온/오프라인 통합 문자 인식 시스템

이장길○ 김동현 조현태 박재화
중앙대학교 휴먼인터페이스 연구실
windfencer@naver.com dhkim@hil.cau.ac.kr htcho@hil.cau.ac.kr jaehwa@cau.ac.kr

Unified Character Recognition System

Janggil Lee○ donghyun Kim hyuntae Cho jaehwa Park
Human Interface Lab in Chung-Ang University

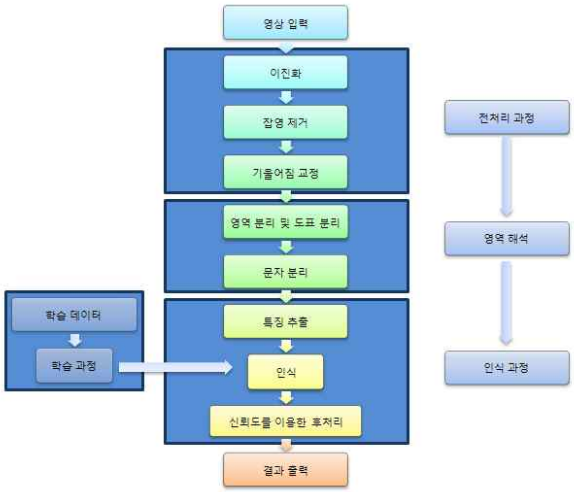
요 약

본 논문에서는 필기체 문자 인식을 위하여 기존의 기술을 검토, 분석 및 새로운 방안을 제안하여 온라인 문자 입력 방식과 오프라인 문자 입력 방식의 적절한 통합화 과정을 제안하고 이를 통하여 온/오프라인 모두에 적용 가능한 통합형 필기체 문자 인식 시스템을 제안하고자 한다. 이를 위해 온/오프라인 문자 입력 방식의 차이를 줄이도록 전처리 과정을 시행하고 데이터 저장 효율을 위해 베지어 곡선 근사화 작업을 진행한다. 또한 이러한 통합 데이터를 처리할 수 있는 인식 엔진을 제안하고자 한다.

I. 서 론

문자 인식기술은 입력 방법에 따라 온라인(On-line)과 오프라인(Off-line)으로 나눌 수 있다. 온라인 인식은 사용자가 전자펜을 이용하여 필기하는 과정을 실시간으로 입력받거나, 터치스크린이나 타블렛 위에 문자를 쓰면, 그 과정을 실시간으로 입력받아 인식하는 방법으로 입력 패턴의 시간적 정보, 위치상의 공간적 정보, 압력 정보 등을 분석하여 인식한다. 오프라인 인식은 문서나 우편봉투에 인쇄되어있거나 쓰인 문자 또는 자동차 번호판에 포함된 문자를 카메라나 스캐너 등의 입력 장치를 이용하여 이미지 파일로 변환한 후에 그 내용을 인식하게 하는 과정을 뜻한다. 다시 인식하고자하는 문자의 대상에 따라 오프라인 인식은 필기체 인식과 인쇄체 인식으로 분류된다. 최근에는 모바일 기기를 이용한 문자 인식기술이 각광을 받고 있다. 모바일 기기에는 터치스크린을 사용하고, 카메라도 장착되어 있기 때문에, 터치방식을 이용한 온라인 인식과 카메라를 이용한 오프라인 인식, 두 가지의 문자 인식 방법이 각각 사용되고 있으며, 인식률과 속도 향상을 위한 노력이 가속화되고 있다.

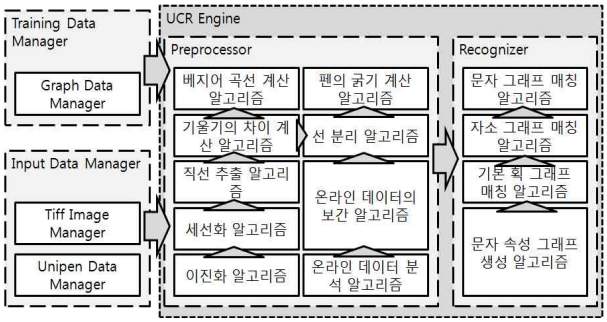
문자 인식의 과정으로는 일반적으로 크게 전처리, 영역 해석, 인식의 세단계로 나눌 수 있다. 문자인식을 위한 이미지는 전처리를 통해 인식하기 좋은 형태로 변환된다. 이 과정에서 불필요한 정보를 제거하고 이진화 변환을 수행한다. 특히 필기체는 글자의 모양이 기울어져 있는 경우가 많으므로 기울기 보정 처리가 필수적이며 문자의 정확한 인식을 위하여, 적절한 크기로 확대 및 축소하는 연산 또한 포함한다.



