Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 72316, 13 pages
doi:10.1155/2007/72316

Research Article

A Massively Parallel Face Recognition System

Olli Lahdenoja,[1, 2] Mika Laiho,[1] Janne Maunu,[1, 2] and Ari Paasio[1]

[1] *Department of Information Technology, University of Turku, Joukahaisenkatu 3-5, 20014 Turku, Finland*
[2] *Turku Centre for Computer Science (TUCS), University of Turku, Joukahaisenkatu 3-5 B, 6th floor, 20520 Turku, Finland*

Received 12 April 2006; Revised 31 August 2006; Accepted 5 October 2006

Recommended by Heinrich Garn

We present methods for processing the LBPs (local binary patterns) with a massively parallel hardware, especially with CNN-UM (cellular nonlinear network-universal machine). In particular, we present a framework for implementing a massively parallel face recognition system, including a dedicated highly accurate algorithm suitable for various types of platforms (e.g., CNN-UM and digital FPGA). We study in detail a dedicated mixed-mode implementation of the algorithm and estimate its implementation cost in the view of its performance and accuracy restrictions.

1. INTRODUCTION

Face recognition is easy for humans but extremely difficult to perform with computer systems reliably in varying environmental conditions. Traditionally, the development of computer-based face recognition systems has concentrated on algorithm design, while the implementation cost in the hardware level in the form of, for example, speed and power consumption has been on a lower priority.

Methods such as PCA (principal component analysis) [1], also called as the eigenface method, and LDA (linear discriminant analysis) [2] aim at a more compact face representation (feature vector). This is performed by minimizing the mutual dependencies between the samples producing more uncorrelated and compact features. A variant of the PCA, two-dimensional PCA (2DPCA), was presented in [3]. As a consequence, the length of a face feature vector was increased with the overall recognition rate. After the computation of the feature vectors, a classifier, such as an SVM (a support vector machine) [4], is used to determine a distance measure between certain faces. This is performed by maximizing the margin between the different sample classes. In elastic bunch graph matching (EBGM) [5], the face is represented as a graph consisting of nodes which are represented by *jets* and edges describing the facial features. The *jet* is a descriptor of a local image region and it can be constructed using a wavelet approach. Some variants of the above-mentioned methods are not well suitable for high-speed implementations with large facial databases [6].

Nonparametric features called local binary patterns (LBPs) have recently shown high discriminative performance in many applications, for example, in face recognition [7]. The results in face recognition were achieved in comparison to state-of-the-art recognition methods including PCA, LDA, Bayesian classifier, and EBGM using a standard CSU (Colorado State University) [8] and FERET (the facial recognition technology) [9] environment.

The term nonparametric refers primarily to the fact that no assumptions are made on the local probability distributions of the image pixel intensities. The main advantages of nonparametric LBP features are invariances against various transformations, such as lighting bias and rotation. The advantages of massively parallel processing include high image processing performance, since the operations are performed simultaneously for a large number of processing units connected together into an array. We intend to use massively parallel processing to accelerate the performance of the LBP-based face recognition, allowing a compact implementation, including the imaging device (CMOS image sensors). Many real world applications can be predicted for this embedded face recognition system, for example, biometric face authentication in security applications.

The cellular nonlinear network (CNN) technology [10] is a powerful tool for high-speed massively parallel image processing. The concept of a programmable massively parallel CNN-UM (CNN-universal machine) processor was proposed in [11] where programming capabilities and memory were to be integrated to the same chip. Several chip implementations have been made to show that extremely high computation power can be included in a single chip [12] and a QCIF (176×144 pixels) resolution binary I/O CNN-UM [13].

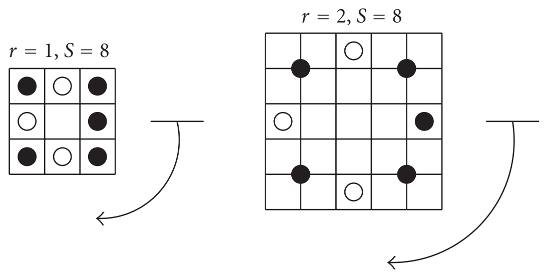
This paper describes a massively parallel face recognition system. The system consists of three parts, which are mixed-mode or CNN-UM-based LBP sampling methodology [14], a LBP-based massively parallel face recognition algorithm [15], and a dedicated mixed-mode hardware for the proposed algorithm (partly described in [16]). The sampling method which we present has the advantage of a speed increase up to 5 times compared to a modern standard computer, with on the other hand, some decrease in sampling accuracy and flexibility in the LBP sampling neighborhood size. In addition to face recognition, where the sampling speed is not critical, other high-speed LBP applications are likely to benefit from this sampling concept.

The presented algorithm is very competitive against the previous LBP-based face recognition algorithms in the recognition accuracy, while a trade-off between the face description length and recognition accuracy remains. The algorithm is flexible in that it can be applied beyond mixed-mode implementations for a massively parallel digital FPGA which does not introduce a significant decrease in the sampling accuracy. The dedicated mixed-mode hardware implementation for the proposed face recognition algorithm is presented in detail in schematic level with simulations considering mismatch and A/D conversion accuracy restrictions. This information can be used for determining whether it is appropriate for certain specific applications to use a fully digital or a mixed-mode implementation. At this point, the dedicated hardware is not fabricated, instead its performance is estimated using simulation tools.

2. THE LOCAL BINARY PATTERN METHODOLOGY

Local binary pattern (LBP) methodology was presented in [17] as a texture measure. It is based on comparing each grayscale pixel to its nearby samples and producing unique binary patterns based on the relative intensities of the pixels. The neighborhood is defined to be circular allowing invariance against rotation. First versions of the LBP were implemented using an eight pixel nearest neighborhood, but later circular neighborhoods with an arbitrary radius have been used. Also neighborhoods with multiple radiuses have been suggested in [18]. As a texture descriptor, the LBPs have shown to be very efficient in the view of computational complexity and recognition accuracy (see [18, 19]).

Recently, the LBP methodology has been applied also on several other computer vision tasks beyond the face recognition and the texture analysis. The LBP was used in [20] for modeling the image background and detecting moving objects. A combined face detection and recognition system was implemented in [21]. Also, many other computer vision applications for the LBP have been implemented, such as pose detection, context-based image retrieval, and industrial paper quality inspection.



Local binary pattern: 11010101
Decimal: 213

FIGURE 1: Local binary patterns with radiuses r of 1 and 2 and sample number S of 8.

2.1. Deriving the LBP feature vectors

Figure 1 represents the derivation (sampling) of an LBP. Comparison of the contour pixel intensities with the center pixel is started from a certain predetermined angle and proceeded in a certain direction, for example, clockwise. The LBP is generated by going through the whole local circle contour. If a circle contour sample is located between two pixels, the actual value of the sample is found out by interpolation (e.g., the average). As a result, each original pixel is replaced by its LBP representation, which is invariant with respect to monotonic grayscale changes and, in certain circumstances, rotation [19]. A histogram of length 2^S, where S is the number of local contour samples, is generated to describe an image or a region in an image. Several methods exist for histogram comparison, such as the chi-square statistics employed in [7].

The number of bins in the histogram can be reduced from 2^S using uniform [22] patterns which consist of at most two circular 0-1 and 1-0 transitions. For example, patterns 11000011 and 00011100 are uniform patterns (the leftmost and rightmost bits are considered neighbors), and patterns 01010100 and 01101101 are not uniform patterns. Further reduction on the number of histogram bins can be achieved by using a discrimination concept called symmetry [23].

