Evolution of Cellular Automata with Conditionally Matching Rules for Image Filtering

Michal Bidlo
Brno University of Technology
Faculty of Information Technology
IT4Innovations Centre of Excellence
Božetěchova 2, Brno, Czech Republic
bidlom@fit.vutbr.cz

***Abstract*—We present an evolutionary method for the design of image filters using two-dimensional uniform cellular automata. Specifically, a technique called Conditionally Matching Rules is applied to represent transition functions for cellular automata working with 256 cell states. This approach allows reducing the length of chromosomes for the evolution substantially which was a need for such high number of states since the traditional table-based encoding would require enormous memory space. The problem of removing Salt-and-Pepper noise from 8-bit grayscale images is considered as a case study. A cellular automaton will be initialised by the values of pixels of a corrupted image and a variant of Evolution Strategy will be applied for the design of a suitable transition function that is able to eliminate the noise from the image during ordinary development of the cellular automaton. We show that using only 5-cell neighbourhood of the cellular automaton in combination with conditionally matching rules the resulting filters are able to provide a very good output quality and are comparable with several existing solutions that require more resources. Moreover, the proposed evolutionary method exhibits a high performance which allows us to design filters in very short time even on a common PC.**

***Index Terms*—Evolution strategy, cellular automaton, conditional rule, image filter, salt-and-pepper noise.**

I. Introduction

Noise elimination from digital images represents a typical low level image processing task [1]. It often represents a significant step in the image pre-processing before performing advanced algorithms such as image segmentation, recognition or classification. Usually the noise elimination has to be implemented by means of non-linear functions (referred to as non-linear image filters) because the noise is inherently non-linear. Therefore, it is not possible to apply mathematical theories known from linear filters, which leads to mostly experimental design of non-linear filters.

This work was supported by Czech Science Foundation project GA19-10137S, by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project IT4Innovations excellence in science - LQ1602" and by the IT4Innovations infrastructure which is supported from the Large Infrastructures for Research, Experimental Development and Innovations project IT4Innovations National Supercomputing Center - LM2015070".

A. Conventional image filter design

In conventional approaches, the image filter typically consists of two parts: a noise detector and a filtering algorithm. In such case, the filtering algorithm is executed only if the noise detector evaluates a given part of image as a noise. Otherwise, the input pixel is sent to the filter output unmodified. This approach allows decomposing the filter design into simpler parts which are then solved separately [1]. On the other hand, some approaches consider the filter as a "black box" that takes image data as inputs, implements a suitable function for the noise elimination and produces a filtered pixel as its output. This approach showed to be very powerful for evolutionary image filter design which will be mentioned in Section I-B. Conventional non-linear image filters of this kind are mostly based on the calculation of median out of the pixel values belonging to a given *filtering window* (usually of size 3x3 or 5x5 pixels). These techniques may be used for comparison with advanced (experimentally designed) filters.

The basic *median filter* (MF) is a special case of order statistic filters [2]. They can be implemented effectively in hardware using the concept of sorting networks as described in [3]. The *weighted median filter* (WMF) represents an extension of this concept, which assigns weights to some values within the filtering window. The *center weighted median filter* (CWMF) [4] is a special case which weights only the central value of the window.

The *adaptive median filter* (AMF) can be considered as a multi-level order statistic filter [5], the goal of which is to detect and subsequently replace corrupted pixels only. Usually, two levels working with the 3×3 and 5×5 filtering window, respectively, are utilised. The filtering is performed in two-phases. In the first phase, a sorting network is used for calculating the minimum, maximum and median value for the pixels inside the filtering window. In the second phase, these values are used together with the value of the original central pixel to decide whether it is affected by noise.

Another MF modification includes the *directional weighted median filter* (DWMF) [6] which utilizes a noise detector

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

that is based on the differences between the central pixel and its neighbours aligned with four main directions (horizontal, vertical and two diagonal). The noise detection is based on calculating the weighted sum of absolute differences between the value of the central pixel and pixels within the given direction and a threshold is used to determine whether the pixel is corrupted by noise and needs to be replaced. This method is applied iteratively (in 5–10 steps) with a decreasing threshold in order to achieve a required output quality.

Finally, let us mention the *pixel-wise MAD* (PWMAD) [7] as an iterative non-linear image filter that addresses the problem of random valued shot noise removal. The noisy pixels are detected using a local variance estimator based on the iterative calculation of the median of the absolute deviations from the median.

There are also other filtering principles that do not strictly utilise the notion of locality (filtering windows), use more complex operations and are therefore suitable for software implementations only (e.g. [1], [8]–[11]).

B. Design of image filters by means of evolutionary algorithms

Evolutionary algorithms [12] have recently been applied in many application domains including image processing. The evolutionary design of image filtering circuits can be found in the area of evolvable hardware methods [13]. Earlier works have dealt with several basic approaches, including the optimization of filter coefficients [14], [15] and stack filter evolution [16]. Currently, the design of image filters at the circuit level is usually performed by means of *Evolution Strategy* using the *Cartesian Genetic Programming* (CGP) representation [17]. A survey of image filter design methods that utilize CGP can be found in [18].

