
음악 구조의 패턴에 기반을 둔 다음(Polyphonic) 피아노 솔로 음악으로부터의 멜로디 추출

Extracting Melodies from Polyphonic Piano Solo Music Based on Patterns of Music Structure

최윤재, Yoonjae Choi*, 이호동, Hodong Lee**, 이호준, Ho-Joon Lee***, 박종철, Jong C. Park****

요약 인터넷의 발달로 사람들이 많은 음악을 쉽게 접할 수 있게 된 만큼 음악 검색 서비스나 음악 추천 서비스와 같은 기능을 위한 응용 시스템이 주목 받고 있다. 이러한 시스템들이 서비스를 제공하기 위해서는 일반적인 음악의 형식인 다음(Polyphonic) 음악에서 멜로디를 추출하는 과정이 필수적이다. 본 논문에서는 다음 음악의 표현 능력이 매우 뛰어난 악기 중 하나인 피아노의 음악으로부터 멜로디를 추출하는 방법을 제안한다. 피아노 음악은 다음의 복잡도가 매우 크기 때문에 피아노 음악에서 멜로디를 추출하는 방법을 연구함으로써 여러 악기로 연주한 일반적인 다음 음악에서 멜로디를 추출하는 데 도움을 줄 수 있으리라 기대한다.

Abstract Thanks to the development of the Internet, people can easily access a vast amount of music. This brings attention to application systems such as a melody-based music search service or music recommendation service. Extracting melodies from music is a crucial process to provide such services. This paper introduces a novel algorithm that can extract melodies from piano music. Since piano can produce polyphonic music, we expect that by studying melody extraction from piano music, we can help extract melodies from general polyphonic music.

핵심어: *Melody Extraction, Piano, Polyphonic Music, Music Information Retrieval*

본 논문은 2008 년 21 세기 프론티어 연구개발 사업(인간 기능 생활지원 지능로봇 기술개발사업)을 통하여 지식경제부의 지원을 받았음.

*주저자 : 한국과학기술원 전산학과 석사과정 e-mail: yjchoi@nlp.kaist.ac.kr

**공동저자 : 한국과학기술원 전산학과 Post Doctor e-mail: hdlee@nlp.kaist.ac.kr

***공동저자 : 한국과학기술원 전산학과 박사과정 e-mail: hojoon@nlp.kaist.ac.kr

****교신저자 : 한국과학기술원 전산학과 교수; e-mail: park@nlp.kaist.ac.kr

1. 서론

최근 인터넷이 발달하면서 사람들은 많은 음악을 매우 쉽게 접할 수 있다. 이에 따라 음악을 의미 있게 저장하고 검색하거나 추천해줄 수 있는 시스템의 필요성도 함께 부각되고 있다. 음악 정보 검색(Music Information Retrieval)이나 허밍 질의(Query by Humming), 내용 기반 음악 추천(Content-based Music Recommendation)이 주목 받는 것도 이러한 이유 때문이다[1,2,3]. 멜로디는 음악을 구분하기 쉽게 해주고 사람들로 하여금 음악을 기억하기 쉽게 해준다고 알려져 있는데[4], 이에 따라 음악 기반 응용 시스템에서는 멜로디를 이용한 서비스를 제공하는 경우가 많다. 특히 음악 정보 검색 시스템이나 허밍 질의 시스템의 경우, 사람들의 편의를 돕기 위해서 사람들이 기억하기 쉽고 음악의 복잡도를 대폭 낮춘 멜로디를 질의로 받는 경우가 많다. 그러나 우리가 듣는 일반적인 음악은 다음(Polyphonic)¹ 음악으로 멜로디²에 해당하는 음들 외에도 반주(Accompaniment)에 해당하는 음들을 포함하고 있다. 그러므로 음악에서 멜로디를 추출하는 것은 사람들에게 도움이 되는 음악 기반 응용 시스템의 성능 향상에 필수적인 과정이라고 할 수 있다[2,5,6].

본 논문에서는 음악에서 멜로디를 추출하는 방법에 대해 논의한다. 멜로디를 추출할 대상으로는 피아노 솔로 음악을 고려한다. 음악을 만들 수 있는 악기는 종류가 매우 많지만 그 중에서 피아노는 가장 넓은 음역을 가지고 상당히 복잡한 다음 음악도 단일 악기만으로 표현할 수 있다. 또한 오케스트라가 연주하는 교향곡을 피아노로 편곡할 수 있을 만큼 피아노는 표현의 정도가 크다. 그렇기 때문에 피아노의 음악에서 멜로디를 추출할 수 있다면 그 방법을 다른 음악에도 적용해서 멜로디를 추출하는 데 큰 도움이 되리라 기대한다.

