Journal of The Korea Society of Computer and Information
Vol. 21 No. 9, pp. 19-25, September 2016

www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2016.21.9.019

Guitar Tab Digit Recognition and Play using Prototype based Classification

Byung-Hyun Baek*, Hyun-Jong Lee**, Doosung Hwang ***

Abstract

This paper is to recognize and play tab chords from guitar musical sheets. The musical chord area of an input image is segmented by changing the image in saturation and applying the Grabcut algorithm. Based on a template matching, our approach detects tab starting sections on a segmented musical area. The virtual block method is introduced to search blanks over chord lines and extract tab fret segments, which doesn't cause the computation loss to remove tab lines. In the experimental tests, the prototype based classification outperforms Bayesian method and the nearest neighbor rule with the whole set of training data and its performance is similar to that of the support vector machine. The experimental result shows that the prediction rate is about 99.0% and the number of selected prototypes is below 3.0%.

▸ Keyword : fret digit recognition, segmentation, virtual block, nearest neighbor rule, prototype selection

I. Introduction

다양한 디지털 기기의 사용이 보편함에 따라 악기 연주와 제작을 위한 다양한 앱(app)과 소프트웨어가 개발되어 활용되고 있다. 연주 학습과 관련된 앱은 기타, 드럼, 피아노 등 악기의 사용자 인터페이스를 보여주며 터치와 동시에 해당하는 소리를 출력한다[1]. Windows 운영체제에서 실행 가능한 Guitar Pro는 기타 악기를 다루는 많은 사람들이 사용하는 소프트웨어로써 기타 및 우쿨렐레 등의 타브 악보 편집 및 제작이 가능하며, 인쇄나 시뮬레이션 기능을 갖춘 다기능 프로그램이다[2]. 악기 연주와 관련하여 많은 앱 또는 특정 운영체제 아래 실행되는 소프트웨어들이 개발되었지만, 악보를 이용한 연주 시스템은 찾아보기 힘들거나, 음표를 인식하고 연주하는데 많은 처리시간이 필요하기 때문에 시간과 장소에 제한이 있다.

기타 타브 악보는 6선의 악보 선에 0에서 24까지 숫자로 구성되는 프렛 숫자(fret digit)로 기술된다. 디지털 기기에 입력된 기타 악보의 코드를 자동 인식하여 연주하는데 프렛 숫자 인식이 선행되어야 한다. 숫자 인식은 이미 일상생활에서도 널리 사용되고 있다. 자동차의 번호판을 인식하는 효율적인 차량관리 시스템이 운영되고 있으며[3], Uber는 스마트 폰 카메라를 사용하여 신용카드의 숫자를 인식하는 편리한 결제 시스템을 제공한다. 숫자 인식은 0~9까지 번호를 추출하여 인식하는 반면 타브 악보는 0~24까지 번호 인식을 목표로 한다. 기 연구된 자동 연주를 위한 악보 인식 시스템은 5선 악보를 대상이며, 인식은 수평 히스토그램을 사용하여 5선을 제거한 뒤 악보에서 사용된 악보 구성 기호들을 인식한다[4,5,6].

본 논문에서 제안하는 타브 코드 인식 시스템은 6선인 타브 악보가 대상이며, 휴대가 가능한 디지털기기에서 사용되는 것을 목적으로 한다. 디지털 기기를 선별하는 기준은 가격 경쟁력과 실시간 구동이 가능하여야 한다. 기준에 적합한 휴대용 기기로 Raspberry Pi[7]를 선택하였으며, 입력된 기타 악보의 프렛 숫자를 자동 인식하여 재생할 수 있어야 한다. 2절에서는 영상처리 기법을 이용한 입력된 기타 악보로부터 프렛 숫자를 추출하여 학습데이터를 생성하는 단계를 설명한다. 3절에서는 최근접 이웃 규칙 기반 초월 구를 이용하는 프로토타입 분류 방법을 제안하고, 4절에서 프렛 숫자 인식 평가를 비교한다. 마지막으로 5절에서는 제안하는 방법의 문제점과 개선방안을 토의한다.

• First Author : Byung-Hyun Beak, Corresponding Author : Doosung Hwang
*Byung-Hyun Baek(bbh@ksign.com), Dept. of Development, Ksign
** Hyun-Jong Lee(guswhd321@naver.com), Dept. of Computer Science, Dankook University
***Doosung Hwang(dshwang@dankook.ac.kr), Dept. of Computer Science, Dankook University
• Received: 2016. 02. 06, Revised: 2016. 03. 24, Accepted: 2016. 06. 07.

