한글 인식에서의 NOISE 처리에 관한 연구

임 헌규, 김 진형

(한국과학기술원 전산학과)

요 약

인쇄체 한글 인식 시스템에서 실제의 문자 인식을 위한 데이타는 한 문자를 구성하는 STROKE들의 집합으로 볼 수 있다. OPTICAL SCANNER 의 출력과 같이 문자 영상이 m * n 의 BINARY ARRAY로 주어질 때 STROKE를 추출하기 위해서는 여러 단계의 전처리 과정을 거쳐야 한다. SCANNER 로부터 입력하는 과정이나 전처리 부분에서 NOISE(실제의 영상에는 존재하는 부분이 없어지게 되거나 그 반대의 경우와 실제의 영상에 나타난 영상의 특성을 잃게 되는 경우) 가 발생하게 된다. 이러한 NOISE 를 고려하지 않았던 기존의 문자 인식 시스템 에서는 특히 연결 되어야 할 부분이 끊어진 경우 정확한 인식이 매우 어려웠다. 본 논문에서는 이러한 NOISE 를 어느 정도 제거하는 방법과 또한 새로운 음소의 내부 표현 방법에 의해 위와 같은 NOISE 에 의하여 하나의 음소가 두 개 이상의 CONNECTED COMPONENT 로 나뉘어져도 인식 수 있는 알고리즘을 제시하고자 한다.

I. 서 론

문자의 인식을 위해서는 OPTICAL SCANNER 로 부터 입력된 영상을 THRESHOLDING에 의하여 우선 이진 형태로 바꾸고 다시 이 이진 영상을 세선화 (THINNING PROCESS) 과정을 거친 후 이 영상으로 부터 STROKE를 추출해 내어 이들로부터 음소 추출 및 문자 인식을 하게 된다. [1] 한글의 인식은 보통 음소 단위로 하고 인식된 음소들을 조합함으로써 하나의 문자를 인식하는 것이 보통이다. [2] 이를 원활히 하기 위해서는 하나의 문자를 몇개의 음소 단위의 COMPONENT 로 분리하여 이 COMPONENT 들을 음소 추출의 기본 단위로 문자 인식이 이루어져야 한다. [2,3]

현재까지의 한 음소의 추출 방법으로는 주로 STRUCTURAL PATTERN에 의한 방법[4,5] 과 SYNTACTIC APPROACH[3,6,7] 가 많이 사용되고 있다. 두 방법 모두 한 음소의 인식을 위해서는 일련의 입력 STROKES 가 KNOWLEDGE BASE의 MODEL STROKES 와 일치되어야 한다. 따라서 하나의 음소는 하나의 CONNECTED COMPONENT 에 포함되어야만 정확하게 문자 인식이 이루어질 수 있다. 그러나 위와 같은 NOISE 가 포함되는 경우는 하나의 음소가 두 개 이상의 CONNECTED COMPONENT 로 나뉘어지게 되거나 두 개 이상의 음소가 하나의 CONNECTED COMPONENT 로 나뉘어지게 되므로 음소 단위의 인식이 매우 어렵게 된다.

본 논문에서는 NOISE 에 의한 문자의 오인식을 최소화 하기 위하여 다음의 세 가지 기법을 제시한다. 첫째는 STROKE 추출 바로 전에 끊어짐의 정도가 크지 않은 곳을 연결할 수 있는 처리를 한다. (제2장) 두번째는 NOISE 에 의하여 연결 부위가 끊어져 하나의 음소가 두 개 이상의 블록으로 나뉘어져 인식 과정 중에 두 블록의 결합이 요구될 때 PARAMETER 의 값을 변경하여 첫번째 방법과 같이 연결하도록 하였다. (제3장 4절 나항) 세번째는 음소의 새로운 표현 방법을 사용함으로써 NOISE 에 의해 한 두 곳이 연결되지 않아도 다른 STROKE에 의해 연결만 되어있으면 음소를 인식할 수 있다.