<그림 1> 문자 인식 단계

2. 온/오프라인 통합 인식 시스템

2.1 UCR 시스템의 내부 구조



<그림 2> 문자 인식 시스템의 구조 및 흐름

UCR 시스템은 위의 그림과 같이 온/오프라인 데이터를 처리하는 모듈과 학습 데이터를 시스템으로부터 읽어 들여 알맞게 처리하는 모듈, 문자를 인식하는 모듈의 구조로 이루어져 있다. 온/오프라인 데이터를 처리하는 모듈에서 오프라인 데이터는 주로 TIFF이미지를 디코딩하여 읽어 들이고 온라인 데이터는 유니펜 데이터를 파싱하여 읽어 들인다. 학습 데이터를 읽어 들이는 모듈은 문자 그래프의 매칭을 위하여 미리 학습된 데이터를 저장하고 이를 필요한 시점에 읽어 들여 문자 인식의 기반 데이터를 관리하는 역할을 담당한다. 마지막으로 UCR 엔진은 문자를 판독하기 위해서 입력 데이터를 전처리하는 과정과 가공된 데이터를 바탕으로 문자를 인식하는 두 부분으로 크게 나눌 수 있다. 오프라인 데이터는 이진화 모듈과 세선화 모듈, 직선 추출 모듈을 거치고 온라인 데이터는 온라인데이터 분석 모듈, 온라인 데이터의 보간 모듈을 거치게 된다. 이렇게 양쪽의 데이터가 가공되어 유사한 형식을 가지게 되며, 그 다음으로 기울기의 차이 계산 모듈과 세선화된 선의 분리 모듈, 베지어 곡선 계산 모듈, 펜의 굵기 계산 모듈을 통하여 문자의 인식에 적합한 데이터가 된다. 그 다음 문자 속성그래프 생성 모듈과 기본 획 그래프 매칭 모듈, 자소 그래프 매칭 모듈, 문자 그래프 매칭 모듈을 거쳐 문자를 인식하게 된다.

2.2 이진화

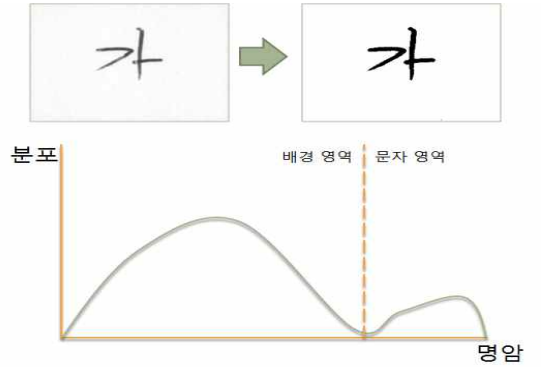
입력받은 문자 이미지는 광학 장치의 성능 한계및 다양한 외부 간섭의 영향으로 노이즈 성분이 생겨 배경과 문자 영역을 구분하기 힘들어진다.

이러한 문제점을 극복하여 어떤 위치에 문자가 존재하고 어떤 위치에 배경이 존재하는 가를 파악하기 위하여 이진화 과정을 수행한다. 이진화가 수행되고 나면 흰색의 바탕과 검은색의 문자만이 남게 되며, 이를 통해 어떤 위치에 문자가 존재하는가를 쉽게 알 수 있다.

본 시스템에서는 문자인식에 가장 적합한 Otsu's Algorithm을 사용하여 이진화를 진행하였다. 이 알고리즘은 문자 영역과 배경 영역이 쌍봉형의 분포를 이룬다고 가정한다. 그리고 이 가정이 성립하게 되면 명암의 분포의 중간지점에 계곡과 같이 화소의 수가 적은 명암값이 있는데, 이 점을 임계값으로 삼아 이진화를 수행하게 된다.

$$F(t) = (Q_1 \times V_1) + (Q_2 + V_2) \quad \cdots\cdots (1)$$

이 공식에서 F(t)는 비용함수이며, Q1과 Q2는 각각 전체 이미지에서 문자 영역과 배경 영역에 해당하는 픽셀이 나타날 확률이다. 또한 V1과 V2는 각각 문자 영역과 배경 영역의 픽셀들의 그레이값의 분산이다. 그렇기 때문에 이 식을 만족하는 값을 구하여 임계값을 결정하고, 이를 통해서 이진화를 수행하면 문자 영역과 배경 영역을 구분할 수 있다.

