Hiroshima University

Research Center for Nanodevices and Systems (RCNS)
System Design and Architecture Research Division

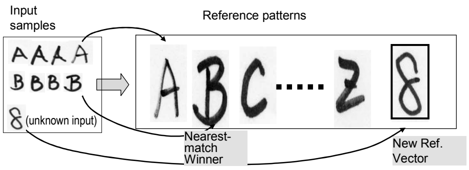
A Human-memory Based Learning Model and Hardware Prototyping in FPGA

A. Ahmadi, M.A. Abedin, H.J. Mattausch, T. Koide, Y. Shirakawa, and M.A. Ritonga
Research Center for Nanodevices and Systems, Hiroshima University

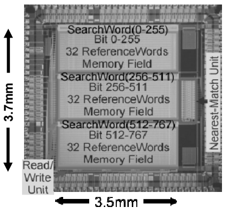
Research Objectives

An Associative-memory based model with Qualifications:

1. **Learning capability for recognition of new samples (handwritten & printed)**
2. **Recognition speed of 10ms per word**
3. **Robustness to Noise, Rotation, Color**



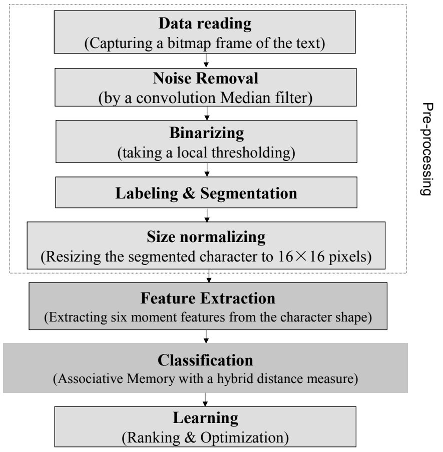
To enhance the pattern matching speed, the classification process is designed on the basis of using a parallel associative memory.



The fully parallel Associative memory designed in our lab using a Hamming distance measure.

Processing Steps

Block diagram of the whole processing steps


Data reading
(Capturing a bitmap frame of the text)

Noise Removal
(by a convolution Median filter)

Binarizing
(taking a local thresholding)

Labeling & Segmentation

Size normalizing
(Resizing the segmented character to 16×16 pixels)

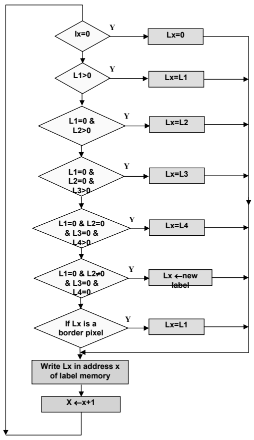
Feature Extraction
(Extracting six moment features from the character shape)

Classification
(Associative Memory with a hybrid distance measure)

Learning
(Ranking & Optimization)

pixel label)

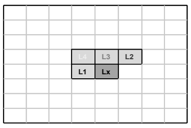
Labeling



(Ix : pixel density (0 or 1)   Lx : pixel label)

0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0
0	0	1	0	1	0	1	0	1	0	0
0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0

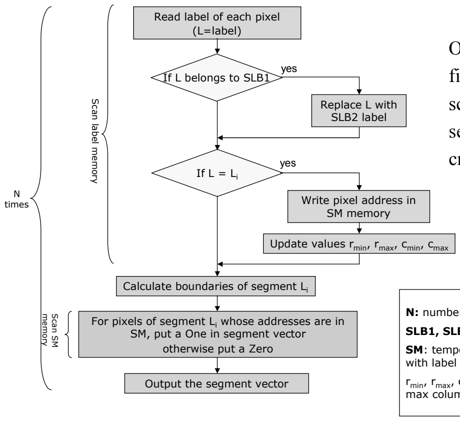
0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	2	0	0	0	3	0
0	0	1	0	2	0	2	0	3	0	0
0	0	0	2	0	0	0	3	0	0	0
0	0	0	0	0	0	0	0	0	0	0



SLB1		SLB2
1	>>	2
2	>>	3

We scan all the input frame pixels sequentially while we keep the labels of preceding neighboring pixels (4 pixels' labels) in 4 registers to determine the current pixel's label.