앞으로 현재까지의 멜로디 추출에 관한 연구들의 결과를 살펴보고 그들의 간단한 특징과 개선점을 알아본다. 그리고 본 논문이 제안하는 방법을 소개한 후 20 곡의 피아노 음악에 그것을 적용해보고 결론 및 향후 계획을 논한다.

2. 관련 연구

¹다음(Polyphonic) 음악은 두 가지 이상의 음들이 동시에 발생하는 음악을 뜻한다. 이와 달리 단음(Monophonic) 음악은 한 번에 두 개 이상의 음들이 발생하지 않는 음악이다. 예를 들어 한 사람의 노래는 단음 음악에 해당한다.

²여기서 멜로디라 함은 음표들의 연속 (Sequence)인데 한 번에 두 개 이상의 음이 발생하지 않는 것을 뜻한다[5]. 즉, 멜로디는 단음 음악이다.

음악에서 멜로디를 추출하는 방법은 크게 두 가지로 나눌 수 있다. 먼저 MP3 나 Wave 포맷의 파일로부터 오디오 정보를 받아들여 신호처리과정을 통해 멜로디를 추출할 수 있다. 이와 같은 방법의 대표적인 예로는 Goto[7]와 Klapuri[8]가 있다. 다른 방법으로는 미디(MIDI)나 NIFF, ESaC 와 같이 음악을 상징적인 기호로 표현한 파일을 받아들여서 멜로디를 추출 하는 것이 있다. 이러한 방향으로서는 여러 종류의 시도가 있었지만 현재까지 Uitdenbogerd 와 Zobel[5]의 Skyline Algorithm, 또는 그것의 변형된 알고리즘이 가장 성공적인 예라고 할 수 있다[9].

오디오 시그널에서 멜로디를 추출하는 대표적인 방법으로는 원본 파일을 입력 받아서 발생한 음표들을 받아 적은 후(Polyphonic Audio Transcription)에 다시 그 음표들로부터 멜로디를 추출하는 알고리즘을 적용하는 방식과[10], 멜로디라고 추측되는 주파수 대역을 추적하는 방식(F0 Estimation)이 있다[7,8]. 이러한 방식들은 실제로 녹음된 음악을 상대로 멜로디를 추출할 수 있다는 장점이 있지만 음악학적인 지식을 효과적으로 사용하기 어렵다는 단점이 있다.

본 논문에서는 멜로디의 효율적인 추출을 위해 음악학적인 지식을 사용하는 것이 중요하다고 판단하여 음표로 표현되어 있는 음악으로부터 멜로디를 추출하는 방법에 대해 집중하여 논의하고자 한다. 그러므로 오디오 시그널에서 멜로디를 직접 추출하는 방식이 아닌 상징적인 기호로 구성된 파일에서 추출하는 방식을 택하기로 한다.

앞에서 살펴본 바와 같이 상징적인 기호로부터 멜로디를 추출하는 방식인 Skyline Algorithm 에서는 두 가지 원칙을 가진다. 첫째는 여러 음이 동시에 발생하는 경우 가장 높은 음이 멜로디에 속한다는 것이고, 둘째는 지속시간이 긴 음의 경우 다른 음이 시작하는 시점이 지나서까지도 지속될 수 있는데, 이때 긴 음의 지속시간을 의도적으로 줄여서 다음에 발생하는 음과 겹치지 않게 한다는 것이다. 그러나 이 알고리즘에는 크게 두 가지 문제가 있다. 하나는 저음 부분이 멜로디인 경우를 제대로 인식하지 못한다는 것이고, 다른 하나는 음의 길이를 줄여가면서 최대한 많은 음을 멜로디에 포함시키기 때문에 원래의 멜로디가 변질될 수 있다는 것이다. 이러한 Skyline Algorithm 의 단점들을 보완하기 위한 다른 연구들이 지속적으로 있었는데, 두드러진 차이점으로는 대부분 하나의 미디 파일에 여러 트랙이나 채널이 있다고 가정하고 그 중에서 멜로디를 담고 있는 트랙이나 채널을 선택하는 데 집중한다는 것이다[2,6,11,12]. 반면 Skyline Algorithm 은 모든 트랙과 채널의 음표를 하나의 트랙으로 압축한 후에 멜로디를 추출하기 때문에 보편성이 훨씬 뛰어나다. 또 Skyline Algorithm 이외의 방식을 사용하는 연구에서는 비교적 구조가 동일한 음악을 택하거나, 음악, 연주자, 또는 미디 파일에 따라 달라질 수 있다는 가정을 세워서 접근한다[2].

