피아노 연주 평가 프로그램을 위한 타격 시간 검출

이준수, 한기영, 백성준, 성굉모
서울대학교 전기·컴퓨터공학부

Detection of Attack Time in Developing the Scoring Program for Piano Music

Jun-Soo Lee, Ki-Young Han, SeongJoon Baek, Koeng-Mo Sung
School of Electrical Engineering & Computer Science, Seoul National University

요약

실제 피아노 연주 녹음에서 연주된 음표를 검출하는 과정은, 녹음된 데이터만으로 원래의 악보를 복원하거나 같은 악보로 연주한 서로 다른 연주를 비교 평가하는데 있어 필수적인 과정이다. 본 논문에서는 피아노 연주 평가 프로그램에 적용될 알고리듬 구상의 첫 단계로서 타격 시간을 찾는 알고리듬을 제안하고, 실제의 연주 sample을 이용한 analysis와 기존의 다른 방법과의 비교를 통하여 그 성능과 효용성을 점검하였다. 또한, 이 알고리듬을 실제 연주 평가 프로그램에 적용시켜 완성된 프로그램에 대해서도 간략히 소개하였다.

1. 서론

연주 평가 프로그램이란, 평가하고자 하는 연주를 읽어들여 연주가 그 곡에 대한 원래의 악보에 비해 어느 정도 충실히 연주했는지를 판단하여 수치화 시키는 프로그램이다. 이러한 평가를 하기 위해서는 읽어들인 음악 연주 파일로부터 연주된 음들을 찾아내어 다시 악보 상에 표시하는 것이 필요하다. 이와 같이 실제 악기로 연주된 음악을 자동으로 다시 원래의 악보로 transcription하는 연구는 이미 오랜 동안 이루어져 왔다 [1]-[3].

이를 위해서는 전체 음악 속에 들어있는 음들을 찾아내고 그 음이 가지고 있는 정보를 정확히 알아내는 과정이 필요하다. 그러한 과정 중에서 가장 먼저 해결해야 하는 과정이 음의 시작 시간을 찾는 일이다. 음의 시작 시간을 찾는다는 것은 우선 그 음의 존재 여부를 판단하는 기준이 될 뿐만 아니라 그 음의 다른 정보들 즉, 음높이와 세기 등을 추정하는 process를 하기 위한 위치상의 기준이 되기 때문이다. 이러한 음의 시작 시간을 일반적으로 onset time이라고 하고, 특히 피아노와 같이 타격을 하여 소리를 내는 악기에 대해서는 타격 시간(attack time)이라고 한다.

피아노가 해머로 현을 타격하여 소리를 내는 악기라는 사실은 피아노 음악으로부터 음의 타격 시간을 알아내는데 있어서 매우 중요한 요소가 된다. 음의 시작부와 지속부가 명확히 구분되지 않는 음성이나 기타 관악기 등과 달리, 피아노는 현을 때리는 방식이므로 음이 시작되는 부분과 시작된 음이 지속되는 부분이 명확하게 구분이 된다. 특히 타격시에 일어나는 시간 축에서의 급격한 음의 amplitude 변화와 주파수 축에서의 noise성분은 피아노 음의 타격 시간을 검출하는 판단 조건으로 사용되어 왔다. 이를 이용하여 Bandpass filtering을 통한 후 high frequency power의 envelope 변화를 기준으로 타격 시간을 구하는 알고리듬이 제안되었고, 그 외에 신호의 RMS power를 이용하여 타격 시간을 추정하는 알고리듬과 comb filter를 이용하여 타격음의 harmonics성분들에 대한 크기의 합을 통해 타격 시간을 구하는 알고리듬 등이 고안되었다 [2]. 또한 이러한 방법들을 복합적으로 사용하면 상당히 정확한 타격 시간을 구할 수 있었다. 그러나 피아노 연주 평가 프로그램을 제작하는 데 있어서는 계산 시간이라는 요소가 중요한 문제가 되기 때문에, 되도록 적은 계산량으로도 타격 시간을 검출해내는 알고리듬이 필요하게 되었다.

본 논문에서는 실제로 피아노 연주 평가 프로그램에 적용하기 위한, 간단하면서도 정확한 타격 시간 검출 알고리듬을 제안하여 그 원리를 분석함과 동시에, 기존의 다른 논문에서 제안된 알고리듬과의 성능 비교를 통하여 그

효용성을 검토하였다. 나아가 본 알고리듬을 토대로 하여 완성된 피아노 연주 평가 프로그램에 대하서도 간략히 소개하였다.

