# 획의 방향 코드 조합에 의한 오프라인 필기체 숫자 인식

## (Off-line Handwritten Digit Recognition by Combining Direction Codes of Strokes)

이 찬 희 † 정 순 호 ††
(Chan-Hee Lee) (Soon-Ho Jung)

**요 약** 본 논문은 한 가지 특징 요소로서 획 방향 코드들만을 사용하는 강건한 오프라인 필기체 숫자 인식 방법을 제안한다. 이 방법은 입력된 숫자 이미지에 대하여 일반적인 8방향 코드를 생성하고 이 코드들의 조합을 다층 신경망에 학습하고 각 숫자를 인식하게 한다. 8방향 코드들은 다양하게 표현된 숫자들의 자기구성 그래프(SOG*:Improved Self-Organizing Graph) 세선화 결과에 의해 만들어지고 이 코드의 사용은 2개 이상의 특징점들을 처리하는 기존의 복합적인 단계들을 단순화 시킨다. 실험결과는 모든 숫자 데이타베이스의 어떤 이미지들에 대해서도 인식률이 일관성 있게 98.85% 이상임을 보여준다.

**키워드** : 숫자 인식, 획 방향 코드, 세선화, SOG*

***Abstract*** We present a robust off-line method recognizing handwritten digits by only using stroke direction codes as a feature of handwritten digits. This method makes general 8-direction codes for an input digit and then has the multi-layered neural networks learn them and recognize each digit. The 8-direction codes are made of the thinned results of each digit through SOG*(Improved Self-Organizing Graph). And the usage of these codes simplifies the complex steps processing at least two features of the existing methods. The experimental result shows that the recognition rates of this method are constantly better than 98.85% for any images in all digit databases.

**Key words** : digit recognition, stroke direction code, thinning, SOG*

## 1. 서 론

오프라인 필기체 문자 인식의 기본이 되는 오프라인 필기체 숫자 인식은 우편 봉투의 자동 분류, 전표나 민원 서류와 같은 각종 서식의 입력 자동화 등 다양한 분야에서 그 응용이 이루어지고 있다[1,2].

이러한 필기체 숫자의 오프라인 인식에는 인식 성능 향상을 위해 두 가지 측면을 고려할 수 있다. 첫 번째는 특징을 추출할 때 인식에 필요한 중요 정보의 손실을 최소화하는 것이고, 두 번째는 특정 패턴에 국한되지 않는 일반화 및 객관적 적응 능력을 가지고 인식률을 극대화 할 수 있는 분류기의 적용이다[1-7].

위의 두 가지 측면 중에서 전자에서 이루어지는 처리 단계를 전처리(Preprocessor)라고 한다. 이 단계에서는 인식 성능의 향상을 위하여 패턴의 형태를 다양하게 변형하는 방법이 사용되며 이 변형 방법은 선형 형태 변형(Linear Shape Variation)과 비선형 형태 변형(Non-linear Shape Variation)으로 크게 분류할 수 있다. 선형 형태 변형의 방법으로는 이동, 신축, 회전, 밀림 등의 정규화 과정이 포함되며, 비선형 형태 변형 방법으로는 점밀도, 선밀도, 획밀도 등의 정규화가 포함된다[3-5].

전처리에서 이와 같이 다양한 변형 방법을 사용하고 있으며 그 방법들과 달리 신경회로망이 갖는 적응적 특성을 이용한 필기체 숫자 인식에 관한 연구가 활발히 진행되고 있다[3]. 그러나 현재까지 신경회로망을 이용한 방식들은 특징 추출의 결함으로 인해 다양한 형태의 왜곡을 흡수하지 못하는 문제점이 있거나 인식률을 높이기 위하여 전처리 과정에서 많은 형태의 변형들을 사용함으로써 전처리 과정이 복잡해지며, 인식기의 입력도 많아지게 된다.

따라서 본 논문에서는 다양하고 많은 필기체 숫자 패

† 학생회원 : 부경대학교 전자계산학과
chlee@aisol.pknu.ac.kr
†† 정 회 원 : 부경대학교 컴퓨터멀티미디어공학부 교수
snow@pknu.ac.kr
논문접수 : 2002년 10월 22일
심사완료 : 2004년 10월 21일

턴들을 SOG*를 이용하여 세선화한 뒤, 분석하여 일반적인 특징으로 획 방향코드를 생성하고 이들 방향코드들의 조합으로 숫자를 인식하는 시스템을 제안하고자 한다.

이 시스템은 전처리 과정에서 세선화를 수행하여 나온 이미지에서 해당되는 숫자 특성의 방향 코드를 추출함으로써 전처리 과정을 단순화 하였다. 또한 이렇게 생성된 방향 코드만으로도 각 획을 특징지을 수가 있으며 단지 방향 코드만이 인식의 입력으로 사용되어 인식기의 입력을 줄일 수 있다. 이 인식시스템은 은닉층을 갖는 3층 퍼셉트론으로 구성하였다.

본 논문은 모두 6장으로 구성되어 있으며 그 내용은 다음과 같다. 2장에서는 전처리의 SOG* 세선화 기법을 기술하고, 3장에서는 특징 추출단계에 대한 구현으로 세선화의 결과로 얻어진 숫자 이미지로부터 특징 점과 획의 분리, 방향 코드를 추출하는 방법을 소개한다. 4장에서는 숫자 인식을 위한 다층 신경회로망의 구성과 방법을 소개하며 5장에서는 본 논문의 실험 및 실험 결과를 분석하고 마지막 6장에서는 결론을 언급한다.

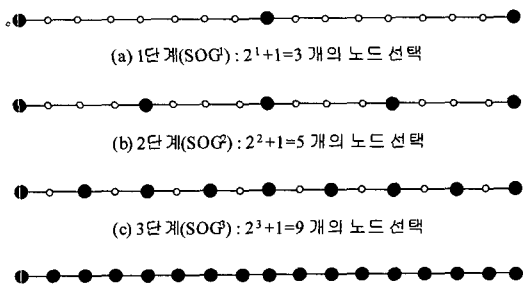
## 2. SOG* 세선화

SOG* 세선화 기법은 점증적 학습 알고리즘[8]과 SOG[9,10]의 변형 방법을 사용한 SOG 기반의 고속 세선화 알고리즘을 사용한다[11]. 여기서 점증적 학습 알고리즘이란 세선화를 위한 학습 과정에서 경쟁층의 대상이며 학습의 대상이 되는 뉴런들의 개수를 단계별로 증가 시켜가며 학습시키는 방법을 말한다. 또한 SOG의 변형 방법이란 기존의 SOG의 학습 과정에서 뉴런들이 각 입력 패턴들을 대상으로 지정된 학습 시간동안을 항상 소모하게 하는 대신에 각 패턴들의 학습 결과의 안정화되는 시점을 파악하여 더 이상의 소모적인 학습 시간을 제거하므로 고속 학습을 유도할 수 있다[11].