In addition to relatively simple non-linear functions composed of operations such as minimum, maximum, or average [19]–[21], advanced concepts have been investigated, e.g. using *bank of filters* [22], switching concept [23], [24] or fault-tolerant image filter design [25]. Later, the image filter evolution was accelerated using Field Programmable Gate Arrays (FPGAs) [26], [27], graphic processing units (GPU) [28], computer clusters and advanced pre-compilation techniques [29]. An advanced extensive study of evolutionary design of switching filters by means of CGP and their comparison with conventional filters can be found in [30].

C. Image filtering using cellular automata

Cellular automata (CA), originally introduced by John von Neumann in [31], represent a massively parallel computational platform suitable for various applications. The basic structure of a cellular automaton assumes a regular structure of *cells*, each of which at a given moment occurs in a *state* from a finite set of states. The behaviour (or *development*) of a CA is considered as a *synchronous* update of all cells according to a *transition function* in discrete time steps. The transition function determines the next state of a cell depending on the combination of states in its neighbourhood. For the purposes of this paper, *uniform two-dimensional (2D) CA* will be considered in which the cells are arranged into a square lattice and there is a *single* transition function according to which all cells are to be updated. The cellular neighbourhood will be defined uniformly for each cell and will consist of a given cell and its immediate neighbours in the north, south, east and west direction (it is a case of 5-cell neighbourhood also called von Neumann neighbourhood). The task of designing a CA for a given task consists of finding a suitable transition function (possibly with appropriate initial states of cells) in order to achieve a given behaviour performed by the CA development.

Cellular automata have also been applied to image processing tasks (including image and video compression, image resizing, edge detections and others). A wider overview can be found in [32]. Rosin applied a sequential floating forward search method for feature selection and identification of good rule sets for a range of tasks, namely noise filtering (also applied to grayscale images using threshold decomposition), thinning, and convex hulls [33]. Selvapeter and Hordijk introduced several modifications to the standard CA concept and applied them to improve the filtering performance. For example, a random CA rule solves the noise propagation present in deterministic CA filters. A mirrored CA is used to solve the fixed boundary problem. The authors showed that these methods can outperform some standard filtering methods [34]. Diosan et al. investigated various methods suitable for image segmentation by means of CA [35]. Kundra et al. applied CA to detect edges in digital images. The approach is based on 2-fold symmetry that implies 51 patterns from which the best rule for image detection can be derived [36].

The utilisation of the pure CA concept for image processing represents a difficult problem. The existing works are mostly limited to simplified cases (e.g. binary images only) or use the CA as a mean to derive suitable steps in order to perform the image processing algorithm. In this paper we apply basic CA whose state will directly represent an image and use an evolutionary algorithm in order to design a transition function (the filtering algorithm) as described in Section I-D.

D. Goal of this paper

In this paper we use a method for efficient representation of transition functions of CA, called Conditionally Matching Rules (CMRs) [37], and apply a variant of Evolution Strategy (ES) in order to find various transition functions suitable for removing Salt-and-Pepper noise from 8-bit grayscale images. The CA states will directly be represented by 256 possible pixel values of the filtered image. The filtering algorithm will be represented by a single transition function of the CA designed by the ES. The main objective is to demonstrate that using the 5-cell neighbourhood (instead of 3×3 or 5×5 windows) and CMR-based transition functions with only basic relational operators (specifically $=, \neq, \leq, \geq$), which represents significantly reduced resources against many other methods, high-quality image filters can be obtained. Moreover, we show that this approach in combination with the evolution strategy exhibits a very good performance which allows designing filters in very short time even on a common PC.

II. Evolution of cellular automata using conditionally matching rules

Conventionally, the local transition function of a CA is represented by a *table* specifying *transition rules* for any possible combination of states in the cellular neighbourhood. For the concept of 2D CA, introduced in Section I-C, each row of the table contains a rule of the form $s_N\ s_W\ s_C\ s_E\ s_S \rightarrow s_C^{new}$, where s_x is the state of a cell at position x in cellular neighbourhood and s_C^{new} is a new state of the cell in the middle of the neighbourhood for the next time step. However, the size of the table grows exponentially depending on the number of cell states and the size of the cellular neighbourhood which makes the representation and design of complex CA very difficult. Although a subset of rules could be specified to represent a transition function, the problem is how to identify the rules suitable for a given task. In this paper we deal with 256 cell states for which there are in total 256^5 possible rules. Designing a complete transition function in the form of the table for such CA is practically impossible because because its storage would require 1 TB of memory (considering a single rule to be encoded as the next state value represented as 1 byte).

A. Conditionally Matching rules (CMRs)

Conditionally matching rules can be considered as an advanced representation method for transition functions of complex CA. The first extensive study of this concept and its suitability for the evolutionary design of complex CA was proposed in [37] and recently in [38]. The main idea of a CMR is, in general, to represent using a single conditional rule more than one conventional table rules. This way the size of the transition function structure can be reduced substantially.

