Chapter 1
Cellular Automata for Efficient Image and Video Compression

Radu Dogaru and Ioana Dogaru

Abstract. This chapter focuses on applications of cellular automata for image and video compression leading to high efficiency implementations (i.e. relatively simple operators and algorithms, and extra functionality such as encryption, achieved with no additional resources). First, a CA-based alternative to compressive sensing is presented, assuming that an image sensor is available where pixels can be addressed randomly. The key idea is to replace the traditional raster scan counter addressing the sensing units, with a "chaotic counter" behaving like a pseudo-random number generator but having in addition the binary synchronization property. Consequently, only a fraction of all pixels (the most relevant) are rapidly scanned. A certain class of hybrid CA (HCA) implements the chaotic counter. A recovery algorithm, implemented in the receiving unit, reconstructs the missing (less relevant) pixels. Another CA-based method presented herein is a vector-quantization method where color or gray images are decomposed in binary bit-planes and compression is achieved by replacing binary blocks within bit-planes with similar binary vectors from an indexed dictionary, previously generated by a properly designed CA. Finally, aspects of efficiently implementing the CA modules present in both schemes, are briefly discussed.

1.1 Introduction and Motivation

In various applications (remote sensing, secure data transmission, compressive sensing [1] some message (represented by the generic sequence of samples) must be passed from a transmitter system (next abbreviated "Tx") to a remote receiver

Radu Dogaru · Ioana Dogaru
University "Politehnica" of Bucharest,
Dept. of Applied Electronics and Information Engineering,
Natural Computing Laboratory, Room B232, Bvd. Iuliu Maniu 1-3,
Sector 6, Bucharest, Romania
e-mail: `radu_d@ieee.org, ioana.dogaru@upb.ro`

(next abbreviated “Rx”) or storage medium with an efficient use of the bandwidth or storage memory. In addition, in order to maintain data security certain encryption algorithms may be also used. The problem as stated above is usually approached by various methods form the mature areas of image and video compression and cryptography. Yet, some applications require low power consumption and consequently a simple mechanism is needed for encoding and decoding processes.

Cellular automata (CA) hold the promise [25] [6] [5] of a very convenient way to achieve both compression and encryption with the benefit of using simple circuit models leading to low power consumption, as often desired when the source of signal is a stand-alone sensor unit powered by battery or/and solar energy. In cryptology, cellular automata are widely used [8], patents on cellular-automata random number generators being among the first associated with CA applications [31].

In this chapter recent research results in this area are summarized, within a general compression framework depicted in Figure 1.1. The particularity of our approach is to exploit complex dynamics emerging in Boolean cellular automata for various tasks in compression and encryption, usually approached with computationally intensive algorithms using various arithmetic operations such as cosine and other kind of transforms, multiplications etc. Locating useful emergent behaviors in CA and their potential applications are topics described in more detail in [12] and in a series of recent papers [13] [17].

The use of cellular automata in various stages of the compression and decompression phases allows the avoidance of arithmetic operators such as multiplication, summation etc. with a dramatic impact on the architectural complexity of the algorithms implementation. Within this paper only the case of lossy compression is considered, which is effective for image and video content associated to the message. Also the focus is on algorithms that would lower the complexity of the Tx unit (often a stand alone smart sensor with critical power consumption requirements).

As seen in Figure 1.1, in order to compress the string x_t two stages (depicted here as A and B) will be considered. As detailed next, they can be applied independently (i.e. A only, or B only) or consecutively. In stage A, the redundancy present in the original message is exploited to reduce the number N of original samples (i.e. image pixels) into a smaller K number of representative samples. In the corresponding A-stage of the receiver (Rx), the missing samples are recovered (with a certain loss) using various interpolation schemes. Stage A is reminiscent of the compressive sampling approach but the similarity is only at the functional level; while compressive sampling approaches produce the $K-$sized vector s_k as the result of multiplying the message with an $N \times K$ matrix with random (non-binary) elements, our approach dubbed “chaotic scan” [19] simply picks some samples from the original message without performing any arithmetic operation at all. It is the role of a cellular automaton to select (by addressing) the K samples from the original message, as it will be detailed in Section 1.2. Consequently, a highly intensive computational algorithm (usually embedded into a sensor with low power consumption requirements) is replaced now with a much simpler implementation of a CA with $n = \log_2 N$ cells addressing the selection of K samples. In terms of FPGA implementation each CA cell is allocated to one single LUT (basic computational unit in FPGA), a far more

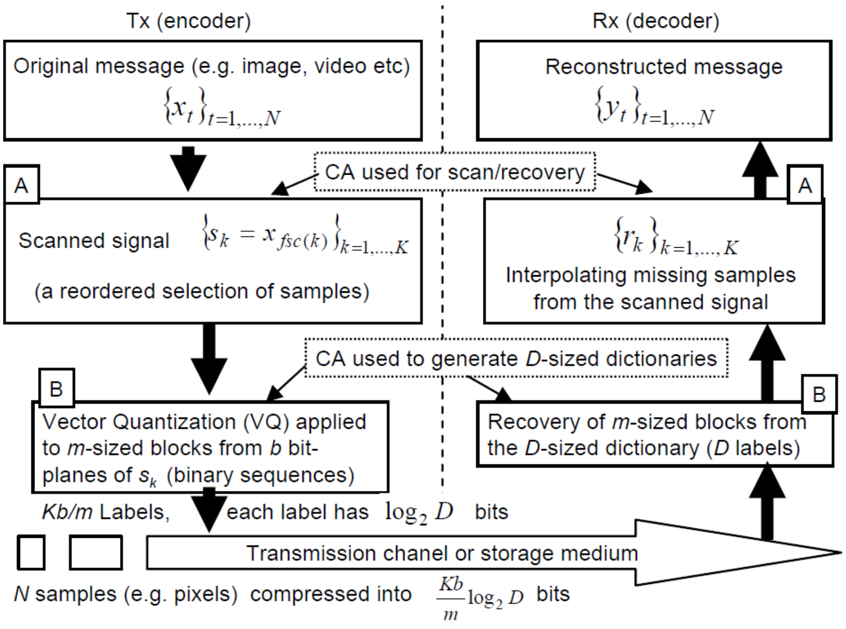
Tx (encoder)
Rx (decoder)
Original message (e.g. image, video etc)
$\{x_t\}_{t=1,...,N}$
Reconstructed message
$\{y_t\}_{t=1,...,N}$
CA used for scan/recovery
A
Scanned signal $\{s_k = x_{fsc(k)}\}_{k=1,...,K}$
(a reordered selection of samples)
$\{r_k\}_{k=1,...,K}$
Interpolating missing samples from the scanned signal
CA used to generate D-sized dictionaries
B
Vector Quantization (VQ) applied to m-sized blocks from b bit-planes of s_k (binary sequences)
Recovery of m-sized blocks from the D-sized dictionary (D labels)
Kb/m Labels, each label has $\log_2 D$ bits
Transmission chanel or storage medium
N samples (e.g. pixels) compressed into $\frac{Kb}{m}\log_2 D$ bits

Fig. 1.1 A general framework for message compression and recovery

effective solution than implementing the matrix multiplication required by the compressive sampling approach. It was shown [19] that using a ratio $K/N = 5\%$ ensures a decent recovery of the original signal, while preserving the most important features of the image (or video sequence). Consequently a compression of up to 20 times can be achieved in the A-stage of the encoder with very little computational effort. More computational effort is required in the A-stage of the Rx unit [19] [16] but in most applications of interest the Rx is usually implemented on a desktop computer with no critical requirementes, while the Tx is often an autonomous low-power sensor where the issue of low complexity is critical.

An identical CA structure as the one used to scan the original message is required in the Rx unit. In order to ensure the correct recovery of information, the CA in the Rx unit must evolve (as a dynamic system) in synchrony with the CA used in the Tx unit. This property is supported, as recently was demonstrated [14] by a proper choice of particular CA rules. Such CA rules ensure both **binary synchronization** and **longest cycle properties**. Binary synchronization means that the entire state of the CA (n bits addressing the t sample of the original message) can be recovered from one single bit per clock received from the Tx CA. Among various chaos synchronization schemes [24] [27] presented in the literature this one is the most robust and requires the less synchronization information. The longest cycle property means that eventually all (or almost all, with an insignificant loss of samples) N samples of the original message are addressed (much like a counting automaton, except the pseudo-random ordering or scan) ensuring that, if desired, all information from the original message is preserved (yet scrambled) without compression i.e. $K \cong N$. In fact, the user-selected K parameter represents the number of cycles advanced by the properly designed CA used in chaotic scan and can be traded off for the quality of the reconstructed message.

