論 文

상태공간탐색을 이용한 한글패턴 인식방법

正會員 金 商 雲* 正會員 李 炳 來** 正會員 朴 圭 泰**

A Recognition Method of HANGEUL Pattern Using a State Space Search

Sang Woon KIM*, Byeong Rae LEE**, Kyu Tae PARK** *Regular Members*

要 約 이 논문에서는 인공지능의 기본적인 문제풀이 기법인 상태공간 탐색을 이용하여 한글을 구성하는 기본자소를 분리하여 인식하는 방법을 제안하였다. 자소분리와 인식과정을 보다 밀접하게 결합하기 위하여 문제를 상태공간에 표현하고, 이 공간을 탐색하여 풀이하였다. 그리고 탐색효율을 향상시키기 위하여 한글의 조합규칙에 입각한 구조정보와 매트릭스 평면에서 각 자소가 갖는 위치정보를 이용하였으며, 컴퓨터실험을 통하여 그 유용성을 확인하였다.

ABSTRACT In this paper, a method of separation and recognition of phonemes from a composite Korean character pattern through a state space search strategy which is a problem solving method in artificial intelligence is proposed. To correlate the separating of phonemes with their recognizing, the problem is represented into the state space, on which a search strategy is performed. For the minimization of search area, the structural information based on the composition rules of Korean characters and the positional information of phonemes in the basic forms are used. And the effectiveness of the approach is shown by a computer simulation.

I. 서 론

한글은 초성, 중성, 종성의 3성음으로 구성되는 조합문자로서 24개의 기본자소(fundamental phonemes)가 기본 6-형식 구조에 의하여 14,364 문자가 체계적으로 조직되며[1], 이 기본자소는 몇개의 기본선소(primitive segments)로 구성된다. 따라서 한글인식의 문제는 문자패턴을 구성하는 기본자소를 분리 추출하여 이를 인식하는 문제가 되며, 기본자소의 인식은 기본선소의 추출과 이들의 연결관계를 식별하는 문제가 된다. 이렇게 볼 때 선소를 추출하여 이들로 구성되는 자소를 분리하는 것이 한글인식의 관건임을 알 수 있다. 그런데 이러한 분리작업과 분리된 자소의 인식은 서로 독립적으로 이루어 질 수 없으며 보다 밀접하게 연계되어 수행되어야만 한다. 따라서 이 논문에서는 패턴의 묘사와 인식이 별도로 수행되는 구문론적 패턴인식(syntactic pattern recognition)

* 明知大學校 工科大學 電子計算學科
Dept. of Computer Science, Myeong Ji University.
** 延世大學校 大學院 電子工學科
Dept. of Electronics, Yonsei University, Seoul 120-749, Korea.
論文番號 : 90-28(接受1989. 2. 13)

기법의 한계[2]를 극복하고, 자소의 분리문제와 분리된 자소의 인식문제를 융통성 있게 결합하기 위하여 상태공간탐색(state space search)을 이용하는 방법을 제안한다.

우선 방향투영을 이용한 방향코딩(direction coding)으로 한글패턴에서 직접 기본선소를 추출한 다음[3], 자소의 분리문제를 선소를 초성자음, 수직모음, 수평모음, 종성자음의 집합으로 분류하는 문제(problem)로 보아 상태공간으로 표현하고, 이 상태공간을 탐색함으로써 문제를 풀이(solving)한다[4]. 이 때 탐색영역을 최소화하기 위하여, 한글을 구성하는 기본자소의 조합규칙에 입각한 구조정보와 기본 6-형식의 구조평면에서 각 자소의 위치정보를 이용한다.

Ⅱ. 상태공간탐색에 의한 문제풀이

상태공간탐색(state space search)에 의한 문제풀이(problem solving)방식은 퍼즐, 게임, 증명문제를 비롯한 많은 지능적인 문제를 풀이하기 위한 방식으로 인공지능(artificial intelligence)에서 문제해결의 기본이 되는 방법이다[5]. 이 방법은 어떠한 공식에 의하여 해를 구하는 것과는 달리 시행착오적인 (trial-and-error) 방법에 의하여 문제를 해결하는 것으로서, 수많은 상태들이 존재하는 공간을 탐색하여 해 (solution)를 찾아내는 방법이다. 여기서 상태공간이라 함은 한 문제로부터 도달 가능한 모든 상태의 집합을 의미하는 것으로서, 주어진 문제를 상태공간 탐색으로 해결하기 위해서는 우선 그 문제를 상태공간에 표현하여야 한다. 상태공간에서의 문제표현은 (S, G, O)의 세 구성요소로 이루어진다. 여기서는 S는 초기상태(initial state)로서 문제풀이를 위한 최초의 상태이고, G는 목표상태 (goal state)로서 초기상태로부터 얻고자 하는 해이며, O는 연산자로서 한 상태를 다른 상태로 변환시키는 일종의 함수라 할 수 있다.

한 상태에 연산자를 적용하여 다음 상태를 얻어내는 과정을 확장(expand)이라고 하며, 확장을 계속함에 따른 상태공간의 형태는 그래프나 트리의 형태를 갖는다. 따라서 상태공간의 어떠한 한 상태를 그래프의 노드로 간주하며, 한 노드 n을 확장하여 노드 ni(i=1,…, N)를 생성시켰을 때, 이들 ni를 노드 n의 후계노드라고 하고, 노드 n을 노드 ni의 부모노드라 한다. 그리고 확장된 노드들의 집합은 리스트 구조로 나타내며 이를 CLOSED라 하고, 생성되어 확장을 기다리는 노드 집합을 리스트 OPEN이라 한다.