The level of symmetry is the minimum between the total number of zeros $|B_i = '0'|$, $|\cdot|$ being the cardinality of the set and the total number of ones $|B_i = '1'|$ in an LBP (1), where index i goes through all bit locations

$$L_{\text{sym}} = \min\{|B_i = '1'|, |B_i = '0'|\}. \quad (1)$$

By using uniform patterns with a high level of symmetry (e.g., 00111100 and 11110000), the number of histogram bins can be reduced significantly (for S of 8, the reduction is 88%) from 2^S.

In [19] rotation invariant categories for the local binary patterns were defined. Certain local binary patterns can be rotated from each other to a minimum value so that each of the patterns in a certain class produces the same decimal value. For example, patterns 01100000, 00110000, and 00011000 belong to the same rotation invariance class since the minimum value that can be extracted by shifting all these patterns is 00000011.

3. LBP-BASED FACE RECOGNITION

In general, the process of recognizing, whether a specific input face (e.g., sensed by a camera or integrated sensors) matches one of the N reference faces stored into the memory, consists of many separate comparison steps. Two faces, the input face and the stored reference face, are compared with each other in each step. A distance measure (e.g., an integer number) is derived based on the comparison result according to (2),

$$d_{\text{match}} = \underset{d}{\operatorname{argmin}}\, d\left(\text{face}_{\text{input}}, \text{face}_{\text{stored},1:N}\right). \qquad (2)$$

Depending on the value of the distance measure d, the probability of the individuals to be matched can be determined. In the LBP face recognition, the face is usually divided into regions that can be used with or without weighting to enhance the spatial accuracy.

3.1. The histogram-based method

In [7] the face recognition was based on dividing face images into several spatially neighboring (e.g., 7×7 with 130×150 8-bit grayscale images) histogram regions, and an LBP histogram was constructed to represent each of these regions. A face descriptor was then a concatenated histogram of all the regions. A weighted chi-square distance (3) between the concatenated histograms was used with w denoting weights and S and M denoting the sample and model distributions, respectively. Index j is specific to a block and index i to a histogram bin,

$$\chi_w^2(S, M) = \sum_{i,j} w_j \frac{\left(S_{i,j} - M_{i,j}\right)^2}{S_{i,j} + M_{i,j}}. \qquad (3)$$

Not all the regions were considered equally important, but when comparing the feature vectors (concatenated histograms) the more informative face areas such as the eyes and the mouth were weighted (multiplied) by a factor that was considered to be optimal for the overall recognition results. In practice, the weighting factors were determined from the effect of that specific region on the overall recognition rate by neglecting all the other regions at a time.

3.2. The occurrence map method

A problem with the histogram approach [7] is related to the block division, since the borders between the blocks may lose information in the face comparison. If an LBP is slightly moved near the border of two blocks, it can move from one block into another. This causes a relatively large effect on the histograms of specific blocks. If the number of blocks is large, the occurrence probability of this block mismatch increases. More importantly, the spatial relations of the LBPs inside a block are not preserved by the histogram representation.

To improve the recognition accuracy of the LBP, we presented the occurrence map method for face recognition in [15]. Since each pixel can be replaced by a unique pattern in the LBP methodology, we chose to represent the occurrences of a certain pattern in an image (or an image region) by a binary occurrence map. For each histogram bin (LBP), a separate binary occurrence map is generated (with the size of the original image), that represents the locations for which that specific pattern occurs. The occurrence maps are compared with each other so that the other map is first dilated into certain directions. With D LBPs there will be an equal number of occurrence maps for describing their locations. The representation with D occurrence maps is possible with a tolerable feature vector length due to the two LBP compression methods, uniformity and symmetry.

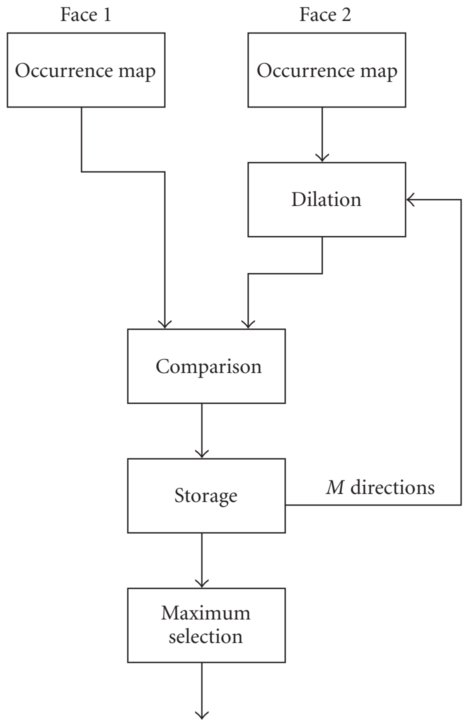


Figure 2: The matching algorithm.

4. FACE RECOGNITION WITH THE OCCURRENCE MAP METHOD

The flow diagram for comparing two faces using the proposed algorithm is shown in Figure 2. The two collections of occurrence maps, one representing the input face and the other representing the stored reference face, are given as the inputs to the matching process. A certain LBP is chosen to be processed. As a consequence, one occurrence map is chosen at a time, from both occurrence map collections (for face one and face two). Also, the image has been divided into blocks so that a spatially corresponding block is selected from both, the input face and the stored reference face.

The binary occurrence map of the face number two is dilated in a predetermined direction (e.g., N, W, S, E, etc.) a predetermined number of times. The dilated occurrence map is compared to the corresponding block of the other

```
SELECT the input face,
FOR each N stored faces,
    FOR all occurrence map pairs (total of D),
        FOR each block,
            FOR each direction M,
                K times Dilations,
                ADD (AND (occurrence maps)),
                STORE (add),
            END FOR,
        END FOR,
        ADD (block specific sums),
    END FOR,
    SELECT overall block direction
    ADD (block and direction specific sums)
END FOR,
MAX(add)
```

Algorithm 1

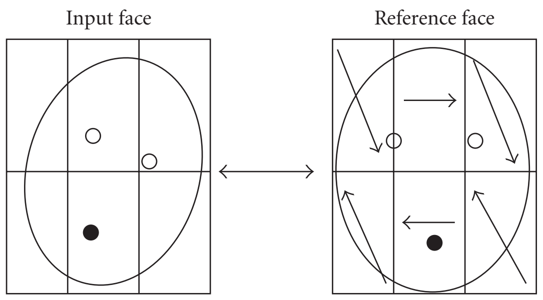


Figure 3: Examples of block matching vectors for matching two faces.

occurrence map (of face number one) by using a (local) AND operator for the whole array simultaneously. The dilation is repeated for M directions. Comparison results representing the similarities between the occurrence maps for specific dilation directions are stored into memory storage. Block specific occurrence map matching results are summed together, so that an optimal dilation direction is searched for each block. This means that the matching direction is specific for the whole block consisting of D occurrence maps. The dilation is performed only for either one of the faces under comparison. Following pseudocode illustrates the steps within each other performed during the comparison algorithm, see Algorithm 4.

Figure 3 represents an example of determining the block matching vectors, indicating the optimal dilation direction. Our method allows tolerance against slight changes in rotation and scaling adaptively between the images. In the case of comparing face images which represent different individuals, the optimal dilation direction is used to match the most informative regions (e.g., the eyes) into corresponding locations. In the case of comparing different images taken from the same individual, the optimal dilation direction is able also to match the features specific to that individual.

4.1. Simulation environment

The performance of the proposed algorithm was simulated in Matlab environment. Face images (130 × 150 × 8 bit) normalized by the CSU (Colorado State University) face recognition evaluation system were used [8]. In [7] the same system was used since it allows comparison to the other existing state-of-the-art face recognition methods. FERET (the facial recognition technology) [9] database is used by the CSU system. The FERET database is divided into five image sets. The gallery set contains frontal images from 1196 people and it is compared against four probe sets. The fb probe set contains 1195 face images with alternative expressions. The fc set contains 194 photos taken in different lightning conditions. The $dup1$ set is taken later in time and it contains 722 images. The $dup2$ set contains 234 face images taken at least a year after the corresponding gallery image. The CSU system uses the eye coordinates of images for normalization. After that, the actual recognition process is performed for the different methods.