SOG*에서의 세선화 단계는 다음과 같이 이루어진다. 숫자 입력 영상은 숫자 이미지 위의 픽셀들이 이차원의 자료로서 좌표 (x,y)를 가지게 되며 이 좌표들이 최대 17개의 뉴런들의 입력으로 사용되어 일차원적(선형적)으로 학습이 이루어진다. 이 결과, 뉴런들이 가지고 있는 가중치 내용들이 성공적인 세선화 형태를 나타내게 된다. 여기서 선형적이란 학습되는 뉴런들 사이의 이웃 관계를 나타낸 것이다. 이 학습 과정에서 학습의 속도를 빠르게 하기 위하여 뉴런들을 점증적으로 추가하는 과정을 그림 1에서 보이고 있다.

첫 번째 단계인 $SOG^1$(그림 1(a))에서는 제일 끝의 두 노드와 그들 중심의 노드(그림 1의 검은색 노드들)가 선택되어 학습을 수행하며, 두 번째 이후 단계부터는 이전 단계에서 선택되어진 n개의 노드들 사이에서 선형 보간법을 이용하여 n-1개의 새로운 노드들을 선택하게 된다. 그림 1에서 검은색으로 표시된 뉴런들은 현 단계에서 학습에 참여하는 노드들이며 흰색으로 표시된 뉴런은 현 단계에서 학습에 참여하지 않는 노드들이다.

처음 1단계부터 $\log_2(M-1)$단계까지 각 단계별로 학습이 참여하는 뉴런의 수를 $2^{level}+1$개로 결정하여 세선화를 수행하게 된다. 여기서 M은 맵의 크기를, level은 단계를 의미한다. 따라서 단계별로 학습에 참여하는 뉴런들의 수는 3, 5, 9, 17, 33, 65,...로 증가하게 된다. 이 중에서 경쟁층 뉴런들의 최대 수를 17개로 정하는 것은 이 연구에서 실험되는 숫자 이미지의 크기를 고려하고 학습 시간과 세선화 결과를 살펴볼 때 가장 적절한 수임을 직관적으로 알 수 있으며 위 단계에 대한 실험에서도 가장 적당한 뉴런들의 개수임을 보인다.



(a) 1단계($SOG^1$) : $2^1+1=3$ 개의 노드 선택

(b) 2단계($SOG^2$) : $2^2+1=5$ 개의 노드 선택

(c) 3단계($SOG^3$) : $2^3+1=9$ 개의 노드 선택

(d) 4단계($SOG^4$) : $2^4+1=$ 전체(17개) 노드 선택

그림 1 SOG* 세선화 과정

다음 그림 2에서는 SOG* 세선화를 적용한 결과를 보여주고 있으며 각 세선화 결과 이미지는 원 이미지의 특성을 유지하는 우수한 세선화 결과를 나타낸다. 또한 기존의 템플릿 윈도우를 이용한 기법[12]과 ART2 신경망을 이용한 클러스터링 기법[13], 그리고 SOG 기법에 비해 빠르며 이 알고리즘의 시간 복잡도는 $O((\log M)^3)$이다[11].

(a) 원 이미지 (b) SOG* 세선화 결과

그림 2 SOG* 세선화 결과

## 3. 특징 추출

필기체 숫자는 필기자의 필기 습관이나 필기도구 등에 의해 많은 변형을 가지게 된다. 특히 오프라인 필기체 숫자 인식의 경우에는 앞의 변형들과 함께 스캐너와

같은 입력 장치에서 발생하는 잡음 등도 고려해야 한다.

이로 인해 기존 논문들의 경우에는 전처리 과정에서 하나의 특징만이 아닌 복합 특징을 이용한 단일 신경회로망 또는 다중 구조의 신경회로망을 이용하여 인식하였다[1-3,14-18]. 그러나 이러한 방법들은 전처리 단계에서의 많은 시간 소비와 인식기가 커지는 문제점을 가지게 된다. 따라서 이러한 전처리의 복잡성을 단순화시키는 방법으로 다음의 과정으로 처리한다. 먼저 세선화된 결과를 이용하여 특징 점을 추출하고, 획을 분리하여 각 획들의 방향 코드 특징을 추출한다. 이 추출된 특징들을 인식에 사용함으로서 전처리 단계가 단순하게 되어 처리 속도의 향상, 인식기의 축소, 그리고 인식률의 향상을 이루도록 하였다.

### 3.1 특징 점의 추출과 획의 분리

획이란 숫자를 특징짓는 요소로서 각 숫자를 표현하기 위하여 사용되는 기본적인 직선이나 곡선의 조합으로 정의한다. 예를 들어 숫자 7은 세로 직선 획 두 개와 가로 직선 획 하나로 이루어져 있다. 세선화 된 이미지에서 어느 한 점으로부터 인접된 여러 개의 획이 존재하며 그 숫자의 특성을 잘 나타내는 그러한 점들을 선택하여 특징 점으로 정의한다.

이러한 특징 점과 인접한 획들의 분리는 동시에 이루어지며, 각 과정은 그림 3과 같이 앞의 단계로부터 세선화 결과로 이산된 17개 점들에 대한 2차원 정보를 입력으로 받아 수행한다. 각 단계는 (1) 시작점을 결정하고, (2) 점의 정렬과 획의 분할 과정을 거쳐서 (3) 8방향 코드를 생성하기 위한 획을 병합하는 과정을 거친다. 이 단계의 처리 과정을 기술하면 다음과 같다.

