論文 89-26-2-10

剩餘數體系를 이용한 자승오차 패턴 클러스터링 프로세서의 실현

(Implementation of the Squared-Error Pattern Clustering Processor Using the Residue Number System)

金 炯 民*, 趙 源 敬*

(Hyeong Min Kim and Won Kyung Cho)

要 約

패턴인식과 영상처리 응용에 이용되는 자승오차 패턴 클러스터링 알고리듬은 특징벡터 행렬의 연산에 상당한 처리시간을 요구한다. 그러므로 본 논문은 병렬처리와 파이프라인 특성을 갖는 잉여수체계를 이용한 고속의 자승오차 패턴 클러스터링 프로세서를 제안한다.

제안된 자승오차 패턴 클러스터링 프로세서는 영상분할 실험으로부터 의미있는 영역으로 나눌수 있는 클러스터의 수에 대하여 만족할 만한 오차를 보이며 80287 수치 연산용 프로세서보다 약 200배 빠름을 보인다. 그 결과 대규모의 데이타를 실시간으로 처리하여야 하는 응용분야에 효과적으로 이용할 수 있음을 확인하였다.

Abstract

Squared-error Pattern Clustering algorithm used in unsupervised pattern recognition and image processing application demands substantial processing time for operation of feature vector matrix. So, this paper propose the fast squared-error Pattern Clustering Processor using the Residue Number System which have been the nature of parallel processing and pipeline. The proposed Squared-error Pattern Clustering Processor illustrate satisfiable error rate for Cluster number which can be divide meaningful region and about 200 times faster than 80287 coprocessor from experiments result of image segmentation. In this result, it is useful to real-time processing application for large data.

I. 서 론

자승오차 패턴 클러스터링 알고리듬은 패턴인식과 영상처리(PRIP) 응용에 많이 이용되기 때문에 고속 연산을 위하여 전용의 하드웨어로 구성하려는 연구가 진행되고 있다.[2,5,6] 예로서 1983년 Hsi-Ho Liu와 K. S. Fu에 의하여 유클리디언 거리를 계산하는 전용 프로세서를 시스토릭 어레이로 구성하였으며 1985년 L. M. Ni와 A. K. Jain은 파이프라인 구조의 기본처리 장치를 이용하여 클러스터링 프로세서를 2차원 시스토릭 구조로 설계하여 좋은 성과를 얻었다.

자승오차 패턴 클러스터링 알고리듬은 특징벡터행렬 계산에 반복적인 연산을 필요로 하기 때문에 상당한 처리시간을 요구한다. 그러므로 본 논문에서는

*正會員, 慶熙大學校 電子工學科
(Dept. of Elec. Eng., Kyunghee Univ.)
接受日字 : 1988年 9月 28日

실시간 처리가 가능하도록 하기 위하여 잉여수체계를 이용한 고속의 클러스터링 프로세서를 설계한다.

기존의 연산회로는 일반적으로 2진수 체계를 이용하는데 2진수체계에서는 고속의 병렬승산기를 구성하기가 대단히 어렵다. 그러나 최근 연구가 활발히 진행되어 온 잉여수체계를 이용하면 고속의 병렬승산기를 쉽게 구성할 수 있다. 잉여수체계는 非重價 수체계로서 캐리정보가 필요없기 때문에 고속연산이 가능하다. 또한 연산모듈이 서로 독립적이므로 병렬형 연산회로의 구성에 적합하며 용이하게 VLSI로 제작할 수 있다.[1] 그러므로 본 논문에서는 잉여수체계를 이용하여 클러스터링 프로세서를 설계하고, 설계된 프로세서를 영상분할 실험에 적용하여 그 결과를 평가한다.

Ⅱ. 자승오차 패턴 클러스터링의 기본개념

1. 라벨할당과정

자승오차 패턴 클러스터링 방법은 입력패턴과 표준패턴간의 유클리디언 거리를 이용한 최소거리 판단을 기본으로 하는 알고리듬으로서 라벨할당과정과 클러스터 중심의 재설정(cluster center updating) 과정으로 이루어진다. 자승오차 패턴 클러스터링 알고리듬의 흐름도는 그림 1과 같다.