본 논문에서는 Skyline Algorithm 이 사용하는 원칙을 활용하여 여러 유형의 음악에 동일한 원칙을 사용하기 보다는 각 유형에 맞는 원칙을 사용해서 멜로디를 추출하는 방식을 살펴 본다.

3. 멜로디 추출 방법

3.1 기존 방법의 문제점 및 해결 방안

음악에서 멜로디를 추출하는 것은 연속된 시간상에서 입력된 여러 음들 중에서 하나의 대표음을 선택하거나, 대표음이 없는 경우 아무것도 선택하지 않는 과정의 연속으로 볼 수 있다. 음을 선택하는 과정에서는 특정한 기준을 사용하는데, Skyline Algorithm 의 예에서도 알 수 있듯이 어느 상황에서나 적용 가능하며 음악적으로 의미 있는 원칙을 찾기는 매우 어렵다. 예를 들어 그림 1³과



그림 1. 쇼팽의 군대 폴로네이즈

같은 음악의 경우 Skyline Algorithm 을 그대로 적용해도 멜로디를 추출하는 데 특별한 문제가 없다. 동시에 발생하는 음들이 모두 같은 지속 시간을 갖고 있으므로 어떤 음도 임의로 지속시간을 줄일 필요가 없고 항상 가장 높은 음이 멜로디이기 때문이다. 그런데 그림 2 와 같은 음악에서는 하단의 오선지에 표시된



음들이 멜로디가 되고 상단의 오선지에 표시된 음들이

그림 2. 베토벤의 월광 3악장

반주가 된다. 그러므로 동시에 발생하는 음들 중에서 높은 음이 멜로디에 속한다는 원칙은 이 상황에서는 적용 가능하지 않다. 그리고 하단의 멜로디가 진행되는 도중에 상단의 반주들이 진행되고 있기 때문에 이 경우 Skyline Algorithm 을 그대로 적용한다면 그림 3에서 보이는 바와 같이 상단의 반주들이 멜로디에 포함되게 된다. 따라서 그림 2 와 같은 경우에는 Skyline Algorithm 의 적용이



³ 본 논문에서는 편의를 위해 악보에서 멜로디를 타원 안의 음으로 표시하였다.

그림 3. 베토벤의 월광 3악장에서 Skyline Algorithm 을 이용해서 멜로디를 추출한 결과

그러므로 음악에서 멜로디를 추출하기 위해서는 각 음악의 유형에 따라 적합한 원칙을 사용하는 것이 정확한 멜로디를 추출하는 데 필요하다. 예를 들어 그림 2 의 음악과 같은 경우 다음과 같은 원칙을 이용해서 멜로디를 분류할 수 있다.

1. 동시에 발생하는 음의 경우 보다 낮은 음이 멜로디에 속한다.
 2. 선택된 낮은 음이 지속되는 기간 동안 입력되는 음은 모두 반주이므로 멜로디에 포함시키지 않는다.
- 위의 두 가지 원칙을 이용하면 그림 2 와 같은 음악에서 저음부의 멜로디를 잘 추출할 수 있다.

또 하나의 예를 들면 그림 4 와 같은 음악은 다음과 같은 원칙을 이용해서 멜로디를 분류할 수 있다.



그림 4. 리스트의 사랑의 꿈

1. 동시에 발생하는 음들 중에서 가장 긴 지속 시간을 갖는 음이 멜로디에 속한다.
2. 동시에 발생하는 음들 중에서 두 음 이상이 같은 길이의 가장 긴 지속 시간을 갖는다면 그 중에서 가장 고음이 멜로디에 속한다.
3. 멜로디에 속하는 음이 지속되는 시간 동안 입력되는 음들은 모두 반주이므로 멜로디에 포함시키지 않는다.

이처럼 음악의 유형에 맞춰 적절한 원칙을 사용하여 멜로디를 추출한다면, 음악의 구조적 특성을 고려하지 않고 동일한 원칙만을 사용하는 기존의 알고리즘들에 비해 보다 정확한 결과를 얻을 수 있을 것이다.

