論文94-31B-6-12

고속 문자 인식을 위한 특징 추출용 칩의 구현

(Implementation of a Feature Extraction Chip for High Speed OCR)

金 赫 九*, 姜 仙 美**, 金 悳 鎭*

(Hyeog Gu Kim, Sun Mee Kang and Duck Jin Kim)

要 約

고속 문자 인식기의 구현을 위한 고속 특징추출 알고리즘을 제안하고 전용칩을 구현하였다. 오프라인 문자 인식의 전 과정을 소프트웨어에 의하여 순차적으로 처리할 경우 고속 문서 인식기의 구현에는 문제점이 있다. 따라서 인식의 각 단계를 기능적으로 분리하여 파이프라인 처리를 통해 시간적인 병렬성을 갖도록 함으로써 고속화 할 수 있다. 본 논문에서는 이러한 단계중 특징추출에 관하여 논하였다. 특징벡터를 얻기 위해 40x40 화소로 정규화된 문자 영상을 분할하여 41개의 소영역(10×10 화소)으로 구성하였다. 소프트웨어에 의해 하나의 소영역의 특징을 추출하는데에는 500개 이상의 명령어가 수행되어야 하나, 제안된 알고리즘은 많은 논리 연산과 반복 loop을 병렬로 구성하여 소영역의 한 열의 특징을 하나의 싸이클에 추출할 수 있는 어레이 구조로 10 싸이클에 소영역의 특징추출이 완료된다. 따라서, 초당 12,000자 이상의 특징추출이 가능하다. 구현은 EPLD를 이용하여 1 chip화 하였으며, 본 연구실에서 개발된 고속 문서 인식기에 이용되어 그 실효성이 입증되었다.

Abstract

We proposed a high speed feature extraction algorithm and developed a feature vector extraction chip for high speed character recognition. It is hard to implement a high speed OCR by software alone with statistical method. Thus, the whole recognition process is divided into functional steps, then pipeline processed so that high speed processing is possible with temporal parallelism of the steps. In this paper, we discuss the feature extraction step of the functional steps. To extract feature vector, a character image is normalized to 40x40 pixels. Then, it is divided into 5x5 subregions and 4x4 subregions to construct 41 overlapped subregions(10×10 pixels). It requires to execute more than 500 commands to extract a feature vector of a subregion by software. The proposed algorithm, however, requires only 10 cycles since it can extract a feature vector of a column of subregion in one cycle with array structure. Thus, it is possible to process 12,000 characters per second with the proposed algorithm. The chip is implemented using EPLD and the effectiveness is proved by developing an OCR using it.

* 正會員, 高麗大學校 電子工學科
(Department of Electronic Engineering,)
* 正會員, 情報通信技術공동연구소
(Research Institute for Information and Communication, Korea Univ.)
接受日字 : 1993年 9月 15日

Ⅰ. 서 론

오프라인 문자 인식 방법중 패턴 정합법을 이용하여 문자를 인식하는 과정을 그림 1에 나타내었다. 패턴 정합법을 이용한 문자 인식의 경우 타 인식 방법에 비해 일반적으로 안정된 후보 문자들을 확보할 수는 있으나 한글이나 한자와 같이 인식 대상 문자의 수가 많을 경우 방대한 계산량으로 인해 처리속도가 느린 단점이 있다.

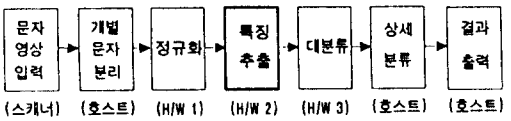


그림 1. 패턴 정합법을 이용한 문자 인식 과정
Fig. 1. Procedure of character recognition using pattern matching method.

본 논문에서는 이러한 단점을 개선하기 위하여 그림 1의 인식 단계를 기능적으로 분리하여 정규화, 특징추출, 대분류 과정을 각각 하드웨어화 하여 호스트 컴퓨터와 함께 4단(stage)의 파이프라인 처리로 시간적 병렬성을 통해 고속화하기 위한 일환으로 고속 특징추출 알고리즘의 제안과 구현에 관하여 논하였다. 그림 2에 파이프라인 처리에 의한 문자 인식 과정을 나타내었다.

