

내용기반 질의법을 이용한 디지털 오디오 콘텐츠 검색 시스템

허성필, 임우영, 한평희
KT 서비스개발연구소 디지털미디어개발팀

Digital Audio Contents Retrieval System Using a Content-based Query Method

Sung-Phil Heo, Woo-Young Lim, Pyong-Hee Han
KT Service Development Laboratory Multimedia Development Team

Abstract - 내용기반 질의법 (Content-based Query Method)은 멀티미디어 데이터가 가지고 있는 고유의 특성을 검색의 단서로 하여 질의하는 방법이다. 따라서 이러한 내용 기반의 디지털 오디오 콘텐츠 시스템은 사용자가 데이터베이스 내에서 찾고자 하는 오디오 관련 정보의 질의 방법으로써 그 노래의 멜로디 정보를 입력함으로써 이루어지게 된다. 본 논문에서는 가수명이나 노래 제목, 혹은 가사의 일부 등 기존의 음악 검색에 필수적인 텍스트 정보인 키워드를 전혀 모르는 상태에서, 휴대폰이나 컴퓨터의 마이크를 통해 자신이 기억하고 있는 노래의 일부분을 흥얼거리는 것만으로, 각종 오디오 정보를 손쉽게 찾아주는 내용기반 질의법을 이용한 디지털오디오 검색 시스템 (MuseFinder)을 소개한다. 또한 실제 유저의 편의성을 고려한 GUI에 기초한 고성능의 검색 시스템을 구현하는데 있어 주요 이슈와 고려사항에 대해서 살펴보고 그 해결 방법을 제안한다.

허밍을 이용한 AIR 시스템에 대한 본격적인 연구는 1990년대부터 이루어져 왔으며 최근 들어 이에 대한 연구가 보다 활발히 이루어지고 있는 상황이다. 예로는 Query-by-Humming, SoundCompass, SuperMBox, MiDiLiB, MELDEX등과 같은 시스템들이 그것이다.

실제 고정도 (high efficiency)의 AIR 시스템을 구현하기 위해서 고려되어야 할 사항으로 개인마다 다른 음정과 템포의 수용, 또한 허밍시 애매한 기억에 기인한 음표의 삽입이나 탈락과 같은 가창 실수 (singing error) 등이 있다. 또 다른 문제로서 허밍 질의 (hummed query)가 완벽하더라도 입력 허밍 신호를 멜로디 매칭에 이용되는 음악 표기로 100% 정확히 변환하기가 어렵다는 점이다.

종래의 AIR 시스템에서 다양한 멜로디 표현과 매칭 방법이 이용되고 있으나, 성능 면에서는 아직 만족할만한 결과를 얻지 못하고 있다. 이러한 문제점들을 해결하기 위해서는 오류에 강한 매칭 방법과 허밍 멜로디의 효과적인 표현방법이 필요하다. 따라서 구간 검출, 특징량 추출, 멜로디 표현, 유사도 측정 등과 같은 단계에서 전술한 문제점들을 효과적으로 해결해야 할 필요성이 있다.

1. 서 론

최근 인터넷과 무선통신의 발달과 더불어 디지털 멀티미디어 (오디오, 비디오, 음악, 이미지)의 사용량이 급증하고 있다. 이러한 멀티미디어 정보의 대두는 현대 생활에 많은 영향을 주고 있으며 특히 인간의 사고 및 미디어의 활용 방법에 커다란 변화를 주고 있다. 방대한 양의 멀티미디어 콘텐츠를 효율적으로 사용 및 관리하기 위해서는 정보 검색 시스템 (Information Retrieval System)이 필요하게 된다 [1], [8].

멀티미디어 정보 검색 방법은 주로 텍스트와 이미지를 중심으로 발달되어 왔으나, 최근 동영상과 오디오 정보에 대한 검색 요구가 증가되고 있는 상황이다. 오디오 정보 검색 (Audio Information Retrieval: AIR) 방법은 크게 텍스트 기반 검색 (Text-based Retrieval)과 내용 기반 검색 (Contents-based Retrieval)으로 분류할 수가 있다. 텍스트 기반 검색은 곡명, 가수명, 가사, 작곡가, 연주가 이름 등 악곡의 특징을 나타내는 서지적인 정보를 키워드로 사용하여 검색하는 방법이다. 한편 내용 기반에 의한 검색은 음악의 연주 내용 등을 검색키로 사용하는 것으로, 대표적으로 사용되는 입력정보로써 유저(users)의 허밍을 입력키로 한 QBH (Query-by-Humming)에 대한 연구가 최근 활발히 이루어지고 있는 상황이다 [2], [3], [4], [10]. 이는 악곡에 대한 선율 정보 (멜로디)를 노래 부름으로써 이것을 검색의 단서로 하여 사용자가 원하는 오디오 정보를 검색하는 방법이다.