첫번째, 두번째 과정을 거쳐 입력 영상이 수정되어 원래의 영상을 회복하게 되고 비록 제거되지 않은 NOISE가 있더라도 세번째 과정에서 커다란 영향 없이 문자 인식을 할 수 있다.

II. 단락된 부분의 검색과 연결

하나의 문자를 인식하기 위해서는 우선 OPTICAL SCANNER에 의해 얻어진 영상을 THRESHHOLDING에 의해 이진 영상으로 전환하여야 한다. 이 이진 영상에 세선화 처리를 함으로써 세선화된 글자의 영상을 얻게된다. [8] 이 세선화된 영상에 "Hilditch의 알고리즘을 적용하여 이진 영상의 각각의 화소가 어떤 유형의 STROKE에 속하는가를 알아낼 수 있다. [1,9] Hilditch의 알고리즘은 다음과 같다.

STROKE의 일부분인 값이 1인 모든 화소에 대하여 다음의 처리를 한다.
* <그림 1>에 나타난 바와 같은 방향성에 대한 값을 결정한다. 단 한 STROKE의 끝 점은 바로 인접한 화소의 값과 같다.
* 주위의 인접한 두 개의 화소의 값과 자신의 값의 평균치를 구한다.
* 위에서 구해진 평균치에 의해 다음과 같이 분류한다.
  H(0.0-22.5) V(67.5-112.5) L(112.5-157.5) R(22.5-67.5)

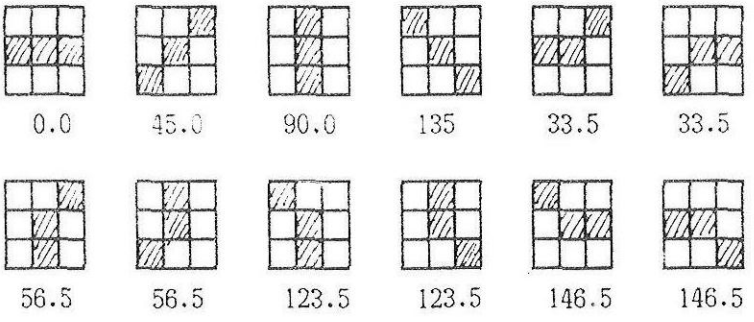

그림. 1 중간 화소의 방향성

위와 같은 알고리즘의 일부를 변경하여 그림2-b의 영상을 처리함으로써 그림2-c와 같은 결과를 얻을 수 있었다. 여기서 '+' 표시가 되어있는 부분이 연결되어야 할 점인데 NOISE에 의하여 끊어진 부분이다.

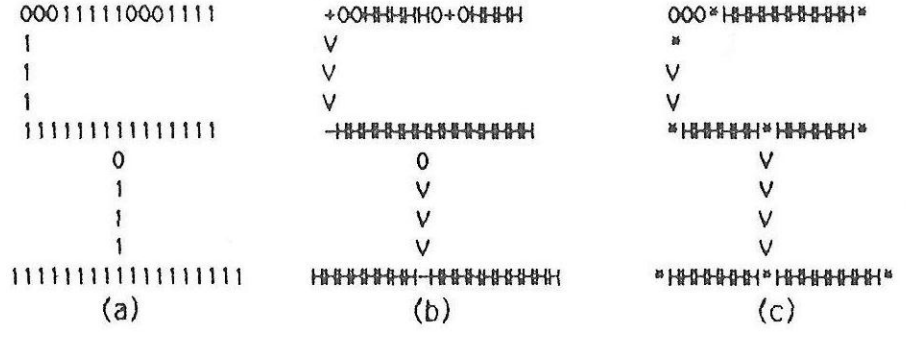

그림. 2 (a) 세선화된 이진 영상 (b)Hilditch의 알고리즘 적용 결과
(c) 단락 부분의 검색과 연결 결과

이 끊어진 부분을 연결하기 위하여 다음과 같은 과정을 수행한다.