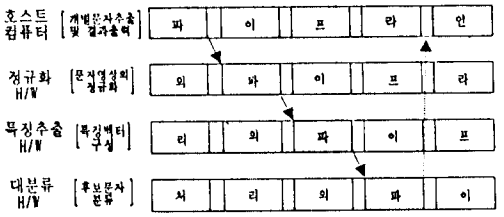


그림 2. 문자 인식의 파이프라인 처리 과정의 예
Fig. 2. The example of pipeline processing for OCR.

문자 인식에 있어서 특징추출 알고리즘은 인식률과 인식 속도에 많은 영향을 미치게 된다. 특징추출 방법에는 mesh 기법이나, Hough, Fouier 변환등 여러가지 방법들이 발표되었으나, 인식률을 비교 검토할 수 있는 객관적인 방법이 제시되지 않은 상태이다.[1] 본 연구에서는 문자 영상을 임의 크기의 mesh 형태로 나누어 각 소영역에서 얻어진 문자 윤곽선소의 방향정보를 이용하여 특징 벡터를 구성하였다. 표 1에는 사용된 소영역의 크기와 구성형태에 따라 특징 벡터를 만들어서 유클리드 거리 계산법에 의한 표준 데이타와의 유사도 계산 결과를 나타내었다. 가장 분류율이 높은 소영역의 형태를 선택하여 사용하였다.

표 1. 소영역 분할에 따른 누적 분류율(KS 2350)
Table 1. Accumulated classification rate according to the seperation of subregions(KS 2350).

단위 :(%)

소영역 분할 (정규화 크기)	후보 문자 순위에 따른 누적 분류율			
	1 위	3 위	5 위	10 위
6x6 (42x42화소)	86.74	94.02	97.07	98.90
7x7 (42x42 화소)	87.56	96.16	97.77	98.95
8x8 (40x40 화소)	89.57	97.80	98.58	99.43
5x5 + 4x4 (40x40 화소)	95.40	99.88	99.95	100.0
7x7 + 6x6 (42x42 화소)	93.32	98.67	99.40	99.52
9x9 + 8x8 (36x36 화소)	89.17	97.22	98.91	99.53

분류 실험은 KS 완성형 한글 2,350자를 한글 2.1의 명조체를 5종류의 크기로 인쇄하여 300dpi로 스캔하여 표 1에 제안된 방법으로 특징을 추출하고 그 값들을 평균하여 표준 벡터를 구성하였다. 실험 문자는 한글 2.1의 명조체(12 point) 2,350자를 재 인쇄하여 이용하였다. 표 2에서와 같이 입력 문자 영상을 40×40 화소로 정규화하여 1차적으로 8x8화소 크기로 5×5개로 분할한 후, 내부를 중첩되게 4x4개로 분할하였을때 가장 높은 분류율을 얻을 수 있었다. 따라서, 하나의 소영역은 주변 화소를 포함하여 10×10 화소로 구성되며, 하나의 문자 영상은 총 41개의 소영역으로 분할 된다. 그림 3에 그 구성 형태를 나타내었다.