For the purposes of this paper, a conditional rule (CMR) for a 2D CA working with 5-cell neighbourhood is defined as $(cnd_N\ s_N)\ (cnd_W\ s_W)\ (cnd_C\ s_C)\ (cnd_E\ s_E)\ (cnd_S\ s_S) \rightarrow s_C^{new}$, where cnd_x denote a condition operator and s_x represent a state value. Each CMR thus consists of pairs: *(a condition and a state value)*, that corresponds to (is evaluated with respect to) a specific cell in cellular neighbourhood, and a new state value. For the experiments presented in this paper, the following conditions are considered: $= 0$, $\neq 0$, $\leq s$, $\geq s$, where s denotes an arbitrary state value from range 0 to 255. This means that in case of conditions $=$ and $\neq$ the cell states are always evaluated against state 0 and conditions $\leq$ and $\geq$ allows evaluating against any given state value. This set of conditions resulted from our long-term experimentation with cellular automata and its suitability was proven in several studies (e.g. recently [37], [38], [39]). A finite sequence of CMRs represents a transition function for a CA.

For example, consider a conditional rule $(\neq 0)(\geq 2)(= 0)(\geq 2)(\geq 1) \rightarrow 2$ and let c_N, c_W, c_C, c_E, c_S be cells in states $2, 3, 0, 3, 3$ respectively. The new state of c_C will be determined by evaluating the CMR-based transition function as follows. The CMRs are evaluated sequentially until a rule is found whose all conditions are true with respect to the cell states in the given neighbourhood. Consider the evaluation of the aforementioned conditional rule according to which $c_N \neq 0$ is true because $c_N = 2$, $c_W \geq 2$ is also true since $3 \geq 2$, $c_C = 0$ which satisfies the third condition and the last two conditions hold too because both c_E and c_S possess state 3. All conditions of the CMR were evaluated as true, therefore this CMR is said to match and will be applied to determine the next state of c_C (i.e. $s_C^{new} = 2$ according to the value on the right side of the CMR). If no matching rule in the sequence is found, then the cell keeps its current state. Note that the CMR-based transition functions can be transformed to the appropriate table rules which preserves the original concept of CAs as described in [37]. However, for the CA working with 256 cell states, that we deal in this paper, the table representation is intractable because of enormous memory requirements as mentioned above.

B. Evolutionary design of CMR-based cellular automata for image filtering

In this section the representation, fitness function and evolutionary algorithm will be described.

A sequence of CMRs is represented by an array of integers of a fixed size. Since 5-cell neighbourhood is considered, each CMR consists of 5 pairs of integers (one integer encodes the condition and the other the state value) and a single integer encoding the new state value. Hence in total a single CMR is composed of $5 \times 2 + 1 = 11$ integers. Note that for the set of conditions described in Section II-A, the following integers are considered for their encoding: 0 for $=$, 1 for $\neq$, 2 for $\leq$ and 3 for $\geq$. Let G denote the number of CMRs in a sequence of CMRs. Then each chromosome, encoding a transition function, consists of $G \times (5 \times 2 + 1)$ integers. The goal of the evolution is to find such a transition function according to which the CA behaviour satisfies (or optimises) given criteria.

For the purposes of image filter design, where I_{fil} denotes the corrupted image (i.e. the image to be filtered) and I_{ref} is its reference (uncorrupted) version, both of size $K \times L$ pixels, *mean square error* (MSE) was chosen as the fitness metric. The aim of the evolution is to minimize this criterion, i.e. the lower the MSE, the better the filter. MSE can be expressed as follows:

$$MSE = \frac{1}{KL}\sum_{i=1}^{K}\sum_{j=1}^{L}(I_{fil}[i,j] - I_{ref}[i,j])^2. \qquad (1)$$

In order to evaluate a candidate solution, represented by a CMR-based transition function for a CA, the $K \times L$-cell CA is initialised by the values of pixels of I_{fil}. Then three[1] steps of the CA are performed using the CMRs encoded in the chromosome. The current cell states are evaluated with

[1]Note this value was determined experimentally and is primarily based on the fact that only 5 pixels enter the filter, only a single transition function is considered, hence some intermediate steps may be required in order to eliminate (a substantial part of) the noise from the filtered image; this fully corresponds to the basic concept of uniform CA which allows us to more deeply investigate advanced techniques in the future.

respect to I_{ref} according to (1). The MSE is assigned to the chromosome as fitness value.

For the evolution of CMR-based transition functions, we applied a variant of (μ, λ)-Evolution Strategy algorithm, where μ is the size of parent population and λ is the number of offspring generated from the parent population ($\lambda > \mu$). The evolution works as follows. The initial parent population is generated randomly. The selection for reproduction, introduced for the purposes of this paper, is performed by a "round robin" scheme as follows. The parent chromosomes are selected deterministically one by one, each of which undergoes mutation of j integers, where j is the index of the parent in the population (from 1 to μ). The mutation is performed by randomly selecting j integers of the chromosome and replacing them by new valid randomly generated values. Since $\lambda > \mu$, after selecting the last parent the selection continues again from the first parent (round robin) until λ offspring are created. Then the offspring are evaluated by the fitness, sorted from the best to worst according to their fitness values and the best μ offspring replaces the original parent population. Moreover, the best-so-far solution is recorded during the evolution and replaced in case of any better one is detected. Note that the mutation of j integers, based on the index j of the parent in the (sorted) population, ensures that the best individual undergoes the smallest modification whilst in case of worse individuals more changes are allowed. The evolution repeats until a maximum number of generations are performed, then the best-so-far individual is returned as a result.