II. 프렛 숫자 추출

입력된 기타 악보로부터 프렛 숫자를 인식하고 연주를 수행하는 과정이 Fig 1에 나타난다. 타브 악보의 영상 처리를 위한 OpenCV 라이브러리[8]와 연주를 구성하는 기타 음원 파일을 재생할 수 있는 Pygame 모듈[9]을 사용한다. 프렛 숫자 인식기는 최근접 이웃 규칙을 이용하는 프로토타입 기반 학습기(prototype based classifier)을 이용한다[10].

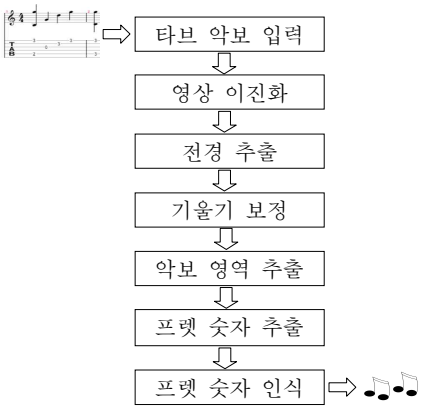


Fig. 1. Process for extracting and playing fret digits

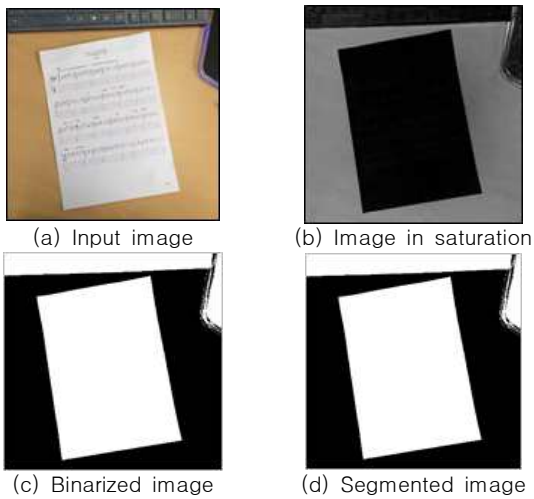
Raspberry Pi의 카메라로부터 입력된 기타 악보로부터 프렛 숫자를 추출하기 위해 이진화, 전경추출, 기울기 보정, 악보 영역과 프렛 숫자 영역을 탐지하여 학습 데이터를 준비한다. 학습 데이터가 증가하면 Raspberry Pi의 메모리 제약으로 인해 모든 데이터를 저장할 수 없다. 그러므로 준비된 학습데이터는 PC에서 프로토타입을 선택하여 Raspberry Pi에서 인식 처리가 가능하도록 한다.

1. 이진화와 전경 추출

입력 데이터는 연주자가 카메라를 통해서 촬영한 악보 영상이다. 연주자가 찍은 영상은 타브 악보를 인식하는데 배경, 잡음 등 불필요한 요소가 포함될 수 있다. 이 요소들은 정확한 프렛 숫자 영역을 탐지 하는데 장애요소이다. 또한, 입력 영상은 기울임이 발생할 수 있다. 제안하는 방법은 타브 악보의 선을 추적하면서 프렛을 탐지하기 때문에 타브 악보의 선이 기울어진 경우 추출한 프렛 이미지의 결과가 제대로 탐지될 수 없으며, 추출된 프렛 이미지가 기울어져 인식률에 영향을 미친다.

언급된 문제를 해결하기 위해 Grabcut 알고리즘을 이용한 전경추출을 수행하고 타브 악보의 선을 탐지하여 임의의 수평선과 비교해 입력된 영상을 수직으로 보정하는 작업을 수행한다. Grabcut 알고리즘은 객체와 배경의 일부를 사용자가 지정하고 지정된 영역의 분포를 이용하여 각 화소의 에너지 함수(energy function)을 정의한 후, 에너지 함수의 최소화를 통해 관심 영역 객체를 분할하는 알고리즘이다[8, 11].

사용자의 입력한 영상을 HSV 색상 모델로 변환 후 채도(saturation) 부분만 추출한다. 추출된 채도 영상에 Otsu 기법[12]을 사용하여 이진화를 수행한다. 이진화된 결과의 외곽선 좌표를 통해서 가장 큰 영역에 외접하는 사각형을 구한다[13, 14]. 이 사각형은 Grabcut 알고리즘에서 픽셀의 전경, 배경에 대한 레이블 과정에 사용된다. Grabcut 알고리즘 과정에서 전경 추출 후 오탐지한 배경픽셀들은 침식 알고리즘으로 제거한다. Fig 2의 (a)는 사용자가 입력한 영상이고 (b)는 입력 영상에서 HSV 색상모델로 변환 후 채도 값만 추출한 영상이다. (c)는 Otsu 방법을 기반한 이진화 결과 이미지 영상이고 (d)는 Grabcut 알고리즘을 수행한 후 침식 알고리즘으로 잡음을 제거한 예이다.