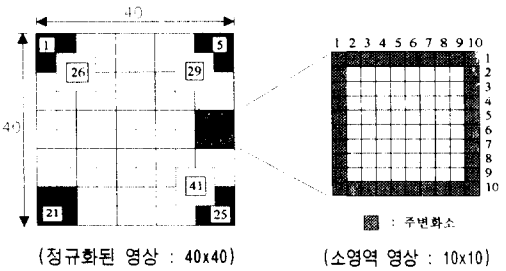


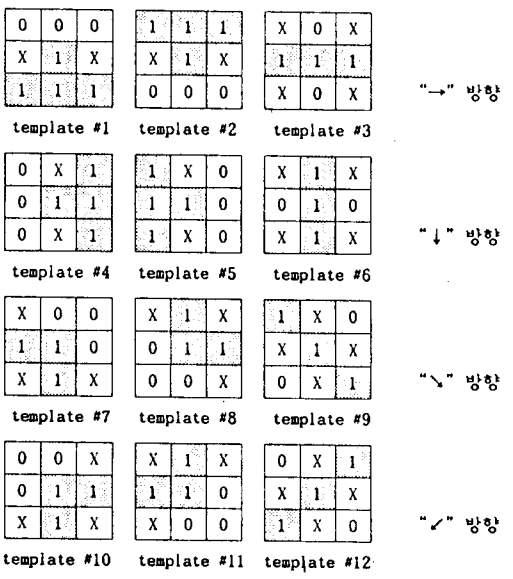
그림 3. 문자 영상의 소영역 분할
Fig. 3. The seperation of subregion for a character image.

일반적으로 어레이 프로세서는 PE(Processing Element)라 불리어지는 연산기(ALU:Arithmetic Logic Unit)가 다수개로 구성되어 있으며, 이러한 PEs는 제어부(control unit)에 의해 동기되어 같은 기능을 동시에 수행할 수 있는 공간적 병렬성(spatial parallelism)을 갖는 구조로 되어있다.[2,3,4] 개발된 특징 추출 프로세서는 방향정보를 추출하는 template 세트와 추출된 방향 정보를 합산하고 누산하는 연산기 및 제어 로직등으로 구성되어 있다.

이러한 구성을 4방향 성분에 대해 병렬로 수행시킬 수 있도록 설계함으로써 한 싸이클에 하나의 열에 대한 방향 정보를 추출하여 연산할 수 있는 다중의 병렬성을 갖는 어레이 프로세서를 구현하였다. 구현은 ALTERA사의 MAX-PLUS tool을 이용하여 설계 및 시뮬레이션을 통해 검증한 후, EPLD(Erasable Programmable Logic Device)구조를 갖는 USIC (User Specific IC)을 이용하여 구현 하였다.[5,6]

Ⅱ. 고속 특징추출 알고리즘

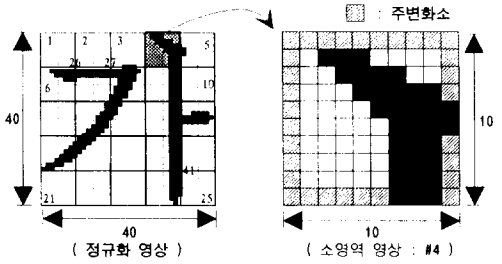
본 논문에 제안된 특징추출 알고리즘은 문자 영상의 윤곽선소의 방향 정보를 추출하여 특징벡터를 구성하였다. 이 정보는 문자영상이 가지고 있는 방향정보와 함께 밀도정보, 위치정보등을 포함하고 있다. 이러한 문자 영상의 가장자리는 스캐닝 상태나 외부 잡음등으로 인하여 균일하지 못하기 때문에 보다 안정적인 정보를 얻기 위하여 수평 및 수직 성분에 대해서는 윤곽선의 한 꺼풀 안쪽의 정보를 이용하여 특징을 추출할 수 있도록 template 세트를 구성하였다. 4방향 정보를 추출하기 위해 제안된 특징추출 template들을 그림 4에 나타내었다.



(0 : 백화소, 1 : 흑화소, X : Don't care)

그림 4. 제안된 특징 추출 templates(4 방향 정보)
Fig. 4. Proposed feature extraction templates(4-direction).

따라서, 정규화 영상을 41개의 소영역으로 분할하여 각 영역에 대한 4 방향의 특징추출 값을 이용하여 164차원의 특징 벡터를 구성하였다. 그림 5에 특징 벡터의 구성 예를 나타내었다. 소영역의 주변 화소는 가장자리의 방향 정보를 얻기 위하여 이웃하는 소영역의 영상 정보를 포함시킨 것이다.



구분 / 소영역 번호	방향별 특징 벡터의 값			
	수 평 (-)	사 선 (/)	수 직 (|)	역사선 (\)
1	0	0	0	0
2	0	0	0	0
3	2	1	0	1
4	3	1	1	6
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
41	0	0	16	0

그림 5. 특징 벡터의 구성 예
Fig. 5. The example of feature vector.