Compared to the CA-based A-stage, in terms of functionality, the closest approach found in the literature seems to be the “holographic scan” [3] although the implementation of this algorithm appears to be more complex than implementing a cellular automaton. Other traditional approaches implementing stage-A are the use of various image transforms (requiring intensive arithmetic computations) such as DCT, Karhuenen-Loeve or PCA, kernel-PCA [26] [28] or neural auto-encoders [23].

Stage-B is basically a Vector Quantization (VQ) stage. Blocks of the original message (or compressed, resulting from the A-stage) are m-sized vectors that would be compared with codebook vectors in a D-sized dictionary (previously prepared to optimize the compression efficiency for a class of messages). Consequently a label (among all D possible) is selected indicating the closest codebook vector. Traditional approaches to VQ employ computationally intensive arithmetic operations performed in fixed-point representations of the variables (including message samples). In contrast, our CA-based approach, to be detailed in Section 1.3, has two simplifying features: i) **the message is divided in binary bit-planes** (i.e. binary message sequences associated to one rank bit in the original sequence) as seen in Figure 1.2 such that m-sized vectors are now m-bit binary words; ii) **the codebook is a list of m-sized binary vectors**, generated by a 2-dimensional CA with a certain rule. The CA rule may be obtained using a training approach (as detailed in this chapter) or it can be the result of a selection process [21] [11]. In effect, computationally intensive arithmetic operators are replaced with simple logic operators and the codebook is simply generated based on CA rule information only (there is no need to send the entire codebook from Tx to Rx). In terms of performance (evaluated here as the PSNR - Power to Signal Noise Ratio - between original and recovered message) the CA-VQ approach is comparable to traditional image compression algorithms (e.g. JPEG) and in fact is more effective for low bit-rates (under 0.25 bits / sample). An FPGA implementation is reported in [32].

Mixed approaches involve exploiting both A and B stages. For instance, one may scan only $K < N$ samples from the original message and submit to the VQ system an incomplete m-sized vector (some bits are not specified since they belong to samples that were not scanned). Still a codebook search can be performed for the best match and the result is an improved compression rate (with an N/K factor) at a slightly degraded PSNR performance.

In order to implement the proposed compression methods various technologies may be used. Of a particular interest is the possibility to embed compression algorithms into smart sensors with low power requirements. Consequently, it is important to choose convenient synthesis solutions such that the whole algorithm is described in a hardware description language (e.g. VHDL or Verilog).

s_1^0	s_2^0	$\cdots$	s_k^0	$\cdots$	s_{K-1}^0	s_K^0	Bitplane 0
s_1^1	s_2^1	$\cdots$	s_k^1	$\cdots$	s_{K-1}^1	s_K^1	Bitplane 1
s_1^2	s_2^2	$\cdots$	s_k^2	$\cdots$	s_{K-1}^2	s_K^2	Bitplane 2
s_1^3	s_2^3	$\cdots$	s_k^3	$\cdots$	s_{K-1}^3	s_K^3	Bitplane 3
s_1^4	s_2^4	$\cdots$	s_k^4	$\cdots$	s_{K-1}^4	s_K^4	Bitplane 4
s_1^5	s_2^5	$\cdots$	s_k^5	$\cdots$	s_{K-1}^5	s_K^5	Bitplane 5
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_1^b	s_2^b	$\cdots$	s_k^b	$\cdots$	s_{K-1}^b	s_K^b	Bitplane b

Fig. 1.2 Organization of signals in b binary sequences called *bit-planes*. Bitplane j is associated to the binary sequence $s_k^j, k = 1 \dots K$. Each sample $s_k$is represented on b bits.$s_k = [s_k^0, s_k^1, \cdots, s_k^{b-1}]$ with $s_k^j \varepsilon \{0,1\}$(binary).

In Section 1.4 an efficient method to implement CA systems in FPGA technologies is briefly discussed in relation with the representation of the CA local rule using Algebraic Normal Form (ANF).

1.2 An Alternative to Compressive Sensing Based on Cellular Automata Scan

1.2.1 Principle of Chaotic Scan

This approach to message compression is a compact alternative to the compressive sensing methods [1] and was first proposed in [19]. The simplified model is given in Fig. 1.3, with regards to an image sensor. The idea may be further expanded to any other kind of multi-dimensional sensor in order to reduce the number of samples effectively transmitted from the sensor. The method is effective assuming that adjacent elements in the image array are highly correlated.

The nonlinear dynamic system (discrete-time, discrete-state) present in both Tx (encoding) and Rx (decoding) units can be any kind of automaton as long as it ensures certain properties to be detailed next. Since its role is to address (count) K pixels (or in general, K samples) from the N samples of the original message it will be called next a "chaotic counter". The term "chaotic" is a simplification from "pseudo-random", the above mentioned system implementing in fact a pseudo-random number generator. As discussed next in more detail, a convenient chaotic counter belongs to a class of 1-dimensional cellular automata having all desired properties for such a system. The contrast with a raster scanning counter is exemplified in Fig. 1.4. Here only 5% of the pixels in the original image) were addressed sequentially during the

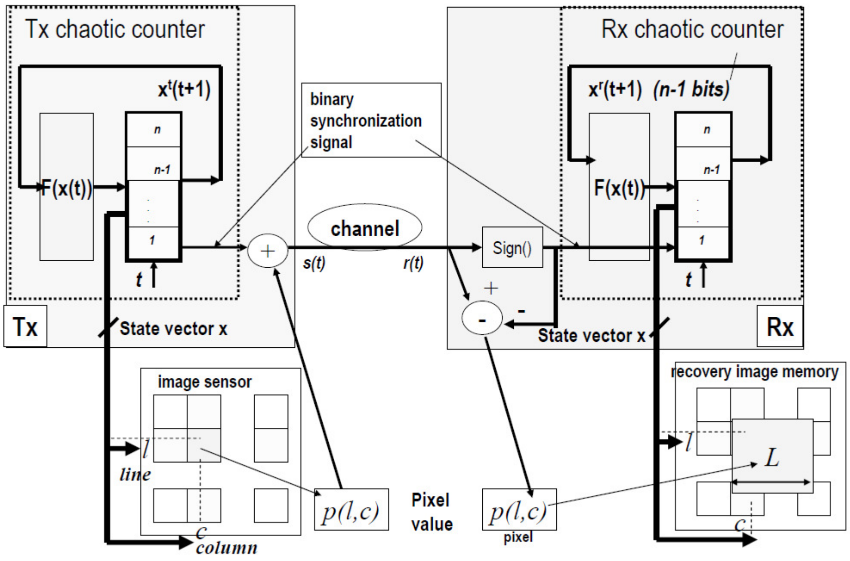

Fig. 1.3 An alternative model for compressed sensing based on chaotic scan: Pixels in a sensor array are chaotically selected by the chaotic counter. At receiving point a similar chaotic counter, synchronized with the one in Tx, is recomposing the image form the serially received pixel value while estimating its *L*-sized neighborhood. A small fraction of samples from the Tx image suffices to recover a good quality replica of the original image with a certain acceptable information loss.

counting process. While in the case of the raster scan a single image strip with no meaningful content is obtained, the chaotic counter locates distant uncorrelated pixels (at apparently random locations) making the image content recognizable. Based on the assumption that pixels in the neighborhood are often correlated, a simple reconstruction scheme is employed to recover the missing pixels. It consists in filling a $L \times L$ window with the same intensity value (the one of the received pixel). As seen in Fig. 1.4, when an optimal L is selected, this reconstruction scheme allows the recovery of most the important semantic image content.

Consequently, the scanning process of K samples is given by the following low complexity and easy to implement algorithm, where $X(i,j)$ represents the original square image message with N samples:

```
RESET the counting automaton
FOR $k=1,..K$
i=rand_i;
j=rand_j;
$s(k)=X(i,j)$;
END
```