2. 시스템 구성

제안하는 오디오 정보 검색 시스템은 그림 1과 같이 몇 개의 주요 구성 요소 (구간검출 모듈, 특징량 추출 모듈, 멜로디 표현 모듈, 검색 모듈 등)로 이루어져 있다.

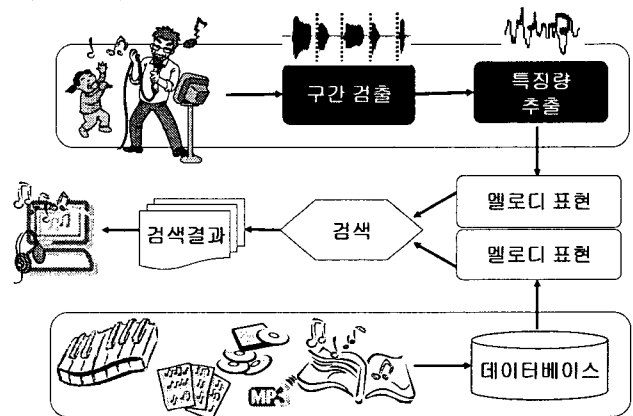


그림 1. 오디오 검색 시스템 구성도

먼저, 유저는 자신이 검색하고자 하는 곡의 멜로디를 자유롭게 허밍한다. 시스템은 입력된 허밍으로부터

터 구간검출 모듈에서 음표 (note)를 검출한 후 검색에 필요한 특징량을 추출한다. 멜로디 표현 모듈에서는 상대화된 특징 값을 생성한다. 특징량 추출 모듈에서는 음장, 음고 등과 같은 음악적인 요소를 추출한다. 검색 엔진인 유사도(similarity) 측정 모듈에서는 DP (dynamic programming) 를 이용하여 입력 허밍과 데이터베이스 곡과의 유사성을 측정한다. 검색 결과로써 악곡 리스트를 유사도가 높은 순서로 정렬 후 곡명/유사도 값/매칭 된 부분 등 악곡에 관한 각종 정보를 출력한다. 유저는 검색 결과 리스트로부터 곡을 선택하여 감상을 할 수도 있다.

3. 구간 검출

구간 검출은 입력 허밍 신호로부터 한 음표의 시작점(onset)과 끝점(offset)의 경계를 식별하는 역할을 한다. 기존에 주로 사용되던 진폭에 기초한 구간 검출 방법은 한 음표의 구간 검출에 적절한 문턱치(threshold value)를 설정하여 이벤트 검출을 수행하는 방법이다[7]. 그러나 이러한 방법은 하나의 음표가 둘 또는 세 개의 음표로 분리되어 검출될 가능성이 있고, 반대로 복수의 음표가 하나로 통합되어서 판단될 가능성도 높다. 따라서 구간 검출의 오류가 오디오 검색 시스템의 성능에 심각한 악영향을 줄 소지가 있다.

그림 2는 진폭에 기초한 구간 검출로써 허밍 파형과 세그먼트이션의 예를 나타낸다. 실제 두개 또는 세개의 음표로 검출되어야 할 구간이 하나의 음표로 검출 (A, C, D) 된다던가, 하나의 음표로 검출되어야 할 구간이 두개의 음표 (B)로 분리되어 검출이 이루어지고 있다. 이는 다음단계인 특징량 추출에 악영향을 주게 된다. 따라서 음장과 음고 정보를 정확하게 얻기 위해서는 높은 정확도의 구간 검출 방법이 요구된다.