*STROKE를 구성하는 모든 점에 대하여 5를 할당하고 나머지 점에는 0을 할당한다.
*모든 ENDING POINT, (i, j) 에 대하여 7을 부여하고 이웃하는 점 (k,l) 의 STROKE 유형과 (i,j) 와 (k,l) 의 위치에 따라서 해당 STROKE가 계속 연결될 수 있는 방향을 찾아 예측되는 m개의 점 (ENDING POINT는 제외) 에 5를 더한다.
*다시 모든 ENDING POINT에 대하여 연결 부위가 예측되는 점들 중 값이 가장 큰 점을 찾으면 그 점이 한 STROKE의 ENDING POINT 혹은 CROSSING POINT,(x,y)가 된다. 이 점으로부터 ENDING POINT (i,j) 사이에 있는 점들의 STROKE 유형은 점 (k,l) 의 값과 같게 한다.
*모든 ENDING POINT와 CROSSING POINT, (i,j) 에 대하여 주위의 STROKE 화소가 두 개 이며 STROKE 유형이 같은 경우는 (i,j) 의 값을 주위의 STROKE 유형과 같게 한다.

이렇게 함으로써 NOISE 에 의하여 두 개의 STROKE로 나뉘어진 것을 한 STROKE로 복구시킬 수 있다.

그림2-b에 m의 값이 2인 경우에서 위의 과정을 수행한 결과가 그림2-c에 있다. 연결이 필요한 곳은 두 곳이 있으나 m의 값에 의하여 한 곳만이 연결되었다. 또한 연결이 필요치 않은 점에서도 새로운 CROSSING POINT가 생길 수도 있어 두 개의 음소가 하나의 CONNECTED COMPONENT (CC)로 뭉쳐지게 되지만 하나의 음소가 두 개 이상의 CC로 나뉘어진 경우보다는 인식하기가 쉬우므로 큰 문제가 되지는 않는다. 최상의 결과를 얻기 위해서는 문자의 크기에 따라 m 값을 조정하여야 한다.

III. 한글 인식

1. 문자의 구성

한글의 문자 구성은 최대 6개의 자음과 모음의 조합에 의해 일정한 6가지의 기본 형태로 분류할 수 있다. 그림3에 6가지의 기본형 중 가장 복잡한 형태인 유형-6의 문자 구성을 보여주고 있다. 그러나 C1과 C2의 경우 복자음일 때는 두 개의 블록으로 나누어지게 되므로 블록을 음소 추출의 기본 단위로 하기 위해서는 유형-6의 문자는 그림4와 같이 나누는 것이 보다 자연스럽다.

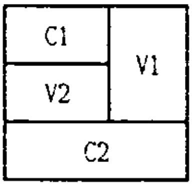

그림. 3 유형-6

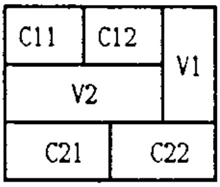

그림. 4 새로운 유형-6

그림4의 각 음소의 위치에 나올 수 있는 음소들은 다음과 같다.

C11= [ㄱ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ ㅎ]
C12= [ㄱ ㄷ ㅂ ㅅ ㅈ]
V1= [ㅣ ㅓ ㅕ ㅏ ㅑ ㅔ ㅒ ㅖ ㅐ]
V2= [ㅗ ㅛ ㅜ ㅠ ㅡ]
C21=C11
C22= [ㄱ ㄷ ㅂ ㅅ ㅈ ㅌ ㅍ ㅎ]

2. 기본 형식의 예측

한 문자는 연결되어있는 STROKE들을 한데 묶어서 몇개의 CONNECTED COMPONENT로 나뉘어질 수 있다. 이것을 음소인식의 기본 단위인 블록(B)으로 하여 각 블록의 상대적인 위치 관계에 의해 문자의 형식을 예측할 수 있다. 즉 각각의 BLOCK이 어떤 음소의 집합에 속하는지를 예측할 수가 있게 되어 비교해야 하는 음소의 수를 줄일 수 있다. 한글 문자의 구성에 있어서 각각의 블록들 간의 관계는 다음과 같이 나타낼 수 있다.