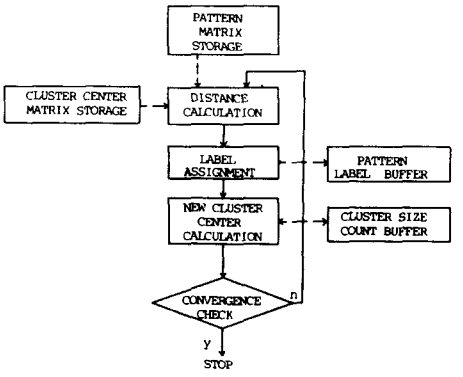


그림 1. 자승오차 클러스터링의 흐름도
Fig. 1. Flow chart of squared-error clustering.

그림 1에서 라벨할당 과정은 아래와 같이 두 단계로 구성한다.

1) 입력패턴과 표준패턴 간의 euclidean distance를 계산하는 과정과
2) 입력패턴을 가장 가까운 표준패턴에 할당하는 과정

입력패턴과 표준패턴간의 거리계산은 (1)식과 같이 pseudo-matrix-multiplication으로 공식화 할 수 있다.

$$[ED]=[C]\circ[P] \tag{1}$$

ED : euclidean distance
P : 입력패턴
C : 표준패턴

$$\begin{bmatrix} ED_0^0 & ED_0^1 \cdots ED_0^{N-1} \\ ED_1^0 & ED_1^1 \cdots ED_1^{N-1} \\ \vdots & \vdots \quad\quad \vdots \\ ED_{K-1}^0 & ED_{K-1}^1 \cdots ED_{K-1}^{N-1} \end{bmatrix} = \begin{bmatrix} C_0^0 & C_1^0 & \cdots C_{D-1}^0 \\ C_0^1 & C_1^1 & \cdots C_{D-1}^1 \\ \vdots & \vdots & \quad\vdots \\ C_0^{K-1} & C_1^{K-1} & \cdots C_{D-1}^{K-1} \end{bmatrix} \circ \begin{bmatrix} P_0^0 & P_0^1 \cdots P_0^{N-1} \\ P_1^0 & P_1^1 \cdots P_1^{N-1} \\ \vdots & \vdots \quad\quad \vdots \\ P_{D-1}^0 & P_{D-1}^1 \cdots P_{D-1}^{N-1} \end{bmatrix}$$

$$= \begin{bmatrix} C^0 \\ C^1 \\ \vdots \\ C^{K-1} \end{bmatrix} \circ [P^0\ P^1 \cdots P^{N-1}] \tag{2}$$

윗식에서 n번째 입력패턴 P(n)과 k번째 표준패턴 C(k)간의 유클리디언 거리는 $ED_k^n = d^2(P^n, C^k)$가 된다. 여기서 N은 입력패턴의 수, k는 표준패턴의 수이며, D는 특징차수이다.

입력패턴을 표준패턴에 할당하는 것은 (3)식에 의한다.

$$L(n)=k,\ d^2(P(n), C(k)) = \min\{ED_0^n, ED_1^n, \cdots, ED_{k-1}^n\} \tag{3}$$

여기서 L(n)은 n번째 입력패턴의 라벨 윗식에서 입력패턴과 k차 표준패턴의 유클리디언 거리가 최소가 된다. 그 결과 n번째 입력패턴의 라벨은 k로 할당된다.

2. 클러스터 중심 재설정과정

라벨할당 과정에서 새로 구성된 패턴 클러스터에 변화가 있을 경우 각 클러스터의 중심좌표값은 새로이 계산되어야 한다. 구하는 식은 다음과 같다.

$$C(k, d) = \frac{1}{|S(k)|} \sum_{n \in S(k)} P(n, d) \tag{4}$$

$|S(k)|$는 k번째 클러스터 S(k)의 패턴수 자승오차 패턴 클러스터링 알고리듬에 의한 패턴 클러스터링은 라벨할당 과정과 클러스터 중심 재설정 과정을 새로 할당된 클러스터의 변화가 한계값 이하가 될때까지 또는 제한횟수까지 반복함으로서 실행된다.

