VERY LOW BIT-RATE VIDEO CODING USING CELLULAR NEURAL NETWORK UNIVERSAL MACHINE

by

Krzysztof Slot, Leon O. Chua, and Tamas Roska

Memorandum No. UCB/ERL M97/46

27 May 1997

VERY LOW BIT-RATE VIDEO CODING USING CELLULAR NEURAL NETWORK UNIVERSAL MACHINE

by

Krzysztof Slot, Leon O. Chua, and Tama Roska

Memorandum No. UCB/ERL M97/46

27 May 1997

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Very Low Bit-Rate Video Coding Using Cellular Neural Network Universal Machine

Krzysztof Ślot[1], Leon O. Chua, Tamas Roska[2]

Nonlinear Electronics Laboratory,
University of California at Berkeley, Berkeley, CA 94720, USA[3]

May 27, 1997

Abstract
A method of video coding for very low bit-rate channels, which is implemented using Cellular Neural Network Universal Machine, is presented in the following report. The presented method combines elements of a standard approach to video coding with elements of second generation video-coding techniques. Inter-frame coding is performed using standard block-based motion estimation procedure, while intra-frame coding is based on vector quantization approach. To satisfy constraints imposed by very low bit-rate channel throughput, a number of bytes which were considered to be used for representing video sequence frames, was assumed to be less than 200. Simulations of the algorithm execution, based on actual CNN UM chip parameter values, show feasibility of using the proposed method for real-time implementation of very low bit-rate video coding.

1. Introduction

This report is concerned with a Cellular Neural Network Universal Machine (*CNN UM*) [1] application for coding of video sequences, which are to be transmitted over very low bit-rate channels, such as voice channels of mobile telephony systems. For existing mobile telephony system standards (such as Pan-European GSM, the American IS-54 and IS-95, and Japanese systems) channel throughput rates range from 6.7 kbits/s to 13.6 kbits/s. Since 10 frames/second is considered to be a minimum frame rate which gives satisfactory smooth perception of motion, no more than approximately 100 bytes can be used for a single frame representation. It follows, that image compression ratios higher than 250:1 must be ensured, if video sequence frames have QCIF (*quarter common intermediate format*) format. Therefore, a very careful and at the same time very fast (real-time) image analysis must be performed to satisfy these extremely high requirements.

Several ways of very low bit-rate video coding realization have been studied so far, however none of them proved to be general and superior with respect to other approaches. The main feature of most of the proposed methods is use of block-based motion estimation procedures as a means for inter-frame coding. Methods which were proposed for intra-frame coding range from transform-based (e.g. DCT-based [2], suggested by H.263 standard), through vector quantization- [3], and quad tree-based techniques to fractal coding [4]. A common denominator for the proposed methods is to

[1] *On leave from Institute of Electronics, Technical University of Lodz, Poland*
[2] *Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, Hungary*
[3] This work has been supported by the Office of Naval Research under grant N00014-96-1-0753

consider images as a collection of pixels, where relevant information is contained in pixel gray level distributions.

Severe problems in obtaining satisfactory results for very low bit-rate video coding using conventional approaches led to the formulation of the concept of so called "second generation" or "object-oriented" video coding techniques [5]. The main idea underlying second generation video coding techniques is to consider images as a collection of objects rather than as a collection of pixels. This approach can potentially lead to very efficient image coding, however, much more complex image analysis is required to generate contents-based image representation. An important feature of second generation techniques is acknowledgment of the importance of Human Visual System (HVS) properties in a coding procedure. A unified framework for object-oriented video coding techniques is to be the MPEG-4 standard, which is expected to be issued by the end of 1997 [6].

The following paper presents a video coding method which combines a conventional approach to video coding with elements suggested by object-oriented techniques. CNN UM is considered to be a physical means of the method implementation, since it offers a sufficient computing speed for performing real-time, complex image analysis. The main elements of the proposed method are the following:

- standard block-based motion estimation procedure for inter-frame coding
- multi-stage vector quantization-based approach for intra-frame image coding
- detection of facial objects (mouth and eyes), which is intended to focus intra-frame image coding algorithm on regions which are of special significance for a subjective evaluation of a result quality

Computer simulations of the algorithm operation which are shown in the paper were based on actual parameters of physically manufactured CNN UM integrated circuits [7],[8]. Since CNN UM is a parallel processing structure, most of the procedures of the presented video coding method were implemented in parallel. An estimated time of the algorithm execution for a single QCIF-format frame ranges from approximately 16 to 58 milliseconds, depending on used parameter value set. To comply with constraints imposed by very low bit channel throughput rates, a number of bytes used for representing every frame (i.e. for encoding both motion information and intra-frame parameters) were set between one hundred and two hundred.

The structure of the paper is the following. A brief description of an architecture of CNN UM which is considered to be a framework for the physical implementation of the proposed method is given in Section 2. A description of a general approach used for a realization of video coding is presented in Section 3. A CNN UM implementation of inter-frame and intra-frame video coding procedures is shown in Section 4. Results of the algorithm application for coding of standard video sequences "Claire" and "Miss America" [9] are described in Section 5.

2. The physical framework of the method

An architecture of the CNN UM which has been used in deriving the presented video coding algorithm is based on actual VLSI realization presented in [7]. The structure which was considered throughout simulations is a two-dimensional array of locally interconnected cells. The size of the array is assumed to fit a size of QCIF image format, i.e. array is composed of 144 rows and 176 columns.

Block diagram of the single circuit processing element - a cell - is shown in Fig.1. The cell core implements dynamic behavior required by the CNN paradigm [10]. Every cell contains also two banks of local memories: analog (LAMs) and logic (LLMs) which are used to store input data as well as intermediate processing results. Finally, processing in every cell can be disabled/enabled using Fixed State Map controlling lines.