III. Experimental results

In this paper we consider the evolutionary design of image filters for filtering Salt-and-Pepper noise as a case study. Note that Salt-and-Pepper noise causes corruption of some pixels of the image by the maximal or minimal value of the pixel value scale. In particular, in case of 8-bit grayscale images, each corrupted pixel possesses either value 0 (i.e. black) or 255 (i.e. white). The intensity of the noise will be represented in percentages. For example, 10% noise of 200×200-pixel image means that 400 randomly selected pixels out of the total 40,000 pixels of the image are corrupted.

A. Objectives of experiments

The experiments conducted in this paper were focused on the following objectives. To demonstrate that the proposed variant of ES in combination with the CMR encoding allows an efficient evolution of high-quality filters with reduced resources of the CA. In particular we show that the evolution is able to provide in a short time image filters that can eliminate (or substantially reduce) Salt-and-Pepper noise of various intensities. Despite the fact that many currently known filters usually work with filtering windows of 3×3 or 5×5 pixels and the filtering algorithms are implemented as complex functions over 8-bit values (e.g. see [30]), in this paper we consider only 5-cell neighbourhood of the 2D CA as the inputs of the filter. Moreover, the training of the filter is performed using a single image which allows keeping a reasonable computational effort. As evident we provide significantly less resources to the design process and claim that even so the ES can provide solutions of high quality. The primary goal here is *not* to overcome the best filters known so far but to propose an alternative scheme (i.e. filtering by means of ordinary cellular automata) and demonstrate its usability.

B. Experimental setup

A wider analysis of ES-based evolution of cellular automata was published in [39] where we identified that the (μ, λ)-ES exhibits a good ability to avoid getting stuck in local optima in the task of designing complex CA. From this analysis we also identified a suitable ES setup for the experiments presented in this paper which is $\mu = 4, \lambda = 8$. For the image filter design, the maximum number of generations will be set to 100,000. Figure 1 shows the image utilised for training (i.e. fitness evaluation of candidate solutions). It is a picture of "Lena"[2] that is often considered for evaluating image processing experiments. The training image was corrupted by 10% noise. Several sets of experiments were conducted with various numbers of CMRs (in particular, we consider 5, 10, 20 and 30 CMRs the transition functions consist of). For each set of experiments 96 independent evolutionary runs were executed on the Salomon cluster[3] equipped by 2 x Intel Xeon E5-2680v3, 2.5 GHz, 12 cores per each computational node.

(a) corrupted by 10% noise (b) reference image

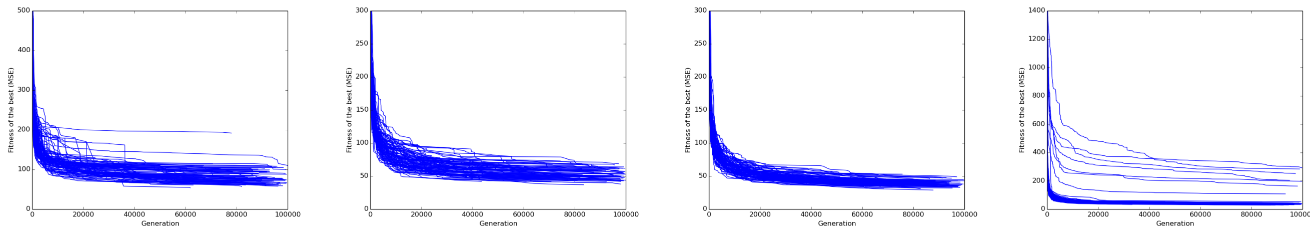
Fig. 1: Training image used for the evolution of image filters.

C. Statistics of evolutionary process

For each considered number of CMRs (5, 10, 20 and 30) 96 filters were obtained from each independent evolutionary run. In order to evaluate the performance of the evolutionary process, we observed the improvements of fitness of the best solution during each run. Figure 3 shows the progress for each run in each set of experiments considering the maximum 100,000 generations. As evident, in most cases there is a rapid improvement of the best fitness during the first 2000 generations and further improvements can be achieved during the rest of the generations for some runs. Therefore, if a filter

[2] https://kasunkosala.wordpress.com/computer-vision-and-digital-image-processing/

[3] see https://docs.it4i.cz/salomon/hardware-overview/



(a) experiments with 5 CMRs (b) experiments with 10 CMRs (c) experiments with 20 CMRs (d) experiments with 30 CMRs

Fig. 3: Progress of improving the best solution during the evolution of independent runs. Note that at the beginngin the fitness was around 6000 but the scale of the limit of the vertical axis was set lower due to illustration purposes. As can be seen, a rapid improvements can be observed in most runs which allows designing (prototype) filters in a couple of minutes).

```
!= 187 == 247 == 161 != 112 == 235 --> 178
==  53 >=  71 == 154 <= 254 <=  49 --> 133
!=  86 >=  68 ==  14 <= 255 >= 107 --> 255
>= 120 !=  85 <=  18 >=  90 >= 113 --> 143
!=  55 !=  49 <=  10 >=  59 >=  77 --> 96
== 203 <=  30 ==  73 >= 198 <=  10 --> 119
>=  43 !=  71 ==  31 >=  74 == 130 --> 247
== 137 >= 246 >=  81 == 207 !=  44 --> 167
!=  30 <=  98 >= 235 >=   1 <= 175 --> 6
>=  52 != 121 <=   6 !=  83 != 252 --> 71
>= 185 !=  49 >= 240 >= 182 != 124 --> 207
>= 222 != 191 <=   4 <= 157 >= 123 --> 0
<= 115 != 105 >= 248 <= 148 != 124 --> 4
>=   9 != 242 <=   6 != 104 != 142 --> 31
!= 109 >=  67 >= 242 != 212 >=  48 --> 160
<=  52 != 196 ==  39 <= 166 !=  48 --> 2
!= 134 >= 158 ==  83 <=  62 ==  60 --> 250
>=  97 != 248 == 158 ==   5 == 196 --> 199
!= 159 == 109 == 244 !=  19 !=  36 --> 254
<= 152 <= 159 >= 160 <= 145 <= 166 --> 133
```