Ⅲ. 클러스터링 프로세서의 설계

1. 클러스터링 프로세서의 기본구성

잉여수체계를 이용한 클러스터링 프로세서의 기본

구성도는 그림 2와 같다. 그림 2의 클러스터링 프로세서는 유클리디언 거리값을 구하는데 이용되는 DSA (difference-squared-accumulator), 최소 거리값을 갖는 클러스터를 선택하는데 필요한 비교기(RMC)와 기타 필요한 정보의 기억을 위한 버퍼 및 카운터로 구성되며 파이프라인 구조를 갖는다. 그리고 표준패턴 기억장치는 표준패턴 데이타의 원활한 공급을 위한 것이다.

FIFO(first in first out)형의 라벨버퍼는 N개 입력패턴들의 라벨을 기억하는데 이용되며 새로운 라벨과 이전의 패스에서 이미 할당되었던 라벨과의 비교에서 변화가 발생되면 라벨의 변화를 기록하는 카운터가 1씩 증가한다.

클러스터 카운터는 K개의 카운터로 구성되며 해당 클러스터에 속한 패턴의 수를 기억한다.

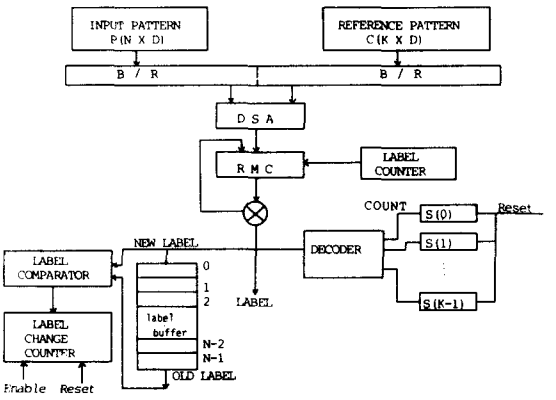


그림 2. 패턴 클러스터링 프로세서의 구성도
Fig. 2. Block diagram of pattern clustering processor.

2. 표준패턴 기억장치의 구성

고속의 연산에 필요한 데이타의 원활한 공급을 위한 기억장치는 그림 3과 같이 설계하였다. 표준패턴 기억장치는 j개의 세그먼트로 나뉘어지고 각 세그먼트는 D차의 특징데이타를 저장하고 한 세그먼트는 2의 冪數로 구성되며 $2^n \geq D$를 만족한다. 이와 같은 기억장치로부터 데이타를 엑세스하기 위하여 2개의 어드레스 카운터가 필요하다. 세그먼트 카운터는 각 세그먼트를 어드레싱(k=ϕ,…, k−1)하며 특징차수 카운터는 각 차수의 특징데이타를 어드레싱(d=ϕ,…, D−1)한다. 특징차수 카운터의 값이 D−1에 이르면 특징차수 카운터는 리세트되고 세그먼트 카운터는 1씩 증가하게 되어 다음 세그먼트의 표준패턴 데이타를 출력시킨다.

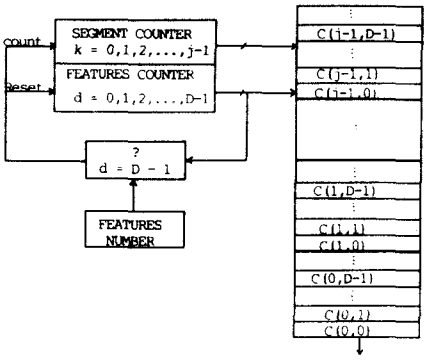


그림 3. 표준패턴 기억장치
Fig. 3. Reference pattern storage.