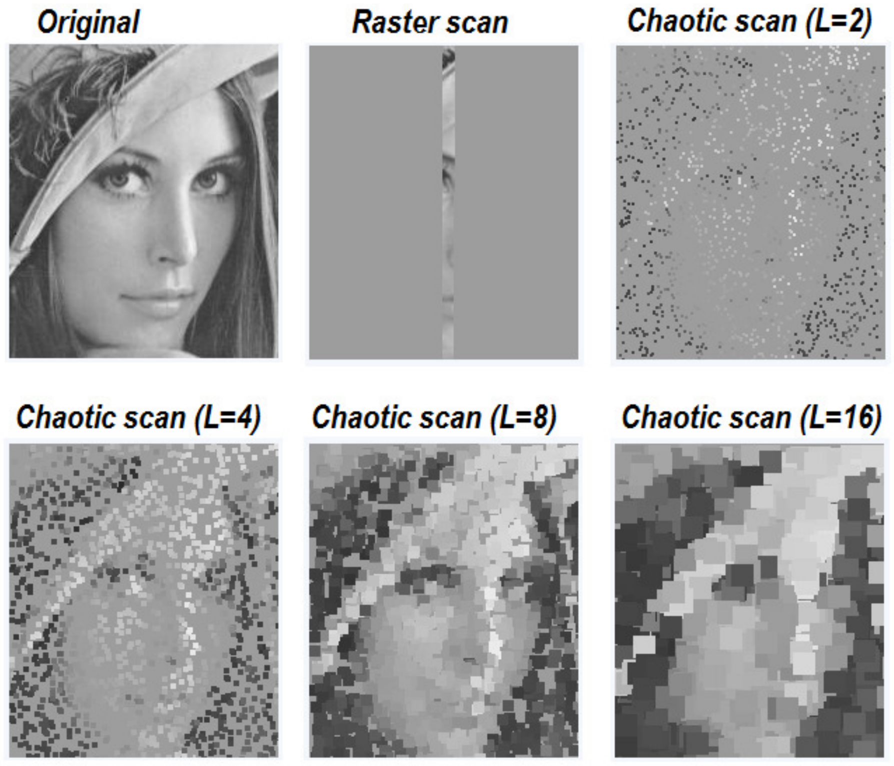

Fig. 1.4 Image recovery with chaotic scan after receiving $K = 0.05N$ pixels (5% of all N pixels of the original image); Too small values of the recovery block size (L) makes the received image difficult to read while using too large L values impede on image details. Depending on the image content there is an optimal L value (here, $L = 8$) such that the semantic content is best revealed.

Without loss of generality in the next we will use **rand_i rand_j** to denote the reading of the next state of the counting automaton (split into $n/2$ bits for line "i" and $n/2$ bits for column "j").

1.2.2 Properties of the Chaotic Counters

In order to understand the properties required for the chaotic counter, we shall consider it as a particular case of discrete-time, discrete-space dynamic system (automaton). Any such dynamic system, also called a *nonlinear map* is defined by a feedback function F imposing a certain profile of the state space (partitioned into certain attractors and their basins of attraction).

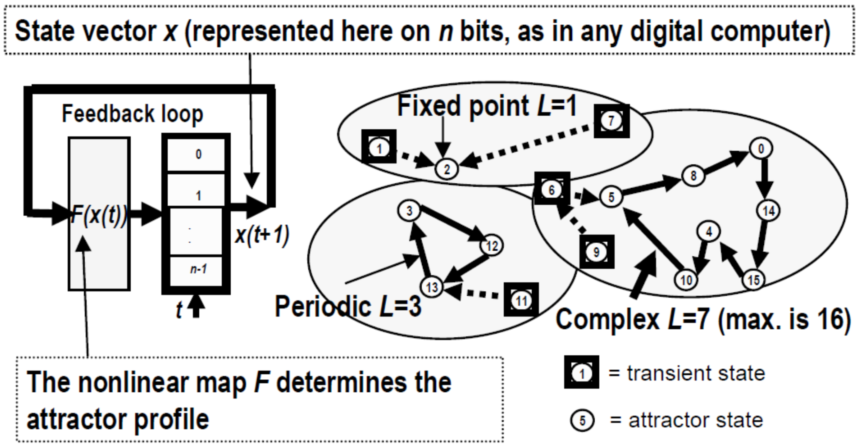

Fig. 1.5 General structure of a nonlinear map or automaton in a digital implementation (discrete-time, finite precision of n bits). In the case of cellular automaton, each bit is associated with a cell. The nonlinear mapping F induces a structure of the state space which is partitioned in attractors, each attractor may also collect "transient states" associated with its basin of attraction. The complexity of attractors is proportional with their length (maximal length is $L = 2^n$) and average distance between consecutive states.

Good chaotic counters must rely on:

i) The existence of a **dominant long attractor cycle** with a length L as close as possible to the maximal length i.e. $N = 2^n$; This property ensures that almost all pixels in the original image are addressed. The existence of transients and of many shorter cycles, imply some loss of pixels (samples) from the original message. Recently, considering CA as a nonlinear dynamical systems [7] defined *conservative* CA (definition applies to any automaton model as depicted in Fig. 1.5) as those having the property that *no state is a transient (or ephemeral) state*. Both LFSR and NLFSR (Linear or Non Linear Feedback Shift Register) used for long as pseudo-random sequence generators have this property as well. The cellular automata in this paper are conservative and consequently they have no transients;

ii) **A "chaotic" character of the dominant longest cycle**; A very long cycle is not necessarily a random one. A good counter-example is the counting automaton used in the traditional raster scan of images. It has a maximal cycle length $L = N$ but the transition from one state to the consecutive one is rather smooth, often only one bit is changing. As discussed above we are interested in pseudo-random counting automata ensuring consecutive distant "jumps" between the coordinates of pixels. To characterize such behaviours, in [18] we introduced a randomness measure that may be conveniently computed. We are in particular interested on the randomness of the *dominant cycle*. The measure of randomness was defined observing that in a "chaotic" automata the average Hamming distance between consecutive binary vector states (as given by the n cell outputs) becomes $n/2$ instead of 1 for counters.

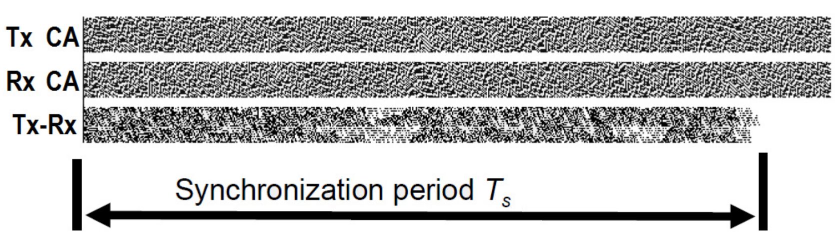

Fig. 1.6 Exemplification of binary synchronization using two identical CA with $n = 31$ cells and rule ID=13474665135 (5 cells neighborhood). Although the initial states of both transmitter (Tx-CA) and receiver (Rx-CA) are different by driving the Rx-CA with 1 single bit from Tx-CA ensures that after a synchronization period T_s the entire state of Rx-CA is identical to the state of Tx-CA.

Therefore, for any arbitrary cycle C_j of length L_j a *scattering coefficient* S_j is defined by averaging the Hamming distances between all consecutive binary vector states in that cycle: $S_j = \frac{1}{nL_j} \sum_{k=1}^{L_j} \sum_{i=1}^{n} |x_i(k) - x_i(k-1)|$ where k is the time index of consecutive states in the cycle j. A *degree of chaos* $\lambda_j = 1 - |2S_j - 1|$ is then defined such that it becomes maximum if $S_j = 0.5$ and zero for the extreme, non-chaotic cases of both fixed points and period 2 cycles (with $S_j = 0$ and $S_j = 1$ respectively). The *degree of chaos* may be regarded as qualitatively similar to the Lyapunov exponent used in continuous-state systems to characterize chaotic behaviours. In our case its largest value is $\lambda_j = 1$ indicates the highest degree of randomness in a finite-length cycle of an automata network;

iii) **Binary synchronization property:** Unlike traditional chaos synchronization [24] in the case of binary synchronization sending only 1 bit from the Tx automaton allows the recovery of the entire state (n bits) in a similar Rx automaton (as seen in Fig. 1.6).

In this case the information needed to resynchronize the Rx is minimal and consists of only 1 bit per clock cycle. Consequently it may be easily embedded and recovered in various forms of modulation/demodulation. This property is not common to CA and it was first investigated for all elementary cellular automata (ECA) in [18]. Within all 256 ECA we found that the binary synchronization property holds only for the conservative rules ID=45 (and its 3 equivalents ID=75,89, and 101). Further work [17] indicates that a precondition to achieve binary synchronization in CA is the existence of an asymmetric cell (i.e. a cell that is sensitive to mirroring the inputs from left to right with respect to the central cell). Also, in order to optimize the dominant cycle length it was found [14] that a hybrid CA model with some of the cells having inverted outputs has a better behavior and allows to design a cellular automaton satisfying all 3 properties mentioned above.