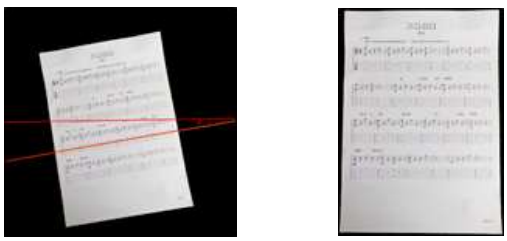
(a) Input image (b) Image in saturation

(c) Binarized image (d) Segmented image

Fig. 2. Foreground extraction process

2. 기울기 보정

악보의 기울기를 보정하기 위해 허프 변환(Hough transformation)을 통해서 이미지 내의 악보 선들을 탐지한다[13]. 검출된 선들 중 수직에 가까운 선들은 배제하고 타브 악보를 대표하는 선만을 추출한다. 임의의 수평선을 기준으로 타브 악보를 대표하는 선의 기울기를 arccos함수에 적용시켜 악보가 기울어진 만큼의 각도를 얻는다. 악보를 선의 기울기가 양수이면 시계 방향, 음수이면 반시계 방향으로 각도만큼 회전한다. 최종적으로 영상 내의 검은 부분을 최소화하기 위해서 악보의 외곽선 정보로 악보를 외접하는 사각형 좌표를 얻어 사각형 만큼 잘라낸다. Fig 3 (a)는 탐지된 선과 임의의 수평선의 각도를 보여주고 (b)는 기울기가 수정된 악보 영상이다.



(a) Input image (b) Skewness-corrected image

Fig. 3. Skewness correction process

3. 타브 악보 영역 추출

기타 타브 악보는 5선 악보와 달리 기타의 6현을 사용하여 6선으로 이루어진 악보이다. Fig 4는 5선 악보와 타브 악보의 예이다. 기본 구조는 가장 상단의 선이 기타의 1번 줄, 그리고 가장 하단의 선이 기타의 6번 줄을 표현한 것이며, 각 선상의 프렛 숫자는 기타 지판에 박혀있는 음의 경계선으로 기타의 음정을 결정하는데 가장 중요한 요소이다.

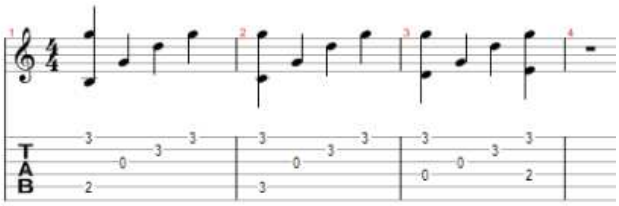


Fig. 4. Example of tab chords

악보에서 기타 연주를 위해 필요한 타브 악보 영역만을 탐색하기 위해서 템플릿 매칭(template matching) 기법을 적용하여 이진화 작업을 수행하며, 이진화 작업을 수행한 결과로 악보 이미지 객체와 배경을 분리시킨다. 이진화의 임계값은 Otsu 임계값 방법을 이용하여 자동으로 설정한다. Fig 5에서 왼쪽에 표시된 사각형 영역은 모든 타브 악보에서 공통적으로 나타나는 영역이다. 이 영역을 템플릿으로 설정하면 전체 악보에서 타브 악보 영역을 찾을 수 있다.

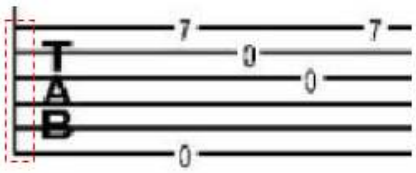


Fig. 5. Template for finding tab section

템플릿 매칭의 결과로 반환되는 값은 타브 악보 영역의 좌측 상단 좌표 값 들이다. 이 좌표 값을 이용하여 영역을 확대해 타브 악보 영역을 정한다. 6선으로 이루어진 타브 악보 영역에서 세로 1개의 픽셀이 1개의 선이다. 타브 악보 영역의 시작 좌표부터 연속되는 선이 존재하는지 탐색함으로써 6개 선의 시작 위치를 찾는다.

