한글 필기체 영상 데이타베이스 PE92의 소개

김 대 환, 방 승 양
포항공과대학 전자계산학과
및
산업과학기술연구소 정보통신 연구분야

An Overview of Hangul Handwritten Image Database PE92

D. H. Kim and S. Y. Bang
Dept. of Computer Science & Engineering, POSTECH
and
Computer & Communication Department, RIST

요 약

한글 문자인식 시스템을 개발하기 앞서 생각해야 할 것이 인식실험에 사용될 문자 데이타를 수집하는 것이다. 이 논문에서는 연구 개발자들에게 문자인식 실험에 필요한 충분한 데이타를 제공하며 필기체 문자 데이타를 표준화하여 문자인식 시스템 상호간의 성능을 객관적으로 평가하기 위하여 한글 필기체 문자 데이타베이스 PE92를 개발하였다.

여기서는 PE92 데이타베이스의 소개로서 먼저 PE92를 수집하는데 있어 고려사항들, 즉 필기자, 수집문자의 수, 수집용지의 규격, 데이타베이스의 저장, 데이타의 압축에 대하여 알아본다. 다음 PE92 데이타베이스의 규격을 알아본다.

1 서론

요즈음 컴퓨터를 이용한 정보처리가 활성화되고 사람과 컴퓨터 간의 인터페이스가 중요해짐에 따라 문자인식에 대한 관심이 고조되고 있다. 기존의 문서나 책, 그외의 사람이 직접 손으로 쓴 글을 컴퓨터에 입력하기 위해서 필요한 기술의 핵심이 문자인식이다. 또 문자인식 알고리즘을 개발하기 앞서 생각해야 할 것이 문자인식 실험에 사용될 문자 데이타를 수집하는 것이다. 문자인식은 오프라인과 온라인 두가지로 구분되며 이 둘은 서로 특성이 다르기 때문에 문자데이타의 형태도 다르다. 여기서는 오프라인인 경우에 대해서만 고려하겠다.

오프라인 문자도 인쇄체 문자와 필기체 문자 두가지로 나눌 수 있다. 인쇄체 한글 문자인식은 현재까지 많은 연구가 이루어져 왔으며 실용화 단계까지 와 있다. 이들 대부분의 연구에서 사용된 문자 데이타는 레이저 프린터의 글꼴을 프린트하여 얻어낸 것으로 대량의 문자를 단시일에 얻을 수 있어 데이타를 수집하는 데 그렇게 많은 시간을 할애하지 않아도 된다. 프린터가 출력할 수 있는 글꼴에는 한계가 있지만 일반적으로 사용되고 있는 인쇄체 글꼴은 대부분이 명조체와 고딕체에 한정되기 때문에 이 두 글꼴만을 인식하더라도 실용화할 수 있을 정도이다. 인쇄체 문자에서 생길 수 있는 변형이라면 단지 스캐너로 읽을 때의 잡음뿐이므로 스캐너를 여러번 읽음으로써 언제든지 변형된 데이타를 얻을 수 있다. 그러므로 인쇄체 한글 문자의 데이타베이스는 절실하게 요구되지 않는다.

그러나 필기체 문자의 경우 문자 데이타를 수집하기가 쉽지 않다. 컴퓨터에 의해서 대량으로 만들어 내기는 불가능할 뿐만 아니라 몇 사람이 쓰는 노력만으로는 모든 필기체 데이타를 만들 수 없으며 유용한 데이타가 되지도 않는다. 사람마다 글씨체가 다르기 때문에 특정 몇 사람의 글씨체만으로 필기체 문자인식 시스템을 평가한다는 것은 의미가 없다. 상당히 많은 양의 글자를 많은 변형의 글씨체로 평가해야만 시스템의 성능을 인정할 수 있다. 현재의 한글 필기체 문자인식은 아직 시작 단계이므로 많은 연구는 이루어지지 않았지만 앞으로의 문자인식 연구는 필기체를 위주로 할 것이며 그에 따라

연구 개발자는 필기체 문자 데이타의 수집에 많은 노력을 할애해야 한다. 그러므로 필기체 문자 데이타베이스는 문자인식 연구에 필수적이다.