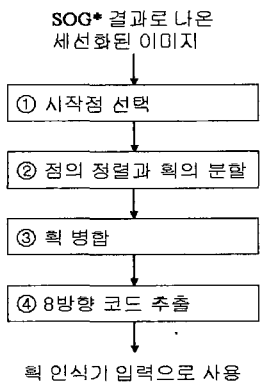



그림 3 특징 점과 획의 분리 과정

첫 번째 시작점 선택과정(그림 3 ①)은 세선화의 결과로 나온 이미지에서 시작점을 선택하는 과정이다. 시작점은 그림 4와 같이 이미지의 입력된 점들에서 최소가 되는 점으로부터 x, y축을 설정하고 그 x축과 y축이 교차하는 점(원점)으로부터 가장 가까운 점을 시작점으로 선택하게 된다.

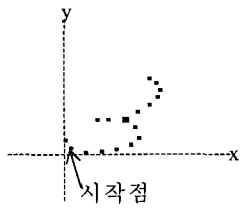



그림 4 시작점의 선택

두 번째로 점의 정렬과 획의 분할 과정(그림 3 ②)은 시작점을 처음으로 이산되어 있는 점들을 연결해서 획을 추적하기 위하여 나머지 점들을 정렬하게 된다. 정렬을 수행하는 과정은 초기에 시작점이 현재 점이 된다. 이 현재 점을 기준으로 가장 가까운 3개의 후보 점을 선택하여 획으로 이어질 수 있도록 선택한다. 점을 선택하는 기준은 현재 점에서 다음 점까지의 최소 거리, 기울기, 원 이미지에서 픽셀의 존재 여부 등의 세 가지 기준을 사용한다. 선택된 현재 점이 시작점이라면 선택된 각 점들과 현재 점 사이에 원래의 이미지에서 글자 부분의 픽셀이 존재하는지의 여부와 점과 점 사이의 거리를 확인하여 다음 점을 선택한다. 둘 이상의 점이 선택 가능하다면 가장 가까운 점을 선택하게 된다. 이때 원 이미지를 검색하는 이유는 실제 획이 아님에도 세선화된 결과에서 거리가 가까운 이유로 선택되는 경우를 방지하기 위해서이다. 반면에 현재 점이 시작점이 아니라면 시작점과 동일한 방법을 거치되 이전 점과의 기울기까지를 고려하여 인접 점을 선택하게 된다. 이때 현재 점에서 인접 점의 개수가 둘 이상이 가능하다면 여러 개의 획이 분산되어 존재함을 의미하므로 바로 이 점이 특징 점이 된다. 또한 점과 점 사이의 거리가 실험적인 임계치 이상이고, 원 이미지 상에 글자 부분의 픽셀이 존재하지 않는다면 획을 분리한다. 그림 5는 각 숫자 이미지에서 선택된 특징 점을 보여주고 있는데 숫자 0, 1, 7등은 일반적으로 특징 점이 존재하지 않는다. 0, 1, 7 등의 숫자는 시작점에서 출발하여 각 점들을 정렬하여 이어가는 과정에서 두 개 이상으로 분리되는 경우가 일반적으로 발생하지 않기 때문이다.

그림 6은 정렬 과정과 특징 점을 추출하는 과정을 보여 주고 있는데 특징 점이라고 표시된 부분을 보면 선택 될 수 있는 점이 원으로 표시한 곳에 포함되는 3개의 점 모두가 가능하다. 원 이미지에서 픽셀이 모두 존재하며 거리도 모두 임계치를 벗어나지 않으나, 이전 점과의 기울기를 고려한다면 우측의 두 개 점이 가능하다. 우측 두 개 점이 3개의 조건인 원 이미지의 픽셀 존재

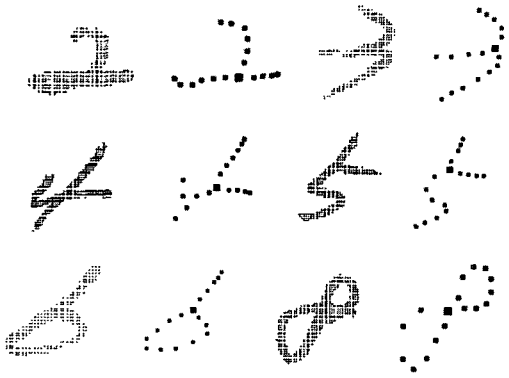



그림 5 원 이미지와 그에 대한 특징점 추출

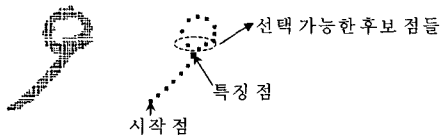



그림 6 정렬과 특징 점 추출

여부, 거리, 기울기를 만족하므로 이때는 가까운 점을 선택하여 이전의 과정을 반복하게 된다.

이러한 과정을 반복하게 되면 그림 6의 세선화 이미지는 그림 7과 같이 하나의 연결된 획으로 구성할 수 있다. 그림 6과 같은 경우에는 특징 점을 기준으로 두 개의 획으로 분리하여 각 획을 구성할 수 있으나 모든 숫자가 하나의 획으로 이어서 표현할 수 있으므로 강제로 분리하지 않고 그림 9와 같이 하나의 이어진 획으로 생성하기 힘든 경우에만 분리하도록 한다.


그림 7 하나의 획으로 구성된 경우

세 번째 획의 병합 과정(그림 3 ③)은 잘못 분할된 획들을 하나의 획으로 병합하기 위한 것이다. 그림 8과 같은 예에서 보듯이 시작점이 끝점이 아닌 경우에 하나로 이어져야 하는 획이 불필요하게 분리된 경우가 발생한다. 그림 8에서 번호는 정렬 순서를 의미하며 1로 표시된 점이 시작점이다. 획을 병합하는 방법은 각 획의 처음과 끝점을 각각 세 가지 조건을 만족하는지의 여부로 병합을 결정한다.

