[5-7] # On-Line 문자인식을 위한 Stroke 분류에 관한 연구


전병환 박충식 김재희
(연세대학교 전자공학과)


# A Stroke Classification for the On-line Character Recognition


Byung Hwan Jun Choong Shik Park Jaihie Kim
(Dept. of Electronic Eng. Yonsei University)



ABSTRACT

This paper suggests an algorithm for the automatic classification of the strokes by extracting directional components in the strokes composed of significant points after preprocessing. A directional change, a directional code connecting the starting point and the end point, a final directional code, a first directional code, and a rotation of the stroke are extracted as the candidate features. And in the proposed stroke classification algorithm, the priority of these features is computed and the number and order of features to be applied is optimally determined for the classification of the given input patterns. The ratio of the correct recognition for the proposed stroke recognition system is 96.0 %.


## I. 서론

최근 고도 정보화시대가 도래하면서 자연언어처리에 관한 연구가 각 방면으로 진전되고 있다. 이의 한 부류인 온라인 문자인식은 간편한 문자입력수단의 하나로써 그 필요성이 높아지고 있다. 온라인 문자인식에 의한 입력방식은 특별한 숙련기술을 필요로 하지 않고 문자를 직접 쓴다는 자연스러운 동작에 의하여 자료를 입력하며 문서작성 도중에 오인식의 발견과 수정을 즉석에서 행할 수 있다. 또한, 기존의 펜에 의한 문서 작성후 타이핑(typing)이라는 두 단계 방식에서 벗어나 펜에 의해 바로 입력이 완료됨으로써 사실상의 인력과 시간절약을 기할 수 있다. 컴퓨터와 전자 타블렛(electronic tablet)의 발전 그리고 알고리듬의 개발에 따라 온라인 필기체 인식 장치가 제품화되기 시작했으며 국내에서도 연구가 활발히 진행되고 있다[1][2].

이 논문은 같은 논문집에 실린 구본석, 김성훈, 그리고 김재희의 "On-Line 문자인식에서의 Preprocessing 알고리듬"과 더불어 온라인 필기체 문자인식을 위한 연구의 일환이다.

입력된 정보는 획(stroke)단위로 처리하는데 펜다운(pen-down)에서 펜업(pen-up)까지를 하나의 획으로 정의한다[1]. 전처리(preprocessing)에서는 노이즈(noise)나 인식시 불필요한 정보를 줄이기위한 거리필터링(ditance filtering), 각 필터링(angular filtering), 그리고 디후킹(dehooking)등이 쓰이는데[3][4], 이에 관한 상세한 내용은 위의 논문을 참조하기 바란다. 전처리를 거친 획은 몇개의 중요한 특징점들로 구성되며, 이 점들로부터 획의 방향특성을 추출하는 특징(feature)들에는 획의 방향변화, 시작점에서 끝점을 잇는 방향코드, 마지막 방향코드, 처음 방향코드, 그리고 획의 회전여부 등이 있다. 이 논문에서는 이들 특징들의 우선순위(priority)를 계산하여 사용할 특징들과 적용순서를 결정함으로써 실제 주어진 입력패턴들을 가장 잘 분류하는 자동 획 분류 알고리듬을 제안하고 또한, 사람이 작성한 획 분류 오토마타와 비교 보완한 후 획 인식 시스템을 구현하였다. 향후 개발될 온라인 필기체 문자인식 시스템에서도 획의 순서뿐만아니라 아래의 예와 같이 현재 획의 끝점에서 다음 획의 시작점을 잇는 방향등의 방향성 특징들이 주로 사용된다.

[그림 1] 문자 인식을 위한 획(stroke)간의 방향성 특징의 예

획 인식 시스템(stroke recognition system)의 구현에 대한 본 논문의 내용은 다음과 같다.

o 획 분류를 위한 특징(feature)들
o 특징(feature)선택을 위한 우선순위(priority)결정 규칙
o 자동 획 분류 알고리듬

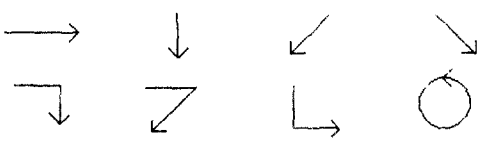
## II. 획 인식 시스템 (stroke recognition system)

본 논문에서는 문자를 크게 한글문자와 영문자(대문자), 숫자, 그리고 특수문자의 두 부류로 나누어 인식하는 것을 원칙으로 한다.