3. DSA(difference-squared-accumulator)의 구성

DSA는 입력패턴과 표준패턴 사이의 유클리디언 거리값을 구하는데 이용된다. 거리값은 (5)식과 같이 구할 수 있다.

$$ED_k^n = d^2(P(n),\ C(k))$$
$$= |P(n,\ 0)-C(k,\ 0)|^2+|P(n,\ 1)-C(k,\ 1)|^2+\cdots+|P(n,\ D-1)-C(k,\ D-1)|^2$$
$$= \sum_{d=0}^{D-1} |P(n,\ d)-C(k,\ d)| \qquad (5)$$

잉여수체계에서의 거리값 계산은 법 m_i일 경우

$$|ED_k^n|_{m_i} = \left|\sum_{d=0}^{D-1} |(|P(n,d)|_{n_i}-|C(k,d)|_{m_i})^2|_{m_i}\right|_{m_i} \qquad (6)$$

과 같이 계산된다.

DSA 연산을 위한 기본 연산모듈을, 잉여수체계를 이용하여 구성한 예는 그림 4와 같다. 그림 4a의 점선내의 뺄셈과 제곱연산은 2진수체계를 이용할 경우 순차적으로 두단계의 연산을 거쳐야 하지만 잉여수체계에서는 하나의 연산모듈로 가능하므로 하드웨어의 크기와 연산 시간을 감소할 수 있다. 그림 4b에서와 같이 $(a_j-b_j)^2$ 형태의 연산이 하나의 연산모듈에 의하여 가능하기 때문에 한 클럭내에 가능하다. 따라서 그림 4의 연산모듈을 사용하는 경우 K 개의 표준 패턴과 D차의 특징차수를 갖는 N개의 입력패턴을 처리하는데 N·K·D 클럭이 소요된다.

4. 혼합기수변환을 이용한 비교기

잉여수체계는 非重價표현이므로 수의 부호, 오버훌로우(overflow) 검출과 크기의 비교를 위하여는 비중가표현을 중가표현으로 변환하여야 한다. 이에 적합한 변환 알고리듬이 혼합기수변환 알고리듬이며

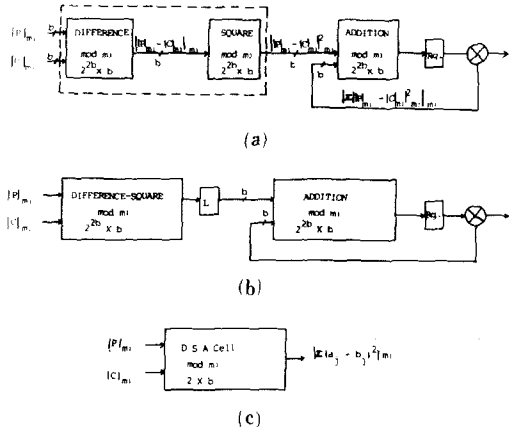


그림 4. Mod m_i인 경우 DSA의 기본 연산모듈
Fig. 4. Computation module of DSA for mod m_i.

병렬성을 높인 병렬혼합기수변환 알고리듬이 제안되고 있다.[3]

본 논문에서 제안한 클러스터링 프로세서는 최종 결과로 라벨만을 출력하므로 잉여수를 2진수체계로 변환할 필요가 없이 잉여수의 부호검출만을 필요로 한다. 따라서, 여기서는 혼합기수변환 알고리듬을 이용한 부호검출기를 이용하였다.

범위[0, M−1]을 갖는 정수 z는 다음과 같이 혼합기수형태로 나타낼 수 있다.

$$z=\sum_{i=1}^{n} \overline{z}_i \cdot q_i \qquad (7)$$

여기서 $q_i=\prod_{i=1}^{i-1} m_i$, $q_1=1$

$\overline{z}_i$는 정수 z의 혼합기수변환 계수이다.

z, z_1, z_2,⋯, z_n의 잉여수 표현은 다음과 같이 정의한다.

$$\begin{aligned} z&=(z_1, z_2,\cdots, z_n)\\ z_1&=(z_1, 0,\cdots, 0)\\ z_2&=(0, z_2,\cdots, 0)\\ &\vdots\\ z_n&=(0, 0,\cdots, z_n) \end{aligned} \qquad (8)$$