4. 프렛 숫자 영역 탐지

하나의 선에는 여러 개의 프렛 숫자가 존재하며, 프렛 숫자는 선의 공백 영역에 위치한다. 프렛 숫자의 위치를 얻기 위해 가상 블록(virtual block)을 이용하여 공백을 탐색한다. 가상 블록은 1×5(세로 1, 가로 5)의 픽셀의 크기를 가지며 Fig 6과 같이 타브 선 왼쪽 시작 부분부터 좌에서 우로 한 픽셀씩 순차적으로 이동할 때마다 생성된다. 블록이 생성되는 범위는 해당 픽셀과 오른쪽으로 인접한 4개의 픽셀을 포함한다.

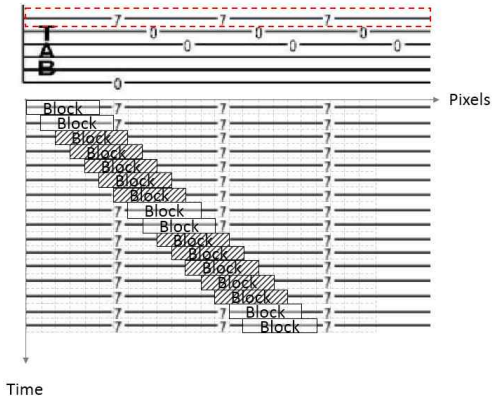


Fig. 6. Example of generating sequential virtual blocks

프렛 영역내 숫자의 정확한 위치를 얻기 위해 가상블록의 크기를 변화시키면서 가로 5픽셀로 설정한다. 가로 6픽셀 이상의 블록은 분할 시 프렛 이미지가 Fig 7과 같이 치우치는 문제가 발생 한다. 타브 선에서 하나의 프렛 숫자와 다른 하나의 프렛 숫자가 6픽셀 이하의 가까운 거리에 위치한 경우에 발생되는 결과이다. 가로 4픽셀 이하의 블록은 잡음에 비교적 취약하여, 연속된 픽셀이 공백이 아님에도 공백으로 나타나 오류가 발생할 가능성이 높다. 치우친 프렛 숫자는 프로토타입 기반 학습기의 인식률을 저하시킬 수 있다.

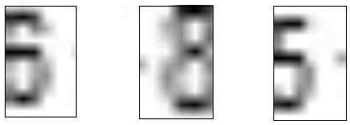


Fig. 7. Example of segmentation error

블록을 이용한 공백 탐색 값을 저장하기 위해 Fig 8과 같이 타브 선의 가로 픽셀 수만큼 0으로 초기화된 공백 count 배열을 생성한다.

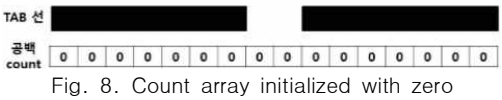


Fig. 8. Count array initialized with zero

타브 선의 각 픽셀의 count 값은 0으로 초기화시킨다. 생성되는 블록은 해당 픽셀과 오른쪽에 인접한 4개의 픽셀을 포함하여 총 5개의 픽셀에 대해 공백 유무를 검사한다. 5개의 픽셀 모두 선으로 판단되면 블록에 포함되는 각 픽셀과 매칭 되는 공백 count 값을 1증가 시킨다. 반면, 블록에 포함되는 5개의 픽셀 중 하나 이상의 픽셀이 공백으로 판단되면 블록에 포함된 각 픽셀과 매칭되는 공백 count 값을 변화시키지 않는다. Fig 9는 블록을 사용하여 공백 count 값을 갱신시키는 간단한 예이다. 각 픽셀마다 생겨나는 블록은 5개의 픽셀을 포함하고, 서로 다른 5개의 블록이 한 픽셀에 대해 공백 유무를 탐지하기 때문에 공백 count 값은 0~5의 범위를 갖는다.

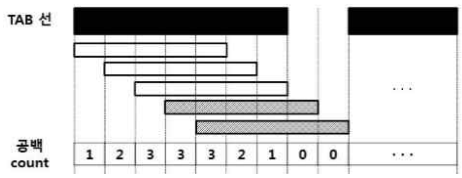


Fig. 9. Example of count array

계산된 공백 count 배열의 인덱스 0부터 배열의 크기 M까지 순차적으로 공백 count가 3인 값을 탐색한다. 처음 공백 count 값이 3인 N위치와 다음 공백 count 값이 3인 N+1위치까지 공백으로 판단한다. Fig 10은 공백을 탐색하는 예를 보여준다.

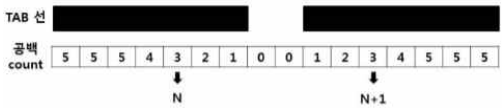


Fig. 10. Example of detection blank in count array

타브 선에서 프렛 숫자가 존재하는 공백은 가로로 약 8픽셀의 크기를 갖는다. 블록의 크기 단위로 공백을 탐지하고, count 배열에서 3인 값을 선과 공백의 경계 값으로 설정하기 때문에 1~2개의 타브 선 픽셀이 존재하더라도 정확한 공백을 탐색할 수 있다. Fig 11은 타브 선에서 공백을 탐지한 후, 분할로 추출된 프렛 숫자의 영역은 8×10의 크기를 갖는다.


