행렬기반의 정합 알고리듬에 의한 음악 기호의 인식

허 경 용[†] · 장 경 식[††] · 장 문 익[†††] · 김 재 희[††††]

요 약

패턴 인식의 분야에서 그래프는 복잡한 대상체의 표현 및 인식의 도구로서 많이 사용되지만, 그래프간의 유사성 비교에는 많은 시간이 소요될 뿐아니라 실제 입력되는 영상은 왜곡으로 인해 저장되어 있는 이상적인 영상과 동일함을 기대할 수 없으므로 유사한 정도를 판별하는 기준이 마련되어야만 한다. 이 논문에서는 행렬을 그래프의 표현 수단으로 사용하였다. 행렬은 표현이 간단하며, 정의되어 있는 연산을 통해 순서 배열 및 매칭 과정을 간단히 수행할 수 있다. 이 때 그래프를 구성하는 노드(node)들을 기하학적 위치에 따라 순서 배열함으로써 그래프를 구성하는 노드들 사이의 대응 관계를 효율적으로 찾을 수 있도록 하였으며, 노드를 묘사하는 인수들과 다른 노드들과의 관계를 통해 상이도를 정의함으로써 기호의 구조를 반영할 수 있도록 하였다. 또한 왜곡으로 인하여 기호를 표현하는 그래프의 노드가 제대로 추출되지 못한 경우는 기호의 구조를 고려하여 보정해 줄 수 있는 분할 과정을 도입하여 해결하였다. 제안한 방법은 악보의 비음표 기호 인식을 통해 실험하였으며, 실험 결과 95% 정도의 인식률을 얻을 수 있었다.

A Matrix-Based Graph Matching Algorithm with Application to a Musical Symbol Recognition

Gyeong-Yong Heo[†] · Kyung-Sik Jang[††] · Moon-Ik Jang[†††] · Jaihie Kim[††††]

ABSTRACT

In pattern recognition and image analysis applications, a graph is a useful tool for complex object representation and recognition. However it takes much time to pair proper nodes between the prototype graph and an input data graph. Futhermore it is difficult to decide whether the two graphs in a class are the same because real images are degraded in general by noise and other distortions. In this paper we propose a matching algorithm using a matrix. The matrix is suitable for simple and easily understood representation and enables the ordering and matching process to be convenient due to its predefined matrix manipulation. The nodes which constitute a graph are ordered in the matrix by their geometrical positions and this makes it possible to save much comparison time for finding proper node pairs. For the classification, we defined a distance measure that reflects the symbol's structural aspect that is the sum of the node distance and the relation distance; the former is from the parameters describing the node shapes, the latter from the relations with other node in the matrix. We also introduced a subdivision operation to compensate node merging which is mainly due to the preprocessing error. The proposed method is applied to the recognition of musical symbols and the result is given. The result shows that almost all, except heavily degraded symbols are recognized, and the recognition rate is approximately 95 percent.

※ 본 논문은 95년도 한국과학재단 연구비 지원에 의한 결과임(과제번호 KOSEF 95-0100-11-01-3).
† 준 회 원 : 연세대학교 기계전자공학부 박사과정
†† 정 회 원 : 부산 동의대학교 멀티미디어학과 조교수
††† 정 회 원 : 연세대학교 기계전자공학부 박사과정
†††† 정 회 원 : 연세대학교 기계전자공학부 교수
논문접수 : 1997년 10월 16일, 심사완료 : 1998년 5월 29일

1. 서 론

구문적 패턴 인식(syntactic pattern recognition) 방법은 통계적 방법(statistical method)과 함께 패턴 인식의 근간을 이룬다. 구문적인 방법은 패턴을 구성 성분(primitive)과 그들 간의 관계(relation)로부터 스트링(string), 트리(tree) 혹은 그래프(graph)로 나타냄을 기본 개념으로 하며, 이 중 그래프가 가장 일반적이고 확장된 개념이다.

그래프 G 는 일반적으로 노드(node)들의 유한집합 N와 이들 노드 쌍에 주어지는 에지(edge) 집합 E로 정의된다. 그래프는 패턴 인식이나 영상 이해(image understanding)에서 복잡한 대상체의 표현에 널리 쓰이지만 몇 가지 문제점이 있다. 즉, 일반적인 그래프 표현에서는 연산의 용이성만을 고려하고 노드들의 상대적인 위치를 고려하지 않으므로, 전체적인 그래프의 형태를 그 표현으로부터 쉽게 유추할 수 없으며, 실제 입력되는 영상은 잡음이나 왜곡에 의해 저장되어 있는 이상적인 영상(ideal image)과 항상 동일하지 않으므로 유사 정도를 판단할 수 있는 척도가 정의되어야만 한다. 또한 그래프간의 유사성 비교는 많은 시간을 요하는 작업이다.

실제 입력되는 영상은 왜곡으로 인해 같은 부류(class)에 속하는 물체라 해도 항상 동일한 입력을 기대할 수 없으므로 두 물체가 같은 부류에 속하는 지는 일반적으로 두 표현이 얼마나 유사한가로부터 판단하며, 이를 위해서 물체간의 유사도(similarity measure) 혹은 상이도(distance measure)의 정의가 필요하다.

그래프간 비교에 드는 시간의 감소와 더불어 왜곡된 기호를 효율적으로 인식하기 위한 방법으로는 항상 존재하는 특징 성분을 먼저 찾아내며 성분의 변형 가능성을 원형 그래프(prototype graph)에 포함시키는 방법[2], 다른 성분들과의 관계를 이용한 이완법(relaxation method)[3], 변형 규칙(embedding rule)을 사용하여 입력된 데이터 그래프의 노드와 관계 자체를 변형시키는 방법[4][5] 등이 있다. 하지만 원형 그래프에 변형을 포함시키는 경우 예측치 못한 변형에는 인식이 불가능하며, 이완법의 경우 성분이 누락되는 경우 관계의 유사성을 판단하기 위한 기준이 마련되어야 하므로 이중의 시간이 소요될 수 있다. 또한 노드와 관계 자체를 규칙에 의해 변형시키는 방법은 원형 그래프로 바꾸기 위해 무의미한 변형을 계속하는 경우가 발생한다. 이외에도 분석 영역에 의존하는 지식을 사용하여 탐색 과정을 줄이는 방법이 있다[6][7][8].

이 논문에서 입력된 기호 영상은 먼저 세선화 과정을 거친 후, 정의된 구성 성분들과 그들의 관계로 이루어지는 그래프로 나타내며 이를 행렬로 표현하였다. 이때 구성 성분들은 행렬 내에서 성분들 간의 상대적인 기하학적 위치에 따라 배열하였다. 그래프를 구성하는 성분들을 임의로 배열할 경우 그래프간의 비교는 모든 가능한 후보들을 비교해야 하므로 많은 시간이 소요되고, 깊이나 넓이 우선 방법으로 배열할 경우는 하나의 성분이 빠질 경우 나머지 성분들의 배열이 크게 달라진다. 하지만 제안한 방법에서는 성분들의 기하학적인 위치를 사용하여 가까이 위치한 성분들은 행렬 내에서도 가까이 있도록 배열하므로, 동일한 기호는 항상 동일한 순서로 성분들이 배열됨을 보장한다. 따라서 그래프간의 비교를 단순화할 수 있고 성분의 누락에도 나머지 성분들의 순서가 달라지지 않으므로 이동(shift)에 의해 간단히 대응되는 노드 쌍을 재배열할 수 있다. 그래프 탐색 시간 측면에서도 맹목적 탐색을 사용하는 경우는 노드 수에 따라 지수적으로 증가하는 시간이 필요하며, Fu[6]의 방법에서도 노드 수의 5승에 비례하는 시간이 필요하다. 하지만 제안한 방법에서는 성분간의 기하학적 위치가 크게 차이나는 경우는 비교하지 않아도 되므로, 노드 수의 자승에 비례하는 시간만이 필요하다.