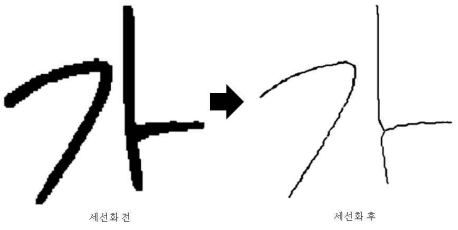


<그림 3> 문자 이미지의 이진화 결과와 알고리즘

2.3 세선화

이진화 된 문자 이미지는 온라인 데이터와는 다르게 필기에 사용한 도구에 따라 서로 다른 굵기를 가지게 된다. 이때 문자의 살을 이루는 굵기 성분은 문자를 인식하는 데에는 불필요한 성분이므로 문자의 구성 뼈대를 추출하여 주변의 불필요한 정보를 제거하고 정보의 양이 최소화 되도록 압축하도록 세선화 과정을 거친다..

본 연구에서는 세선화 알고리즘으로 Zhang과 Suen의 세선화 알고리즘을 사용한다. 이 알고리즘은 대표적인 병렬적 처리 알고리즘중의 하나이다. 알고리즘의 수행 과정에서 두 개의 부분 반복을 갖는데, 하나는 픽셀 l(i,j)가 다음 4가지 조건이 모두 True가 될 때, 삭제를 위하여 마크되거나 삭제되는 것이다. 그 조건의 첫 번째는 연결된 점이 최소한 하나는 존재하는 것이고, 두 번째는 2개에서 6개 사이의 점이 연결되어 있는 것이다. 세 번째는 현재의 점이 i, j의 위치에 있고 해당 이미지에서 점의 색상이 I(i, j)라고 할 때, I(i,j+1), I(i-1,j) 그리고 I(i,j-1)의 색상이 하얀색인 것이다. 마지막 네 번째 조건은 I(i-1,j), I(i+1,j) 그리고 I(i,j-1)의 색상이 하얀색인 것이다. 두 번째 부분 반복은 첫 번째의 부분 반복과 세 번째와 네 번째 조건이 다르다. 세 번째 조건은 I(i-1,j), I(i,j+1) 그리고 I(i+1,j)의 색상이 하얀색인 것이고, 네 번째 조건은 I(i,j+1), I(i+1,j) 그리고 I(i,j-1)의 색상이 하얀색인 것이다. 위의 알고리즘을 반복적으로 수행한 뒤, 더 이상 삭제할 픽셀이 없어지게 되면 알고리즘의 수행을 종료한다. 그렇게 하면 세선화된 이미지를 얻을 수 있다.

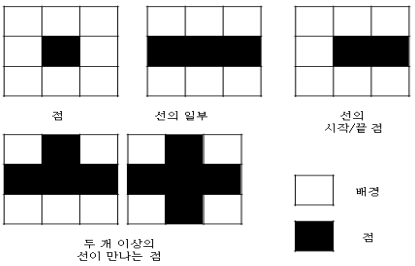


<그림 4>문자 '가'의 세선화 결과

2.4 세선화 된 이미지에서 선의 추출

세선화가 이루어 진 이미지는 뼈대만 남아 있기 때문에 마치 온라인 필기 데이터와 같은 모양으로 보인다. 하지만 이미지이기 때문에 어느 위치가 선의 시작이고 어느 위치가 선의 끝인지가 확실하지 않다. 또한 선이 아닌 점도 판단할 수 있어야 하며, 두 선이 연결되어 있는 지점을 정확하게 판단하고 분할할 수 있는 알고리즘이 필요하다.

세선화 된 이미지를 선으로 추출하기 위해서는 선의 시작점 및 끝점, 점, 두 개 이상의 선이 만나는 점, 선을 구성하는 점을 구분할 수 있어야 한다. 이러한 구분은 한 점의 주위로 주변에 몇 개의 점이 있는지를 세어보면 쉽게 알 수 있다.