Fig. 2: The evolved CMR representation of the F89 filter

needed to be designed very quickly, it would be possible to shorten the evolution time substantially by reducing the number of generations. We measured that such runs can be performed on a common PC or modern laptop in order to obtain a filter of average quality within several tens of seconds in average (or up to a few minutes for larger numbers of CMRs). This is particularly true for experiments with 10 and 20 CMRs in which we identified the best filters out of all experiments. This aspect represents one of the contributions of this paper – a possibility to obtain filters of a reasonable quality in very short time. Note that in [30] the average computation time on Xeon X5670, 2.93 GHz using an optimised CGP implementation and 250,000 generations was about 4 hours.

D. *Evaluation of evolved filters*

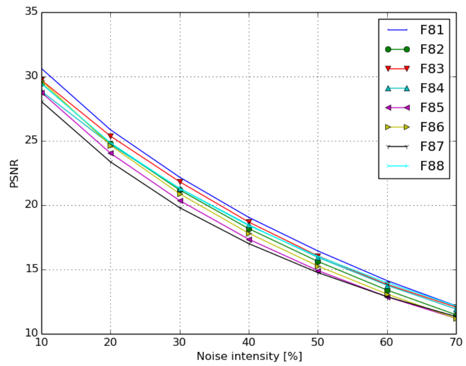
In order to evaluate the evolved filters, the Peak Signal-to-Noise Ratio (PSNR) was calculated for various images and noise intensities according to (2) which represents a common metric for the evaluation of image quality (the higher the PSNR, the better the filter and image quality).

$$PSNR = 10\ \log_{10} \frac{255^2}{\frac{1}{KL}\sum_{i=1}^{K}\sum_{j=1}^{L}(I_{fil}[i,j] - I_{ref}[i,j])^2} \tag{2}$$

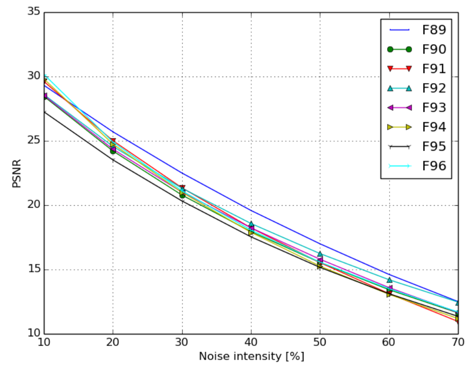
After training the filters during evolution we used a set of 24 test images from Vašíček's database[4] corrupted by the noise from 10% to 70% for the evaluation of obtained results. Figure 4 shows analysis of filters obtained from selected evolutionary runs in each set of experiments. The PSNRs exhibit similar (but not exactly the same) values for all sets of experiments in most cases, i.e. we obtained many various filters of similar (average) quality. As may be seen in Fig. 4c and 4d, for 30 CMRs in chromosomes there are several results (specifically F49, F51, F73, F74) that do not fit the usual PSNR course. In these cases, the evolution got stuck and was not able to further improve a suboptimal solution. Such filters do not work as expected and usually destroy the filtered image. The best filters achieved PSNR about value 30 for 10% noise which decreases to 12–15 in most cases for the highest noise intensity considered (70%).

In order to compare the results obtained in this paper with other solutions (both conventionally designed and evolved), the appropriate PSNRs are presented in Fig. 4e and 4f [30]. A significant difference of these solutions from our filters is that they all work with filtering windows 3×3 or 5×5 pixels and use advanced operations which means that their implementations are more complex. On the other hand, the proposed method using CMRs requires only basic relational operators. Despite this fact, the average PSNRs of the the conventional (median) filters from [30] are comparable to (some of them are even worse than) the values of our solutions. This is particularly true for Fig. 4e, where a single-step filtering is considered. Although the best evolved filters in [30] exhibit better PSNRs (between 30 and 35), the difference against the proposed results is not very high (the better filtering quality is expectable because of more resources used for their implementation). The image quality of those filters may further be improved by applying multi-step approach (similar to the CA development) as shown in Fig. 4f. The observations obtained from this comparison indicate that similar filtering quality can be obtained using the simplest concept of 2D cellular automata and with less computational effort which was the objective of this paper.