여기서 $z<M=\prod_{j=1}^{n} m_j$, $z_i<M$, i=1, 2,⋯, n이다.

z_i의 혼합기수표현은 다음과 같고

$$z_i=[0, 0,\cdots, 0, \overline{z}_{ii}, \overline{z}_{i,i+1},\cdots, \overline{z}_{i,n}] \qquad (9)$$

$\overline{z}_{ii}$는 법 m_i에 대한 z_i의 혼합기수변환 계수이다.

z값은 병렬혼합기수변환 계수로부터 얻을 수 있다.

$$\begin{aligned} z&=|\sum_{i=1}^{n} z_i|_M\\ &=|\overline{z}'_1q_1+\overline{z}'_2q_2+\cdots+\overline{z}'_nq_n+C_nM|_M\\ &=\overline{z}'_1q_1+\overline{z}'_2q_2+\cdots+\overline{z}'_nq_n \end{aligned} \qquad (10)$$

여기서 $\overline{z}'_i$는 i번째의 병렬혼합기수변환 계수이며 z_1, z_2,⋯, z_n의 법 m_i에 대한 혼합기수변환 계수의 합으로 구해진다.

앞의 과정에 의해 얻어진 혼합기수변환 계수로부터 z의 부호를 검출하는 방법은 다음의 과정에서 얻어진다.

정수 z는 다음과 같이 쓸 수 있다.

$$z=|z|_M=K\cdot P+|z|_P \qquad (11)$$

여기서 $K\geq 0$, $0<P\leq M$

함수 S(z)를 다음과 같이 정의한다.

$$\begin{aligned} &S(z)=0 \quad \text{if} \quad K=0\\ &S(z)=1 \quad \text{if} \quad K>0 \end{aligned} \qquad (12)$$

정수 P=M/2라고 가정하면 S(z)는 0 또는 1값을 갖는다.

서로소인 법들중 하나가 짝수일 경우

$$P=M/2=m_1m_2\cdots m_{n-1}m_n/2 \qquad (13)$$

여기서 $m_P=m_n/2$로 놓는다.

하드웨어 구성을 위하여 짝수인 법을 이용하면

$$m_n=2^W\cdot S \qquad (14)$$

여기서 $W>0$, $S>0$, $|S|_2=1$

W=1인 경우 m_P는 조건 $|m_P|_2=1$을 만족하며 부호검출은 간단한 형태를 취한다. W=1, S>0, $|S|_2=1$에 대하여 다음 식을 만족하면 S(z)=0이다.

$$||z|_{m_n}|_2=||z|_P|_2 \qquad (15)$$

좌변은 $|z|_M$으로부터 얻을 수 있다.

$$||z|_M|_2=||z|_{m_n}|_2=|z_n|_2=\beta_0 \qquad (16)$$

β_0는 잉여수 z_n의 LSB와 같다.

우변은 $||z|_P|_2=|\overline{z}'_0+\overline{z}'_1m_1+\cdots+\overline{z}'_{n-2}m_1m_2\cdots m_{n-2}+\overline{z}'_Pm_1m_2\cdots m_{n-1}|_2$ (17)

모든 m_i, i≠n에 대하여

$$|\prod_{i=1}^{n-1} m_i|_2=1$$

이므로

$$||z|_P|_2=|\sum_{i=0}^{P} |\overline{z}'_i|_2|_2 \qquad (18)$$

(16)과 (18)식을 (15)식에 대입하여 다음을 얻는다.

$$\beta_0=|\sum_{i=0}^{P} |\overline{z}'_i|_2|_2 \qquad (19)$$

여기서 $|\overline{z_i}|_2$는 계수 $\overline{z_i}$의 LSB와 같다.

(19)로부터 z의 부호는

$$S(z)=|\sum_{i=0}^{P}|\overline{z_i}|_2+\beta_0|_2 \qquad (20)$$

과 같이 구할 수 있다.

프로세서에서 대소비교의 대상은 거리 값이므로 항상 양의 값을 가지며 두 수를 a, b라면

$0\leq|a-b|_M<M/2$일 경우 $a\geq b$

$M/2\leq|a-b|_M<M$일 경우 $a<b$이다.