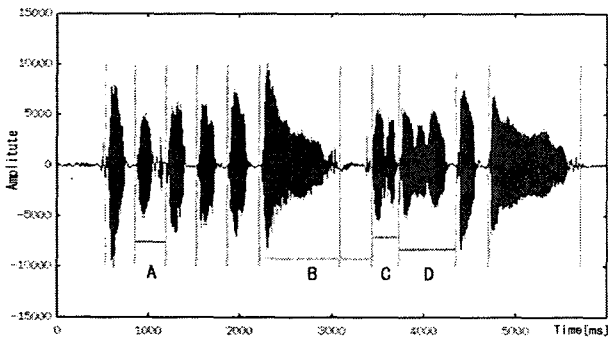


그림 2. 구간 검출 오류의 예

높은 성능의 구간검출을 위해 허밍 방법은 무성파 열음과 모음으로 이루어진 /타/ 또는 /차/ 등과 같은 단음절로 노래를 부르는 가장법을 이용한다. 이러한 가장법은 유저에게 있어 그렇게 부담이 되지 않는 가장법이다[2], [3], [10]. 고정도의 구간 검출을 위해 진폭의 차분에 주목하여 차분필터를 도입한다. 또한 마이크로부터 들어오는 호흡음을 하나의 음표로 판단하여 추출하는 케이스가 있는데 이것을 제거하기 위해 대역통과필터를 도입한다. 그림 3은 필터처리에 의한 고정도의 구간 검출 방법의 예를 보여주고 있다.

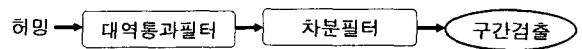


그림 3. 필터처리에 의한 구간 검출의 예

4. 특징량 추출

피치는 음의 높낮이를 나타내는 특성을 지니고 있다. 피치가 높을수록 높은 음정을 나타낸다. 따라서 입력 허밍에 대해 음고 정보를 얻기 위해서는 정확한 피치 추출이 필요하다. 그러나 아무리 성능이 뛰어난 피치 추출 알고리즘이라도 옥타브의 변화가 존재하는 허밍에 대해서 실제 기본주파수가 아닌 배음을 검출하는 경우가 있다[5], [6]. 따라서 이러한 문제점을 해결하기 위하여 본 논문에서는 복수 피치 후보 (Multiple Pitch Candidates: MPC) 를 검색의 키로써 사용하는 것을 제안한다. 본 절에서는 음장, 음고 정보 및 MPC와 신뢰도 (Confidential Measure: CM) 를 얻는 방법과 피치 성능 추출 평가에 대한 내용에 대해서 기술한다.

4.1 음장 정보 추출

음표 구간을 검출함으로써 음장(span) 정보를 얻을 수 있다. 이때, 어느 음표의 시작시각부터 그 음표의 종료시각의 차를 음장으로 하는 방법 (on-off)과 음표의 시작시각부터 다음 음표의 시작시각의 차를 음장으로 하는 방법 (on-on)이 있다.

실제 비교적 음장이 긴 이분음표를 허밍할 때 유저에 따라 이분음표의 길이만큼 정확하게 노래를 부르는 경우도 있지만, 짧게 노래를 부른 후 나머지 시간은 이분 음표의 시간만큼 휴식을 취한 뒤 다음 음표를 노래 부르는 경우가 있다 (스타카토 형태의 허밍). 즉 동일한 곡에 대해서 유저에 따라서 노래를 부르는 스타일이 다르게 된다. 따라서 음장을 특징량으로 사용하는 경우 on-off 와 on-on 중에 어느 쪽이 실제 DB에 제대로 대응되는가에 대한 조사를 할 필요가 있다. 따라서 다음과 같은 방법으로 허밍 데이터의 분석을 행한다.

먼저 각각의 유저가 부른 곡에 대해 실제 허밍된 부분에 대응되는 오디오 DB와의 음장 관계를 분석한다. 이때 허밍 템포 (동일 곡에 대해 유저별로 빠르게 노래를 부르거나, 느리게 노래를 부름)가 유저별로 다르므로 오디오 DB에서 허밍되어진 구간의 길이를 기준으로 식 (1)을 이용하여 정규화(normalization)를 취한다. 여기서, $HUM(i)$ 는 i 번째의 허밍 음표의 길이, $MIDI(i)$ 는 i 번째 MIDI 음표의 길이를 나타낸다. 식(2)의 $HUM(last)$ 와 $HUM(first)$ 는 각각 허밍의 마지막 음표의 시작시각과 최초음표의 시작시각을 나타내며, $MIDI(last)$ 와 $MIDI(first)$ 는 각각 허밍되어진 MIDI의 마지막 음표의 시작시각과 최초 음표의 시작시각을 나타낸다. $ratio_{norm}$ 가 1보다 크면, 유저가 실제 DB곡 보다 빠르게 노래를 부른 것을 알 수 있고, $ratio_{norm}$ 가 1보다 작으면 느리게 노래를 부른 것을 의미한다.