탐색과정에서 노드가 확장되는 순서를 결정하는 탐색전략(search strategy)에는 리스트 OPEN을 큐 구조로 운영하여 노드가 생성된 순서대로 확장시키는 넓이 우선(breadth-first)탐색, OPEN을 스택구조로 운영하여 가장 최근에 생성된 노드를 먼저 확장시키는 깊이우선(depth-first) 탐색, 그리고 OPEN에 있는 노드들 중에서 목표상태에 가장 쉽게 도달될 수 있다고 판단되는 노드를 우선하여 확장시키는 경험적(heuristic) 탐색이 있다. 이 전략에서는 OPEN의 노드들에 순서를 정하기 위하여 식(1)의 평가함수(evaluation function) 를 이용한다.

$$f(n)=g(n)+h(n) \tag{1}$$

여기서 f(n)은 노드 n의 유망성을 수치로 표현한 평가값이고, g(n)은 초기상태에서 노드 n까지의 최소비용경로를 찾기 위하여 소모한 경로비용이며, h(n)은 노드 n에서 목표상태에 이르는 최적경로에 대한 비용으로 경험적 정보에 의하여 산출한다. 여기서 g(n)과 h(n)이 f(n)에 미치는 상대적 비중은 식(2)로 나타낼 수 있다.

$$f(n)=(1-\alpha)\cdot g(n)+\alpha\cdot h(n),\ 0\leqq\alpha\leqq1 \tag{2}$$

여기서 α가 매우 커서 $\alpha=1$이 되면 목표상태에 이르기 위하여 경험적 정보에만 의존하는 경우이고, 반대로 α가 매우 작아 $\alpha=0$이면 경험적 정보를 전혀 이용하지 못하는 경우로서, 확장노드의 선택은 단지 현재까지 소모한 비용만을 참조하게 됨으로써 탐색전략은 넓이우선탐색으로 된다. 이와

같이 α=0인 경우의 탐색전략을 특별히 균일비용(uniform-cost) 탐색이라고 하며, 한글인식과 같이 목표상태를 미리·예측할 수 없어서 목표상태에 이르는 최적경로에 대한 경험적 정보(heuristic information)를 얻을 수 없는 경우에 적용할 수 있다. 균일비용탐색에서는 목표상태에 이르는 최소비용(minimal cost) 경로를 찾기 위하여 식(3)의 비용함수(cost function)를 이용한다. 여기서 g(nj)은 초기상태부터 확장한 노드 nj까지의 경로비용으로써, ni는 nj의 부모노드이고, C(ni, nj)는 노드 ni에서 노드 nj에 이르는 경로의 비용함수이다.

$$g(nj)=g(ni)+C(ni, nj) \quad (3)$$

넓이우선탐색과 균일비용탐색 전략은 반드시 해를 제공하지만 이 탐색전략들을 실제문제에 적용하기 위해서는 해를 찾기 위한 탐색노력의 양이 합리적 수준의 것이어야 한다. 즉 탐색효율(search efficiency)을 향상시키기 위하여 탐색영역을 최소화하는 전략이 마련되어야 한다. 이를 위한 방법에는 탐색깊이를 제한하는 방법과 확장노드 수를 제한하는 방법등이 있다[6]. 이 논문에서는 연산자 적용을 제한하고, 최소비용의 노드를 우선 확장시키는 방법으로 생성노드수를 줄여서 탐색 효율을 향상시킨다.

Ⅲ. 상태공간에서의 문제표현

1. 풀이할 문제설정

서론에서 언급한 바와 같이 한글은 초성자음, 수평모음, 수직모음, 종성자음으로 조합됨으로 문자패턴 P는 식(4)와 같이 리스트(list)로 표현할 수 있으며, 이 리스트의 원소(elements) Ci, Vh, Vv, Ct는 각각 초성자음(initial consonant), 수평모음(horizontal vowel), 수직모음(vertical vowel), 종성자음(terminal consonant)을 의미한다. 그런데 이 원소들은 패턴을 구성하는 기본소자를 나타내는 것으로, 이들 역시 기본선소를 원소로 하는 리스트로서 리스트 P의 서브리스트(sublist)가 된다.

$$P=(Ci, Vh, Vv, Ct) \quad (4)$$

서브리스트 Ci, Vh, Vv, Ct의 원소를 ei(i=1,…, K)라 할 때, 원소 ei의 데이터 구조는 선소의 종류와 선소의 연결관계를 이용하여 식(5)와 같이 표현한다.

$$ei=(code, node1, node2),\ i=1,\cdots, K \quad (5)$$

여기서 code는 기본선소의 방향코드로서 사선선소(slash)는 1, 수직선소(vertical) 2, 역사선 선소(back-slash) 3, 수평선소(horizontal) 4, 원형선소(circle) 5의 속성을 갖는다. 그리고 node1과 node2는 각각 선소의 시점과 종점의 마디로서(feature, coordinate)의 두 필드로 구성 된다. 여기서 feature 는 선소의 연결관계를 나타내는 특징으로서 끝점(end), 구석점(corner), 분기점(branch), 교차점(cross)의 속성을 갖게되고, coordinate는 N*N 매트릭스 평면에서 특징점의 위치를 나타내는 좌표이다.