4.2. Feature vector generation

Sample number S of eight was used in [7] with the radius of two to obtain the best recognition accuracy. This resulted in the initial length of the histogram representing a certain region (block), of 256. This could be reduced to 59 by using only uniform patterns. By taking the advantage of the symmetry, the number of bins of a region could further be reduced to 30 with a negligible effect on the average recognition rate [23].

A total of 30 binary occurrence maps (of size 130 × 150) can be used to describe the locations of each LBP (or a group of LBPs) in an image uniquely. In a certain location (array cell) there is only a single unique LBP at a time. These 30 occurrence maps can therefore be encoded losslessly using 5 bits of accuracy, resulting into the initial length of the face feature vector of 126 × 146 × 5 bits. The total length (including all face regions) of the histogram representation in [7] with S of eight was 2301 8-bit bins. Compared to this, the feature vector generated by using the binary occurrence maps and the proposed algorithm will be approximately four times larger.

One alternative for generating more compact feature vectors would be to subsample the images or the LBP images into a lower vertical and/or horizontal resolution, since the feature vector length is reduced quadratically with respect to the sampling ratio. However, eventually, there will exist a trade-off between the recognition accuracy and the resolution.

A suitable amount of consecutive dilation operations was experimentally found to be three. The images were sized according to the CSU normalization procedure as in [7] and 30 histogram bins (or equivalently 30 occurrence maps) were used. Using more dilations caused the dilated occurrence

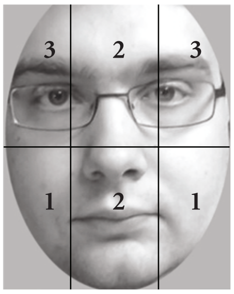


Figure 4: Division into 3×2 blocks and the respective weights.

Table 1: FERET results with the occurrence map algorithm.

Method	fb	fc	dup1	dup2	Average
LBP-AM(8,2) weighted	97%	86%	71%	67%	80%
LBP(8,2) [7] weighted	97%	79%	66%	64%	76%
LBP(12,2) [23] weighted	95%	85%	64%	66%	77%
LBP-AM(8,2) nonweighted	95%	85%	69%	63%	78%
LBP(8,2) [7] nonweighted	93%	51%	61%	50%	64%
PCA MahCosine	85%	65%	44%	22%	54%
Bayesian MAP	82%	37%	52%	32%	51%
EBGM optimal	90%	42%	46%	24%	51%
LDA ldasoft	73%	47%	45%	18%	46%

map of the face number two to cover an unnecessarily large portion of the other occurrence map and the occurrence probability of false matches during the comparison operation was increased. The normalized face images were divided in the simulations to a total of six blocks as shown in Figures 3 and 4.

We performed simulations with and without block weighting. The weighting was applied for each specific block according to values shown in Figure 4. The optimal comparison results for the blocks were multiplied by the weights to enhance the effect of face areas which are considered as the most significant in the view of the recognition (e.g., the eyes) and decrease the effect of less significant areas on the final result. The weights chosen are close to that used in [7]. A proper size for the regions was determined from iterative simulations. The sizes of the blocks that resulted in the best recognition accuracy were much larger (total of 6) compared to that of [7] where 49 blocks were used for the same sized images. Effect of slight displacement in face images to the weighted recognition accuracy should therefore be less in our method than in [7]. Furthermore, the number of block dilations can be tuned in our algorithm in order to take into account larger displacements, if needed.

As the amount of dilation directions was increased from four to eight, also the recognition accuracy was improved, since the adaptivity of the algorithm was increased. Therefore, eight directions were used in the implementation of the algorithm (N, NE, E, SE, etc.).

4.3. Simulation results

The results of the simulations for $LBP(S, radius)$ are shown in Table 1. They consist of rank one recognition rates, which means that the exact match of the input image and the reference individual is required. The LBP-AM (adaptive matching) is used to denote the proposed occurrence map algorithm. The reference results are the same as in [7, 23], since the same CSU normalization procedure has been used, allowing an objective comparison. The LDA, PCA, EBGM, and Bayesian recognition rates are the standard implementations of the CSU system.

Table 1 shows that the proposed algorithm outperforms the previous LBP-based algorithms, [7, 23] in the recognition accuracy and also the other standard CSU implementations of PCA, LDA, Bayesian MAP, and EBGM for each FERET set. The increase in recognition rate with weighting is approximately 4% compared to [7] and 3% compared to [23]. The improvement in the recognition accuracy without weighting is as much as 14.0% compared to [7]. If the algorithm is to be implemented as an embedded FPGA (digital sampling applied) the recognition rates are not expected to decrease, since mismatch is not present. However, analog sampling exposes the recognition rates for a slight decrease as demonstrated in the later sections.

5. HARDWARE ALGORITHM CODESIGN

For the LBP extraction and face recognition, two different mixed-mode massively parallel hardware architectures are considered. First, a standard CNN-UM can be used which has the advantage of being a flexible general-purpose system allowing larger manufacturing volumes with a lower cost. On the other hand, the performance of the face recognition system in the terms of silicon area, power consumption and speed can be optimized by a dedicated massively parallel hardware implementation. The dedicated implementation requires simultaneous hardware algorithm codesign. Both implementations mentioned above consist of an array of processing cells which operate in mixed mode, that is, carrying out computations based on both digital and analog processings. Some issues in mixed-mode processing have to be carefully taken into account, to ensure that the system functions properly.

5.1. Internal accuracy

Mixed-mode processing is well suitable, for example, for integrated near-sensor processing, where the sensed phenomena are continuously valued. Then there is no need for time-consuming A/D conversion in the data acquisition phase. In practice, the limited internal accuracy of semiconductor devices affects the results of continuously valued array operations. A main source of internal inaccuracy is device mismatch, which affects even basic current thresholding and multiplication operations. The effect of mismatch can be reduced, roughly speaking, by increasing the size of the devices. Therefore, there exists a design trade-off between the processing accuracy and the array resolution, since the number of cells in an array depends on the size of a single cell (assuming a fixed array size). In a dedicated massively parallel implementation, mismatch is used as a design constraint. It should be small enough to ensure correct (enough) operation, but on the other hand, oversizing the cells should be avoided to achieve a proper array resolution.

5.2. *Read-out of results*

Massively parallel mixed-mode systems can gain extremely high computation power, but in addition to the internal accuracy, the read-out of results is usually a performance bottleneck. One alternative is to use embedded A/D converters inside each array cell, but this has the disadvantage of increasing the area of the cells. Another approach is to use a limited number of A/D converters so that the read-out from the cells is performed sequentially. On the other hand, this decreases the overall computation speed. In certain applications, we are not interested in the exact results among each cell, but, for example, the sum of the results among a certain subset of cells. In this case it is possible to use a read-out scheme such as proposed in [24], where a unit current switch is placed into each cell and these are wired together into a common output. The output current is then A/D converted. This functionality is not embedded into a traditional CNN-UM, since it requires, for example, extra wiring, but can relatively easily be implemented on a dedicated massively parallel array.

5.3. *Architectural issues*

The architecture of the massively parallel platform assesses strict limitations on the types of algorithms that can be reasonably mapped on it. Most importantly, since the array consists of identical cells connected usually only on each other in the local 8 neighborhood (e.g., in the case of CNN-UM), only operations that can be defined by a template describing this kind of neighborhood can be executed. In image processing, most of the operations needed can be mapped into a local form. It would be helpful in many applications, if the size of the neighborhood could be increased. However, this causes a need for excessive wiring in the layout of a massively parallel array, and is therefore difficult. Also, a large number of inputs into each cell would increase the cell size and might affect the internal accuracy of the computations. The LBP methodology is relatively well suited on massively parallel processing, since in most applications, neighborhoods with eight to twelve neighbor connections are enough for optimal performance [7].