PE92의 제작 목적은 두가지가 있다. 첫째, 연구 개발자들에게 문자인식 실험에 필요한 데이타를 제공한다. 필기체 문자 데이타는 많은 사람의 노력과 많은 수의 글자를 요구한다. 한글의 글자수는 많이 사용하는 글자만 하더라도 1000자가 넘으며 충분히 다양한 글씨체를 수집하려면 이 1000자에 대하여 100벌 정도는 필요하다. 이같은 방대한 양을 단시일 내에 얻기란 쉽지 않으며 이 정도 양에는 미치지 못하더라도 연구자가 어느 정도 만족할 만한 실험을 하기 위한 데이타를 얻는데도 문자인식 알고리즘 개발에 소요되는 시간 이상을 소비하게 된다. 그러므로 필기체 문자인식에 대한 연구의 의욕이 있더라도 데이타 수집에 많은 시간과 노력이 소비되기 때문에 제대로 시도하지 못할 수도 있다. 따라서 PE92 데이타베이스는 필기체 문자인식에 필요한 충분한 데이타를 제공하여 필기체 문자인식 분야에 손쉽게 접근할 수 있게 하며 연구자에게는 오직 문자인식 알고리즘에만 전념할 수 있게 한다.

둘째, 필기체 문자 데이타의 표준화이다. 앞으로 필기체 문자인식의 연구가 활성화되고 많은 연구결과들이 발표되어 그 연구에 사용된 인식 알고리즘의 성능이 알려지게 된다. 그러나 단순히 그 논문에서 제시된 인식률만 가지고는 문자인식 시스템의 성능을 객관적으로 평가할 수 없다. 왜냐하면 서로 다른 연구들이 서로 다른 데이타를 가지고 실험하여 얻은 인식률이기 때문에 서로 간의 비교 기준이 없다. 그러므로 가장 객관적으로 평가할 수 있는 방법은 같은 실험 데이타에서 얻어진 인식률을 비교하는 것이다. 즉 하나의 표준 데이타를 선정하여 연구자들은 이 데이타를 가지고 인식 실험한 인식률만을 발표한다면 이 결과는 누구라도 수긍할 수 있는 인식률이 될 것이다. PE92는 우리나라에서 만들어진 최초의 한글 필기체 문자 데이타베이스이므로 표준데이타로서의 역할을 충분히 할 수 있다. PE92는 이 두가지 목적을 달성하기 위해서 인식실험에 충분한 양의 글자 수와 세트 수를 제공하며 다양한 변형을 수용한다.

2 PE92의 수집

이 장에서는 PE92를 어떻게 수집했으며 그에 대한 고려사항은 무엇인지 알아본다. 고려사항으로는 필기자, 수집문자의 수, 수집용지의 규격, 데이타베이스의 저장, 데이타의 압축이다.

2.1 필기자

필기자는 되도록 같은 남녀비율, 광범위한 직업층, 광범위한 연령층의 사람들을 흡수하도록 노력하였다. 이를 위해서는 어느 특정 집단을 대상으로 수집하는 방법보다 가족을 대상으로 수집하는 것이 바람직하다. 왜냐하면 가족은 남녀 비율이나 연령층, 직업층이 일반적으로 광범위하기 때문이다. 여기서 한가지 고려해야 될 점은 다양한 변형의 글씨체를 수집해야 하는 것이다. 비록 많은 계층의 사람을 대상으로 수집하였다 하더라도 글씨체의 변형이 다양하지 않을 수 있다. 다른 계층의 사람이라도 비슷한 글씨체로 쓰는 사람이 있는 반면 같은 계층의 사람이라도 성격이나 습관 등에 따라 독특하게 글씨를 쓰는 사람도 있다. 여기서 보다 비중을 두어야 할 것은 다양한 글씨체 수집이다. 많은 사람들이 한 글씨체를 선호하여 이 글씨체를 데이타로 많이 수집하였다 하더라도 문자인식의 실험을 위해서는 이 글씨체의 대표적인 표본 하나만 있으면 충분하며 다른 유사한 것들은 자연히 적용될 수 있기 때문이다.