한글패턴을 식(4), (5)의 리스트로 표현할 때, 한글인식 문제는 미지의 패턴을 구성하는 선소들의 리스트인 NP의 원소들을 서브리스트 Ci, Vh, Vv, Ct로 분류한 다음, 이들을 원소로하는 리스트 P를 구하는 문제가 된다. 예를들어 그림 1과 같은 '관'자의 인식문제는 식(6)과같이 설정할 수 있다.

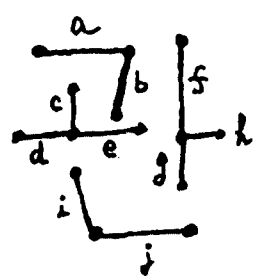


그림 1 인식을 위한 예제문제
Sample Problem for recognition.

NP=(a, b, c, d, e, f, g, h, i, j)
Ci=( )
Vh=( ) (6)
Vv=( )
Ct=( )

2. 초기상태 생성

예제문제를 상태공간에 표현하기 위하여 우선적으로 초기상태 S를 생성하여야 한다. 이를 위하여 문자패턴의 매트릭스 평면에서 좌측 중 하단에서 시작하여 우측으로 진행하는 수평선소를 수평모음의 후보로하고, 또한 우측상단에서 시작하여 하측으로 진행하는 수직선소를 수직모음의 후보로 하여 식(7), (8), (9)와같이 초기상태를 생성한다.

NP=(a, b, c, e, f, g, h, i, j)
Ci=( )
Vh=(d) (7)
Vv=( )
Ct=( )

NP=(a, b, c, d, e, g, h, i, j)
Ci=( )
Vh=( ) (8)
Vv=(f)
Ct=( )

NP=(a, b, c, e, g, h, i, j)
Ci=( )
Vh=(d) (9)
Vv=(f)
Ct=( )

3. 목표상태 설정

예제문제에서 '관'자의 'ㄱ'을 초성자음 리시트인 Ci로, 'ㅗ'를 수평모음 리스트 Vh로, 'ㅏ'를 수직모음 리스트 Vv로, 'ㄴ'을 종성자음 리시트인 Ct로 각각 옮겨놓은 상태에 해당함으로, 목표상태 G는 식(10)과 같이 설정할 수 있다.

NP=( )
Ci=(a, b)
Vh=(c, d, e) (10)
Vv=(f, g, h)
Ct=(i, j)

4. 연산자 정의

예제문제에서 상태공간에 노드를 확장시킨다고 하는 것은 NP의 원소를 서브리스트 Ci, Vh, Vv, Ct중의 한 리스트로 옮기는 것이다. 따라서, 연산자는 어느 리스트로 옮기는가에 따라 다음과 같이 네가지를 정의할 수 있다.

(1) NP-TO-Ci : 리스트 NP의 원소를 서브리스트 Ci로 이동
(2) NP-TO-Vh : 리스트 NP의 원소를 서브리스트 Vh로 이동
(3) NP-TO-Vv : 리스트 NP의 원소를 서브리스트 Vv로 이동
(4) NP-TO-Ct : 리스트 NP의 원소를 서브리스트 Ct로 이동

Ⅳ. 상태공간의 목표탐색

1. 상태노드의 테이터 구조

초기상태에서 출발하여 목표상태를 찾아서 확장을 계속하는 과정은 탐색트리로 추정될 수 있다. 이 탐색트리의 상태노드(state node) SN은 한글문자를 구성하는 각 선소가 Ci, Vh, Vv, Ct의 리스트로 분리된 상태로서 식(11)과 같은 리스트 구조로 묘사한다.

SN=(NP, Ci, Vh, Vv, Ct, Ci-B, Ci-R, Vv-L, Vv-P, Vh-R, Ct-U, COST) (11)

여기서, Ci-B와 Ci-R은 초성의 최하단 i좌표와 최우단 j좌표이고, Vv-L은 수직모음 최좌단 j좌표, Vh-P는 수평모음 수평성분의 i 좌표, Ct-U는 종성자음의 최상단 i좌표, 그리고 COST는 현상태까지 경로비용이다.

2. 탐색영역의 최소화

일단 시작노드가 생성되면 여기에 연산자를 적용하여 노드 확장을 수행한다. 여기서 중요한 것은 가능한한 경로 비용이 적은 노드만을 확장하여 탐색영역을 최소로 하는 것이다. 그런데 한글패턴을 구성하는 한 선소가 분류될 수 있는 부류는 4가지로서 n개의 선소를 갖는 패턴의 경우 4^n개의 분류가능성이 존재하며 이러한 모든 경우를 탐색하는 것은 비효율적이다. 따라서 이 논문에서는, 한글문자의 각 선소에 대하여 N*N 매트릭스에서의 위치정보에 따라 계산된 경로비용이 큰 후계노드와 한글 생성규칙에 위배되는 후계노드는 확장시키지 않는 방법으로 탐색영역을 최소화시킨다. 즉 한 상태노드의 경로비용 (식(11)에서 COST)은 서브리스트 Ci, Vh, Vv, Ct의 선소들이 N*N 크기의 매트릭스 평면에서 어떻게 위치하는가에 따라 산정된다. 예를 들면 초성자음 Ci의 한 선소 e가 'ㅇ'이거나 'ㅁ'일때 이에 대한 비용산출의 예는 그림 2와 같다.

```
if middle-i(e)< N/6 then Cost←0.0
else if middle-i(e) <2*N/6 then COST←0.1
else if middle-i(e) <3*N/6 then COST←0.5
else COST←1.0
```

그림 2 비용산출의 예
Example of COST evaluation.