Sizes of CNN arrays currently in use are sufficient for a large variety of image processing applications, for example, face recognition (a common size of the arrays is becoming to be at least 128×128). If resolution is limited or larger images need to be processed, it is possible to read image data one or a few blocks at a time and process these blocks separately with the CNN or a dedicated array processor. The speed and the complexity of the implemented algorithm, however, are dependent on how many processing cells are available assuming a certain input frame resolution.

6. CNN-UM TEMPLATE ALGORITHMS FOR LBP PROCESSING

In the following, we show either how the CNN-UM can be used only for LBP sampling with a large variety of different potential applications or how it can be used for face recognition with either the histogramming or with the occurrence map method. A sample number of eight is a practical maximum for radius one LBP transform. Consequently, eight logical memories (LLMs) must be allocated for LBPs, since a standard CNN operates in 8 nearest neighborhood.

6.1. *Obtaining the local binary pattern*

In order to extract the local binary patterns, the input image is written to cell input and the state is initialized to zero. S threshold operations are used to generate the S-bit local binary pattern (one bit per direction) and the result of each comparison is stored in an LLM (local logic memory). As a result, LLMs of individual cells contain the LBP pattern in that specific location. The LBP process of comparing sample points one by one to the center point can be modeled with [10],

$$\dot{x}_{ij} = -x_{ij} + a \cdot y_{kl} + \sum_{kl \in N_r} B_{ij,kl} u_{kl}, \tag{4}$$

where a is the center element of the A-template. The cell inputs are denoted with u_{kl} which are multiplied and summed with the B-template. The x_{ij} and $\dot{x}_{ij}$ are the cell state and the rate of change of the cell state, respectively. The center element equals unity and all other elements of A are zero. The output nonlinearity here is the threshold function

$$y_{ij} = \varphi(x_{ij}) = \frac{|x_{ij} + 1| - |x_{ij} - 1|}{2}. \tag{5}$$

The comparison is applied circularly for S different directions.

The bias I is zero for all the following comparison templates. The templates in Table 2 are constructed according to (6) and are used to implement the LBP transform with the radius of one. With the CNN-UM, the LBP radius can be one (or in some cases two) covering most of the practical purposes. Increasing the sample number increases the amount of comparison templates so that the number of comparison template operations is equal to the number of samples S, for one input frame. For example, using a neighborhood of one with a sample number of four, only templates T_2, T_4, T_6, and T_8 are used with a certain input image. Interpolation between two adjacent pixels is shown in the template of (7) with the

Table 2: Threshold templates for LBP with r of one and S of eight.

Template	b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8
T_1	1	−1	0	0	0	0	0	0	0
T_2	1	0	−1	0	0	0	0	0	0
T_3	1	0	0	−1	0	0	0	0	0
T_4	1	0	0	0	−1	0	0	0	0
T_5	1	0	0	0	0	−1	0	0	0
T_6	1	0	0	0	0	0	−1	0	0
T_7	1	0	0	0	0	0	0	−1	0
T_8	1	0	0	0	0	0	0	0	−1

radius of two and sample point direction of threshold template T_3. In this equation, the average of the two pixels with weights of a half is thresholded against the center pixel. Also, a larger area can be used for the interpolating neighborhood in a similar manner. If the radius is two, the algorithm works exactly the same way, with a radius of one, except that the center pixel is now thresholded against a circle with a larger radius,

$$B_{r=1} = \begin{bmatrix} b_1 & b_2 & b_3 \\ b_8 & b_0 & b_4 \\ b_7 & b_6 & b_5 \end{bmatrix}, \tag{6}$$

$$B_{r=2} = \begin{bmatrix} 0 & 0 & 0 & -1/2 & 0 \\ 0 & 0 & 0 & 0 & -1/2 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{7}$$

6.2. *LBP occurrence map generation and histogramming*

The LBP occurrence maps are directly extracted from the results of the thresholding operations. The operations such as dilation and the AND operation are readily available in the CNN-UM.

Pattern matching can be used to construct a histogram of the LBP patterns of an image or a region of an image. One option of creating the histogram of LBPs is to match them using the local logic unit (LLU) of a CNN-UM. In the beginning of the comparison operation, LBPs are stored in S local memories. A logic OR operation is applied bit-serially for the contents of those LLMs that are required to be LO. The result of this is inverted and written into a vacant LLM. Furthermore, a logic AND operation is applied bit-serially for the result of the inversion and the contents of those LLMs that are required to be HI. The result is HI when the LBP matches the required pattern. This operation requires a relatively large number of bit-serial logic operations for each bin of the histogram. For example, if S is 8, and the number of ones in the desired pattern is 3, we need 4 LLU OR-operations (for the bits that are to be zeros), one LLU inversion operation (for the result), and 3 LLU AND-operations (for the bits that are to be ones).

The sum of the ones (one indicating an LBP match) in the binary occurrence maps that result after the subsequent pattern matchings equals the amplitude of the histogram bin of a certain local binary pattern. If the input image fed to the CNN contains only a certain block or blocks of the actual input image, the pattern matching has to be performed for each block or a subset of blocks separately. Then, the decrease in parallelism increases the overall execution time. Assuming that the input frame is processed by P blocks at a time and the image is divided into total of R blocks, the amount of pattern matching and comparison operations is multiplied by R/P. The ratio of R and P can be decided based on the input resolution and speed requirements of the application.

6.3. *Selecting symmetrical and uniform patterns*

The selection of uniform and symmetrical patterns can be carried out by applying the pattern matching only for these specific patterns. A category of LBPs can be unified into a single occurrence map by using a cell specific AND operator applied for the whole array simultaneously. The patterns that are neither uniform nor symmetrical can be concatenated into the same occurrence map by a similar approach, using "do not care" conditions in pattern matching, or by summing the A/D converted results outside the array for these specific LBPs. A similar approach can be used for the dedicated mixed-mode hardware implementation for unifying the occurrence maps, by using AND operator implemented in the neighborhood logic unit.

6.4. *A modified CNN-UM cell for LBP sampling*

A slightly modified CNN cell can perform the pattern matching using a CNN pattern matching template [25]. Figure 5 shows the modified part of a CNN cell with $S = 8$. The first modification is that S LLMs need to be accessible simultaneously so that they can act as multiplier inputs instead of cell input. Whether the inputs to the multipliers come from the memories or from the cell input is programmable via switches. Moreover, the outputs of the multipliers can be programed to be redirected either to neighbors (normal CNN operation) or to the state node of the cell itself. In other words, the multipliers normally used for neighborhood operations are utilized in pattern matching operations.

A 3×3 pattern matching template for second-order uniform LBP of 00111000 is shown in the template of (8). The B template consists of the pattern to be matched so that a minus one corresponds to a white pixel, a plus one to a black pixel, and a zero to a "do not care" condition. The bias z is defined as 0.5-N where N is the number of pixels required to be either black or white,

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} -1 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & -1 \end{bmatrix}, \qquad z = -7.5. \tag{8}$$

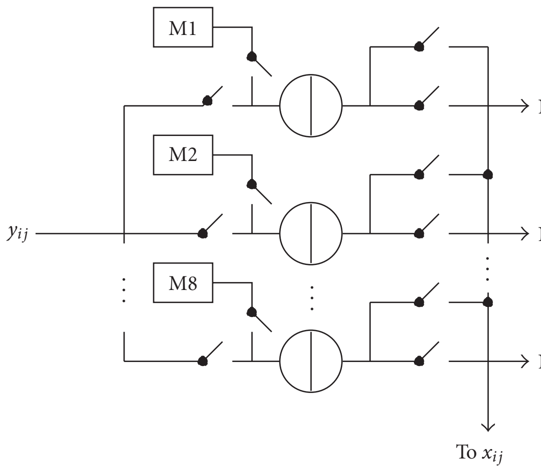


Figure 5: CNN cell modifications.