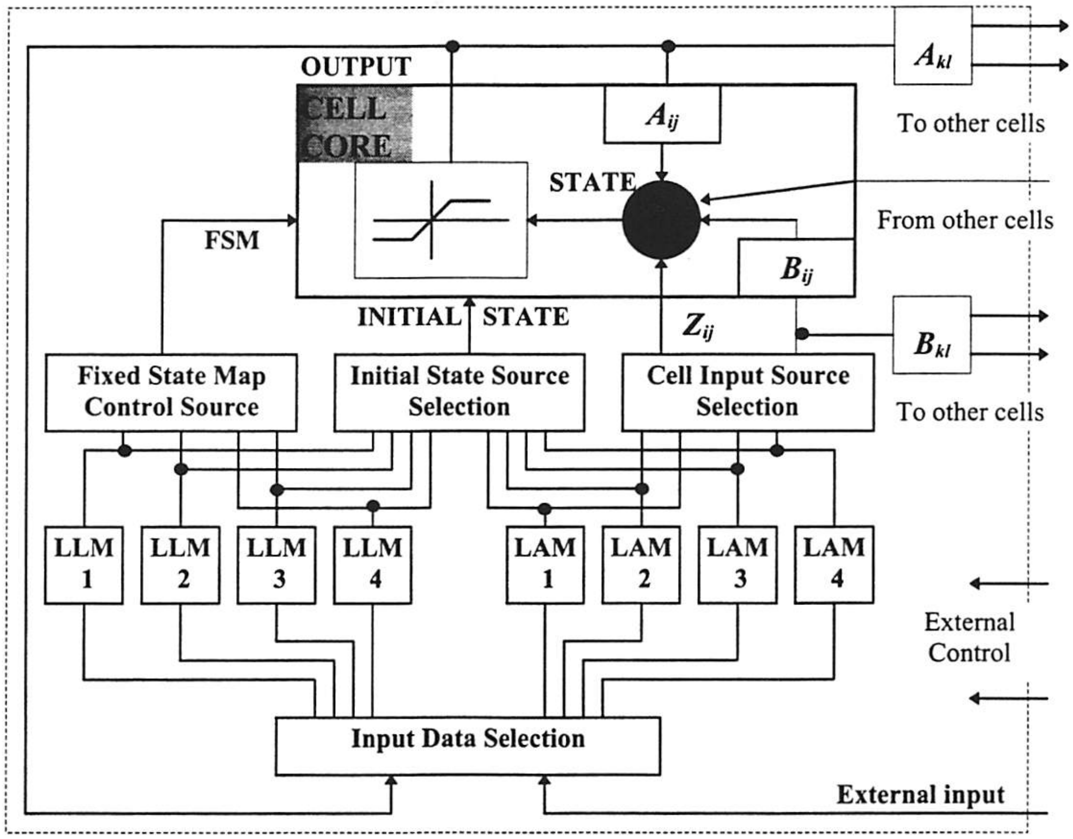


Fig.1. Block diagram of the circuit cell

Templates of size *3x3* of the following general form:

$$\mathbf{A} = \begin{bmatrix} a_{-1-1} & a_{0-1} & a_{1-1} \\ a_{-10} & a_{00} & a_{10} \\ a_{-11} & a_{01} & a_{11} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} b_{-1-1} & b_{0-1} & b_{1-1} \\ b_{-10} & b_{00} & b_{10} \\ b_{-11} & b_{01} & b_{11} \end{bmatrix} \qquad \mathbf{Z}$$

where **Z** denotes space-variant bias array, can be implemented using the considered CNN UM chip.

3. A general structure of video coding algorithms

Block diagram which illustrates a standard approach to video coding is presented in Fig.2. The procedure is composed of two main steps: motion estimation (inter-frame coding) and coding of motion prediction error (intra-frame coding).

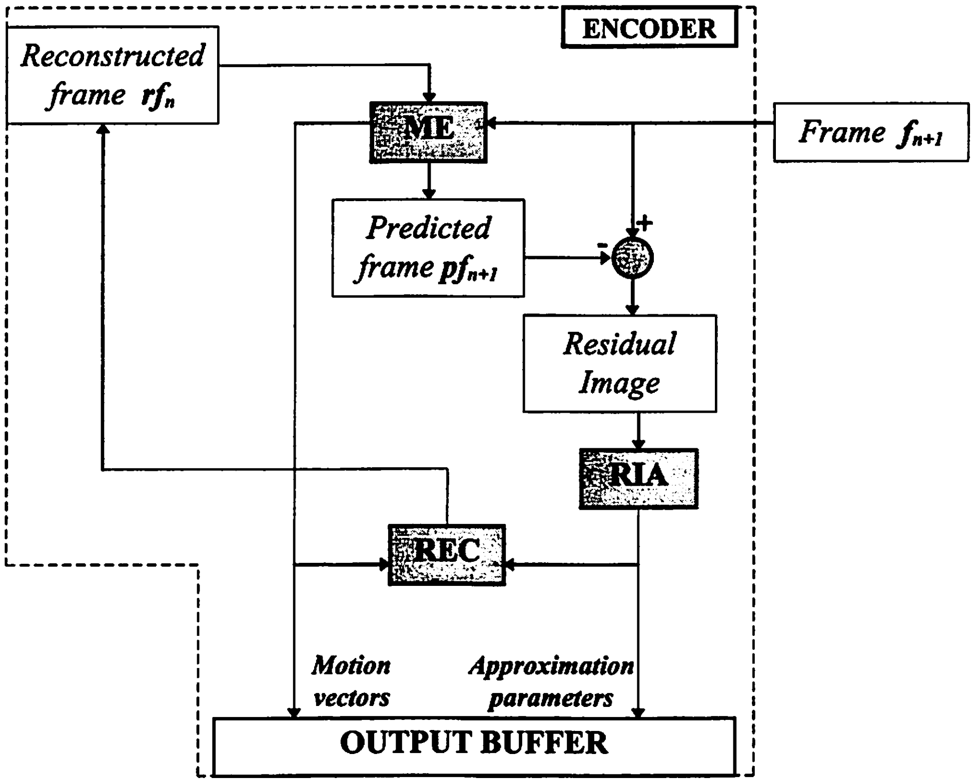


Fig.2. Block diagram of video coding procedure

A sequence frame f_{n+1} and a reconstructed form of the preceding frame - rf_n - constitute the input of the video coding procedure. The first operation of the algorithm is the motion estimation procedure (ME). The goal of motion estimation is to determine changes in consecutive image frames which are due to image objects displacement and to represent these changes using appropriately selected parameters. In block-based techniques, translations of image blocks (usually of size *8x8* or *16x16* pixels) are considered to be the only form of recognized motion. Parameters which characterize motion for these techniques are *motion vectors*, associated with every image block.

Motion vectors and the $\boldsymbol{rf_n}$ image are used to form a prediction of the frame $\boldsymbol{f_{n+1}}$ (denoted $\boldsymbol{pf_{n+1}}$) which is being subtracted from the actual frame $\boldsymbol{f_{n+1}}$, yielding *Residual Image.* Residual image represents a motion prediction error and it constitutes an input for the next part of the procedure, referred to as *Residual Image Approximation* (**RIA**).

The goal of RIA procedure is to generate a good approximation of residual image by means of possibly small number of parameters, providing this way information for minimizing motion estimation error.

The last element of the video coding procedure is a reconstruction of the frame $\boldsymbol{f_{n+1}}$ (block **REC**) which is based on motion vectors, approximation parameters and the image $\boldsymbol{rf_n}$. A result of this operation - an image $\boldsymbol{rf_{n+1}}$ - will be used as a reference for coding the consecutive frame $\boldsymbol{f_{n+2}}$ ($\boldsymbol{rf_{n+1}}$ is an image which is being reconstructed on the receiving part of the system). A result of the video coding procedure which is passed to the system output buffer and then is transmitted over the channel, consists of motion vectors and residual image approximation parameters.

4. CNN UM implementation of the video coding algorithm

The video coding method presented in this report implements the general idea presented in the previous section. Since block-based motion estimation techniques has been recommended in [6] as a way of inter-frame coding, this approach has been adopted in the algorithm. Of many existing ways of intra-frame coding implementation, v*ector quantization* (VQ) technique has been selected. The basic idea of vector quantization is to approximate image areas with best matching elements selected from some pre-defined code-book. One of possible ways which can be used in estimating a similarity between two-dimensional functions is to evaluate results of their two-dimensional cross-correlation. Since the basic operations executed by CNNs are two-dimensional cross-correlation, CNN UM is well-suited for implementing VQ-based intra-frame coding algorithms.

