물고기에 의한 실시간 음악 생성기

장선연+, 이만재
한국정보통신대학교 디지털 미디어 연구소
{yoursun74, manjai}@icu.ac.kr

Musical Synthesizer using fish movement

SunYean Jang, Manjai Lee
Digital Media Lab, Information and Communications University

요 약

음악이란 사람의 느낌을 인위적인 작업을 통해 만들어내는 산물이라고 할 수 있다. 작곡자가 사물을 통해서 또는 어떤 영감을 통해서 만들어내고 있는 것이 대부분의 음악이었다. 이런 것들은 다분히 작곡자의 의도나 듣기 좋도록 구성된 짜임새를 갖음으로써 청취자로 하여금 음악의 표현에 대한 편견을 갖게 할 수도 있다. 인위적임에서 벗어난 음악을 만들거나 우연의 음악을 만들거나 또는 간단한 조작을 통해서 다양한 음악을 만들어 내는 작업들은 많이 있어왔다. 그러나, 이것도 사람이 개입하지 않고서는 안 된다는 인위성을 배제할 수 없다. 따라서 본 논문에서는 컴퓨터 비젼을 이용하여 물고기의 움직임과 상태를 통해 스스로 음악을 만들어 내면서 기존의 음악이 가진 인위성을 제거 하려고 하였다. 그리고 사람의 작업 없이도 음악의 조화로움을 실시간으로 이끌어 낼 수 있도록 하여 좀 더 자연스런 음악으로서의 가치를 갖는 Musical Synthesizer를 구현하였다.

1. 서 론

기존의 음악들은 작곡이나 연주를 사람을 통해서 하지 않으면 안 되는 것들이 대부분이다. 작곡가가 여러 가지 시도를 통하여 음악을 만들어내고, 그 음악들은 다양한 악기들로 연주가 되어진다. 현대에 들어서는 전통적인 악기보다는 전기신호로 소리를 만들어 내는 방식들이 다양해졌고, 어떤 음악장르에 있어서는 electronic device만으로 연주를 하는 경우도 있다. 이런 전자 악기들은 전통악기가 갖지 못하는, 다양한 음을 만들어 준다는, 장점 때문에 많은 사람들로 하여금 애용되고 있다. 하지만 이런 악기들도 작동상에 있어서 매우 복잡하고, 원하는 음악을 만들기 위해서는 많은 학습이 필요하다.

이런 복잡성을 해결하고, 간단한 조작을 통해 다양한 음악을 만들 수 있는 미디어들이 많이 시도되어 왔다. MIT의 Audiopad는 기존의 많은 knob들로 조작되어지는 음악기기를 puck(동그란 모양의 input device)이라는 매체를 통해서 간단히 음을 만들어 낸다. puck의 위치와 간격, 그리고, 보드위에 투영되는 영상이 음의 요소를 만들 수 있도록 도와주며, 간단한 조작만으로 다양한 음악을 쉽게 만들 수 있도록 해준다[1]. 또한 puck을 통한 조작은 아주 직관적인 면을 강조하여, 몇 번의 조작으로 쉽게 자연스런 음악이 된다는 것을 사용자에게 알려준다.

하지만, Audiopad같은 기기조차도 사람의 인위적인 조작 없이는 자연스러운 음악을 만들 수 없다. 눈으로 보는 사물의 움직임을 음이라는 것과 매칭 함으로써 시각과 청각이 실 시간적으로 같은 느낌을 갖도록 한다면, 작곡자의 작업을 통하는 절차 없이 자연스러운 음악을 즉시 만들어 낼 수 있다.

본 연구에서는 사람의 조작을 통하지 않고, 어항 속 물고기의 자연스러운 움직임과 모습만으로 음악을 만들어 내고, 사람의 의도가 아닌 물고기의 모습자체가 음악과 동시성을 갖을 수 있도록 유도하는 새로운 음악 생성 시스템의 구현을 시도하였다.

2. 설계 및 구현

Musical Synthesizer는 각 물고기의 위치를 파악하기 위해 컴퓨터 비젼을 사용한다. 물고기의 위치는 실시간으로 변하고, 복수의 물고기가 존재하므로 종별로 물고기를 구분하고, 같은 종끼리의 거리와 각 개체의 위치를 음을 구성하는 기본 정보로 이용한다.