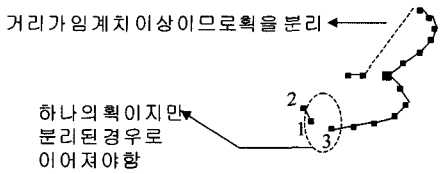



그림 8 획 분리와 특징 점 추출 후 획의 분포

그림 9에서는 획 분리와 특징 점을 추출한 후의 획의 분포를 알아보기 쉽도록 선으로 연결하여 표현하였다. 그림 8에서는 3이라는 숫자가 모두 3개의 획으로 구성되어 있음을 알 수 있는데 각 획들의 시작점과 끝점 각각의 관계를 조사하면 원으로 표시된 부분이 거리와 기울기 그리고 원 이미지의 픽셀 존재 모두를 만족하므로 하나의 획으로 이어지는 것이 바람직하다는 것을 알 수 있다. 획을 병합한 후의 이미지는 그림 9와 같이 짧은 획의 순서는 반대로 바뀌게 되어 15개의 점이 하나의 획으로 구성이 된다.

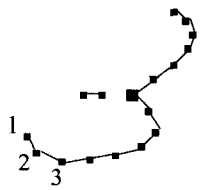



그림 9 병합한 후의 획 분포

### 3.2 방향 코드의 추출

3.1의 모든 과정을 거친 후 마지막으로 8방향 코드 추출 단계(그림 3 ④)에서는 세선화된 이미지에 대응하는 방향 코드를 8개의 절대 방향을 이용하여 생성한다. 8방향 코드는 그림 10과 같은 8개의 절대 방향을 의미하고[16,20], 이 코드는 방향 코드로써 각 획과 숫자들이 지니고 있는 특징들을 표현 할 수 있으며 각 획들의 점들이 모두 이어짐을 가정하여 점들간의 방향 성분을 추출할 수 있다.

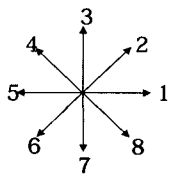



그림 10 8방향 코드

그림 9의 숫자 3을 예로 들어 각 방향 코드를 추출하면 두 개의 획 각각의 방향 코드는 획1(8 8 1 1 1 2 3 4 2 2 2 3 3 4)과 획2(5)로 추출이 되는데 동일한 코드

들이 반복적으로 나오는 것을 알 수 있다. 동일한 코드가 반복적으로 나오는 것은 동일 방향으로 계속 이어짐을 의미하므로 동일 코드들을 하나의 코드로 축소가 가능하게 되어 최종 추출되는 방향 코드는 획1(8 1 2 3 4 2 3 4), 획2(5)가 생성된다. 반복되는 코드들을 제거하여 코드를 생성한 결과 하나의 획을 구성하는 데는 최대 12개의 코드만이 필요하다.

이렇게 생성된 방향 코드는 인식기의 입력으로 들어가게 되는데 분리된 획들의 공간 정보(위치) 또한 중요한 특징으로 사용한다. 공간 정보는 그림 9의 예에서처럼 두 개 이상의 획으로 분리된 경우에는 각 획들이 어느 위치에 존재하며 다른 획과 어떻게 연관되어 있는지가 아주 중요하다. 획이 분리가 되는 경우는 특징 점을 기준으로 분리가 되므로 특징 점을 포함하는 획의 특징 점 부분과 그 외의 획과 점들의 관계를 공간 정보로써 활용할 수 있다. 그림 9를 예로 든다면 특징 점을 포함하는 획에서 특징 점의 왼쪽에 가로 직선이 하나 붙어 있는 경우로 판단할 수 있게 된다. 이러한 공간 정보는 각 획들의 방향 코드가 인식기에서 획으로써 인식을 한 후에 그 획들을 조합할 때에 정보로 사용한다.

획의 분리와 방향 코드를 추출한 결과 표 1과 같이 stroke 0부터 stroke 14까지 모두 15개의 획으로 집약되었으며 각 획들이 의미하는 획의 모양들을 설명하였다. 각 획에서 자체적으로 하나의 숫자를 구성하는 것(stroke 0~6)과 하나의 획으로는 숫자를 구성하지 못하고 두 개 이상의 획이 함께 존재하여 하나의 숫자를 구성하는 획들로 나누어 질 수 있다. 예를 들면 숫자 4는 stroke 8과 stroke 13이 조합되어 구성되며, stroke 7과 stroke 11이 조합하여 숫자 2를 구성하는 경우이다.

표 1 각 획 모양

| 획 | 모양 |
|---|---|
| stroke 0 | 0 |
| stroke 1 | 3 |
| stroke 2 | 4 |
| stroke 3 | 5 |
| stroke 4 | 6 |
| stroke 5 | 7 |
| stroke 6 | 9 |
| stroke 7 | - |
| stroke 8 | \| |
| stroke 9 | \ |
| stroke 10 | / |
| stroke 11 | ⌐ |
| stroke 12 | ⌐ |
| stroke 13 | ⌐ |
| stroke 14 | ⌐ |

## 4. 인식기 구성

인식기는 은닉층을 갖는 3층 퍼셉트론 구조로 오류 역전파 알고리즘으로 구성하였고 그 구성도는 그림 11과 같다.

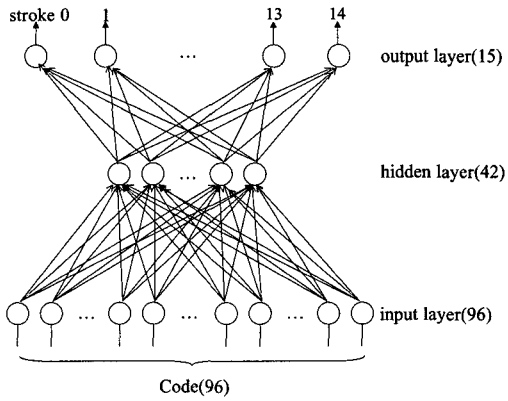



그림 11 인식기 구성도

인식기의 입력은 전처리 과정에서 추출한 각 획의 12개의 방향 코드를 사용하며, 각 코드당 8개의 방향이 있고 방향이 없는 것은 0으로 두었기 때문에 표 2와 같이 8개의 입력이 사용된다.

표 2 방향 코드의 입력값

| 방향 코드 | 입력값 |
|---|---|
| 0 | 00000000 |
| 1 | 10000000 |
| 2 | 01000000 |
| 3 | 00100000 |
| 4 | 00010000 |
| 5 | 00001000 |
| 6 | 00000100 |
| 7 | 00000010 |
| 8 | 00000001 |

따라서 입력 뉴런의 개수는 8×12로 총 96개의 뉴런이 필요하며 출력은 stroke 0부터 14까지의 15개의 획 표현을 위해 15개, 은닉층의 뉴런의 수는 그림 12와 같이 실험에 의해서 42개로 결정하였다.

