PDA상에서의 한글 필기체 매칭 알고리즘
(A Hangul Script Matching Algorithm for PDA)

조 미 경 † 조 환 규 ††
(Mi Gyung Cho) (Hwan Gue Cho)

요 약 전자 잉크 데이터는 펜 기반 컴퓨터나 PDA(Personal Digital Assistants)등에서 자연스럽고 편리한 데이터 입력을 제공하기 위해 펜으로 입력한 데이터를 온라인 문자 인식기를 이용하여 아스키 문자로 변환하지 않고 스크립트 형태로 저장하는 데이터를 말한다. 전자 잉크 데이터를 사용하기 위해 가장 중요한 것 중 하나는 전자 잉크 데이터의 검색 문제이다. 본 연구에서는 전자 잉크 데이터를 획 특징 벡터 형태로 저장하고, 이를 이용해서 잉크 데이터를 검색하는 매칭 알고리즘을 제안하고 구현하였다. 제안된 매칭 알고리즘은 입력된 데이터를 곡률(curvature)을 이용하여 기본획(primitive stroke)으로 분리하고 기본획의 종류를 결정한 다음 획 특징 벡터를 생성한다. 그리고 동적 프로그래밍 기법에 의해 획 특징 벡터의 거리값을 계산한다. 제안된 매칭 알고리즘을 이용하여 다양한 실험을 하였으며 한글 스크립트로 구성된 경우 97.7%이상의 매칭률을 보여 주었고 한글 및 한자 혼합 스크립트에서는 94%이상의 매칭률을 보여 주였다.
키워드 : PDA, 매칭 알고리즘, 한글 필기체, 휴대용 컴퓨터

Abstract Electronic Ink is a stored data in the form of the handwritten text or the script without converting it into ASCII by handwritten recognition on the pen-based computers and Personal Digital Assistants(PDAs) for supporting natural and convenient data input. One of the most important issue is to search the electronic ink in order to use it. We proposed and implemented a script matching algorithm for the electronic ink. Proposed matching algorithm separated the input stroke into a set of primitive stroke using the curvature of the stroke curve. After determining the type of separated strokes, it produced a stroke feature vector. And then it calculated the distance between the stroke feature vector of input strokes and one of strokes in the database using the dynamic programming technique. We did various experiments and our algorithm showed high matching rate over 97.7% for only the Korean script and 94% for the data mixed Korean with the Chinese character.
Key words : PDA, matching algorithm, hangul script, mobile computer

1. 서 론

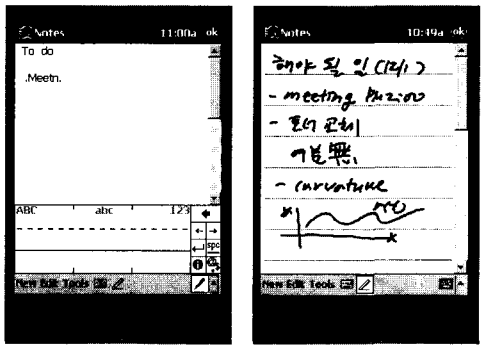
PDA와 같은 소형 펜 기반 컴퓨터 환경에서는 스타일러스(stylus) 펜이 주된 입력 수단이 된다. 이러한 환경에서 데이터 입력 방식은 소프트 키보드를 이용한 방식, 펜으로 입력한 데이터를 온라인 문자 인식기를 이용하여 아스키코드로 바꾸는 방법 그리고 전자 잉크 데이터가 사용된다. 소프트 키보드는 소형 휴대 단말기의 크기 제약으로 인해 많이 사용되고 있으며 펜으로 소프트 키보드를 클릭하여 문자를 입력하는 방법으로 자연스러운 입력 방식은 되지 못한다. 온라인 문자 인식기는 펜으로 입력한 데이터를 아스키코드로 변환하려고 시도한다. 이것은 일단 아스키코드로 변환하면 저장과 검색 등 많은 연산에서 효율적이며 일반적인 컴퓨터에서처럼 데이터를 처리할 수 있기 때문이다. 문제는 문자 인식 시스템의 인식 오류와 인식 속도에 있다. 인식 오류가 발생하면 사용자는 인식기의 출력을 지운 다음 다시 원하는 문자를 펜으로 입력해야 되는 번거로움이 있다. 온라인 인식 시스템의 인식 오류를 줄이기 위해 그래피티(graffiti) 문자 집합만을 입력하도록 하는 방법도 있다. 제한된 문자 집합에 대해 지정된 획순으로 입력을 하게 하므로 문자 인식 문제를 단순하게 만들어 인식률을 높이는 방법이다. 그러나 이러

· 이 논문은 정보통신부에서 지원하는 대학IT연구센터 육성지원사업의 수행결과입니다.
† 정 회 원 : 동명정보대학교 정보공학부 전임강사
mgcho@tit.ac.kr
†† 종신회원 : 부산대학교 전기전자정보컴퓨터공학부 교수
hgcho@pusan.ac.kr
논문접수 : 2001년 12월 24일
심사완료 : 2002년 7월 16일

한 방법들은 사용자가 일상 생활에서 사용하는 문자와 다른 그래피티 문자를 익혀야 되는 번거러움이 따른다.

펜 기반 컴퓨터에서 데이터 입력의 불편함을 극복하기 위해 Lopresti 등은 인식 시스템을 대체할 수 있는 새로운 방법으로 전자 잉크 데이터(electronic ink data)를 우선적으로 처리할 수 있는 시스템을 제안하였다[1,2,3]. 전자 잉크 데이터란 펜으로 입력한 문자, 심볼, 그림 등 펜 스트록(pen-strokes) 데이터를 의미한다. 사용자가 펜으로 입력한 데이터를 인식할 필요 없이 잉크 데이터 자체로 저장하였다가 처리할 수 있는 시스템을 만들자는 것이다. 만약 인식이 요구되어지면 필요할 때만 인식을 하도록 하는 것이다. 이 방법의 주된 장점은 사용자에게 입력 문자의 종류에 전혀 제한을 주지 않는다는 것이다. 문자 인식 시스템의 경우 숫자나 알파벳, 한글, 몇 개의 특수 문자만을 사용하도록 하고 있다. 하지만 잉크 데이터 자체를 처리하는 시스템에서는 이러한 제한을 둘 필요가 없다. 물론 잉크 데이터 검색을 위해 패턴 매칭과 연관된 문제들을 해결해야 되지만 기존의 문자 인식에 비해 조금 더 쉬운 문제이다.

그림 1은 온라인 문자 인식을 사용하는 펜 컴퓨팅 환경과 전자 잉크 데이터를 첫 번째 데이터 범주로 처리하는 시스템의 예를 보여주고 있다. 그림 1의 (a)는 문자 인식기의 인식 오류가 발생한 경우로 인식기의 출력을 지우고 다시 문자를 입력해야 한다. 또한 문자의 입력은 정해진 인식기 창에서만 이루어져야 한다. 그림 (b)에서는 전자 잉크로 간단한 메모를 작성한 예를 보여주고 있다. 사용자는 펜을 이용하여 종이에 글을 작성하듯이 원하는 내용을 입력 문자의 종류에 제한 없이 사용하므로 그림에서 보듯이 전자 잉크가 더 자연스럽고 편리한 도구임을 알 수 있다.