Segmentation


Read label of each pixel (L=label)
If L belongs to SLB1
yes
Replace L with SLB2 label
If L = L_i
yes
Write pixel address in SM memory
Update values r_{min}, r_{max}, c_{min}, c_{max}
Calculate boundaries of segment L_i
For pixels of segment L_i whose addresses are in SM, put a One in segment vector otherwise put a Zero
Output the segment vector
Scan label memory
Scan SM memory
N times

Once the labeling process is finished, the label memory is scanned once again for segmentation of distinct labels created in the labeling process.

N: number of separate labels (L_i , i=1…N)

SLB1, SLB2: buffers for recording equivalent labels

SM: temporary memory for saving addresses of pixels with label L_i

r_{min}, r_{max}, c_{min}, c_{max}: min row, max row, min column, max column of segment L_i

Feature Extraction

For each segmented character following moment-based features are selected so that they have the minimum dependency on size and variation of data.

- ***Total mass*** *(number of pixels in a binarized character)*
- ***Centroid*** *(determined by an averaging on horizontal and vertical projection of character)*
- *Elliptical parameters*
  - ***Eccentricity*** *(ratio of major to minor axis)*
  - ***Orientation*** *(angle of major axis)*
- ***Skewness***

In principle skewness is defined as the third standardized moment of a distribution as

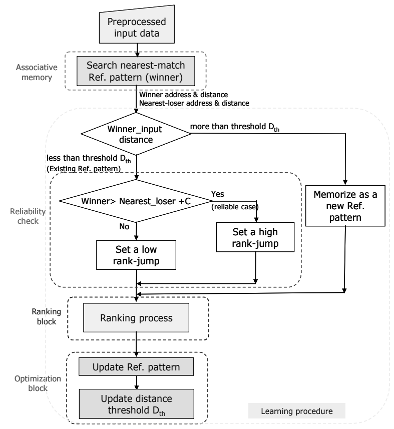
$$\gamma = \mu_3 / \sigma^3$$

but to simplify the calculations we take a simpler measure of Karl Pearson defined as

$$\gamma = 3\ (\text{mean} - \text{median}) / \text{standard deviation}$$

and calculate horizontal and vertical skewness, separately.

Learning Algorithm

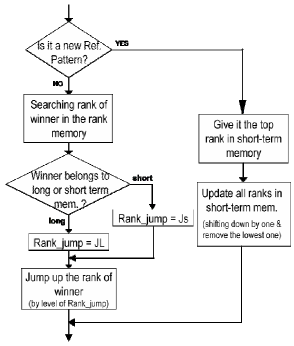


The core concept of learning process is based on a short/long term memory which is very similar to memorizing procedure in the human brain. The transition of reference patterns between short-term and long-term storages is carried out by means of a ranking algorithm.

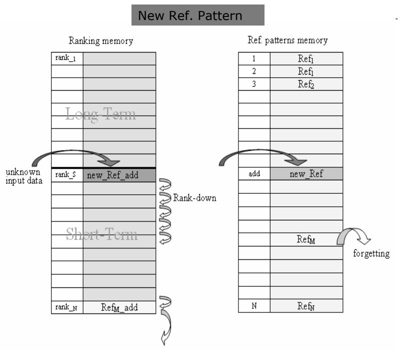
D_{w-i} : Winner-Input distance
D_{th} : Threshold value for D_{w-i}
D_{n-I} : Nearest-Loser to Input distance
J_L : Rank-jump value in long-term mem.
J_S : Rank-jump value in short-term mem.
C : Constant value

Ranking Process

Each reference pattern is given an "unique rank" showing the occurrence level of the pattern. The rank is increased by a dynamic jump value in case of a new occurrence and reduced gradually when there is no occurrence of matching and other reference patterns getting higher ranks.

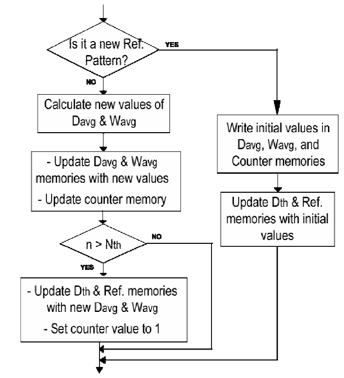


Is it a new Ref. Pattern?
YES
NO
Searching rank of winner in the rank memory
Winner belongs to long or short term mem. ?
short
long
Rank_jump = Js
Rank_jump = JL
Jump up the rank of winner
(by level of Rank_jump)
Give it the top rank in short-term memory
Update all ranks in short-term mem.
(shifting down by one & remove the lowest one)



Optimization Process

Optimization includes a dynamic process for updating of reference patterns magnitude as well as distance thresholds by using the data of coming input samples and taking an averaging technique.