두 그래프간의 상이도는 각 성분간 상이도의 합으로 정의되며, 그래프간 상이도로부터 두 그래프가 동일한 물체를 묘사하는지 판별한다. 성분간 상이도는 대응되는 성분의 형태 차이에 의한 상이도와 기호 전체적인 구조를 고려하는 관계 상이도의 합으로 정의하며, 전자는 성분의 모양을 묘사하는 인수의 비교를 통해 얻고, 후자는 그래프 내에서 다른 성분들과의 연결 관계를 비교하여 얻는다.

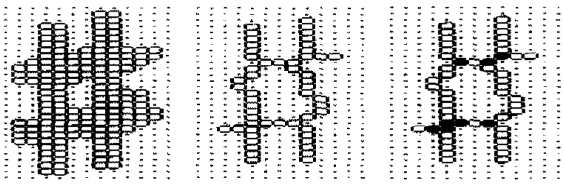
2장에서는 기호를 행렬로 나타내는 과정을, 3장에서는 나타내어진 기호를 비교하는 과정을 그리고 4장에서는 이를 악보의 비음표에 적용한 실험 결과에 대해 논하였다.

2. 그래프 표현

2.1. 구성 성분의 추출

인식을 위해 기호는 직선과 곡선의 구성 성분을 노

드로 하고 이들의 연결 관계를 에지로 하는 그래프로 표현하였고, 이를 행렬의 형태로 나타내었다.



(그림 1) 세선화 및 교차점 추출
(Fig. 1) Thinning and crossing point extraction

주어진 기호는 먼저 세선화(thinning) 과정[9,10]을 거쳐 기호의 골격[11]을 얻고(그림 1) 직선 근사화를 통해 기호를 근사적으로 묘사하는 기호점(symbol point)을 얻는다. 기호점은 네 종류가 있으며, 기호점 중 교차점과 끝점은 세선화 후 마스크(mask)를 사용하여 구하고, 변각점과 경로점은 직선 근사화 과정에서 기호점들 간의 내각을 이용하여 구한다.

<u>기호점의 정의</u>

1. 교차점(Crossing Point) : 세 개 이상의 선분이 만나서 생기는 점을 교차점이라 한다.
2. 끝점(Ending Point) : 하나의 선분만이 연결되어 있는 점을 끝점이라고 한다.
3. 변각점(Angle Point) : 두 선분이 90° 이내의 각을 이루며 만나는 점을 변각점이라 한다.
4. 경로점(Route Point) : 두 선분이 90° 보다 큰 각을 이루며 만나는 점을 경로점이라 한다.

교차점과 끝점은 세선화 수행 후 3×3 마스크에 의해 각각 식 (1)과 (2)로 구한다. 이들 교차점과 끝점을 특히 특징점(feature point)이라 한다. 이 때 이웃점(neighbor point)은 3×3 마스크에서 8 방향으로 연결된 점들을 말한다.

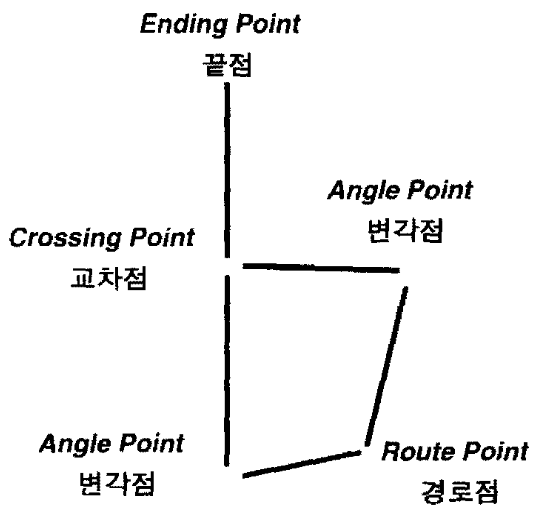
if (*NeighborNo(point)* ≥ AND *Transition(point)* ≥ 3) then (CROSSING POINT) (1)

if (*NeighborNo(point)* = 1) then (ENDING POINT) (2)

식 (1), (2)에서 *NeighborNo*(·)는 이웃점 중 흑화소의 개수를 말하며, *Transition*(·)은 이웃점을 시계 방향으로 탐색할 때 흑화소에서 백화소로 변하는 횟수로, 이는 다른 화소 기반(pixel based) 세선화 알고리듬 그러하듯 세선화 결과가 완전 8-연결이 성립하지 않으므로 필요하다.

특징점이 얻어진 후, 특징점 사이에 존재하는 연결된 흑화소들은 체인 코드(chain code) 형태로 얻어서 *Rosenfeld*의 방법에 의해 직선 근사화[12,13,14]를 행한다. 직선 근사화를 거쳐 남은 점들은 기호를 근사적으로 묘사할 수 있는 점으로, 이 점들과 특징점들이 기호점이 된다. 이 때, 체인 코드가 시작되는 점과 끝나는 점인 특징점과, 그 사이에 존재하는 기호점들의 합을 가지(branch)로 정의한다. 가지를 구성하는 점 중, 특징점이 아닌 점들은 기호점들간의 내각을 계산하여 90° 이내의 내각을 갖는 점은 변각점으로 표시하고 90° 보다 큰 각을 갖는 기호점은 경로점으로 곡선의 일부로 포함시킨다. 변각점은 교차점과 더불어 성분들이 만나는 점으로 변각점은 두 개의 성분이, 교차점은 세 개 이상의 성분이 만나 이루어진다.

(그림 2)는 내림표에 대한 직선 근사화의 예이다. 이 기호는 끝점과 교차점 사이에 하나의 가지가 얻어지고, 교차점을 중심으로 폐곡선을 형성하는 또 하나의 가지가 얻어진다. 후자는 (그림 2)에서 보듯이 두 개의 직선과 경로점을 포함하는 하나의 곡선으로 나누어진다.

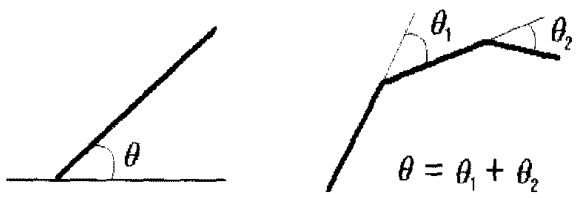


(그림 2) 직선 근사화 이후의 기호점
(Fig. 2) Critical points after line approximation

얻어진 구성 성분은 식 (3)으로 표시된다.

$$Primitive = (c_1, c_2, l, \theta, type, \overrightarrow{p_1}, \overrightarrow{p_2}) \quad (3)$$

여기서 c_1과 c_2는 구성 성분이 시작되고 끝나는 기준점을, l은 구성 성분의 길이를 나타낸다. 이 때 길이 l은 기하를 둘러싸는 최소 직사각형에 대해 상대적인 길이로 정의되며 이는 같은 기호라도 표시되는 위치에 따라 약간의 크기 차이가 있으므로 절대 길이를 사용하는 경우는 이를 보정할 수 없기 때문이다. θ는 성분이 직선인 경우에는 수평축에 대한 기울기를, 곡선인 경우에는 회전 정도로 각각 정의된다.



(그림 3) θ 의 정의
(Fig. 3) Definition of θ

$type$은 구성 성분의 종류를 나타내며 직선과 곡선의 두 종류가 있다. 그래프를 구성하는 각 구성 성분의 정의는 다음과 같다.

구성 성분의 정의

1. 직선 : 경로점이 아닌 두 개의 기호점으로 이루어진 가지의 부분집합
2. 곡선 : 세 개 이상의 기호점으로 이루어지며, 양 끝점은 경로점이 아닌 기호점이고 그 사이에 경로점이 하나 이상 존재하는 가지의 부분집합

$\overrightarrow{p_1}$과 $\overrightarrow{p_2}$는 성분을 둘러싸는 최소 직사각형의 좌상단과 우하단 좌표를 나타낸다.

노드를 정의하는 일곱 개의 성분 중 상이도(distance measure)를 계산할 때 쓰이는 성분은 l, θ, $type$의 세 개이고, $\overrightarrow{p_1}$과 $\overrightarrow{p_2}$는 순서 배열을 위해, c_1과 c_2는 행렬 표현에서 노드들의 연결 관계를 얻기 위해 사용된다.