정규화 되어진 음장이 오디오 DB의 음장에 대해서 어느 정도의 길이를 가지는가에 대한 상대음장을 식(3)에 의해서 계산한다.

$$HUM_{norm}(i) = HUM(i) * ratio_{norm} \quad (1)$$

$$ratio_{norm} = \frac{MIDI(last) - MIDI(first)}{HUM(last) - HUM(first)} \quad (2)$$

$$length'(i) = \frac{HUM(i)_{norm}}{MIDI(i)} * 100 \quad (3)$$

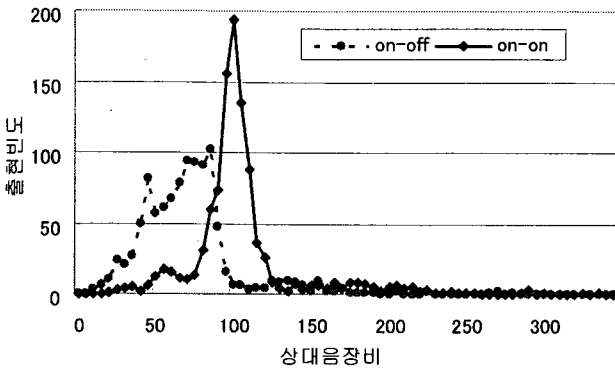


그림 4. 허밍 on-off와 on-on에 대한 상대음장 분포

허밍 on-off와 on-on에 대한 상대음장의 분포는 그림4와 같다. 만일 유저가 오디오 DB의 음장과 동일한 길이로 노래를 불렀다면 상대 음장비(Relative interval ratio)가 100인 곳에 피크가 생성될 것이다. 그러나 on-off의 경우에는 상대값 100의 출현빈도가 적으며, 또한 분산이 넓게 펼쳐져 있으므로 실제 오디오 DB의 길이와 다르게 노래를 부르는 경향이 많다는 것을 알 수 있다. 이에 반해, on-on의 경우에는 상대값 100의 부분에 피크가 있으므로, 스타카토 형태로 노래를 부르는 스타일을 흡수하는데 적합한 파라메타인 것을 알 수가 있다. 따라서 특징량으로써 음장정보의 경우에는 on-on을 사용하는 것이 바람직하다는 것을 알 수 있다.

4.2 음고 정보 추출

특징량으로써 음고의 경우는 구간검출에서 얻어진 각 음표에서 음정이 안정되어 있는 중심 프레임으로부터 피치 추출을 수행한다. 피치는 음의 높낮이를 나타내는 특성을 지니고 있다. 따라서 피치가 높을수록 높은 음정을 나타낸다.

피치 추출 방법은 켈스트럼 분석(cepstrum analysis)에 의한 방법을 사용한다[5]. 그림 5는 피치 추출의 기본적인 흐름을 보여준다. 먼저 FFT+SQR을 이용하여 입력 신호로부터 파워 스펙트럼을 구한다. 다음으로 켈스트럼을 구하기 위해 로그리즘(logarithm)과 IFFT를 적용한다. 기본주파수 영역에 존재하는 켈스트럴 피크(cepstral peak)를 피치로서 선택한다. 마지막으로, 피크의 큐프런시(quefrensy)를 주파수 영역으로 변환하여 기본주파수를 구한다.

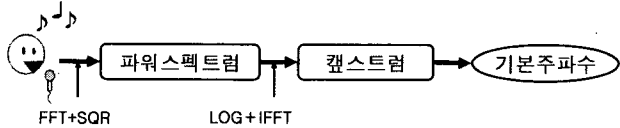


그림 5. 기본 주파수 추출 방법

4.3 복수 피치 후보의 추출

특징 추출에서 얻어진 피치는 하나가 아닌 복수 피치 후보를 질의 엔진(query engine)으로 보내며, 각 켈스트럼 피크의 에너지로부터 신뢰도를 계산한다. 여기서의 신뢰도는 제일 후보(first candidate)의 에너지 값으로 정규화한 값이다.