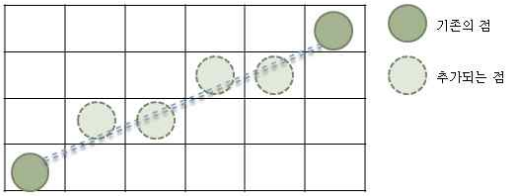


<그림 5> 이웃하는 점의 개수에 따른 분류

2.5 온라인 선의 직선 보간

온라인 데이터의 경우 오프라인 데이터와는 다르게 각 선과 선의 구분이 명확하다. 또한 자료가 입력된 순서대로 들어있기 때문에, 시간 정보를 문자 인식기에서 사용할 수 있다. 하지만 이러한 온라인 데이터도 터치스크린이나 입력장치의 특성에 의해서 입력된 좌표가 음의 값을 가지기도 하고 점과 점사이의 간격이 시스템에 따라 다르다. 그렇기 때문에 온라인 문자 데이터를 오프라인 문자 데이터와 유사하게 만들기 위한 작업이 필요하다.

본 논문에서는 온라인 데이터의 각 점들 사이를 선형으로 보간하는 알고리즘을 이용하여 온라인 문자 데이터를 오프라인 문자 데이터와 유사하게 만든다. 우선 보간을 수행하기 전에 온라인 데이터의 점들을 모두 포함하는 사각형을 계산한다. 계산된 사각형의 x, y방향 중에서 음의 값을 가지는 데이터가 존재하면, 음의 값을 가지는 데이터가 존재하지 않을 때까지 양의 방향으로 쉬프트 연산을 수행한다. 위와 같은 과정을 거쳐서 음의 좌표를 제거한 다음 점과 점 사이를 선형으로 보간해야 한다. 이 과정을 위해서는 두 점의 x좌표의 차이를 Δx라 하고 두 점의 y좌표의 차이를 Δy라고 한다. 이때 Δx가 음의 값을 가지게 되면, Δx가 양의 값을 가질 수 있도록 두 점의 순서를 바꾼다. 이후 Δy를 Δx로 나눈 값의 절댓값을 계산한다. 계산 결과가 0보다 작으면 처음의 점에서 x값을 1씩 증가시키면서 두 점을 지나는 직선의 방정식에 x값을 대입한 뒤, 나오는 y의 값의 좌표에 새로운 점을 추가한다. 계산결과가 0보다 크거나 같으면 처음의 점에서 y의 값을 1씩 증가시키면서 두 점을 지나는 직선의 방정식에 y의 값을 대입한 뒤, 나오는 x의 값의 좌표에 새로운 점을 추가한다. 이 과정을 선상의 모든 점들에 대해서 수행하여 선을 구성하는 두 이웃한 점들이 2픽셀 이상의 간격을 갖지 않도록 한다.

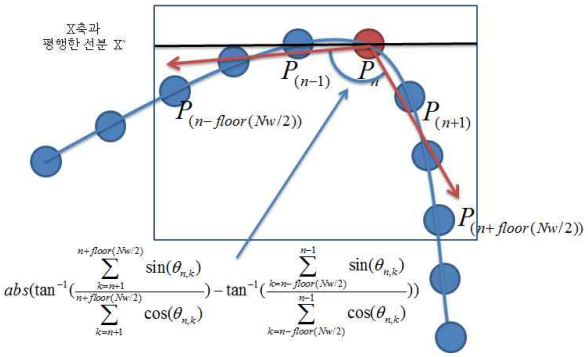


<그림 6> 온라인 문자 데이터상의 점과 점사이의 선형 보간

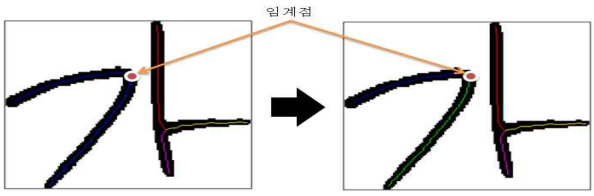
2.6 기울기의 차이 계산

문자 이미지에서 추출된 선은 온라인 문자 데이터의 경우와 같이 선의 시작과 끝이 분명하지 않다. 또한 꺾인 선이 있는 경우는 그 지점을 나누어야 할지, 아니면 그대로 하나의 선으로 보아야 할지를 결정하는 일이 힘들다. 이 문제를 해결하기 위하여 하나의 선의 각 위치에서의 기울기를 계산한 이후 특정 임계값을 기준으로 선을 분리하도록 한다.