6.5. *Effect of mismatch on histogram-based face recognition*

Determination of the LBP of a pixel requires a comparison of the pixel value to its neighbors. Comparison can be performed easily in the analog domain by subtracting two currents from each other. However, since the comparison is performed in the analog domain, the mismatch of the analog devices corrupts the result. We chose to simulate the effect of mismatch in the face recognition with histogramming.

We carried out simulations as to how much analog mismatch is tolerated in the comparison operation. The standard FERET database [9] of different sets of facial images normalized by the CSU system [8] was used. The simulations were carried out so that prior to determining the feature vector of an image, normally distributed noise generated with Matlab was added to the image so that a noisy pixel value becomes

$$P_{i,j;\text{noisy}} = P_{i,j} + n_{i,j} \cdot \frac{\%}{100} \cdot 255, \tag{9}$$

where $n_{i,j}$ is an element of a zero mean, standard deviation one, normally distributed noise matrix, and 255 is the maximum pixel intensity. Table 3 shows the results of the face recognition simulation with different percentages of mismatch. A radius of two with $M = 8$ was used. The results show that the mismatch affects mostly those images that were taken in different illumination conditions. The results shown in Table 3 were simulated so that all images, both the reference images and the probe images in fb, fc, dup1, and dup2, were corrupted by noise. If noise was only added to the probe images, the degradation in the average recognition accuracy was about half of that shown in Table 3.

6.6. *Effect of quantization in analog read-out*

We also examined the effect of quantization in the analog read-out on the recognition accuracy in Table 4. We chose to use feature extraction without weighting and used the histogram method [7]. With the block size that we used, the maximum dynamic range was between zero and 378 ($21 * 18$). With the occurrence map algorithm, the block size is larger, but also the magnitude of the sum of ones after the comparison phase is much smaller due to the coding into occurrence maps. We noticed that the bin amplitudes of more than 255 corresponding to eight-bit accuracy had no effect on the recognition accuracy. The upper limit of the dynamic range was then divided by a scaling factor, for example, four when six-bit accuracy was used. In practice the recognition rate did not decrease even with seven-bit accuracy. With six-bit accuracy, the average recognition result was decreased only by 0.5%. Six- or seven-bit accuracy can rather easily be reached by the ADC. With five-bit accuracy, the effect on recognition rate was larger resulting in a total decrease of 3.3%. This means that summing up the currents from all cells in a block and converting with a 7-bit A/D converted give practically an unaltered performance.

Table 3: Mismatch effects for histogramming with weighting.

Mismatch %	fb	fc	dup1	dup2	Average
0	97%	80%	66%	64%	77%
1%	96%	73%	66%	65%	75%
2%	95%	69%	64%	58%	71%
3%	94%	63%	62%	57%	69%
5%	92%	55%	57%	49%	63%

Table 4: FERET histogram-based face recognition results with quantization.

Method	fb	fc	dup1	dup2	Average
LBP optimal nonweighted $r = 2, M = 8$	93%	52%	61%	49%	63.8%
LBP 6 bits nonweighted $r = 2, M = 8$	93%	49%	61%	50%	63.3%
LBP 5 bits nonweighted $r = 2, M = 8$	92%	47%	56%	47%	60.5%

7. A DEDICATED LBP HARDWARE

A dedicated hardware architecture for the adaptive occurrence map matching algorithm consists of external memory, a massively parallel processor array (including integrated current-mode imaging sensors), a control unit, and a memory interface which includes a cache. The organization of these units is illustrated in Figure 6.

The size of the external memory depends on the number of stored face feature vectors. With reference face database of 100 images, the size of the external memory becomes approximately 9.76 Mbits or 1.22 Mbytes (100 images $*$ 150 $*$ 130 pixels $*$ 5 bit/pixel). The memory interface performs the

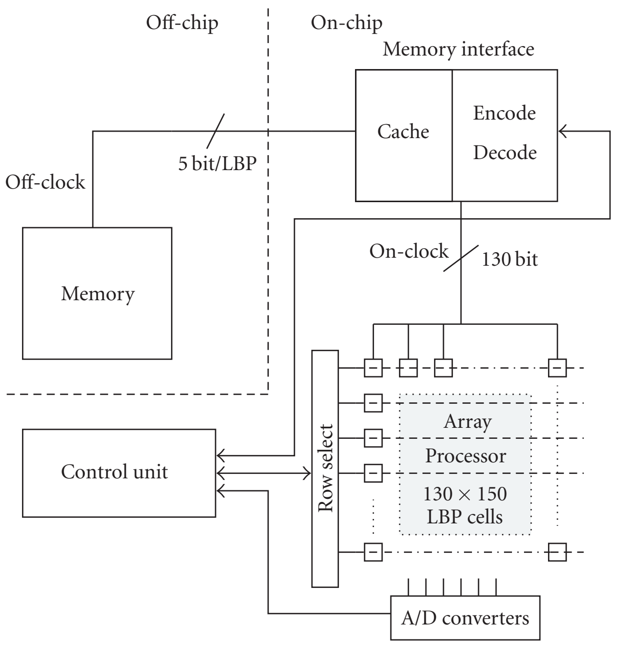


Figure 6: Architecture of the recognition system.

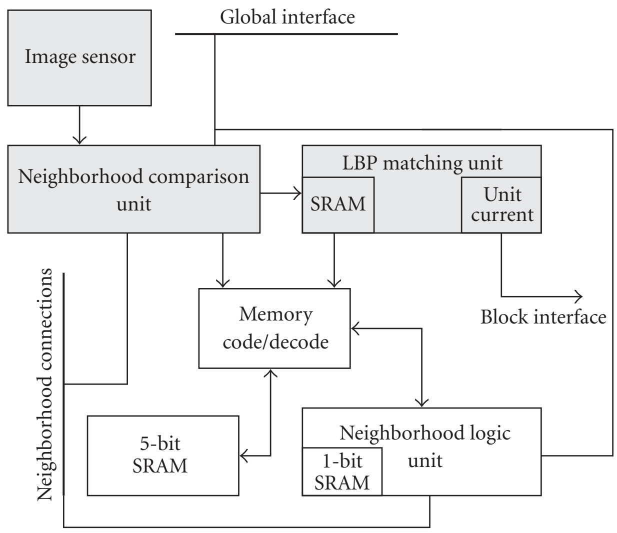


Figure 7: LBP processing cell for the adaptive matching algorithm.

decoding of LBP images to binary occurrence maps which are used by the array processor. We propose using a cache of one feature vector (reference face) which results the size of approximately 97.5 kbits or 12.2 kbytes. The cache is integrated on the same chip with the array. The array processor itself performs only binary operations, in addition to the LBP feature extraction (sampling). The control unit is integrated on the same chip and sends instructions and receives data to/from the array processor and to/from the memory interface. The control unit also processes the (A/D converted) results from the array.

7.1. A dedicated LBP processing cell

The dedicated LBP processing cell in Figure 7 can be used only for LBP sampling and possibly histogram generation, or it can be used for the dedicated occurrence map algorithm. The parts of the LBP cell which are targeted for LBP sampling are highlighted with gray. The full LBP cell for the face matching algorithm based on the occurrence maps includes CMOS image sensor, instruction code/decode unit, neighborhood comparison unit, LBP matching unit, memory decode/code unit, SRAM memory, and neighborhood logic unit. The neighborhood comparison unit, image sensor, and the LBP matching unit form the architecture of the general dedicated LBP processing hardware. Therefore, they are discussed in more detail in the later sections.