좌우 관계 = [(C11) - (C12) (C11 C12 V2) - (V1) (C21) - (C22)]

상하 관계 = [(C11 C12) - (V2) (C11 C12 V2 V1) - (C21 C22)]

포함 관계 = [([(C11 C21) V2] V1) (C21 C22)]

위에서의 포함 관계는 각 음소의 상대적인 크기에 관한 관계로 C11 블록과 C12 블록을 하나의 블록으로 볼 때 V2는 그 블록의 밑에 있음을 말해주고 있다.

위와 같은 블록간의 위치 정보에 의하여 한 문자의 기본 형식을 예측할 수 있다. 그러나 '눈' 이나 '곳' 과 같이 하나의 음소가 그 이상의 블록들로 구성되는 음소는 인식 중의 복잡한 처리를 피하기 위해서 C11 이나 C21 의 위치에 해당하는 블록들 중 하나의 소평 STROKE만으로 되어 있는 경우는 그 블록의 밑에 있는 블록과 합하여 하나의 블록으로 하여 처리한다. 또한 그림 5 와 같이 하나의 블록 B1 이 다른 블록 B2 에 포함되는 경우는 B1 의 크기가 일정 수준 이상이 되어 하나의 블록으로 고려될 수 있으면 B2를 두개의 블록으로 나누어 처리한다.

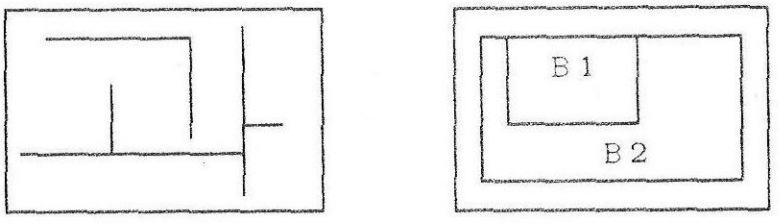

그림. 5 블록의 포함 관계

3. 음소의 내부 표현 방법

음소의 인식을 위해서는 해당 음소에 대한 시스템 내부의 표현이 가능한 한 정확하게 표현되어야 하며 또한 어느 정도의 NOISE 에는 영향을 받지 않도록 표현되어야 한다. 입력 데이타의 NOISE 를 감안해 보면 STROKE의 정확한 위치 및 정확한 연결 부위, 일정한 순서등에 의한 음소의 MATCHING 보다는 전체적인 균형, 즉 STROKE의 상대적 위치 관계 및 연결 상태등에 의해 음소를 MATCHING하는 것이 보다 정확한 인식이 되므로 본 시스템에서는 다음과 같은 음소의 지식 표현 방법을 사용한다.

```
음소  =  (명칭  S1  S2  . . .  SM)
Si    =  (S#  TYPE  ADR RL NST)
S#    =  한 음소 내에서의 STROKE들의 일련 번호
TYPE  =  H  V  R  L 의 SUBSET
ADR   =  (A B) ;INPUT STROKE의 기울기의 허용 범위
```

```
RL   =  (A B) ;STROKE들의 평균 길이에 대한 상대치 허용 범위
NST  =  (C1 C2 ... Ck) ;이웃한 STROKE들
Ci   =  (RP ST# CR)
RP   =  R(IGHT) L(EFT) U(P) D(OWN) L-D L-U R-D R-U  ;STROKE들의 상대적인
                                                    위치관계
CR   =  (A B) ;연결 점의 S#에 대한 상대적 위치의 허용 범위
        연결되지 않은 경우는 nil 이 된다.

('ㄷ (1) (H) (15  -5 ) (0.7  1.2) ( (L-D 2 (0.0 1.0)) (D 3 NIL)))
     (2) (V) (105  80) (0.8  1.2) ( (R-U 1 (0.0 1.0)) (R-D 3  (0.9 1.0))))
     (3) (H  (30  -5 ) (0.7  1.2) ( (L-U 2 (0.0 0.1)) (U 1 NIL)))
```