(a) (b)

그림 1 (a) 문자 인식기 환경 (b) 전자 잉크 데이터 환경

휴대용 컴퓨터에서 할 수 있는 많은 일들 중 기존의 인식 시스템을 굳이 사용할 필요가 없는 몇 가지 이유는 첫째, 사용자들이 PDA 등 휴대용 컴퓨터에서 매일 하는 대부분의 일들은 모두 전자 잉크 영역(electronic ink domain) 내에서 처리할 수 있는 것들이다[2,3]. 예를 들어 간단한 메모나 주소록 기록 등을 들 수 있다. 둘째, 전자 잉크 데이터의 경우 펜으로 입력하는 데이터의 범주를 제한하지 않기 때문에 사용자가 표현할 수 있는 정보의 형태가 다양하다. 셋째, 나중에 문자 인식이 필요할 경우 인식을 수행할 수 있도록 하는 기능을 추가할 수 있다.

펜 기반 컴퓨터에서 전자 잉크 데이터의 처리와 검색을 위한 연구는 1990년대 초반부터 이루어져 다양한 연구 결과가 나왔다. 전자 잉크 데이터의 검색을 위한 몇 가지 대략적인 잉크 매칭 알고리즘들(approximate ink matching algorithms)이 제안되었으며[3,4] 대용량의 데이터 베이스에서 전자 잉크 데이터를 효율적으로 검색하기 위한 방법으로 은닉 마르코프 네트워크를 이용한 방법과 R-tree를 이용한 방법들이 제안되었다[5,6]. 그런데 제안된 방법들은 영어 스크립트를 주된 입력 형태로 개발되었다. 펜으로 영어와 한글 스크립트를 작성할 때 영어는 연속된 원호의 합으로 분해될 수 있는 글자인 반면 한글의 경우 획이 꺽이는 위치와 획의 방향이 영어에 비해 뚜렷하다는 차이점이 있다. 영어 스크립트 매칭을 위해 제시된 알고리즘을 구현하여[4] 한글 스크립트에 적용한 결과 매칭률이 매우 저조하게 나타났다. 따라서 영어 스크립트에 적용했던 매칭 알고리즘을 한글 스크립트에 그대로 적용하는 것은 무의미하며 한글 스크립트 위주의 전자 잉크 데이터를 사용하기 위해서는 한글의 기하학적인 특성을 고려한 매칭 알고리즘의 개발이 필요하다.

본 연구에서는 한글 위주의 스크립트를 전자 잉크 데이터 형태로 PDA에서 사용하기 위한 매칭 알고리즘을 개발하고 구현하였다. 개발된 매칭 알고리즘은 잉크 데이터를 기본획 단위로 나눈 후 동적 프로그래밍 기법을 적용한다. 소형 휴대용 컴퓨터의 하드웨어적인 제약을 고려하여 효율적인 속도를 보장하면서 검색률을 높일 수 있도록 고안되었다. 제안된 알고리즘의 성능을 실험하기 위해 개발된 시스템은 전자 잉크 데이터 기반의 개인 정보 관리 시스템이다. 인명(人名)과 전화 번호, 전자 메일 등의 개인 정보와 함께 간단한 메모를 하여 데이터 베이스로 저장하였다가 사용자가 인명을 쿼리(query)로 주었을 때 원하는 데이터를 검색해 주는 시스템이다.

논문은 다음과 같이 구성된다. 본 연구에서 개발한 전자 잉크 데이터 매칭 시스템의 구조와 기능을 2장에서 설명한다. 3장에서는 전자 잉크 매칭을 위해 제안된 매칭 알고리즘에 대해 설명할 것이다. 그리고 4장에서 실험 모델에 따른 다양한 실험 결과에 대해 언급하고 5장에서 결론을 맺는다.

2. 전자 잉크 데이터 매칭 시스템

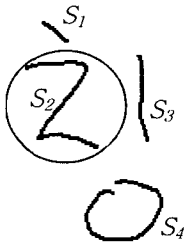
전자 잉크 데이터는 펜으로 입력한 획들(strokes)의 집합이다. 획은 pen-down에서 pen-up사이에 입력된 점들(points)의 순서 있는 집합으로 정의된다. 잉크 데이터 *Ink*는 사용자가 펜을 이용하여 입력한 획들 S_i의 집합으로 다음과 같이 표현할 수 있다.

$$Ink=\langle S_i\rangle, 1\le i\le n$$
$$S_j=\langle(x_{ij}, y_{ij}, t_{ij})\rangle, 1\le i\le m_i \quad (식1)$$

식 1에서 S_i는 사용자가 입력한 i번째 획을 의미하고 원소 x_{ij} ,y_{ij} ,t_{ij}는 각 각 S_i의 j번째 점의 x 좌표 값, y 좌표 값, 그리고 점이 입력되어진 시간 순서를 나타내며, m_i는 S_i를 구성하는 점들의 개수를 의미한다. 그림 2는 네 개의 획으로 구성된 전자 잉크 데이터의 예이다.

펜으로 데이터를 입력할 때 동일한 사람이 동일한 글자를 입력해도 두 번 이상 쓸 때 똑같이 쓸 수 없기 때문에 잉크 데이터의 정보는 달라지게 된다. 그러므로 잉크 데이터 검색에서

정확히 일치하는 데이터 검색은 불가능하다. 따라서 잉크 데이터 검색은 잉크 데이터의 대략적인 매칭(approximate matching)이라고 하는 것이 적절한 표현이다. 전자 잉크 데이터의 검색 문제는 두 개의 전자 잉크 데이터 *P(pattern), T(text)*가 주어져 있을 때 P가 T 내에 포함되어 있는지, 포함되어 있으면 어느 위치에 있는지를 결정해 주는 것이다. 고전적인 스트링 매칭과 다른 점은 P와 T 사이에 완벽한 매칭이 발생하지 않는다는 것이다.



$Ink=\langle S_1, S_2, S_3, S_4\rangle$

그림 2 잉크 데이터의 예

펜으로 입력한 필기체는 동일한 글자라 하더라도 사용자에 따라 필기하는 형태와 획 순서가 다를 수 있다. 제안된 매칭 알고리즘은 이러한 펜 데이터의 특성을 고려하여 필기자 종속(writer-dependant)으로 개발되었다. 필기자 종속 매칭 알고리즘의 경우 필체가 다르면 동일한 글자에 대해서도 다른 모양의 객체로 간주하기 때문에 필체가 다르면 데이터 베이스내의 동일한 글자를 쿼리로 주더라도 원하는 정보를 찾을 수 없다. PDA와 같은 소형 정보 단말기의 경우 개인전용 단말기라는 특성을 가지고 있으므로 필기자 종속 매칭 알고리즘이 적절하며 이는 필기 형태가 다른 사람들에 대해 보안의 기능도 제공할 수 있다.