Fig. 11. Example of fret digits

III. 프로토타입 기반 분류

1. 프로토타입 기반 인식 방법

프렛 숫자 인식을 위해 프로토타입 기반 최근접 이웃 알고리즘을 사용한다[11]. 주어진 프렛 숫자 데이터는 각 프렛 숫자 클래스를 대표할 수 있는 프로토타입 집합의 새로운 훈련 데이터이며, 분류 학습은 최근접 이웃 규칙을 이용한다. 훈련 데이터로부터 각 데이터가 중심으로 하는 클래스 영역 내 대표하는 초월구(hypersphere)을 생성한다. 초월구의 반지름은 최대 거리의 동일 클래스 데이터와 최소 거리의 다른 클래스 데이터의 중간이다. 프로토타입은 가능한 많은 동일 클래스 데이터를 대표하는 데이터를 우선으로 선택한다. 새 프렛의 클래스는 가장 가까운 프로토타입의 클래스로 예측한다.

주어진 분류 문제 $X=\{(x_i,y_i)|i=1,\cdots,n\}$ 에서 각 x_i는 d-차원 벡터$(x_i\in R^d)$이며 $y_i\in 1,\ \cdots, x_c$이다. X는 C개의 훈련 데이터 집합으로 구성되어 $X=\bigcup_{i=1}^{C}X^i$가 되며 $X^c=\{(x_i,c)|i=1,\ \cdots,n_c\}$는 클래스 c의 훈련 데이터이다. 그리고 $n=\sum_{C=1}^{C}n_c$가 된다. 프로토타입 선택 방법은 분류 문제 x로부터 각 클래스 내 데이터를 대표할 수 있는 적은 수를 가지는 데이터 집합인 프로토타입 $P=\bigcup_{i=1}^{C}P^i$를 선택한다. 최근접 이웃 알고리즘의 사용 시 선택된 프로토타입은 클래스의 상수 영역을 대표하는 클래스 영역 데이터로 가정되어 영역 내 위치하는 테스트 데이터의 분류는 가장 가까운 프로토타입의 클래스로 예측된다.

분류 문제의 각 데이터가 대표하는 동일 클래스 내 데이터의 부분 집합은 유사도 거리를 이용하여 계산한다. 클래스 C의 데이터 x로부터 거리 r_x내에 위치한 동일 클래스 데이터 집합은 x가 대표하는 데이터들을 포함한다. 거리 r_x는 모든 데이터와 거리를 구하여 동일 클래스 내 가장 큰 거리값 r_1과 다른 클래스 중 가장 작은 값 r_2를 갖는 거리의 중간값으로 결정한다. 이 과정에 따라 x의 모든 데이터의 클래스 영역 내 포함되는 데이터를 나타내는 대표집합 S를 생성한다.

Procedure SelectPrototype(S,X,C):
//$S(x)=\{z|d(x_j,z)\le r_x \text{ and } l(x_j)=l(z)\}$
//$X=\{(x_j,c)|i=1,\ \cdots,n \text{ and } c\in\{1,\ \cdots,C\}\}$
//C: 클래스 수
//$P^c, c=1,2,\ \cdots,C$
$P=\Phi$
for $c=1$ to C do
  $P^c=\Phi; X^c=\{x_j|(x_i,c)\in X\}$
  while $\triangle obj(x_j)>:0$
    $x_j=\text{argmax}_{x_i\in X^c}\triangle obj(x_i)$
    $P^c=P^c\cup\{x_j\}$
  end while
  $P=P\cup P^c$
end for
return P

Fig. 12. Prototype selection algorithm

Fig 12은 분류문제 X, 각 데이터의 대표 집합 S를 입력으로 하여 각 클래스 단위의 프로토타입 선택을 하기 위해 제안

된 알고리즘이다. C는 클래스 수이며 x_j^i가 새로운 프로토타입으로 선택될 값 $\triangle obj(x_j^i)$는 지금까지 선택된 동일 클래스내 데이터의 수와 x_j^i가 대표하는 데이터 수로 결정한다.

$$\triangle obj(x_j^i) = \left| X^c \cap S(x_j) \setminus \cup_{x_i \in P^c} S(x_i) \right|$$

Procedure PrototypeClassifier(P, x):
//P: 프로토타입 집합
//x: 테스트 데이터
$t = \text{argmin}_{(z,y)\in P} \; d(x,z)$
return t

Fig. 13. Classification algorithm based prototype

알고리즘의 출력은 클래스별 선택된 프로토타입 집합 $P = \bigcup_{i=1}^{C} P^i$ 이며 $|P^c| \ll |X^c|$ 이다.