4.1. Motion Estimation Procedure

The proposed video coding method implements block-based inter-frame coding, i.e. every frame is considered to be a collection of non-overlapping square blocks. Following recommendations given by H.26x and MPEG-4 standards, motion estimation is performed for blocks of size *8x8* pixels. Blocks are subject to horizontal and vertical shifts within the range [*-15* .. *+15*] pixels. The sequence of the following operations is being repeated during estimating a motion introduced by a frame $\boldsymbol{f_{n+1}}$ (see Fig.3):

1. Frame shifts

Frame $\boldsymbol{rf_n}$ is being shifted by one pixel either in horizontal or vertical direction in every iteration. Assuming that frame $\boldsymbol{rf_n}$ is implemented using cell LAMs which are currently selected to act as cell inputs, frame shifts can be realized using a simple template of the form:

$$\mathbf{A} = [0], \qquad \mathbf{B} = \begin{bmatrix} 0 & 1-x & 0 \\ x & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{Z} = \mathbf{0}$$

where:

$$x = \begin{cases} 0 & for \quad shift-down \quad operation \\ 1 & for \quad shift-right \quad operation \end{cases}$$

2. *Frame comparison*
A similarity between frames is being determined by analyzing absolute values of the frame subtraction result, averaged within blocks of size *8x8* pixels. If frame $\boldsymbol{rf_n}$ is represented by inputs of array cells and frame $\boldsymbol{f_{n+1}}$ is a CNN bias array (**Z**), then the frame subtraction is realized using the template:

$$\mathbf{A} = [\,0\,], \qquad \mathbf{B} = [\,-1\,], \qquad \mathbf{Z}$$

Absolute value of the obtained result is being computed in two steps. First, the subtraction result is being thresholded (at zero-level), using the template:

$$\mathbf{A} = [\,4\,], \qquad \mathbf{B} = [\,0\,], \qquad \mathbf{Z} = \mathbf{0}$$

Next, frame subtraction results for which thresholding yielded negative values, are being inverted. This can be done using the following template:

$$\mathbf{A} = [\,0\,], \qquad \mathbf{B} = [\,-1\,], \qquad \mathbf{Z} = \mathbf{0}$$

and a fixed state map, initialized with inverted results of the thresholding operation. Finally, absolute values of frame subtraction results (which represent comparison error) are subject to averaging within 8x8 pixel blocks. This is performed by means of the following averaging template:

$$\mathbf{A} = [0], \qquad \mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \qquad \mathbf{Z} = \mathbf{0}$$

To satisfy constraints imposed by CNN UM chip physical implementation this template is appropriately decomposed, using the procedure presented in [11].

3. *Result analysis*
   If for any block an averaged error computed using the presented sequence of steps is lower than the one obtained in previous iterations, a motion estimation result is being appropriately updated.

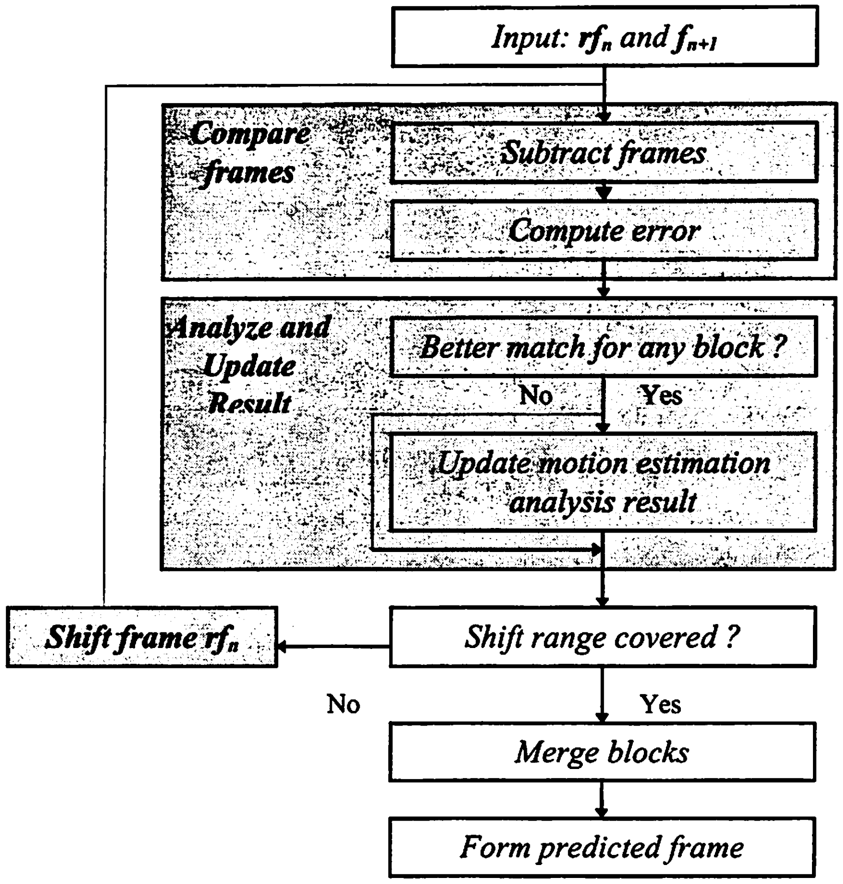


Fig.3. Block diagram of the CNN UM motion estimation algorithm

A set of motion vectors, associated with every block of a frame $\boldsymbol{rf_n}$ is a result of the motion estimation procedure. Motion vectors contain coordinates of the displacement of a corresponding block, which resulted in the best match against the corresponding region of the frame f_{n+1}. To increase a compression ratio, this result is subject to further analysis: if motion vectors of four neighboring blocks are the same, they are grouped into a single block of size *16x16* pixels.

The last part of motion estimation procedure is a formation of predicted form of the frame f_{n+1} (pf_{n+1}). To decrease block artifacts, additional filtering is performed, by convoluting boundary block regions with an appropriate averaging filter.

4.2. Residual image approximation

The second part of the proposed video coding method - residual image approximation - is based on pattern matching technique. A general idea is to approximate residual image using a set of best matching patterns from a pre-defined code-book. A number of used patterns should be sufficiently small, to satisfy constraints imposed by very low channel throughput. Patterns are two-dimensional discrete pixel gray-level distributions and they are represented by two-dimensional matrices with appropriately normalized elements. Pattern matching is realized by performing cross-correlation between residual image and code-book matrices. A result of matching an image **U** (**U**=[u_{ij}]) with a matrix $\mathbf{T}^v$ ($\mathbf{T}^v$ =[T_{kl}]) is an image $\mathbf{Y}^v$ ($\mathbf{Y}^v$ = [y^v_{ij}]), such that:

$$y^v_{ij} = \sum_k \sum_l u_{i+k,j+l} T^v_{kl} \qquad i=1..M, j=1..N, k,l=-R/2..R/2$$

where: *M,N* denote respectively image height and width, and code-book elements are of size *RxR.*

Note, that the presented formula describes the operation of an uncoupled CNN, where the matrix $\mathbf{T}^v$ corresponds to **B**-template. Therefore, CNNs provide a straightforward framework for implementation of vector quantization techniques.

The similarity of the given image region with a code-book element is proportional to the magnitude of the correlation result. A straightforward implementation of the pattern matching procedure is to apply all code-book elements to the image of interest, and then to select the best fitting ones as the image approximating set (using correlation magnitude to be a selection criterion). However, two main problems arise while attempting to use this approach in the very low bit-rate video coding case.
The former problem is concerned with a subjective evaluation of an image quality. For the class of images containing human faces (which is of interest from the point of view of video-telephony applications) this evaluation diverges from the objective norms. Even slight distortions in reconstructing facial expressions give unacceptable results, while quite big errors made in other regions can be tolerated. Therefore, special emphasis should be put on approximating facial image regions in the video coding procedure.
The latter problem concerns a choice of size of code-book elements. On the one hand, a code-book used for image approximation should contain an exhaustive amount of possible patterns and since a number of possible patterns grows exponentially with a pattern size, 2-D functions of small size should be used. On the other hand, to cover possibly large areas using a small number of code-book elements, their size should be as large as possible.