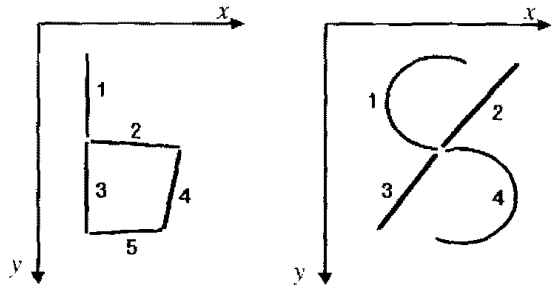
2.2. 순서 배열

구해진 성분들은 행렬을 구성하기에 앞서 순서 배열을 행한다. 노드의 순서 배열을 위한 규칙은 Wei의 방법[3]을 참조로 하여 나음과 같이 정의하였다.

순서 배열 규칙

1. 성분을 둘러싸는 최소 직사각형의 y축 중심 위치가 작은 것이 우선한다.
2. 성분을 둘러싸는 최소 직사각형을 y축으로 투영시켰을 때 80% 이상 겹치는 경우는 x축 중심 위치가 작은 것이 우선한다.

이 논문에서 정의한 성분들은 서로 교차하는 경우가 없고 만나는 경우만 있으므로 위와 같은 간단한 규칙만으로도 일관된 순서를 얻을 수 있다. 이러한 기하학적 순서 배열은 동일한 기호에 대해 성분들이 동일한 순서로 배열됨을 보장하며 성분이 누락되는 경우에도 나머지 성분들의 배열이 달라지지 않는다. 따라서 대응되는 노드 쌍을 찾는 작업이 모든 후보를 비교하는 과정에서 위치가 비슷한 일정 범위의 성분들만을 비교하는 과정으로 줄어들며, 간단한 이동에 의해 기호를 표현하는 행렬들이 적절한 대응 관계를 가지도록 재구성할 수 있다. 이 규칙에 의한 순서 배열의 예가 (그림 4)이다.



(그림 4) 순서 배열
(Fig. 4) Ordering

2.3. 그래프의 행렬 표현

그래프의 표현을 위해 이 논문에서는 행렬을 사용하였다. 행렬은 표현이 간단하며 수학적으로 연산이 잘 정의되어 있으므로 사용하기가 편리하다. 기호를 묘사하는 행렬은 식 (4)와 같이 정의한다.

$$Matrix = (\overrightarrow{P}, \overrightarrow{R}) \quad (4)$$

이 때 $\overrightarrow{P}$는 기호의 구성 성분인 노드(node)로 행

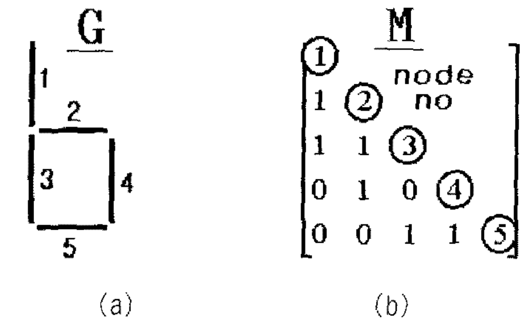
렬에서 대각 성분으로 나타나고, $\vec{R}$은 노드 쌍의 관계를 나타내는 에지(edge)로 행렬에서 비대각 성분으로 나타난다. 각각의 정의는 식 (5), (6)과 같다.

$$\vec{P} = (\ l,\ \theta,\ type\ ) \qquad (5)$$

$$\vec{R} = \begin{cases} 1 & : \text{두 구성 성분이 동일한 기호점을 공유할 때} \\ 0 & : \text{그외의 경우} \end{cases} \qquad (6)$$

노드 $\vec{P}$는 상이도 계산에 쓰이는 세 개의 인수만을 가지며 이들은 식 (3)에서 정의된 것을 따른다. 에지 $\vec{R}$는 이진 관계(binary relation)로 두 구성 성분이 동일한 기호점을 공유하는지, 서로 만나는지(touching)의 여부를 나타낸다. 즉, n번 노드와 m번 노드가 같은 기호점을 공유하는 경우 m행 n열의 원소와 n행 m열의 원소는 1로 나타나며 기호점을 공유하지 않는 경우는 두 성분이 영(zero)으로 나타난다.

기호의 표현 예가 (그림 5)이다. (그림 5)에서 (a)는 순서 배열된 기호를 나타내며 이 성분들을 행렬로 표시한 것이 (b)이다. (c)는 각 노드들의 특징값을 나타낸 것이다.



(a) (b)

no.	l(%)	θ(°)	type
1	50	90	LINE
2	40	0	LINE
3	50	90	LINE
4	50	90	LINE
5	40	0	LINE

(c)

(그림 5) 기호의 행렬 표현
(Fig. 5) Matrix representation of a symbol

[illegible] 행렬은 대칭 행렬(symmetric matrix)을 이루며 구현과 계산의 편의를 위해 대각 성분과 그 아래 부분만을 표기하고 매칭에서도 이들 부분만을 사용한다.

3. 그래프 매칭

기호를 묘사하는 행렬은 저장되어 있는 원형(prototype)과의 비교를 통해 어떤 기호인지 판별한다. 이 때 두 기호간의 이질적인 구조 정도를 나타내는 상이도(distance measure)로써 입력 기호와 원형 기호의 그래프를 표현하는 두 행렬간의 거리를 구하여 최소의 상이도를 갖는 원형으로 입력된 기호를 판별한다. 이 때 최소의 상이도가 임계치를 넘는 경우, 즉 구조적으로 많은 훼손이 발생한 경우는 비인식으로 처리하였다.

행렬의 정합 과정은 우선 두 행렬이 같은 크기를 갖도록 조정한 후, 기호를 구성하는 노드들 중 가장 유사한 노드들이 행렬 내에서 같은 행에 오도록 재배열(rearrangement)하고, 왜곡에 의해 성분들이 나뉘어 지지 않은 부분은 분할을 통해 보정한 후 두 행렬의 상이도를 계산한다.

3.1 크기 조정

모든 기호가 같은 수의 노드로 구성되지 않으며 같은 기호라도 잡음에 의한 손실로 다른 수의 노드를 가질 수 있다. 이처럼 서로 다른 개수의 노드를 갖는 행렬을 비교할 때 대응되는 노드들이 행렬에서 같은 위치에 오도록 하기 위해서는 두 행렬은 같은 크기를 가져야 한다. 따라서 먼저 행렬의 크기를 조정한다. 행렬을 축소하는 경우는 노드의 삭제나 합병이 필요하므로 행렬은 확장만이 가능하도록 하였고, 저장된 원형 행렬(prototype matrix)과 입력된 데이터 행렬 중 크기가 작은 행렬을 나머지 행렬과 동일한 크기를 갖도록 확장한다. 확장은 식 (7)로 정의된다.

$$Expansion(M_n,\ m) = M_m\ (\text{단},\ m > n) \qquad (7)$$

여기서 M_n과 M_m는 각각 확장되기 전과 확장된 후의 행렬을, n과 m은 확장 전후의 행렬 크기를 가리킨다. 행렬은 확장만이 가능하므로 $m > n$의 조건을 만족하여야 한다. 확장의 예가 식 (8)로 크기 2인 행렬을 3으로 확장하였으므로 3행에 더미 원소('•'로 표시)가

접 4되었다.

$$Expansion(\begin{bmatrix} 1 \\ 1 & 2 \end{bmatrix}', '3')' = \begin{bmatrix} 1 \\ 1 & 2 \\ \cdot & \cdot & \cdot \end{bmatrix} \quad (8)$$

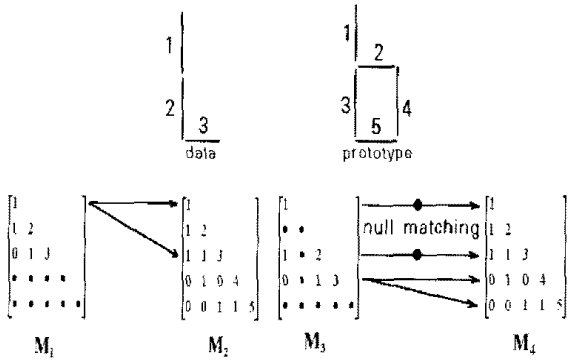
3.2 노드 재배열

확장된 행렬의 노드들은 확장되지 않은 행렬의 노드와 1 : 1 대응을 가지도록 1행의 노드부터 순차적으로 재배열된다. 먼저 확장된 행렬의 한 노드를 기준 노드로 하고 이 노드와 대응이 가능한 노드를 확장되지 않은 행렬에서 선택한다. 확장된 행렬은 확장된 정도만큼 더미 노드를 가지며 확장되지 않은 행렬은 확장된 행렬에 비하여 여분 노드를 가지게 되므로, 확장된 행렬의 한 노드와 대응이 가능한 확장되지 않은 행렬의 노드는 여러 개가 될 수 있다. 확장된 행렬의 한 노드에 대응 가능한 확장되지 않은 행렬의 이들 노드들을 노드의 비교 범위라고 한다.