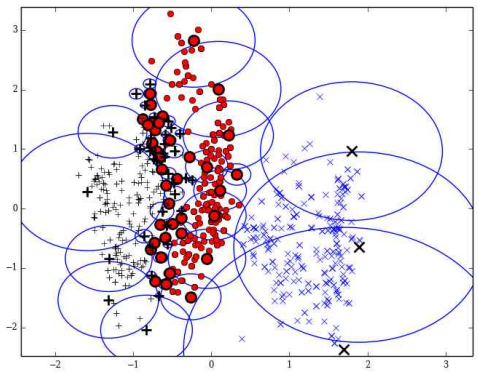


Fig. 14. Example of selecting prototype in 3-Class classification

Fig 14는 3-클래스 분류 문제에서 Fig 12의 프로토타입 선택 방법으로 선택된 경우이다. 학습 데이터는 900개로 임의로 발생시켰으며 3진 클래스 분류에 대한 실험결과이다. 77개의 프로토타입이 선택되어 축소율 8.5%를 보인다. 근접한 두 클래스 간 경계면에서 선택된 프로토타입 비율이 높으며, 그렇지 않은 클래스 들은 소수의 프로토타입이 대표하는 데이터의 비율이 높게 나타난다. 제안하는 프로토타입 선택 방법은 클래스별 영역 분포를 반영하며, 클래스 경계면에 위치한 데이터 선택비율이 높게 나타나는 경향이 있다.

Ⅳ. 실험결과

1. 프렛 데이터 수집

프렛 데이터를 수집하기 위해 초보자들이 연주할 수 있는 13개의 클래식 곡과 18개의 외국 곡, 그리고 22개의 한국 곡을 선별했다. Table 1은 선별된 53개의 악보로부터 추출한 각 프렛 데이터의 수이다. 추출된 프렛 숫자는 0~15까지 이며 프렛 숫자 별 약 400~4,600개의 데이터를 수집하였다. 수집된 프렛 데이터를 바탕으로 총 23,692개의 훈련 데이터 집합이 구성되었다.

제안하는 프렛 숫자 추출 시간을 수평히스토그램 기반 Grassfire 알고리즘과 비교하였다[5]. Table 2는 Romance 악보의 총 336개 프렛에 대해 두 방법의 비교이다. Grassfire 방법은 재귀 호출로 인해 처리 시간이 제안 방법보다 높았다. 제안 방법은 템플릿 매칭 기법으로 프렛 숫자를 탐지하기 때문에 처리 시간이 적게 소요되는 것으로 분석된다.

Table 1. The number of fret digits

프렛	수	프렛	수
0	4,595	8	1,593
1	1,068	9	1,175
2	2,428	10	1,414
3	2,252	11	385
4	1,131	12	1,293
5	1,853	13	509
6	837	14	441
7	2,212	15	506

Table 2. The comparison of preprocessing methods(sec)

구분	영역 탐지	프렛 추출	전체 시간
Grassfire 방법	4.37	3.78	8.24
제안방법	0.05	0.11	0.18

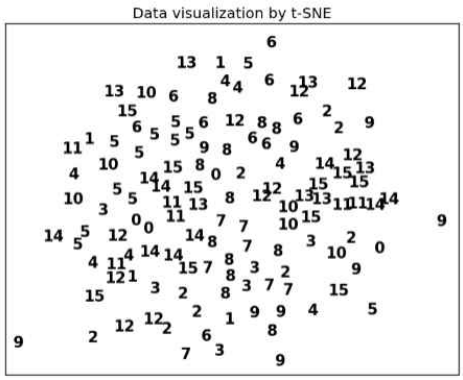


Fig. 15. Example of visualization with t-SNE method

Fig 15는 최근접 이웃 규칙이 사용하는 유클리디안 거리를 계산 후 PCA를 적용시켜 2개의 주요성분에 따른 클래스 클러스터들을 가시화 시킨 것이다[15]. 클래스별 클러스터 분포가 산재되어 있어 주어진 분류문제는 학습에 클래스 영역이 여러 입력 영역에서 나타나 프렛 숫자 인식 문제는 다중 분류 문제로 분석된다. 이러한 문제는 선형 학습기의 선택은 적절하지 않아 비선형 분류 문제 학습에 적절한 학습 알고리즘의 선택이 요구된다. 선택된 k-최근접 이웃 규칙 분류는 k 값에 의존된다. 주어진 문제는 $k > 1$인 경우 Fig 15로부터 높은