그림 5와 같은 한 문자의 특징 벡터를 소프트웨어에 의하여 구성하는 데에는 그림 6과 같이 많은 논리 연산과 반복 loop가 요구된다.

이러한 반복적인 논리 연산과 반복 loop는 병렬 처리를 통하여 고속화가 가능하다. 그림 7에 4개의 방향 성분을 추출하는데 사용된 12개의 templates를 논리 게이트로 구성한 예를 나타내었다. 따라서, 하나의 3x3윈

도우에 대해 한 클럭 싸이클에 12개의 특징추출 template를 이용하여 모든 방향정보를 추출할 수 있다.

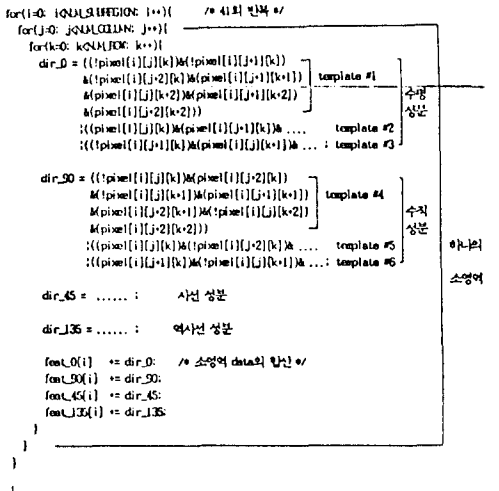


그림 6. 소프트웨어에 의한 특징추출 알고리즘
Fig. 6. The algorithm of feature extraction by software.

그림 7의 논리 게이트를 이용하여 하나의 3x3윈도우에 대해서는 한 싸이클에 특징추출이 가능하나, 소영역(10x10)에 대한 정보를 모두 얻기 위해서는 그림 6의 loop을 반복해서 수행하여야 한다. 이러한 반복 loop을 병렬처리를 통하여 한 클럭 싸이클에 한 열의 정보(3x10 화소)가 추출될 수 있도록 그림 7의 특징추출용 templates를 소영역의 행의 수만큼 병렬로 구성함으로써 2차 병렬성을 갖도록 하였다.

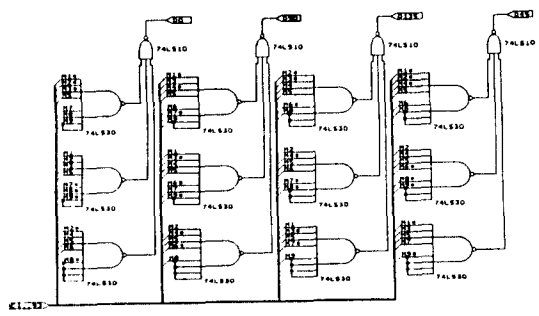


그림 7. 논리 게이트를 이용한 특징추출 templates
Fig. 7. The feature extraction templates using logic gates.

이를 통해 각각의 templates에서 추출된 방향정보는 식(1)에서와 같이 같은 방향 성분끼리 합산된 후, 하나의 소영역에 대해 누산될 수 있도록 구성함으로써 3차의 병렬성을 갖도록 설계하였다.

$$\begin{aligned} S_{1-} &= D_{11-} + D_{21-} + \cdots + D_{m1-} \\ S_{1|} &= D_{11|} + D_{21|} + \cdots + D_{m1|} \\ S_{1/} &= D_{11/} + D_{21/} + \cdots + D_{m1/} \\ S_{1\backslash} &= D_{11\backslash} + D_{21\backslash} + \cdots + D_{m1\backslash} \end{aligned} \qquad 1(a)$$

여기서, S1은 1열에서의 방향 성분의 합이고 D는 각 윈도우의 방향성분 이다. m : 소 영역 행의 갯수 (본 연구에서는 m=8)