To provide satisfactory results of VQ-based residual image coding, an algorithm which takes into account an importance of both of presented problems, has been proposed (Fig.4). Two major modifications were introduced to the simple pattern matching approach:

- facial objects detection procedure has been added at the beginning of the algorithm
- residual image is being approximated using different code-books which are being applied in the pipelined manner

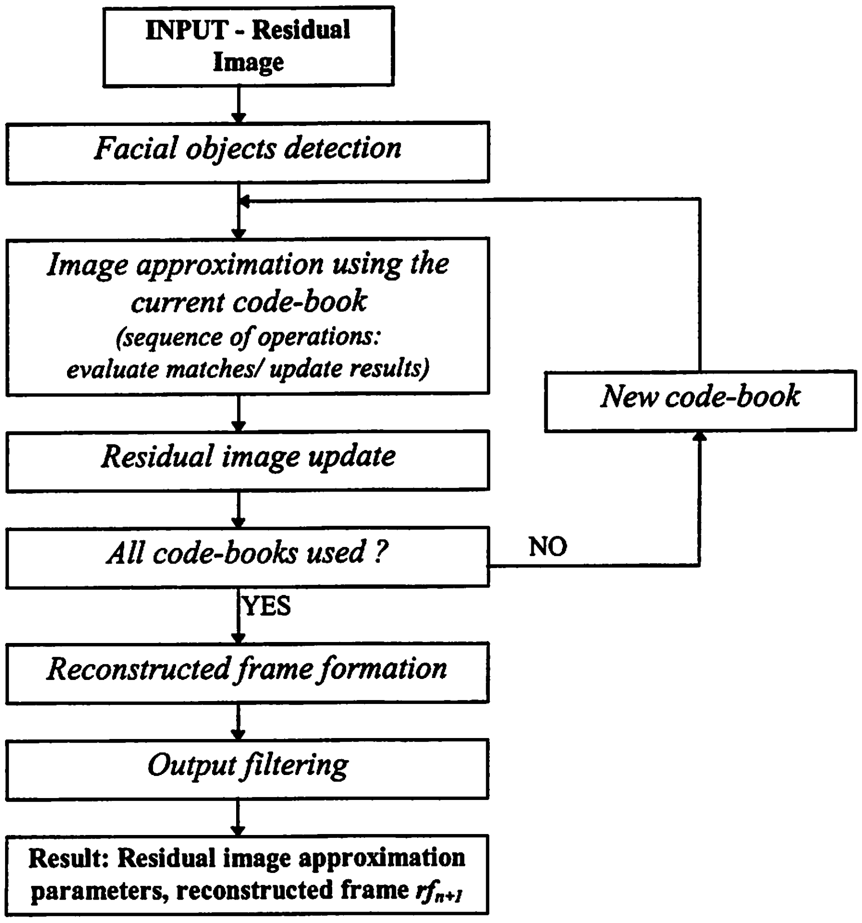

INPUT - Residual Image
Facial objects detection
Image approximation using the current code-book (sequence of operations: evaluate matches/ update results)
New code-book
Residual image update
All code-books used ?
NO
YES
Reconstructed frame formation
Output filtering
Result: Residual image approximation parameters, reconstructed frame rf_{n+1}

Fig.4. Block diagram of residual image approximation procedure

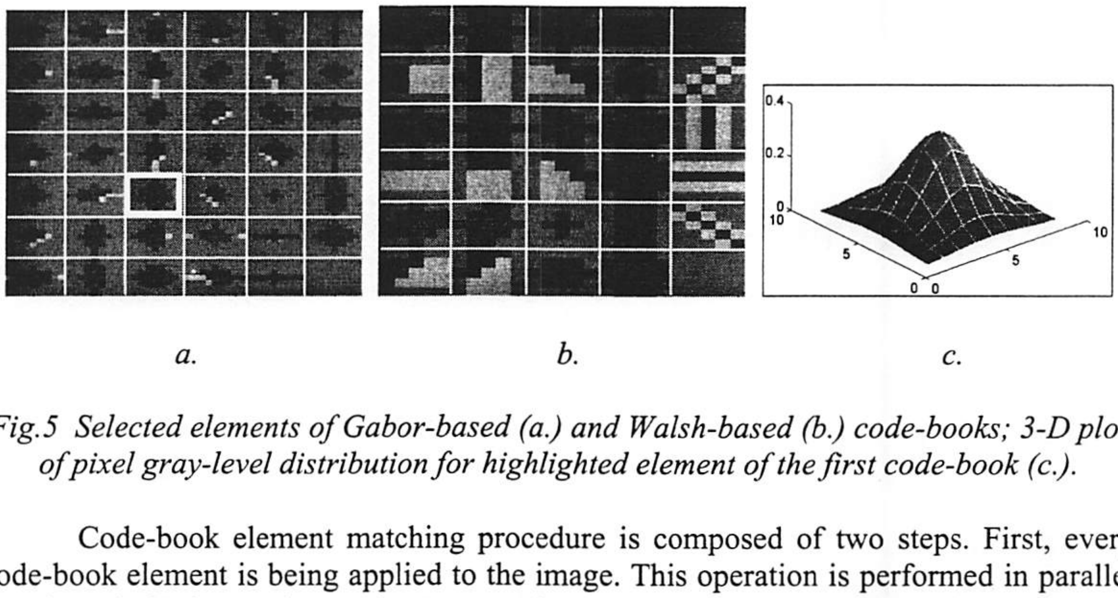
The goal of facial objects detection is to determine image regions where facial expression changes occur in consecutive frames. Since pattern matching technique can be realized very efficiently by CNNs, this way was selected to implement facial objects detection. A code-book used for facial feature detection has been designed to cover most typical shapes observed for changes in eye and mouth expressions. A result of facial objects detection procedure is a binary mask, which marks regions located around detected facial features. A size and shape of these regions are determined by residual image properties in areas adjacent to detected features. Residual image regions marked by the produced binary mask will be handled with a special attention during the remaining part of the coding procedure.

The main idea underlying an application of a sequence of pipelined pattern matching procedures is to gradually refine the quality of residual image approximation. Three consecutive pattern matching steps, each utilizing a different code-book, have been used in the proposed method. After each of the pattern matching steps is completed, residual image is being updated and it is fed as an input to the next step.

The code-book which is used in the first pattern matching step contains 110 elements of size *9x9* which implement mostly low-order 2-D Gabor functions. The main role of the first pattern matching step is to approximate relatively large and smooth residual image regions. The maximum size of template elements which are used in the algorithm - *9x9* - is determined by limited accuracy of their physical implementation in actual CNN UM chips. Some of elements of the code-book used in the first part of the procedure are shown in Fig.5a.

Majority of the second code-book 100 elements implement smoothed two-dimensional, low order Walsh functions of size *6x6*. The second code-book is intended to approximate these areas of residual image, where abrupt changes in an image intensity are encountered and some of the code-book elements are shown in Fig.5b.