휴대용 컴퓨터의 발전이 가속화되고 있지만 아직 데스크 탑 컴퓨터에 비해 하드웨어 및 소프트웨어 환경에 제약이 따른다. PDA 제품들을 살펴보면 최신 모델이 메모리 32M와 프로세서 213 Mhz를 장착하고 있다[7]. 이런 하드웨어 환경을 고려할 때 매칭 알고리즘은 데이터 베이스가 커지더라도 효율적인 속도를 보장하도록 단순해야 한다. 본 연구에서는 휴대용 컴퓨터에서 전자 잉크 데이터 매칭을 위해 단순하면서도 높은 매칭률을 보여주는 매칭 알고리즘과 이를 기반으로 하는 매칭 시스템의 개발을 목적으로 하였다.

개발된 매칭 시스템의 구조는 그림 3과 같다. 펜을 통해 입력된 데이터들은 전처리 과정을 거친다. 전처리 과정은 재표본(resample), 삐침 제거, 평활화(smoothing) 순서로 이루어진다. 펜으로 흘려 쓴 한글의 경우 글자간 이어 쓰기와 자소 내, 자소 간 이어 쓰기가 발생할 수 있다. 본 연구에서는 입력 데이터로 필기체의 자소 내와 자소 간 이어 쓰기를 허용하였다. 따라서입력된 데이터들은 획의 곡률을 이용하여 기본획 단위로 분리하는 과

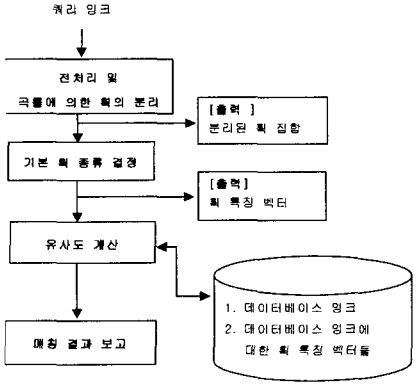
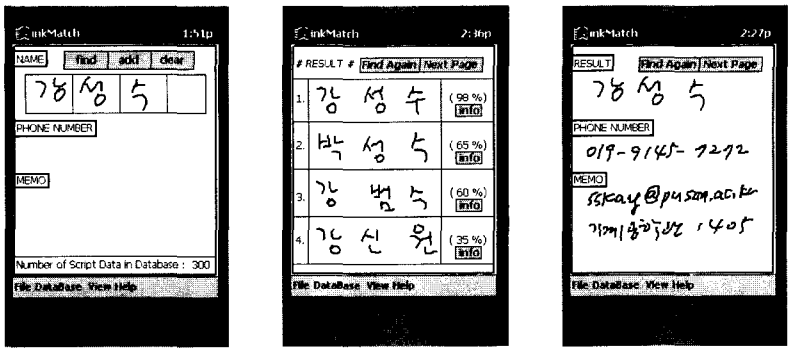


그림 3 매칭 시스템의 구조



(a) (b) (c)

그림 4 구현된 매칭 시스템 수행 결과: (a) 쿼리 잉크 입력 (b) 매칭 결과 (c) 상세한 정보 보기

정을 거친다. 기본획은 휴대용 컴퓨터에서 빠른 매칭 속도를 보장하기 위해 한글의 기하학적인 특성을 고려하여 일곱 개를 지정하였는데 그 종류는 3장에서 설명될 것이다. 다음 단계로 곡률에 의해 분리된 기본획들의 종류를 결정하여 획 특징 벡터(stroke feature vector)를 생성한다. 획 특징 벡터는 입력된 데이터를 기본획 단위로 분리한 정보를 가지고 있는 벡터이다. 획 특징 벡터가 생성되면 동적 프로그래밍에 의해 쿼리의 획 특징 벡터와 데이터 베이스 내 잉크 데이터들에 대한 획 특징 벡터들 사이의 거리값을 계산하여 가장 적은 거리값을 가지는 데이터를 매칭 결과로 돌려준다.

그림 4는 제안한 매칭 알고리즘으로 구현된 매칭 시스템의 수행 예이다. 인명 데이터는 매칭률을 높이기 위해 글자를 구분하여 입력하도록 하였다. 데이터 베이스에는 300개의 인명 데이터와 개인 정보가 저장되어 있다. 그림 (a)에서 사용자가 "강성수"라는 데이터를 입력하고 "find"를 수행하면 시스템이 데이터 베이스내의 데이터들을 비교하여 가장 유사한 네 개의 데이터를 매칭 결과로 찾아준 것을 (b)에서 볼 수 있다. 네 개의 데이터를 보여 주는 이유는 소형 단말기의 크기 제약 때문이다. 매칭 결과에 있는 숫자는 정확도를 보여 주는 것으로 1순위로 찾아준 매칭 결과는 98%의 정확도를 가짐을 알 수 있다. 그림 (c)는 매칭 결과로 찾아준 인명데이터에 대한 상세한 정보를 보기 위해 "info" 단추를 누른 결과이다.

3. 전자 잉크 데이터 매칭 알고리즘

3.1 곡률에 의한 획 분리

본 연구에서는 입력된 데이터를 기본획 단위로 나누기 위해 획을 구성하는 점들의 곡률 값을 사용하였다. 그림 5에 있는 곡선을 나타내는 벡터 함수를 $r(t)$라 하고 호의 길이가 s로 매개 변수화되었다고 가정하자. 그러면 $r'(t)$가 곡선 c의 접선 벡터이므로 곡선의 단위 접선 벡터 T는 식 2와 같이 나타낼 수 있다.

$$T = \frac{r'(t)}{\|r'(t)\|} = \frac{dr/dt}{ds/dt} = \frac{dr}{ds} \qquad (식2)$$

곡선 c에서 s가 증가할 때 T는 c를 따라서 방향은 바뀌어도 길이는 변하지 않으면서 움직인다. 그림 5에서 벡터 T의 방향은 점 P_1과 P_2사이의 곡선 부분에서는 변화가 없지만 P_3와 P_4 사이와 P_4와 P_5 사이의 곡선 부분을 따라서는 방향의 변화가 매우 현저하게 나타난다. 한 점에서 곡선 c의 곡률 k는 단위벡터 T의 호의 길이에 관한 변화율로써 식 3과 같이 계산된다.

$$k = \left\| \frac{dT}{ds} \right\| = \left\| \frac{dT/dt}{ds/dt} \right\| = \frac{\|T'(t)\|}{\|r'(t)\|} \qquad (식3)$$

