Journal of the Korean Institute of Industrial Engineers
Vol. 17, No. 2, Dec , 1991

인쇄체 한글 문자 인식에 관한 연구†

장승석* · 장동식*

The Recognition of Printed HANGUL Character

Seoung-seok Jang* and Dong-sik Jang*

Abstract

A recognition algorithm for Hangul is developed by structural analysis to Hangul in this theses. Four major procedures are proposed : preprocessing, type classification, separation of consonant and vowel, recognition. In the preprocessing procedure, the thinning algorithm proposed by CHEN & HSU is applied. In the type classification procedure, thinned Hangul image is classified into one of six formal types. In the separation of consonant and vowel procedure, starting from branch-points which are existed in a vowel, character elements are separated by means of tracing branch-point pixel by pixel and comparison with proposed templates. In the same time, the vowels are recognized. In the recognition procedure, consonants are extracted from the separated Hangul character and recognized by modified Crossing method. Recognized characters are converted into KS-5601-1989 codes.

The experiments show that correct recognition rate is about 80%-90% and recognition speed is about 2-3 character persecond in three types of different input data on computer with 80386 microprocessor.

1. 서 론

영상화된 문자를 해당 코드로 변환시키는 문자 인식 기능은 각종 문서 정보의 효율적인 컴퓨터 입력 장치로서 여러 분야에 있어 그 필요성이 인정된다. 이러한 문자 인식 기능은 컴퓨터에서 문자 입력 수단으로 이용되었던 키보드를 대치함으로써 주변 기기에 의한 정보 처리의 병목현상을 감소시

† 이 논문은 한국과학재단의 목적기초연구비 지원으로 이루어졌음.
* 고려대학교 공과대학 산업공학과.

켜서 현대의 정보화, 자동화 사회의 구현에 크게 기여할 것이라고 예상된다.

한글은 자음과 모음의 기본 자소들이 결합하여 하나의 문자를 생성 하는데 생성 가능한 문자수는 14,000여자 되며 [6], 일상 생활에 사용되는 문자만 해도 1,500여자나 된다[9]. 이와같이 한글 문자 구성의 복잡성과 엄청난 문자 갯수 때문에 개개의 한글 문자를 인식하려는 방법은 인식 시간과 비교를 위한 표준 데이터 베이스 구성에 많은 문제점을 내포하고 있다. 따라서 본 연구에서는 문자 고유의 특징을 그대로 유지하면서 연산량을 줄이기 위해 입력 영상에서 문자의 골격만을 추출하는 세선화 (Thinning) 과정을 거치고 세선화된 한글 문자에서 자음과 모음을 분리함으로써 인식 대상을 24개의 기본 자소로 한정하고 이를 개별적으로 인식한 후 결합하는 방법을 취하였다.

한글의 구조적인 특징을 이용하는 기존의 많은 방법에서는 한글의 기본요소(Primitive)를 획(Stroke)으로 정의 하는데, 세선화된 문자 이미지에서 획을 추출할 경우 노이즈의 영향과 불필요한 굴곡점으로 인해 인해 인식이 어렵게 되며 많은 시간이 소요되는게 보통이다[5] 따라서 본 연구에서는 자음과 모음의 구조적인 특징을 이용하면서도 기존의 방법을 따르지 않음으로써 인식속도를 빠르게 하고 노이즈와 문자 형태, 크기에 덜 민감한 방법을 이용하였다.

2. 한글 인식 알고리즘

2-1. 한글 문자 이미지 입력 및 세선화

한글 문자 이미지가 Bitmap 상태로 입력되면 다음 그림 1과 같이 글자의 영역을 가로 4등분, 세로 6등분 한다. 이렇게 함으로써 자음과 모음이 존재하는 영역을 parameter들로 표시할 수 있게 된다.

다음으로 세선화 과정을 거치는데, 본 연구에서

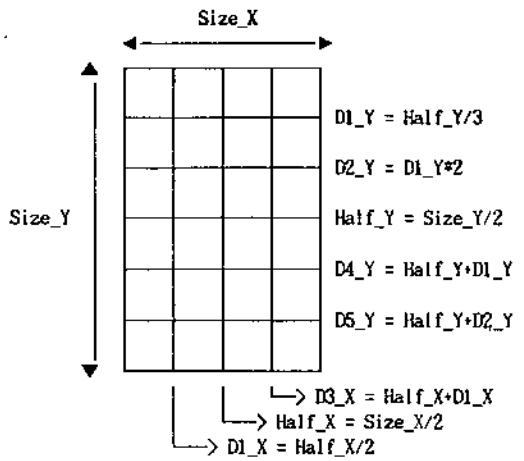


그림 1. 한글 문자 영역 분할

사용한 세선화 알고리즘은 Zhang and Suen이 제시한 fast parallel thinning algorithm[2]을 Chen and Hsu가 수정한 것인데 Zhang and Suen algorithm의 장점을 유지하면서 단점(과도한 shrinking 현상, 8-neighbor connectivity 문제)을 극복한 것이다[1].