2. 제안된 방법

A. 알고리듬의 구성

제안된 방법은 기본적으로 피아노음의 amplitude envelope 특성을 이용하고 있다. 그림 1에서 보는 바와 같이 악기음은 크게 4가지 영역으로 구분하여 볼 수 있다[4]. 이 중에서 피아노의 경우는 attack section이 매우 가파른 성향을 가지며 decay section 또한 감쇄가 빠른 편이다. 이를 이용하여 신호의 크기의 변화율을 구함으로써 attack time을 알아낼 수 있다.

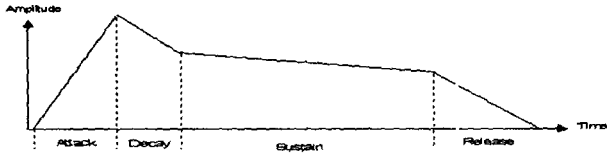

그림 1 피아노 악기음의 amplitude envelope

이번에 제안된 알고리듬에 대한 구성을 아래의 그림 2에서 보이고 있다.

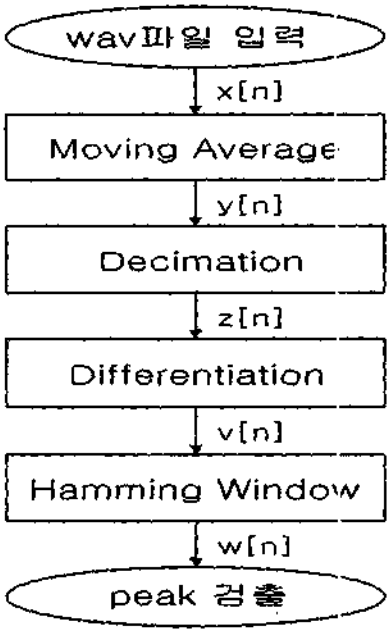

그림 2 알고리듬의 순서도

우선 피아노 연주가 녹음된 wav파일을 읽어들인다. 이번 실험에서는 22.05kHz의 sampling rate를 가지는 16bit 모노 신호에 대하여 실험을 행하였으나, 이는 언제든지 변화될 수 있다. 이렇게 입력된 신호는 기본적으로 모든 신호값이 -0.5에서 +0.5사이의 값을 갖도록 normalized되며 그렇게 입력된 신호가 그림 2에서 보여지는 x[n]이 된다.

그 다음은 어느 정도 완만한 신호 데이터의 magnitude 곡선을 얻기 위하여 moving average 과정을 거치는 단계이다. 이것은 일종의 lowpass filtering의 역할을 한다고 볼 수 있는데, 그 과정을 거쳐 얻어진 신호 y[n]은 아래의 식 1과 같다.

$$y[n]=\frac{1}{2L+1}\sum_{l=-L}^{L}x[n+L] \qquad \text{(식 1)}$$

그 다음 계산량을 적절한 수준으로 줄이기 위하여 프로그램의 다른 성능에 지장을 주지 않는 범위 내에서 y[n]을 식 2와 같이 decimation하여 신호 z[n]을 얻는다.

$$z[n]=y[Mn] \qquad \text{(식 2)}$$

다음은 z[n]의 differentiation 신호 v[n]을 구하는 과정이다. 타격 시간은 신호의 magnitude가 최대인 시점을 기준으로 하는 것이 아니고, 타격 직후 magnitude가 급격하게 상승하는 것을 기준으로 구해야 하기 때문에, 신호의 differentiation을 계산하여 이때 양의 peak값을 갖는 지점을 타격 시간으로 생각하기 위해서이다. v[n]은 간단히 z[n]과 z[n-1]의 차로서 얻어질 수 있다(식 3).

$$v[n]=z[n]-z[n-1] \qquad \text{(식3)}$$

위에서 말한 대로, 이렇게 해서 얻어진 신호 v[n]에서 양의 peak가 되는 지점이 타격 시간이 된다. 그런데 differentiation 하여 얻어진 신호 v[n]이 부드러운 곡선의 모양이 아니기 때문에 정확한 peak의 위치를 찾는데 어려움이 있다. 정확한 peak의 위치를 찾기 위하여 hamming window H[n]을 통과시켜 완만한 defferentiation 신호 x[n]을 얻는다(식 4). 이렇게 해서 최종적으로 얻어진 신호 w[n]에 대하여, w[n]의 양의 peak인 지점들을 구하면 타격 시간을 얻을 수 있다.