여기서 middle-i(e)는 선소 e의 중심 i좌표이다. 이와 같이 산정한 각 선소의 COST를 누계한 값이 해당노드의 경로비용으로서, 자소를 분리시키는 척도가 된다. 따라서 노드를 생성할 때마다 식(3)으로 경로비용을 계산하여 OPEN에 등록시키고, 다음 확장시킬 노드는 OPEN에서 최소비용의 노드를 선택하는 균일비용탐색을 수행하여, 넓이우선탐색을 수행할 때 보다 탐색영역을 최소화시킨다.

또한 OPEN의 최소비용 노드를 확장시킬 때 확장 가능한 모든 노드를 생성시키는 대신에 조합규칙에 위배되지 않는 노드만 생성시킴으로서 탐색영역을 최소화시킨다. 예를 들면 리스트 NP의 한 선소 e를 초성자음 Ci로 옮기면 연산자 NP-TO-Ci 가 적용되지 않는 경우는 그림 3과 같다.

```
if bottem(e) >Vh-P
else if right(e) >Vv-L
else if center-i(e)> Cl-U
else if bottom(e) >Ci-B and (HAVE-O(Ci) or
HAVE-M (Ci)
else if up(e)< Ci-B and (primitive(e)=5)
```

그림 3 NP-TO-Ci 연산자가 적용되지 못하는 예
A ill-operation of operator NP-TO-Ci.

여기서 right(e), up(e), center-i(e), bottom(e)는 각각 선소 e의 우단 j좌표와 상 중 하단의 i좌표이고, primitive(e)는 선소 e의 방향코드, HAVE O(Ci)은 현 상태에서 초성자음 Ci에 'ㅇ'이 존재하면 TRUE, 그렇지 않으면 FALSE를 출력하고, HABE-M(Ci)는 Ci에 'ㅁ'이 존재하면 TURE, 그렇지 안으면 FALSE를 출력하는 함수이다.

3. 목표노드 판정과 자소인식

탐색 과정 중 목표노드의 검사는 우선 리스트 Vh와 Vv의 선소들이 다음과 같은 수평모음과 수직모음의 조합규칙에 부합하는가를 검사한다. 검사중 해당 노드가 이들 법칙에 위배 된다면, 그 노드는 이미 조합가능성이 없는 노드이므로 더 이상 확장을 하지 않는다. 그렇지 않으면 NP=ϕ가 될 때까지 확장을 계속하다가 NP=ϕ가 되면 자모의 조합인식을 시도하여 인식이 되면 목표노드로 판정한다.

(1) 수평 모음 조합 규칙 : 수평 성분은 1개이고, 수직 성분은 2개이하이며, 수직 모음이 존재할 경우 수직성분은 1개 이하이다.

(2) 수직 모음 조합 규칙 : 수직 성분은 1개나 2개이고, 수평 성분은 2개 이하이며, 수평모음이 존재할 경우 수평성분은 1개 이하이다.

한글은 모음 위주로 이루어지므로 우선 모음을 조합인식한 다음 자음을 인식한다. 모음 인식은 리스트 Vv와 Vh에서 선소의 갯수와 위치에 따라 수직모음, 수평모음을 조합한 후, 이들을 결합한

형태가 표1에 존재하면 인식된 것으로 하고, 실패할 경우에는 OPEN=ϕ의 조건이 될 때까지 확장을 계속한다. 자음 인식은 모음인식에 성공하였을 때에만 시도하며, 리스트 Ci와 Ct의 원소들이 조합된형태가 각각 표 1에 존재하면 인식에 성공한 것으로 한다.

표 1 기본 자소의 종류
Kinds of Korean phonemes

초성 자음(Ci)	ㄱ, ㄲ, ㄴ, ㄷ, ㄸ, ㄹ, ㅁ, ㅂ, ㅃ, ㅅ, ㅆ, ㅇ, ㅈ, ㅉ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ
수평 모음(Vh)	ㅗ, ㅛ, ㅜ, ㅠ, ㅡ
수직 모음(Vv)	ㅏ, ㅑ, ㅓ, ㅕ, ㅐ, ㅒ, ㅔ, ㅖ
종성 자음(Ct)	ㄱ, ㄲ, ㄴ, ㄷ, ㄹ, ㅁ, ㅂ, ㅅ, ㅆ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ,ㄱㅅ, ㄴㅈ, ㄴㅎ, ㄹㄱ, ㄹㅁ, ㄹㅂ, ㄹㅅ, ㄹㅌ, ㄹㅍ, ㅂㅅ