식1(a)에 의하여 소영역의 한 열에 해당되는 방향 성분의 합은 한 클럭 싸이클에 의하여 구해지게 된다. 이와 같은 방법으로 소영역에 대한 방향 성분의 합을 누산하는 과정을 수평 성분(' - ')에 대하여 살펴 보면,

$$\begin{aligned} S_{-} &= S_{1-} + S_{2-} \cdots S_{n-} \\ &= \sum_{j=1}^{n} S_{j-} \\ &= \sum_{j=1}^{n} \sum_{i=1}^{m} D_{ij-} \\ &= \sum_{j=1}^{n} (\sum_{i=1}^{m} D_{ij-}) \end{aligned} \qquad 1(b)$$

n: 소 영역 열의 갯수(본 연구에서는 n=8)

이 된다. 같은 방법으로 나머지 3 방향에 대하여도 표기할 수 있다.

식 $(\sum_{i=1}^{m} D_{ij-})$ 는 한 클럭에 의해 값이 구해지기 때문에 소영역의 각 방향 성분 모두를 추출 하는데에는 n+2 (주변화소 포함)개의 클럭 싸이클이 소요된다.

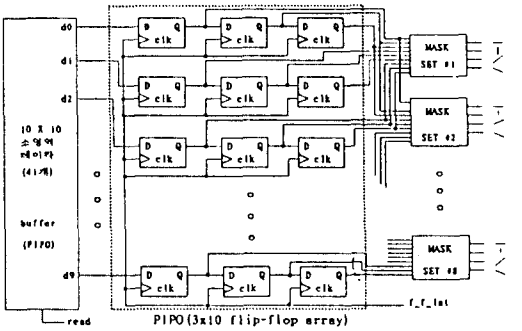


그림 8. PIPO 및 templates(8조)의 구성
Fig. 8. The structure of PIPO and templates(8 sets).

따라서, 하나의 소영역의 특징을 추출하는데에는 10클럭 싸이클이 소요되고, 한 문자의 특징벡터를 구성하는데에는 410개의 클럭 싸이클이 소요된다. 특징추출을 위한 3x10화소를 저장하기 위한 PIPO (Parallel In Parallel Out)와 특징추출 templates 8조를 병렬로 위치시킨 구성도를 그림 8에 나타내었다.

앞에서 기술된 특징추출 templates 세트를 이용하여 추출된 방향 성분을 합산하고 누산하는 기능과, 제어부의 설계는 다음 절에서 논한다.

Ⅲ. 특징 추출용 칩의 구현

앞 절에서 기술된 고속 특징추출 알고리즘의 전체적인 구성도를 그림 9에 나타내었다.

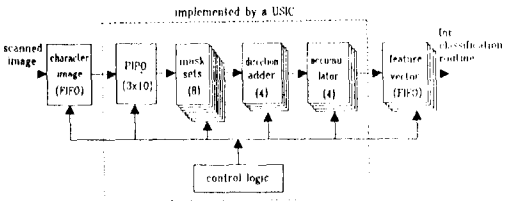


그림 9. 특징추출용 칩의 구성도
Fig. 9. The block diagram of feature extraction chip.

전체 구성도 중에서 PIPO와 templates의 구성은 앞절에서 언급되었고, 하나의 방향성분에 대한 합산기와 누산기를 살펴보면 그림 10과 같다. 합산기(adder)는 특징추출 templates 세트를 통해 각 열에서 추출된 방향정보들을 같은 방향 성분끼리 합하는 역할을 하고, 누산기는 합산기에서 더해진 각 열의 방향 성분들을 하나의 소영역에 대해 누적하는 기능을 한다. 이러한 누산기는 합산기와 래치(latch)로 구성된다.

제어부(control unit)는 특징추출 회로의 각 부분을 제어하는 역할을 수행하는데, 외부로 부터 특징추출 시작(start) 신호를 받으면 특징추출을 위해 41개의 소영역 데이터를 저장하고 있는 버퍼에서 데이터를 읽어 특징추출 하드웨어의 PIPO에 저장한다. 저장된 데이터에 특징추출 templates을 매칭시켜 얻어진 정보를 합산기에 의해 합산하면, 그 결과를 누산기에서 누산하여 래치에 저장시키도록 제어한다.