그림5 'ㄷ'의 표현 예

위의 표현 방식에 의한 'ㄷ'의 표현 예가 그림5에 있다. 두개의 수평 STROKE와 하나의 수직 STROKE로 되어 있으며 위의 수평 STROKE는 입력된 STROKE의 각도 차가 15에서 -5사이에 있으면 MATCHING이 허용되며 세 STROKE의 평균 길이에 대한 상대적인 길이가 0.7 배와 1.2 배의 범위에 속해야 한다. 또한 이 수평 STROKE에 연결된 STROKE는 2번의 수직 STROKE로서 1번 STROKE의 앞쪽 끝에 연결되어 있음을 나타내고 있다. 그리고 연결되지 않은 같은 TYPE의 수평 STROKE가 아래쪽에 있음을 표현하고 있다. STROKE 2, 3에 대해서도 같은 방법으로 표현된다. STROKE의 상대적인 길이를 표현해 줌으로써 하나의 블록에서 MATCH 된 한 음소를 떼어낼 때 연결되어 하나의 STROKE처럼 된 STROKE를 찾아내어 다른 STROKE가 MATCH 된 것처럼 되지 않도록 할 수 있다. STROKE의 TYPE이 H V R L 의 SUBSET인 것은 음소의 한 STROKE가 두 개 이상의 STROKE 유형이 될 수 있기 때문이다.

4. 문자의 인식

앞에서와 같은 처리 과정에 의해 어느 정도 NOISE가 제거된 영상으로부터 STROKE들을 추출하고 몇 개의 CONNECTED COMPONENT (혹은 블록)를 구성하여 각각의 블록이 속하는 음소의 집합이 예측되어 문자의 형신을 예측할 수 있다. 이 예측된 문자의 기본 형식에 의해 다음의 인식 과정을 수행한다.

입력 데이타 B1 B2 B3 .... Bm 에 대하여 (3장 2절)
[STEP1] Bi에 예측되는 음소의 집합, Gi를 구한다. (3장 2절)
(MATCH가 완료된 블록의 정보를 이용)
[STEP2] 모든 Bi에 대하여 Gi의 모든 음소와 MATCHING (제3장4절-가항)
(C11=> C12 => V2 => V1 => C21 => C22 의 순으로 Bi를 선택)
[STEP3] MATCH 결과가 'INCOMPLETE'이면 MATCH 된 음소들은 STACK에 넣어놓고 예상되는 위치의 블록 Bj를 결합. (제3장4절-나항) [STEP1]으로
[STEP4] MATCHED 이며 MATCH 되지 않고 남은 STROKE가 있으면 MATCHE된 것과 MATCHE 되지 않은 STROKE들의 블록으로 나눈다.
[STEP5] MATCHED 이면 가장 복잡한 (가장 많은 STROKE로 구성된) 음소를 인식된 음소로하고 나머지는 STACK에 넣는다.
[STEP6] UNMATCHED 이면 STACK의 TOP에 있는 음소의 인식 단계까지 BACKTRACKING 만일 STACK이 비어 있으면 RETURN 'UNMATCHED-CHAR.
[STEP7] [STEP1]에서 부터 반복

위의 수행 과정중 STACK에 저장되는 음소들은 한 자소 (C11 C12 V1 V2 C21 C22 중의 하나)로서 인식되었으나 보다 더 복잡한 음소가 MATCHE된 것이 있어 뒤로 밀려있던 것이다. [STEP6]가 수행된다는 것은 그 복잡한 음소의 MATCHING이 잘못 인식된 것임을 말해준다.

가. 음소의 MATCHING

하나의 블록 B 와 한 음소 P 와의 MATCH 는 다음과 같이 수행된다.