Finally, the third code-book contains 46 elements of size 3x3 and 2x2. The role of the last step is to approximate small residual image areas as well as to eliminate all errors introduced during the two previous steps of the procedure.



a. *b.* *c.*

Fig.5 Selected elements of Gabor-based (a.) and Walsh-based (b.) code-books; 3-D plot of pixel gray-level distribution for highlighted element of the first code-book (c.).

Code-book element matching procedure is composed of two steps. First, every code-book element is being applied to the image. This operation is performed in parallel for the whole image by setting appropriate CNN UM template and performing an execution phase. For example, a form of the template which is used to evaluate a match with a code-book element shown in Fig.5c is the following:

$$\mathbf{A} = 0, \quad \mathbf{B} = \begin{array}{ccccccccc} 0.02 & 0.05 & 0.12 & 0.19 & 0.23 & 0.19 & 0.12 & 0.05 & 0.02 \\ 0.05 & 0.16 & 0.37 & 0.62 & 0.73 & 0.62 & 0.37 & 0.16 & 0.05 \\ 0.12 & 0.37 & 0.86 & 1.42 & 1.67 & 1.42 & 0.86 & 0.37 & 0.12 \\ 0.19 & 0.62 & 1.42 & 2.33 & 2.76 & 2.33 & 1.42 & 0.62 & 0.19 \\ 0.23 & 0.73 & 1.67 & 2.76 & 3.26 & 2.76 & 1.67 & 0.73 & 0.23 \\ 0.19 & 0.62 & 1.42 & 2.33 & 2.76 & 2.33 & 1.42 & -0.62 & 0.19 \\ 0.12 & 0.37 & 0.86 & 1.42 & 1.67 & 1.42 & 0.86 & 0.37 & 0.12 \\ 0.05 & 0.16 & 0.37 & 0.62 & 0.73 & 0.62 & 0.37 & 0.16 & 0.05 \\ 0.02 & 0.05 & 0.12 & 0.19 & 0.23 & 0.19 & 0.12 & 0.05 & 0.02 \end{array}, \quad I=0$$

Since CNN UM chips, which are considered as a framework for the proposed method realization, provide only nearest-neighbor cell interconnections, the template presented above is subject to an appropriate decomposition (again, based on the method described in [11]).

The second part of the code-book pattern matching procedure is a selection of a set patterns which are to be used for residual image representation. Element selection process is performed at first for regions marked by the priority mask. To increase total image area which is covered by chosen patterns, a simple mechanism which prevents their extensive overlap is introduced. Namely, residual image is split into blocks of size of currently used templates, and it is assumed that no more than a single element can be selected within each block.

A result of the residual image approximation procedure is a set of parameters which contains:

- indices of code-book elements selected for residual image representation
- locations of these elements within an image
- exact values of correlation obtained for these elements

Given this information, image rf_n and motion vectors, frame f_{n+1} is being reconstructed and appropriately filtered (to eliminate noise-like errors) in the last step of the intra-frame coding procedure, yielding the image rf_{n+1}.

4.3. CNN UM algorithm for realization of the pattern matching procedure

A sequence of steps - *execute a template / analyze and update results* - is the main category of operations performed during realization of the video coding algorithm. During motion estimation part of the video coding algorithm, template executions implement frame shifting, frame subtraction and error calculation, and output of these operations is used for updating results obtained for previous steps. During residual image approximation part of the algorithm, template execution implements matching of a consecutive patterns with the residual image and results of these operations are used to update results obtained so far.

To implement a procedure of the parallel analysis and update of intermediate processing results using CNN UM, an appropriate algorithm, which exploits cell local analog and logic memories, has been developed. Let us consider the case of pattern matching procedures. A processing result is being formed in an iterative way - every consecutive template application might contribute to its final form. Three arrays of values are being used to accumulate and maintain intermediate results produced during preceding iterations of the procedure. Since CNN cell output magnitudes provide a measure of a given pattern match, the first array is used to store largest values which were encountered so far for each cell. This array of values is referred to as *Match Map* and it is implemented by using a single LAM of every cell. Since only a single, best-matching pattern per template-sized block (*9x9* for the first, *6x6* for the second and *3x3*

for the third code-book) is of interest, only a maximum cell output for every block is being recorded. Therefore, Match Map has a mosaic-like structure - all entries within a block are set to the same value (i.e. to local maximum).
To determine locations of best matching patterns within each template-size block, the second array of values is used. Elements of this array are binary - a single 'high' entry per block is used to indicate an exact position, where the best matching code-book element was centered. The array is referred to as a *Location Map* and it is implemented using a single LLM of every cell. Finally, the third array of values is used to identify the best matching element (code-book index) selected for every block.

The following CNN UM algorithm, presented in Fig.6, implements a single iteration of the "execute template/analyze and update results" operation performed during residual image approximation part of the algorithm.

- Matching of a consecutive pattern

  The operation is realized by executing appropriate CNN template. Resulting cell output magnitudes, which represent matches of the currently used pattern with all image regions, are temporarily saved in cell local analog memories.

- Local maxima determination.

  Maximum cell outputs are being determined for template-sized blocks in this step. This operation can be done using a nonlinear LOCMAX template [12], however another approach, which speeds up the processing, has been implemented. The basic idea is to compare outputs of cells located at the distance 2^k from each other (where k is a positive integer) and assign the higher value to both cells. The procedure is performed first in horizontal and then in vertical direction; it requires application of simple $1x3$ templates and appropriate fixed state maps.
  After the operation is completed, outputs of cells which belong to each block are set to the maximum result of correlation encountered in this block. An array of values produced during the procedure is referred to as an *iteration match map.*

- Local maxima location determination.

  A parallel implementation of search for local maxima locations is done by subtracting iteration match map from a pattern matching result (stored in the first phase of the algorithm) and thresholding the result at the level $T<0$. In effect, a binary image, containing a single black pixel in every image block is produced. This image is stored in one of cell's LLMs.

- Match Map update.

  Match Map update is performed in parallel in the following two steps. First, current version of the match map is being subtracted from the iteration match map, and the result is being thresholded at the zero-level (a single template is sufficient to perform this task). This way areas where better matches were obtained in the current iteration are marked and they are used as a fixed state map in the second step of the procedure, which is a selective update of the Match Map. The areas of iteration match map which are marked by the fixed state map are being copied to the Match Map, yielding its new form.

- Location map update

  The last operation of the procedure is an update of the pattern location map, which is realized similarly to the match map update. For the blocks which are selected to be updated, location map is being modified.