이 뉴런구조의 인식기에서 학습에 대한 기존의 방법은 데이타베이스의 각 숫자마다의 일부 데이타를 선택하여 신경망 인식기의 학습에 참여시키고 학습에 참여되지 아니한 다른 데이타는 학습 결과를 시험하는 자료로 사용하였다. 따라서 이러한 인식기의 학습과정에서는 좀 더 좋은 결과를 얻기 위하여 최악의 경우에 데이타베이스의 모든 데이타를 학습하게 되는데 이러한 경우

에 신경망의 학습시간 및 신경망의 크기가 커지는 문제점이 발생될 수 있다. 그러나 이 논문에서 제안되는 신경망 인식기는 각 숫자가 가지는 획 패턴의 특징을 추출하여 사전에 학습함으로써 학습 대상이 모든 숫자 데이타베이스에 독립적이다. 또한 입력의 수가 제한되어 학습시간도 절약될 수 있고 입력되는 각 숫자의 다양한 이미지와 상관없이 성공률의 일관성을 유지하고 강건하게 인식결과를 내놓을 수 있다. 이 인식기 내부 뉴런들의 개수 및 입출력의 개수들의 조정은 획들의 패턴수와 그 조합 과정에서 산술적이거나 실험적으로 이루어 졌다. 은닉층 뉴런수의 결정은 실험적으로 이루어 졌는데 그림 12에 나타난 것과 같이 학습시 은닉층의 뉴런의 개수가 42개 이상이 되면 100% 수렴하게 되고 수렴 속도도 수렴률에 비례하여 은닉층의 뉴런의 수가 42개 일 때 가장 적절하였다.

이상에서 언급된 전처리 과정에서부터 인식에 이르는 전체 과정을 그림 13에 나타낸다. 먼저 숫자 이미지를

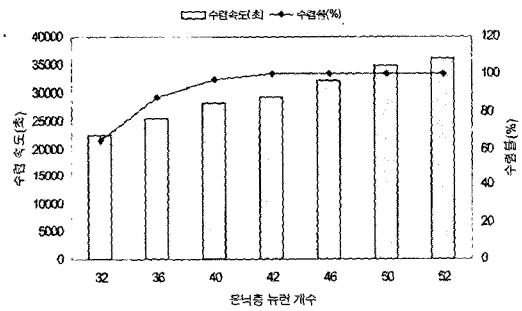



그림 12 은닉층 뉴런수에 따른 수렴 속도

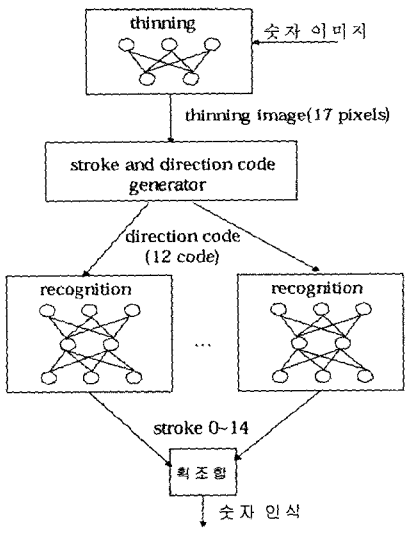



그림 13 전체 흐름도

SOG* 세선화로 수행하여 17개의 점으로 추출하고 추출된 이 점들간의 관계로써 특징 점과 획을 추출하여 각 획별로 8방향 코드를 생성한다. 추출된 각 획들의 방향 코드는 인식기에 입력되어 각 획을 인식하여 나온 획들의 조합으로 최종 숫자를 인식하게 된다.

## 5. 실험 및 실험 결과

### 5.1 실험

본 논문의 실험은 팬티엄 III 1GHz, VC++ 6.0의 환경에서 이루어 졌다. 실험에 사용된 숫자 데이타베이스는 Canada Concordia 대학교, 한국의 연세대학교 그리고 KAIST의 필기체 숫자 데이타베이스를 이용하였다. 실험에 사용된 필기체 숫자 데이타베이스 중에서 Concordia 대학교의 데이타베이스는 0부터 9까지의 숫자 각 600자씩 6,000자를 모두 테스트하였다. 연세대학교의 데이타베이스는 4,089자씩 총 40,890자가 있으며, 그 중에서 숫자 이미지의 일부가 손상되어 형체가 불분명한 것을 제외한 40,000자 중에서 상위 20,000자를 테스트하였고 KAIST 데이타베이스는 23명의 필기자가 쓴 숫자 이미지가 총 5,786자로 구성되어 있고, 이를 인식 테스트에 모두 사용하였다.

전처리 과정의 결과로 추출된 각 획의 12개의 방향 코드가 인식 시스템의 96개의 입력 뉴런에 각각 들어가게 되며 인식 시스템의 학습 단계에서 학습률은 0.02로, 최대 학습 횟수는 20,000번으로 제한하고, 인식 단계에서 역시 학습률 0.02, 임계값은 0.9로 하였다.

본 논문의 실험은 다른 실험과 달리 데이타베이스의 일부를 학습하고 일부를 테스트하는 방식이 아니라, 각 숫자의 모든 이미지 특성들을 수동으로 분석하여 추출한 획들을 방향 코드로 변환한 뒤, 이를 학습에 사용한다. 그리고 위에서 언급한 숫자 데이타베이스의 모든 데이타는 학습에는 사용하지 않고 오로지 인식기의 테스트용 데이타로만 사용한다. 따라서 학습 단계에서 각 획별로 특성을 파악하여 미리 방향 코드를 작성하여 학습을 하였다. 이때 유사 패턴으로 인한 인식의 저하가 발생하는데 이런 경우는 가중치를 더 주어 학습을 시킨다. 예를 들어 1의 패턴 중에 코드가 3인 경우는 숫자 7의 코드 3 5와 유사하여 인식의 임계값을 넘지 못하는 경우가 발생하는데 이런 현상을 없애기 위해 1의 패턴에 대해 가중치를 더 주면 임계값을 넘어 정확히 인식할 수 있다.