한글은 조합문자이기 때문에 문자수가 극히 많음에도 불구하고 문자를 구성하는 기본 자소의 수는 24자에 불과하고 더우기 자소를 구성하는 획은 8개에 지나지 않는다. 반면, 영문자나 숫자 그리고 특수문자 등은 4개이내의 획이 하나의 문자를 형성하는 특성 때문에 영어 대문자 26자, 숫자 10자, 그리고 특수문자 11자의 총 47개 문자에 불과하나 이를 구성하는 획의 수는 총 32개에 이른다. 따라서 서로 다른 특성을 갖는 두 문자그룹을 동시에 인식하기보다는 나누어 인식하는 것이 사용특징(feature)의 단순화와 적용할 특징의 감소를 가져오며 처리시간도 효과적으로 줄일 수 있다.

<획의 종류>

o 한글문자를 이루는 획의 종류 (8개)

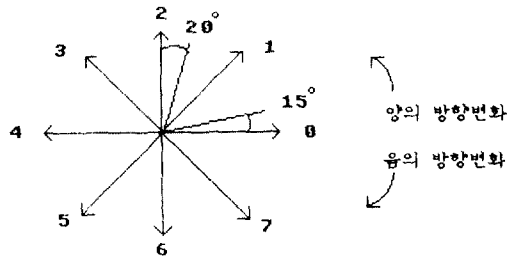


o 영문자, 숫자, 특수기호를 이루는 획의 종류(32개)의 몇가지 예

### (1) 획 분류를 위한 특징(feature)들

이 논문에서 획 인식을 위해 사용한 방식은 획이 지닌 특징들의 집합으로 특정 획을 나타내는 '특징 분석(feature analysis)'방식에 해당한다. 획의 분류를 위해 사용할 수 있는 특징에는 여러가지가 있을 수 있지만 획 부위 간의 상대적인 크기차이나 문자전체 크기에 거의 영향을 받지 않으면서 획의 형태에 대한 정보를 유지할 수 있는 방향성 특징들을 주로 사용함으로써 알고리듬의 단순화와 처리시간의 감소를 이루도록 한다.

방향코드는 8방향의 코드로 하되 수직 수평에 대해 균일하지 않은 각을 할당함으로써 일반적인 필기형태에 적합하도록 하는데, 이 차이는 필기형태마다 달라지지만 타블렛상에서 쓸때 수평선보다는 수직선을 곧게 쓰기 어렵다는 점에 주목하여 수평성분에 30°, 수직성분엔 40°를 배당한다. 또한, 방향부호가 증가하는 방향변화를 양의 방향변화, 방향부호가 감소하는 방향변화를 음의 방향변화로 정의한다.




[그림 2] 방향코드와 방향변화

--- 특징(feature)들 ---

1) 획의 방향변화

예를 들어, 그림 3의 ①은 방향변화가 없는 경우이고, ②는 음의 방향변화만 있는 경우이며, ③은 양의 방향변화였다가 다시 음의 방향변화로 바뀐 경우이다. 이 논문에서는 이와 같은 방식으로 총 7개의 서로 다른 그룹으로 나누었다.

2) 획의 시작점에서 끝점을 잇는 방향코드

그림 3의 ①은 0의 코드값이 할당되고, ③은 5의 코드값이 할당되 며, 시작점과 끝점이 일정거리보다 작은 ④와 같은 경우는 방향코드대신 근접을 뜻하는 값 8을 할당한다. 그러나 제한거리를 크게하면 두 점으로 이루어지는 짧은 직선획까지 근접으로 처리하게 되므로, 세점 이상의 획에 대해서만 근접을 인정하고 두점인 경우는 그대로 방향코드를 유지하도록 한다.

3) 획의 마지막 방향코드

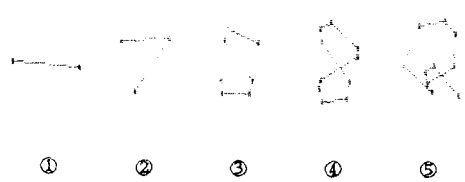
그림 3의 ②는 마지막 방향코드가 5이다.

4) 획의 처음 방향코드

그림 3의 ②는 처음 방향코드가 0이다.

5) 획의 회전여부

획의 누적된 마지막 방향값과 시작방향부호와의 차이가 8이상이면 회전을 인정한다. 그림 3에서는 ⑤만이 회전이 있고 나머지는 회전이 없다.




[그림 3] 전처리(preprocessing)를 거친 입력 패턴의 예

이상의 5가지 특징은 주로 방향정보를 처리하는데, 그 중 1)번 특징은 방향만으로 획의 전체 형태 정보를 유지하는 가장 중요한 특징이다. 따라서, 이 논문에서는 1)번 특징을 반드시 먼저 사용하고 그 다음 4개의 특징을 선별하여 정하도록 한다. 또한 알고리듬상 나머지 4개의 특징은 1)번 모듈(module)에 첨부할 수 있으므로 별도의 모듈보다는 하나로 모듈화하는 것이 알고리듬의 간소화뿐만 아니라 처리 시간면에서도 바람직하다.