곡률 정의에 의해 원의 한 점에서의 곡률은 원의 반지름의 역수가 되기 때문에 획을 이루는 세 점으로 구성된 곡선에서 세 점이 일직선상에 있지 않으면 곡률은 0 이상의 값을 가지게 된다. 직선에 가까운 점들로 구성된 곡선은 0에 가까운 값을 가지게 되며 곡선의 방향이 바뀌면서 꺾기게 되면 곡률은 급격히 증가하게 된다. 따라서 입력된 획들을 하나의 곡선으로 생각할 때 획을

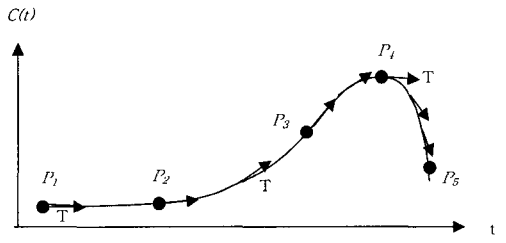


그림 5 곡선에 대한 곡률 값의 변화

구성하는 점들에 대한 곡률을 계산하면 기본획으로 분리시킬 위치를 찾을 수 있다. 획을 구성하는 임의의 점에 대해 임계치 이상의 곡률을 가지면 그 점에서 획을 분리하였다. 임계치는 아래와 같은 값을 사용하였다.

임계치=획 전체의 곡률 평균
+획 전체의 곡률 표준 편차 값/2

그림 6은 세 개의 자소에 대한 곡률 변화량을 보여준다. 획에서 수평이나 수직, 사선 등 직선 모양을 나타내는 부분의 곡률은 $10\times E^{-10}$이하의 값을 가졌다. 이러한 값들을 포함하여 $10\times E^{-03}$이하의 값들은 모두 0으로 처리하였다. 그림 6에 있는 세 개의 자소는 타원 안에 있는 점에서 획의 분리가 이루어졌으며, 분리가 이루어진 획에서의 위치는 (a), (b), (c)에서 점으로 표시된 곳이다. 타원 A의 경우 임계치 이상의 곡률을 가지는 연속적인 세 개의 점이 나타난다. 이러한 경우는 (a)에서 보는 것처럼 완만하게 곡선이 꺽이는 부분에서 발생한다. 이와 같이 두 개 이상의 연속적인 임계치 이상의 값들이 나타날 경우 한 점으로 취급하여 획의 분리가 일어나도록 하였다. 따라서 자소 (a)의 경우 하나의 획이 세 개의 기본획으로, (b)는 세 개의 획이 네 개의 기본획으로, (c)의 경우 두 개의 획이 세 개의 기본획으로 분리된다.

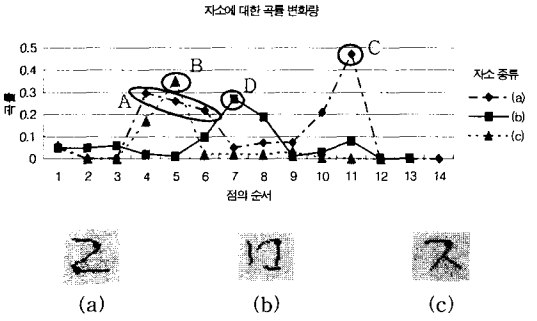


그림 6 자소에 대한 곡률 변화량과 획 분리 위치

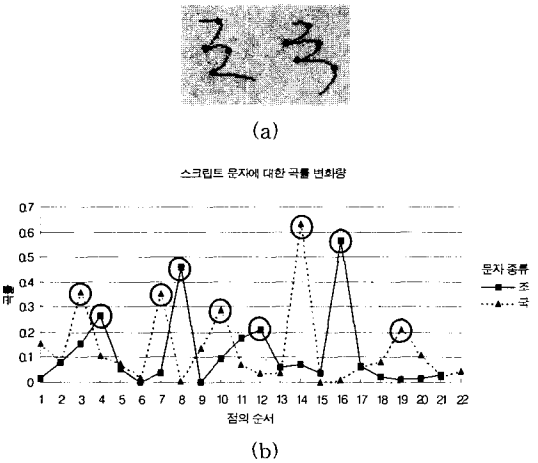


그림 7 스크립트 문자에 대한 곡률 변화량 : (a) 입력 스크립트 (b) 곡률 변화량

그림 7은 스크립트 문자에 대한 곡률 변화량을 보여준다. 문자 "조"의 경우 곡률 평균은 0.1269, 표준 편차 0.154의 값을 문자 "국"의 경우 평균은 0.1266, 표준 편차는 0.148의 값을 가졌다. 평균과 표준 편차를 이용한 임계치 설정에 따라 두 문자 모두 곡률 값 0.20이상일 때 획이 분리되었다. 결과적으로 "조"는 다섯 개의 기본획으로, "국"의 경우 여섯 개의 기본획으로 분리되었다. 그림 (a)의 스크립트에서 "조"의 중간 부분이나 "국"의 마지막 부분처럼 완만하게 휘어지는 "ㄱ" 부분의 곡률은 2.0 내외의 값을 가졌다. 이처럼 "ㄱ"이나 "ㄴ" 형태의 스크립트를 완만한 곡선으로 입력할 때 곡률 변화량은 비교적 낮은 편이기 때문에 필기 형태에 따라 획을 분리시키지 못하는 경우가 발생한다. 따라서 이러한 경우를 고려하여 완만하게 흘려 쓴 "ㄱ"이나 "ㄴ"를 기본획의 종류에 포함시켰다.

3.2 기본획 종류

한글의 기하학적 모양에 따른 곡률 변화량을 실험한 결과를 토대로 본 연구에서는 일곱 개의 기본획을 지정하였다. 더 많은 종류의 기본획을 지정하여 사용할 수도 있다. 실제 문자 인식 시스템에서는 60개 이상의 기본획을 지정하여 사용하고 있다[8,9,10]. 그러나 본 연구는 문자 인식이 아닌 대략적인 잉크 데이터 매칭이라는 것과 휴대용 단말기의 성능을 고려하여 효율적인 처리를 위해 가능한 작은 수의 기본획을 지정하였다. 곡률을 이용하여 한글 흘림 글자를 분리하는 실험을 통해 곡률에 의해 분리된 기본획들은 표 1에서 정의한 일곱 개의 범주 안에 모두 포함되었다.

자소 "ㄱ"은 곡률에 의해 1번과 2번 기본획으로 분리될 수 있고 사용자의 필체에 따라서는 1번과 3번 기본획이나 5번 기본획이 될 수도 있다. 자소 "ㅈ"의 경우 1번, 1번, 3번, 4번 기본획 순서로 분리되거나 2번, 1번, 3번, 4번 기본획 순서로 분리될 수도 있고 또 다른 기본획들로 구성될 수도 있다. 또한 동일한 사용자의 필체라도 입력할 때마다 획의 정보는 달라질 수도 있다. 기본획 5번과 6번의 경우 "ㄴ", "ㄱ", "ㄷ", "ㄹ", "ㅁ", "ㅋ", "ㅌ" 등을 완만하게 흘려 쓴 경우에 발생한다. 7번 기본획은 "ㅇ", "ㅎ"의 자소에서 나타난다.