각 소영역의 특징추출이 완료되면 출력 버퍼(FIFO)에 저장시키고, 한 문자에 대한 처리가 완료되면 인터럽트(interrupt)신호를 발생시켜 특징추출이 완료되었음을 알린다. 특징추출 데이타를 인출해 가는것을 인터럽트 응답(interrupt acknowledge) 신호로 받아들이고, 다음 문자의 특징을 추출하기 위해 시작 신호를 확인하게 되므로써 같은 과정을 반복하게 된다.

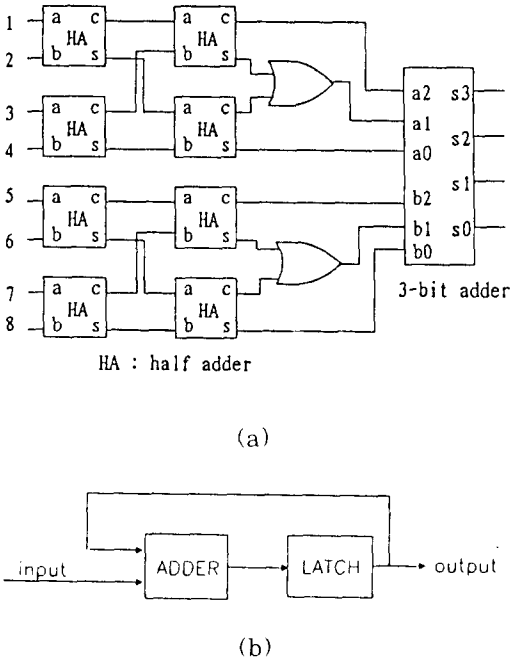


그림 10. 합산기와 누산기
(a) 합산기 (b) 누산기
Fig. 10. The adder and accumulator
(a) adder (b) accumulator.

이상에서 기술된 각각의 기능중 입력 및 출력의 버퍼(FIFO)를 제외한 나머지 부분은 Altera사의 MAX-PLUS tool을 이용하여 시뮬레이션을 통해 동작 상태를 검증한 후, 192개의 Macro-cell을 가진 USIC (EPM5192JC)을 이용하여 구현하였다. 그림 11에 하드웨어의 외부 사양을 나타내었다.

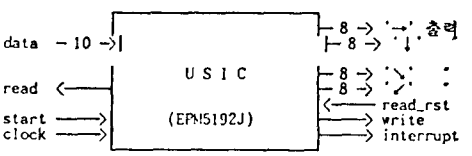


그림 11. 구현된 하드웨어의 외부 구조
Fig. 11. Pin configuration of implemented H/W.

구현된 칩을 이용하여 특징을 추출하는데 소요되는 시간과 IBM PC (386-33MHz)에서 C-언어로 기술된 소프트웨어를 이용하여 처리한 시간을 비교하여 표 2에 나타내었다.

표 2. 특징추출 소요 시간의 비교
Table 2. Comparison of feature extraction time.

실 험 시 스 템	소요 시간
IBM PC (386-33MHz)	17 ms/자
구현된 하드웨어	82 μs/자

실험 결과로 부터 구현된 특징추출 칩의 성능은 초당 12,000자 이상의 특징을 추출할 수 있으며, 또한 호스트 컴퓨터와 파이프라인 처리를 통한 시간적 병렬성으로 고속 문자 인식기의 구현을 위하여 제안된 고속 특징추출 알고리즘이 이용 가능하다. 실제로 본 연구실에서 구현된 문자 인식기(한글 2,350자를 대상으로 초당 140자를 인식 가능)에 이용되어 그 실효성을 을 입증하였다.