표 3은 각 획별 패턴의 개수이다. 패턴의 수가 적은 획에서 패턴이 많은 획과 유사 패턴이 발생하는 경우 앞의 코드 3의 예처럼 가중치를 더 부여하게 되면 효과적인 인식을 기대할 수 있다.

표 3 각 획별 패턴의 수

| 획 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 개수 | 79 | 45 | 19 | 71 | 29 | 7 | 49 | 21 | 16 | 5 | 4 | 78 | 21 | 18 | 21 |

표 3의 각 패턴들은 획의 특징을 고려하여 만든 것들이다. 표 3에 보는 것처럼 단순한 세로 획(stroke 8)의 경우에도 패턴이 많은 이유는 필기체 숫자의 특징상 필기자의 필기 습관에 의한 변형이 발생하기 때문이다. 가장 기본적인 세로 획의 코드는 3 또는 7 이지만 필기된 숫자의 기울어진 정도에 따라 2 6이 나올 수도 있고, 숫자를 곧 바로 쓰지 않고 휘어서 필기한다면 단순하게 두 개의 코드 값 만으로는 표현할 수 없기 때문이다. 다른 숫자들의 경우도 유사하다. 본 논문의 실험에 사용된 필기체 숫자 데이타들은 그림 14에 나타내었다.

### 5.2 실험 결과

그림 14에 나타낸 세 개의 필기체 숫자 데이타베이스에 대해 테스트 한 결과로 다음 표 4와 같은 인식률을 얻었다. Concordia DB에서는 전체 6,000자 중에서 5,933자가 정인식 되었고, 24자가 오인식, 43자가 미인식 되었다. Yonsei DB에서는 전체 20,000자 중에서 19,770자가 정인식, 90자가 미인식, 140자가 미인식 되었다. KAIST DB에서는 전체 5,786자 중에서 5,743자가 정인식, 11자가 오인식, 32자가 미인식 되었다.

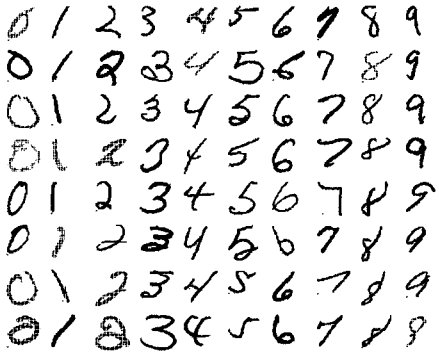
표 4 전체 인식률

| 숫자 DB | 정인식 | 오인식 | 미인식 |
|---|---|---|---|
| Concordia | 98.88 % | 0.40 % | 0.72 % |
| Yonsei | 98.85 % | 0.45 % | 0.70 % |
| KAIST | 99.25 % | 0.20 % | 0.55 % |

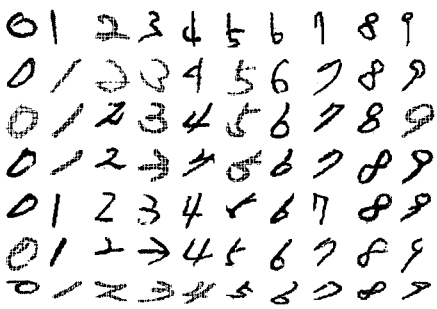
각 숫자별로 인식률을 측정한 결과는 그림 15에서처럼 패턴이 단순한 1과 7은 다른 숫자에 비해 인식률이 아주 높은 반면 4, 8의 인식률은 다소 낮게 나타난다. 그 이유는 1, 7은 획이 단순하고 획 패턴 개수도 작으며, 다른 숫자들과의 패턴이 완전 독립적이다. 다른 숫자들은 그 획 패턴의 수가 1, 7에 비해 많으며, 유사한 패턴이 일부 존재한다. 따라서 유사 패턴에 대해 오인식을 하는 경우가 발생한다.

그림 16에는 각 숫자별로 원 이미지와 획을 분리하여 방향 코드를 생성한 최종 결과를 보여주고 있다. 0, 1, 7, 9와 같은 숫자들은 하나의 획으로 각 숫자를 표현할 수 있으며 나머지 숫자들은 둘 이상의 획의 조합으로 각각 표현할 수 있고, 각 획의 분리가 정확하게 이루어짐을 알 수 있다.

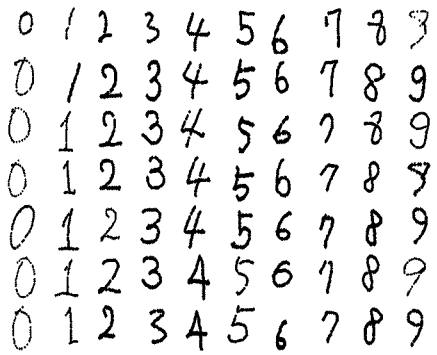
그림 17에는 오인식된 숫자 이미지와 미인식된 숫자 이미지를 나타내었는데 사람이 판단하기에도 어려운 것