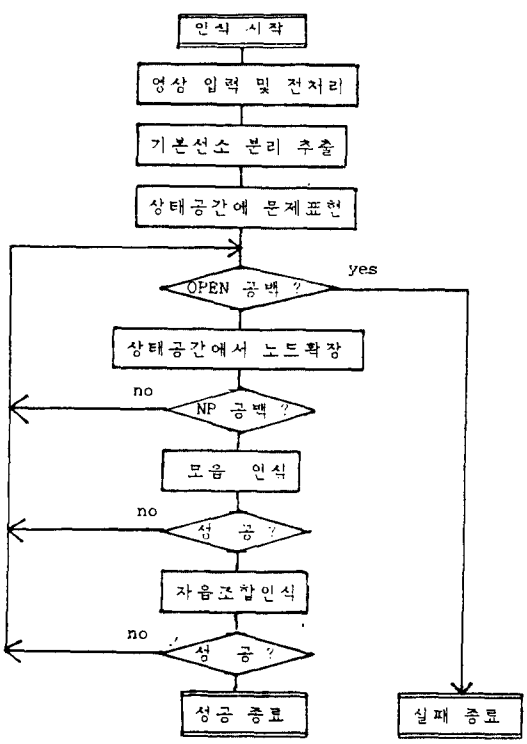


그림 4 전체 처리과정의 흐름도
Overall procedure for simulation

V. 실험 및 결과고찰

1. 실험 시스템 개요

제안한 기법의 전체 처리과정은 그림 4와 같다. 이 알고리즘은 C언어로 작성하여 IBM-PC(AT)와 호환성이 있는 개인용 컴퓨터(CPU : 80286-8, 8MHz)로 실험하였다. 문자영상을 CCD 카메라로 촬영하여 이진화, 정규화 등의 전처리를 수행한 패턴의 예는 그림5(a)와 같다.

2. 기본선소 추출

그림 5(a)를 구성하는 선소는 사선, 수직, 역사선, 수평방향의 직선 선소와 원형선소이다. 여기서 원형선소는 그 기하학적 특성이 직선선소와 다르고, 기본자소중 사용 빈도수가 가장 높은 선소이기 때문에 직선선소와 더불어 기본선소로 정의하였으며 직선선소에 우선하여 추출하였다. 이 논문에서는 원형선소를 추출하기 위하여 내부 경계선(inner boundary)이 폐곡선을 이룬다고 하는 특징과 내부 경계선에서 외부 경계선(outer boundary)으로 투영한 측면도(profile)가 반달(crescent) 모형이 된다는 특징을 이용하였다. 그림 5(a)의 패턴에서 원형선소를 우선 추출한 결과는 그림 5(b)이다. 제안한 원형선소 추출 알고리즘으로 종래에 제안되었던 장축과 단축의 비를 계산하거나, 이웃 화소들과의 곡률(curvature) 계산에 의한 방법으로 해결할 수 없었던 선소를 추출할 수 있었다.

원형선소를 추출하고 남은 패턴에서 직선선소를 추출하는 과정은 세단계로 이루어진다. 첫번째 단계는 방향코딩(direction coding) 단계로서 패턴을 구성하는 모든 흑 화소에 대하여 네 방향으로 투영한 투영값 중에서 최대값을 갖는 방향의 코드를 해당 화소의 코드값으로 정하는 처리로서 각 화소가 어느 방향의 선소에 포함되는가를 나타낸다. 두번째 단계는 화면 분리 단계로서 방향코딩된 패턴을 방향코드에 따라 네개의 방향화면으로 분리하는 처리이다. 그림 5(a) 패턴을 사선(/), 수직(|), 역사선(\), 수평선소(−)로 각각 방향분리한 결과는 그림 5(c), (d), (e), (f)이다.

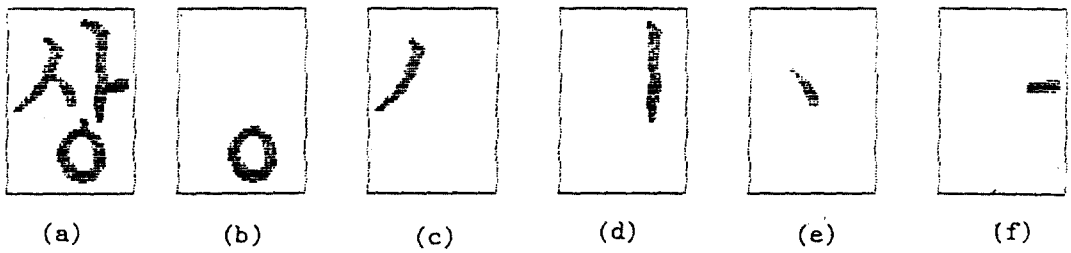


그림 5 전처리결과의 예
Result of preprocessing.

마지막 세번째 단계에서는 각 방향화면의 선소들에게 다른 선소가 접속되는지를 조사하여 접속되었을 경우 접속 위치를 절단하여 기본선소를 추출하는 단계이다. 이 알고리즘을 이용하여 그림 5(a)의 패턴으로 부터 기본선소를 추출하는 과정은 그림 6과 같다. 우선 그림 6(a)와 같이 원형선소를 추출한 다음, 그림 6(b), (c)의 사선선소, (d), (e)의 수직선소, (f)의 역사선 선소, (g)의 수평선소의 순서로 추출하였다. 여기서 기본선소는 편의상 1-화소 폭의 선으로 나타내었다.