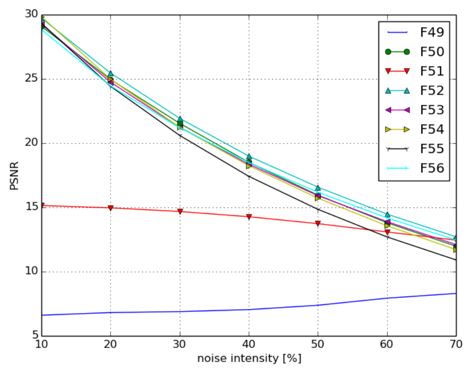
[4]https://www.fit.vutbr.cz/~vasicek/imagedb/



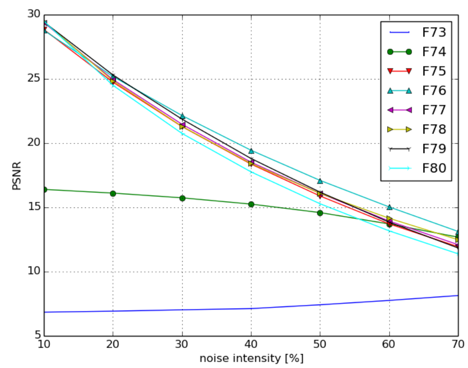
(a) runs 81–88 using 20 CMRs



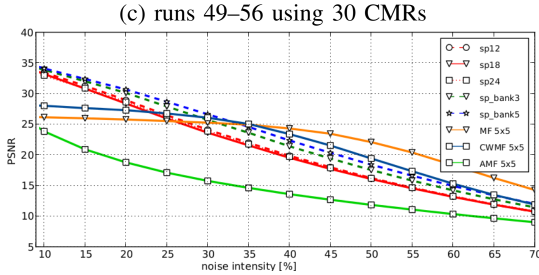
(b) runs 89–96 using 20 CMRs



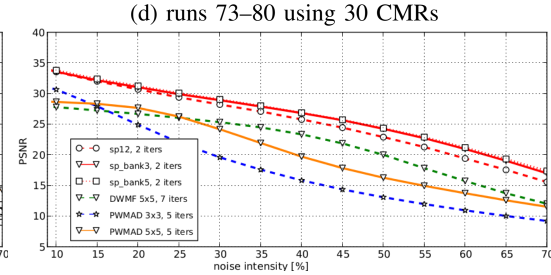
(c) runs 49–56 using 30 CMRs



(d) runs 73–80 using 30 CMRs



(e) single-step filtering from [30]



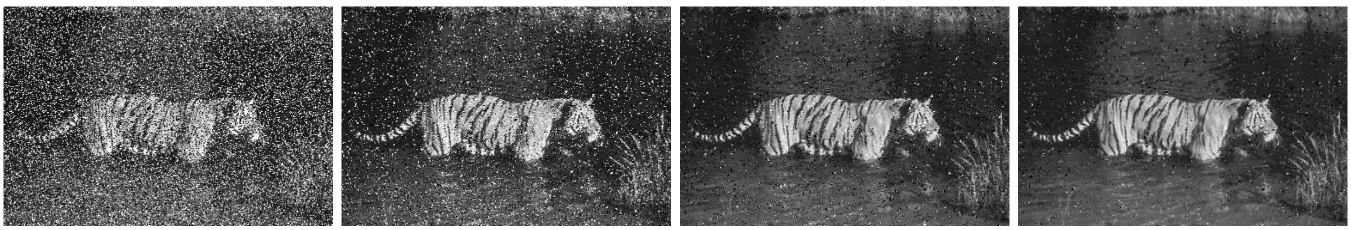
(f) multi-step filtering from [30]

Fig. 4: Analysis of filters from selected evolutionary runs. The filtering quality is expressed as average PSNR values evaluated for each filter and noise intensity from 10% to 70% using 24 test images from Vašíček's database (https://www.fit.vutbr.cz/˜vasicek/imagedb/). In all cases (a-d) the PSNRs were evaluated after 3 CA steps. Figures (e, f) are extracted from [30] for comparison purposes – the "sp" filters were evolved, the rest is conventionally designed.

Although the PSNR allows us to quantify the filter quality, it can not substitute visual evaluation of filtered images. For this purpose we chose one of promising filters and used it for filtering several images. Specifically, filter F89 will be considered from Fig. 4b whose PSNR analysis indicates its ability to filter higher noise intensities. Its CMR representation discovered by the evolution is shown in Figure 2. Figure 5 demonstrates filtering of 10% noise of the training image in 3 CA steps. Since this image was used to evaluate candidate filters during evolution, it is no surprise that the filter provides very good output quality. Although some pixels evidently remained corrupted, their values are closer to those of the reference image. However, it is important to remember that the filter works with 5 input pixels only and the result was evaluated after 3 CA steps. We verified that similar output quality can be observed on other images corrupted by 10% noise. A more interesting evaluation of the proposed filter is shown in Fig. 6 where 30% noise is considered. Note that neither such high noise intensity nor this image was seen by the filter during evolution. As can be observed, the filtered image also exhibits a very reasonable quality with only a few remaining noisy pixels. Again, 3 CA steps were sufficient to achieve this result. In order to demonstrate good features of the filter, it is important to determine whether it can generalise

(a) corrupted image (b) 1st step (c) 2nd step (d) 3rd step

Fig. 5: Demonstration of filtering 10% noise from the training image by one of the best evolved filters in 3 CA steps.



(a) corrupted image (b) 1st step (c) 2nd step (d) 3rd step

Fig. 6: Demonstration of filtering 30% noise by one of the best evolved filters in 3 CA steps.