Table 3. The comparison of learning performance

Cls	Bayes	1-NN	SVM-LIN		SVM-RBF		PBL	
			F-score	SVs	F-score	SVs	F-score	No
0	0.72	1.00	1.00	25(0.5%)	1.00	71(1.5%)	1.00	25.4(0.5%)
1	0.99	0.99	1.00	17(1.6%)	0.99	49(4.6%)	0.99	20.3(1.9%)
2	0.99	0.99	1.00	26(1.1%)	1.00	63(2.6%)	0.99	45.9(1.8%)
3	0.99	0.99	1.00	30(1.3%)	1.00	91(4.0%)	0.84	30.7(1.3%)
4	0.99	0.99	1.00	42(3.7%)	0.98	101(8.9%)	0.99	40.5(3.5%)
5	0.98	0.99	1.00	55(3.0%)	0.93	99(5.3%)	1.00	60.6(3.2%)
6	0.99	1.00	1.00	40(4.8%)	0.99	77(9.2%)	1.00	45.3(5.4%)
7	1.00	0.99	1.00	42(1.9%)	0.99	95(4.3%)	0.99	40.4(1.8%)
8	0.67	1.00	1.00	35(2.2%)	0.99	76(4.8%)	0.88	65.1(4.0%)
9	0.84	1.00	1.00	34(2.9%)	1.00	86(7.3%)	1.00	45.9(3.9%)
10	0.99	1.00	1.00	21(1.5%)	0.99	64(4.5%)	1.00	35.1(2.4%)
11	1.00	1.00	0.99	16(4.2%)	0.99	51(13.2%)	0.99	30.3(7.8%)
12	0.98	0.99	1.00	27(2.1%)	0.99	65(5.0%)	0.99	50.4(3.8%)
13	0.79	1.00	1.00	13(2.6%)	0.99	42(8.3%)	0.99	35.1(6.8%)
14	0.99	0.99	1.00	19(4.3%)	0.97	49(11.1%)	0.99	45.5(10.3%)
15	0.64	0.99	0.99	14(2.8%)	0.98	46(9.1%)	1.00	45.7(9.0%)
Avg.	**0.91**	**0.99**	**1.00**	**456(2.53%)**	**0.99**	**1125(6.49%)**	**0.99**	**662.2(2.79%)**

일반화 성능을 보장할 수 없다. 그 이유는 클래스 영역이 여러 곳에서 나타나는 다중 분류 문제이기 때문이다. 그러므로 실험 평가에서 $k = 1$로 선택하였다.

2. 인식결과

프로토타입 기반 학습 성능 평가는 5 식 교차 검증(5-way cross-validation)으로 수행하였다. 학습 성능을 위하여 최근접 이웃 알고리즘, 베이지안 학습기, 지지벡터기계(support vector machine, SVM), 그리고 프로토타입 기반 학습(PBL) 알고리즘을 비교하였다.

Table 3은 준비된 학습데이터로부터 생성된 테스트 데이터 대한 F-score 평가결과이다. PBL이 선택한 프로토타입 수(No)와 SVM의 지지벡터 수(SVs)가 제시되었다. 선형커널을 이용하는 지지벡터기계의 실험(SVM-LIN)에서 *C=10*, RBF 커널을 이용하는 지지벡터기계의 경우 *C=0.5*와 $\gamma = 3$로 설정하였다. 프로토타입 기반 분류 학습은 16개 프렛 클래스에 대해 평균 약 99.0%로 나타났다. 프렛 0, 5, 6, 9, 10, 15 등의 경우 100.0% 테스트 결과를 보여 분류 학습에 준비된 데이터의 수와 다양성이 충분하기 때문인 것으로 분석된다. 반면, 프렛 3, 8의 경우 다양한 학습데이터의 추가가 요구된다. 이러한 결과는 베이지안 학습보다 높으며, 최근접 이웃 규칙과 유사한 결과이다. 제안 방법의 성능은 지지벡터기계와 비교시 SVM-LIN과 SVM-RBF의 중간에 위치한다. 학습데이터의 성능비교는 SVM-LIN의 100.0%, SVM-RBF의 99.1%로 실험되었다.

프로토타입 기반 학습에서 선택된 프로토타입의 평균 비율은 약 3.0% 이하로 매우 적은 수의 프로토타입으로 최근접 이웃 규칙 대등한 일반화 성능을 나타냈다. 한편, 제안 방법이 선택한 프로토타입의 비율은 SVM-LIN 의 지지벡터의 비율 2.53%에 비해 약간 높게 나타났으나, SVM-RBF 보다는 낮았다. 그러므로 하드웨어 제한을 갖는 Raspberry Pi에서 PBL 또는 SVM-LIN의 적용이 더 적합하다.