(2) 특징(feature)선택을 위한 우선순위(priority)결정 규칙

주어진 획들을 획코드(stroke code)별로 분류할 수 있는 획 분류기(stroke classifier)를 설계하는 일은 중요하면서도 까다로운 작업이므로 사람이 경험적(heuristic)으로 일일이 사용할 특징들을 결정하는 것은 번거롭고 비합리적이다. 이를 보완하기 위해 이 논문에서는, 입력데이타에서 추출된 특정 패턴의 집합이 같은 부류로 판정될 수 있도록 하는데 필요한 특징을 최적으로 결정함으로써 자동으로 획 분류 알고리듬을 생성하는 알고리듬에 대해 다루고 있다.

매 단계마다 특징의 선택을 위해 우선순위(priority)를 구하는데, 그 규칙은 다음과 같으며 우선순위가 가장 큰 것이 선택된다.

$$Pi = ( Base - Pi1 + Pi2 ) \times TRi$$

여기서,

Pi는 feature i의 전체 priority값

Base는 Pi값이 항상 양수가 되도록 정해진 상수

Pi1는 feature i로 분류했을 때 생기는 각 그룹마다 추가로 혼합되어있는 pattern class의 수의 합

Pi2는 어떠한 그룹에도 혼합되지않고 완전히 구분된 pattern class의 수

TRi는 feature i의 처리시간에 대한 효율 (0 ~ 1 사이의 값)

(3) 자동 획 분류 알고리듬

매 단계마다 위의 우선순위(priority) 규칙에 의해 결정된 특징에 의해 주어진 데이타를 특징값(feature value)에 따라 분류한다. 특징값에 따라 구분된 그룹들은 사용되지 않는 경우와 분류가 완료된 경우 그리고 아직 혼합되어 있는 경우로 나누어지는데, 분류가 완료된 그룹은 하나의 획코드만이 들어있으므로 더 이상의 특징을 사용할 필요가 없이 인식이 완료된다. 이 경우 그 획코드에 해당하는 모든 패턴의 분류가 이루어졌음을 뜻하는 것은 아니다. 분류가 완료되지않아 여러 종류의 획코드가 섞여있는 그룹에 대해서는 다음 단계에서 이미 사용한 특징들을 제외한 나머지 특징들 중에서 우선순위를 계산하여 마찬가지 방식으로 분류해 나간다. 특징이 부족하거나 부적합한 경우에 획 분류가 완전히 이루어지지 못할 수가 있으나, 이 논문에서는 앞에서 정한 방향성 특징들 만으로 분류가 이루어졌다.

이러한 획 분류 알고리듬의 자동생성은 복잡한 분류기의 설계가 컴퓨터의 도움으로 손쉬워지며, 실질적인 데이타로 분류 알고리듬을 생성하므로 입력패턴의 속성이 충분히 고려된 인식시스템의 설계가 가능한 장점이 있다. 그러나 입력데이타로 쓰이는 획들이 허용가능한 형태까지 충분하지 않거나 비합리적인 데이타가 포함될 경우 획 분류 알고리듬은 최적의 상태가 되지 못하고 오인식이나 미인식 그리고 인식에 불필요한 제한요소를 지니게 된다. 따라서 본 논문에서는 4명의 필기자로 부터 필요한 획 데이타를 한글인 경우는 자소단위로 영문자, 숫자, 특수문자인 경우는 문자단위로 입력하게 하여 획 분류 알고리듬을 얻은 다음, 데이타의 부족으로 생기는 일부 미비한 부분을 보안하여 획 인식 시스템이 제대로 구성되도록 하였다.

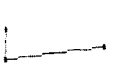
## III. 실험 및 결과 고찰

입력장치는 240(samples/sec)의 sampling rate, 510(points/inch) 정도의 resolution을 갖는 전자기(electromagnetic)방식의 타블렛 디지타이저(tablet digitizer)를 사용한다.

문자정보는 타블렛(tablet)상의 펜끝(pen-tip)의 움직임에 따른 x, y좌표값으로 입력되며, 스크린(screen)상의 분할된 해당 위치인 x, y좌표값으로 변환된다. 한글인 경우는 64x64 화소로 분할되고 영문자(대문자), 숫자 그리고 특수기호는 48x48 화소로 분할된다. 입력된 정보는 전처리에서 노이즈성 데이타와 불필요한 데이타를 제거한 다음 구현된 획인식 시스템에서 해당 획코드를 출력하게 하였다. 실험을 위해 4명으로부터 한글은 자소단위로, 영문자, 숫자 그리고 특수문자는 문자단위로 입력을 받아 각 획에 대한 인식을 실시했다.