비교 범위가 결정되면, 비교 범위에 있는 노드들 중에서 기준 노드와 가장 유사한 노드를 노드의 모양과 행렬 내에서의 위치에 의해 찾아 대응 노드로 결정하게 된다. 행렬의 확장으로 인해 이들 기준 노드와 대응 노드는 각 행렬에서 다른 행에 위치할 수 있으므로 기준 노드의 위치를 이동시켜 기준 노드가 대응 노드와 동일한 행에 오도록 이동시킴으로써 확장된 행렬을 재구성한다.

(그림 6)에서 M_1의 1번 노드는 M_2의 1번에서 3번까지의 세 개의 노드와 비교가 가능하다. 이는 M_1이 2만큼 확장되었기 때문이며 M_1의 2, 3번 노드가 M_2의 4, 5번 노드와 대응될 수 있기 때문이다. 확장 정도가 클수록 비교 범위도 커진다. 이에 비해 M_3의 3번 노드는 M_4의 4번부터 5번까지 두 개의 노드와 비교 가능하다. 이는 M_1의 1번 노드와 달리 M_4의 2번 노드가 M_3의 널 노드와 매칭되었기 때문이다. 즉, M_3의 널 노드 중 하나가 M_4의 노드와 매칭됨으로 인하여 매칭된 널 노드 이후의 노드들은 대응 가능한 노드 범위가 줄어들게 된다.

두 행렬의 확장 전후의 크기가 각각 n, m이고 널 매칭된 노드 쌍의 수가 l 이라면, 확장된 행렬의 i 행 노드는 확장되지 않은 행렬의 i 행 이후 $|m-n|-l+1$ 개 노드가 비교 범위에 있게 된다. 그림 6.에서 m, n은 각각 5와 3이며, M_1의 1번 노드의 경우 이전에 널 매칭된 쌍이 없으므로 M_2의 1행부터 3(= |5 - 3| - 0 + 1)개의 노드가 비교 범위에 있다. 이에 비해 M_3의 2번 노드의 경우는 이전에 널 매칭된 쌍이 하나 있으므로 M_4의 3행부터 2(= |5 - 3| - 1 + 1)개의 노드가 비교 범위에 있게 된다.



(그림 6) 노드의 비교 범위
(Fig. 6) Comparative region of a node

노드간의 유사성 판단은 비교 범위에 있는 노드들의 노드 상이도를 비교하여 상이도가 가장 작은 노드를 찾는 작업으로 성분의 모양과 기호 내에서 성분의 상대적 위치를 고려하여 식 (9)로 정의된다.

$$NodeDistance = ShapeDistance + PositionDistance \quad (9)$$

식 (9)는 노드간 모양의 차이를 나타내는 *ShapeDistance*와 행렬 내에서 노드의 위치 차이에 의한 상이도인 *PositionDistance*의 합으로 구성된다.

식 (9)에서 *ShapeDistance*는 노드의 모양을 묘사하는 세 가지 성분에 의한 상이도의 합으로, 식 (10)로 정의된다. 여기서 $\overrightarrow{P_1}$은 $(l_1, \theta_1, type_1)$이고, $\overrightarrow{P_2}$는 $(l_2, \theta_2, type_2)$로 식 (3)을 따르며 *LengthPenalty*는 두 성분의 길이 차이에 대한 벌점을 나타내는 상수이다.

$$\begin{aligned} ShapeDistance = & \ |l_1 - l_2| \cdot \text{LengthPenalty} \\ & + AngleDistance(\overrightarrow{P_1}, \overrightarrow{P_2}) \\ & + TypeDistance(\overrightarrow{P_1}, \overrightarrow{P_2}) \end{aligned} \quad (10)$$

식 (10)에서 *AngleDistance*(·)와 *TypeDistance*(·)는 각각 식 (11), (12)로 정의된다.

$$AngleDistance = \begin{cases} |\theta_1 - \theta_2| \cdot \mathrm{AnglePenalty1} \\ \quad \text{if } type1 = type2 \\ \mathrm{AnglePenalty2} \cdot \theta \\ \quad \text{if } type1 \neq type2 \\ \quad (\theta \text{ 는 } \theta_1,\ \theta_2 \text{ 중 곡선인 성분의 회전각}) \end{cases} \tag{11}$$

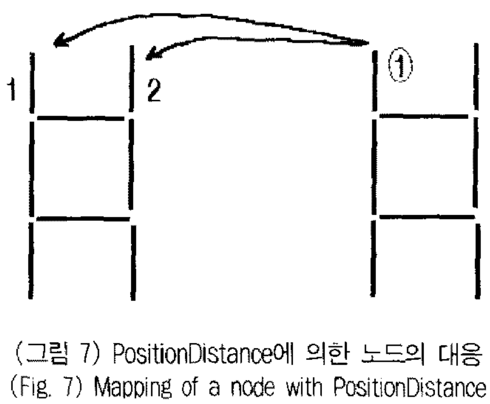
$$TypeDistance = \begin{cases} \mathrm{TypePenalty} & \text{if } type1 \neq type2 \\ 0 & \text{if } type1 = type2 \end{cases} \tag{12}$$

이 때 *TypePenalty*는 종류가 다른 노드에 대한 벌점을, *AnglePenalty1, 2*는 각각 종류가 같을 때와 다를 때 각도 차이에 주어지는 벌점을 나타내는 상수이다. 종류가 같은 노드들은 그들 각각이 갖는 각도의 차이를 취하지만, 종류가 다른 노드의 경우는 직선의 경우 회전 정도가 영(zero)이므로 곡선의 회전 정도를 노드의 각도 차이로 간주한다.

식 (9)에서, 행렬 내에서의 위치 차이에 의한 *PositionDistance*는 식 (13)으로 정의된다.

$$PositionDistance = |position1 - position2| \cdot \mathrm{PositionPenalty} \tag{13}$$

이 때 *position*은 행렬 내에서 노드가 갖는 행(또는 열)의 값을 나타내며 *PositionPenalty*는 행렬 내에서의 위치 차이에 대한 벌점을 나타내는 상수이다.



(그림 7) PositionDistance에 의한 노드의 대응
(Fig. 7) Mapping of a node with PositionDistance

이 위치 상이도는 (그림 7)에서처럼 유사한 성격을 갖는 노드가 연속으로 나올 경우 순서 배열(ordering)의 결과를 고려하여 앞의 노드에 대응되도록 하기 위함이다. (그림 7)에서 ①-1의 대응이 ①-2의 대응보다 행렬 내에서의 위치 차이에 의해 작은 상이도를 가지게 되므로 쌍을 이룬다.