IV. Conclusions

본 연구에서는 타브 악보를 바탕으로 연주를 실행시켜 곡에 대한 이해를 돕고, 휴대용 기기에서도 실행 가능한 시스템을 제안하였다. 연산량을 최소화시키기 위해 기존에 연구된 악보 인식 방법에서 5 선을 제거하는 과정을 배제하고, 가상 블록을 사용하여 음을 찾았다.

타브 악보는 프렛 숫자 외에 다양한 기호들이 존재하지만 본 연구에서 개발된 시스템은 다양한 기호들을 배제하고 프렛 숫자만 처리하였다. 또한, 인쇄된 타브 악보에서 나타나는 6 선이 수평이라고 가정했다. 마지막으로 프렛은 0~24까지의 범위를 갖기 때문에 프렛 데이터가 충분히 확보된다면 모든 프렛 숫자를 반영하는 시스템 설계가 가능하다.

REFERENCES

[1] B. K. Hwang, “An Implementation of Smartphone-based Multiple Musical Instruments Application supporting Social Playing,” Journal of Digital Contents, Vol. 12, No. 4, pp. 575-583, Dec. 2011.

[2] Guitar Pro, *http://www.guitar-pro.com.*
[3] S. H. Jung, Y. W. Kwon, C. B. Sim, "An Efficient Car Management System based an Object-Oriented Modeling using Car Number Recognition and Smart Phone," Journal of Korea Institute of Electronic Communication Science, Vol. 7, No.5, pp. 1153-1164, Oct. 2012.
[4] G. H. Park, S. Y. Oh, H. J. Son, J. M. Yoo, S. H. Kim, G. S. Lee, "Decision-Tree Algorithm for Recognition of Music Score Images Obtained by Mobile Phone Camera," Journal of The Korea Contents Association, Vol. 8, No. 6, pp. 16-25, May. 2008.
[5] K. B. Kim, W. J. Lee, Y. W. Woo, "Automatic Recognition and Performance of Printed Musical Sheets Using Fuzzy ART," Journal of Korea Institute of Electronic Communication Science, Vol. 6, No. 1, pp. 84-89, Feb. 2011.
[6] J. M. Yoo, G. H. Kim, G. S. Lee, "Music Recognition by Partial Template Matching," Journal of The Korea Contents Association, Vol. 8, No. 11, pp. 85-93, Nov. 2008.
[7] Raspberry Pi document, *https://www.raspberrypi.org/documentation/.*
[8] OpenCV Document, http://docs.opencv.org.
[9] Pygame, http://www.pygame.org.
[10] S. Y. Shim, D. H. Hwang, "Prototype based Classification by Generating Multidimensional Spheres per Class Area," Journal of The Korea Society of Computer and Information, Vol. 20, No. 2, Feb. 2015.
[11] Carsten Rother, Vladimir Kolmogorov, Andrew Blake, "GrabCut : Interactive Foreground Extraction using Iterated Graph Cuts," in Proc. SIGGRAPH '04 ACM SIGGRAPH 2004 Vol. 23, Issue 3, Aug. 2004, pp. 309 - 314, 2004.
[12] N. Otsu, "A threshold selection method from gray-level histogram", *IEEE Tran.* Systems Man Cybernet, vol. SMC-9, pp.82-88 , 1979
[13] J. R. Parker, "*Algorithms for Image Processing and computer vision 2nd Edition*", Wiley publishing, 2011.
[14] Puneet, Naresh Kumar Garg, "Binarization Techniques used for Grey Scale Images," International Journal of Computer Applications, Vol.71, No.1, pp. 8-11, June 2013.
[15] L.J.P. van der Maaten and G.E. Hinton. "Visualizing High-Dimensional Data Using t-SNE", Journal of Machine Learning Research, Vol.9, pp.2579-2605, Nov. 2008.

Authors

Byung Hyun Baek received the B.S. degrees in Computer Science from Dankook University, Korea, in 2016,
He is currently a assistant in Development Department, Ksign. He is interested in machine learning and image processing.

Hyun Jong Lee is going to receive the B.S. degrees in Computer Science from Dankook University, Korea, in 2017,
He is interested in machine learning and image processing.

Doosung Hwang received B.S. and M.S. from Chungnam University in 1986 and 1990 respectively, and Ph.D. in the department of Computer Science, Wayne State University, USA, in 2003.
He is currently a professor in Dept. of Computer Science, Dankook University. He is interested in machine learning, parallel processing and image processing.