표 1 기본획의 종류

기본획의 종류	ㅡ	ㅣ	/	\	ㄱ	ㄴ	ㅇ
기본획의 값	1	2	3	4	5	6	7

곡률에 의해 획 분리가 이루어지면 분리된 기본획의 종류를 결정해야 된다. 기본획의 종류를 결정할 때 사용한 정보는 기본획을 구성하는 연속된 점들의 방향 정보와 각도이다. 방향 정보는 RIGHT, LEFT, UP, DOWN의 네 가지 정보만을 사용하였다. 기본획의 종류를 결정하기 위해 기본획을 구성하는 점들의 방향 정보와 각도를 차례로 구한다. 예를 들어, 연속된 방향 정보와 각도가 RIGHT 값을 가지면서 각도의 크기가 30도를 넘지 않을 때 기본획 "ㅡ"가 된다. 기본획 7번의 경우 사용자의 필체에 따라 다양하게 나타날 수 있지만 세 가지 이상의 방향 정보를 모두 포함하는 특징을 가지고 있다. 예를 들어, LEFT, DOWN, RIGHT, UP이나 DOWN, RIGHT, UP 등의 순서로 나타난다. 따라서 이러한 방향 정도를 가질 경우 기본획 7로 결정하고 곡률에 의한 획 분리 작업에서 제외시킨다.

그림 7 (a)의 두 스크립트 문자의 경우 각 문자에 대해 입력된 획은 하나이지만 곡률에 의해 "조"는 다섯 개의 기본획으로 "국"은 여섯 개의 기본획으로 분리된다. 획의 종류는 "조"의 경우 1번, 3번, 1번, 3번, 1번 기본획 순서로, "국"의 경우 1번, 3번, 1번, 3번, 1번, 2번 기본획 순서로 결정된다. 기본획들의 종류를 결정하는 과정이 끝나면 그 결과를 획 특징 벡터에 저장하게 된다. 각 문자에 대한 획 특징 벡터는 다음과 같은 형식을 가진다.

<기본획의 개수, 첫 번째 기본획의 종류, 두 번째 기본획의 종류, ..., 마지막 기본획 종류>

그림 7의 "조"의 경우 획 특징 벡터는 <5, 1, 3, 1, 3, 1>이 되고, "국"의 경우 <6,1,3,1,3,1,2>이 된다.

3.3 매칭 알고리즘

3.3.1 문자 단위 거리값 계산

펜에 의한 데이터 입력이 이루어지면 전처리 과정과 획 분리, 기본획 종류 결정 과정을 거쳐 획 특징 벡터를 생성하여 전자 잉크 데이터 좌표 값 정보와 함께 저장한다. 데이터 입력에서 각 문자를 구분하여 입력하였기 때문에 거리값 계산은 각 문자에 대해 이루어진다. 거리값 계산은 문자에 대한 획 특징 벡터들 사이의 최적의 거리값을 계산하는 문제이다. 따라서 최적화 문제를 풀기 위해 잘 알려진 동적 프로그래밍 기법을 사용하였다[11]. 동적 프로그래밍 기법으로 거리값을 계산하기 위해 다섯 개의 편집 연산(edit operation)- 교환, 삭제, 삽입, 결합, 분리 연산을 사용하였다.

쿼리 데이터를 구성하는 문자와 데이터 베이스내의 대응되는 문자의 획 특징 벡터가 각 각 $(n, p_1, p_2, \ldots, p_n)$, $(m, t_1, t_2, \ldots, t_m)$ 라고 가정하자. 획 특징 벡터에서 n, m은 기본획의 개수를 의미하고 나머지 원소들은 기본획 중 하나의 값을 가지므로 $1 \le p_i, t_i \le 7$ 사이의 값을 가진다. 쿼리 데이터의 i번째 기본획과 데이터 베이스 내 데이터의 j번째 기본획까지의 거리값을 계산하는 재귀식은 식 4와 같다.

$$char\_dist(i,j) = \min \begin{cases} char\_dist(i-1,j) + del(p_i) \\ char\_dist(i,j-1) + ins(t_j) \\ char\_dist(i-1,j-1) + sub(p_i,t_j) \\ char\_dist(j-1,j-2) + split(p_i,t_{j-1}t_j) \\ char\_dist(i-2,j-1) + merge(p_{i-1}p_i,t_j), \\ \quad 1 \le i < n, 1 \le j < m \end{cases} \quad (식4)$$

식 4에서 사용되는 초기값은 다음과 같이 지정한다.

$$\begin{aligned} &char\_dist(0,0) = 0 \\ &char\_dist(i,0) = char\_dist(i-1,0) + del(p_i) \\ &char\_dist(0,j) = char\_dist(0,j-1) + ins(t_j), \\ &\qquad 1 \le i < n, 1 \le j < m \end{aligned} \quad (식5)$$

식 4에서 각 항의 연산이 의미하는 것은 그림 8과 같으며 ①은 삭제 연산($del(p_i)$), ②는 삽입 연산($ins(t_j)$), ③은 교환 연산($sub(p_i,t_j)$), ④는 분리 연산($split(p_i,t_{j-1}t_j)$), ⑤는 결합 연산($merge(p_{i-1}p_i,t_j)$)이 적용되는 예이다. 기본획에 대한 연산 비용은 기본획들 사이의 기하학적인 유사도를 기준으로 만든 비용 행렬(cost matrix)에 미리 정의해 두었다. 결합 연산과 분리 연산은 기본획 종류중 5번획과 6번획이 나타날 때만 적용된다. 쿼리 데이터에서 기본획 5번 "ㄱ"이 발견될 경우 데이터 베이스 획 특징 벡터에서 연속되는 획 종류가 (1,2)이나 (1,3)이면 한글의 기하학적 특징상 쿼리 데이터의 5번 획을 분리하는 것이 유리하므로 분리 연산이 이루어진다. 반대로 데이터 베이스에서 기본획 "ㄱ"이 발견되어지면 쿼리 데이터의 연속되는 두 개의 획을 조사하여 (1,2)나 (1,3)이 존재하면 결합 연산이 이루어진다. 기본획 종류 6번에 대해서도 동일한 규칙이 적용된다.