본 논문에서는 노이즈의 방해를 적게 하기 위해서 윈도우의 크기(Nw=7)를 정하여 그 윈도우 내부의 범들의 값을 평균하여 기울기를 구하도록 한다.

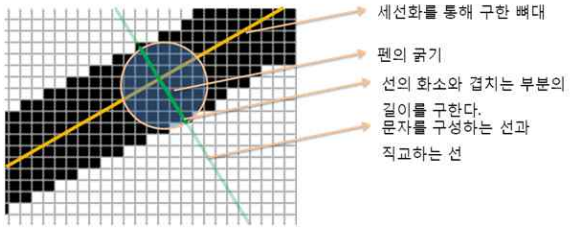


<그림 7> 기울기의 차이 계산



<그림 8> 선의 분리 결과

2.7 펜의 굵기 계산



<그림 2-9> 선의 굵기 계산 과정

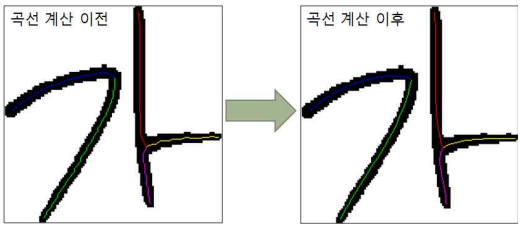
오프라인 문자 데이터는 펜으로 쓰인 글자를 이미지로 만든 것이기 때문에, 온라인 데이터와는 다르게 펜에 따라 굵기가 존재하게 된다. 이러한 펜의 굵기는 세선화과정에서의 삐침선이나 획의 굴곡 등에 영향을 주게 된다. 이러한 이유로 인하여 세선화 결과에서 선의 끝이나 두 개 이상의 선이 만나는 지점, 획이 꺾이는 지점 등에서 삐침선이 발생하게 되는데, 이 삐침선의 크기는 선의 굵기보다 작기 때문에 선의 굵기를 계산하면 쉽게 제거할 수 있다.

2.8 베지어 곡선 계산

입력된 직선의 경우는 세선화를 거쳐서 만들어진 직선이기 때문에 선이 반듯하지 못하고 많은 양의 데이터를 포함하고 있다. 그렇기 때문에 데이터의 양을 줄이고 선을 부드럽게 표현하기 위해서 베지어 곡선으로 만들 필요가 있다. 베지어 곡선은 1970년 베지어에 의해 제안되었으며, 한 개의 베지어 곡선은 다음과 같이 표현된다.

$$R = R(u) = (Rx(u), R(y(u))) \quad \text{.....(2)}$$
$$= (1-u)^3 R_0 + 3(1-u)^2 u R_1 + 3(1-u)u^2 R_2 + u^3 R_3$$
$$(0 \le u \le 1)$$

베지어 곡선은 파라메타 u(0<=u<=1)에 대하여 4벡터(8개의 계수)로 전개되며 이 곡선은 u=0와 u=1일 때 양 끝점 R0와 R3를 통과하며 R0에서 R0로부터 R1으로 향하는 접선과 R3에서 R2로부터 R3으로 향하는 접선을 갖는다.
위의 식을 통하여 베지어 곡선을 추정할 수 있다.

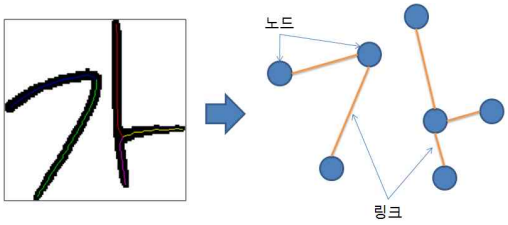


<그림 10> 베지어 곡선 계산 결과

2.9 문자 속성 그래프 생성

입력받은 문자 데이터를 노드와 링크의 연결정보를 이용한 그래프 구조로 바꾸면 확률 및 통계 정보와 구조적 정보를 모두 이용할 수 있다는 점에서 많은 이점이 있다. 그래프의 매칭을 위해서는 입력 받은 문자 데이터를 그래프 구조로 바꾸는 알고리즘이 필요하게 되는데 본 시스템에서는 링크의 길이와 기울기 정보를 속성으로 가지는 방안을 제안한다.

입력받은 각 선들을 구성하는 점들을 모두 포함하는 가장 작은 사각형을 계산한 후 높이를 12등분한 것을 단위 길이 L로 설정 한 후 각 링크로 구성된 선의 길이가 단위길이 L의 몇 배가 되는지를 소서점을 버린값을 이용하여 길이의 속성값으로 이용하고 각 링크의 각도 속성 값은 360도를 16등분 하여 각 링크의 각도가 포함되는 영역의 값을 각도 속성으로 사용한다.