3.2 추출 방법의 구현

본 논문에서는 다음과 같은 네 가지 음악 유형에 각각 다른 멜로디 추출 방법을 제안한다.

1. 멜로디를 주로 오른손으로 연주하는 경우 (MR)
2. 멜로디를 주로 왼손으로 연주하는 경우 (ML)
3. 오른손과 왼손이 각기 다른 반주를 하면서 멜로디가 따로 존재하는 경우 (MC)
4. 동시에 발생하는 음들의 지속시간이 모두 동일하고 그 외에 음들 간의 겹침이 없는 경우 (ME)

추출 알고리즘은 그림 5 와 같은 순서로 기술될 수 있다. 그림 5 의 순서도에서 각 유형에 따라 네 번째와 다섯

번째 단계가 차이가 있다. 각 유형에 따른 네 번째와 다섯 번째 단계를 수도코드로 작성하면 다음과 같다.

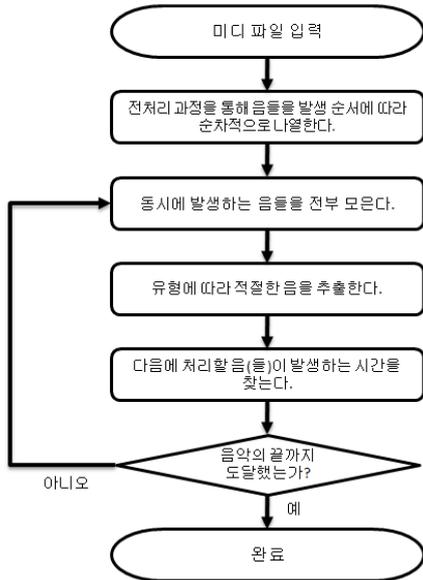


그림 5. 멜로디 추출 알고리즘

3.2.1 MR을 위한 알고리즘 (MRA)

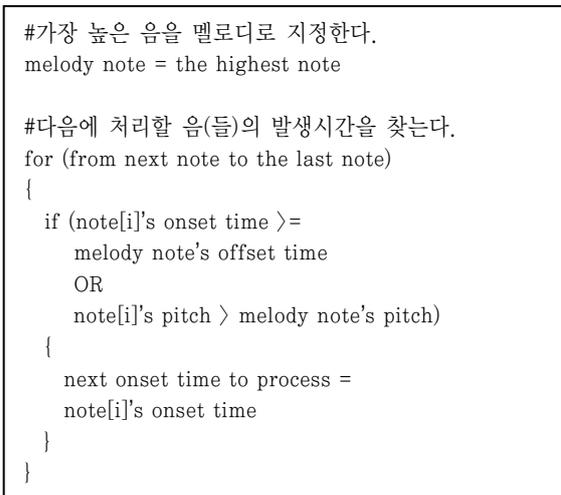


그림 6. MRA의 수도코드

그림 6의 알고리즘의 for 문에서 멜로디로 지정된 음보다 더 높은 음이 발생한 경우를 찾는 이유는 그림 7을 보면 알 수 있다. 그림 7의 두 번째 마디 시작 부분에서는 동시에 발생하는 음이 없으므로 네모로 표시된 음이 멜로디로 지정되는데, 표시된 음은 2분음표이므로 정작 중요한 상단 오선지의 멜로디에 해당하는 음들마저도 가려버린다. 그러므로 위와 같은



그림 7. 드뷔시의 아라베스크 1번

방법을 택하지 않는다면 멜로디에 해당하는 음들을 모두 잃게 될 위험부담이 있다.

3.2.2 ML을 위한 알고리즘 (MLA)

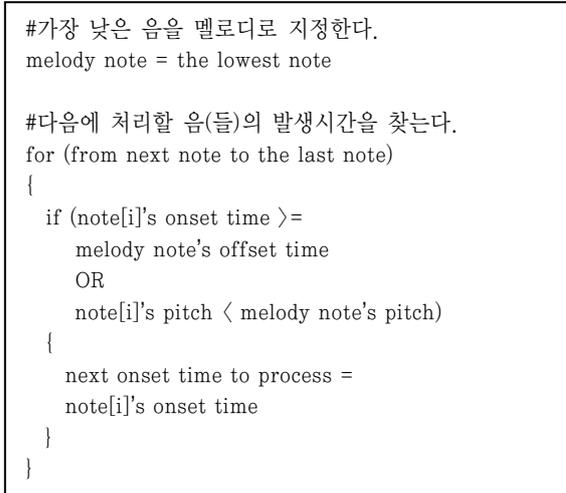


그림 8. MLA의 수도코드

그림 8의 알고리즘이 보이는 것과 같이 MLA 역시 MRA와 같은 이유로 for 문에서 멜로디로 지정된 음보다 더 낮은 음이 발생하는 경우를 찾는다.