위의 과정에 의하여 구성한 비교기는 그림 5와 같다.

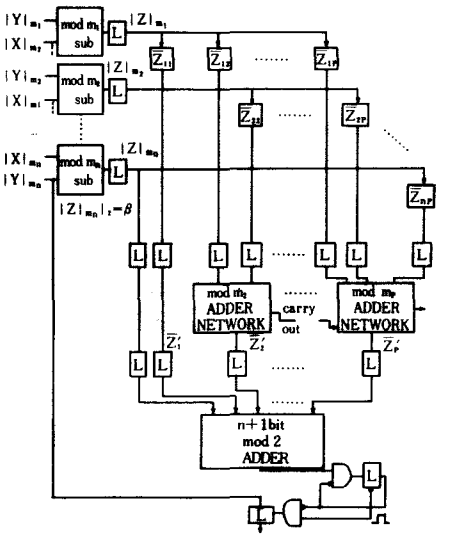


그림 5. 혼합기수변환을 이용한 비교기
Fig. 5. Comparator using mixed radix conversion.

비교기는 출력단에 래치를 이용하여 한계값으로 초기화시키고 D클럭마다 현재까지의 최소 거리값을 출력한다. 따라서 오버홀로우가 발생하지 않는다.

클러스터링 프로세서의 입력패턴 처리시간은 다음과 같다.

초기지연시간 : $t_c+t_{DSA}+t_{RMC}$

한 입력패턴 처리시간 $T(1, K, D)=$

$t_c+t_{DSA}+t_{RMC}+KD$

N개 입력패턴 처리시간 $T(N, K, D)=$

$$t_c+t_{DSA}+t_{RMC}+(N-1)td+NKD \qquad (21)$$

여기서 단위는 클럭사이클이며 클럭사이클은 기본 연산모듈의 연산시간에 의하여 결정된다.

td : $t_{RMC}-1$

t_c : 2진수/잉여수 변환기의 소요 클럭사이클 수

t_{DSA} : DSA의 소요 클럭사이클 수

t_{RMC} : 대소비교기의 소요 클럭사이클 수

Ⅳ. 실험 및 고찰

본 논문에서 잉여수체계를 이용하여 구성한 클러스터링 프로세서의 이론적인 타당성과 응용실험을 위하여 디지탈 컴퓨터를 이용하여 시뮬레이션을 실행하였다. 실험방법은 IBM PC/AT로 XENIX system V에서 C언어를 이용하여 B/R변환기, DSA와 비교기를 각각의 부함수로 작성하여 각 모듈의 동작을 정의하고 클러스터링 프로세서의 시뮬레이션을 진행하였다. 또한, 잉여수체계를 이용한 경우와 기존의 실수처리에 의한 결과를 비교하기 위하여 2진수체계의 부동소숫점 연산을 이용한 패턴 클러스터링 프로그램을 작성하였다.

클러스터링 프로세서의 응용실험을 위하여 영상분할에 필요한 특징을 sobel 연산자를 이용하여 추출하였다.

Sobel 연산자는

$$[S1]=\begin{bmatrix}1&0&-1\\2&0&-2\\1&0&-1\end{bmatrix}\quad[S2]=\begin{bmatrix}1&2&1\\0&0&0\\-1&-2&-1\end{bmatrix} \qquad (22)$$

영상분할을 위한 특징은

X1=monochrome brightness
X2=sobel log magnitude
X3=sobel phase
X4=(3×3) mode of brightness
X5=(3×3) mode of sobel log magnitude
X6=(3×3) mode of sobel phase

그리고 초기 클러스터의 중심값으로 영상의 입력화소중 임의의 화소를 취하였다.

클러스터링 프로세서의 각 패스에 소요되는 처리시간은 입력패턴의 수가 많을 경우 초기 지연시간을 무시할 수 있으므로 (21)식으로부터 다음과 같이 표시할 수 있다.

$$T(N, K, D)\approx(N-1)td+NKD \qquad (23)$$

위 식에서 단위는 클럭사이클이다.