*STROKE의 중심점에 의하여 블록 내의 STROKE들을 SORTING
*B 의 (첫번째) STROKE(BS)와 MATCH 되는 P 의 STROKE (PS)를 찾는다.
*P 의 모든 STROKE가 MATCH 되었으면 RETURN 'MATCHED'
*BS에 연결된 STROKE와 그에 MATCH 되는, PS에 연결된, 그러나 아직 MATCH 되지 않은 모든 STROKE S를 찾는다.
*B 가 EMPTY 이면 RETURN 'INCOMPLETE'
*BS 와 PS의 연결된 STROKE들 중 같은 위치에 서로 다른 STROKE가 있으면 'UNMATCHED'
*S의 모든 STROKE에 대하여 2번째 과정부터 반복.

위와 같은 수행에 의하여 모든 예상 음소에 대한 MATCH 가 이루어진다. 이렇게 하여 한 블록에 대해 두개 이상의 MATCHED 음소가 있을 수 있고 동시에 INCOMPLETE MATCH 도 있을 수 있다. INCOMPLETE MATCH가 있을 경우는 MATCH 된 음소와 INCOMPLETE된 음소들을 넘겨준다. 이와 같은 처리는 MATCH 될 수 있는 음소들 중에 가장 복잡한 음소의 MATCH 를 위함이다.

나. 두 블록의 결합

한 음소가 NOISE 등에 의하여 두개 이상의 블록에 나뉘어져 있는 경우 한 블록의 MATCH 결과가 'INCOMPLETE'가 되어 주위의 다른 블록과 합쳐 하나의 블록으로 처리되야 한다. 즉 이러한 경우는 NOISE 의 정도가 커서 STROKE 추출 전에 처리되지 못한 부분이다. 이것은 전단계에서의 NOISE 처리시에 m 의 값이 작음으로 해서 처리되지 못한 것이다. 그러나 m 의 값을 너무 크게하면 관계없는 주위의 블록의 불필요한 부분에 CROSSING POINT가 생겨나게 된다. 그러나 블록의 결합시에는 두 블록이 연결되어야 하므로 보다 큰 m 의 값으로 제2장에서와 같은 처리를 하더라도 크게 문제되지는 않는다. 이렇게 하여 단락되었던 두 블록을 결합할 수 있다.

다. MATCHING의 예 (" 람")

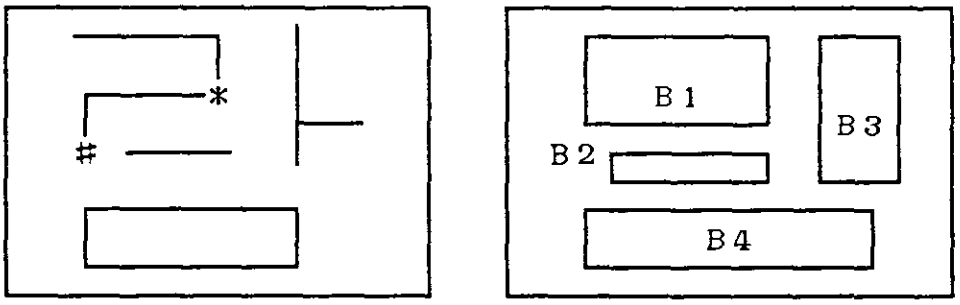

그림. 6 MATCHING의 예제

그림6과 같은 입력 영상에 대한 문자 인식의 예를 보도록 하자. 우선 STROKE 추출전에 제2장에서 논의한 처리를 하면 '*' 지점은 연결이 될 수 있다. 그러므로 우선 생성되는 블록은 B1 B2 B3 B4 등 네 개가 된다. 이때 블록들은 각각 C11 V2 V1 C21 등으로 예측된다. 우선 B1에 대한 C11 에 속하는 음소들과의 MATCH 가 이루어진다. 'ㄱ' 은 MATCHED, 'ㄹ' 은 INCOMPLETE가되어 B1과 B2가 결합된다. 그리고 'ㄹ' 도 MATCH 된다. 이때 STACK 에는 'ㄱ' 이 있다. B1-B2 가 C11 으로 인식이 완료되었고 블록의 크기등 서