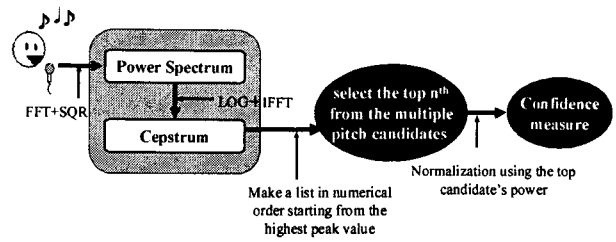


그림 6. 복수 피치 후보 및 신뢰도 추출 방법의 흐름도

5. 멜로디 표현 및 검색 알고리즘

입력된 허밍은 유저에 따라 템포(tempo)와 조(tonality)가 데이터베이스와 다르기 때문에 정규화를 수행할 필요가 있다. 따라서 전후 음표에 대해 상대음장비(relative interval ratio), 상대음고차(relative pitch difference)를 구해 개인차를 흡수한다.

검색에 있어서 데이터베이스 내에 음표의 길이가 I 인 어떤 악곡의 시계열 R 과 입력 허밍의 음표 길이가 J 인 시계열 Q 에 대해 어떤 점 i, j 에 있어서의 DP 매칭 매트릭스 $g(i, j)$ 는 반복적인 계산으로 구한다[6].

만일 음표의 길이가 I 인 데이터베이스의 시계열 $R = \{i_0, i_1, i_2, \dots, i_{I-1}\}$ 과 음표의 길이가 J 인 입력 허밍의 시계열 $Q = \{j_0, j_1, j_2, \dots, j_{J-1}\}$ 는 음장과 음고의 세트로서 구성되어 있다면, 두개의 시계열 i 와 j 에 대한 유사도 $d(i, j) = |i - j|$ 는 반복적으로 계산이 가능하고 이러한 가정에 의해 유사도에 대한 거리 계산은 다음과 같은 방법에 의해서 구한다.

경계영역에서의 계산 조건,

$$g(0, j) = d(i_0 - j), \text{ where } 0 \leq j \leq J-1, \text{ and}$$

$$g(i, 0) = \infty, \text{ where } 0 \leq i \leq I-1$$

그 외 일반적인 계산 절차

$$g(i, j) = \min \begin{cases} g(i-2, j-1) + d(i, j) \\ g(i-1, j-1) + d(i, j) \\ g(i-1, j-2) + 2d(i, j) \end{cases}$$

where $i=1, \dots, I-1$ and $j=1, \dots, J-1$

$$d(i, j) = \gamma p(i, j) + (1 - \gamma) t(i, j). \quad (4)$$

이때 $d(i, j)$ 의 페널티는 로컬 패스에 따른 무게 값(weight value)으로 페널티를 적용한다.

여기서 i 와 j 는 음표의 값을 나타내고, p 는 허밍과 데이터베이스의 음고에 대한 거리 값, t 는 음장에 대한 거리 값. γ 는 두개의 서로 다른 속성의 특징량을 하나로 통합하기 위한 가중치(weight value)를 나타낸다.

입력허밍으로부터 구한 특징량과 데이터베이스로부터 구한 특징량과는 연속 DP를 이용하여 매칭된다. 따라서 질의 엔진은 제안하는 복수 피치 후보를 수용할 수 있도록 식 (4)는 3차원으로 확장할 필요성이 있다.

그림 7은 복수피치후보를 가진 입력허밍이 3차원으로 확장된 DP 평면에서 실제 데이터베이스와 매칭이 이루어지는 메커니즘을 보여주고 있다. 허밍과 데이터베이스와의 매칭을 살펴보면, 제안된 알고리즘은 피치후보점의 모든 후보점과의 조합을 계산하고 최적의 후보점과 패스를 선택 결정한다.