3.2.3 MC를 위한 알고리즘 (MCA)

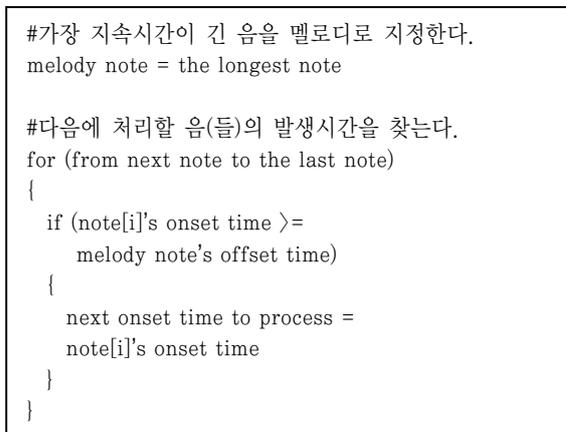


그림 9. MCA의 수도코드

MCA는 MRA나 MLA와 달리 음의 높낮이를 멜로디의 추출 기준으로 삼지 않으므로 for 문에서 하나의 조건만을 본다.

3.2.4 ME를 위한 알고리즘 (MEA)

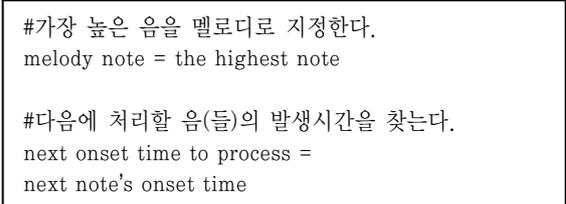


그림 10. MEA의 수도코드

MEA는 Skyline Algorithm을 차용하였다. ME의 경우에는 동시에 발생하는 음들 외에는 서로 겹치는

경우가 없으므로 지속시간을 고려할 필요가 없다. 그렇기 때문에 다른 알고리즘들과 달리 MEA 에는 다음에 처리할 음(들)의 발생시간을 찾는 데 for 문이 필요 없다.

4. 실험 및 결과

위에서 소개한 네 가지의 알고리즘을 이용해서 20 곡의 피아노 음악으로부터 멜로디를 추출하였다. 실험에 사용한 피아노 음악은 표 1 과 같다. 모든 음악은 발췌음악(excerpt)으로서 10 초부터 1 분여까지 길이가 다양하다. 음악 샘플은 Noteworthy Composer 를 이용해서 midi 파일로 제작했다⁴⁵.

표 1. 샘플로 사용한 음악 목록

번호	음악 이름
1	베토벤, 피아노 소나타 Op.2 No.1
2	베토벤, 월광 소나타 2 악장
3	베토벤, 월광 소나타 3 악장
4	베토벤, 비창 1 악장
5	베토벤, 비창 2 악장
6	쇼팽, 즉흥 환상곡
7	쇼팽, 발라드 1번
8	쇼팽, 군대 폴로네이즈
9	쇼팽, 에튀드 Op.10 No.3 이별의 곡
10	쇼팽, 녹턴 Op.9 No.2
11	리스트, 사랑의 꿈
12	드뷔시, Clair de Lune
13	드뷔시, 아라베스크 1번
14	라흐마니노프, 파가니니 주제에 의한 광시곡 18번 변주곡
15	조지 윈스턴, Joy
16	조지 윈스턴, 캐논 변주곡
17	요코 칸노, Mast in the Mist
18	요코 칸노, Piano Bar
19	스티브 바라캣, Rainbow Bridge
20	막심 마라비차, Claudine

⁴ 사용한 음악은 <http://nlpstar.kaist.ac.kr/~yjchoi/midi.zip>에서 확인할 수 있다.

