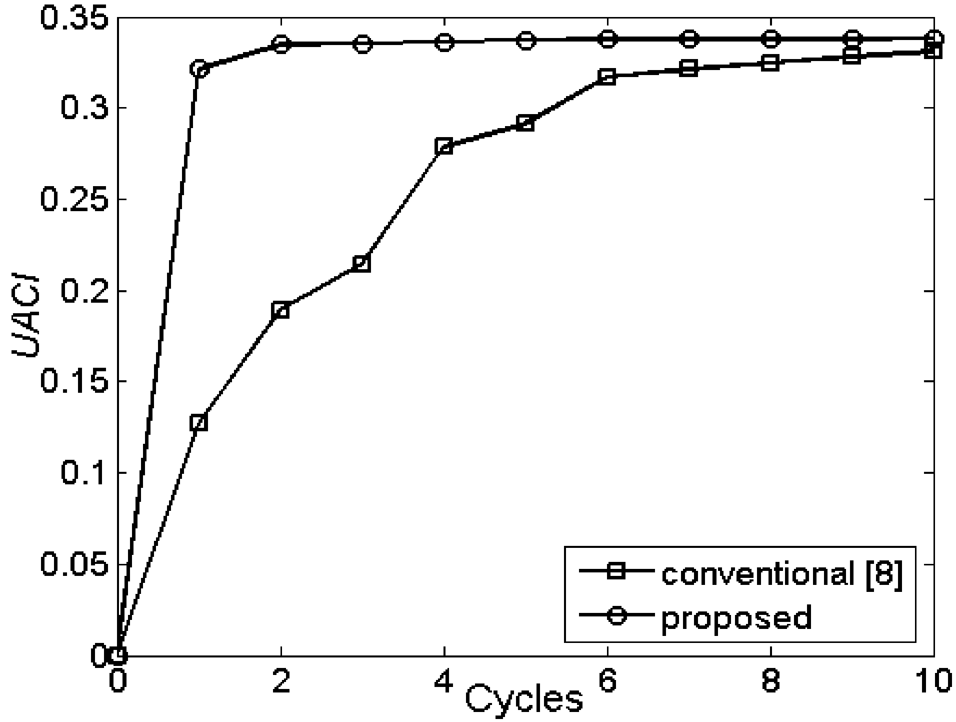
(d) difference $|C^1 - C^2|$

(e) $D^1 = D(C^1, K^1)$

(f) $D^2 = D(C^1, K^2)$

Results: Computation Analysis





Only **3 cycles** needed to achieve enhanced security and resist cryptanalytic attacks with proposed system

Conclusions

- Novel Latin rectangle scrambler for confusion
- Bi-directional pixel diffusion for robustness to differential attacks
- Completely invertible
- Reduced overall number of cycles to achieve enhanced security

References

1) N. Pareekh, “Image encryption using chaotic logistic map”, *Elsevier Image and Vision Computing*, pp. 926–934, Feb 2006
2) Y. Mao, G. Chen, S. Lian, **“**A Novel Fast Image Encryption Scheme Based on 3D Chaotic Baker Maps”, *Intl. Journal of Bifurcation and Chaos*, vol. 14, no. 10, pp. 3613-3624, Oct 2004
3) S. Keshari, S. Modani, “Image Encryption Algorithm based on Chaotic Map Lattice and Arnold cat map for Secure Transmission”, *IJCST*, vol. 2, no. 1, Mar 2011
4) Y. Wu, Y. Zhou, J. P. Noonan, K. Panetta, and S. Agaian, **“**Image Encryption using the Sudoku Matrix”, *SPIE Digital library***,** vol. 7708, 2010
5) Y. Wu, Y. Zhou, J. Noonan, and S. Agaian, “Design of image cipher using Latin squares”, *Elsevier Jour. Information Sciences*, Nov 2013
6) L. Krikor, “Image Encryption Using DCT and Stream Cipher”, *European Journal of Scientific Research*, vol.32, no.1, pp.48-58, 2009
7) S. Borujeni, M. Eshghi, “Chaotic image encryption system using phase-magnitude transformation and pixel substitution”, *Springer Jour. Telecommunication Systems*, pp. 1-13, May 2011
8) S. Behnia, A. Akhshani, H. Mahmodi, A. Akhavan, “A novel algorithm for image encryption based on mixture of chaotic maps,” Chaos, Solitons and Fractals, vol. 35, no.2, pp. 408–419, 2008

Appendix

x_0 – double (0.0, 1.0)	nD – integer (>= 25)
y_0 – double (0.0, 1.0)	pD – double (0.0, 0.5)
nC – integer (>= 2)	vD_1 – integer (0, 255)
iD – double (0.0, 1.0)	vD_2 – integer (0, 255)

$$v(b) = \sum_{i=1}^{q} b_{-i} 2^{-i}$$

$$GD_{i,j}^{P} = \frac{1}{4} \sum_{i'j' \in \{-1,+1\}} \left[P(i,j) - P(i',j')\right]^2$$

$$EGD^{P} = \frac{\sum_{i=2}^{H-1} \sum_{j=2}^{W-1} GD^{P}(i,j)}{(H-2)\times(W-2)}$$

$$\boxed{GDD(P,C) = \frac{EGD^{P} - EGD^{C}}{EGD^{P} + EGD^{C}}}$$

$$\boxed{N(C^1, C^2) = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{Diff(i,j)}{M \times N} \times 100\%}$$

$$Diff(i,j) = \begin{cases} 0, \text{ if } C^1(i,j) = C^2(i,j) \\ 1, \text{ if } C^1(i,j) \neq C^2(i,j) \end{cases}$$

$$\boxed{U(C^1, C^2) = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{\left|C^1(i,j) - C^2(i,j)\right|}{256 \times M \times N} \times 100\%}$$