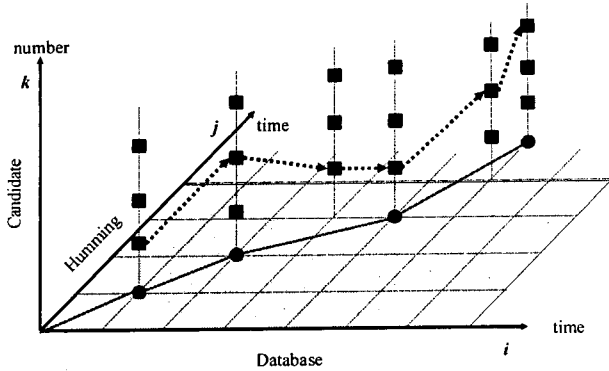


그림 7. 3차원 연속 DP알고리즘을 이용한 매칭 흐름의 예

6. 실험

평가실험에는 음표의 구간 검출의 정확도에 대한 평가 실험과, 검색에 사용되는 특징량의 조합에 따른 성능평가를 행하였다. 즉, 검색키로써 음장정보만을 사용한 경우, 음고정보만을 사용한 경우, 음고와 음장 정보를 동시에 사용한 경우, 복수피치후보점을 사용한 경우 등 검색 결과에 어떠한 파라메타가 유효한지를 살펴 보았다.

6.1 실험 조건

실험에 사용된 오디오 데이터베이스는 주로 동요로 구성되어 있으며 전체 155곡을 사용하였다. 이것은 멜로디 라인을 가진 MIDI (Musical Instrument Digital Interface)로부터 구축하였다. 허밍 데이터는 음악에 대해 전문적인 교육을 받지 않은 5명의 이용자가 2번씩 허밍한 320곡을 사용하였다.

가창방법은 곡명만 제시하고 사용자는 마이크가 있는 헤드 셋을 이용하여 자신이 알고 있는 임의의 부분 (곡의 중간 부분도 가능)의 멜로디를 자유로운 템포, 자유로운 음조로 노래를 부르게 하였다. 표 2는 실험 조건을 보여 주고 있다. 허밍의 평균 음표수는 9.8 음표였고, 평균 허밍 시간은 4.6초였다. 성능평가의 척도로써 검색 정도(retrieval accuracy)를 사용하였다.

표 1. 실험조건

악곡 데이터베이스	동요 155곡
허밍 데이터	5명이 동일곡을 2번씩 허밍
샘플링 주파수	16 kHz
분석 윈도우	64 ms 해밍 윈도우
분석 프레임	8 ms
구간 검출	BPF: 600~1500 Hz 일차 차분 필터
특징량	음장, 복수피치후보, 신뢰도

6.2 음표의 구간 검출 정확도 평가

제안한 구간 검출 방법의 정확도를 조사하였다. 세그멘테이션의 오류율은 식(5)와 같이 정의한다. 음표구간 검출 결과, 세그멘테이션 오류율은 평균 3.5%였다. 대부분의 유저에 있어 세그멘테이션 오류가 거의 없었으나, 유저D의 경우 다른 유저와 비교해 보았을 때 검출 오류가 컸다. 분석 결과, 유저D의 경우 단음절 /타/ 로 허밍을 하지 않은 것이 오류의 주원인이었다.

$$\text{세그멘테이션오류율(\%)} = \frac{\text{삽입음표수} + \text{탈락음표수}}{\text{정답음표수}} * 100 \quad (5)$$

표 2. 세그멘테이션 오류율

유저	정답음표수	삽입음표수	탈락음표수	오류율 (%)
유저A	514	4	1	1.0
유저B	495	9	2	2.2
유저C	431	10	2	2.8
유저D	438	24	15	8.9
유저E	433	9	5	3.2
합계	2311	56	25	3.5

6.3 허밍 데이터 음표수 분포 분석

허밍시 가창방법은 곡명만 제시하고 유저는 마이크가 있는 헤드 셋을 이용하여 자신이 알고 있는 임의의 부분 (곡의 중간 부분도 가능)의 멜로디를 자유로운 템포, 자유로운 음조로 노래를 부르게 하였다. 그림 8은 유저가 허밍한 데이터의 음표수 분포를 보여준다. 가장 짧게 노래를 부른 경우 네개의 음표부터 가장 길게 노래를 부른 경우는 30개의 음표까지 다양한 분포를 보였다. 7개의 음표부터 15개의 음표까지 출현빈도가 많았으며, 평균적으로 11.6 음표로 허밍을 하였다.