(그림 6)에서 행렬 M_1의 2행 노드는 M_2의 2행에서 4행까지를 비교 범위로 가지며, 원형 그래프에서 볼 때 M_1의 2번 노드는 M_2의 3번 노드와 성분의 모양과 기호 내의 상대적인 위치가 가장 유사하여, 최소의 노드 상이도를 가지므로 쌍을 이룬다. M_1의 2번 노드와 M_2의 3번 노드가 쌍을 이루게 되면, 이들이 행렬 내에서 같은 위치에 오도록 M_1의 2번 노드를 2행에서 3행으로 이동시켜야 하며 이는 식 (14)에 정의된 이동 연산자(shift operator)를 통해 이루어진다.

$$Shift(M,\ p,\ HowMuch) = M' \tag{14}$$

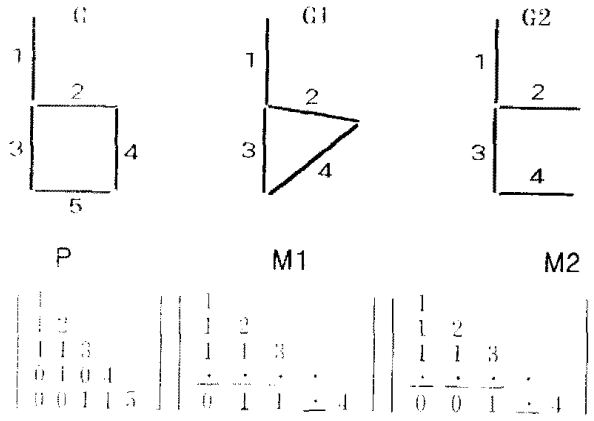
여기서 M과 M'는 이동 전후의 행렬을, p는 이동을 시작할 행의 위치를, *HowMuch*는 이동 정도를 나타낸다. 그 예가 식 (15)로 2행부터 1만큼의 이동을 시행하였으므로 행이나 열이 2인 모든 원소들이 더미로 채워지고 2행의 원소들은 3행으로 옮겨졌다.

$$Shift\left(\begin{bmatrix} 1 & & \\ 1 & 2 & \\ \cdot & \cdot & \cdot \end{bmatrix}, 2, 1\right) = \begin{bmatrix} 1 & & \\ \cdot & \cdot & \\ 1 & \cdot & 2 \end{bmatrix} \tag{15}$$

3.3 분할(subdivision)

(그림 8)에서 G와 G_1의 쌍이 G와 G_2의 쌍보다 그 모양에서 더 유사하다. 하지만 이를 행렬로 표현할 경우 P와 M_2의 쌍이 P와 M_1의 쌍보다 더 유사한 원소들을 가진다. 행렬 M_1, M_2에서 밑줄 친 부분이 P와 다른 원소들이나, 이는 기호의 구조적인 측면에서 바람직하지 못하므로 이를 보완하기 위해 분할 과정을 도입한다. 분할은 일반적인 일대다 정합의 하나로 순서배열된 노드들의 관계(relation)을 고려하는 방법으로 새로이 정의하였다.

분할은 (그림 8)에서 그래프 G_1의 4번 노드를 원형의 4번과 5번 노드와 대응시켜 둘로 나누는 역할을 한다.



(그림 8) 훼손된 기호와 그 행렬
(Fig. 8) Distorted symbols and their matrices

분할은 식 (16)을 반추한 때 확장된 행렬의 더미 노드와 인접한 노드가 관여하여 발생한다.

$$\text{if}\,[(A_{ij} = null)\,AND$$

$$(Relation(A_{i-\alpha\, i+\alpha}) = Relation(B_{ij})\ OR$$

$$Relation(B_{i+\alpha\, i+\alpha})]$$

$$\text{then}\,[A_{ij} \leftarrow *$$

$$Relation(A_{ij}) \leftarrow Relation(B_{ij}) \tag{16}$$

$$Relation(A_{i-\alpha\, i+\alpha}) \leftarrow Relation(B_{i-\alpha\, i+\alpha})]$$

여기서 A와 B는 각각 확장된 행렬과 확장되지 않은 행렬을 나타내며, A_{ij}는 A 행렬의 i 행에 있는 노드를 가리킨다. α는 보통 1의 값을 가지며, 실제 분할이 일어나는 노드는 $A_{i-\alpha\, i+\alpha}$ 이므로 넌 노드인 A_{ij}에 인접한 하나의 노드가 분할에 함께 참여함을 나타낸다. 그림 8.에서 행렬 M_1의 경우 3번이나 4번 노드가 4행의 넌 노드와 함께 분할에 참여할 수 있다.

식 (16)에서 $Relation(\cdot)$은 한 노드가 그래프를 구성하는 다른 모든 노드들과 갖는 관계로 행렬 내에서 노드를 가리키는 대각 성분의 왼쪽과 아래쪽에 나타난다. 그 예가 (그림 9)로 원으로 표시된 부분이 3번 노드가 갖는 관계이다. 이 때 물음표(?)는 확장된 더미이거나 노드 자체를 표시한다.

식 (16)의 두 번 째 조건은 확장되지 않은 행렬의 인접한 두 노드 $B_{i-\alpha\, i+\alpha}$ 와 B_{ij}가 갖는 관계의 논리합

$$P = \begin{vmatrix} 1 & & & & \\ 1 & 2 & & & \\ ① & ① & ● & & \\ 0 & 1 & ① & 4 & \\ 0 & 0 & ① & 1 & 5 \end{vmatrix}$$

$$Relation(P_{33}) = (1,\ 1,\ ?,\ 0,\ 1\ )$$

(그림 9) 한 노드가 갖는 관계
(Fig. 9) Relation of a node

(OR)이, 넌 노드 A_{ij}의 이웃 노드인 $A_{i-\alpha\, i+\alpha}$가 갖는 관계와 같음을 표시한다. 그림 8.에서 행렬 P의 4번 노드가 갖는 관계는 (0, 1, 0, ?, 1)이고 5번 노드가 갖는 관계는 (0, 0, 1, 1, ?)이다. 이들의 OR-합은 표 1.에 의해 (0, 1, 1, ?, ?)이 되고, 이는 M_1의 4번 노드가 갖는 관계 (0, 1, 1, ?, ?)와 같다. 따라서 M_1의 4번 노드는 분할될 수 있다. 이 때 두 관계의 비교는 〈표 2〉를 따른다.

〈표 1〉 OR 연산
〈Table 1〉 OR operation

A	B	A or B	A	B	A or B	A	B	A or B
0	0	0	1	0	1	?	0	?
0	1	1	1	1	1	?	1	?
0	?	?	1	?	?	?	?	?

〈표 2〉 관계 비교 연산
〈Table 2〉 Relational comparison operation

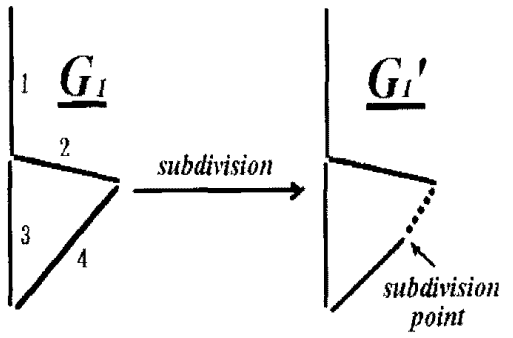
A	B	A = B	A	B	A = B	A	B	A = B
0	0	○	1	0	×	?	0	○
0	1	×	1	1	○	?	1	○
0	?	○	1	?	○	?	?	○

○ : 두 관계가 같다. × : 두 관계가 같지 않다.

P의 3번 노드와 4번 노드가 갖는 관계는 각각 (1, 1, ?, 0, 1)과 (0, 1, 0, ?, 1)으로 이들의 논리합을 구하면 (1, 1, ?, ?, 1)이 되고 이는 M_1의 3번 노드가 갖는 관계 (1, 1, ?, ?, 1)와 같으므로 분할이 발생할 수 있다. 이같이 넌 노드의 양쪽에서 분할이 가능한 경우는 대응 노드 쌍의 상이도가 큰 쪽에서 분할이 발생된다. 즉, M_1의 3번 노드는 P의 3번 노드와 쌍을 이루

[illegible] 4번 노드와 P의 5번 노드간 형태 차이가 더 크기 때문에 여기서 분할이 발생한다.