자동으로 획 분류기를 생성함으로써 최적의 단계로 모든 분류를 이룰 수 있었다. 즉 모든 특징을 다 적용하여 완전한 분류를 할 경우 각 획 인식마다 5단계의 처리가 필요하나 본 논문의 경우 평균 3.2단계로써 획 인식이 이루어졌다. 특히, 한글의 경우 주로 방향 특징이 확실히 차이가 나는 직선성의 획으로 구성되기 때문에 2~3개의 특징만으로 충분히 구분이 가능하며 이와 같이 획 인식에서 절약한 시간은 획의 순서와 위치관계를 사용하는 문자인식에서 유용하게 활용할 수 있다.

또한, 제안한 자동 획 분류 알고리듬은 실질적인 데이타를 취급함으로써 사람이 설계할 때 미처 생각하지 못했던 요인들까지 손쉽게 처리하는 특징을 갖는다. 한 예로 그림 4와 같이 'ㄹ'의 마지막 획으로 사용되는 'L'의 경우 시작점과 끝점을 잇는 방향코드가 7이 아닌 0이 될 수도 있다.



[그림 4] 사람이 고려하지 못한 획인식의 예

이 논문에서 사용한 방향성 특징(feature)들은 직선성 획의 인식에는 적합하나 원이나 곡선인 경우에는 획의 형태 정보를 충분히 유지하지 못하는 특성을 갖는다. 특히, 방향 변화 특징과 시작점과 끝점의 근접만으로 인식하는 '0'나 '8'의 경우 인식률이 다소 떨어지는데 이의 보완이 필요하며, 회전특징(circulation feature)의 경우엔 완전한 회전이 일어나지 않거나 방향변화가 중간에 바뀌게 되면 회전이 없는 것으로 처리하므로 매우 제한된 경우에만 사용되는 단점을 갖는다. 획의 허용 범위가 넓어지면 현재의 특징들만으로 인식이 불가능한 경우가 발생할 수 있는데 이런 경우 현 특징의 보완이나 새로운 특징의 추가가 필요하다.

본 시스템의 획 인식률은 표 1과 같다.

| | | |
|---|---|---|
| 정인식 | 499 획 | 96.0 % |
| 오인식 | 8 획 | 1.5 % |
| 미인식 | 13 획 | 2.5 % |
| 총 계 | 520 획 | 100 % |

[표 1] 본 시스템의 획 인식률

## IV. 결론

이 논문에서는 온라인 문자인식의 기본이되는 획 인식기의 구현을 위한 자동 획 분류 알고리듬을 개발하였다.

타블렛(tablet)과 스타일러스(stylus)를 통해 들어온 문자데이타를 전처리에서 노이즈와 불필요한 정보를 제거하여 인식부에서 처리해야할 정보량을 줄였으며, 사용할 특징과 그 적용순서를 자동으로 정하는 획 분류 알고리듬에 의해 구현한 획 인식기에서 해당하는 획코드를 출력하게 하였다. 자동 획 분류 알고리듬은 사람이 구현하기 어려운 최적의 획인식 오토마타를 손쉽게 컴퓨터가 찾아내도록 하며, 실질적인 데이타를 사용하여 구현함으로써 입력패턴의 속성을 충분히 고려할 수 있게 하는 장점을 갖는다. 또한, 구현된 획인식기는 궁극적으로 온라인 필기체 문자인식을 위한 연구이며 앞으로의 문자인식이 주로 획의 순서에 의해 이루어지므로 획인식은 사실상 문자인식의 핵심이다.

여기에서 발표되는 연구는, 본 연구팀이 추진중인 인공지능 워드 프로세서나 팬 컴퓨터에서의 한글, 영문자, 숫자, 특수문자, 그리고 문서편집부호(gesture) 등의 인식을 위한 시스템 개발의 일부이다.

## V. 참고 문헌

[1] 이의동, 김태균, "補强文脈自由文法을 利用한 筆記體 한글 온라인 認識," 전자공학회논문지, 제 24 권, 제 5 호, pp. 37-44, 1987, 9.

[2] 정복만, 권오석, 김태균, "온라인 입력 한글의 적응학습과 인식에 관한 연구," 한국정보과학회 논문지, 제 16 권, 제 5 호, 1989, 9.

[3] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The State of the Art in On-Line Handwriting Recognition, IEEE Trans. Pattern Anal. Machine Intell., vol. 12, No. 8, pp. 787-808, Aug. 1990.

[4] C. C. Tappert, "Speed, Accuracy, Flexibility Trade-Offs in On-Line Character Recognition," IBM Res. Rep. RC13228, Oct. 1987.