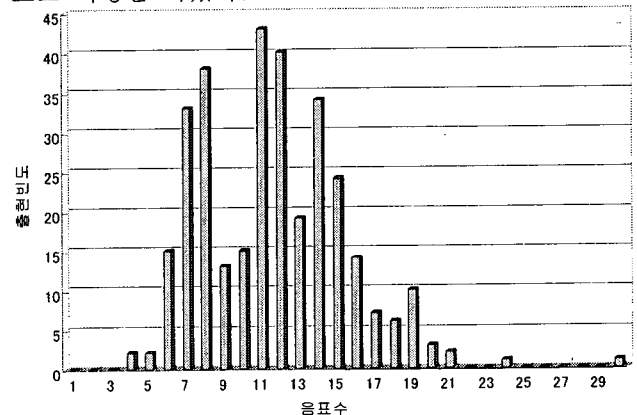


그림 8. 허밍 데이터의 음표수 분포

6.4 성능 평가

제안한 방법의 성능을 비교 평가하기 위해 종래의 악곡 검색 시스템을 구현하여 서로의 성능을 비교하였다. 악곡 검색의 실험 결과는 그림 9와 같다. 여기서 "Coarse-to-Fine"와 "Category 27"는 참고문헌 [9]의 방법을 나타낸다. 또한 "Conventional"은 정적 멜로디 표현에 의한 방법을 나타내며 "Dynamic"은 제안한 다이내믹 멜로디 표현법을 사용한 결과를 나타낸다.

위의 결과로부터, 제안한 다이내믹 멜로디 표현 방법이 정적 멜로디 표현 방법 보다 3-7포인트 더 높은 검색 정도를 얻었다. 또한 복수피치후보를 사용함으로써 검색정도는 73.9%에서 86.5%로 향상이 되었다. 또한 종래 AIR 시스템의 "Coarse-to-Fine"과 "Category 27"의 경우 1위에 랭크된 곡의 결과가 각각 78.4% 와 81.6%였다. 그러나 제안한 특징량을 사용한 경우 "TOI+ Pitch(3)+ CM"가 종래의 "Category 27"의 결과 보다 검색정도는 1위 랭크곡에 대해서 약 5%의 높은 검색 결과를 얻었다. 따라서 제안한 다이내믹 멜로디 표현과 복수피치후보 방법을 동시에 사용한 경우 종래의 방법보다 좋은 결과를 얻은 것을 알 수가 있다.

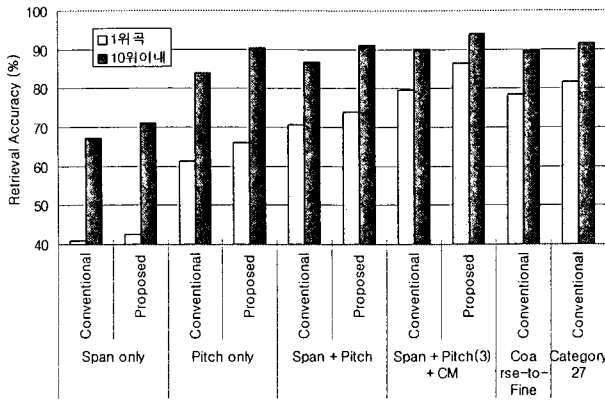


그림 9. 다양한 특징량에 의한 검색 정도의 비교

6.5 시스템 구성

실제 GUI (Graphic User Interface)에 기초한 QBH (Query-By-Humming) 시스템인 *Muse Finder*를 구현하였다. 본 시스템은 MS 윈도우 플랫폼에서 Microsoft Visual C++를 이용하여 크게 5가지의 클래스로 나누어져 있다. 개략적인 구성 요소는 음성 입/출력을 관리하는 클래스, 입력허밍 신호표시 클래스, 특징량 (복수피치후보, 스패) 추출 클래스, 검색엔진 클래스, 모든 다이아로그 관리를 위한 메일다이아로그 클래스로 나누어져 있다.

*Muse Finder*는 그림 10과 같이 크게 3개의 윈도우로 구성되어 있으며, 입력허밍에 대한 각종 정보를 디스플레이해주는 사운드 정보 윈도우 (Sound Information Window), 특징 정보 윈도우 (Feature Information Window)는 하나의 피치부터 복수의 피치 후보의 패턴과 스패정보를 함께 볼 수 있는 윈도우, 검색 정보 윈도우 (Retrieval Information Window)는 검색된 결과의 리스트를 보여준다. 검색 결과는 곡명, 매칭된 지점 등을 보여주며 사용자는 듣고 싶은 곡을 마우스로 클릭 함으로써 음악을 감상할 수도 있다.