분할은 확장된 A 행렬의 널 노드 부분 A_{i0}에 가상 노드(*)를 삽입하고, $A_{i,n+1}$가 다른 노드들과 갖는 관계는 $B_{i,n+1}$가 다른 노드들과 갖는 관계로, A_{ij}가 다른 노드들과 갖는 관계는 B_{ij}가 다른 노드들과 갖는 관계로 대입하는 과정이다. 식 (16)에서 '←'가 대입을 나타낸다.



$$M_1 = \begin{vmatrix} 1 & & & & \\ 1 & 2 & & & \\ 1 & 1 & 3 & & \\ \cdot & \cdot & \cdot & \cdot & \\ 0 & 1 & 1 & \cdot & 4 \end{vmatrix} \Bigg| \begin{vmatrix} 1 & & & & \\ 1 & 2 & & & \\ 1 & 1 & 3 & & \\ 0 & 1 & 0 & * & \\ 0 & 0 & 1 & 1 & 4 \end{vmatrix} = M'$$

(그림 10) 분할
(Fig. 10) Subdivision

(그림 8)에서는 행렬 M_1의 널 노드와 4번 노드가 분할에 참여하였으며 그 결과가 (그림 10)이다. (그림 10)에서 M'의 가상 노드와 4번 노드가 다른 노드들과 갖는 관계는 P의 4번과 5번 노드가 다른 노드들과 갖는 관계를 대입한 것이다. 이 때 M'의 5행 4열의 관계가 분할의 역할을 한다. 즉, (그림 10)의 분할된 그래프 G_1'에서 점선으로 표시된 부분이 분할을 통해 생성된 가상 노드를 가리키며 행렬 M'의 5행 4열 원소가 가리키는 연결이 바로 분할 부위(subdivision point)의 연결이다.

3.4 그래프 상이도 계산

변환을 마친 행렬간의 상이도는 식 (17)에 의해 계산된다.

$$D(A, B) = \sum_{i=1}^{n}[ShapeDistance(A_{ii}, B_{ii}) + RelationDistance(A_{ij}, B_{ij})] + (m - n - l) \cdot \text{MissingPenalty} \quad (17)$$

여기서 m과 n은 분할 수정 이전의 각 행렬 크기이고, l은 분할이 발생한 회수를 나타낸다. 식 (17)은 크게 노드의 대응이 존재하는 부분(Σ부분)과 노드의 손실에 대한 부분으로 구성된다. 노드의 대응이 존재하는 부분의 상이도는 노드를 표시하는 인수들에 의한 형태 상이도(*ShapeDistance*)와 그래프 내에서 다른 노드들과 갖는 관계에 의한 관계 상이도(*RelationDistance*)의 두 부분으로 나누어진다. 노드의 손실에 대한 부분은 확장된 다른 노드와의 대응에 의한 널 매칭 부분으로 이는 노드의 손실에 대한 벌점인 *MissingPenalty*에 의해 계산된다. 널매칭된 노드의 수는 확장된 정도에 분할이 시행된 수를 뺀 것으로 확장된 만큼 널 노드가 생성되지만 분할에 의해 가상 노드가 생성되므로 이를 빼준 값이다.

식 (17)에서 노드의 대응이 존재하는 부분의 상이도 중, 노드의 형태 차이에 의한 $ShapeDistance(\cdot)$는 식 (10)을 따르며, 분할된 가상 노드(*로 표시된다.)와의 거리나 확장된 널 노드와의 거리는 영(zero)으로 주어진다. 가상 노드의 경우는 분할에 의해 생성되었으므로 상이도를 계산할 인수가 없기 때문이며, 널 노드의 경우는 손실에 대한 부분에서 계산해 주므로 중복된 벌점을 피하기 위해서이다.

식 (17)에서 다른 노드들과의 관계에 의한 $RelationDistance(\cdot)$는 식 (18)로 정의된다.

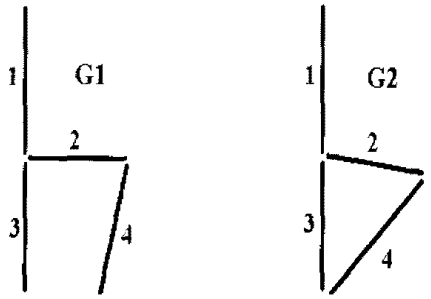
$$RelationDistance(A_{ii}, B_{ii}) = \sum_{j=1}^{n} [[Relation(A_{ii})]_j \oplus [Relation(B_{ii})]_j] \cdot \text{RelationPenalty} \quad (18)$$

여기서 n은 확장된 후 행렬의 크기를, *RelationPenalty*는 서로 다른 관계에 대한 벌점을 나타내는 상수이다. 각각의 행렬에서 대응되는 노드들이 갖는 관계를 얻은 후, 그들의 각 원소(element)가 같은 값을 갖는지, 즉 다른 노드들과 같은 관계를 갖는지 확인하여 다른 관계에 대해서는 *RelationPenalty*로 벌점을 가한다. 식 (18)에서 j는 한 노드가 다른 노드들과 갖는 관계(그림 9)의 j 번째 원소를 나타낸다.

식 (18)에서 한 노드가 다른 노드와 갖는 관계의 이질성은 배타적 논리합(exclusive OR)을 통해 알아낸다. (그림 11)은 관계 연산의 예를 보인 것으로 물음표(?)로 표시된 부분은 노드 자체이거나 확장된 디비를 나타낸다. XOR의 연산은 〈표 3〉을 따른다. (그림 11)에서 G_1의 3번 노드는 4번 노드와 연결되어 있지 않고, G_2의 경우는 두 노드가 연결되어 있으므로 RelationPenalty를 부가하게 된다.

〈표 3〉 XOR 연산
〈Table 3〉 XOR operation

A	B	A ⊕ B	A	B	A ⊕ B	A	B	A ⊕ B
0	0	0	1	0	1	?	0	0
0	1	1	1	1	0	?	1	0
0	?	0	1	?	0	?	?	0



$$P = \begin{bmatrix} 1 & & & \\ 1 & 2 & & \\ 1 & 1 & 3 & \\ 0 & 1 & 0 & 4 \end{bmatrix} \quad M = \begin{bmatrix} 1 & & & \\ 1 & 2 & & \\ 1 & 1 & 3 & \\ 0 & 1 & 1 & 4 \end{bmatrix}$$

$$RelationDistance(P_{33}, M_{33})$$

$$= \sum_{j=1}^{5}[(1,1,?,0)_j \oplus (1,1,?,1)_j] \cdot \text{RelationPenalty}$$

$$= [(1\oplus1)+(1\oplus1)+(?\oplus?)+(0\oplus1)] \cdot \text{RelationPenalty}$$

$$= (0+0+0+1) \cdot \text{RelationPenalty}$$

$$= 1 \cdot \text{RelationPenalty}$$

(그림 11) 관계 연산
(Fig. 11) Relational operation

4. 실험 결과

제안한 방법은 악보에서 나타나는 비음표 기호의 인식을 통해 실험하였다. 이 논문에서 대상으로 하는 기호는 음자리표(높은 음자리표, 낮은 음자리표), 쉼표(1분, 2분, 4분, 8분, 16분), 음높이 기호(올림표, 내림표, 제자리표), 반복 기호(D.S., D.C., Segno, Coda), 코드 기호(A ~ G, M, m, 7), 늘임표 그리고 붙임줄/이음줄 등 총 26종이다.

각 기호의 원형은 이상적인(ideal) 형태로 저장되며, 각 기호의 원형 개수는 하나로 하였다. 다수의 원형이 설정되면 비교에 많이 시간이 소요될 뿐아니라 서로 다른 기호들 간의 거리가 감소하여 경계에 놓이는 입력의 경우 오인식이 증가하기 때문이다.

Penalty 값들은 전향 탐색(forward search)을 통해 결정되었다. 실제 사용된 값은 표 4.와 같다. 이 때 *AnglePenalty1*, *2*와 *LengthPenalty*는 0.1의 단위로, 나머지들은 1의 단위로 값을 결정하였다. Penalty 값들의 변화에 따른 인식률의 변화는 그리 심하지 않았지만, 이는 실험 대상인 기호들 간에 유사성이 크지 않기 때문으로 다른 기호들에 적용하기 위해서는 다른 Penalty값들이 결정되어야할 것으로 생각된다.