그림 1은 필기자의 연령별 분포를 보여준다. 수집용지에 이름과 나이, 성별, 직업을 기입하는 란이 있는데 연령분포와 남녀분포는 이를 기준으로 통계를 내었다. 따라서 기입하지 않은 사람은 포함되지 않으므로 실제 참여한 사람은 이보다 더 많다. 특히 직업 란은 기입하지 않은 사람이 많아 이 통계는 제외하였다. 평균연령은 27.11세이며 최저연령은 15세이고 최고연령은 54세이다. 남녀 분포는 남자가 338명, 여자가 216명으로 남자가 여자의 1.56배 이다.

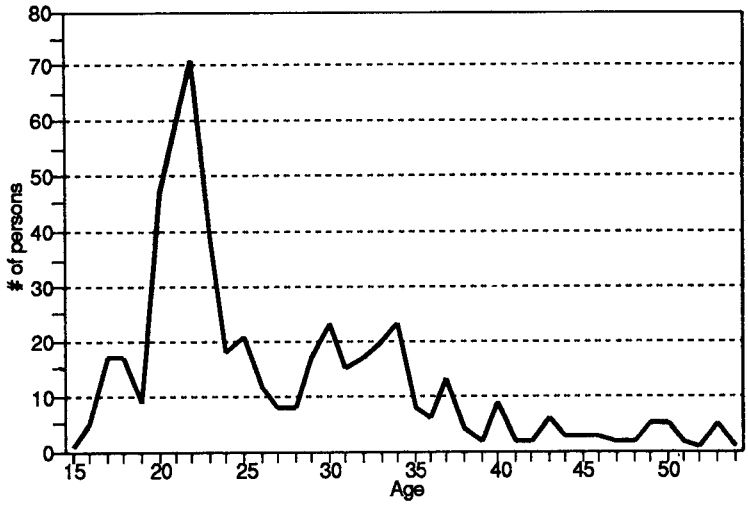

그림 1: 필기자의 연령별 분포

2.2 수집문자의 수

PE92에서 대상으로 하는 문자 인식 시스템은 오프라인 한글 필기체 문자인식 시스템이며 이 문자 인식 시스템은 음절 단위로 인식함을 가정한다. 한글의 총 글자수는 11172자인데 PE92에서 대상으로하는 글자수는 2350자이다. 이는 KS 완성형 한글의 글자 수이다. 한글의 모든 조합 문자 수인 11172자를 다 수집할 수 있다면 이상적이지만, 11172자에서 2350자 이외의 글자는 특수한 경우가 아니고는 거의 사용되지 않을 뿐 아니라 이 글자가 전체 인식률에 미치는 영향은 무시할 정도이다. 또 인식시스템이 10000자 이상의 문자를 대상으로 적절한 시간 안에 실험하기는 거의 불가능하다. 그러므로 방대한 양의 데이타를 얻는데 소비되는 시간과 노력에 비해 실제 실험용으로는 별로 유용하지 않다. 지금의 문자인식이 필기체는 물론이고 인쇄체의 경우라도 시스템의 속도나 용량의 한계 때문에 1000자 이상의 문자를 대상으로 실험한 경우는 거의 드물다. PE92가 선택한 2350자 중에서도 거의 사용되지 않는 글자들이 많기는 하지만 적어도 KS 완성형 문자까지는 모두 포함할 수 있는 데이타베이스를 만들기 위해서이다.

수집문자의 세트 수는 100벌로 한다. 데이타의 세트 수는 많을 수록 좋지만 수집하는데 필요한 시간과 노력을 감안한다면 무턱대고 많이 수집할 수는 없다. 필요하다면 추가적으로 수집 가능하기 때문에 그렇게 큰 문제는 되지 않으며 인식실험하는데는 100벌로도 충분하다.