Next, CA or automata systems fulfilling all the above three properties will be mentioned as “good chaotic counters”. As seen in the next subsection, expanding the neighborhood to 5 cells allows the identification of more CA rules holding the

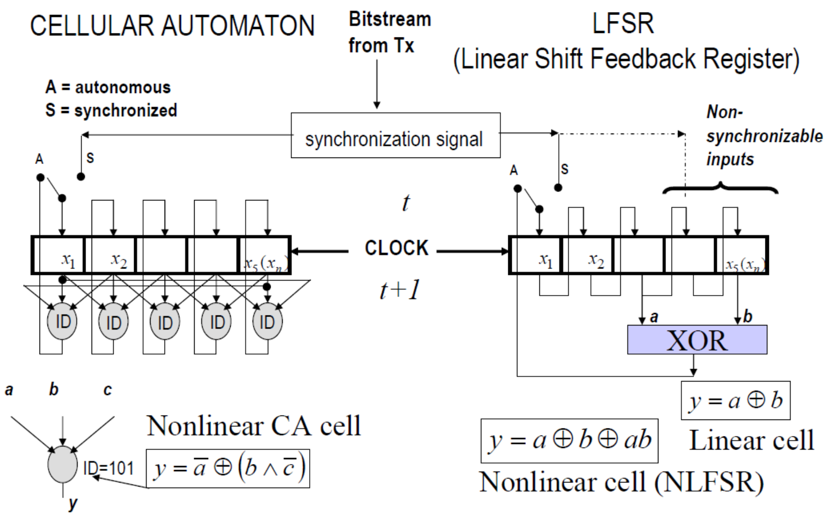

Fig. 1.7 Structure of HCA (left) and LFSR or NLFSR (right) chaotic counters. Unlike HCA, binary synchronization can be used with LFSR/NLFSR only for the cells in the shift-register area, i.e. without cryptographic protection. In the case of HCA cryptographic protection is ensured since synchronization is only possible if the the Rx-CA structure (ID, mask vector) is identical for the Tx-CA. Consequently, the key is given by the specific CA structure.

"good chaotic counter" property. The problem of locating good chaotic counters in the very large space of all possible 2^{32} rules for the 5-cell neighborhood CA is a computationally demanding problem, and so far we solved it only for the case of up to 4 inputs cell within a 5-cell neighborhood.

1.2.3 Designing Good Chaotic Counters as Hybrid Cellular Automata

Ordinary chaotic maps (i.e. logistic, tent, etc.) cannot be used as good chaotic counters because their finite computing precision implementations often produce cycles with only a very small fraction of state vectors (each addressing a pixel in the image sensor array) belonging to the counting cycle. Consequently, only a small fraction of the sensing elements will be addressed, compromising the information acquisition process [15]. As discussed above, a hybrid cellular automaton model proved to be very effective in ensuring the properties of a good chaotic counter. To explain the HCA structure let us consider the case $m = 3$ (3 cells neighbourhood). It expands naturally to a 5-cell or larger neighbourhood. Figure 1.7 presents the HCA automaton structure compared to the widely known LFSR. Note that both of them may be operated in either autonomous mode (as it is the case in the transmitter system Tx) or with one input forced by the synchronization signal (as it is the case in the receiving system Rx).

The discrete-time dynamics of the hybrid cellular automata (HCA) is given by the next equation, which applies synchronously to all n cells (a cell is identified by an index $i \in \{1,2,..n\}$):

$$x_i^T(t+1) = m_i \oplus Cell\left(x_{i-1}^T(t), x_i^T(t), x_{i+1}^T(t)), ID\right) \tag{1.1}$$

where the upper index "T" stands for the transmitting CA counter, $\oplus$ is the logical XOR operator and $Cell(u1,u2,u3,ID)$ is a Boolean function with 3 binary inputs ($u1,u2$, and $u3$), also called the CA (local) rule. A periodic boundary condition is assumed i.e. the leftmost cell ($i=1$) is connected to the rightmost one ($i=n$). The binary mask vector $\mathbf{m}=[m_1,m_2,..,m_n]$ can be optimized [14] (so far our programs perform optimization in reasonable time for $n \le 29$) to obtain a maximal cycle length ($r = L/2^n \to 1$). The above equation (1.1) easily extends to larger neighborhoods such as m =*5* by adding 2 additional inputs to the cell, located on the rightmost and leftmost positions (*i-2*, and *i+2*). For any neighborhood the relationship between inputs and the output local CA rule can be characterized in two different ways while conversion functions are available via [30] :

a) Truth-Table (TT) representation: This is the most widely used representation. The rule is characterized by a binary vector $Y = [y_{N-1}, y_{N-2}, ..., y_0]$. Its representation in decimal basis is called a rule identifier (ID). The output y_k is a binary number assigned to the cell's output when its inputs ordered as a binary vector $[u_n, u_{n-1}, .., u_1]$ are the binary representation of k;

b) Algebraic Normal Form (ANF): This form is described by a binary vector $C = [c_0, c_1, ...c_N]$ (using the method in [30] a unique conversion from Y to C and vice-versa exists) such that its coefficients are multipliers of an algebraic representation on the GF2 exemplified next for the case of m=3 neighborhood:

$$y = c_0 \oplus c_1u_1 \oplus c_2u_2 \oplus c_3u_2u_1k_3u_3 \oplus c_4u_3 \oplus c_5u_3u_1 \oplus c_6u_3u_2 \oplus c_7u_3u_2u_1 \tag{1.2}$$

Note that in general (for any size m of the neighbourhood) c_k is the multiplier of a product (logical AND) of all input variables in a binary vector $[u_n, u_{n-1}, \ldots, u_1]$ corresponding to 1 in the associated binary vector representing k. For example, in the case of $m=3$ for $k=5=101_2$ only the inputs corresponding to 1 in the input string $u_3u_2u_1$are selected to be multiplied resulting in the term $c_5u_3u_1$.

The most important results of our research in designing HCA that are good chaotic counters are summarized in Figure 1.8:

i) First, we have **a number of HCA with 3 inputs** where the ANF representation of the rules is given by the formula: $y = m_i \oplus u_a \oplus u_b \oplus u_cu_b$ where a,b,c are cell position indexes such that $a-b=b-c=h$ where h is an integer. In all these cases a computationally intensive program is run to find the best mask (the one maximizing the dominant cycle length L). For instance the HCA with rule ID=1347465135 has the best mask found so far 19801 (decimal representation of mask vector $\mathbf{m}=[m_1,m_2,..,m_n]$) leading to L =*N-1=131071* in the case of n =17 cells (i.e. addressing an image of size 512×1024). The problem with all HCA in the above category is that they are conservative (no transient and possible to optimize

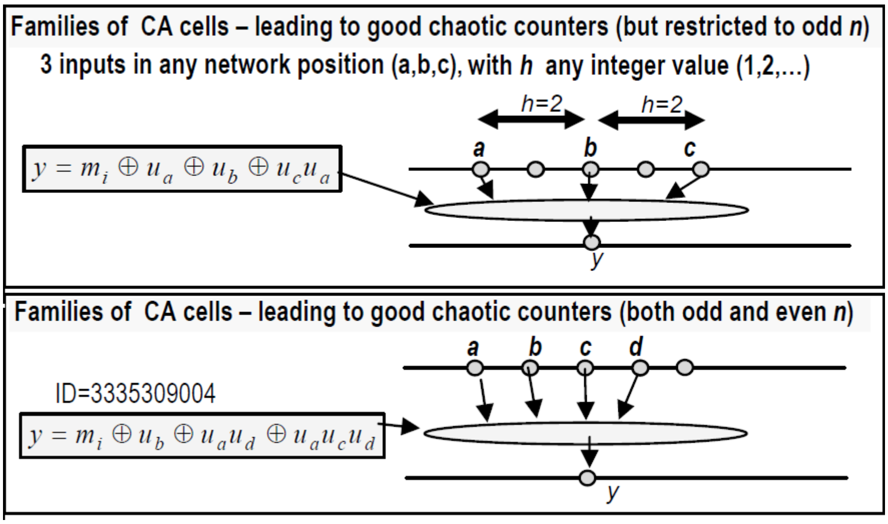
Families of CA cells – leading to good chaotic counters (but restricted to odd n)
3 inputs in any network position (a,b,c), with h any integer value (1,2,...)
h=2
h=2
a
b
c
$y = m_i \oplus u_a \oplus u_b \oplus u_c u_a$
y
Families of CA cells – leading to good chaotic counters (both odd and even n)
a
b
c
d
ID=3335309004
$y = m_i \oplus u_b \oplus u_a u_d \oplus u_a u_c u_d$
y

Fig. 1.8 Cell rules in a 5-cell neighbourhood in order to obtain good HCA chaotic counters

for the longest cycle) only for an *odd* number of cells n; This will make them useful in image and video only for sensing arrays with an aspect ratio of 2:1.

ii) Recently we were able to locate a good counting automaton within the class of 4 inputs positioned in a 5-cell neighbourhood. This automaton (ID=3335309004) is **conservative for any number of cells, including even ones**, as required by the commonly used square image sensors. A near-optimal mask for the case $n = 18$ (i.e. addressing a 512×512 pixels image) is 17855. For this mask $L = 261990 = N - 154$. The loss of 154 pixels is insignificant for such a compression scheme where the number of scanned samples $K << N$

1.2.4 Message Recovery and Examples

It is assumed that in order to reconstruct the image **Y** a counting automaton similar to the one used in the measurement process is available. Also, it is operated in synchrony with the one available in the Tx unit. They would either start from the same initial state or they may be synchronized using the binary synchronization property. The recovery counting automata follows the same dynamics as in the measurement process during the scanning of the$K$ received samples.