〈표 4〉 벌점 상수들의 값
〈Table 4〉 Values of Penalties

벌점 종류	값	적용되는 경우
MissingPenalty	95	성분의 누락
AnglePenalty1	1.3	다른 종류의 성분들이 가지는 각도 차이
AnglePenalty25	1.0	같은 종류의 성분들이 가지는 각도 차이
TypePenalty	10	성분의 종류 차이
LengthPenalty	1.8	성분의 길이 차이
PositionPenalty	20	성분의 행렬 내 위치 차이
RelationPenalty	25	성분들이 갖는 관계의 차이

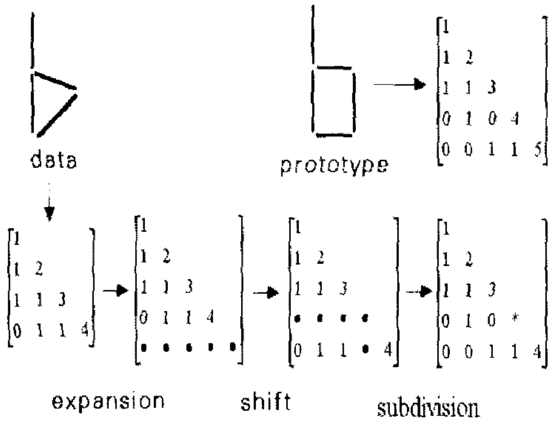
인식대상 기호에 대한 인식 결과가 표 5.이다. 실험에 사용된 각 비음표 기호들은 출판사가 각기 다른 악보에서 오선 제거 및 영역 분리를 통해 얻어진 기호들이다. 실험에서 구조적으로 크게 훼손되지 않은 경우는 거의 모든 기호를 인식할 수 있었으며, 인식률은 94.7%를 얻었다. 1분 쉼표와 2분 쉼표의 경우에는 오

〈표 5〉 인식률
〈Table 5〉 Recognition rate

기호	총 개수	인식된 수	기호	총 개수	인식된 수
높은 음자리표	46	44	낮은 음자리표	46	39
내림표	92	89	Segno	9	9
올림표	107	103	Coda	7	7
세자리표	50	48	code A	35	33
1분 쉼표	8	8	code B	39	38
2분 쉼표	12	11	code C	42	41
4분 쉼표	24	22	code D	37	36
8분 쉼표	52	48	code E	33	32
16분 쉼표	13	12	code F	42	41
D.S.	5	5	code G	25	25
D.C.	5	5	code m	27	26
늘임표	13	12	code M	29	28
붙임줄/이음줄	82	75	code 7	49	43
		인식률 : 94.7 %		929	880

선 제거 후에 동일한 모양을 가지게 되므로 오선에 대한 상대적인 위치가 추가로 사용되었다. 또한 오인식된 기호의 경우에도 악보의 기보법을 사용하여 후처리할 경우, 많은 부분 정정이 가능하다. 즉, 한 마디 내에서 음표와 쉼표의 길이 합은 일정하므로 쉼표의 길이가 잘못 인식된 경우 정정이 가능하다. 〈표 5〉의 인식률은 기보법에 의한 후처리 전의 인식률이다.

내림표(flat)에 대한 전형적인 인식 과정의 예가 (그림 12)로 순서 배열된 성분들이 원형과의 비교를 통해 재배열되고 한 번의 분할이 발생하였다. 또한 하나 이



(그림 12) 내림표의 인식 과정
(Fig. 12) Recognition process of a flat

상의 노드가 손실된 경우에도 노드 재배열에서 한 번 이상의 이동(shift)으로 노드 쌍을 찾아 줄 수 있었으며, 기호점이 올바로 찾아지지 않은 경우에도 분할을 통해 보정이 가능하였다.

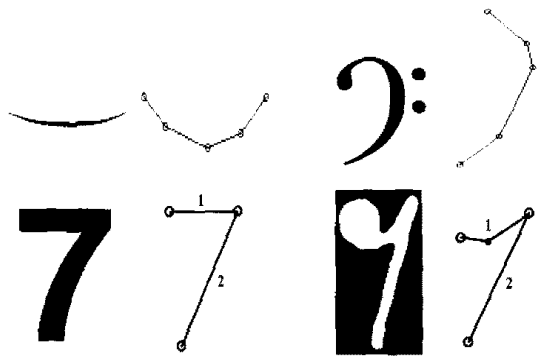
두 그래프를 비교할 때 대응 노드를 찾기 위한 탐색 경로의 확장 정도는, 맹목적 탐색의 경우 노드 수에 따라 지수적으로 증가한다. 이에 비해 *Fu*의 방법에서는 heuristic을 사용하여 feasible condition을 만족하는 종단 노드(terminal node)들만을 확장함으로써, 두 그래프의 노드 수가 M, N인 경우 $O(M^2N^2(M+N))$의 시간 복잡도를 가진다. 이에 비해 제안한 방법의 경우는 $O(M^2+N^2)$의 시간 복잡도를 가진다.

제안한 방법은 크게 세 단계의 과정으로 정합이 진행된다. 먼저 노드들을 순서 배열하고, 이들을 재배열하여 정합하게 된다. 순서 배열의 경우에는 bubble sort 알고리듬을 기준으로 생각할 때, $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$의 시간 복잡도를 가지므로 두 그래프의 노드 수가 각각 M과 N인 경우 $O(M^2+N^2)$로 표시될 수 있다. 재배열 과정에서는 비교 범위에 있는 노드들과만 비교를 행하게 되므로 $M \cdot (N-M-I+1)$의 비교 횟수를 가지며 null matching된 경우를 고려하지 않은 경우에도 $M \cdot (N-M+1)$의 비교 횟수를 가지게 된다. 이 때, 노드 수가 크

[illegible] 경우 즉, A 개 > 4 인 경우는 매칭 후보에서 제외되므로 위의 식은 $M+(K-4)$로 표현될 수 있고, 이 때 K는 4보다 작은 값을 가지는 상수이다. 따라서 제안한 방법의 시간 복잡도는 $O(M)$이 된다. 마지막 매칭 과정은 노드 수에 선형적으로 비례하는 시간을 가지므로 시간 복잡도는 $O(N)$이 된다. 따라서 제안한 알고리즘의 전체적인 시간 복잡도는 $O(M+N)$으로 표시할 수 있다.

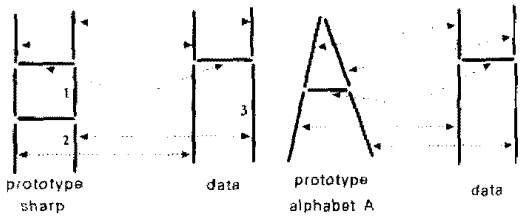
인식률의 측면에서는 Fu의 방법이 제안한 방법에 비해 약간 높은 95.9%를 나타내었다. 이는 훼손이 많은 기호 영상의 경우 제안한 방법의 경우는 순서 배열에 실패하여 인식에 실패하는 경우가 발생하지만 Fu의 방법은 보다 많은 영역을 탐색함으로써 이를 보상할 수 있기 때문이다. 기보법을 사용하여 후처리를 한 경우 이들 두 방법은 97.2%의 거의 동일한 인식 결과를 보였다.

인식 결과 (그림 13)의 쌍에서 가장 많은 오인식이 발생하였다. 이들 쌍은 그래프로 표현하는 경우 유사한 형태를 가지며, 특히 낮은음자리표와 붙임줄/이음줄의 경우에는 제안한 그래프의 표현법이 곡선의 휘어진 방향을 판별해줄 수 없기 때문이다.



(그림 13) 유사한 기호의 쌍
(Fig. 13) Error class pair

올림표(sharp)에서 발생한 오인식의 예가 (그림 14)이다. 여기서 올림표는 알파벳 'A'로 결과가 나왔다. 이는 성분들 간의 상대적인 크기만을 고려하고 기호 자체의 절대 크기 정보를 전혀 사용하지 않았기 때문이다. 이를 해결하기 위해서는 절대 크기 정보를 고려함과 동시에 다른 노드들과의 관계에 의한 분할 뿐아니라 노드 자체의 성질에 의한 분할이 이루어져야 한다. 즉, (그림 14)에서 1번과 2번 노드를 합한 경우 노드의 특성이 3번 노드와 비슷해지므로 이를 고려하여 3번 노드를 분할해주는 과정이 필요하다.