그림 10은 그림 9에 있는 쿼리 데이터 "난"과 데이터 베이스내의 데이터 "간" 에 대해 편집 연산에 의해 거리값이 계산되는 과정을 보여 준다. 그림 10의 첫 번째 열과 행은 쿼리와 데이터 베이스 문자의 획 특징 벡터

에 있는 기본획 정보들을 나타내는데 '-'표시는 초기값 설정(식 5)에서 삭제와 삽입 연산이 수행된 것을 표시한다. 삭제와 삽입 연산을 위한 비용은 동일하게 15를 사용하였다. 일곱 가지 획들 사이의 연산 비용은 비용 행렬에 정의된 값으로 동일한 획 사이의 연산 비용은 0, 그렇지 않은 경우는 획들 사이의 기하학적인 모양에 따라 연산 비용을 지정하였다. 예를 들어, 기본획 1번과 2번 사이의 연산 비용은 1번과 3번의 연산 비용보다 큰 값을 가지는데 이는 수평선과 수직선보다 수평선과 사선이 더 비슷한 모양을 가지기 때문이다. 각 단계에서 선택된 연산은 타원으로 표시된 것으로 교환, 교환, 분리, 결합 연산 순으로 수행된 것을 알 수 있다. 그림 10에서 '**' 표시된 곳은 임시적으로 선택되었지만 다음 단계에서 분리와 결합 연산이 수행되므로 최종적인 비용에서는 고려되지 않는 연산이다.

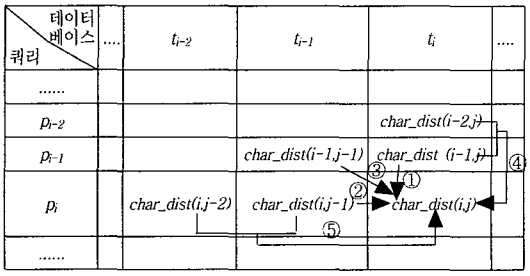


그림 8 편집 연산의 종류

획 특징 벡터 : 5,2,1,6,2,1 획 특징 벡터 : 5,1,2,2,1,6

(a) 쿼리 데이터 (b) 데이터 베이스 데이터

그림 9 거리값 계산에 사용된 데이터

쿼리 획 특징 벡터 \ DB 획 특징 벡터	-	1	2	2	1	6
-	0	15	30	45	60	75
2	15	10(*)	25	40	55	70
1	30	25	20(*)	35	40	63
6	45	40	35	28(**)	20(*)	40
2	60	55	50	35	35	28(**)
1	75	70	65	50	35	20(*)

그림 10 거리값이 계산되는 과정

3.3.2 문자열 단위 거리값 계산

쿼리에 대한 매칭 결과를 얻기 위해서는 전체 문자열에 대한 거리값을 계산해야 한다. 본 연구에서는 데이터베이스내의 모든 데이터에 대한 거리값을 계산하는 오버헤드를 줄이기 위해 두 단계로 매칭 과정을 분리하였다. 먼저 획 특징 벡터 추출 후 생성된 기본획의 개수를 비교하여 데이터 베이스의 데이터 중 30%를 선택하고, 다음 단계에서 선택된 데이터에 대해서만 거리값을 계산하였다. 곡률에 의해 입력 데이터를 기본획으로 분리하고 기본획 5번과 6번을 두 개의 기본획으로 계산해 주면 동일한 필체의 글자에 대한 기본획의 개수 차이는 최대 1개 이하, 평균 0.24개 나는 것으로 실험 결과를 통해 나타났다. 입력된 데이터와 추출된 획 특징 벡터가 다음과 같다고 하자.

쿼리로 입력된 문자열 : $c_1, c_2, \ldots, c_n$

특징 추출 후 생성된 각 문자의 획 특징 벡터 :

$$c_i = \langle n_i, f_{i1}, f_{i2}, \ldots, f_{in_i} \rangle$$

데이터 베이스내 k번째 데이터의 문자열 :

$$c_1^k, c_2^k, \ldots, c_m^k, \quad 1 \le k \le N$$

데이터 베이스의 k번째 각 문자의 획 특징 벡터 :

$$c_i^k = \langle m_i, , f_{i1}^k, f_{i2}^k, \ldots, f_{im_i}^k \rangle$$

쿼리와 데이터 베이스의 문자열을 나타내는 $c_i(1 \le i \le n)$, $c_j^k(1 \le j \le m, 1 \le k \le N)$는 한 개 이상의 획으로 구성된다. 여기서 n의 값은 쿼리 문자열 개수를, k는 데이터 베이스내의 해당 데이터의 인덱스를, m은 데이터베이스 내 k번째 데이터의 문자열 개수를 의미한다. 그리고 N은 데이터베이스의 크기를 나타낸다. 획 특징 벡터내의 원소 f_{ij}, f_{ij}^k는 표 1의 기본획을 의미하는 1에서 7사이의 정수 값을 가지며 n_i, m_i는 기본획 개수를 나타낸다. 첫 번째 단계는 쿼리와 데이터 베이스 내 모든 데이터에 대해 기본획의 개수 차이를 계산하여 데이터 베이스에서 30%의 데이터를 선택한다. 이는 획 특징 벡터 c_i, c_i^k을 이용하여 아래 식과 같이 구한다.

$$diffStroke(query,k) = \prod_{i=1}^{\max\{n,m\}} \left(|n_i - m_i| + \alpha \right) \qquad (식6)$$

n개의 문자로 구성된 쿼리와 m개의 문자로 구성된 데이터 베이스내 k번째 문자열에 대해 대응되는 문자사이의 기본획의 개수의 차이를 구하여 꼽한 값이 $diffStroke(query, k)$이다. 상수 α는 대응되는 문자끼리 기본획의 개수가 같은 경우 전체 값이 0이 되는 것을 막기 위한 값이다.

두 번째 단계에서는 첫 번째 단계에서 선택된 데이터

들에 대해서 식 4에 의해 각 문자에 대한 거리값을 계산한 후 전체 문자열에 대한 거리값을 구한다. 각 문자에 대해 계산된 거리값으로 전체 문자열의 거리값을 구하는 식은 아래와 같다.

$$totalDist(query,k) = \prod_{i=1}^{\max\{n,m\}} (char\_dist(f_{i1}, f_{i1}^{k}) + \beta) \quad (식7)$$

쿼리와 데이터 베이스내 k번째 데이터에 대한 거리값은 대응되는 문자사이의 거리값을 곱한 값이 된다. 상수 β는 한 문자라도 거리값이 0인 경우 전체 거리값이 0이 되는 않도록 하기 위한 상수 값이다.

4. 실험 및 결과

4.1 실험 데이터 및 실험 모델

실험을 위한 데이터는 무작위로 추출한 인명(人名) 집합을 사용하였고 입력 형태는 한글과 한자 스크립트를 허용하였다. 프로그램은 Embedded Visual C++로 작성했으며 실험은 PDA(카시오페아 E-125 모델)에서 수행하였다.

세 가지 요소를 사용하여 매칭률 R을 측정하였으며 그 모델은 다음과 같다.

$$R = M(n, t, r)$$

여기서 n은 데이터 베이스에 있는 인명 데이터의 개수, t는 쿼리를 입력한 횟수, r은 매칭 결과의 순위(rank)를 의미한다. 쿼리 데이터 입력 횟수 t를 모델의 요소로 사용한 것은 원하는 매칭 결과를 얻지 못할 때 다시 검색을 시도하도록 하기 위해서이다. $M(300,2,1)$은 300개의 인명 데이터를 가진 데이터 베이스에서 찾고자 하는 쿼리를 두 번 입력했을 때 원하는 결과를 1순위로 찾아줄 비율을 의미한다.