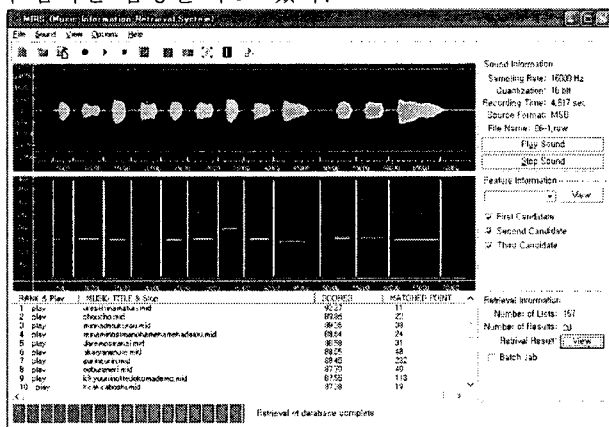


그림 10. 디지털 오디오 콘텐츠 검색 시스템의 프로토타입

7. 결 론

사용자가 기억하고 있는 일부분만의 멜로디를 자유로운 템포, 장조로 허밍 함으로써 찾고자 하는 오디오에 대한 정보의 검색이 가능한 오류에 강한 실질

적인 내용기반 질의법을 이용한 오디오 콘텐츠 검색 시스템을 구현하였다.

본 논문에서는 사용자의 입력 허밍의 오류 및 신호 처리시 발생하는 오류 등에 강한 검색 방법을 제안하였다. 실제 사용자의 입력 허밍은 개인차에 의한 템포, 장조의 변화와 애매한 기억에 의한 음표의 삽입/탈락 등의 여러 가지 요인으로 인해 데이터베이스 곡과 다른 입력이 들어오게 된다. 이러한 문제점을 해결하기 위해서 최적의 상대값을 DP 패스에 따라 다이내믹하게 취하는 방법을 제안하였다. 한편 입력 허밍이 완벽하다고 할지라도, 허밍으로부터 정확한 피치 정보를 얻기가 어렵다. 이러한 피치추출 오류를 고려하여 복수피치추출 방법을 제안하였다. 제안된 방법은 3개까지의 복수피치후보를 사용할 경우 99.7%의 검색 성능을 보였다. 또한 유사도 측정 알고리즘에 있어서 제안된 특징량을 효율적으로 사용하고 피치추출 오류에도 강한 매칭을 위해 DP평면의 공간을 3차원으로 확장하였다.

평가실험에서 제안된 방법의 경우 top-10의 검색정도는 94.1%의 결과를 얻었다. 이것은 종래 AIR 시스템보다 우수한 성능의 결과를 보였다.

[참 고 문 헌]

- [1] D. Feng, W.C.Siu and H. Zhang, "Multimedia Information Retrieval and Management: Technological Fundamentals and Applications," Springer-Verlag, 2003.
- [2] T. Kageyama, K. Mochizuki and Y. Takashima, "Melody Retrieval with Humming," In Proc. Int. Computer Music Conference, 1993.
- [3] A. Ghias, J. Logan, D. Chamberlin and B. C. Smith, "Query By Humming: Musical Information Retrieval in an Audio Database," In Proc. ACM Multimedia, 1995.
- [4] S. Pauws, "CubyHum: A Fully Operational Query by Humming System," In Proc. ISMIR 2002, 2002.
- [5] Toshihiko Shirokaze, Shozo Makino and Ken'iti Kido, "Extraction of Fundamental Frequency Using Temporal Continuity over an Input Speech," Trans. IEICE Vol.73-A, No.9 pp.1537-1539, 1990.
- [6] L. R. Rabiner and B. H. Juang, "Fundamentals of Speech Recognition," Prentice-Hall, 1993.
- [7] Rodger J. McNab, Lloyd A. Smith and Ian H. Witten, "Signal Processing for Melody Transcription," In Proc. of the 19th Australasian Computer Science Conference, 1996.
- [8] R. J. McNab, L. A. Smith, I. H. Witten, C. L. Henderson and S. J. Sunningham, "Toward the digital music library: tune retrieval from acoustic input," In Proc. ACM Digital Libraries, Bethesda, 1996.
- [9] Tomonari Sonoda, Makataka Goto and Yoichi Muraoka, "WWW-based Music Retrieval System," pp.343-352, In Proc. ICMC'98, 1998.
- [10] J. S. Roger Jang, H. Lee and J. Chen, "Super MBox: An Efficient/Effective Content-based Music Retrieval System," In the ninth ACM Multimedia Conference, pp. 636-637, 2001.