The reconstruction algorithm is:

```
RESET the counting automaton
INITIALIZE all pixels in Y with 0.5;
FOR $k=1,..K$
$i$=rand_i; $j$=rand_j;
```

$$\mathbf{Y} = \mathbf{B}_{i,j} s(k) + \mathbf{Y} \circ (\mathbf{1} - \mathbf{B}_{i,j}) \tag{1.3}$$

```
END
```

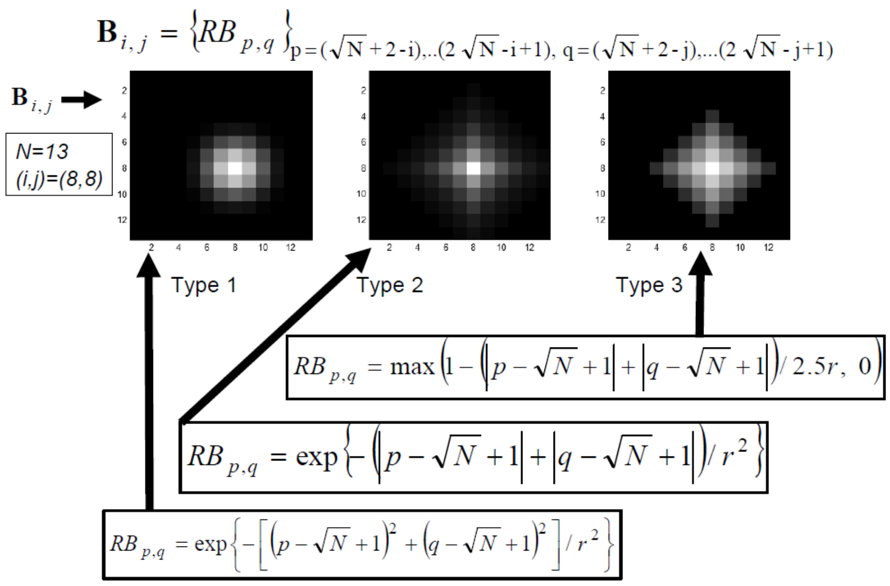
$\mathbf{B}_{i,j} = \{RB_{p,q}\}_{p=(\sqrt{N}+2-i),..(2\sqrt{N}-i+1),\ q=(\sqrt{N}+2-j),...(2\sqrt{N}-j+1)}$
$\mathbf{B}_{i,j}$
N=13
(i,j)=(8,8)
Type 1
Type 2
Type 3
$RB_{p,q} = \max\left(1-\left(\left|p-\sqrt{N}+1\right|+\left|q-\sqrt{N}+1\right|\right)/2.5r,\ 0\right)$
$RB_{p,q} = \exp\left\{-\left(\left|p-\sqrt{N}+1\right|+\left|q-\sqrt{N}+1\right|\right)/r^2\right\}$
$RB_{p,q} = \exp\left\{-\left[\left(p-\sqrt{N}+1\right)^2+\left(q-\sqrt{N}+1\right)^2\right]/r^2\right\}$

Fig. 1.9 Various sliding bases and their associated radial basis functions

In the above, **1** is a square matrix with all unity elements, both **Y** and $\mathbf{B}_{i,j}$ are matrices with the same size as the original image **X** and $\circ$ denotes element wise matrix multiplication (i.e. if $\mathbf{C} = \mathbf{A}\circ\mathbf{B}$ then $c_{i,j} = a_{ij}b_{ij}$). Consequently the recovery process is an iterative process given by recursive equation (1.3), with a total of at most $2N$ multiplications (using a pice-wise linear basis function, the number of multiplications can be further reduced). The *sliding basis* $\mathbf{B_{i,j}}$ is computed using a radial basis kernel inspired from [20]. The detailed formulae and an example for 3 types of basis functions are given in Figure 1.9.

The sliding basis implements the equivalent of a fuzzy membership function with a maximal value 1 on the position (i, j) of the current measurement $(d = 0)$. Neighboring pixels at distance d will correspond to the decaying amplitude of the kernel function according to a *radius parameter* r to be determined. Such a radial basis function is necessary to reconstruct the neighboring pixels that were not sampled during the measurement process and it is expected to be in relationship with the correlation model of the signal **X**.

In terms of recovery performance we estimate the PSNR (measured in dB) of the reconstructed image **Y** (in this case the noise is the difference between original and the reconstructed image) as a function of K. Such a curve may be also considered as a rate-distortion curve since the compression rate is proportional to K for a given N.

In the following, the "Lena" image with $N = 256 \times 256$ samples (pixels) is considered. It was found that the choice of the basis function (among the 3 types mentioned in Fig. 1.9) has little influence on the PSNR value. Consequently, in order to reduce the computational effort, the type 3 basis (a piecewise-linear approximation of the Gaussian) is the most suitable. The type-3 basis is 0 almost everywhere except $(2r+1)^2$ pixels. Consequently the number of multiplications required in equation (1.3) reduces from $2N$ *to only* $2(2r+1)^2$.

The pre-computing of RB and its storage allows to avoid calculations of $B_{i,j}$ for each new measurement position as the counter automaton advances. In the case of a fixed radius (e.g. choosing $r = 2$, for a similar reconstruction error as in [4]) the number of multiplications in the recovery algorithm is of the order of 10^6 for $K = 20000$ (easily computed in less than 0.1 second on actual computers). For reference, in [2] where an improved, highly effective reconstruction algorithm is considered, more than 100 seconds are reported for a similar number of measurements. This comparison indicates a significant speed-up of our image scan method with several orders of magnitude, not only in the measurement but also in the reconstruction phase, when compared to compressive sensing.

The Influence of Radius *r*: As expected, the radius r has a significant role on the quality of the reconstructed image. It corresponds to L in the naïve reconstruction scheme discussed in Fig. 1.4. As seen in Figure 1.10, for a given r the PSNR improves almost linearly up to a value K, then it saturates. In order to avoid saturation r must be smaller. But on the other hand, a small r is in contrast with a small K because the sliding basis cannot provide good reconstruction of all pixels located in the neighborhood of the measurement. The reconstructed image will look noisy (as it happens for $r = 1.5$ in Fig. 1.10) but will improve when $r = 3$. Consequently, it follows that r must adapt to the value of K. Based on experiments we propose the following formula: $r(K) = log_2(N) - log_2(K)$. Such a value may be used as a fixed one in the recovery scheme but an adaptive radius is also possible, for instance by updating the radius value any time$K$ becomes a power of 2. Experiments with both schemes revealed no major quality differences in the reconstructed image, except that the adaptive radius scheme would lead to more computational effort (recalculating the sliding basis each time the radius value changes).

Note that in the above experiments $K = 5000$ is about 7.6% of all pixels in the original image corresponding thus to a compression rate of about 13 times or a rate of 0.61 bpp (bits per pixel) assuming a coding of each pixel with 8 bits. In order to provide a comparison with compressive sensing approaches, the same medical image as in [4] is considered. While our method leads to a slight degradation in the PSNR when compared to the above mentioned compressive sensing method (PSNR = 24,1 dB for ours, instead of 26.5 dB in [4] for K= 10000 measurements) the visual quality of reconstructed images is rather similar (Figure 1.11). Such a slight degradation is acceptable given the important reduction in the overall complexity, particularly the complexity of the sensing device.