실험 모델에서 가장 이상적인 결과는 찾고자 하는 쿼리를 한번 입력했을 때 1순위로 찾아주는 것이다. 하지만 어떤 사람도 동일한 글자를 두 번 이상 쓸 때 똑같이 쓸 수 없다. 이런 펜글씨의 특징으로 인해 대략적인 매칭이 이루어지며 따라서 항상 1순위로 쿼리를 매칭해 준다고 보장할 수 없다. 구현한 시스템은 한 페이지에 4순위까지의 매칭 결과를 보여 주므로 찾고자 하는 데이터가 1순위에서 4순위까지의 순서를 가지면 매칭이 성공적으로 이루어졌다고 여긴다. 데이터 베이스 내의 k번째 인명 데이터에 대한 매칭 성공 여부는 아래와 같다.

$$Hit(k,1) = \begin{cases} 1, & if\ match\ rank\ is\ from\ 1\ to\ 4\ for\ k \\ 0, & otherwise \end{cases} \quad (식\ 8)$$

$$Hit(k,t) = Hit(k,1) \vee Hit(k,2) \vee \cdots \vee Hit(k,t) \quad (식\ 9)$$

식 8에서 $Hit(k, 1)$의 값이 0일 때는 매칭이 실패한 경우이므로 원하는 데이터를 찾기 위해 쿼리를 다시 입력할 수 있다. 동일한 쿼리 데이터를 찾기 위해 t번 입력한다면 이때의 매칭 성공 여부는 식 9와 같다.

4.2 데이터의 개수와 매칭률

데이터 베이스의 크기에 따른 매칭률을 실험하기 위해 데이터 개수가 50, 100, 130, 150, 200, 250, 300인 데이터 집합을 준비하였다. 모든 데이터에 대해 실험하기 힘들므로 데이터 집합에서 각각 30%, 50%에 해당하는 샘플을 랜덤하게 추출하여 세 개의 실험 데이터 집합을 만들었다.

표 2는 세 개의 데이터 집합에 대한 매칭률을 보여주며 그림 11은 샘플 크기가 전체 데이터의 30%일 때의 평균 매칭률을 보여준다. 실험 모델에서 t값이 1일 때와 3일 때, 즉 찾고자 하는 쿼리를 한번 입력했을 때와 매칭이 실패했을 경우 세 번까지 동일한 쿼리를 입력했을 때 나타난 매칭 결과를 보여 준다. 데이터 베이스의 개수가 100개 이하일 경우 t의 값에 관계없이 100%의 매칭률을 보여 주었다. 하지만 개수가 많아지면 매칭률이 조금씩 떨어지는 것을 볼 수 있다. 매칭률이 t=1일 때와 t=3일 때를 비교해 보면 평균적으로 약 1% 차이가 났다. 300개의 데이터 집합을 기준으로 볼 때 한 번 쿼리를 입력했을 때 매칭이 성공할 비율은 96.3%이고 매칭이 실패할 경우 세 번까지 쿼리를 입력하면 97.7%로 향상되었다. 이는 입력할 때마다 조금씩 달라지는 펜 데이터의 특징 때문이다.

표 2 세 종류의 데이터 집합에 대한 매칭률과 평균 매칭률

샘플 종류 \ 데이터 크기		50	100	130	150	200	250	300
t=1	데이터 1	100	100	100	99	98	98	97
	데이터 2	100	100	98	98	97	96	96
	데이터 3	100	100	100	98	99	97	96
	평균	100	100	99.3	98.3	98	97	96.3
t=3	데이터 1	100	100	100	100	100	99	98
	데이터 2	100	100	100	99	97	97	96
	데이터 3	100	100	100	100	100	98	99
	평균	100	100	100	99.7	98.7	98	97.7

표 2의 데이터 집합 종류에 따른 매칭률을 비교해 보면 t=1일 때 데이터 2인 경우가 데이터 1인 경우에 비해 매칭률이 상대적으로 떨어진다. 이것은 샘플로 선택된 데이터 중에서 사용자의 일반적인 필체와 다른 데이터

를 상대적으로 많이 포함하고 있기 때문이다. 따라서 데이터 베이스에 데이터들을 저장할 때 사용자의 일반적인 필기 형태로 주의를 기울여 입력한다면 매칭률이 그림 11에서 보여주는 것보다 더 높아질 것이다. 그림 12은 샘플 크기가 50%일 때의 결과이다. 샘플 크기가 30%인 데이터와 비교해 볼 때 매칭률은 차이가 나지 않는다. 이것은 표본의 크기가 매칭률에 영향을 미치지 않음을 보여준다. 따라서 전체 데이터에 대한 매칭률이 샘플 데이터에 대한 매칭률과 비슷할 것으로 짐작할 수 있다.

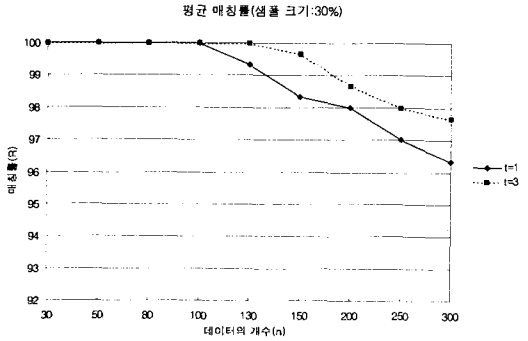


그림 11 데이터 개수와 쿼리 입력 횟수에 따른 평균 매칭률(샘플 크기 : 30%)

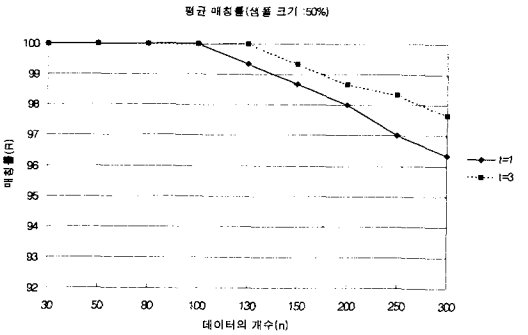


그림 12 데이터 개수와 쿼리 입력 횟수에 따른 평균 매칭률(샘플 크기: 50%)

4.3 한자 혼합 데이터의 매칭률

한글과 한자의 인명을 혼합하여 사용하는 경우를 위해 전체 데이터에서 한자 인명 데이터가 차지하는 비율이 50%가 되도록 만든 데이터 집합에 대한 매칭 실험을 수행해 보았다. 실험은 한글 데이터처럼 전체 데이터에서 30%, 50%의 크기의 표본 데이터를 임의로 선택하여 수행하였다. 그림 13의 전체적인 매칭률을 볼 때 한글만으로 구성된 데이터 집합에 비해 떨어진다. 이것은 한자 데이터를 쿼리로 주었을 경우 매칭 성공 비율이 떨어지기 때문인데 한글과 다른 한자 고유의 기하학적인 모양으로 인해 표 1에서 정의한 기본획으로 커버하지 못하는 획들이 한자에 존재하기 때문이다. 따라서 한글과 다른 한자 고유의 기하학적 특성을 표현하는 획을 기본획에 포함시키면 한자 스크립트의 매칭률이 높아질 것이다.