2-2. 유형 분류

한글은 모음의 형태와 종성의 존재 유무에 따라 6가지 유형으로 분류할 수 있는 데[7], 6가지 한글 문자의 유형은 그림 2와 같다.

한글을 6가지 유형중의 하나로 분류하는 이유는, 자모 분리 과정과 자소 인식 과정에서 처리가 쉽고 정확하게 되도록 하기 위해서이다.

한글 유형을 결정하는 일이 중요한 만큼 유형분류를 실패했었을 때의 손실도 크다. 왜냐하면 유형분류가 잘못되면 이후의 모든 과정이 무위로 끝나고 잘못된 인식 결과를 나타내기 때문이다.

입력된 한글 문자의 유형을 알기 위해서 본 연구에서는 먼저 대분류를 행한다. 대분류란 입력 문자를 한글의 6형식 중 어느 한 형식으로 결정하기

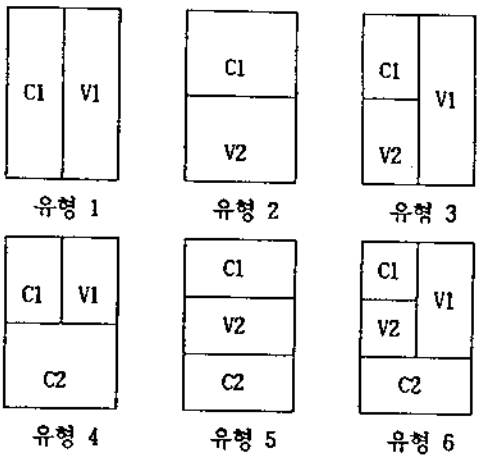


그림 2. 한글 문자의 6가지 유형

전에 긴 수직 모음이 존재하는 한글(유형 1과 유형 3), 긴수평 모음이 존재하는 한글(유형 2와 유형5), 그 외의 한글(유형 4와 유형 6)의 3가지 그룹으로 분류하는 것을 말한다[4].

본 연구에서 정의한 긴 수직, 수평 모음이란 입력 한글 문자의 세로, 가로방향 길이의 0.85 이상의 길이를 가진 선분을 말한다.

입력된 한글 문자가 대분류에 의해 3가지 한글 그룹중 어느 하나에 속하게 되면 정확한 유형을 알기 위해 다음과 같이 소분류를 행한다.

① 유형 1과 유형 3의 분류－짧은 모음 V2의 존재 유무에 의해.

② 유형 2와 유형 5의 분류－자음 C2의 존재 유무에 의해.

③ 유형 4와 유형 6의 분류－짧은 모음 V2의 존재 유무에 의해.

2-3. 모음의 특징추출

모음 V1이 될 수 있는 자소는 아, 야, 어, 여, 애, 얘, 에, 예, 이이고 모음 V2가 될 수 있는 자소는 오, 요, 우, 유, 으이다. 모음들은 모두 한 두개의 수평, 수직 직선으로 연결되어 있는데, 두 직선이 연결되는 점 즉, 분기점(Branch-point)의 방향과 갯수는 모음을 결정할 수 있는 중요한 특징(Feature)이 된다. 또한 자음과 모음이 붙어 있으면 또 다른 분기점이 생기므로 자모 분리에서도 분기점을 이용할 수 있다.

본 연구에서는 모음의 특징으로서 모음 V1과 V2를 pixel by pixel로 따라 가면서 분기점들을 찾는다.