이 100벌 중 70벌은 불특정 다수의 다양한 글씨체를 수집하였고 30벌은 한 사람이 한 벌 2350자를 모두 쓴 글씨체를 수집하였다. 인식 시스템의 성능이 좋아지면 특정 글씨체만이 아니라 불특정 다수의 다양한 글씨체도 인식해야 하며 어느 정도 잡음이 들어 있더라도 인식할 수 있어야 한다. 그러므로 비슷한 글씨체라도 조금씩의 변형이 있는 많은 사람의 글씨체가 필요하다. 이 조건을 만족시키기 위해서 처음 70벌은 많은 사람의 다양한 글씨체를 수집한 것으로 한사람이 100~500자 정도를 쓴 것이다. 반면 인식 시스템을 처음 개발하려는 사람들에게는 다양한 글씨체는 무리가 있다. 처음부터 문자의 모든 변형을 알고리즘에 고려해 넣어야 다양한 글씨체를 인식할 수 있으므로 기본적인 알고리즘에 대하여 하나씩 실험해 볼 수 없다. 이와같이 특수한 글씨체, 즉 필자 종속적인 문자를 실험하는 시스템을 위해서는 같은 필체의 문자세트가 필요하다. 그래서 100벌 중 나머지 30벌은 70벌의 다양한 글씨체 중에서 대표적인 글씨체를 가진 30명을 선택하여 이 사람들에 대해서 한벌 모두를 쓰게 하였다.

그림 2: 수집용지 3번을 필기자가 쓴 예

2.3 수집용지의 규격

수집용지 한장에는 최대한 100자까지 쓸 수 있으므로 필기체 한 벌 2350자를 모두 수집하려면 24장의 용지가 필요하다. 1~2번 용지는 97자를 쓰며, 3~24번 용지는 98자를 써서 각 용지에 고르게 분포되게 한다. 그림 2는 수집용지 3번을 필기자가 쓴 예를 보였다.

PE92는 잡음이 섞이지 않은 깨끗한 데이타를 수집하기 위하여 여러 조건을 고려해 보고 용지와 펜을 선택하였다. 실용화 단계에서는 문서에 잡음이 들어갈 수도 있고 펜이 선명하지 않아 흐리게 나올 수도 있어 인식 시스템은 이 경우도 고려해야 한다. 그러나 인식실험을 위한 데이타베이스를 만들 때는 잡음이 없는 데이타를 만들어야 한다. 문자인식 시스템이 초기에 개발될 때는 깨끗한 데이타에만 인식이 가능하며 다음 차츰 잡음에 강한 시스템으로 확장한다. 그러나 처음부터 잡음이 섞인 데이타로는 초기 개발이 힘들게 된다. 또한 잡음이 섞인 데이타는 깨끗한 데이타로부터 인위적으로 만들 수 있다. 연구자가 자기가 고안한 인식 방법의 우수성을 증명하기 위해서 표준 데이타에 어떤 잡음을 섞어서 데이타의 질을 떨어뜨려 실험할 수도 있다. 그러나 원래부터 잡음이 섞인 데이타에서 깨끗한

데이타를 만들어 내기는 불가능하다. 이 같은 상황을 고려할 때 처음 만드는 데이타는 가능하면 잡음이 섞이지 않은 것이어야 한다.

선명한 데이타를 얻기 위해서는 문자의 명도와 배경의 명도는 많이 차이 날 수록 좋으므로 가능하면 문자는 검게, 배경은 희게 하도록 한다. 사용된 수집용지는 일반적으로 많이 이용되고 있는 흰색의 A4 용지를 사용하였으며 용지의 두께는 두꺼울 수록 배경은 희게 나타나므로 좋다. 펜은 진하고 번지지 않으며 튀지 않고 선이 가는 것이 좋다. 이런 펜을 사용해야 선명한 문자를 얻을 수 있기 때문이다. 펜의 색깔은 검은 색이외에는 안된다. 스캐너로 문서를 읽을 때 gray level로 읽으므로 다른 색깔의 펜으로 쓴다면 검은 색의 명도보다 낮은 명도의 문자가 얻어지므로 선명하지 않게 된다. 따라서 여러 종류의 펜을 실험해 보아 위의 조건에 가장 알맞은 펜을 선택하여 이 펜으로만 글씨를 쓰게 하였다.

문자를 쓰는 칸은 $9mm \times 9mm$로 한다. 일본의 한자 데이타베이스 ETL9[5]의 수집에서 사용된 칸의 크기는 $8mm \times 9mm$이다. 또한 원고지의 칸의 크기는 일정하지 않지만 $9mm \times 9mm$ 정도이다. 그러므로 한글 문자를 쓰기에는 $9mm \times 9mm$로도 충분하다. 그림 2에서 이 칸들을 볼 수 있다. 예시문자는 고딕체 프린터 글꼴을 사용하였다. 예시문자를 필기체로 쓴다면 필기자는 그 글씨체를 모방할 수 있다. 즉 예시문자는 어떠한 장식이나 기교가 들어가지 않은 단순한 글씨체가 좋다. 인쇄체 문자의 경우도 명조체보다는 고딕체가 삐침이 들어가지 않고 단순하여 모방의 가능성이 더 줄어든다.