기존 연산회로와의 비교를 위하여 cronkite 영상을 80287 coprocessor(100nsec의 클럭 사용)를 이용할 경우와 제안한 클러스터링 프로세서로 처리할 경우의 처리속도를 표 1에서 비교하였다. 그 결과 약 200배 정도 제안된 프로세서의 속도가 빠른 것을 알 수

표 1. 처리시간의 비교
Table 1. Comparison of processing time.

Unit(Sec.)

CLUSTER	PROPOSED PROCESSOR	80287 CO-PROCESSOR
2	0.0246	5.50
4	0.0442	11.10
8	0.0836	22.31
16	0.1622	44.72
32	0.3190	88.55

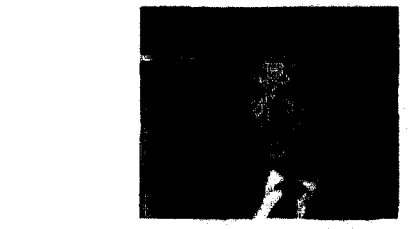
있다.

또한, 오차 비교를 위하여 몇가지 영상에 대한 영상분할 실험을 하였다. 영상분할 실험의 결과, 클러스터의 수가 적을 경우에는 2 진수체계를 이용한 부동소숫점 연산결과와 잉여수체계를 이용한 결과가 거의 일치하였다. 그러나 클러스터 수의 증가에 따라 오차도 증가하였으며 결과는 표 2 와 같다.

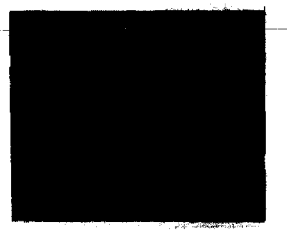
표 2. 영상분할 실험결과
Table 2. Experimental result of image Segmentation.

Image	Grey Level	Cluster Number	Pattern Number	Misclassified Pattern Number	Error (%)
"LINCOLN"	64	2	4096	5	0.12
		3	4096	70	1.70
"HOUSE"	64	2	4096	0	0.
		3	4096	0	0.
		4	4096	160	3.90
"CRONKITE"	256	2	16384	0	0.
		3	16384	72	0.43
		4	16384	612	3.73

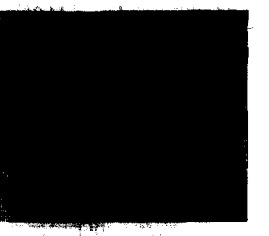
그림 6 은 영상분할 실험에 의하여 구해진 결과의 예이다. 그 결과 육안으로는 거의 오차를 인식할 수 없었다. 따라서 제안된 클러스터링 프로세서는 많은 데이타를 실시간으로 처리해야 하는 영상처리 분야에 효과적으로 이용할 수 있음을 확인할 수 있었다.



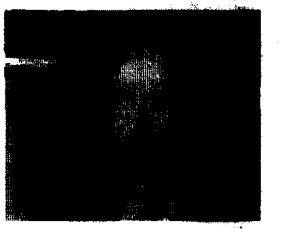
(a) Original image



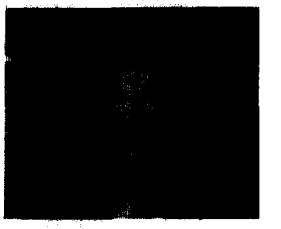
(b) 2 cluster of float



(c) 2 cluster of RNS



(d) 3 cluster of float



(e) 3 cluster of RNS

(f) 4 cluster of float

(g) 4 cluster of RNS

그림 6. Cronkite 영상에 대한 영상분할 결과
Fig. 6. Image segmentation result of cronkite image.

V. 결 론

본 논문에서는 잉여수체계를 이용하여 통계적 패턴 클러스터링을 실행하는 자승오차 패턴 클러스터링 프로세서를 설계하였다. 설계된 클러스터링 프로세서를 대규모의 특징벡터행렬 데이타를 실시간으로 처리해야 할 필요가 있는 영상분할에 적용한 결과, 패턴인식과 영상처리 분야에 효과적으로 이용할 수 있음을 확인하였다.