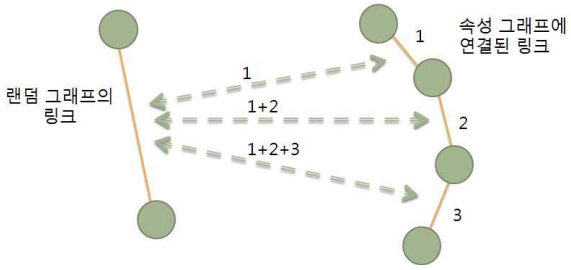


<그림 11> 문자 데이터를 링크와 노드로
변환하여 그래프로 구성한 모습

2.10 기본 획 그래프 매칭

한글은 자소를 조합하여 하나의 글자를 만드는 문자이다. 이러한 자소는 여러 개의 획으로 구성되는데, 이 획들을 기본 획으로 정의하고 이를 조합하여 자소 모델에 매칭할 수 있도록 하면 더더욱 연산 량이 줄어들게 된다.

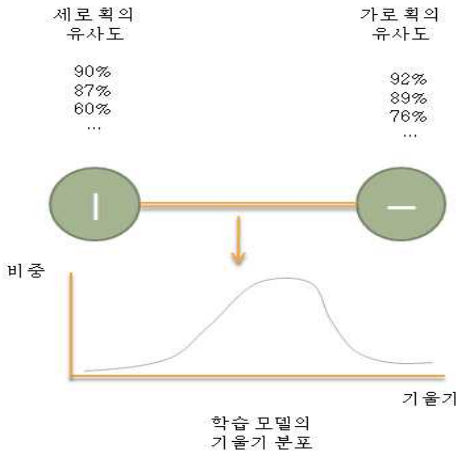
입력받은 데이터에서 생성된 그래프를 속성 그래프라고 정의하고, 미리 학습된 획 모델에서의 그래프를 랜덤 그래프라고 정의한다. 랜덤 그래프의 현재 링크를 A, 노드를 N이라고 한다. 또한 속성 그래프의 현재 링크를 a, 노드를 n이라고 한다. 탐색을 할 때에 N에 연결된 링크A가 1개이면 매칭 결과를 저장하고 랜덤 그래프의 다음 노드를 N으로 하고 속성 그래프의 다음 노드를 n으로 정한다. A나 a가 이미 방문되었으면 빈 매칭 결과를 저장하고 다음 링크를 방문한다. 노드 N과 노드 n에 연결된 링크 A와 a를 순서대로 매칭하고 각각의 매칭 확률을 계산한다. 속성 그래프의 경우 원래 이어진 획이라도 다른 선이 겹쳐지거나 꺾인 점을 분할하는 알고리즘을 거치는 과정에서 여러 개의 링크로 나누어 질 수 있기 때문에, 랜덤 그래프의 링크인 A는 그대로 두고 a와 n을 다음 것으로 변경 매칭한다. 노드 N과 n에 연결된 다른 링크가 없으면 N과 n을 다음 노드로 변경한다. 더 이상 비교할 노드가 없게 되면 매칭 결과를 통합하고 알고리즘을 종료한다.