Ⅳ. 결 론

패턴 정합법을 이용한 문자 인식 방법은 인식 대상 문자가 많을 경우 속도가 느린 단점이 있다. 이러한 단점을 보완하기 위하여 인식 단계를 기능적으로 분리하여 하드웨어화 함으로써 각 단계별 시간적 병렬성으로 고속화 할 수 있다. 특징추출 단계는 많은 논리 연산과 반복 수행의 특성을 가지고 있는데, 이 특성을 이용하여 고속 처리가 가능한 알고리즘을 제안하고 구현함으로써 고속 문서 인식기에 이용 가능하도록 하였다. 제안된 하드웨어는 다수의 PE (Processing Element)와 제어 로직으로 구성되어 있으며, 클럭에 동기되어 한 싸이클에 다수 개의 연산이 이루어지는 어레이 프로세서이다.

본 연구에서 사용한 특징추출 방법은 40x40화소로 정규화된 문자 영상을 41개로 분할한 소영역으로부터 윤곽선소의 방향성분을 추출하여 총 164개의 특징 벡터를 구성하는 것이다. 제안된 알고리즘에 의해 구현된 하드웨어와 IBM PC에서 C-언어로 기술된 알고리즘에의한 처리시간을 비교해 본 결과, 약 200배의 처리 속도의 향상을 가져왔으며, 초당 12,000자 이상의 특징을 추출할 수 있다.

칩은 EPLD구조를 가진 USIC(EPM5192JC)을 이용하여 하나의 칩으로 구현하였다. 구현된 칩을 본 연구실에서 구현한 고속 문자 인식기에 사용하여 그 실효성을 입증하였다.

參 考 文 獻

[1] Moshe Kushnir, "Recognition of handprinted Hebrew characters using Features selected in the Hough Transform space", *Pattern Recognition* vol.18, No.2, pp.103-114, 1985.

[2] Kai Hwang, F.A.Briggs, *Computer Architecture and Parallel Processing*, pp.20-22, McGRAW-HILL, 1989.

[3] J.F.Cavaagh, Digital *Computer Arithmetic, Design and Implemen- tation*, pp.107-117, McGRAW-HILL, 1984.

[4] S.Y.Kung, S.C.Lo, S.N.Jean, J.N. Hwang, "Wavefront Array Processors - Concepts to Implementation", *IEEE COMPUTER* JUL. 1987, pp18-33.

[5] Data Book, ALTERA, 1992.

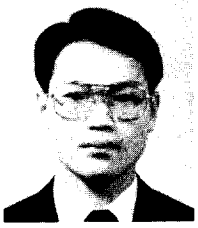
[6] *Application Handbook*, ALTERA, 1992

[7] D.L.Stover and R.D.Iverson, "A One-Pass Thinning Algorithm and it's Parallel Implementation", *Computer Vision*, Graphics and Image Processing Vol.40, pp.30-40, 1987.

[8] J.T.Kuehn, J.A.Fessler, H.J.Siegel, "Parallel Image Thinning and Vectorization on PASM", *IEEE*, pp.368-374, 1985.

(※ 본 연구는 삼성전자 위탁 과제로 수행 되었음)

著 者 紹 介



金 赫 九(正會員)

1961年 3月 19日生. 1984年 2月 금오공대 전자공학과 졸업(공학사). 1990年 8月 고려대학교. 산업대학원 전자통신공학과 졸업(공학석사). 1994年 2月 고려대학교 대학원 전자공학과 졸업(공학박사). 현재 고려대학교부설 정보.통신기술공동연구소 연구원. 주관심 분야는 문서인식, 마이크로프로세서 응용 및 영상처리 등임.

•

金 惠 鎭(正會員) 第 29卷 A編 第 8號 參照
현재 고려대학교 전자공학과 교수

姜 仙 美(正會員)

1959年 7月 28日生. 1981年 2月 고려대학교 전자공학과 졸업(공학사). 1988年 6月 에얼랑겐-뉘렌베르그 대학교 전자공학과 졸업(Diplom). 1992年 8月 고려대학교 대학원 전자공학과 졸업(공학박사). 1992年 11月 ~ 1994年 2月 고려대학교부설 정보.통신기술공동 연구소 연구조교수. 현재 고려대학교 산업대학원 객원조교수. 주관심 분야는 패턴인식, 영상처리 분야 등임.