Further improvements (i.e. obtaining a better PSNR for a given K) are expected, at the expense of computational complexity, by introducing novel reconstruction sliding bases, tailored to the nature of the images to be compressed. Though, given the very low complexity of the compression stage such a method is suitable for embedding in various smart sensing devices with low power consumption

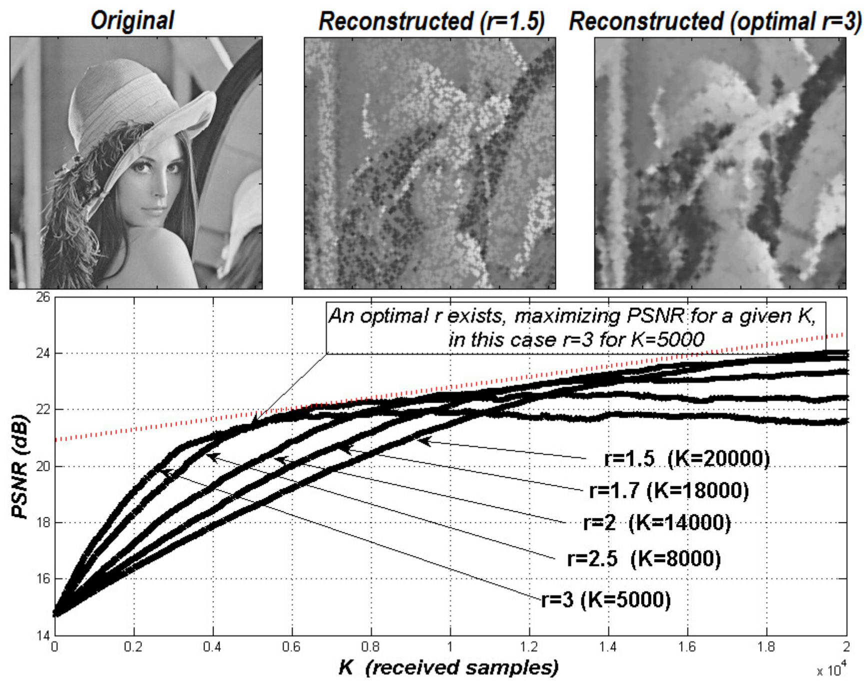

Fig. 1.10 Influence of radius r and K on the quality of the reconstructed image; Here the basis function is of type 3 (piecewise-linear) having the lowest computational complexity

requirements. The method can be applied for video sequences as well, and recent work [29] suggests that it can help reduce the complexity of the motion detection algorithms and may have other benefits given by the simplicity of adjusting the scanning rate depending on the presence or absence of motion (low rate or small K per frame for still or almost still images and higher rates when motion occurs). Furthermore, given the longest cycle property one still has the choice to reconstruct the original image without loss. In all cases (for any degree of compression) the chaotic scan introduces a form of encryption (a key associated with the mask and the ID of the cellular automaton must be known at the Rx unit, in order to properly recover the original message) for free.

1.3 Image Compression Based on Dictionaries Generated by Cellular Automata (CA-VQ)

1.3.1 The General Framework of Dictionary Based Compression

The main idea in this case is to use a properly designed cellular automaton to generate a codebook formed of a number D of *m-sized* binary vectors. The compression

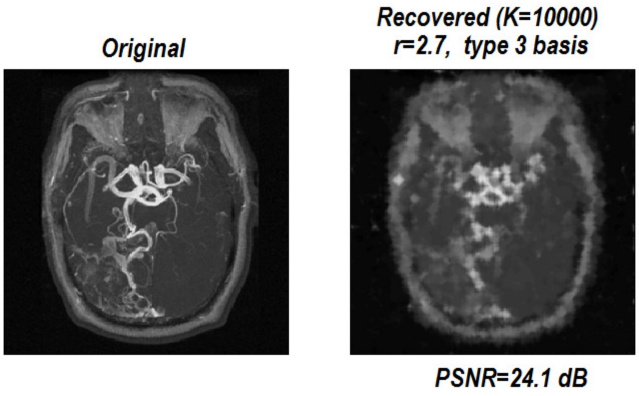

Fig. 1.11 The result of chaotic scan for $K = 10000$ and optimal radius $r = 2.7$

of the original message is done on a certain number b of bit-planes from the original message (e.g. an image).

Within this section we consider the message signal to be identical to the original one i.e. $K = N$. For each block, a search through the codebook reveals the best match (associated to a label encoded with $\log_2 D$ bits) and consequently in the decoding stage (assuming the existence of the same codebook in the Rx unit) the initial block is replaced with one of the D code-vectors in the codebook. Consequently, each bit-plane may be treated independently i.e. algorithm parameters such as the CA rule (ID) generating the codebook, its size D, the block size m, may be optimized such that the bit error (B_{err}) for that specific bit-plane is minimized. In the following b =0 denotes the most significant plane, there is also an option in choosing the exact number b of bit-planes knowing that the most important in term of overall performance (measured as the PSNR of the recovered image with respect to the original one) is bit-plane 0. The compression performance in this case is always expressed in *bits per pixel* (*bpp*). For instance, while using for each b =*6* bit-planes m =*64* (8×8 blocks), D =64, the rate is computed with the following general formula:

$$bpp = (b \log_2 D) / m \tag{1.4}$$

For the above particular values (quite usual with this compression scheme) a 0.56 bpp rate results. An exemplification of the encoding and decoding stages for the CA-based dictionary VQ system is given in Figures 1.12 and 1.13.

1.3.2 Learning CA-Based Dictionaries and Performance Evaluation of the CA-VQ System

As indicated above, the most important aspect in optimizing the CA-based compression scheme is the choice of the CA structure and its rule, since it generates the dictionary. In previous works [21] [11] a guided search through the space of

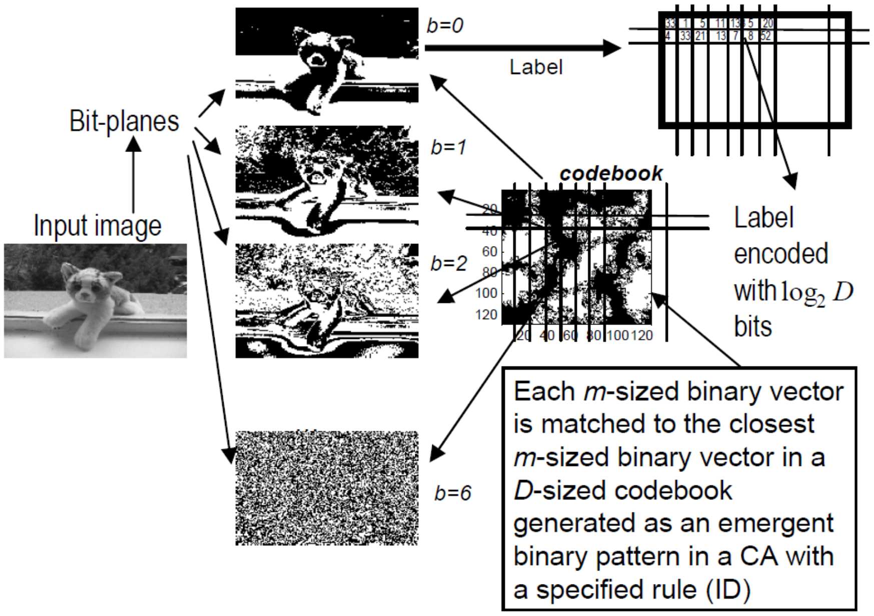
Bit-planes
Input image
b=0
Label
b=1
b=2
b=6
codebook
Label encoded with $\log_2 D$ bits
Each m-sized binary vector is matched to the closest m-sized binary vector in a D-sized codebook generated as an emergent binary pattern in a CA with a specified rule (ID)

Fig. 1.12 Encoding process using CA-generated dictionaries and binary vector quantization

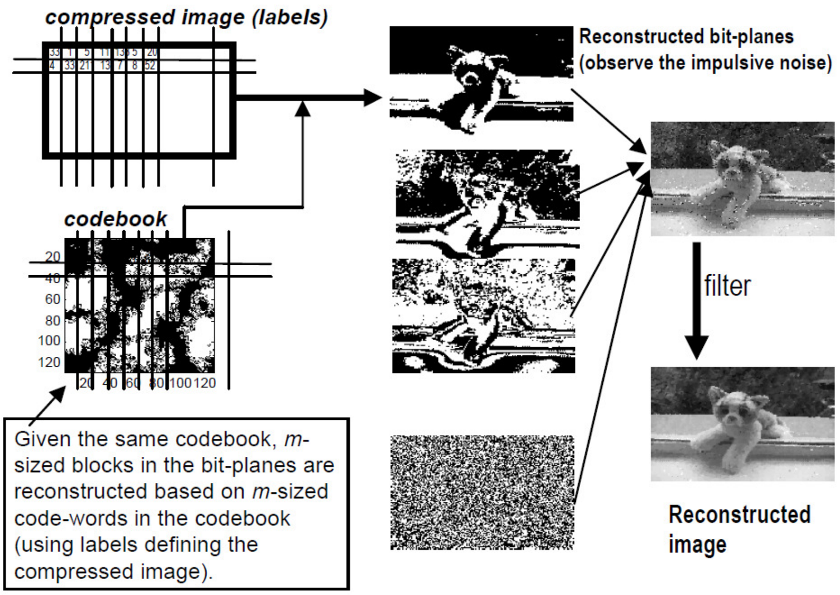
compressed image (labels)
codebook
Reconstructed bit-planes (observe the impulsive noise)
filter
Reconstructed image
Given the same codebook, m-sized blocks in the bit-planes are reconstructed based on m-sized code-words in the codebook (using labels defining the compressed image).