Is it a new Ref. Pattern?
YES
NO
Calculate new values of Davg & Wavg
- Update Davg & Wavg memories with new values
- Update counter memory
n > Nth
NO
YES
- Update Dth & Ref. memories with new Davg & Wavg
- Set counter value to 1
Write initial values in Davg, Wavg, and Counter memories
Update Dth & Ref. memories with initial values

Updating Steps:

1- Updating Ranks

(ranks are used to control the life-time of each Ref. pattern in the memory)

2- Updating Ref. Pattern magnitude

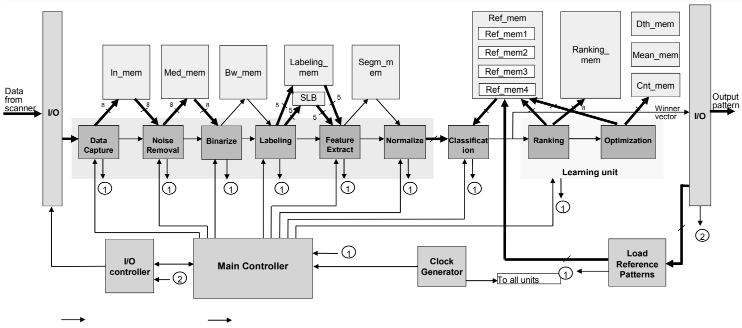
(used for optimizing the classification)

3- Updating Distance Thresholds (D_{th})

(used to optimize the classification reliability as well as generation of new Ref. patterns)

Hardware Prototyping

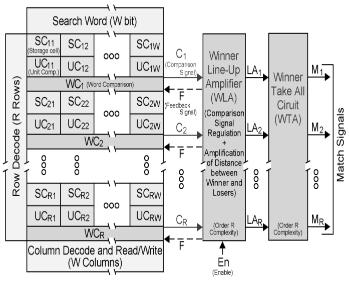
The system was implemented on an FPGA platform taking following steps: *Verilog HDL design*, *design synthesis*, *functional simulation*, *route & placement*, *timing analysis & simulation*, and *FPGA programming*.



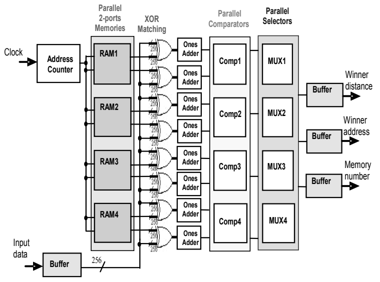
Block diagram of the system architecture. The system has a pipelined structure in both level of processing blocks and processing elements within each block, to provide a maximum capability of parallel processing.

Associative Memory Model

As for implementation of classification block and in order to model the fully-parallel functionality of the associative memory to be used as the main classifier of the system, we applied four dual-port SRAM memory blocks each contains 32 data words in the FPGA.



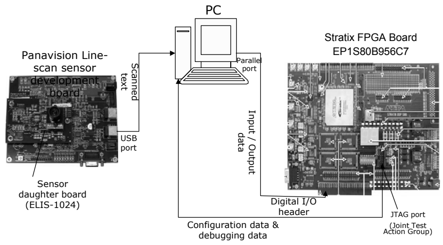
Architecture of associative memory already designed as an LSI chip.



Schematic of classification block mapped with 4 dual-port RAM blocks in an FPGA architecture. 16 matchings are performed within one clock cycle.

FPGA Programming

The whole system was fitted into an FPGA platform without need to external resources. An average clock frequency of 20 MHz is selected for all processing blocks, however a higher clock speed is also applicable.