<그림 12> 랜덤 그래프의 링크와 속성 그래프에 연결된 링크의 매칭

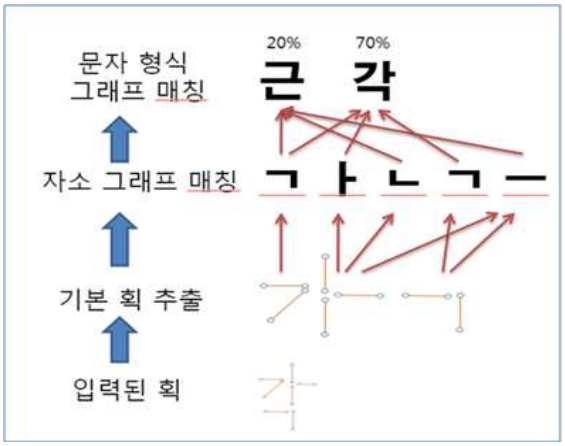
2.11 자소 그래프 매칭

각각의 기본 획들을 매칭하고 나면 그 획들을 이용하여 자소를 매칭하게 된다. 자소는 각각의 자소의 구조를 그래프의 구조로 가지고 자소를 구성하는 획의 위치에 어떠한 획들이 들어가는가를 바탕으로 그래프를 구성한다. 자소에 들어가는 기본 획은 각각 다른 길이와 각도를 가지고 있기 때문에, 입력된 속성 그래프의 각도와 길이가 자소 그래프에서 나타나는 빈도를 이용하여 유사도를 계산할 수 있다. 또한 연결된 획과 획이 이루는 기울기를 바탕으로 자소 그래프의 링크의 각도 속성을 정의한다. 링크의 각도 속성은 입력된 문자에서 연결된 획들이 어떠한 기울기를 이루는지를 계산한 다음, 계산된 각도가 해당 자소 그래프에서 나타나는 빈도를 사용하여 유사도를 계산할 수 있다. 또한 'ㅁ'과 같은 자소의 경우 유사한 속성을 가지는 획이 등장하게 되는데, 원래의 문자그래프에서 같은 획이 여러 개의 노드에 매칭 될 수 있기 때문에 동일한 획이 여러 노드에 매칭되지 않도록 한다.



<그림 13> 자소 그래프의 유사도 계산

입력받은 데이터에서 생성된 그래프를 속성 그래프라고 정의하고, 미리 학습된 획 모델에서의 그래프를 랜덤 그래프라고 정의한다. 이때의 속성 그래프의 노드는 각각의 기본 획이며 노드의 속성으로는 각 기본 획의 매칭 확률을 가진다. 속성 그래프의 링크는 획과 획을 연결하게 되며, 링크의 속성으로는 기울기를 가지게 된다. 이 기울기는 각 획의 중심점을 연결하는 선분의 기울기를 이용하여 계산한다. 랜덤 그래프의 현재 링크를 A, 노드를 N이라고 한다. 또한 속성 그래프의 현재 링크를 a, 노드를 n이라고 한다. 탐색을 할 때에 N에 연결된 링크A가 1개이면 매칭 결과를 저장하고 랜덤 그래프의 다음 노드를 N으로 하고 속성 그래프의 다음 노드를 n으로 정한다. A나 a가 이미 방문되었으면 빈 매칭 결과를 저장하고 다음 링크를 방문한다. 노드 N과 노드 n에 연결된 링크 A와 a를 순서대로 매칭하고 각각의 매칭 확률을 계산한다. 노드 N과 n에 연결된 다른 링크가 없으면 N과 n을 다음 노드로 변경한다. 더 이상 비교할 노드가 없게 되면 매칭 결과를 통합한다. 속성 그래프의 노드들이 같은 선을 사용하는 경우 유사도를 0으로 만들어 후보에서 제외한다.



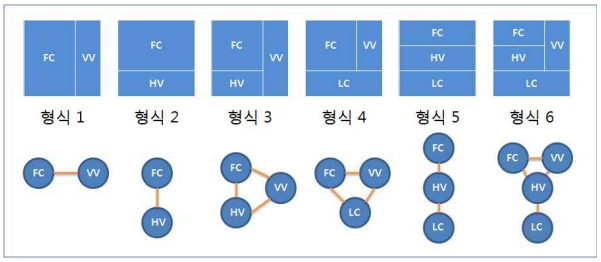
<그림 14> 문자 매칭 알고리즘의 계층도

2.12 문자 그래프 매칭

자소 그래프를 매칭하고 나면 각각의 자소마다 문자에서 존재할 수 있는 확률이 나오게 된다. 그렇다고 해서 높은 확률을 가지는 자소를 조합한다고 입력한 문자가 인식이 되지는 않는다. 자음 모델과 모음 모델이 같은 획을 사용하여 충돌이 생기기도 하고, 'ㅏ'와 같은 모음 글자의 상단은 'ㄴ'으로 인식될 가능성 도 있는데, 반대로 'ㄱ'과 같은 자음은 'ㅓ'와 같은 자음으로 인식될 확률도 존재하게 된다. 이러한 문제를 해결하기 위해서 문자 그래프를 매칭할 때 중복되는 링크를 제거하고 한글 6형식을 바탕으로 각 자모의 위치를 그래프로 만들어서 문자를 매칭함으로써 자음이 모음의 위치에 가거나 모음이 자음의 위치에 가면 유사도가 떨어지도록 하였다. 또한 남은 획이 있을 경우 남은 획의 길이가 길수록 인식률이 낮아지도록 하여 '라'와 같은 글자가 '가'로 인식 되는 일이 없도록 하였다.