$$w[n]=z[n]*H[n] \qquad \text{(식 4)}$$

B. Parameter의 결정

위에서 설명한 과정을 통하여 타격 시간을 구하는데 있어서 중요한 것이 (식 1)이나 (식 2)에 나오는 L, M과 같은 parameter들의 값을 결정하는 것이다. 여기서 우리가 결정해야 하는 parameter로는 (식 1)의 moving average에 관계된 상수 L, (식 2)의 decimation 상수 M, 그리고 (식 4)에서 hamming window의 size 등이 있다. 물론 최종 신호 w[n]에서 양의 peak를 판정할 때 필요한 threashold 값도 결정해 주어야 한다. 이러한 parameter 들의 값을 결정하는 과정은 정확한 타격 시간을 구하는 데에 매우 결정적인 영향을 미친다.

우선 (식 1)의 moving average를 구하는 데 있어서, 최소한 지켜져야 할 것은 moving average를 하는 영역의 크기가 신호의 1/4 파장보다는 작아야 한다는 것이다. 즉, (2L+1)의 값이 우리가 취급하는 신호 중에서 가장 높은 주파수의 1/4 파장보다 작아야 moving average를 하는 의미가 있다고 볼 수 있다. 또한 계산량을 많이 늘리지 않으면서 효과를 어느 정도 볼 수 있는 조건과, 뒤에 거치는 decimation을 고려하여 L=5으로 결정하였다.

및 배로 decimation할 것인가를 결정하는 parameter M은 다음과 같은 기준에서 정하였다. decimation을 크게 할수록 계산량은 감소되지만, 시간축에 대한 resolution과 sampling할 수 있는 최대 주파수의 크기가 줄어들게 되어 정확한 결과를 내기 어렵다. 피아노 연주 평가 프로그램에서는 타격 시간을 구한 뒤에 그 음의 음고에 대해서도 측정할 수 있어야 하므로, 적어도 C_6(1044Hz)음까지는 주파수 정보의 손실이 없도록 해야 한다. 이를 고려해 봤을 경우, 22.05kHz를 사용하는 현재의 프로그램에서 M=10 이상으로 하면 sampling할 수 있는 최대 주파수가 1.1kHz 이하로 내려가기 때문에 위험할 수 있다. 한편 M=10으로 할 경우, 10sample 간격은 22.05kHz에서 약 0.4ms이기 때문에 시간축에서 이정도의 오차는 용납할 수 있을 정도이다.

Hamming window의 size와 peak의 threashold 값을 결정하는 것은 실제 여러 연주 신호들을 시뮬레이션하여 결정하였다. hamming window의 size는 작을수록 계산량이 줄어들지만 반면에 신호의 smoothing 효과를 기대하기 어렵다. 또한 peak의 threahold 값은 너무 크면 타격 시간을 놓치게 되고, 너무 작으면 잘못된 타격을 검출하게 된다. 충분한 시뮬레이션 끝에 hamming window의 size를 100(22.05kHz에서)으로, peak의 threahold를 w[n]의 크기 기준 0.0001로 결정하였다.

3. 실험 결과

제안된 알고리듬을 실제 피아노 연주 샘플에 적용시켜 그 성능을 테스트해 보았다. 그림 3(a)는 3개의 타격을 포함한 피아노 연주 신호 x[n]이다. 이 x[n]이 알고리듬의 순서대로 식 1의 moving average와 식 2의 decimation 과정을 거치면 그림 3(b)와 같은 신호 z[n]이 얻어지게 된다. 피아노 음의 magnitude 곡선의 envelop을 유지하고 있음을 확인할 수 있다. 또한 그림 3(a)와 그림 3(b)를 비교하여 보면, 그래프의 가로축 단위를 살펴보았을 때, decimation의 영향으로 sample의 개수가 1/10 으로 줄어든 것을 확인할 수 있다. 그림 3(c)는 그림 3(b)의 z[n]을 differentiation한 신호 v[n]이다. 피아노 음이 타격된 위치에 양의 peak가 생겨나게 됨을 확인할 수 있다. 마지막으로 그림 3(d)는 그림 3(c)의 v[n]을 hamming window와 convolusion시켜서 얻어진 신호 w[n]이다. v[n]을 가지고는 그림 3(c)에서 보이는 바와 같이 정확한 peak의 위치를 찾기가 어렵기 때문에, hamming window와 convolusion 시켜 일종의 smoothing 효과를 준 다음 peak의 위치를 검색하게 된다.