1) 모음 V2에서의 분기점 추출

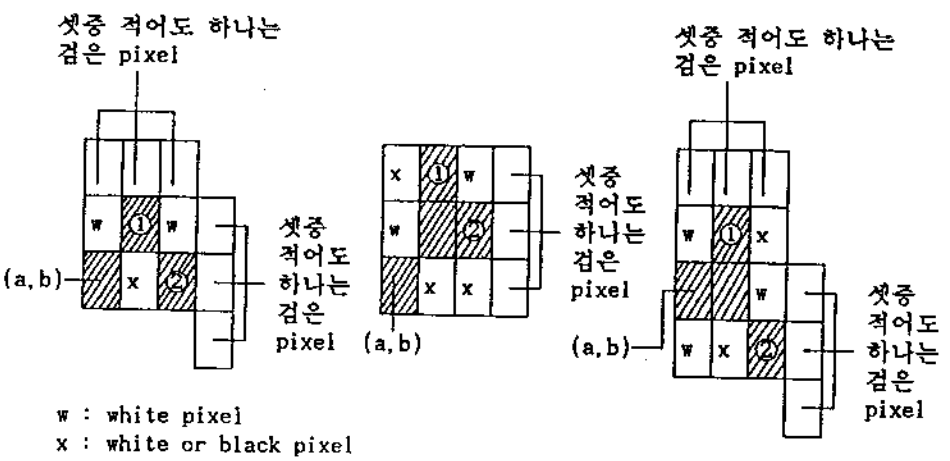


그림 3. 윗쪽 방향의 분기점

가. 윗쪽 방향의 분기점 추출(pixel(a, b)가 중심)

그림 3의 template들 중 하나가 발견되면 윗쪽 방향의 분기점으로 인식하고 ① 위치를 기억한다. 그리고 ② 위치로 이동하여 계속 분기점을 찾아 나간다.

(2) 아래쪽 방향의 분기점 추출(pixel(a, b)가 중심)

그림 4의 template들 중 하나가 발견되면 아래쪽 방향의 분기점으로 인식하고 ① 위치를 기억한다. 그리고 ② 위치로 이동하여 계속 분기점을 찾아 나간다.

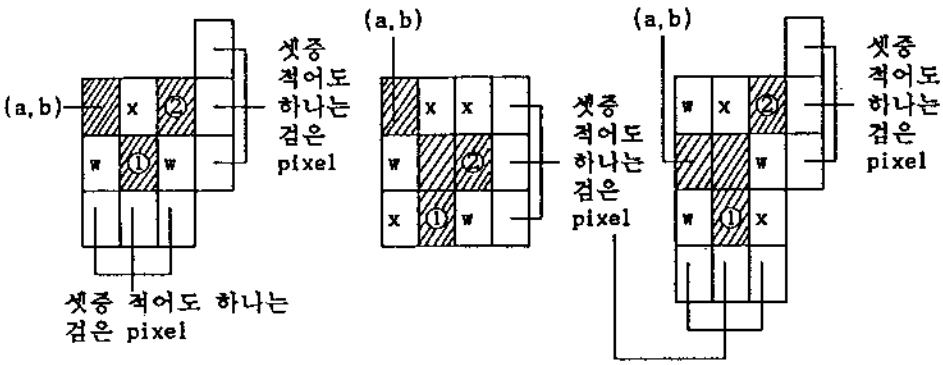


그림 4. 아래쪽 방향의 분기점

나. 모음 V1에서의 분기점 분출

모음 V1이 될 수 있는 모음들 중에서 애, 얘, 에, 예는 수직 모음이 두개 존재하는데 대분류 단계에서 찾은 것은 오른쪽 수직 모음이었다. 그러나 분기점을 찾기 위해서 본 연구에서는 왼쪽 수직모음이 필요하므로 먼저 왼쪽 수직모음이 존재하는지를 조사하여 있으면 아래의 분기점을 찾는다.

(1) 왼쪽방향의 분기점 추출(pixel(a, b)가 중심)

다음 그림 5의 template들 중 하나가 발견되면 왼쪽방향의 분기점으로 인식하고 ① 위치를 기억한다. 그리고 ② 위치로 이동하여 계속 분기점을 찾아 나간다.

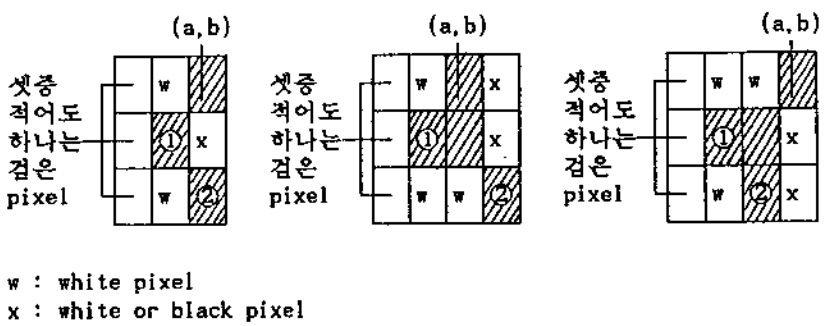
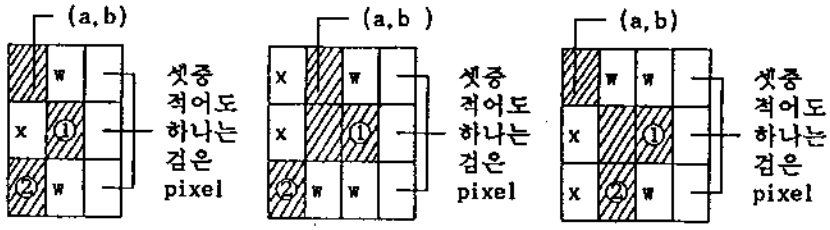