The neighborhood comparison unit performs the LBP sampling with a radius of two using eight samples (four of which are interpolated from their neighbors' values). The functionality and implementation of this unit are explained later in this section. The matching unit performs comparison of an LBP stored into the cell with a certain specific globally selected 8-bit LBP. If an exact match is recorded, the matching unit activates the current source specific to that cell, which is routed to the block interface. The block denotes a certain subset of LBP cells for which the weighting and A/D conversion are applied later. The memory code/decode unit performs coding of the LBP occurrence maps into a binary weighted form. For example, the binary weighted representation for a maximum of 32 occurrence maps consists of 5 bits which are needed for the local memories to store the input face. For each cell, the occurrence map indicates whether a specific LBP occurs in that cell or not. This unit also decodes the occurrence maps from the 5-bit SRAM memory into a total of 30 occurrence maps. The neighborhood logic unit is used for the directional dilation operation for specific directions. A 1-bit SRAM is included into the neighborhood logic unit for storing one occurrence map of the other face image under comparison. This occurrence map is read from the on-chip memory interface, as the full LBP image stored into the 5-bit SRAM (the input image) can be read using the on-chip sensors.

7.2. Implementation of the neighborhood comparison unit

Figure 8 shows the neighborhood comparison unit. The current I_cell feeds the input current of the pixel. It could be obtained, for example, from an in-cell image sensor or a D/A converter. Transistors $M1$–$M6$ are analog transistors and the rest of the transistors shown are minimum-sized switches. The neighborhood comparison unit has 12 connections to neighbors on the circle of radius 2. The neighbors are coupled so that when, for example, control signal N is active, the unit receives input current from a cell two rows above and conveys its current to the cell two rows below (see the indexes in the figure). Notice that the current of transistor $M4$ is mirrored to transistors $M5$ and $M6$ at a ratio of 0.5. This

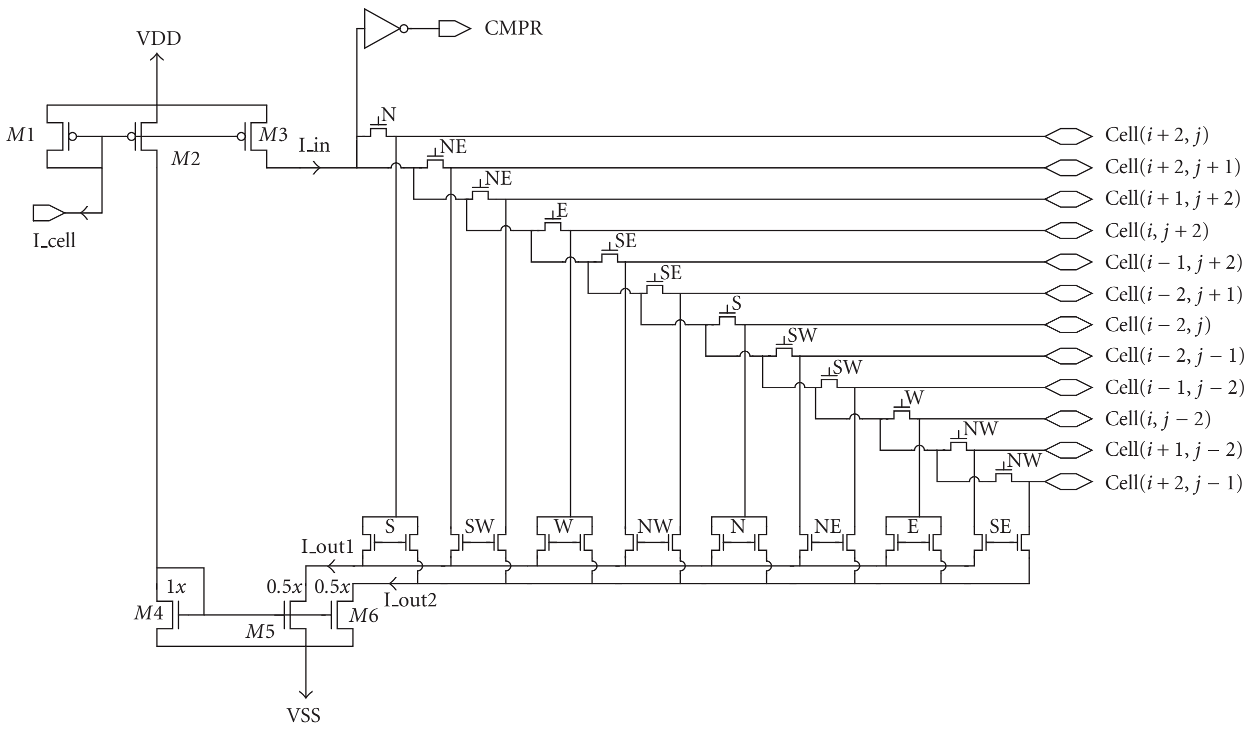


Figure 8: Neighborhood comparison unit.

is because when processing directions NE, NW, SE, and SW the cell should receive inputs from two different neighbors, the currents of which are interpolated. An inverter serves as a comparator and the comparison result is available at *CMPR*.

7.3. Implementation of the LBP matching unit

Figure 9 shows the proposed circuit for matching LBPs within the pixels in order to generate the occurrence map representation. The comparator output *CMPR* is fed to eight SRAMs. All these are read out simultaneously and they provide either an inverted or noninverted output. After the comparison circuit has written the LBP into the SRAMs, the matching using the programmable multi-input NOR is started. Notice that the SRAM and the NOR share eight control signals (N, NE, E, SE, S, SW, W, and NW) with the neighborhood comparison unit. These control signals can be used to program which pull-down paths of the NOR are possible. The weak pull-up transistor is turned on with *BIAS* during evaluation.

The matching is a two-phase process with the proposed circuit. First, the SRAMs provide the NOR with the inverted LBP. The control bits associated with LBP bits that are required to be HI are taken HI. In other words, only selected pull-down paths are enabled. Now, if all inverted LBP bits that are connected to activated pull-down paths in the NOR are LO, the result is HI. This is written to an SRAM in the output unit through an inverter. Second, the noninverted LBPs are fed to the NOR, the control bits enable the pull-down paths associated with those LBP bits that are required to be LO and the result is fed to the lower NOR input in the output unit. If the LBP matches the pattern under search, both inputs of the NOR in the output unit are zero and the gate voltage of analog output transistor M_OUT is HI.

7.4. Optimizing the performance of the LBP cell

In order to get an idea of how large the transistors $M1$–$M6$ of the neighborhood comparison unit should be, simulations were performed. The simulations were carried out with Eldo level 53 parameters of a 0.13 um CMOS process. Ideally, the standard deviation of the difference current would be zero. Figure 10 shows the standard deviation of the difference current divided by 2.5 μA (in percents) for two different combinations of the sizes of the analog transistors. The upper curve was simulated with the transistor $M4$ and the combination of $M5$ and $M6$ sized to 0.75/6, while the PMOS transistors were sized to 1/4 (sizes are in micrometers). The corresponding transistor sizes for the lower curve were 1/8 and 1.5/6, respectively. The standard deviations were determined from 50 Monte Carlo iterations. If the intensity of the pixel would be represented with currents ranging from zero to 2.5 μA, the standard deviation would be around one percent with the larger transistors (see Table 3).