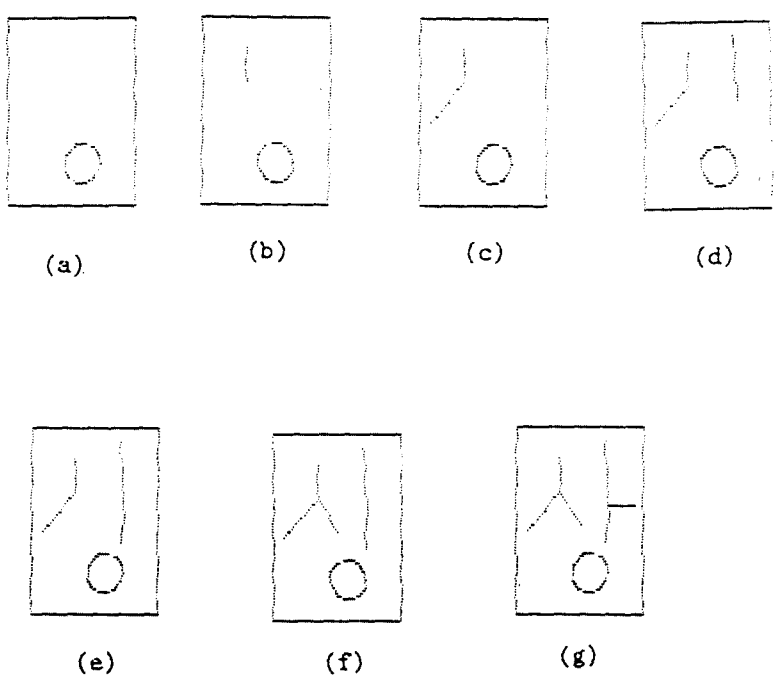


그림 6 기본선소를 추출하는 과정의 예
Procedure of primitive segments extraction

기본선소를 추출한 다음 초기상태를 생성하고 목표상태의 해를 탐색하기 위하여 연산자에 의한 노드확장을 계속한다 그림 7(a)는 그림 5(a) 패턴에서 선소 'ㅣ'를 수직모음의 후보로한 초기상태로서 식(8)에 해당되며, 시각적 이해를 위해 매트릭스로 표시하였다. 그림 7(b)는 '상'자의 'ㅅ'을 초성자음 Ci로, 'ㅏ'를 수직모음 Vv로, 그리고 'ㅇ'을 종성자음 Ct로 분류한 목표상태로, 수평모음은 존재하지 않기 때문에 Vh=ϕ 이다.

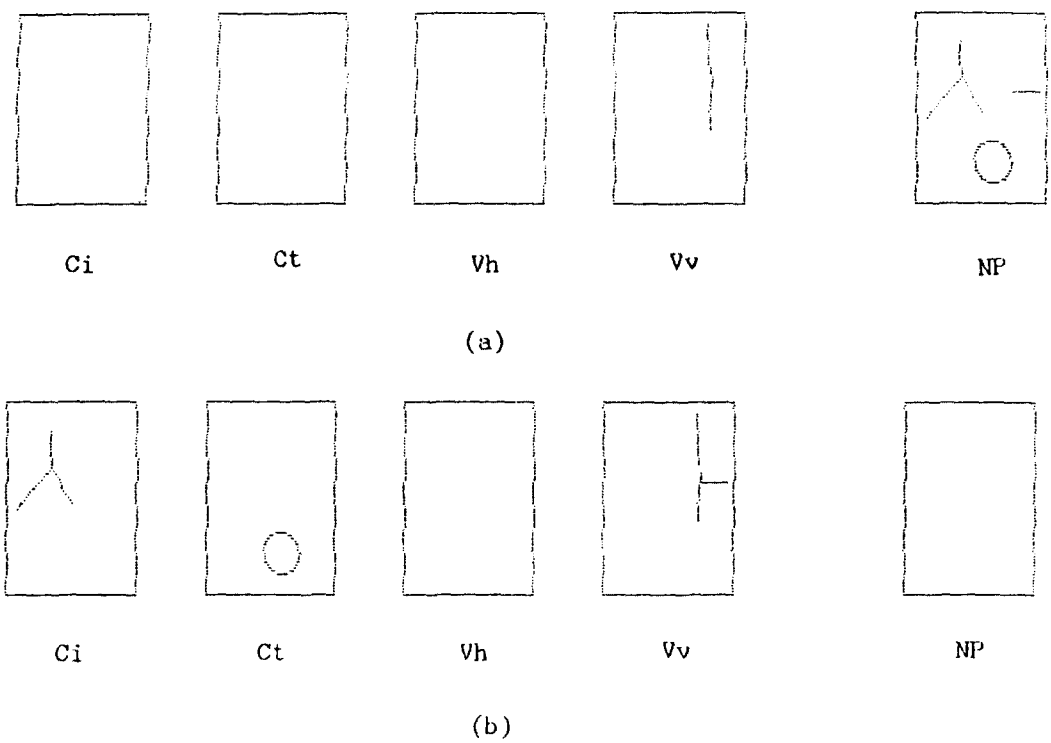


그림 7 초기상태와 목표상태의 노드
Nodes of initial and goal state.

표 2 인식결과
Results of recognition

형식	F1	F2	F3	F4	F5	F6	계(%)
실험문자수	67	106	38	225	364	35	835
인식문자수	66	100	36	209	352	31	794
인식율(%)	98.5	94.3	94.7	92.8	96.7	88.5	94.44

3. 결과에 대한 고찰

상용문자에 대한 객관성을 고려하여 일간신문 12면과 논문 3편에서 취합한 총 93,109문자에서 빈도수가 높은 835 종류의 명조체 문자를 대상으로 실험한 결과 94.4(%)의 인식율을 얻어 제한기법의 유용성을 확인하였다. 기본 형식별(Fi, i=1,…, 6) 인식율은 표2와 같다.

인식에 실패한 실패요인은 다음과 같이 여섯가지 유형으로 분류할 수 있다. 여기서 (1), (2), (3) 요인은 전처리 단계에 관한 것으로 인식에 실패한 41문자중 43.90(%) 이었고, (4), (5), (6) 요인은 조합인식에 관한 것으로 56.10(%)이었다.