각 용지에 쓰여진 문자의 배열은 무작위하게 배열하였다. 이렇게 함으로써 유사한 문자를 계속 썼을 때 생기는 손의 피로를 줄일 수 있다. 문자의 배열을 가나다 순으로 배열했을 때 같은 자음이 연속해서 배열되기 때문에 손에 피로가 가해진다. 잦기 순으로 배열했을 때는 단순한 글자는 대부분 잦기 순위 높은 곳에 배치되고 복잡한 글자는 낮은 곳이 배치되기 때문에 이 경우에도 역시 피로가 가해지게 된다. 따라서 가장 좋은 방법은 무작위로 배치하여 어느 곳에도 비슷한 문자들이 연속해서 배치되는 것을 막는 것이다.

2.4 데이타베이스의 저장

스캐너로 데이타를 받을 때는 gray level로 받는다. 지금의 문자인식 방법은 대부분이 black/white 영상으로 실험을 하지만 gray level로 받는 것이 원래의 영상을 정확하게 나타낼 수 있다. 차후에 gray 영상에서 b/w 영상으로의 변환은 threshold 처리를 하여 간단하게 처리되지만 역변환은 불가능하다. 앞으로 gray 영상을 이용한 문자인식 알고리즘이 나올 수 있으며 영문의 경우 벌써 발표되었다. 또 일본의 문자 데이타베이스는 거의 대부분이 gray로 수집한다. 그러므로 gray level 영상을 얻는 것이 미래를 생각하더라도 더 유용한 데이타가 될 것이다. 여기서는 256(8 bit) gray level을 선택하였다. 스캐너의 제한 때문에 256 gray level을 선택하였지만 이 정도면 충분히 영상의 명도를 깨끗하게 나타낼 수 있다. 1byte(8bit)가 한 점을 나타낼 수 있기 때문에 다루기도 편리하다. 스캐너의 해상도는 300dpi이며 한 문자의 크기를 100×100으로 정규화해서 영역분리한다. 다음 영역분리한 문자 데이타를 보조기억장치에 저장해야 하는데 PC에서는 거의 불가능하다. 보조 기억장치 중 가장 용량이 큰 것이라도 1.44Mbyte 밖에 되지 않는다. 특수한 보조기억장치, 즉 광디스크, 카트리지 등을 사용할 수도 있지만 일반화가 되어있지 않아 불편하다. 현재로서는 PC는 대상에서 제외하고 워크스테이션에서만 사용할 수 있게 만들었다. 워크스테이션은 보조 기억장치로 카트리지가 널리 사용되고 있으므로 150Mbyte 카트리지 테입에 저장하였다. 이때 문자세트 한벌이 (100×100 pixel) × 1byte × 2350자 = 23.5Mbyte 만큼의 용량을 가지는데 2.5절의 데이타 압축을 이용하면 16Mbyte 정도의 크기를 가진다. 100벌 모두를 저장하려면 1.6Gbyte 정도가 필요하므로 PE92는 150Mbyte 카트리지 11개를 사용하였다.

2.5 데이타의 압축

PE92는 데이타베이스의 크기가 방대하기 때문에 압축하여 저장하는 방법이 필요하다. 먼저 생각해 볼 수 있는 것이 이미지 압축기법인데 이 기법을 사용하면 큰 효율로 압축할 수 있지만 원래의 정보

를 많이 잃어버린다. 거의 90% 이상까지도 압축이 가능하지만 압축률이 클 수록 많은 정보를 잃는다. PE92에서는 정보를 잃어버리지 않고 압축하기 위해서 이진 화일의 압축기법을 사용하였다. PE92의 문자 데이타가 영상이라 많이 압축될 것으로 보이지만 gray level이면 이진 화일과 거의 동일하기 때문에 그렇게 큰 압축률을 낼 수 없다. 여기서는 이진화일의 압축에 사용되는 기법들 중 대표적인 것을 실험하여 압축률과 압축시간, 범용성을 고려해서 가장 적절한 것을 선택하였다. 실험의 대상으로 선택된 기법은 Compress, Lharc, Zip, Huffman 이다.[8, 9]