그림 5. 왼쪽 방향의 분기점

(2) 오른쪽 방향의 분기점 추출(pixel(a, b)가 중심)

다음 그림 6의 template들 중 하나가 발견되면 오른쪽 방향의 분기점으로 인식하고 ① 위치를 기억한다. 그리고 ② 위치로 이동하여 계속 분기점을 찾아 나간다.



w : white pixel
x : white or black pixel

그림 6. 오른쪽 방향의 분기점

2-4. 자음과 모음 분리

활자나 레이저 프린터의 문자를 스캐너로 읽은 입력 문자 영상은 각각의 자모가 분리되어 있지 않고 서로가 연결되어 있는 경우가 많다. 이것은 기본 자모의 인식을 어렵게 하여 문자 인식에 장해 요인이 된다. 따라서 자모의 분리는 필수적이며 본 논문에서는 모든 자음과 모음은 서로 붙어 있다고 가정하고 이를 분리해 나간다. 그러나 유형 분류 단계를 거치면서 알게된 유형 정보를 이용하여 그 유형에 존재하는 자음과 모음만 분리하면 된다.

본 연구에서의 자모 분리는 모음 V1이나 V2에 존재하는 분기점들로부터 시작하는데, 모음 V2에 존재하는 윗쪽 방향의 분기점에서는 ⓐ, 아래쪽 방향의 분기점에서는 ⓑ와 같은 template를 사용하여 윗쪽, 혹은 아래쪽으로 나아가면서 3-neighbor ① ② ③을 조사하여 검은 pixel의 갯수를 세고 이것과 현재의 위치 정보를 이용하여 자모음을 분리할 것인가를 따지게 된다. 그리고 모음 V1에 존재하는 왼쪽 방향의 분기점에서만 아래와 같은 template를 사용하여 왼쪽으로 나아가면서 3-neighbor ① ② ③을 조사하여 검은 pixel의 갯수를 세고 이것과 현재의 위치 정보를 이용하여 자모음을 분리할 것인가를 따지게 된다.

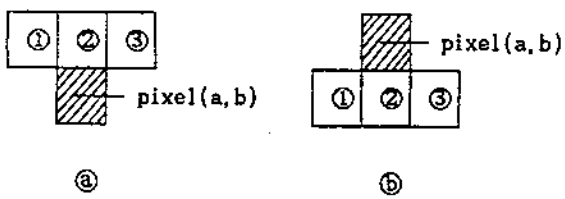


그림 7. 윗쪽, 아래쪽 방향의 templates

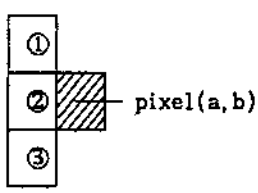
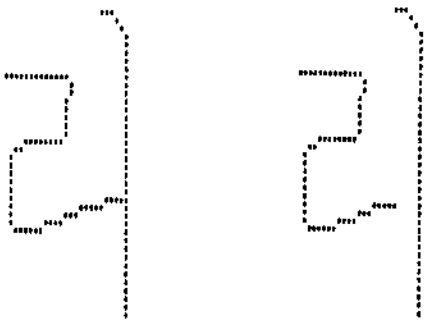


그림 8. 왼쪽, 방향의 templates

다음은 각 유형별로 분리된 결과를 나타낸 것이다.