입력패턴과 표준패턴간의 유클리디언 거리에 의한

최소거리 판단에 바탕을 둔 자승오차 패턴 클러스터링 알고리듬은 일반적인 디지탈 컴퓨터를 이용할 경우에 특징벡터행렬 연산의 반복적인 특성에 의하여 상당한 처리시간을 필요로 하기 때문에 실시간 처리가 어렵다. 그러나 제안된 클러스터링 프로세서는 병렬처리와 파이프라인 구조로 설계하였으며, 잉여수체계를 이용하여 고속의 $(a_j-b_j)^2$ 연산을 실행할 수 있기 때문에 실시간 처리를 필요로 하는 응용분야에 이용할 수 있을 것으로 기대된다. 또한, 연산모듈의 규칙적 배열에 의하여 회로를 구성할 수 있으므로 VLSI로 제작하기가 용이한 장점이 있다.

앞으로의 연구과제는 본 논문에서 제안한 자승오차 패턴 클러스터링 프로세서를 패턴인식과 영상처리 분야에 효과적으로 응용할 수 있는 알고리듬에 대한 연구와 알고리듬에 따른 하드웨어 크기의 최소화에 대한 연구가 필요하며 결과로 패턴인식과 영상처리 분야에 응용하면 실시간 처리의 실현이 가능할 것으로 기대된다.

參 考 文 獻

[1] 조원경, "RNS를 이용한 연산 프로세서의 설계에 관한 연구", 한양대학교 박사학위 논문, 6, 1986.

[2] L.M. Ni and A.K. Jain, "A VLSI systolic architecture for pattern clustering," *IEEE Trasn. on PAMI*, vol. PAMI-7, no. 1, Jan. 1985.

[3] C.H. Huang, "A fully parallel mixed-radix conversion algorithm for residue number applications," *IEEE Trasn. on Comp.* vol. C-32, no. 6, pp. 398-403, April 1983.

[4] Z.D. Ulman, "Sign detection and implicit-explicit conversion of numbers in residue arithmetic," *IEEE Trans. on Computers* vol. C-31, no. 6, pp. 590-594, June 1983.

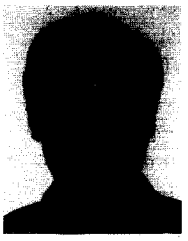
[5] K.S. Fu, "*VLSI for pattern recognition and image processing,*" Springer-verlag. March 1984.

[6] H.H. Liu and K.S. Fu, "VLSI architecture for minimum-distance classification," *Proc. Intern. Conf. on Computer Design: VLSI in Computers*, 1983.

[7] N.S. Szabo and R.I. Tanaka, "Residue arithmetic and its application to computer technology," McGraw-Hill, 1967.

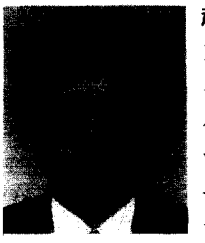
[8] G.B. Coleman and H.C. Andrews, "Image segmentation by olustering," *IEEE Proc.* vol. 67, no. 5, pp. 773-785, May 1979. *

著 者 紹 介



金 炯 民(正會員)

1961年 11月 2日生. 1985年 2月 경희대학교 전자공학과 졸업. 공학사 학위 취득. 1988年 8月 경희대학교 대학원 전자공학과 졸업. 공학석사학위 취득. 주관심분야는 패턴인식과 영상처리, RNS, 신호처리 등임.



趙 源 敬(正會員)

1948年 3月 19日生. 1971年 2月 경희대학교 전자공학과 졸업. 공학사 학위 취득. 1974年 2月 한양대학교 대학원 전자공학과 졸업. 공학석사학위 취득. 1986年 8月 한양대학교 대학원 전자공학과 졸업. 공학박사학위 취득. 1980年～현재 경희대학교 전자공학과 부교수. 주관심분야는 디지탈 회로설계 및 신호처리, 패턴인식, RNS, 영상처리 등임.