⁵ 정답 세트에 해당하는 해당하는 멜로디만 담은 음악은 <http://nlpstar.kaist.ac.kr/~yjchoi/midi.zip>에서 확인할 수 있다.

20 곡의 음악을 MR, ML, MC, ME 의 네 가지 유형으로 분류해보면 표 2 와 같이 나눌 수 있다. 피아노 음악에서는 오른손이 멜로디를 연주하고 왼손이 반주를 연주하는 것이 기본적인 형태이므로 MR 유형의 음악이 많은 것을 확인할 수 있다

표 2. 유형 별 음악 목록

유형	음악 번호
MR	10, 12, 13, 14, 15, 16, 17, 19, 20
ML	1, 3, 6
MC	5, 9, 11
ME	2, 4, 7, 8, 18

각 음악 샘플에 네 가지의 알고리즘을 적용해 본 결과는 표 3 부터 표 6 과 같다. 평가는 정확도(Precision)와 재현율(Recall)을 사용했다. 정확도와 재현율의 측정 방법은 각각 수식 1, 수식 2 와 같다.

$$\text{정확도} = \frac{\sum(\text{정답 멜로디와 일치하는 음의 개수})}{\sum(\text{시스템이 추출한 멜로디의 음의 개수}) * 100} \quad (1)$$

$$\text{재현율} = \frac{\sum(\text{정답 멜로디와 일치하는 음의 개수})}{\sum(\text{정답 멜로디의 음의 개수}) * 100} \quad (2)$$

실험 결과에서 나타난 것과 같이 재현율은 각 유형에 적합한 알고리즘을 사용한 경우 전반적으로 높은 수치를 기록하고 있다. 본 연구에서는 추출된 멜로디의 정확성을 중요한 사항이라 판단했기에 정확도를 중심으로 결과를

표 3. MR 유형 음악들의 실험 결과

음악 번호	MRA	MLA	MCA	MEA
10	90.63	2.04	79.31	55.77
12	81.20	8.70	32.00	68.42
13	90.74	17.65	64.13	59.30
14	89.39	0.00	31.58	49.58
15	90.91	0.00	88.10	50.63
16	94.03	0.00	63.04	74.16
17	95.83	0.00	93.62	49.46
19	95.89	2.78	77.42	65.42
20	95.00	12.57	84.21	31.32
평균정확도	91.51	5.19	68.16	56.01
평균재현율	94.21	7.61	60.52	95.10

자세히 분석한다.

표 3 을 보면 MR 유형의 음악에는 다른 알고리즘들에 비해서 MRA 가 가장 좋은 성능을 보인다는 것을 알 수 있다. Skyline Algorithm 에 해당하는 MEA 의 경우 평균재현율은 매우 높지만 평균정확도가 떨어지는 것이 확연히 드러난다.

표 4. ML 유형 음악들의 실험 결과

음악 번호	MRA	MLA	MCA	MEA
1	0.00	92.00	82.61	0.00
3	0.00	80.39	58.82	0.00
6	0.00	40.00	40.00	0.00
평균정확도	0.00	70.80	60.48	0.00
평균재현율	0.00	97.44	82.70	0.00

표 3 에서 나타나듯이 MRA 가 전반적으로 좋은 결과를 보이기는 하지만 ML 유형의 음악인 1, 3, 6 번에 적용했을 때에는 멜로디를 거의 추출하지 못하는 문제점을 보이고 있다. 표 4 가 보여주듯이 같은 ML 유형에 속하는 1, 3, 6 번 음악은 MLA 를 사용했을 때 훨씬 나은 결과를 보인다. 그런데 6 번의 경우는 1, 3 번에 비해서 그 정확도가 확연히 떨어진다. 6 번 음악은 그림 11 과 같은



그림 11. 쇼팽의 즉흥환상곡

악보를 가지고 있는데 그림을 보면 알 수 있듯이 이 음악에서 멜로디는 왼손만으로 연주되고 있다. 하지만 MLA 를 이용해서 멜로디를 추출해도 정확도가 40 퍼센트밖에 나오지 않는 이유는 midi 파일을 작성한 Noteworthy Composer 에서 찾을 수 있다. Noteworthy Composer 는 음의 겹침 현상을 최대한 줄이기 위해서 모든 음들의 지속시간을 조금씩 단축시키는데 그림 11 의 악보를 보면 알 수 있듯이 상단 오선지의 음은 매우 빠르게 전개되는 반면에 하단 오선지의 음은 하나하나가 매우 지속시간이 길다. 이 차이 때문에 각 음들의 지속시간이 단축되는 정도가 달라지고, 지속시간이 긴 하단 오선지의 멜로디음들이 원래의 지속시간보다 일찍 소멸되기 때문에 생긴 시간적 틈새로 상단 오선지의 빠른 반주음들이 끼어들어 정확한 멜로디를 뽑기 어려워지는 것이다. Noteworthy Composer 의 이 특성은 다른 음악들의 멜로디 추출 결과에도 영향을 미치겠지만 6 번과 같이 전체 성능에 큰 영향을 끼치는 경우는 없었던 것으로 확인할 수 있었다.