(i.e. is able to filter other images than that on which it was trained, possibly with higher noise intensity). For this purpose the "selfie" of the author of this paper corrupted by 50% noise was chosen that exhibits a significantly different shape than both the training image and that evaluated in Fig. 6. In this case the filtering process was performed in 5 steps and the result is shown in Fig. 7. As can be seen, it is very hard to retrieve some details from the corrupted image (Fig. 7a). Clear shapes start to appear already after 2nd step (Fig 7c) and the best result (not changing significantly with further CA steps) can be achieved after 4th and 5th step – see Fig 7e and 7f respectively. One may conclude that, given the filter setup and noise intensity, the filter exhibits a reasonable output quality[5].

IV. Conclusions

An evolutionary method was presented for the design of image filters using two-dimensional uniform cellular automata. A technique called Conditionally Matching Rules was applied to represent transition functions for cellular automata working with 256 cell states. This approach allowed reducing the length of chromosomes for the evolution substantially which was a need for such high number of states since the traditional table-based encoding would require enormous memory space. The problem of removing Salt-and-Pepper noise from 8-bit grayscale images was considered as a case study. A cellular automaton was directly initialised by the values of pixels of a corrupted image and the evolution was applied to design a suitable transition function that is able to eliminate the noise from the image during ordinary development of the cellular automaton. We showed that high-quality filters can be obtained using only 5-cell neighbourhood of the cellular automaton and the best results provide a good output quality which is comparable with several existing solutions that require more resources. Moreover, the proposed evolutionary method demonstrated a high performance allowing us to design filters in very short time on a common PC.

The proposed method represents the simplest approach of filtering images by means of cellular automata in which the cell states are directly represented by the values of pixels. Considering the obtained results, there may be a possibility to further improve the concept. For example, Moore's 3×3-cell cellular neighbourhood can be used, an advanced selection of particular (subset of) CMRs to determined the next state may be considered or various sequentially applied transition functions might be performed. Moreover, further research will also be focused on more complex noise scenarios (e.g. random noise or impulse burst noise).

[5]the uncorrupted version of this image can be viewed at https://www.fit.vut.cz/person/bidlom/

References

[1] E. R. Dougherty and J. T. Astola, Eds., *Nonlinear Filters for Image Processing*, ser. SPIE/IEEE Series on Imaging Science & Engineering. SPIE/IEEE, 1999.

[2] M. O. Ahmad and D. Sundararajan, "A fast algorithm for two-dimensional median filtering," *IEEE Transactions on Circuits and Systems*, vol. 34, pp. 1364–1374, 1987.

[3] Z. Vasicek and L. Sekanina, "Novel hardware implementation of adaptive median filters," in *Proc. of 2008 IEEE Design and Diagnostics of*

(a) corrupted image (b) 1st step (c) 2nd step (d) 3rd step (e) 4th step (f) 5th step

Fig. 7: Demonstration of filtering 50% noise by one of the best evolved filters in 5 CA steps.

Electronic Circuits and Systems Workshop. IEEE Computer Society, 2008, pp. 110–115.

[4] S. Ko and Y. Lee, "Center weighted median filters and their applications to image enhancement," *IEEE Transactions on Circuits and Systems*, vol. 15, pp. 984–993, 1991.

[5] H. Hwang and R. Haddad, "Adaptive median filters: new algorithms and results," *IEEE Transactions on Image Processing*, vol. 4, no. 4, pp. 499–502, April 1995.

[6] Y. Dong, "A New Directional Weighted Median Filter for Removal of Random-Valued Impulse Noise," *IEEE Signal Processing Letters*, vol. 14, no. 3, pp. 193–196, 2007.

[7] V. Crnojevic, V. Senk, and Z. Trpovski, "Advanced impulse detection based on pixel-wise MAD," *IEEE Signal Processing Letters*, vol. 11, no. 7, pp. 589–592, 2004.

[8] P. Koivisto, H. Huttunen, and P. Kuosmanen, "Training-based optimization of soft morphological filters," *Journal of Electronic Imaging*, vol. 5, no. 3, pp. 300–322, 1996.

[9] P. Koivisto, J. Astola, V. Lukin, V. Melnik, and O. Tsymbal, "Removing Impulse Bursts from Images by Training-Based Filtering," *EURASIP Journal on Applied Signal Processing*, vol. 2003, no. 3, pp. 223–237, 2003.

[10] M. Nikolova, "A variational approach to remove outliers and impulse noise," *J. Math. Imaging Vis.*, vol. 20, no. 1-2, pp. 99–120, 2004.

[11] J. Zhu, S. Wang, X. Wu, and K. F.-L. Chung, "A novel adaptive svr based filter asbf for image restoration," *Soft Comput.*, vol. 10, no. 8, pp. 665–672, 2006.

[12] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Berlin Heidelberg: Springer-Verlag, 2015.

[13] J. D. Lohn and G. S. Hornby, "Evolvable hardware: Using evolutionary computation to design and optimize hardware systems," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 19–27, 2006.

[14] J. Dumoulin, J. Foster, J. Frenzel, and S. McGrew, "Special Purpose Image Convolution with Evolvable Hardware," in *Real-World Applications of Evolutionary Computing – Proc. of the 2nd Workshop on Evolutionary Computation in Image Analysis and Signal Processing EvoIASP'00*, ser. LNCS, vol. 1803. Springer-Verlag, 2000, pp. 1–11.