한편, 기존에 제안되었던 알고리듬과의 성능 비교 분석을 위해 같은 sample 피아노 연주곡을 가지고 타격 시간을 찾는 시뮬레이션을 해 보았다. 기존의 알고리듬은 참고 문헌 2에서 제안된 알고리듬을 사용하였는데, 이것은 high frequency power, band-pass filtering 후의 RMS power, comb filtering 등을 통한 방법을 연쇄적으로 적용하도록 고안된 알고리듬으로서, 앞의 방법부터 차례로

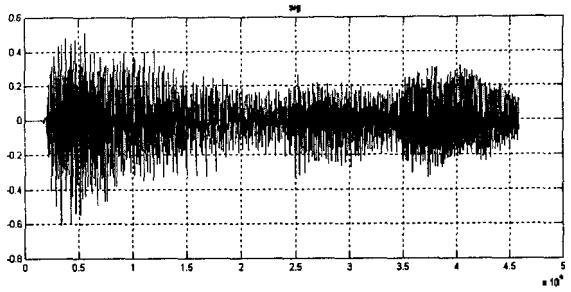

그림 3(a) 입력받은 sample wav 파일 신호 : x[n]

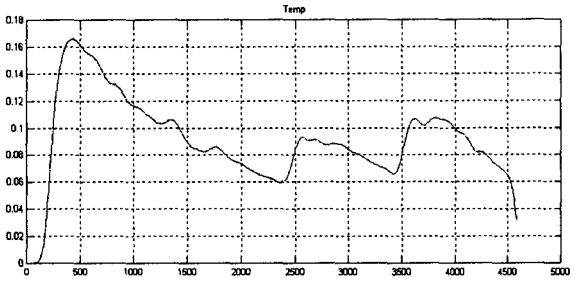

그림 3(b) Moving average와 decimation 과정을 거친 후의 신호 : z[n]

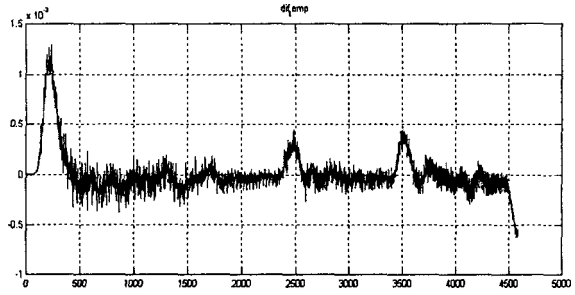

그림 3(c) z[n]을 differentiation한 신호 : v[n]

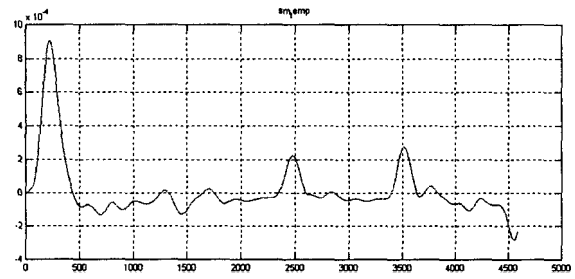

그림 3(d) v[n]을 hamming window 통과시킨 신호 : w[n]
(그래프의 가로축 단위는 모두 sample index)

시도하면서 앞의 방법으로 만족할만한 값을 얻지 못했을 경우 다음 방법을 통해 다시 계산하는 방식으로 구성되어 있다.

성능 비교에 사용된 sample 곡은 J.S.BACH의 "Well-Tempered Clavier Book.I No.16 Fugue b-flat

minor" 의 앞부분이다. 성능 평가 항목은 찾아낸 타격 시간들의 오차와 그 평균, 표준편차, 최대값, 그리고 타격 시간 검출에 걸린 시간으로서 그 결과를 아래 표 1에 정리해 놓았다.