입력받은 데이터에서 생성된 그래프를 속성 그래프라고 정의하고, 미리 학습된 획 모델에서의 그래프를 랜덤 그래프라고 정의한다. 이때의 속성 그래프의 노드는 각각의 자소이며 노드의 속성으로는 각 자소의 매칭 확률을 가진다. 속성 그래프의 링크는 자음과 모음을 연결하게 되며, 링크의 속성으로는 기울기를 가지게 된다. 이 기울기는 각 자소의 중심점을 연결하는 선분의 기울기를 이용하여 계산한다. 랜덤 그래프의 현재 링크

를 A, 노드를 N이라고 한다. 또한 속성 그래프의 현재 링크를 a, 노드를 n이라고 한다. 탐색을 할 때에 N에 연결된 링크A가 1개이면 매칭 결과를 저장하고 랜덤 그래프의 다음 노드를 N으로 하고 속성 그래프의 다음 노드를 n으로 정한다. A나 a가 이미 방문되었으면 빈 매칭 결과를 저장하고 다음 링크를 방문한다. 노드 N과 노드 n에 연결된 링크 A와 a를 순서대로 매칭하고 각각의 매칭 확률을 계산한다. 노드 N과 n에 연결된 다른 링크가 없으면 N과 n을 다음 노드로 변경한다. 더 이상 비교할 노드가 없게 되면 매칭 결과를 통합한다. 속성 그래프의 노드들이 같은 선을 사용하는 경우 유사도를 0으로 만들어 후보에서 제외한다. 원래의 문자 그래프에서 매칭에 사용된 속성 그래프의 자소들이 사용하는 선들을 제외하고 남은 선들의 길이가 얼마인지를 계산한 다음, 남은 길이가 길수록 유사도에 패널티를 주어 많은 선들이 남을수록 유사도가 낮도록 판정한다.



<그림 15> 형식에 따른 문자 그래프 모델

3. 결론

본 논문에서는 전처리 과정을 통하여 온라인 문자 입력과 오프라인 문자 입력 데이터를 통합화 한 후 베지어 곡선 추정을 통하여 처리야 하는 데이터의 양을 줄이고 데이터의 질을 향상시킨 후 최종적으로 그래프 매칭을 통하여 온/오프라인 통합 필기체 인식에 적합한 시스템을 제안하였다.

이를 통하여 기존의 온라인 입력 혹은 오프라인 입력과 같이 한쪽에 특성화된 시스템이 아닌 온라인 입력과 오프라인 입력을 동시에 처리할 수 있는 시스템이라는 점에서 요즘 각광받고 있는 다양한 모바일 스마트기기에 적은 비용을 통해 적용이 가능하다는 점에서 많은 이점이 있을 것으로 판단된다.

추후에는 이러한 온/오프라인 통합 인식 시스템을 모바일 기기에 적용시 발생할 수 있는 다양한 문제점들을 극복하는 방안과 더불어 이러한 시스템에 적합한 통합 필기체 입력 사용자 인터페이스를 연구 개발할 계획이다.

<참고문헌>

[1] Charles C. Tappert, Ching Y. Suen, Toru Wakahara, "The State of the Art in On-Line Handwriting Recognition", IEEE Transactions On Pattern Analysis and Machine Intelligence Vol. 12, No.8 August 1990.

[2] Luiz E. Soares de Oliveira, Edouard Lethelier, Flávio Bortolozzi, Robert Sabourin, "A New Segmentation Approach for Handwritten Digits," icpr, vol. 2, pp.2323, 15th International Conference on Pattern Recognition (ICPR'00) - Volume 2, 2000

[3] Due, Anil K. Jain, Torfinn Taxt, "Feature extraction methods for character recognition-A survey", Pattern Recognition, Vol. 29, No. 4, pp. 641-662, 1996

[4] Luiz S. Oliveira, Robert Sabourin, Flávio Bortolozzi, Ching Y. Suen, "Automatic Recognition of Handwritten Numerical Strings: A Recognition and Verification Strategy," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 11, pp. 1438-1454, Nov. 2002

[5] Richard Romero, David s. Touretzky and Robert H Thibadeau, "Optical Chainese Character Recognition Using Probabilistic Neural Networks", Pattern Recognition, vol, 30, no. 8. pp. 1279-1292, 1997