[15] S. Marshall, N. Harvey, and D. Greenhalgh, "Design of morphological filters using genetic algorithms," in *Tenth European Signal Processing Conference*. EURASIP, 2000, pp. 389–392.

[16] R. Porter, "Evolution on FPGAs for Feature Extraction," Ph.D. dissertation, Queensland University of Technology, Brisbane, Australia, 2001.

[17] J. F. Miller, *Cartesian Genetic Programming*. Springer-Verlag, 2011.

[18] L. Sekanina, L. S. Harding, W. Banzhaf, and T. Kowaliw, *Image Processing and CGP*, ser. Natural Computing Series. Springer Verlag, 2011, pp. 181–215.

[19] L. Sekanina, "Image filter design with evolvable hardware," in *Applications of Evolutionary Computing*, ser. LNCS, vol. 2279. Springer Verlag, 2002, pp. 255–266.

[20] A. Burian and J. Takala, "Evolved Gate Arrays for Image Restoration," in *Proc. of 2004 Congress on Evolutionary Computing CEC'04*. IEEE Publ. Press, 2004, pp. 1185–1192.

[21] Y. Zhang, S. Smith, and A. Tyrrell, "Intrinsic Evolvable Hardware in Digital Filter Design," in *Applications of Evolutionary Computing*, ser. LNCS, vol. 3005. Quimbra, Portugal: Springer, 2004, pp. 389–398.

[22] Z. Vasicek and L. Sekanina, "Reducing the area on a chip using a bank of evolved filters," in *Evolvable Systems: From Biology to Hardware*, ser. LNCS, vol. 4684. Springer Verlag, 2007, pp. 222–232.

[23] T. Sun and Y. Neuvo, "Detail-preserving median based filters in image processing," *Pattern Recognition Letters*, vol. 16, pp. 341–347, 1994.

[24] Z. Vasicek, M. Bidlo, L. Sekanina, and K. Glette, "Evolutionary design of efficient and robust switching image filters," in *Proceedings of the 2011 NASA/ESA Conference on Adaptive Hardware and Systems*. IEEE Computer Society, 2011, pp. 192–199.

[25] Z. Bao, F. Wang, X. Zhao, and T. Watanabe, "Fault-tolerant image filter design using ga," in *TENCON 2010 - 2010 IEEE Region 10 Conference*. IEEE, 2010, pp. 897–902.

[26] Z. Vasicek and L. Sekanina, "Hardware accelerator of cartesian genetic programming with multiple fitness units," *Computing and Informatics*, vol. 29, no. 7, pp. 1359–1371, 2010.

[27] J. Wang, Q. S. Chen, and C. H. Lee, "Design and implementation of a virtual reconfigurable architecture for different applications of intrinsic evolvable hardware," *IET computers and digital techniques*, vol. 2, no. 5, pp. 386–400, 2008.

[28] S. Harding, "Evolution of image filters on graphics processor units using cartesian genetic programming," in *2008 IEEE World Congress on Computational Intelligence*, J. Wang, Ed., IEEE Computational Intelligence Society. Hong Kong: IEEE Press, 1-6 Jun. 2008.

[29] Z. Vasicek and K. Slany, "Efficient phenotype evaluation in cartesian genetic programming," in *Proc. of the 15th European Conference on Genetic Programming*, ser. LNCS 7244. Springer, 2012, pp. 266–278.

[30] Z. Vašíček, M. Bidlo, and L. Sekanina, "Evolution of efficient real-time non-linear image filters for fpgas," *Soft Computing*, vol. 17, no. 11, pp. 2163–2180, 2013.

[31] J. von Neumann, *The Theory of Self-Reproducing Automata*. A. W. Burks (ed.), University of Illinois Press, 1966.

[32] P. Rosin, A. Adamatzky, and X. Sun, *Cellular Automata in Image Processing and Geometry*. Springer, 2014.

[33] P. L. Rosin, "Training cellular automata for image processing," *IEEE Trans. on Image Processing*, vol. 15, no. 7, pp. 2076–2087, July 2006.

[34] P. J. Selvapeter and Wim Hordijk, "Cellular automata for image noise filtering," in *2009 World Congress on Nature Biologically Inspired Computing (NaBIC)*, Dec 2009, pp. 193–197.

[35] L. Diosan, A. Andreica, and A. Enescu, "The use of simple cellular automata in image processing," *Studia Universitatis Babe-Bolyai Informatica*, vol. 62, pp. 5–14, 06 2017.

[36] P. Kundra, H. M. Singh, V. Kumar, and P. Juneja, "Digital image edge detection using cellular automata," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 7S2, pp. 332–337, 2019.

[37] M. Bidlo, "On routine evolution of complex cellular automata," *IEEE Trans. on Evolutionary Computation*, vol. 20, no. 5, pp. 742–754, 2016.

[38] ——, *Advances in the Evolution of Complex Cellular Automata*, ser. International Joint Conference, IJCCI 2016 Porto, Portugal, November 9-11, 2016 Revised Selected Papers. Springer International Publishing, 2019, pp. 123–146.

[39] ——, "Comparison of evolutionary development of cellular automata using various representations," *MENDEL Soft Computing Journal*, vol. 2019, no. 1, pp. 95–102, 2019.