로의 관계 정보에 의하여 B3 B4 는 각각 V1 C21으로 예상되고 B3가 'ㅣ' 로 인식된다. 다음은 B4가 'ㅁ' 으로 인식된다. B1의 'ㄹ' 과의 MATCHING 과정은 우선 맨 위의 H-STROKE 에 대하여 MATCH 가 되고 그에 연결된 V-STROKE 가 S에 포함된다. S의 한 V-STROKE와 연결된 그 다음의 H-STROKE가 인식 되고 또다른 V-STROKE가 인식되었지만 아직 하나의 H-STROKE가 MATCH 되지 않았기 때문에 'INCOMPLETE'가 되므로 B1과 B2가 결합해야 한다. 이 때 제 3장 4절 나항에 의해 연결되지 않은 # 위치에서 두 STROKE가 연결된다. 이때 B1 과 B2의 예상 음소는 C11 의 모든 음소들이 아니고 단지 'ㄹ' 만이 된다. 그리하여 결국에는 'ㄹ' 로 인식된다. STACK 에 있는 'ㄱ' 은 더 복잡한 음소 'ㄺ' 이 잘못 인식된 것이 아니므로 불필요하다. 이렇게 하여 한문자의 인식이 완료된다.

IV. 결 론

본 논문에서는 한글 인식 시스템에서의 NOISE 처리에 관한 연구가 소개되었다. NOISE 의 처리에는 두가지 방향이 있는데 한 방향은 NOISE 를 직접 제거하는 방법으로 제2장과 제3장 3절 나항에서 설명 되었다. 이 두 법의 처리로 인하여 별다른 문제점 없이 어느 정도 단락된 경우에 대해서 이 처리를 거치지 않고 인식한 경우에 비하여 문자 인식율이 훨씬 높아졌다. 또 다른 방향으로는 음소의 새로운 내부표현 방법에 의한 음소의 MATCHING 방법으로서 제3장 3절에서 논하였다. 각 음소들끼리의 상호 연결 관계를 음소의 지식 표현에 이용 함으로써 비록 연결 부위 중 한 곳이 끊어졌더라도 다른 STROKE들에 의해 연결되어 있으면 음소의 인식이 가능함을 보였다.

참 고 문 헌

[1] Hideo Ogawa and Keiji Taniguchi "Preprocessing for Chinese Character Recognition and Global Classification of Handwritten Chinese Characters*", Pattern Recognition, Vol. 11, PP 1-7, 1979

[2] Choi, B. U., T. Ichikawa and H. Fujita, "A Method of the Extraction of Phonemes in Hangeul Recognition", 전자공학회지 제18권 제2호 Apr,1981

[3] S. H. Ahn, "Syntactic Recognition of Printed Korean Characters" M.S. Thesis, Dept. of CS, KAIST, Seoul Korea, 1985

[4] S. K. Han, "The Recognition of Korean Characters by the Structural Approach" M.S. Thesis, Dept. of CS, KAIST, Seoul Korea, 1984

[5] Jong Wook Park and Joo Keun Lee, "Pecognition of Handwritten-Hangeul by Shape Patter", 전자공학회지 제22권 제5호, 9,1985

[6] J. K. Lee, J. C. Namkung and Y. K. Kim, "A Study on the partial separation for subpatterns and recognition of the Hanguel patterns", "J. KIEE", Vol, 18, No.3 June 1981, PP. 1-8

[7] J.C. Namkung, "A Study on the partial seperation and recognition of Hanguel patterns by the indexwindow algorithm", Ph.D. Dissertation, Inha Univ., Inchon Korea, 1982

[8] NABIL JEAN NACCACHE AND RAJJAN SHINGHAL, "SPTA:A Proposed Algorithm for Thinning binary Patterns", IEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS VOL. SMC-14, NO.3, MAY 1984

[9] C.J.Hilditch, "Liner Skeleton from sSuare Cupboards", Machine Intelligence, Vol.G, PP. 403-320,1969