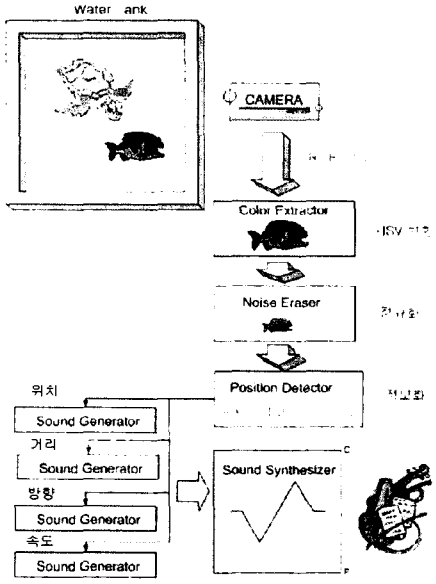
2.1. 개념

어항 속의 물고기는 외부의 환경이나, 다른 종간의 상호작용, 같은 종간의 군항, 또는 먹이를 찾거나 짝짓기 행동 등 다양한 변화를 보여준다. 그 중에서도 우리가 쉽게 인식할 수 있는 움직임의 행태는 물고기의 위치(어항에서의 높이, 상하좌우 방향)와 움직이는 속도, 방향 등임을 쉽게 인지할 수 있다. 이런 요소들은 사람들에게 시각적인 정보를 전달하여, 조화, 혼란, 조용함과 화려함 등을 느낄 수 있도록 해주며, 이런 느낌들은 음악으로 들었을 때 같은 느낌을 갖도록 해준다.

2.2. 내부 작동

어항의 전면에는 카메라를 설치하였다. 그 카메라를 통해서 들어오는 영상을 컴퓨터가 실시간으로 받아 처리 한다. 전달된 영상은 Color Extractor를 통해서 원하는 색의 영역을 추출하고, Noise Eraser로 남아있을 수 있는 다른 물체의 잔상을 제거한 후 Position - Detector에 의해서 인식하고자 하는 물고기의 좌표,

속도, 방향, 거리정보를 뽑아낸다. 전체적인 흐름은 그림 1과 같다.



[그림 1] Musical Synthesizer의 Architecture

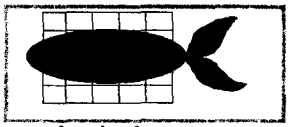
2.2.1. Vision System

카메라는 RGB 데이터의 영상을 15 fps로 컴퓨터에게 보낸다. 그런데, RGB영상은 외부 조명의 영향에 민감하게 반응하여, 같은 이미지에 대해서도 넓은 대역의 변환 값을 갖으므로, 사물을 인식하는 것에는 한계가 있다[2].

그림 3과 그림 4는 각각 카메라에서 갓 넘어온 영상과 Color Extractor를 통한 영상이다. 카메라에서 전달된 영상은 HSV로 실시간 변환되며, 인식하고자 하는 개체의 색을 이용하여, HSV 데이터에서 해당 영역을 찾아낸다. 그 찾아진 영역만을 저장하여, 다른 외부 물체들을 이미지에서 제거한다. 해당 영역을 걸러낸 후에도 다른 물체의 잔상들이 남아있으므로 해당 영역 외의 색을 지워준다.

위의 과정을 통하여 남게 된 영역은 찾고자 하는 물고기의 위치라고 가정하고, 그 위치에 일정한 크기의 마스크를 씌워서 마스크에 들어온 물체가 그 영역에 일정한 크기(%)를 차지하면, 물고기로 인식한다.

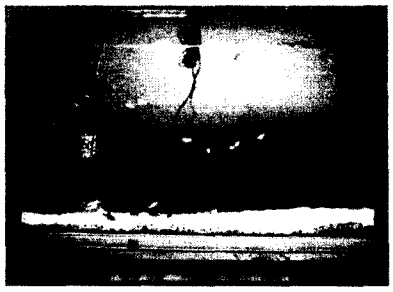
$$Gain = \frac{\sum_{i=0}^{Cols \times Rows} H(i)}{Cols \times Rows} \times 100 \cdot \quad H(i) = \begin{cases} 1 \text{ if color(i) = Red} \\ 0 \text{ if color(i) } \neq \text{ Red} \end{cases}$$



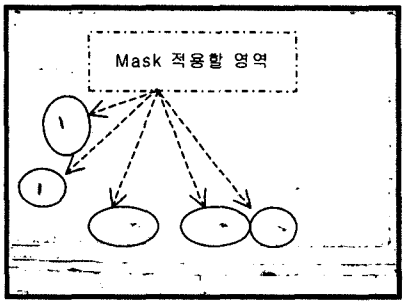
[그림 2] Mask

그림 5는 해당 물고기 좌표를 X로 표현한 것이다.

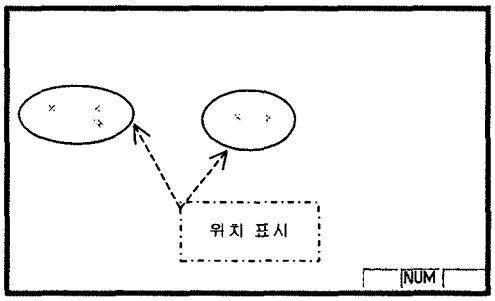
그림 2는 추출된 영역에 마스크를 씌웠을 때 어떻게 물고기를 찾아내는지를 보여준다. 5 X 5의 Grid에 빨간색으로 채워진 영역이 95%(Gain) 이상이면, 이 영역에 들어온 물체는 찾고자 하는 물고기로 인식된다.