PC
Panavision Line-scan sensor development board
Scanned text
USB port
Parallel port
Input / Output data
Stratix FPGA Board EP1S80B956C7
Sensor daughter board (ELIS-1024)
Digital I/O header
Configuration data & debugging data
JTAG port (Joint Test Action Group)

Fitter Status	Successful
Quartus II Version	4.2 Full Version
Top-level Entity Name	OCR
Family	Stratix
Device	EP1S80B956C7
Timing Models	Final
Total logic elements	15,451 / 79,040 (19 %)
Total pins	281 / 692 (40 %)
Total virtual pins	0
Total memory bits	2,854,174 / 7,427,520 (38 %)
DSP block 9-bit elements	28 / 176 (15 %)
Total PLLs	1 / 12 (8 %)
Total DLLs	0 / 2 (0 %)

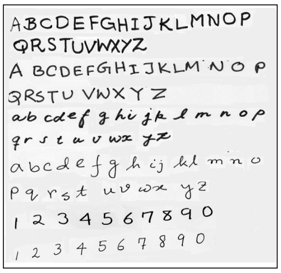
An Altera Stratix family FPGA device (EP1S80) is applied for hardware prototyping

Resource usage for the whole system including preprocessing (excepting feature extraction), classification, and learning blocks, implemented in an Altera Stratix FPGA.

Experimental Results: Classification (1)

The system performance was evaluated by using real data samples. A number of 35 datasets (832 samples) of English characters written by four different writers were used for the experiments.

Handwritten samples used as input data



Classification results for four datasets from different writers during the learning period (taking no initial ref. patterns).

Writer	**Dataset A** (52 samples)		**Dataset B** (52 samples)		**Dataset C** (52 samples)		**Dataset D** (52 samples)	
	Mis classify	New Ref. added	Mis classify	New Ref. added	Mis classify	New Ref. added	Mis classify	New Ref. added
1	9	42	12	25	11	19	19	14
2	11	36	9	26	16	19	10	31
3	10	36	6	20	10	21	12	12
4	11	39	14	20	13	22	5	20
Total (%)	19.7	73.6	19.7	43.8	24	38.9	22.1	37

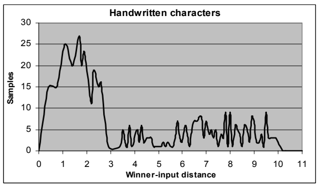
The number of patterns added as new references as well as the misclassification rate is high in the beginning of the process but when the learning goes on and system adjusts continuously the reference pattern memory and distance thresholds, the misclassification rate reduces dramatically.

Experimental Results: Classification (2)

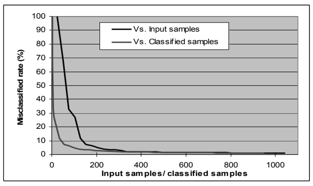
Classification results for two test datasets after a period of learning.

Writer	**Test Set 1** (26 samples)		**Test Set 2** (26 samples)	
	Mis classify	New Ref. added	Mis classify	New Ref. added
1	0	0	2	5
2	1	0	1	6
3	1	1	3	7
4	1	0	2	4
Total (%)	2.8	0.9	7.4	20.4

The average misclassification rate is around 5% which is reasonable for this type of application.



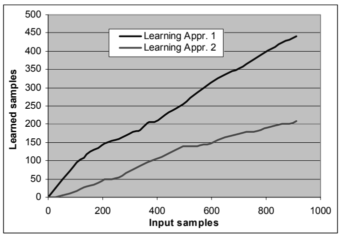
Histogram of winner-input distance.



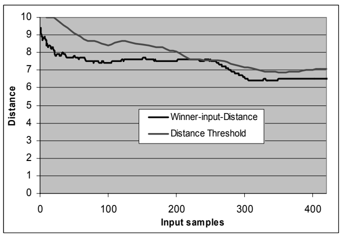
Changes in the misclassification rate during the learning process.

Experimental Results: Learning

Results of Learning speed



Learning speed in the experimental tests on handwritten data.



Distribution of the average of winner-input distance and threshold values (D_{th}).

By learned pattern we refer to reference pattern which is already entered in the long-term memory. In approach 1 all the new Ref. patterns are entered in the short-tem memory first, while in approach 2 if the long-term memory is not yet full, the new Ref. pattern will be located in an unoccupied address of long-term memory.

Conclusions

- **A learning model based on a short/long-term memory and an optimization algorithm for constantly adjusting the reference patterns was proposed.**
- **The whole system blocks excepting feature extraction part were implemented in an FPGA platform and system performance was evaluated with a simulation program using real data of handwritten characters.**
- **In order to enhance the search speed in the classification block, we are planning to use a fully parallel associative memory implemented in an LSI architecture, as the main classifier of the system.**
- **Comparing to other learning models, however this prototype model is not yet robust enough but is advantageous in terms of a simple learning algorithm, high classification speed, and a hardware-friendly structure.**