MCA 는 MC 유형에 속하는 5, 9, 11 번 음악에서 좋은 성능을 보인다. 그런데 표 5 를 보면 알 수 있듯이 5, 9 번의 경우에는 11 번과 달리 MRA 가 더욱 높은

표 5. MC 유형 음악들의 실험 결과

음악 번호	MRA	MLA	MCA	MEA
5	97.96	7.27	85.00	38.76
9	100.00	2.04	70.97	59.70
11	16.20	24.21	74.51	16.08
평균정확도	71.39	11.17	76.83	38.18
평균재현율	84.88	23.62	73.89	86.19

정확도를 보이고 있다. 평균재현율 또한 MRA 가 더 높은 수치를 보이고 있다. 이것은 5, 9 번 음악의 특성에서 비롯되는 것인데, 5, 9 번의 유형은 11 번과 같지만 그림 12, 13 를 보면 알 수 있듯이 멜로디가 언제나 가장 높은 음에 있기 때문에 MRA 로 더욱 잘 처리 될 수 있다. 이로 인해서 5, 9 번에는 MRA 를 적용해도 무리가 없으며 오히려 MCA 보다 좋은 결과가 나오는 것이다. 하지만 11 번의 경우에는 그림 4 와 같이 멜로디에 해당하는 음들이 낮은 음들 쪽에도 있기 때문에 MRA 로는 멜로디를 성공적으로 추출할 수 없다.



그림 12. 베토벤의 비창 2 악장



그림 13. 쇼팽의 이별의 곡

표 5. ME 유형 음악들의 실험 결과

음악 번호	MRA	MLA	MCA	MEA
2	100.00	0.00	100.00	100.00
4	100.00	7.69	92.31	100.00
7	100.00	0.00	97.06	100.00
8	100.00	0.00	93.33	100.00
18	100.00	38.10	92.86	97.83
평균정확도	100.00	9.158	95.11	99.57
평균재현율	98.852	8.384	90.456	98.852

표 6 이 나타내듯이 ME 유형에 속하는 음악들은 Skyline Algorithm 을 그대로 차용해도 멜로디를 추출하는 데 거의 문제가 없다. 흥미로운 사실은 MRA 가 MR 유형의 음악들뿐만 아니라 ME 유형의 음악에까지도 높은 성능을 보인다는 것이다. ME 유형의 음악은 그 음들의 지속시간이 단순하고 언제나 가장 높은 음이 멜로디이므로 MRA 를 적용해도 무리가 없다. 그런데 더욱 흥미로운 점은 MRA 가 MEA 보다 언제나 정확도가 더 높다는 것이다. MEA 의 경우 동시에 발생하는 음들 이외의 음들 간의 겹침을 고려하지 않기 때문에 예외 상황이 발생할 경우 MRA 에 비해 정확도가 떨어질 수 있다. 8 번 음악의 악보인 그림 14 의 경우 이런 예를 보여주고 있다. 하지만 MRA 가 MEA 보다 항상 높은 성능을 보이는 것은 ME 유형의 음악을 상대로 할 때뿐이고, 다른 유형의 음악에서는 MEA 가 더 좋은 성능을 보일 수도 있다. 이 문제는 추후에 더욱 많은 유형의 음악을 다루면서 연구되어야 할 것이다.