(1) 방향코딩 착오에 기인한 경우
(2) 기본선소 추출착오에 기인한 경우
(3) 접속특성 추출착오에 기인한 경우
(4) 탐색영역 제한(151.552KB)에 기인한 경우
(5) 확장노드의 COST산정 착오에 기인한 경우
(6) 자소 조합규칙의 착오에 기인한 경우

전처리 단계의 착오는 사선이나 역사선의 기울기가 일정하지 않은 'ㅅ', 'ㅈ', 'ㅊ' 등의 선소를 포함한 패턴과 직선선소와 직선선소의 끝점이 접속되어 특징이 소멸되는 '녀', '업' 등의 패턴에서 발생하였다. 그리고 인식단계의 착오는 목표상태에 도착하기전에 할당된 기억공간을 모두 소비한 경우와 조합규칙등을 이용한 비용산출이 잘못된 경우이다. 그림 8(a)는 특징점 추출이 잘못되어 '평'이 '펑'으로, 그림 8(b)는 '희'가 '힉'으로 오인식된 경우의 예이고, 그림 8(c)는 '괄'자에서 초성자음이 'ㄱ'로 되는 노드보다 수평모음의 일부와 결합하여 'ㅍ'로 조합되는 노드에 낮은 비용이 산정되어, 기억공간을 모두 소비함으로서 인식에 실패한 경우의 예이다. 따라서 비용산정의 규칙을 보강하고, 식(5)의 속성에 선소의 두께를 고려한 형상 정보(shape information)를 추가함으로써 인식율을 향상시킬 수 있을 것으로 사료된다.

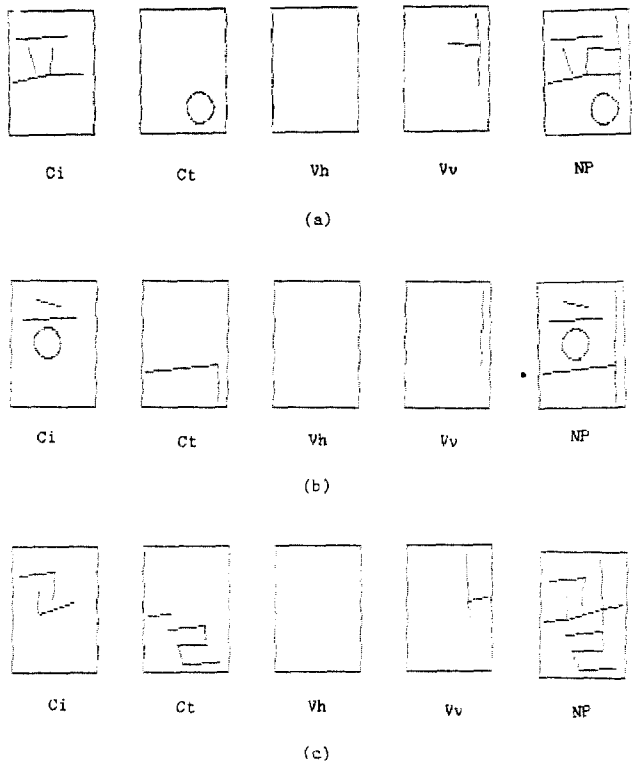


그림 8 인식에 실패한 경우의 예
Examples of ill recognition.

탐색전략을 실용적인 문제에 적용할 때는 경로의 비용과 목표를 찾기 위한 탐색의 비용을 고려하게 된다. 문자인식에서는 탐색할 목표상태가 미리 정의될 수 없기 때문에 목표상태에 이르는 경로에 대한 경험적 정보를 이용하는 탐색전략은 적용할 수 없다. 따라서 상당히 소모적인 방법이지만 문제의 해를 반드시 제공하는 맹목적 탐색방법의 하나인 균일비용탐색을 이용하였다. 균일비용과 넓이우선탐색을 각각 수행했을 때 확장된 노드수를 비교하기 위하여 각 형식별로 빈도수가 높은 패턴을 대상으로 실험하여 표 3과 같은 결과를 얻었다.

표 3 균일비용과 넓이우선탐색의 노드수 비교
Comparision of umiform-cost and breadth-first search.

패턴	SPACE*	UCS*	BFS*	패턴	SPACE	UCS	BFS
F10	4**5	12	14	F40	4**5	5	5
F11	4**10	27	27	F41	4**8	13	13
F23	4**7	18	24	F42	4**4	4	4
F13	4**6	13	13	F43	4**13	34	34
F14	4**2	2	2	F44	4**11	27	29
F15	4**4	5	6	F45	4**11	81	81
F16	4**7	30	46	F46	4**9	18	20
F17	4**10	38	134	F47	4**11	27	27
F18	4**4	8	8	F48	4**9	25	37
F19	4**8	23	29	F49	4**6	15	17
F20	4**2	2	2	F50	4**9	12	35
F21	4**7	25	53	F51	4**10	8	25
F22	4**5	5	5	F52	4**12	14	51
F23	4**6	27	65	F53	4**9	11	19
F24	4**10	28	73	F54	4**12	10	31
F25	4**6	9	22	F55	4**11	13	45
F26	4**5	9	22	F55	4**10	11	18
F27	4**5	13	51	F57	4**9	15	92
F28	4**7	17	41	F58	4**9	11	45
F29	4**3	5	7	F59	4**8	29	153
F30	4**4	4	15	F60	4**9	28	44
F31	4**10	21	83	F61	4**10	67	190
F32	4**8	47	195	F62	4**12	103	1241
F33	4**8	20	69	F63	4**9	13	60
F34	4**6	15	51	F64	4**12	32	102
F35	4**11	57	803	F65	4**11	27	133
F36	4**7	14	47	F66	4**10	43	60
F37	4**6	12	36	F67	4**9	19	56
F38	4**9	59	757	F68	4**9	16	55
F39	4**6	13	64	F69	4**8	16	95