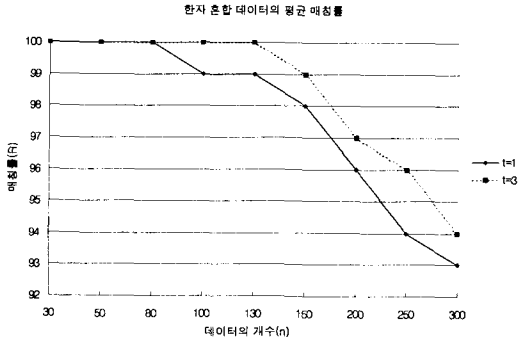


그림 13 한자 혼합 데이터의 평균 매칭률

5. 결 론

본 연구에서는 펜 기반 휴대용 컴퓨터에서 펜으로 입력한 스크립트 데이터를 온라인 문자 인식 단계를 거치지 않고 전자 잉크 형태로 저장 및 검색하기 위한 매칭 알고리즘을 제안하고 구현하였다. 제안된 알고리즘은 전처리 과정과 함께 획의 곡률을 계산하여 기본획으로 분리하는 과정을 거친다. 그리고 기본획 종류 결정에 의해 획 특징 벡터를 생성하고 이를 이용한 동적 프로그래밍 기법에 의해 거리값을 계산한다. 거리값을 계산할 때 사용한 편집 연산은 삭제, 삽입, 교환 연산 외 한글의 특징을 고려한 결합, 분리 연산을 사용하였다.

실험 결과를 통해 보여준 매칭 알고리즘의 장점은 다음과 같다. 첫째로 제안된 매칭 알고리즘의 실용성이다. 일반적으로 휴대폰이나 휴대용 컴퓨터의 주소록에 유지하는 인명 데이터의 개수는 보통 300개 내외이다. 제시한 매칭 알고리즘은 데이터 베이스의 개수가 300개일 때에도 한글 스크립트인 경우 98%에 가까운 매칭률을 보여 주었다. 둘째 빠른 검색 속도이다. 데이터 베이스의 개수가 300개일 때 평균 매칭 속도는 0.8초였다. 이는 데스크 탑 컴퓨터와 비교하여 하드웨어적인 제약을 가지고 있는 PDA의 환경을 고려할 때 매우 빠른 속도라 할 수 있다. 셋째는 필기자 종속 매칭 알고리즘이 가

지는 장점으로써 필체가 다른 사람에 대해 보안 기능을 제공해 준다는 것이다. 필기자 종속 매칭의 경우 동일한 문자에 대해서도 필체가 다른 경우 다른 모양의 패턴으로 인식한다. 따라서 필체가 다른 사람이 사용할 경우 원하는 데이터를 찾아줄 확률이 낮아지기 때문에 개인 정보에 대한 보안 기능을 제공해 줄 수 있다.

향후 연구 과제는 첫째, 데이터 베이스에 없는 데이터를 쿼리로 주었을 때 이를 처리하기 위한 방법을 연구하는 것이다. 현재는 가장 유사한 데이터를 매칭 결과로 찾아준다. 둘째, 한자 스크립트의 매칭률을 향상시키기 위해 한자의 기하학적인 특성을 반영한 기본획 집합을 정의하고 매칭 알고리즘에 반영하고자 한다. 마지막으로 스크립트를 입력할 때 문자를 구분하지 않고 입력할 때 매칭률을 향상시키기 위한 방법을 고안하는 것이다.

참 고 문 헌

[1] W. Aref, D. Barbara, D. Lopresti, and A. Tomkins, "Ink as a first-class datatype in multimedia databases," Multimedia Database, Springer-Verlag, 1995.

[2] Walid G. Aref, Ibrahim Kamel, and Daniel P. Lopresti, "On Handling Electronic Ink," ACM Computing Surveys, Vol. 27, No 4. pp. 564-567, 1995.

[3] Lopresti, D. Snd Tomkins, A., "On the searchability of electronic ink," In Proceedings of the International Workshop Front. in Handwriting Recognition, pp. 156-165, 1994.

[4] D.P, Lopresti and A. Tomkins, "Approximate matching of hand-drawn pictogram," In proceeding of the Third International Workshop on Frontiers in Handwriting Recognition, pp. 102-111, 1993

[5] Walid Aref and Daniel Barbara, "Supporting Electronic Ink Database," Information Systems, Vol. 24, No. 4, pp. 303-326, 1999.

[6] Ibrahim Kamel and Daniel Barbara, "Retrieving Electronic Ink by Content," IEEE Proceedings of International Workshop on Multimedia Database Management Systems, pp. 54-61, 1996.

[7] 마이크로소프트웨어 매거진 2001년 4월호.

[8] 권오성, 권영빈, "스트링 정합 방법에 기반한 온라인 자소 인식", 한국정보과학회 논문지 제21권 제5호, pp. 750-755, 1994.

[9] 신봉기, 김진형, "은닉 마르코프 모델 네트워크에 의한 온라인 흘림 필기 한글 인식", 한국정보과학회 논문지 제21권 제 9호, pp.1737-1745, 1994.

[10] Hang Joon Kim, Pyeoung Kee Kim, "On-line recognition of cursive Korean characters using set of extended primitive strokes and fuzzy functions," Pattern Recognition Letters, Vol. 17, pp19-28, 1995.

[11] Sara Baase, *Computer Algorithm*, pp.232-247, Addison-Wesley Publishing Company, 1989.

조 미 경

1990년 부산대학교 전산통계학과 졸업(학사). 1992년 부산대학교 계산통계학과 졸업(이학석사). 1998년 부산대학교 전자계산학과 졸업(이학박사). 1999년 3월 ~ 2001년 8월 신라대학교 강의전담교수. 2001년 9월 ~ 2002년 8월 부산대학교 컴퓨터및정보통신연구소 기금교수. 2002년 9월 ~ 현재 동명정보대학교 정보공학부 전임강사. 관심분야는 모바일 컴퓨팅, 그래픽스, 지리정보시스템

조 환 규

1984년 서울대학교 계산통계학과 졸업(학사). 1986년 한국과학기술원 전자계산학과 졸업(공학석사). 1990년 한국과학기술원 전자계산학과 졸업(공학박사). 1990년 ~ 현재 부산대학교 전기전자정보컴퓨터공학부 교수. 관심분야는 그래프 이론, 생물정보학, 그래픽스