(a) Concordia 대학의 필기체 숫자 데이타베이스



(b) 연세대학교 필기체 숫자 데이타베이스



(c) KAIST 필기체 숫자 데이타베이스

그림 14 인식에 사용된 숫자 이미지들

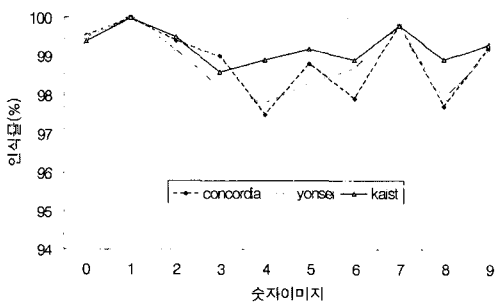



그림 15 각 숫자별 인식률

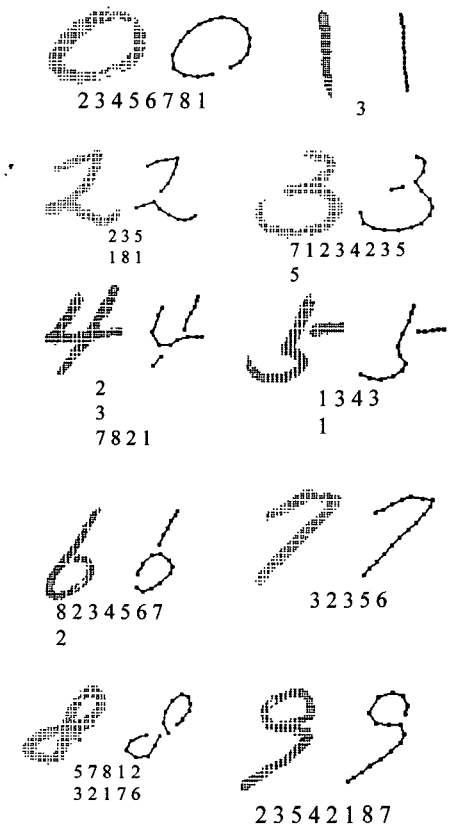
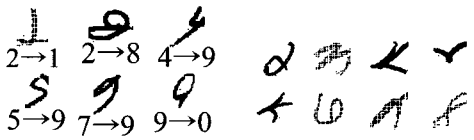



그림 16 획 분리와 방향코드 생성후의 결과

들이 대부분이었다. 오인식의 주된 원인은 역시 방향 코드의 유사함에 의해 발생하였다. 그림 17(a)에서 2와 1이 서로 오인식되는 경우가 발생하는데 이런 경우는 필기자의 필기 습관에 의한 것이고, 나머지 오인식은 세선화된 결과에 의한 것들이었다.




(a) 오인식된 이미지 (b) 미인식된 이미지

그림 17 오인식과 미인식된 이미지들

### 5.3 실험 결과 비교

위 실험 결과를 기존 논문들의 인식률과 사용된 특징들을 함께 비교해 보면 표 5와 같다. 표 5에서 보는 바와 같이 기존의 대부분 방법들이 일반적으로 Concordia Univ. 숫자 DB를 사용하고 이에 대한 인식률은 94.45%~ 98.20%이고 기타 DB를 사용하는 경우에는 94.0%~99.4%에 해당한다. 또한 기존의 방법들은 인식 과정에서 특징들을 복합적으로 사용하고 있다. 따라서 이들을 처리하기 위해서는 여러 단계의 신경망들을 순차적으로 사용하거나 다중 구조를 사용하여 상당한 처리시간을 요구한다.

이에 반해 본 연구의 인식 과정에서는 간단하게 획의 방향 코드들로 구성된 단 하나의 특징만을 사용하여 표 5에서 보이는 바와 같이 세 개의 DB에 대하여 98.85%~99.25%의 인식률을 보인다. 더불어서 기존의 방법들은 학습된 데이타베이스에 대해서만 표 5에서 보여진 인식률을 보장하지만[1,2,8,16,17,22] 본 연구의 인식기는 생성된 획의 특징들이 일반적인 모든 숫자의 특징을 나타내고 이 것들을 학습하였으므로 실험에 사용되지 않은 숫자 DB의 이미지에 대하여서도 같은 인식률을 제공할 수 있다.

## 6. 결 론

본 논문에서는 오프라인 필기체 숫자를 인식하기 위하여 한 개의 특징요소로 획 방향코드를 사용하였다. 다양한 필기체 숫자 패턴들을 분석하여 각 숫자가 지니고 있는 특징을 추출하기 위하여 먼저 SOG* 세선화 기법을 이용하여 세선화 한다. 이렇게 세선화 된 결과를 이용하여 획으로 분리하며, 분리된 획을 8방향 코드로 생성한 후 반복 코드를 제거하여 각 숫자의 특징을 가지는 일반적인 획의 방향 코드로 제작하였다. 이 코드들을 학습 단계에서 신경망으로 학습하고, 시험 단계에서는 생성된 획과 공간 정보를 이용하여 최종적으로 숫자를 인식하게 된다. 3개의 숫자 데이타베이스에 대해 실험한 결과 98.85%~99.25%의 인식률을 나타내었다.

기존의 복합적인 특징을 사용하여 복잡한 전처리 과정을 가지는 방법에 비해 한 개의 특징만을 사용하여 특징 추출에 소요되는 시간을 줄였다. 또한, 모든 숫자 패턴에 대한 일반적인 획의 방향 코드를 제작하여 이를 특징으로 사용하므로 숫자의 기울기나 크기 등에 제약을 없앴고 생성된 획의 방향 코드가 각 숫자의 특징을 모두 표현할 수 있으므로 어떤 숫자 이미지에 대해서도 98.85% 이상의 인식률을 가질 수 있다.

## 참 고 문 헌

[1] 김영준, 이성환, "유전자 알고리즘과 결합된 다층 클러스터 신경망을 이용한 무제약 필기체 숫자의 오프라인 인식", 한국정보과학회논문지, 제21권 제8호, pp.1468-1479, 1994.

[2] 박창순, 김두영, "오프라인 필기체 숫자 인식을 위한 다양한 특징들의 성능 비교 및 인식률 개선 방안", 한국정보처리학회논문지, 제3권 제4호, pp. 915-925, 1996.

표 5 논문별 인식률과 사용된 특징들

| 참고문헌 | 숫자 DB | 인식률 (%) | 사용된 특징 | 연구 년도 |
|---|---|---|---|---|
| 김영준 외1 [1] | Concordia Univ. | 97.80 | 국부특징(방향정보), 전역특징(영상압축) | 1994 |
| | ETR | 99.40 | | |
| 박창순 외1 [2] | Concordia Univ. | 97.10 | 윤곽선의 4방향, 교차거리, 교차, 망, 투영 | 1996 |
| | Dong A Univ. | 98.00 | | |
| 방극준 외3 [3] | NIST | 97.48 | 선형 형태 변환(세선화, 기울기 교정, 덧씌우기), 비선형 형태 변환 (점밀도, 교차횟수, 획간 간격) | 1996 |
| 임길택 외2 [14] | Concordia Univ. | 98.00 | 원 이미지, 회전 이미지, 이동 이미지 | 2000 |
| 강희중 외1 [15] | Concordia Univ. | 96.61 | 다수인식기의 확률적 결합 | 2000 |
| 류강수 외2 [17] | Concordia Univ. | 98.20 | 망, 교차, 투영, 교차 거리, 특정 궤적 | 1995 |
| 류강수 외1 [18] | 자체 구성 | 98.00 | 망, 교차, 투영, 교차 거리, 특정 궤적, 방향 벡터 | 1996 |
| 김현돈 외1 [21] | Concordia Univ. | 94.45 | 유전자 알고리즘+신경망 +가중치 조절 | 2001 |
| 박주양 [22] | Concordia Univ. | 95.00 | 특징 벡터 값, 숫자 영상의 직선 길이, 숫자 영상의 직선의 방향 값 | 1996 |
| 윤종민 외3 [23] | NIST | 99.00 | 기울기, UDLRH 오목성, 배쉬, Haar | 1997 |
| Travieso 외3 [24] | 자체 구성 | 94.00 | 외곽선 추출, 세선화(또는 골격선) | 1999 |
| 본논문 | Concordia Univ. | 98.88 | 세선화를 통한 획의 방향 코드 | |
| | Yonsei Univ. | 98.85 | | |
| | Kaist | 99.25 | | |