Compress, Lharc, Zip는 기본적으로는 LZW 기법을 쓰고 여기에 다른 기법들을 추가하여 효율을 높이고 있다. Huffman coding은 Entropy 이론을 이용한 것으로 코드가 나올 확률에 의거하여 코드의 길이를 결정한다. 이외의 기법 중 Arithmetic coding, Order-n coding은 Huffman coding과 같이 Entropy 이론을 근거로 하며 Huffman coding과 거의 비슷하지만 조금 나은 압축률을 가진다. 그러나 압축시간이 너무 길기 때문에 대상에서 제외했으며 이들 여러 기법을 혼합하여 사용해도 높은 압축률을 낼 수 있지만 너무 많은 압축시간을 소비하기 때문에 제외했다.

압축기법	화일이름	원래크기	압축후	시간	difference	압축후	시간
Compress	*80.1.seg.Z*	970003	668139	0:10	*80.1.d.Z*	639187	0:10
Lharc	*80.1.seg.lzh*	970003	660428	0:39	*80.1.d.lzh*	632562	0:41
Zip	*80.1.seg.zip*	970003	656838	0:41	*80.1.d.zip*	628901	0:47
Huffman	*80.1.seg.huf*	970003	710156	1:18	*80.1.d.huf*	574317	0:57

가) *80.1.seg*와 *80.1.d*의 압축비교

압축기법	화일이름	원래크기	압축후	시간	difference	압축후	시간
Compress	*80.5.seg.Z*	980003	651777	0:09	*80.5.d.Z*	636097	0:10
Lharc	*80.5.seg.lzh*	980003	647175	0:41	*80.5.d.lzh*	629813	0:42
Zip	*80.5.seg.zip*	980003	643122	0:43	*80.5.d.zip*	625649	0:46
Huffman	*80.5.seg.huf*	980003	688039	1:15	*80.5.d.huf*	572846	0:59

나) *80.5.seg*와 *80.5.d*의 압축비교

표 1: 압축기법들의 압축크기와 시간 비교, difference 변환 후의 압축 비교

표 1는 각 압축기법들을 두개의 화일 *80.1.seg*와 *80.5.seg*에 대하여 실험한 결과를 보여주고 있다. *80.1.seg*를 Compress로 압축한 화일은 *80.1.seg.Z*이고 Lharc로 압축한 화일은 *80.1.seg.lzh*이며, Zip은 *80.1.seg.zip*, Huffman은 *80.1.seg.huf*이다. 여기서 보듯이 Compress는 다른 기법과 비교할 수 없을 만큼 빨랐다. 범용성 면에서도 UNIX 시스템이면 모두 사용할 수 있으므로 좋았다. 물론 압축 프로그램을 데이타베이스와 함께 라이브러리로 제공해도 되지만 범용성이 있으면 상당히 편리하다. 압축률은 다른 기법에 비해 못하지만 압축시간, 범용성 면에서 월등하기 때문에 낮은 압축률을 보상해준다. 따라서 PE92에서는 Compress 기법을 선택하였다.

이번 연구에서 또다른 한가지 기법을 도입하였다. PE92의 데이타베이스는 이진화일과는 다른 특성을 가지고 있다. 즉 데이타가 영상이라는 것이다. 그런데도 일반 이진화일과 같은 방법으로 압축한다면 이런 성질을 전혀 이용하지 않는 것이다. 영상은 인접한 점들끼리의 명도는 급격하게 변하지 않고 연속으로 변한다는 특성을 가지고 있다. 특히 PE92의 경우는 gray level의 문자로 문자와 배경의 경계에서는 문자의 명도와 배경의 명도가 급격하게 변하지 않고 연속으로 변한다. 이와 같은 특성을 이용하여 연속된 점들간의 차이(difference)를 얻는 변환을 하여 압축을 하면 좀더 나은 압축률을 얻을 수 있다. 극단적으로 선형으로 변하는 영상의 경우 difference에 의해서는 점들이 하나의 값으

로 나타내진다. 항상 이렇게 좋은 경우만은 아니라도 연속인 경우 difference 변환을 하면 점의 값이 비슷하게 나올 가능성이 많다. 표 1에서 원래화일 *80.1.seg*로 difference 변환한 것이 *80.1.d*이며 이로 Compress한 화일이 *80.1.d.Z*이다. 이 실험결과에서 보듯이 어떤 압축 알고리즘을 사용하더라도 difference 변환을 하면 항상 압축률이 향상되었다.