(그림 14) 올림표의 오인식
(Fig. 14) Misrecognition of sharp

현재의 분할 과정에서는 노드의 특성을 전혀 변화시키지 않으며 인접한 두 노드에서만 분할이 발생하지만, 이 방법을 다른 분야에 응용하기 위해서는 노드의 인수도 분할시켜주는 과정과 인접하지 않은 노드에서 분할이 발생하는 경우에 대한 보완이 필요할 것으로 생각된다.

5. 결 론

이 논문에서는 왜곡에 의해 변형된 기호를 기호의 구조적인 측면을 고려하여 인식하기 위해 행렬을 기반으로 하는 그래프 매칭 방법(graph matching method)을 제안하였다.

제안한 방법에서 기호는 행렬 형태의 그래프로 나타내어진다. 입력된 기호 영상은 세선화를 거친 후 노드로 정의된 구성 성분, 즉 직선과 곡선으로 표현된다. 이들 노드와 노드들 간의 연결을 나타내는 이진 관계를 바탕으로 기호를 묘사하는 행렬을 구성하며 이 행렬의 조작을 통하여 매칭 과정이 간단히 수행되도록 하였다. 이들 노드를 배열할 때, 이 논문에서는 노드의 기하학적인 위치에 따라 배열하였다. 이 방법에서는 같은 기호에 대해 항상 같은 순서의 노드 배열을 보장하므로 노드 쌍을 찾는 작업을 간단히 수행할 수 있고, 하나 이상의 노드가 손실되는 경우에도 다른 노드들의 순서에 영향을 주지 않으므로 훼손된 기호에 대해서도 무리없이 인식할 수 있었다. 또한 기호들이 다른 노드들과 갖는 관계의 비교를 통한 분할 과정을 도입하였다. 이

는 해석으로 인해 하나 이상의 노드가 추출되어 보완 기호를 기호의 구조적인 측면을 고려하여 보정하기 위함이다. 훼손된 기호를 위해서 성분들의 모양과 다른 성분들과의 관계를 통해 상이도를 성의함으로써 가장 유사한 원형으로 판별하도록 하였다. 그래프를 비교하는 시간 면에서도 노드의 사상에 비례하는 시간 복잡도를 가지므로 상당한 시간 단축 효과를 볼 수 있었다.

제안한 방법은 악보에서 비음표 기호의 인식을 통해 실험하였으며, 실험 결과 구조적으로 훼손이 심하지 않은 경우는 거의 모든 기호를 인식할 수 있었다. 보다 정밀한 전처리 과정을 도입함으로써 더 나은 인식 결과를 얻을 수 있을 것으로 기대된다.

향후 노드들 간의 관계에 의한 분할 뿐아니라 노드 자체의 성격에 의한 분할 과정의 도입, 분할 시에 노드의 인수도 나누어 줄 수 있는 방법, 분할 조건의 보완 그리고 현재는 하나의 행렬만을 확장하시만 두 행렬을 모두 확장하여 최소의 상이도를 갖도록 노드들을 배열하는 방법에 대한 연구가 진행된다면 다른 인쇄 기호 및 문자의 인식에도 무리 없이 적용될 수 있으리라 생각된다.

참 고 문 헌

[1] Ragael C. Gonzalez and Michael G. Thonason, *Syntactic Pattern Recognition An Introduction*, Addison Wesley (1778)

[2] Si Wei Lu, Ying Ren and Ching Y. Suen, "Hierarchical Attributed Graph Representation and Recognition of Handwritten Chinese Characters," *Pattern Recognition* Vol. 24 No. 7, pp. 617-632 (1991)

[3] S. L. Xie and Minsoo Suk, "On Machine Re-cognition of Hand-Printed Chinese Characters by Feature Relaxation," *Pattern Recognition* Vol. 21 No. 1, pp. 1-7 (1988)

[4] Mariusz Flasinski, "On the Parsing of Deterministic Graph Languages for Syntactic Pattern Recognition," *Pattern Recognition* Vol. 26 No. 1, pp. 1-16 (1993)

[5] M. Flasinski, "Parsing of edNLC-graph Gra-mmar for Scene Analysis," *IEEE Transactions on Pattern Recognition and Machine*

[6] M. A. Eshera and King Sun Fu, "A Graph Distance Measure for Image Analysis," *IEEE Transactions on Systems, Man and Cybernetics* Vol. SMC-14 No. 3, pp. 398-408 (1984)

[7] Q. Y. Shi and King Sun Fu, "Parsing and Translation of (Attributed) Expansive Graph Languages for Scene Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. PAMI-5 No. 5, pp. 472-485 (1983)

[8] Horst Bunke, "Attributed Programmed Graph Grammars and Their Application to Schematic Diagram Interpretation," *IEEE Transactions on Pattern Recognition and Machine Intelligence* PAMI-4, pp. 574-582 (1982)

[9] P. S. P. Wang and Y. Y. Zhang, "A Fast and Flexible Thinning Algorithm," *IEEE Transactions on Computers* Vol. 38 No. 5, pp. 741-745 (1989)

[10] H. E. Lu and P. S. P. Wang, "A comment on "A Fast Parallel Algorithm for Thinning Digital Patterns"," *Communications of the ACM* Vol. 29 No. 3, pp. 239-242 (1986)

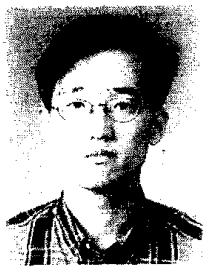
[11] Nabil Jean Naccache and Rajjan Shinghal, "An Investigation into the Skeletonization Approach of Hilditch," *Pattern Recognition* Vol. 17 No. 3, pp. 279-284 (1984)

[12] Azriel Rosenfeld and Emily Johnston, "Angle Detection on Digital Curves," *IEEE Transactions on Computer*, pp. 875-878, September (1973)

[13] Azriel Rosenfeld and Joan S. Weszka, "An improved Method of Angle Detection on Digital Curves," *IEEE Transactions on Computer*, pp. 940-941, September (1975)

[14] Cho-Huak Teh and Roland T. Chin, "On the Detection of Dominant Points on Digital Curves," *IEEE Transactions on Pattern Recognition and Machine Intelligence* Vol. 11

No. 8, pp. 859-872, August (1989)



허 경 용

1994년 연세대학교 전자공학과 (공학사)

1996년 연세대학교 대학원 전자공학과(공학석사)

1998년 현재 연세대학교 대학원 전자공학과 박사과정 휴학중

관심분야 : 패턴인식, 서명검증, 인공지능, 지능 에이전트

장 경 식

1989년 연세대학교 전자공학과 (학사)

1991년 연세대학교 대학원 전자공학과(공학석사)

1996년 연세대학교 대학원 전자공학과(공학박사)

1994년~1997년 대우전자 전략기술 제1연구소 선임연구원

1998년~현재 부산동의대학교 멀티미디어학과 조교수

관심분야 : 컴퓨터 비전, 인공지능

장 문 익

1971년 연세대학교 전자공학과 (공학사)

1976년 미국 Georgia 주립대학교 대학원 공과대학 전기공학과(MA)

1998년 현재 연세대학교 대학원 전자공학과 박사과정 재학중

1996년 ~ 현재 효성데이터시스템 대표이사 사장

관심분야 : 패턴인식, 문자인식, 영상처리

김 재 희

1979년 연세대학교 전자공학과 (학사)

1982년~1984년 Case Western Reserve University 전기공학과(공학석사) (공학박사)

1984년~현재 연세대학교 기계·전자공학부 전자공학전공 교수

관심분야 : 전문가 시스템, 정보융합등의 인공지능과 문자인식, 지도인식, 서명검증, 얼굴인식등의 패턴인식분야