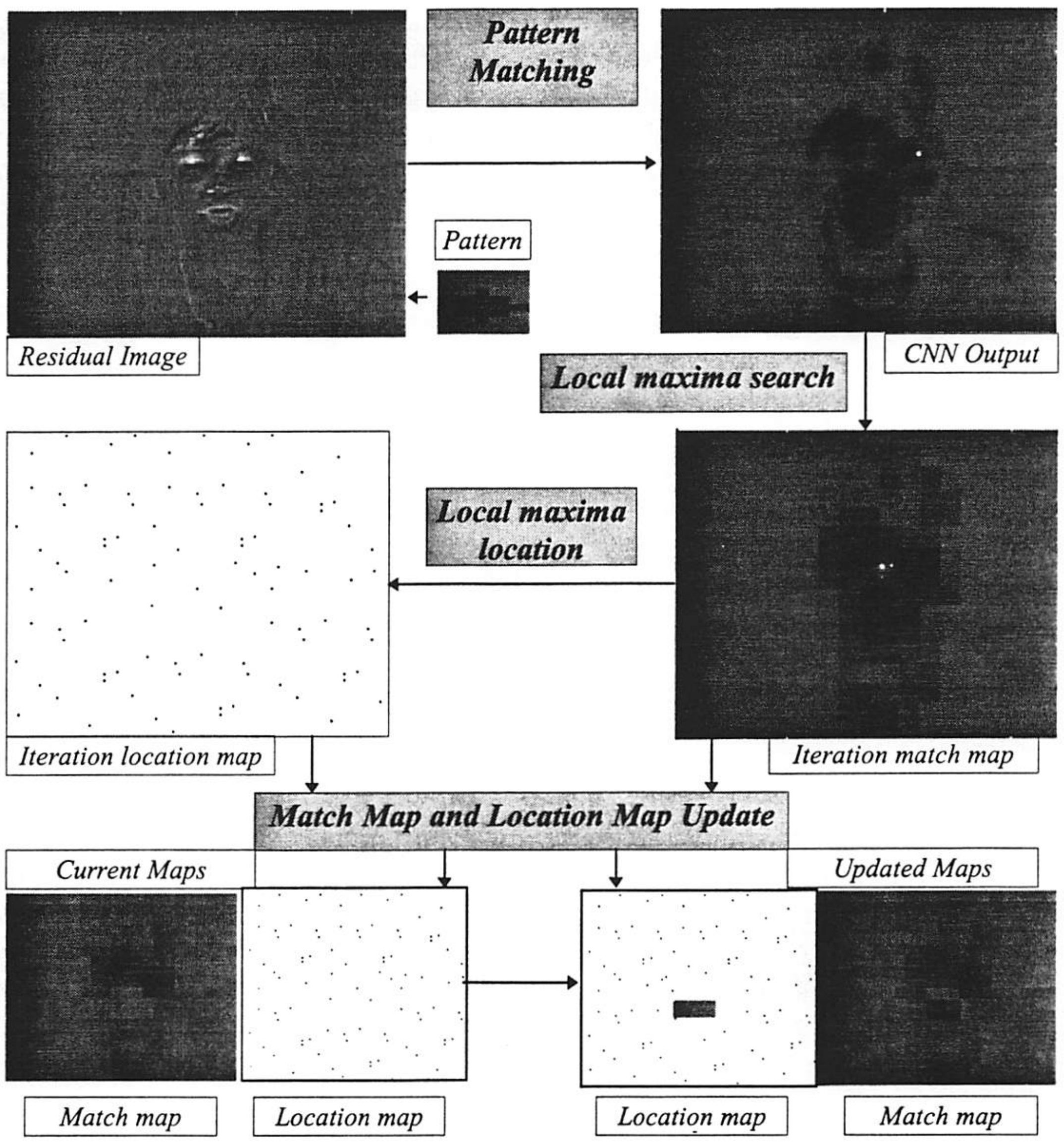


Fig.6 A single iteration of residual image approximation procedure. Shaded area in updated location map indicates region updated in current iteration.

After all code-book elements have been applied to the image, match map shows how well various patterns fit to various image regions. The last operation of image approximation procedure (performed for every code-book) is a selection of a set of best fitting elements. Instead of using any of data sorting (basically sequential) algorithms, the Match Map values are subject to an iterative thresholding procedure. After each step has been performed, a threshold value is being appropriately adjusted, so eventually only a required number of code-book elements is selected (unless no sufficiently good matches are present in the match map). This procedure can be easily implemented in parallel, and requires a use of simple CNN templates.

An algorithm which is used to analyze results and perform appropriate updates during motion estimation part of the video coding procedure is analogous to the one presented above. The only operation which is assumed to be realized outside the CNN UM is a final step of pattern matching procedure - a formation of an image which approximates residual image. This image contains patterns selected during image approximation procedure, placed in appropriate locations and scaled to match intensities of original image regions.

5. Simulation results

The proposed algorithm operation has been simulated using CNN UM prototyping system software. Two standard test sequences "Claire" and "Miss America" were used as a source of input data. Ten frames per second rate was assumed as a video stream frequency. The main goals of simulations were:

- to test a quality of the proposed video coding algorithm, assuming a very compact frame representation (100 and 200 bytes per frame)
- to estimate a speed of the algorithm execution assuming that it is run on a CNN UM chip.

An algorithm flow which illustrates processing of a single frame of the test sequence "Claire" has been presented in Fig.7. In the presented case, hardly no motion occurs for consecutive frames, however there is a significant change in a facial expression. Therefore, motion estimation procedure results in a very poor prediction of the frame f_{n+1}. The whole task of correct image reconstruction is being performed during the residual image approximation phase. Forty code-book elements were used to obtain a final approximation of the residual image.