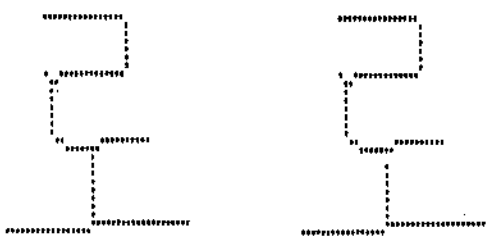
〈 유형 1 〉



분리전 분리후

그림 9. 유형 1에서의 자모분리

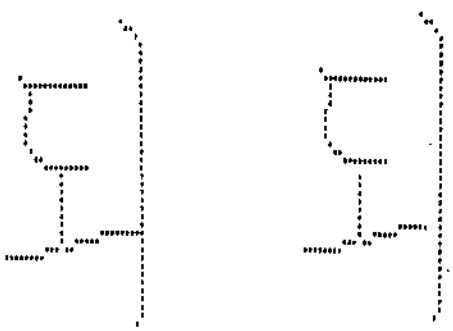
〈 유형 2 〉



분리전 분리후

그림 10. 유형 2에서의 자모분리

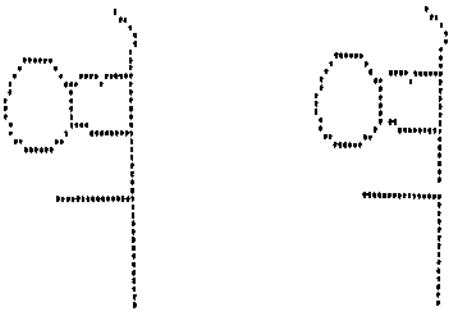
〈 유형 3 〉



분리전 분리후

그림 11. 유형 3에서의 자모분리

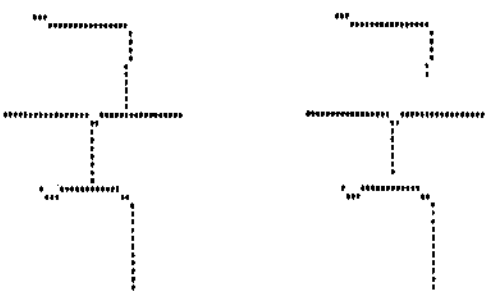
〈 유형 4 〉



분리전 분리후

그림 12. 유형 4에서의 자모분리

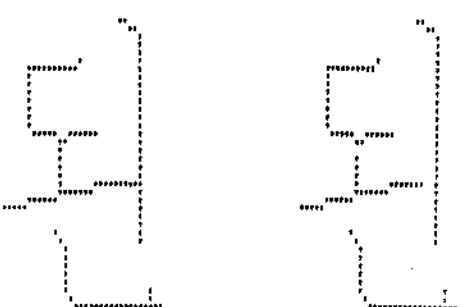
〈 유형 5 〉



분리전 분리후

그림 13. 유형 5에서의 자모분리

〈 유형 6 〉



분리전 분리후

그림 14. 유형 6에서의 자모분리

2-5. 모음 인식

자음과 모음이 성공적으로 분리가 된 후 모음에 존재하는 분기점은 모음 고유의 특성이 되므로 모음을 쉽게 인식할 수 있다.

1) 모음 V1의 인식

(1) 한개의 수직 모음이 존재할 때

왼쪽 방향 분기점이 없을 때 오른쪽 방향의 분기점의 갯수에 따라 이, 아, 야를 인식할 수 있으며, 왼쪽 방향 분기점이 1개이면 어, 2개이면 여로 인식한다.

(2) 두개의 수직 모음이 존재할 때

왼쪽 방향 분기점이 없을 때 오른쪽 방향의 분기점의 갯수에 따라 애와 얘를 인식할 수 있으며, 왼쪽 방향 분기점이 1개이면 에, 2개이면 예로 인식한다.

2) 모음 V2 인식

아래쪽 방향 분기점이 없을 때 윗쪽 방향 분기점의 갯수에 따라 으, 오, 요를 인식할 수 있으며, 아래쪽 방향 분기점이 1개이면 우, 2개이면 유로 인식한다.

2-6. 자음추출 및 인식

자음과 모음이 분리되고 모음이 인식된 후에 자음을 인식하기 위해서 본 연구에서는 한글 문자로부터 따로 자음을 추출하여 인식하는 과정을 행한다. 먼저 초성 자음인 C1을 추출하여 인식하고 그 다음 종성 자음 C2가 있으면 추출하여 인식한다. 종성 자음의 존재 여부는 유형 분류 단계가 끝나면 알게 된다.

자음을 추출한 뒤 다음 그림 15와 같이 정규화 (Normalizing) 작업을 한다. 정규화 작업이란 추출된 자음의 크기에 상관없이 이를 인식하기 위해 먼저 자음의 제일 위, 아래, 왼쪽, 오른쪽의 위치를 알아내어 이를 기준으로 자음의 크기를 구한 다음 가로 방향 길이의 1/4과 2/4, 3/4 위치에서 오른쪽 옆으로 직선을 긋는 작업을 말한다. 그리고 이 직선들을 각각 Line/X/2 그리고 Line/Y/1, Middle /Y, Line/Y/2라 정의한다.