* SPACE : 상태공간의 전체 노드수
* UCS: 균일비용탐색의 확장 노드수
* BFS: 넓이우선탐색의 확장 노드수

여기서 Fij는 i-형식의 j번째 패턴을 의미하며, 'SPACE'는 주어진 문자를 인식하기 위하여 생성될 수 있는 상태공간의 전체 노드수이고, 'UCS'는 균일비용탐색을 수행하였을 때 생성되는 노드수, 'BFS'는 넓이우선탐색을 수행하였을 때 생성되는 노드수이다. 여기서 UCS≦BFS의 관계가 성립함으로서, 이 논문에서 탐색효율을 향상시키기 위하여 이용한 경로비용함수가 바르게 설정되었음을 알 수 있고, 또한 SPACE≫UCS와 SPACE≫BFS 의 관계가 성립함으로써, 탐색영역을 최소화하기 위하여 설정한 연산자 적용규칙이 타당하였음을 알 수 있다.

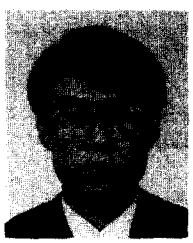
Ⅳ. 결 론

이 논문에서는 구문론을 이용한 문자인식 방법의 한계를 극복하고, 패턴 묘사와 인식 과정을 유기적으로 결합하기 위하여 인공지능의 기본적인 문제풀이 기법인 상태공간탐색을 이용하여 한글을 인식하는 방법을 제안하였고, 컴퓨터 실험을 통하여 그 유용성을 확인하였다. 상태공간 탐색 알고리즘은 원칙적으로 해를 제공하지만, 많은 경우에 있어서 탐색을 위한 연산시간 및 기억공간의 제한 때문에 실용적으로 사용하기 위하여는 탐색효율을 향상시키는 방법이 필요하다. 이를 위하여 한글 조합규칙에 입각한 구조정보와 기본 6-형식의 구조평면에서 각 자소가 갖는 위치정보를 이용하여 확장노드 수를 제한하였다.

상용문자의 대부분이 간단한 구조의 문자이고, 또한 높은 인식율과 처리시간은 서로 비례하기 때문에 실용의 문자인식 시스템은 다단계 인식 시스템(multi-pass recognition system)으로 구현하는 것이 바람직하다고 사료되어 이에 대한 연구와 실시간 처리를 위하여 제안 알고리즘을 하드웨어화시키는 연구가 필요하다.

參 考 文 獻

1. 이주근, "한글문자의 인식에 관한 연구(Ⅳ)", 대한전자공학회 논문지, 제9권, 제4호, pp. 197~204, 1972.
2. Laveen N. Kanal, "Problem-solving models and search strategies for pattern recognition", IEEE Trans. Pattern Anal. and Machine Intell., vol. PAMI-1, No. 3, pp. 245~251, Apr. 1979.
3. Kyu Tae Park, Sang Woon Kim, and Byeong Rae Lee, "A study on primitive segments extraction from printed Korean character by means of a directional projection", Proceedings of 1987 Joint Technical Conference on Cicuits and Systems, vol. CAS87-72, pp. 67~71, Jul. 1987.
4. Kyu Tae Park, Sang Woon Kim, and Byeong Rae Lee, "Recognition of printed Korean characters using a uniform- cost search strategy", Proceedings of IEEE Region 10 Conference, vol. 1, pp. 61~65, Aug. 1987.
5. Nils J. Nilsson, Problem-Solving Methods in Artificial Intelligence, New York, McGraw-Hill, 1971.
6. Jaihie Kim and Jisoo Sohn, "An efficient search method and its application to the planning of airport loading bridge assigments", Proceedings of IEEE Region 10 Conference, vol.2, pp. 487~491, Aug. 1987.



金 商 雲(Sang Woon KIM) 正會員

1956年 3月13日生

1974年 3月~1978年 2月：韓國航空大學 通信情報工學科卒

1978年 3月~1980年 2月：延世大學校 大學院 電子工學科卒(工學碩士)

1980年 3月~1988年 2月：延世大學校 大學院 電子工學科卒(工學博士)

1984年 4月~1989年 3月：韓國放送通信大學 電子計算學科 助教授

1989年 4月~現在：明知大學校 工科大學 電子計算學科 助教授

朴 圭 泰(Kyu Tae PARK) 正會員

1933年 6月11日生

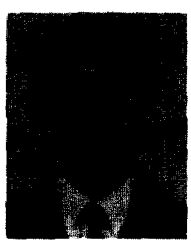
1953年~1957年：延世大學校電氣工學科 卒業

1957年~1964年：延世大學校 大學院 電子工學科 卒業(工學碩士)

1962年~1964年：英國 LONDON UNIVERSITY, MSC(工學碩士)

1967年~1969年：英國SOUTHAMPTON UNIVERSITY, Ph. D. (工學博士)

1970年~現在：延世大學校 教授



李 炳 來(Byeong Rae Lee) 正會員

1963年10月29日生

1981年 3月~1985年 2月：延世大學校 電子工學科 卒

1985年 3月~1987年 2月：延世大學校 大學院 電子工學科 卒 (工學碩士)

1987年 3月~現在：延世大學校 大學院 電子工學科 博士課程