Fig. 1.13 Decoding process using the CA-generated dictionaries

all 1024 outer-totalistic two-dimensional CA with 5-cell (von Neumann) neighborhood was done and several useful ID rules were proposed, maximizing the PSNR for the a given compression rate. In the same work, various block sizes and number of bit-planes were considered, typical values being $D = 32$ or 64 and $m = 64$ to 256 (for very high compression rates). Comparing our compression method to JPEG led to the conclusion that CA-VQ gives better performance than JPEG for high compression rates ($bpp < 0.3$) such compression rates being also associated with faster computation.

Herein we present a different approach for choosing the rule of a CA codebook suitable for compressing a class of images. Unlike in the previous approaches, the local rule is no more restricted to outer-totalistic and although the same von-Neumann neighborhood is used, the space of possible rules is now extremely large (2^{32} rules). Moreover, the CA rule can be individually tuned per each bit-plane.

The CA-based dictionary has the following structure: Given m and D (usually both are powers of 2) the array size M of the 2-dimensional $M \times M$ square CA with periodic boundary condition is determined as $M = \sqrt{Dm}$. For instance, if $D = 64$ and $m = 64$, the size of the dictionary CA is $M = 64$. The codebook is composed of all square D blocks of m size each cropped from the emergent pattern obtained in this CA after a certain number T of iterations when the initial state is an arbitrary (but known at Rx) random binary state with equal number of bits in 0 and 1. The initial state, the rule ID, and the number of iterations T may be regarded as a key of the compression scheme providing a rudimentary form of encryption in addition to the basic compression functionality.

The CA rule is obtained from a simplified learning process using a certain bit-plane image (binary image) of arbitrary size. The original bit-plane image is perturbed using a uniform distribution i.e. a percentage α of its bits are flipped (1 becomes 0 and vice-versa) resulting in an input image for the 5 inputs CA cell (a sliding 5-cell neighborhood is scanning all N pixels of the image as inputs, while the central cell from the associated image is taken as the desired output. For each of the 32 possible 5-bit entries (each encoded as an integer i in the associated truth table) two numbers are stored: n_0^i indicating the number of times the desired output was 0 and n_1^i the number of times the desired output was 1. For N_i occurrences of the input code i it follows that $n_0^i + n_1^i = N_i$. Finally each output for the line i of the truth table is assigned 1 if $n_1^i > N_i/2$ and 0 else. For the rare cases with $N_i = 0$, 1 or 0 is picked randomly with a probability 0.5 as the corresponding output in the truth table. The resulting truth table is then associated with the cell ID, as shown in Fig. 1.14, where $\alpha = 0.12$ was optimized such that the recovery scheme will minimize the bit error on the most significant bit-plane. Given a choice for m and D, the CA codebook is generated by running the CA with the previously determined ID for a certain number of iterations.

The above strategy was motivated by the goal to have a cell ID such that the resulting CA will converge to a stationary pattern preserving most of the relevant details in the original bit-plane. A noisy input was found necessary in order to ensure the diversity of input codes (5-bit words) necessary to construct the associated truth table.

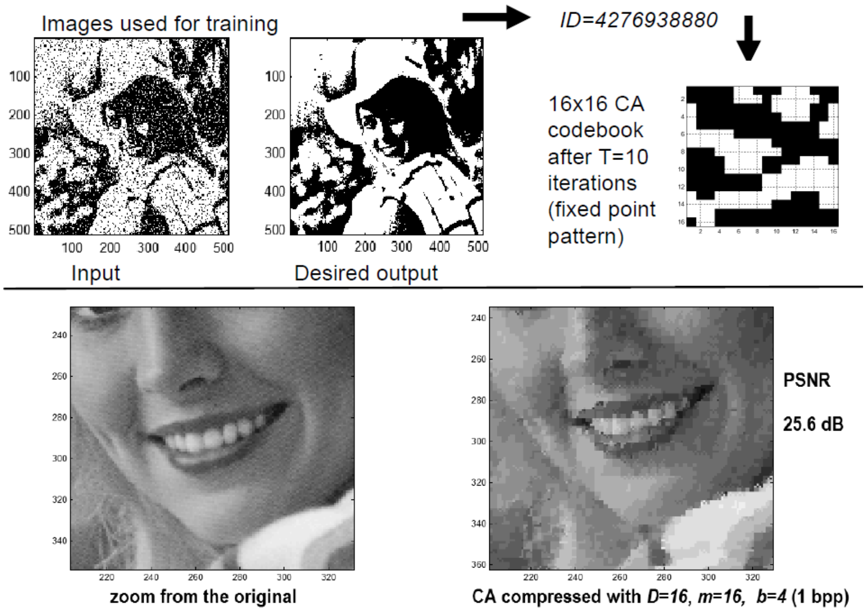
Images used for training
ID=4276938880
16x16 CA codebook after T=10 iterations (fixed point pattern)
Input
Desired output
PSNR
25.6 dB
zoom from the original
CA compressed with D=16, m=16, b=4 (1 bpp)

Fig. 1.14 Generating a CA codebook via a learning mechanism and its use in a CA-VQ compression scheme

As seen in Fig. 1.14, for a particular choice of the codebook size $D = 16$ and block size $m = 16$ (4×4 window) the running of the 16×16 CA with ID=4276938880 for $T = 10$ iterations would reveal the codebook. A zoom of both the original image (512×512 pixels) and the recovered one using this codebook for the most significant *4* bit-planes shows the typical losses of this compression scheme. The PSNR is about 25 dB and the quality of the image is acceptable given the rate of 1 bit per pixel. Further optimization of performance is still possible, for instance the use of different, optimized codebooks, for each bit-plane, or a different learning scheme.

In terms of computational complexity, the CA-VQ approach requires more computational effort in the encoding stage and less effort in the decoding stage. It is an opposite situation from the case of chaotic scan compression. For a message with N bits and a particular bit rate (codebook size D and block size m) assuming that the codebook is calculated and stored, the compression process requires comparisons (between blocks and code-words in the codebook) being more effective for large block sizes and small dictionaries (also ensuring highest compression but lowest reconstruction quality). The above compares not so favorably to simply reading $K < N$ samples in the chaotic scan method. But on the other hand, the complexity of the CA-VQ compression process still remains linear in the number of pixels, being much lower than for traditional compression methods. The decompression stage in CA-VQ is m times faster (i.e. N/m operations of reading the codebook) and involves

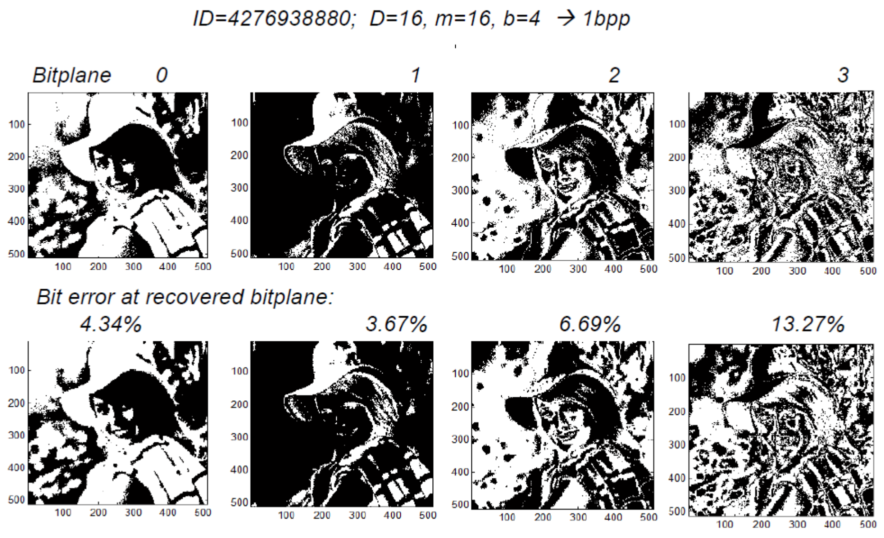

Fig. 1.15 Distribution of bit errors on each of the 4 bit-planes used in the CA-VQ compression scheme for the particular example shown in Fig. 1.14

no comparison. This situation compares favorably to the relatively large number of operations (multiplications) required by the basis-based recovery scheme of the chaotic scan method.

1.4 Hardware Description and Synthesis of CA Using Algebraic Normal Form

In the above, two different image (or video) compression approaches were proposed, both based on Cellular Automata (CA). Since CA is an important part of both algorithms, it is important to find convenient and efficient ways to implement them in various technologies. While usual PCs or microcontrollers may be used to implement the cellular automata, maximal speed and efficiency is achieved when CA models are implemented in a fully parallel fashion. Particularly, when the aim is of implementing the above algorithms as part of a low-power smart sensing device it is of interest to provide CA code for hardware description languages.

In [9] we provide a convenient methodology to generate VHDL representations using the Algebraic Normal Form conversion discussed previously. The result would be a fully integrated sensor system with encryption, compression and other capabilities, as discussed in [19]. Such a sensor would perform compressed sensing using a different, computationally more efficient approach.