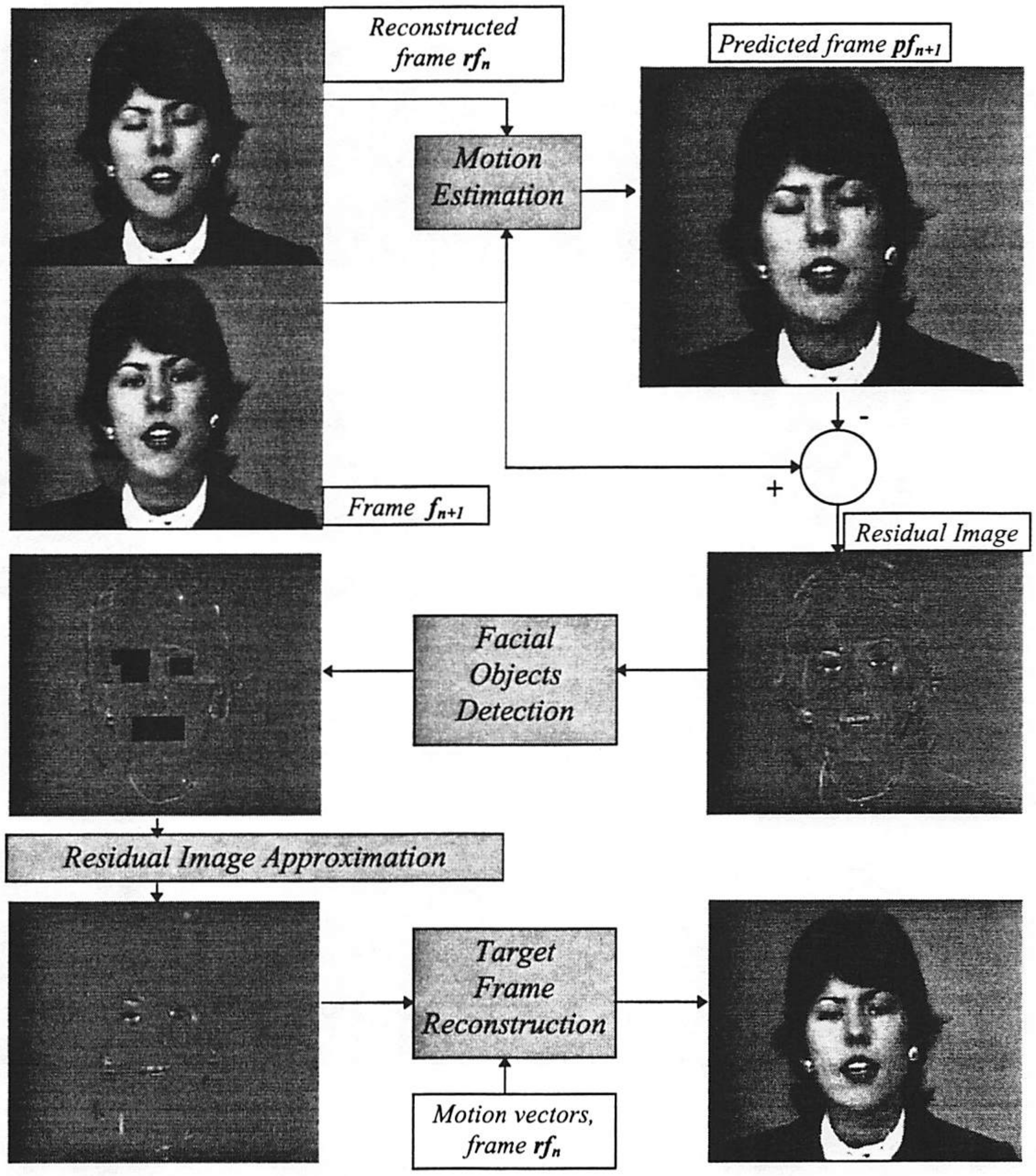

Reconstructed frame rf_n
Predicted frame pf_{n+1}
Motion Estimation
Frame f_{n+1}
-
+
Residual Image
Facial Objects Detection
Residual Image Approximation
Target Frame Reconstruction
Motion vectors, frame rf_n

Fig. 7 Main steps of video coding procedure

Results of coding sequences "Claire" and "Miss America" for two different number of bytes used to represent frames: 100 and 200, are shown respectively in Fig.8a,b and Fig.9a,b. A number of code-book elements used for frame approximation depended on the amount of motion information (motion parameters were encoded using Huffman code) and varied from 20 to 60.

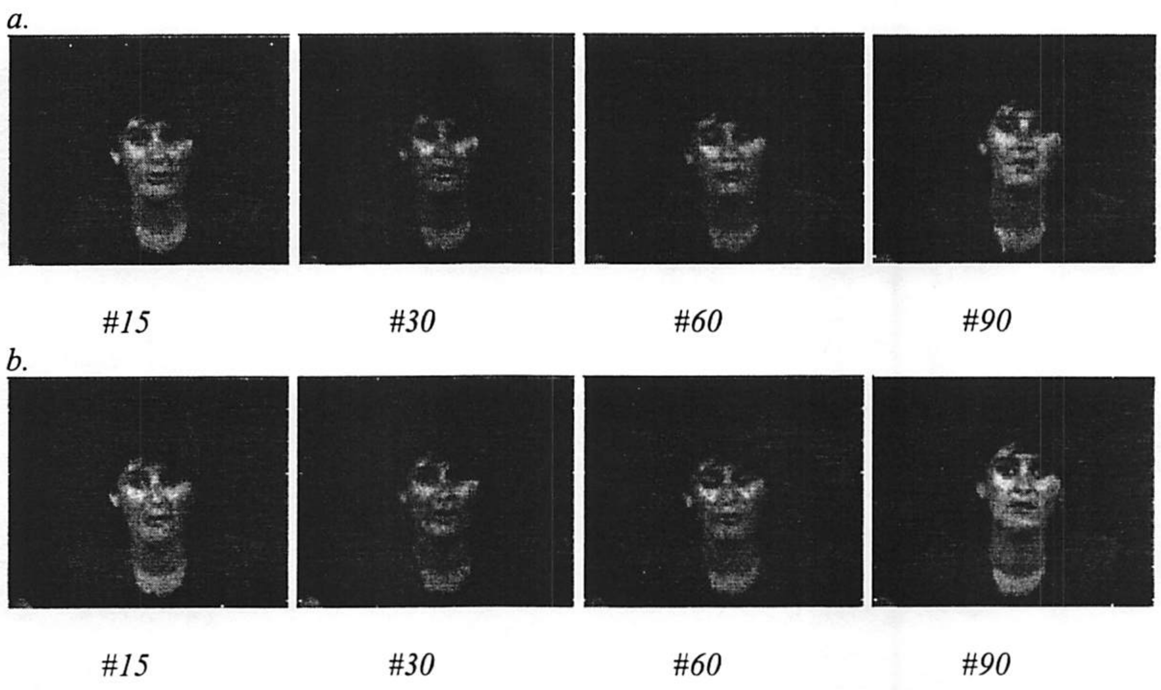


Fig.8 Reconstructed frames for "Miss America" test sequence for 100 bytes (a.) and 200 bytes (b.) per frame representation.

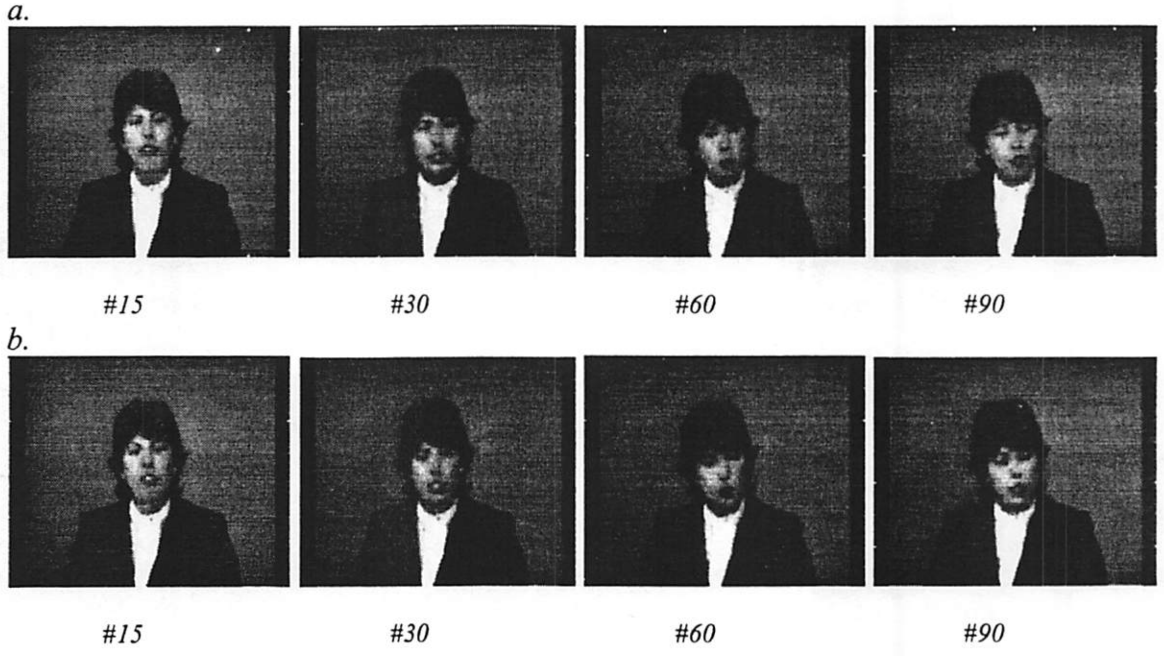


Fig.9 Reconstructed frames for "Claire" test sequence for 100 bytes (a.) and 200 bytes per frame representation.

For most of the presented frames, quality of images reconstructed using 100 bytes per frame is not significantly worse than the one obtained using 200 bytes per frame.