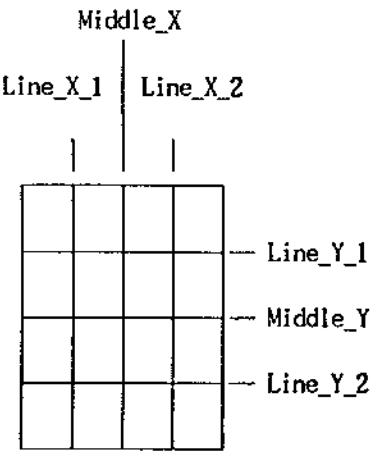


그림 15. 정규화 작업

자음 인식의 방법으로는 먼저 교차특징을 이용한다. 문자에 임의의 검출선을 그어 문자의 획(Stroke)과 만나는 횟수를 살펴보는 방법이다. 그림 16은 Dimond에 의해 제안된 Snode 법으로 오래된 간단한 방법이다[3]. 즉 문자속의 임의의 점을 정해 (일반적으로 2점) 그곳으로부터 검출선을 상하로

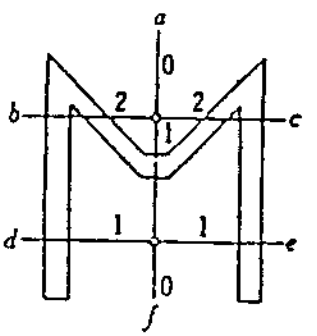


그림 16. Snode법

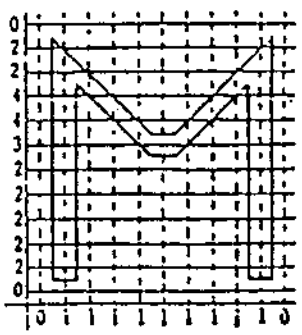


그림 17. Crossing법

그어 통과하는 획의 수를 센다. 이 방법은 문자변형에 크게 영향을 받지 않아 초기의 필기체 문자 인식에 많이 사용되었다. 그러나 category 수가 많을 경우 적용이 곤란하다는 문제점을 가지고 있다. 이러한 문제점을 해결하기 위해서 Doyle에 의해 Crossing 법이 제안되었다[3]. 이 방법은 그림 17에 나타낸것과 같이 검출선을 균일하게 다수의 수평, 수직선을 그어 통과하는 획의 수를 검출하는 방법이다.

본 연구에서는 Doyle의 Crossing법에 기본을 두고 있는데, Crossing법과 다른점은 수평, 수직 검출선의 갯수를 3개로 국한시키며, 검출선과 획과 만나는 곳의 위치 정보도 이용한다는 것이다. 따라서 자음 인식은 Line/X/1, Line/X/2, Line/Y/1, Line/Y/2선과 자음과 만나는 횟수와 위치 정보를 이용함으로써 가능하다. 이것으로써 인식이 불가능할때는 Middle/X 선, Middle/Y 선, 그리고 특정한 template를 이용한다. 또한 대부분의 자음이 유형과 초성, 중성 위치에 따라 형태가 변하기 때문에 이를 고려해야 한다.

몇가지 예를 들면 다음과 같다.

* 기역(ㄱ)의 경우—두가지 종류가 존재

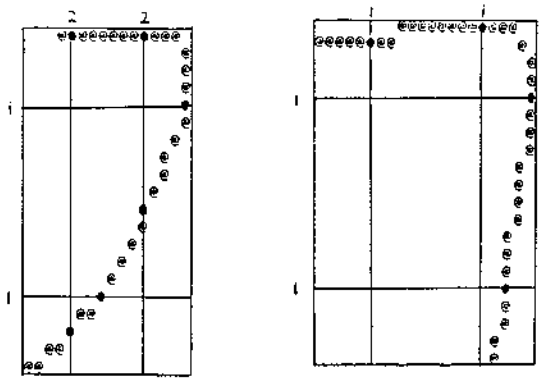


그림 18. 자음 기역(ㄱ)의 표현

미음과 이응은 위와같은 방법으로는 구별이 어렵다. 그러나 이응에는 좌우 대각선 방향의 선분이 존재하는 것을 알 수 있다. 따라서 좌우의 대각선 선분이 존재하면 이응으로 인식한다.