8. LBP PROCESSING PERFORMANCE

Embedding the LBP sampling with the recognition system allows a compact integrated solution for face recognition, with either a CNN-UM or a dedicated massively parallel hardware. The sampling of LBPs is relatively fast even without a massively parallel hardware [7] and, in a practical view, the

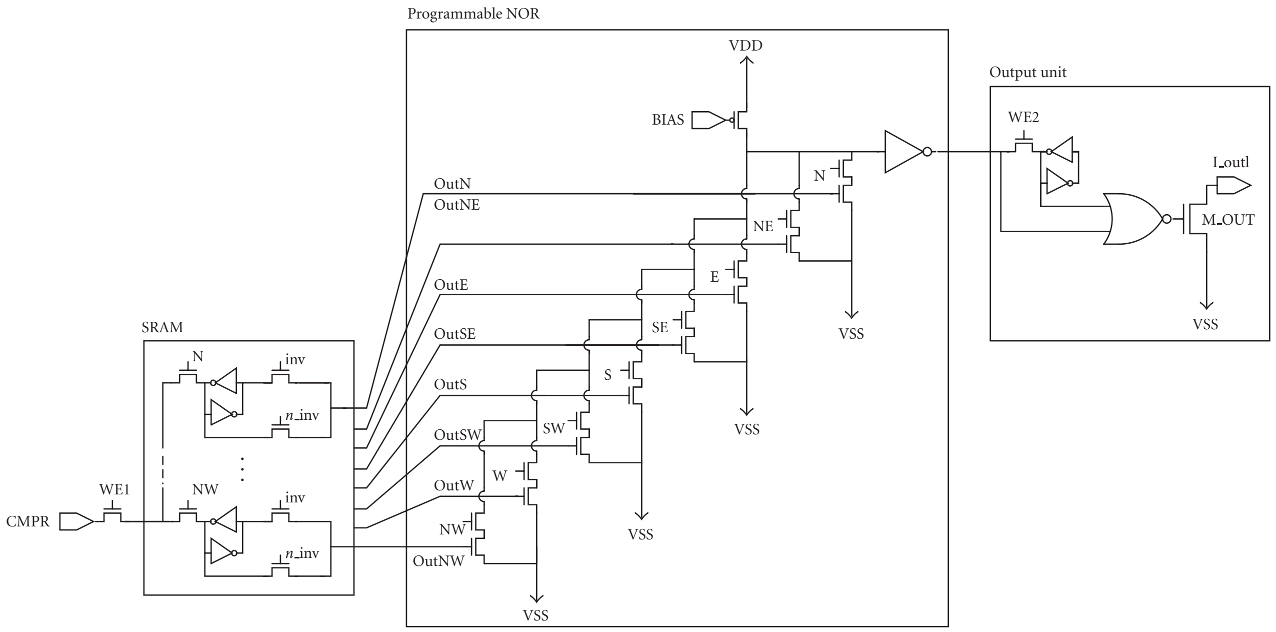


Figure 9: LBP matching unit.

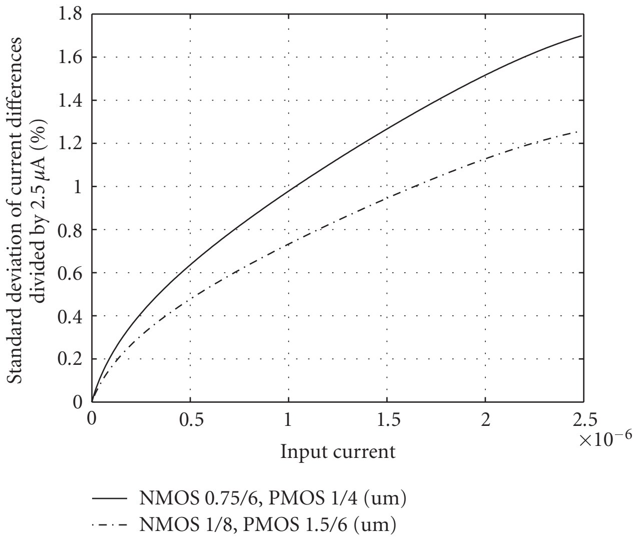


Figure 10: Effect of mismatch on the sampling accuracy.

accuracy improvements in the occurrence map-based face matching are more significant than the improvement in the sampling phase, which however, could be taken as an advantage in other applications.

8.1. Performance of the CNN-UM histogramming

Using the template analyzer [26] the estimated execution time for each of the comparison templates is 2.00τ and 2.88τ for pattern matching templates (τ being the CNN time constant). Excluding the memory delays and template programming delays, the total time used by the threshold templates is 16τ (assuming a sample number of eight). If all the 256 patterns are to be extracted, the total time for pattern matching operations will become 737.3τ ($2.88 * 256\tau$) assuming that all the blocks are processed in a single frame. However, it was mentioned that it is possible to use a "do not care" condition to reduce the amount of pattern matching operations when, for example, only uniform and symmetrical patterns are used which will further reduce this time.

If the input image is read directly from in-cell sensors, early A/D conversion can be avoided. Assuming that there is a total of 30 symmetrical uniform patterns corresponding to bins and that the image is divided into blocks as proposed, the total amount of successive A/D conversions will be 30 if it is assumed that there is a peripheral A/D converter for each block. The D/A conversion is implemented automatically by summing of binary unit currents from the cells. The integration of the A/D converters will not cause problems since we have shown that the accuracy requirements for the converters are not severe (see Table 4). If the time for a single A/D conversion is 0.5 μs, the total time for conversion will be 15 μs.

By using τ of 1 μs, the total time for LBP sampling becomes 768.3 μs (1302 faces/s). As a reference, for the same images that we used in the simulations, the LBP feature vector extraction time for one face image for AMD Athlon 1800 MHz was 3.49 ms (285 faces/s) with weighting (blocks outside face are ignored) and 4.14 ms (240 faces/s) without weighting [7].

8.2. Performance of the dedicated mixed-mode algorithm

Using the read-out scheme proposed in [24], there will be a need for executing $D * M$ A/D conversions (in parallel for each block) at each comparison where D is the number of occurrence maps and M is the number of search directions. Let us assume the elapsed time for the parallel A/D conversions being approximately 0.5 microseond. With a reference image database of 100 images, this results the elapsed time of 12 ms for comparing a single input image with all the stored reference images (assuming D of 30 and M of 8). The total comparison time for one stored reference image would be about 120 μs. This could be further improved by searching the block specific optimal dilation direction only for certain patterns and calculating the comparison result then for the best direction estimate. In [27] it was demonstrated that with the binary programmable CNN, the measured propagation time (for dilation) was short, only 16.3 ns with 3.3 μA being unit current and 0.7 V being operating voltage.

When estimating the overall performance of the system, the external (off-chip) memory speed and bandwidth need to be considered. We propose using 40 data pins from the chip to interface with the memory. The on-chip memory interface encodes this data to a total of 130 lines which are routed to the cell array. The memory structure of the cell array consists of row and column decoders, which are used to select one row at a time from the 130×150 array for writing. If the external memory would operate with a 7 MHz off-chip clock, the total bandwidth through the 40 pins becomes 280 Mbits/s. With a 100 image, database, this would result 0.035 s (29 fps) elapsed when scanning through the whole memory. For a single image the reading time from the off-chip memory would become about 348 μs. Since a cache is used, the occurrence map matching can be performed simultaneously with the memory read operations. The on-chip clock of 15 MHz is enough for synchronizing the 130 lines routed to a certain array row through the column decoder. This would result an elapsed time of 0.03 s for processing through the whole external memory of 100 faces. For a single face the reading time from cache would be about 300 μs. As a conclusion, the system could operate beyond 25 recognized faces per second with a 100 image reference database. If the database was larger, say 1000 faces, it can be estimated that the time used for recognizing a face would be about 0.35 s.

9. DISCUSSION

Power supply, clock generation, and user interface as well as possible camera and camera interface are needed for the dedicated face recognition system to function. Integrating the image sensors will reduce power consumption, since direct A/D conversion is not needed in the imager. Furthermore, the read-out of face matching results can be performed in parallel with a dedicated hardware for each block with small unit currents, which should be more efficient compared to, for example, an FPGA implementation. Whether an FPGA, a CNN-UM, or a dedicated hardware gives the best performance depends on the targeted application and it depends on the final application whether the improvement on the recognition accuracy is worth implementing a special hardware, or whether to use a general purpose computer hardware with standard LBP histogramming. For applying the occurrence map algorithm, a massively parallel hardware seems the only practical alternative.