3 PE92의 규격

PE92 데이타베이스는 100개의 tar 화일로 구성되어 있다. 한 tar 화일은 2350자의 문자를 포함하는 한글 문자세트 한벌이다. 이 tar 화일은 순서대로 *"1.set.tar"*, *"2.set.tar"*, ⋯ , *"100.set.tar"*로 이름 붙여졌으며 *"1.set.tar"*, ⋯, *"70.set.tar"*의 70벌은 다양한 글씨체를 모은 데이타이며 *"71.set.tar"*, ⋯, *"100.set.tar"*의 30벌은 한사람이 한벌 모두를 쓴 데이타이다. 이들 데이타는 사람에 의해 인식 실험하여 인식하기 쉬운 데이타에서 인식하기 어려운 데이타 순으로 어느 정도 정렬되어 있다. 이 정렬은 10벌 단위로 되어 있다. 즉 *"21.set.tar"*은 쉬운 데이타, *"30.set.tar"*은 어려운 데이타이고 그 사이는 인식이 쉬운 정도에 따라 정렬되어 있다. *("21.set.tar"*, ⋯, *"30.set.tar")*, *("31.set.tar"*, ⋯, *"40.set.tar")*, ⋯, *("91.set.tar"*, ⋯, *"100.set.tar")* 들의 각 10벌내에서는 정렬되어 있다. 단 *("1.set.tar"*, ⋯, *"20.set.tar")*은 20벌이 정렬되어 있다. 그러나 서로 다른 10벌들과는 정렬되어 있지 않다. 물론 전체적으로 정렬되어 있는 것이 이상적이지만 데이타를 한꺼번에 수집하기 어려울 뿐만 아니라 이 데이타에서 한꺼번에 문자를 영역분리하기도 쉽지 않다. 따라서 어느 정도의 데이타가 수집되면 이를 가지고 데이타베이스를 만들므로 부분적인 정렬로 밖에 할 수 없다. 한 벌의 문자세트는 24장의 용지화일로 구성된다. 즉 $n.set.tar = \{ n.1.d.Z, n.2.d.Z, \cdots, n.24.d.Z \}$, $n=1, \cdots, 100$이다. 용지화일 *n.m.d.Z*에서 n은 문자세트의 벌 수, m은 용지의 장 수이며, d는 difference를 의미하고 Z는 Compress 화일임을 의미한다. *n.1.d.Z*와 *n.2.d.Z*는 97개의 문자가 있으며, *n.3.d.Z*, *n.4.d.Z*, ⋯, *n.24.d.Z*는 98개의 문자가 있어 전체 한 벌에는 2350개의 문자가 있다. 용지화일 *n.m.d.Z*는 *n.m.seg* 화일을 변환하여 만든 것으로 다음 형식으로 구성되어 있다.

*n.m.seg*는 다음과 같은 화일형식을 가지고 있다.

	오프셋	내용
헤더	0 byte	R (문자의 행 수 = 100)
	1 byte	C (문자의 열 수 = 100)
	2 byte	N (문자의 갯수 = 97 또는 98)
문자 데이타	3 ~ R·C+2 byte	n번 용지의 첫번째 문자
	R·C+3 ~ 2R·C+2 byte	n번 용지의 두번째 문자
	⋯	⋯
	(N-1)·R·C+3 ~ N·R·C+2 byte	n번 용지의 N번째 문자

각 문자는 R·C개의 점으로 이루어지며 각 점은 한 byte의 gray level이다. 이 gray level은 0에서 255까지의 값을 가지며 값이 클수록 명도가 커진다. 즉 0은 검은색이며 255는 흰색이다. 화일의 크기는 *1.1.seg*가 97자의 문자를 가지고 있으므로 헤더 3byte와 문자데이타 100 × 100 × 97 byte를 합하면 970003 byte 이다.