[3] 방극준, 조남신, 강창언, 홍대식, "필기체 숫자인식을 위한 병렬 자구성 계층 신경회로망", 전자공학회논문지, 제33권 B편 제7호, pp.1193-1202, 1996.

[4] 이성환, "패턴 인식의 원리 2", 홍릉과학출판사, 1997.

[5] 이성환, "문자 인식 이론과 실제 II", 홍릉과학출판사, 1997.

[6] Keiji Yamada, "Inverse Recall Neural Network Model and Feedback Pattern Recognition," IEEE International Conference on Neural Networks, pp. 399-406, 1993.

[7] Lianwen JIN, Kwokping CHAN and Bingzheng XU, "Off-line Chinese Handwriting Recognition using Multi-Stage Neural Network Architecture," IEEE International Conference on Neural Networks, pp. 3083-3088, 1995.

[8] Y.P. JUN, H. Y, J.W. CHO, "L* Learning: A Fast Self-Organizing Feature Map Learning Algorithm Based on Incremental Ordering," IEICE Trans. INF & SYST., Vol E76-D, No. 6, June 1993.

[9] M. Altuwaijri, M. Bayoumi, "A New Thinning Algorithm for Arabic Characters Using Self-Organizing Network," Circuits and Systems, ISCAS '95, IEEE International Symposium on, Vol 3, pp. 1824-1827, 1995.

[10] P. Ahmed, "A neural network based dedicated thining method," Pattern Recognition, 16, pp. 585-590, 1995.

[11] 이찬희, 정순호, "개선된 SOG 기반 고속 세선화 알고리즘(SOG*)", 한국정보처리학회논문지, 제8-B권 제6호, pp. 651-656, 2001.

[12] John Cowell, Fiaz Hussain, "Thinning Arabic Characters for Feature Extraction," IEEE Information Visualization, Proceedings, Fifth International Conference, pp. 181-185, 2001.

[13] Majid M. Altuwaijri, Magdy A. Bayoumi, "A Thinning Algorithm for Arabic Characters Using ART2 Neural Network," IEEE Transactions on Circuits and Systems-II, Vol 45, No 2, 1998.

[14] 임길택, 남윤석, 진성일, "회전 및 이동 영상을 이용하는 모듈 구조 신경망 기반 필기체 숫자 인식", 한국정보처리학회논문지, 제7권 제6호, pp. 1834-1843, 2000.

[15] 강희중, 이성환, "무제약 필기 숫자를 인식하기 위한 다수 인식기를 결합하는 의존관계 기반의 프레임워크", 한국정보과학회논문지, 제27권 제8호, pp. 855-863, 2000.

[16] 김백섭, 송혜정, "필기체 문자인식을 위한 체인코드에 기반한 효율적인 전처리 및 특징추출", 한국정보과학회논문지(B), 제25권 제12호, pp. 1788-1796, 1998.

[17] 류강수, 김우태, 진성일, "다중 특징과 모듈화된 신경회로망을 이용한 인쇄체 및 필기체 혼용 숫자 인식",

전자공학회논문지, 제32권 B편 제10호, pp. 1347-1357, 1995.
[18] 류강수, 진성일, "모듈화된 신경회로망 중간층 출력의 재학습에 의한 필기체 숫자 인식", 한국정보과학회논문지(B), 제23권 제9호, pp. 931-940, 1996.
[19] 김종석, 홍연찬, "클러스터 신경망을 이용한 우편번호 인식 시스템의 설계", 퍼지 및 지능시스템학회논문지, Vol. 11, No. 2, pp. 132-140, 2001.
[20] H. Freeman, "On the Encoding of Arbitrary Geometric Configuration," IEEE Trans. on Electronic Computers, Vol. EC-10, pp. 260-268, June 1961.
[21] 김현돈, 조성배, "유전자 알고리즘을 사용한 구조적응 자기구성 지도의 최적화", 퍼지 및 지능시스템학회논문지, Vol. 11, No. 3, pp. 223-230, 2001.
[22] 박주양, "문서 영상 이진화와 필기체 숫자 인식", 석사 학위논문, KAIST, 1996.
[23] 윤종민, 신요안, 차형태, 정규식, "신경망 가지치기에 근거한 특징 선택의 필기체 문자 인식에의 응용", 한국정보과학회논문지(B), 제24권 제10호, pp. 1043-1052, 1997.
[24] Travieso, C.M., Morales, C.R., Alonso, I.G., Ferrer, M.A., "Handwritten Digits Parameterisation for HMM based Recognition," Image Processing and Its Applications, Seventh International Conference on (Conf. Publ. No. 465), Vol. 2, pp. 770-774, 1999.

이 찬 희

2000년 부경대학교 전자계산학과 학사
2002년 부경대학교 전자계산학과 석사
현재 부경대학교 전자계산학과 박사과정
관심분야는 인공지능, 신경회로망, 패턴 인식, 컴퓨터 비전

정 순 호

1980년 서울대학교 수학교육과 학사. 1982년 한국과학기술원 전산학과 석사. 2000년 한국과학기술원 전산학과 박사. 1982년~1986년 삼성전자 종합연구소 주임연구원. 1986년~1987년 C&B Technology Co. 개발부장. 1987년~현재 부경대학교 컴퓨터멀티미디어공학부 교수. 관심분야는 인공지능, 컴퓨터 비전, 가상현실, 컴퓨터 보안