However, in cases when a consecutive frame introduces a large amount of motion and simultaneously, it brings significant changes in facial expressions, results obtained for the former compression rate are of poor quality, which is illustrated in Fig.10.

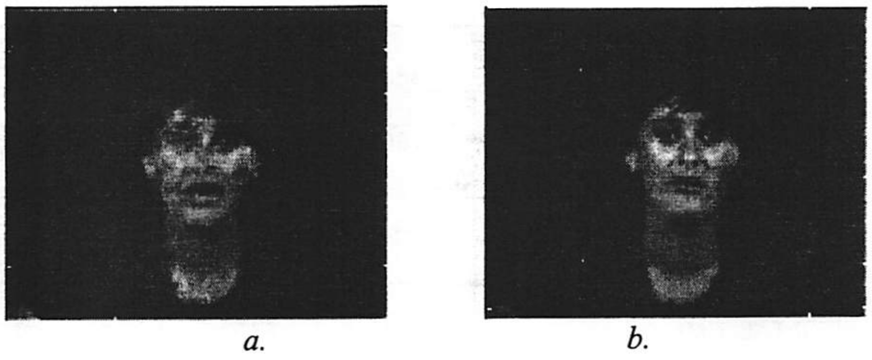


Fig.10 Results of frame reconstruction using 100 bytes per frame (a.) and 200 bytes per frame (b.) for the case when a large amount of motion and substantial changes in facial expressions are present

CNN UM parameter values used in simulations were based on measurement results obtained for various physical realizations of CNN UM chips ([7] - parameter set **A**, [8] - parameter set **B**) and are summarized in Table 1.

Operation	A	B	
Template loading time	1.9	0.1	us
Time constant	200	200	ns
Internal image transfers	200	200	ns
FSM switching time	100	50	ns
Row input/output (analog)	200	200	ns
Row input/output (digital)	200	200	ns

Table 1: Parameter values for CNN UM chips

Table 2 summarizes an amount of various elementary operations performed during execution of the video coding algorithm in CNN UM. Only the main and most time-consuming operations were listed. It can be seen, that the most extensive computations are being performed during a code-book elements matching. One can also see, that template loading is one of the most frequently performed algorithm operations. Therefore, an increase in a speed of template loading execution, significantly increases a speed of the video coding task completion. This can be seen when one compares total time required for loading all necessary templates for the used parameter sets **A** and **B**, which differ significantly in the template loading time.

	Template Execution	Template Loading	Control	Internal Image Transfers	External row transfers
Motion Estimation	13 860	4790	1200	3500	-
Facial Feature Detection	4667	2700	1700	1700	-
Code-book matching	21806	13576	6624	10210	-
Result Sorting and Residual Image Update	1952	1952	0	2007	8051
TOTAL OPERATIONS	42 285	23 018	9 524	17 417	8 051
TOTAL TIME **A** [us]	8457	43 735	952	3483	810
TOTAL TIME **B** [us]	8457	2 300	476	3483	810

Table 2: Elementary operations performed during the algorithm execution

	Parameter set **A**	Parameter set **B**
Algorithm Execution time [ms]	**58**	**16**

Table 3: Estimated execution time for both parameter sets

Finally, Table 3 presents estimation of algorithm execution time, made for both parameter sets, listed in Table 1. It can be seen, that even in the case of parameter set **A** total processing time is almost twice shorter than the time interval available for processing for 10 frames per second frame rate. Algorithm execution speed obtained for the parameter set **B** is sufficient to encode video sequences using CNN UM chips which implement arrays of lower size than *176x144* (CNN UM chips containing *176*144* cells are not available at the moment). Since algorithm operations have local properties, it is possible to perform the whole procedure as a sequence of operations performed separately for appropriately overlapped parts of an image. Therefore, a size of CNN UM chip reported in [7] - *64x64* - appears to be sufficient for a single-chip realization of the presented video coding method.

Conclusion

The presented report shows that CNN Universal Machine integrated circuits can be considered as an attractive tool for performing very low bit-rate video coding. The main advantage of the CNN UM - a processing speed - allows for application of complex computational procedures in real-time. For the presented video coding method, total execution time required to encode a single frame ranges from 16 to 58 milliseconds under an assumption that single CNN UM chip contains a sufficient number of cells. Although this is not the case for currently available chips, it was pointed out that also CNN UM ICs of much smaller size (e.g. the one reported in [7]) are capable to provide real-time video coding using the presented method.

The main problem which has to be further studied concerns a quality of reconstructed images, which appeared to be non-satisfactory if frames are to be represented by only 100 bytes. Two main directions of research should be considered to improve the video coding algorithm performance.

The first one is to improve the inter-frame coding procedure. A possible approach within block-based methods framework is to perform motion estimation with a half-pixel resolution, however this approach might result in increasing a number of bytes required to represent a frame. Another way is to implement object-oriented motion estimation. This solution would result in a significant increase in time required to estimate motion (images must be segmented), but it could result in more compact and accurate representation of motion information.

The second research direction which can improve the algorithm performance, is concerned with further development of intra-frame coding methods. As far as the presented VQ-based method is concerned, an extension and modification of used code-books can be considered. Another approach which can be studied is to combine pattern matching techniques with other video coding methods (e.g. transform-based methods can be applied for coding regions of interest), however, to implement this idea, efficient CNN UM implementation of these methods should be developed.

References

[1] Roska T., Chua L.O. *The CNN Universal Machine: An Analogic Array Computer,* IEEE Tr. On CaS,-II, vol.40, pp.163-173, March 1993

[2] J. Streit , L. Hanzo, *An Adaptive Discrete Cosine Transformed Videophone communicator for Mobile Applications*, IEEE International Conference on Acoustics, Speech and Signal Processing,May 1995, Detroid, USA.

[3] Neff R., Zakhor A. *Matching-Pursuit Based Video Compression*, UC Berkeley Draft, 1996

[4] J. Streit, *A Fractal Video Communicator,* IEEE Vehicular Technology Conference (VTC), pp. 1030-1034, Stockholm, Sweden, 1994.

[5] Torres L., Kunt M. *Video Coding: The Second Generation Approach,* Kluwer 1996

[6] Touradj Ebrahimi: *MPEG-4 Video Verification Model: A Video Encoding/Decoding Algorithm Based on Content Representation*, draft

[7] R. Dominguez-Castro, S. Espejo, A. Rodriguez-Vazquez, R. Carmona, P. Foldesy, A. Zarandy, P. Szolgay, T. Sziranyi, and T. Roska: *A 0.8 um CMOS Programmable Mixed-Signal Focal-Plane Array Processor with On-Chip Binary Imaging and Instructions Storage*, accepted for IEEE Int. Journal of Solid State Circuits, July 1997

[8] Jose M. Cruz Moreno *VLSI Implementation of Cellular Neural Networks*, PhD Thesis, University of California at Berkeley, EECS dept., 1996

[9] *Journal of Video Coding Related Links* - http://www.ee.princeton.edu/~ykchen /coding.html

[10] Chua L.O, Yang L. *Cellular Neural Networks: Theory and Applications,* IEEE Trans. CaS, vol.35, pp.1257-1290, Oct.1988.

[11] Crounse K., Chua L.O. *Arbitrary Spatial Convolution via CNN Universal Machine with 3x3 Templates: Methods and Issues,* UCB/ERL Memo M96/5, Jan.1996

[12] CADET, CSL-CNN software library (Templates and Algorithms), MTA SzTAKI, Budapest, Hungary, 1997