A detailed description of our FPGA implementations is also given in [10]. Specific to all of them is the development of a software module, called next CA-description module and written in C++ to automatically generate the VHDL code used for the usual synthesis steps. The same code may be used for FPGA designs as well as for dedicated VLSI chips. The CA-description module inputs a description of the CA (neighborhood size, rule, mask, number of cells, etc.) in a user-friendly manner and generates VHDL modules to be used in various hardware implementations of the above mentioned algorithms. The key issue in generating the most important part of the VHDL code is the very good correspondence between the ANF description and the possibility to express it in a few VHDL line codes. A particular example is give next: A one-dimensional CA is defined as having 7 cells, a certain mask vector (1100010) and ID=101. The resulting VHDL line describing the entire HCA is:

REG<= “1111111” xor c xor a xor (b and a) xor mask;

The above corresponds to the following particular form of the ANF representation:

$$y = 1 \oplus u_3 \oplus u_1 \oplus u_2 u_1 \tag{1.5}$$

The above variable REG represents the entire CA array and the variables *a,b,c* are constructed to represent shifted versions of REG according to specific neighborhood to be implemented.

So far the CA-description module can implement either 3-cell neighborhoods or 5-cell neighborhoods for the HCA model (the traditional, homogeneous CA model is a particular case of HCA with all 0 elements in the mask vector). Various FPGA target devices were considered and in all cases the resulted implementation was found to be very efficient. For instance, in the case of Xilinx FPGA’s 1 LUT was assigned for HCA designs with up to 3 inputs and mask vector while 2 LUTs suffice to implement an entire cell (including its local memory) in the maximal case considered so far of 5 cell neighborhoods. Similarly, for FPGA devices from Altera (Cyclone II EP2C35F672C6 device on the DE2 board provided by the University Program) one basic computational unit (LE – logic element) is assigned for 3-inputs HCA cells and 2 times more for the case of 5-inputs. Note the very efficient allocation of one cell per FPGA logic register. The above results confirm that cellular automata with very large number of cells ($n = 33216$ in the case of the chip on the DE2 board) can be easily realized in low cost series FPGA. The same VHDL description may be used to generate part of specialized sensor chips (e.g. in addition to low power image sensors e.g. [22]) using an ASIC design flow.

References

1. Baraniuk, R., Cevher, V., Duarte, M., Hedge, C.: Model-based compressive sensing. IEEE Trans. Inform. Theory 56, 1982–2001 (2010)
2. Baron, D., Sarvotham, S., Baraniuk, R.G.: Bayesian compressive sensing via belief propagation. IEEE Trans. Signal Process. 58, 269–280 (2010)
3. Bruckstein, A.M., Holt, R.J., Netravali, A.N.: Holographic representation of images. IEEE Trans. Image Processing 7, 1583–1597 (1998)
4. Candes, E., Romberg, J.: Practical signal recovery from random projections. In: Proc. SPIE Conf., Wavelet Applications in Signal and Image Processing XI (2005)

5. Capellari, L., Milani, S., Cruz-Reyes, C., Calvagno, G.: Resolution scalable image coding with reversible cellular automata. IEEE Trans. on Image Processing 20(5), 1461–1468 (2011)
6. Chen, R.J., Lai, J.L.: VLSI implementation of the universal 2-D CAT/ICAT systems. In: Proceedings of the 11th IEEE International Conference on Electronics, Circuits and Systems (ICECS), pp. 187–190 (2004)
7. Chua, L.O.: A nonlinear dynamics perspective of Wolfram's New Kind of Science (vol. I-IV). World Scientific Series on Nonlinear Science, Series A, vol. 57,68,76. World Scientific Publishing Company (2011)
8. Das, D.: A survey on cellular automata and its applications. In: Krishna, P.V., Babu, M.R., Ariwa, E. (eds.) ObCom 2011, Part I. CCIS, vol. 269, pp. 753–762. Springer, Heidelberg (2012)
9. Dogaru, I., Dogaru, R.: Algebraic normal form for rapid prototyping of elementary hybrid cellular automata in FPGA. In: Proceedings of the ISEEE Conference, Galati, Romania (2010)
10. Dogaru, I., Dogaru, R., Damian, C.: FPGA implementation of chaotic cellular automaton with binary synchronization property. In: Proceedings of 8th International Conference on Communications (COMM), Bucharest, Romania (2010)
11. Dogaru, R.: CA-VQ: A simple compression scheme using codebooks generated by cellular automata. In: Proceedings NSIP 2007 (International Workshop on Nonlinear Signal and Image Processing), Bucharest, Romania (2007)
12. Dogaru, R.: Systematic design for emergence in cellular nonlinear networks with applications in natural computing and signal processing. SCI, vol. 95. Springer, Heidelberg (2008)
13. Dogaru, R.: A fast method for classification of emergent dynamics in cellular automata based on uncertainty profiles. Journal of Control Engineering and Applied Informatics 11, 18–25 (2009)
14. Dogaru, R.: Hybrid cellular automata as pseudo-random number generators with binary synchronization property. In: Proceedings of the International Symposium on Signals Circuits and Systems (ISSCS 2009), Iasi, Romania (2009)
15. Dogaru, R.: HCA101: A chaotic map based on cellular automata with binary synchronization properties. In: Proceedings of the 8th Int. Conference on Communications (COMM 2010), Bucharest, Romania (2010)
16. Dogaru, R.: A low complexity image sensing method using pseudo-random scan and recursive reconstruction with radial basis functions. In: Proceedings of the ISEEE Conference, Galati, Romania (2013)
17. Dogaru, R., Dogaru, I.: Uncertainty profiles for predicting complex nonlinear dynamics in cellular automata: the case of five cells neighborhood. In: Proceedings of the International Symposium on Nonlinear Theory and its Applications (NOLTA 2010), Krakow, Poland, September 5-8 (2010)
18. Dogaru, R., Dogaru, I., Kim, H.: Binary chaos synchronization in elementary cellular automata. Int. J. Bifurcation Chaos 19, 2871–2884 (2009)
19. Dogaru, R., Dogaru, I., Kim, H.: Chaotic scan: A low complexity video transmission system for efficiently sending relevant image features. IEEE Trans. on Circuits and Systems for Video Technology 20, 317–321 (2010)
20. Dogaru, R., Murgan, A.T., Ortmann, S., Glesner, M.: A modified RBF neural network for efficient current-mode VLSI implementation. In: Proceedings Micro-Neuro 1996, February 12-14, pp. 265–270. IEEE Press, Laussane (1996)

21. Dogaru, R., Tetzlaff, R., Glesner, M.: Semi-totalistic CNN genes for compact image compression. In: The IEEE Proceedings of 10th International Workshop on Cellular Neural Networks and Their Applications, Istanbul, Turkey (2006)
22. Fernandez-Berni, J., Carmona-Galan, R., Carranza-González, L.: FLIP-Q: A QCIF resolution focal-plane array for low-power image processing. IEEE Journal of Solid-State Circuits 46, 669–680 (2011)
23. Hinton, G.E., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science 313, 504–507 (2007)
24. Kolumban, G., Kennedy, M.P., Chua, L.O.: The role of synchronization in digital communication using chaos-part ii: Chaotic modulation and chaotic synchronization. IEEE Trans. Circuits and Syst. I 45, 1129–1140 (1998)
25. Lafe, O.: Data compression and encryption using cellular automata transforms. Engng. Applic. Artif. Intell. 10, 581–591 (1997)
26. Lee, J., Verleysen, M.: Nonlinear dimensionality reduction, 3rd edn. Springer (2007)
27. López-Mancilla, D., Cruz-Hernández, C.: Output synchronization of chaotic systems: Model-matching approach with application to secure communication. Nonlinear Dynamics and Systems Theory 5, 141–156 (2005)
28. van der Maaten, L., Postma, E., van den Herik, J.: Dimensionality reduction: a comparative review. Tilburg University Technical Report TiCC-TR 2009-005 (2009)
29. Mitrea, C., Ionescu, B., Dogaru, R.: A pseudo-random scan perspective to the motion detection paradigm. In: Proceedings of the ISEEE Conference, Galati, Romania (2013)
30. Ronjom, S., Abdelraheem, M., Danielsen, L.E.: TT and ANF representations of boolean functions. Online database of Boolean functions (2013), `http://www.selmer.uib.no/odbf/help/ttanf.pdf` (last accessed September 28, 2013)
31. Wolfram, S.: Random sequence generators. US Patent 4691291A (1987)
32. Zipf, P., Hinkelmann, H., Shao, H., Dogaru, R., Glesner, M.: An area-efficient FPGA realisation of a codebook-based image compression method. In: Proceedings of FPT 2008, pp. 349–352 (2008)