그림 14. 쇼팽의 군대 플로네이즈 (네모로 표시한 부분은 상단의 오선지와 하단의 오선지의 음들의 지속시간이 동일하지 않다)

한 가지 주목할 만한 점은 위의 실험 결과들이 보여주듯이 MCA 는 MC 유형의 음악뿐만 아니라 다른 유형의 음악에서도 좋은 성능을 보인다는 것이다. 이것은 테스트 샘플로 사용한 음악들 중 많은 경우 동시에 발생하는 음들 중에서 가장 긴 음이 멜로디에 해당하기 때문이다. 예를 들어 1 번 음악과 20 번 음악은 각각 ML, MR 에 속하는 음악이지만 동시에 발생하는 음들 중에서 가장 지속시간이 긴 음이 멜로디인 경우가 빈번하다는 공통점을 갖고 있기 때문에 MCA 에서도 멜로디 추출이 어느 정도 이루어지는 것이다. 이러한 공통점은 그림 15, 16 에서 확연히 드러난다.



그림 15. 베토벤의 피아노 소나타 Op.2 No.1



그림 16. 막심 마라비차의 Claudine

5. 결론 및 향후 계획

본 논문에서는 음악 구조의 패턴에 기반을 두어서 피아노 음악으로부터 멜로디를 추출하는 방법을 소개했다. 그리고 주어진 음악을 네 가지 유형으로 분류한 후에 각각에 맞는 알고리즘을 구현하여 20 개의 피아노 음악에 적용시켜 보았다. 비록 실험에 사용된 샘플의 개수가 충분하다고 말하기는 어렵지만 음악의 패턴에 따라 다른 원칙을 이용하면 멜로디를 더욱 정확하게 추출할 수 있음을 확인했다. 이런 멜로디 추출 방법을 앞으로 더욱 보완, 개발하면 다양한 장르의 음악으로부터 정확하게 멜로디를 추출하여 허밍 질의 시스템의 데이터베이스 구축이나, 멜로디 기반 음악 비교 모듈의 성능 향상에 큰 도움이 될 수 있으리라 기대한다.

향후에는 본 연구에서 사용한 음악 구조의 패턴들을 더욱 음악학적으로 의미 있게 세분화하고, 다루지 않았던 패턴들도 정의함으로써 제안한 방법을 이용해서 보다 다양한 음악을 대상으로 정확하게 멜로디를 추출할 수 있도록 할 계획이다. 그리고 하나의 음악에도 여러 패턴이 존재할 수 있으므로 음악을 임의의 단위로 잘라서 각 단위가 어떤 패턴에 속하는지 자동적으로 파악하는 시스템을 구현해서 각 유형에 맞는 적절한 멜로디 추출기법을 적용할 수 있도록 할 계획이다.

참고문헌

- [1] Dannenberg et al., "The MUSART Testbed for Query-by-Humming Evaluation," *Computer Music Journal*, pp. 41-47, 2004.
- [2] Velusamy, Thoshkanna, Ramakrishnan, "A Novel Melody Line Identification Algorithm for Polyphonic MIDI Music," *Proc. MMM Conference*, pp. 248-257, 2007.
- [3] Cano, Koppenberger, Wack, "Content-based Music Audio Recommendation," *Proc. ACM international conference on Multimedia*, pp. 211-212, 2005.
- [4] Selfridge-Field, "Conceptual and Representational Issues in Melodic Comparison," W.B. Hewlett, & E. Selfridge-Field (Eds.), *Melodic Similarity - Concepts, Procedures, and Applications*, Cambridge, Massachusetts: MIT Press, 1998.
- [5] Uitdenbogerd, Zobel, "Manipulation of Music For Melody Matching," *Proc. ACM international conference on Multimedia*, pp. 235-240, 1998.
- [6] Shan, Kuo, Chen, "Music Style Mining And Classification By Melody," *Proc. IEEE International Conference on Multimedia*, Vol. 1, pp. 97-100, 2002.

- [7] Goto, "A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Communication*, pp. 311-329, 2004.
- [8] Ryyanen, Klapuri, "Transcription of the Singing Melody in Polyphonic Music," *Proc. ISMIR*, pp. 222-227, 2006.
- [9] Isikhan, Ozcan, "A Survey of Melody Extraction Techniques For Music Information Retrieval," *CIM08 Abstract Proceedings*, pp. 82-83, 2008.
- [10] Suyoto, Uitdenbogerd, Scholer, "Searching Musical Audio Using Symbolic Queries," *Proc. IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 16, pp. 372-381, 2008.
- [11] Chai, "Melody Retrieval On The Web," MS Thesis, Massachusetts Institute of Technology, 2001.
- [12] Ozcan, Isikhan, Alpkocak, "Melody Extraction on MIDI Music Files," *Proc. IEEE International Symposium on Multimedia*, pp. 414-422, 2005.