* 니은(ㄴ)의 경우—두가지 종류가 존재

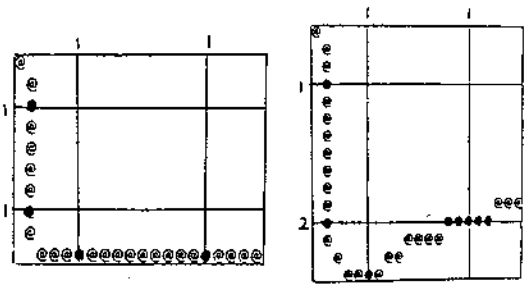


그림 19. 자음 니은(ㄴ)의 표현

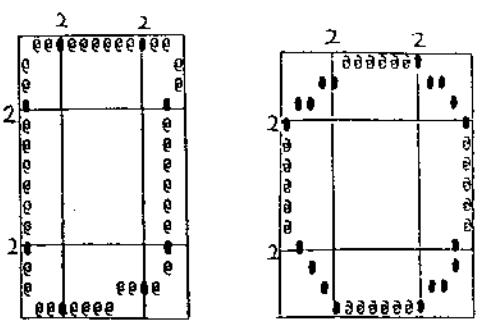


그림 20. 자음 미음(ㅁ)과 이응(ㅇ)의 구별

2-7. KS code로의 변환

인식된 자음과 모음을 결합하여 먼저 조합형 한글 코드로 변환한다. 조합형 한글 코드는 2바이트 조합형 한글 코드체계를 따르는데 원리는 다음과 같다 [8]

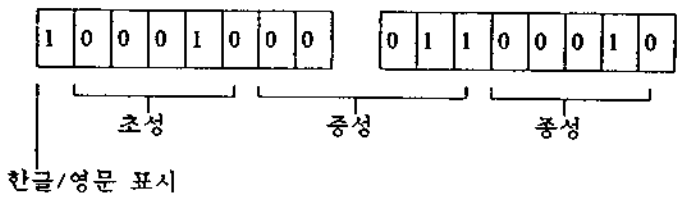


그림 21. 2바이트 조합형 코드의 원리

한글은 초성의 갯수가 전체 19개, 중성의 갯수가 21개, 종성의 갯수가 27개로 5비트 (2^5=32개)만 있으면 각 성을 표시할 수 있다.

위 그림과 같이 2바이트 조합형 한글은 연속된 2개의 바이트를 뒤에서 부터 5비트를 묶어서 각각 종성, 중성, 초성을 표시하는 원리로, 나머지 한 비트에는 1을 설정하여 한글 코드임을 컴퓨터에 알리고 있다.

마지막 단계로서 조합형 코드를 완성형 코드(KS 코드와 동일)로의 변환은 간단한 프로그램으로 가능하다.

3. 실험 결과

본 연구의 실험에서 프로그램은 TURBO C로 구현했고 IBM-PC 386(20 Mhz) 상에서 실행을 시켰다. 글자의 크기와 종이의 재질이 다른, 국정교과서의 국민교육헌장과 일반 잡지 발췌문의 3가지 종류의 한글문서 데이터를 대상으로 실험을 했다.

이들 데이터는 스캐너(Scanner)에 의해 받아 들여져 각 파일에 저장된 것으로 각 파일에는 글자들이 각각 418바이트의 크기안에 담겨져 있다. 그리고 418바이트내에서 처음 6바이트에는 글자 크기에 대한 정보를 담고있다.

각 데이터 파일에는 마침표, 물음표와 같은 기호의 숫자들도 포함되어 있고, 스캐너를 통해서 데이터를 만드는 과정에서 잘못된 글자도 있는 데이들 모두 인식에서 제외하였다. 그리고 인식시간을 세선화 과정부터 KS 5601 코드가 출력될때 까지로 정하였다. 인식시간을 알기위해 TURBO C의

표 1. 본 알고리즘의 실험 결과

데 이 터 파 일 명	인식에 사용된 글자(개)	정인식(개)	오인식(개)	인식율(%)	평균인식시간 (초/글자)
잡지발췌문 1 (평균50×50)	151	138	13	91.39	0.686
잡지발췌문 2 (평균30×30)	345	268	77	77.68	0.302
국민교육헌장 (평균45×45)	380	352	28	92.63	0.423

clock()이라는 함수를 사용했는 데 이 함수는 두 사건사이에 경과된 processor의 시간을 잰다.

제반 실험 결과는 표 1과 같다.

잡지발췌문 2의 인식율은 매우 낮았다. 가장 큰 이유는 글자의 크기가 작고 종이의 재질이 좋지않아 대부분의 자음, 모음이 붙게 되어 이들을 정확히 분리하지 못한 데 있다.