The occurrence map representation of LBPs with the capability for adaptive block search could also be benefited in other applications beyond face recognition. In LBP-based motion analysis [20], the occurrence maps would be an elegant way to search, for example, motion specific to a certain direction from the image scene.

In the following, we address two different scenarios where the system could be used as such or with some external hardware and software. First, controlling of people transpassing through a security check. In this scenario the recognition time for one face could be while waiting, say seconds, and the database of stored faces would be larger, say 1000 images. A separate face detection and normalization system would not necessarily be required. In another scenario, an individual among a mass of people would be searched without necessarily informing the individual of the surveillance. Then a separate face detection and normalization system would be needed.

10. CONCLUSIONS

This paper described a framework for implementing a massively parallel face recognition system. The system consists of three parts, which are mixed-mode or CNN-UM-based LBP sampling method, a massively parallel face recognition algorithm, and a dedicated mixed-mode hardware for the proposed algorithm. The LBP sampling process of CNN-UM or a dedicated hardware utilizing current-mode operation can be generalized also for other high-speed LBP applications beyond the face recognition. An example of a potential application where the sampling speed would be important is industrial paper quality inspection using computer vision and local binary patterns [28].

The adaptive face matching algorithm could also be implemented as an embedded FPGA implementation. The implementation of the neighborhood logic unit, memory code/decode unit, and instruction code/decode unit, which has been so far considered only in the architectural level, would then be relatively straightforward since digital logic design could be used. Also, the mismatch would not cause any significant decrease to the recognition accuracy if the LBP sampling was performed digitally.

The dedicated massively parallel face recognition algorithm was shown to perform accurate face recognition, with a maximum increase in the recognition accuracy with weighting of 4% compared to [7] and 3% compared to [23]. The improvement in the recognition accuracy without weighting was as much as 14.0% compared to [7]. Furthermore, the face recognition algorithm is adaptively tolerant to slight changes in face orientation and scaling.

ACKNOWLEDGMENT

This work was funded by the Academy of Finland Projects no. 205746 and no. 106451.

REFERENCES

[1] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[2] A. M. Martinez and A. C. Kak, “PCA versus LDA,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228–233, 2001.

[3] J. Yang, D. Zhang, A. F. Frangi, and J.-Y. Yang, “Two-dimensional PCA: a new approach to appearance-based face representation and recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 131–137, 2004.

[4] C. J. C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[5] L. Wiskott, J.-M. Fellous, N. Krüger, and C. D. von Malsburg, “Face recognition by elastic bunch graph matching,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 775–779, 1997.

[6] J. Ruiz-del-Solar and P. Navarrete, “Eigenspace-based face recognition: a comparative study of different approaches,” *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 35, no. 3, pp. 315–325, 2005.

[7] T. Ahonen, A. Hadid, and M. Pietikäinen, “Face recognition with local binary patterns,” in *Proceedings of 8th European Conference on Computer Vision (ECCV ’04)*, vol. 3021 of *Lecture Notes in Computer Science*, pp. 469–481, Prague, Czech Republic, May 2004.

[8] D. S. Bolme, J. R. Beveridge, M. Teixeira, and B. A. Draper, “The CSU face identification evaluation system: its purpose, features, and structure,” in *Proceedings of 3rd International Conference on Computer Vision Systems (ICVS ’03)*, pp. 304–313, Graz, Austria, April 2003.

[9] P. J. Phillips, H. Wechsler, J. Huang, and P. J. Rauss, “The FERET database and evaluation procedure for face-recognition algorithms,” *Image and Vision Computing*, vol. 16, no. 5, pp. 295–306, 1998.

[10] L. O. Chua and L. Yang, “Cellular neural networks: theory,” *IEEE Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1257–1272, 1988.

[11] T. Roska and L. O. Chua, “The CNN universal machine: an analogic array computer,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 40, no. 3, pp. 163–173, 1993.

[12] G. L. Cembrano, A. Rodríguez-Vázquez, R. C. Galán, F. Jiménez-Garrido, S. Espejo, and R. Domínguez-Castro, “A 1000 FPS at 128 × 128 vision processor with 8-bit digitized I/O,” *IEEE Journal of Solid-State Circuits*, vol. 39, no. 7, pp. 1044–1055, 2004.

[13] A. Paasio, A. Kananen, K. Halonen, and V. Porra, “A QCIF resolution binary I/O CNN-UM chip,” *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 23, no. 2, pp. 281–290, 1999.

[14] O. Lahdenoja, M. Laiho, and A. Paasio, “Local binary pattern feature vector extraction with CNN,” in *Proceedings of the 9th IEEE International Workshop on Cellular Neural Networks and Their Applications (CNNA ’05)*, pp. 202–205, Hsinchu, Tawian, May 2005.

[15] O. Lahdenoja, J. Maunu, M. Laiho, and A. Paasio, “A massively parallel algorithm for local binary pattern based face recognition,” in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS ’06)*, p. 4, Kos, Greece, May 2006.

[16] M. Laiho, O. Lahdenoja, and A. Paasio, “Dedicated hardware for parallel extraction of local binary pattern feature vectors,” in *Proceedings of the 9th IEEE International Workshop on Cellular Neural Networks and Their Applications (CNNA ’05)*, pp. 27–30, Hsinchu, Tawian, May 2005.

[17] T. Ojala, M. Pietikäinen, and D. Harwood, “A comparative study of texture measures with classification based on feature distributions,” *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[18] T. Mäenpää, “The local binary pattern approach to texture analysis—extensions and applications,” Dissertation, University of Oulu, Oulu, Finland, 2003.

[19] T. Ojala, M. Pietikäinen, and T. Mäenpää, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[20] M. Heikkilä and M. Pietikäinen, “A texture-based method for modeling the background and detecting moving objects,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 657–662, 2006.

[21] A. Hadid, M. Pietikäinen, and T. Ahonen, “A discriminative feature space for detecting and recognizing faces,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR ’04)*, vol. 2, pp. 797–804, Washington, DC, USA, June-July 2004.

[22] T. Mäenpää, T. Ojala, M. Pietikäinen, and M. Soriano, “Robust texture classification by subsets of local binary patterns,” in *Proceedings of 15th International Conference on Pattern Recognition (ICPR ’00)*, vol. 3, pp. 935–938, Barcelona, Spain, September 2000.

[23] O. Lahdenoja, M. Laiho, and A. Paasio, “Reducing the feature vector length in local binary pattern based face recognition,” in *Proceedings of IEEE International Conference on Image Processing (ICIP ’05)*, vol. 2, pp. 914–917, Genova, Italy, September 2005.

[24] P. Dudek, “A flexible global readout architecture for an analogue SIMD vision chip,” in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS ’03)*, vol. 3, pp. 782–785, Bangkok, Thailand, May 2003.

[25] CNN Software Library, Ver. 1.1, Chapter 3, http://lab.analogic.sztaki.hu.

[26] CNN Template Analysator, TemInfo, http://lab.analogic.sztaki.hu.

[27] J. Flak, M. Laiho, A. Paasio, and K. Halonen, “VLSI implementation of a binary CNN: first measurement result,” in *Proceedings of the 8th IEEE International Workshop on Cellular Neural Networks and Their Applications (CNNA ’04)*, p. 129, Budapest, Hungary, July 2004.

[28] M. Turtinen, M. Pietikäinen, and O. Silvén, “Visual characterization of paper using isomap and local binary patterns,” *IEICE Transactions on Information and Systems*, vol. E89-D, no. 7, pp. 2076–2083, 2006.