다음은 gray 화일의 Compress시 효율을 높이기 위한 전처리로서 difference 화일 변환 기법이다.

```
t = 0;
for(i=0; i<RC; i++) {
        s = d[i];
        d[i] = t - s;
        t = s;
}
```

위의 알고리즘에 의해서 데이타들은 다음과 같이 변환된다. 처음의 0은 데이타에는 포함되지 않은 초기값이다.

원래의 화일 :	0	d[0]	d[1]	d[2]	···	d[RC-1]
처리된 화일 :	0	-d[0]	d[1]-d[0]	d[1]-d[2]	···	d[RC-2]-d[RC-1]

위의 diffrence 변환에 의해서 *n.m.seg*는 *n.m.d*로 바뀌게 되며 다음 이 화일을 Compress하여 *n.m.d.Z*가 만들어진다.

4 결론

본 연구에서 한글 필기체 문자 데이타베이스인 PE92를 제작하였다. 이 논문에서는 PE92를 수집하는데 있어 고려사항들인 필기자, 수집문자의 수, 수집용지의 규격, 데이타베이스의 저장, 데이타의 압축에 대하여 알아보았으며 PE92 데이타베이스의 규격을 알아보았다. PE92는 한글 2350자 100벌을 수집하여 필기체 문자인식 연구 개발자들에게 충분한 실험 데이타를 제공할 것이며 문자인식 평가의 표준화에도 기여할 것이라 기대된다. 향후 PE92가 많은 인식 실험에 이용되면 데이타베이스 자체의 품질에 대한 평가가 이루어져 잘못된 점, 추가할 점 등이 밝혀질 것이다. 그러면 다음에는 이를 보완한 더 나은 데이타베이스를 제작할 수 있을 것이다.

참고 문헌

[1] H. Yamada, S. Mori, *An Analysis of the Hand-Printed Character Data Base I*, 전통연휘보, 39, 8, pp. 580~599, 1975, 8.

[2] T. Saito, H. Yamada, K. Yamamoto, S. Mori, *An Analysis of Handprinted Character Data Base V - Evaluation of KYOIKU-KANJI Characters by Pattern Matching Approach* -, 전통연휘보, 45, 1, pp. 49~77, 1981, 1.

[3] T. Saito, H. Yamada, K. Yamamoto, *An Analysis of Handprinted Character Data Base VI - An Analysis of KYOIKU-KANJI Characters by Directional Pattern Matching Approach* -, 전통연휘보, 46, 12, pp. 695~725, 1982, 12.

[4] T. Saito, H. Yamada, K. Yamamoto, M. Yasuda, *An Analysis of Handprinted Character Data Base VII - An Intuitive Analysis of KYOIKU-KANJI Characters* -, 전통연휘보, 47, 4, pp. 261~275, 1983, 4.

[5] Taiichi Saito, Hiromitsu Yamada, Kazuhiko Yamamoto, *On the Data Base ETL9 of Handprinted Characters in JIS Chinese Characters and Its Analysis*, 일본 전자통신학회 논문지, Vol. J68-D, No. 4, pp. 757~764, 1985, 4.

[6] Kazuo Toraichi, Ryoiichi Mori, Iwao Sekita, Kazuuhiko Yamamoto, Hiromitsu Yamada, *Handprinted Chinese character Database*, Computer Recognition and Human Production of Handwriting, World Scientific Publ. Co., pp. 131~148, 1989.

[7] Masaaki Yoneda, Hiroyuki Hase, Mitsuru Sakai, *A consideration on the Evalution of Character Variation*, 일본 전자정보통신학회 논문지 D-II, Vol. J75-D-II, No. 1, pp. 103~110, 1992, 1.

[8] Mark Nelson, *Arithmetic Coding and Statistical Modeling*, Dr. Dobb's Journal, pp. 16~29, 1991, 2.

[9] Timothy Bell, Ian Witten, John Cleary, *Modeling for Text Compression*, ACM Computing Surveys, Vol. 21, No. 4, pp. 557~591, 1989, 12.

[10] 포항공과대학, "한글 필기체 영상데이타베이스의 구축", 한국전자통신연구소 최종연구보고서, 1992, 6.