대부분 오 인식은 자모 분리의 실패에 기인한 것이며 유형 분류의 실패에 의한 것도 몇개 발견되었다. 따라서 본 알고리즘에서 인식율을 높이기 위해서는 다음의 몇가지 사항이 개선되어야 한다.

① 입력 data를 만드는 과정에서의 정확성 및 가능한한 노이즈 제거.

② 유형 분류의 정확성.

③ 자음, 모음 분리 알고리즘의 개선.

인식 속도면에서 잡지발췌문 1은 문자당 평균 0.686초 걸렸는데, 이는 문자 크기가 대부분 50×50을 넘기 때문에 이를 처리하는 데 많은 시간이 걸렸기 때문이다. 그외의 데이터 파일에 있는 문자들은 문자당 0.3~0.4초가 걸려 초당 2~3자 정도의 인식 속도를 나타내고 있다. 그러나 전체 인식 시간의 70% 이상을 세선화 과정이 차지하고 있는 데, 이 세선화 과정은 대부분이 반복적인 다중 루프 형태의 작업이므로 이를 하드웨어화 하기에 적당하다. 따라서 세선화 과정이 고속의 하드웨어로 구현된다면 인식 속도는 적어도 3배 이상 개선되리라 생각된다.

그외에도 한글 문자들이 같이 있는 기호와 숫자에 대해서도 이들을 인식할 수 있는 연구가 계속 요구된다.

4. 결 론

본 연구는 최근 많은 연구가 활발히 진행되고 있는 문자 인식 분야 중 인쇄체 한글 인식에 관한 것이다. 한글은 생성 가능한 문자 수가 방대하고 비교적 적은 수의 기본 자소들이 서로 결합하여 하나의 문자를 형성하기 때문에 문자의 유사성이 커서 이를 인식하는 데 많은 어려움이 있는 실정이다.

본 연구에서는 문자의 특징을 그대로 유지하면서 자모 분리와 인식에 필요한 연산량을 줄이기 위해 입력 영상에서 문자의 골격만을 추출하는 세선화 과정을 거치고 세선화된 한글 문자에서 붙어 있는 자모를 분리함으로써 인식 대상을 기본 24개의 자소로 한정하고 이를 개별적으로 인식한 후 결합하는 일반적인 방법을 택하였다. 그러나 자모 분리단계에서 모음에 존재하는 분기점들을 이용하여 자모 분리와 동시에 모음을 인식하게 하였고, 자음 인식에 있어서도 자음의 초성, 종성에 따른 변화와 자형, 크기에 별로 구애 받지 않는 방법을 취하였다.

본 인식 알고리즘의 인식율이나 인식 속도는 아주 좋다고는 할 수 없다. 그러나 인식 시간의 70% 이상을 세선화 과정이 차지함을 고려해 볼 때, 세선화 과정의 하드웨어화는 현재의 초당 2~3자의 인식 속도를 크게 개선시키리라 믿는다. 그리고 오 인식의 가장 큰 요인인 자모 분리과정에서의 실패에 대한 연구를 계속함으로써 인식율도 개선시킬 수 있을 것이다.

참고문헌

[1] Chen, Y. S. and Hsu, W. H., "A modified fast parallel algorithm for thinning digital patterns," Pattern Recognition Vol. 7, pp.99-106, 1988.

[2] Zhang, T. Y. and Suen, C. Y., "A fast parallel algorithm for thinning digital patterns," Commun. ACM, Vol. 27, pp.236-239, 1984.

[3] 김명원, 이영직, 이춘복, "신경회로망을 이용한 특징추출," 한국전자통신연구소, '89 신경회로망 응용 Workshop, pp.39-49, 1989.

[4] 이광호, "다중 활자체 한글 인식을 위한 자모의 분리," 한국과학기술원, 전산학과, MCS-87 276, 1989.

[5] 이승호, 김진영, "한글의 구조적 인식을 위한 자획 추출에 관한 연구," 한국 정보 과학회 논문집, 1987.

[6] 이주근, "한글 문자 인식에 관한 연구," 전자공학회지, 제 9 권, 제 4 호, pp. 25-32, 1972.

[7] 이주근, 남궁재찬, 김영건, "한글 pattern에서 Subpattern 분리와 인식에 관한 연구," 전자공학회지, 제18권, 제 3 호, pp. 1-8, 1981.

[8] 박현철, "한글 코드 체계 그 알파와 오메가," 마이크로소프트웨어, 3월호, 1989.

[9] 한글 기계화 연구, 한글 기계화 연구소, 1975.