	총타격수	오차평균	오차편차	오차 최대값	계산시간
기존 알고리듬	25	1.85	10.15	31.9	1
제안된 알고리듬	25	6.30	10.84	33.3	0.44

표 1 성능 비교 시뮬레이션 결과
(오차의 단위는 ms이며, 계산 시간은 normalize 됨)

시뮬레이션 결과, 오차의 평균값이 각각 1.85ms와 6.30ms로서 제안된 알고리듬이 타격 시간의 위치를 찾는 면에 있어서 조금 성능이 떨어진다고 할 수 있다. 그러나 그 성능상의 차이는 약 5ms로서 실제 연주되는 피아노 음악의 템포 수준에서는 별로 차이를 느끼지 못하는 수준이라고 할 수 있으며, 오차의 편차와 최대값에 있어서는 두 알고리듬이 비슷한 값을 보이고 있다. 기기에 반해 계산 시간의 측면에서는 제안된 알고리듬이 기존의 알고리듬보다 2배 이상 빠른 속도로 타격 시간을 찾아낸다는 것을 알 수 있다. 피아노 연주 평가 프로그램에 있어서 처리 속도는 매우 중요한 요소이므로, 검출 오차가 용납 가능한 범위 내에 있다면 계산 속도가 빠른 알고리듬을 사용하는 것이 유리하다고 할 수 있다.

끝으로 본 알고리듬을 적용하여 제작한 피아노 연주 평가 프로그램의 실행 모습을 아래 그림 4에 첨부하였다. 간략히 설명을 하면, 평가의 기준이 되는 MIDI 연주 파일과 평가하고자 하는 연주를 녹음한 wav 파일을 읽어들인 다음, 두 연주를 비교하여 음정, 박자, 템포 등의 항목을 통해 연주를 평가하는 프로그램이다.

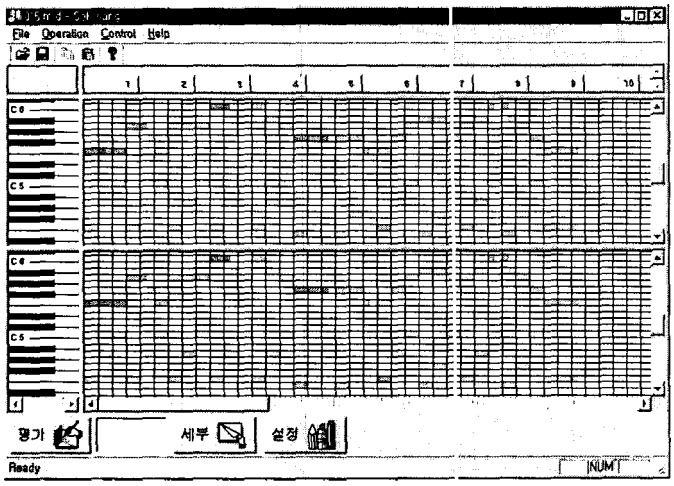

그림 4 피아노 연주 평가 프로그램의 외관

4. 결론

본 논문에서는 피아노 연주 평가 프로그램에 적합한 타격 시간 검출 알고리듬을 제안하였다. 기존의 알고리듬보다 간단한 방법을 이용하여, 실제 연주 평가 프로그램에 사용되는데 문제가 없는 정확도를 확보하면서 계산 시간을 대폭 줄일 수 있는 알고리듬이 가능함을 확일할 수 있었다. 향후 parameter들의 값을 더 효율적으로 결정할 수 있는 방법에 대한 연구와, 연주 평가 프로그램에 적용되는 다른 알고리듬과 연계된 타격 시간 검출 방법 등에 관한 연구 등이 병행되어야 하겠다.

5. 참고문헌

1. K.D.Martin, "A Blackboard System for Automatic Transcription of Simple Polyphonic Music", MIT Media Laboratory
2. E.C.Scheirer, "Extracting expressive performance information from recorded music", Master's thesis, Program in Media Arts and Science, Massachusetts Institute of Technology, 1995
3. R.Keren, Y.Y.Zeevi, D.Chazan, "Automatic transcription of polyphonic music using the multiresolution Fourier transform", Department of Electrical Engineering Technion Israel Institute of Technology, 1998
4. <http://www.midi.org>, "Tutorial on MIDI and music synthesis", MIDI manufacturers assosiation