[그림 3] 색 추출 이전의 물고기



[그림 4] 색 추출된 물고기



[그림 5] 좌표(X)로 변환된 물고기

실제 오브젝트를 추출해 내는 작업은 Local Scale Selection을 통해서 할 수 있으나, 개체가 다수일 때는 계산량이 많아져서 소프트웨어만으로는 구현할 수 없다[3]. 현재 640 X 480 이미지를 사용하고 있어 오브젝트 추출을 하지 안은 상태에서도 CPU의 점유율이 상당히 높은 편이다.

본 어플리케이션에서는 모든 것을 소프트웨어로 처리하였다. 매 순간 들어오는 이미지를 HSV로 바꿔줌과 동시에 색의 범위를 조사하여, 물고기임을 확인하는 마스크 작업을 즉시 수행하고, 이미 물고기임이 확인된

영역은 계산범위에서 제외 시킴으로써 계산량을 줄였다. 하지만, 성능상의 큰 이득은 얻을 수 없었다. 사용된 컴퓨터의 간단한 사양이다.

* CPU : P4 2.6Ghz RAM : 512Mbyte
* Camera Network : IEEE1394

[표 1] Image Processing 전과 후의 비교

	물체추출 중	물체 추출 이전
CPU 점유율	56%	36%
Memory 점유율	3.1%	2.4%
Frame Rate	4fps	15fps

표 1은 이미지 프로세싱을 할 때와 안 할 때를 비교한 것이다. 이미지 프로세싱 시에는 계산량의 증가로, Frame Rate이 현저하게 떨어지는 것을 볼 수 있다. 음악을 만들기 위해서는 위치에 관한 정확한 정보가 요구되기보다는 개체간의 상대적인 비교 정보만으로도 충분하므로 Frame rate에 대한 손실들은 무시하기로 했다.

2.2.2. Sound Generator와 Synthesizer

위치, 거리, 방향, 속도의 정보를 이용하여 음을 만들어 낸다. 비트는 매우 규칙적인 소리와 강약을 표현할 수 있어 사물의 움직임을 소리로 표현하는데 적당하다.

비트는 Sound Generator에서 만들어지는데, 이 비트들을 Synthesizer가 리듬으로 만드는 과정을 수행한다.

, 리듬의 대표적인 악기는 타악기이다. 대부분의 타악기는 비선율적이어서, 음정을 조정할 수 없고 특정 폭에 연결된 소리만을 가지고 있다. 그런 특정한 폭을 이루는데 47개의 서로 다른 타악기를 이용하여 32가지의 시퀀스를 구성하였다. Synthesizer는 이 소리들을 선택 가능한 박자와 볼륨에서 반복적으로 연주한다

본 어플리케이션에서 생성되는 비트 사운드들은 MMA(Midi Manufacturer' s Association)에서 관리하고 있는 미디 규정집을 기본으로 한다[4]. 물고기의 상하 위치에 따라 Sound Generator에서 47개의 타악기가 선택되고, 좌우위치에 따라 Synthesizer에서 한 싸이클(32박자) 내에서 소리 낼 순서가 결정된다.

시간과 타악기 소리는 2차원 배열이다. 즉, 물고기의 2차원 좌표는 악기의 선택과 동시에 소리가 날 시점을 결정한다. 640 X 480의 이미지 좌표는 32 X 47의 Synthesizer 내의 배열과 사상 되며, 32 컬럼(Column)은 32박자를, 47 로우(Row)는 악기종류를 나타내게 된다.

물고기의 속도는 연주 싸이클의 시간 간격을 조절한다.음의 템포는 한 박자 당 1초에서 10밀리 초의 범위를 갖고, 로가리듬적으로 속도가 변하게 된다. 물고기간의 거리는 연주 싸이클을 조절한다. 32박자를 총 8개의 부분으로 나누어서 거리가 멀면 8개의 마디를 모두 연주하지만, 거리가 짧을수록 마디가 줄어들어 박자가 짧아지는 효과를 얻을 수 있다.

거리계산은 가중치 값을 주어 상대적인 값이 된다. 어플리케이션에서는 외부의 다른 미디나 사운드장치를 사용하지 않고, 컴퓨터 내에 장착된 사운드 카드를 통해 음을 생성한다. 컴퓨터의 사운드 카드는 Realtek AC' 97 Audio를 사용한다.

2.3. 실험 결과

본 실험을 통해 다음과 같은 결과를 얻게 되었다.

a. 조명 : 외부에 설치된 조명은 표면에서 대부분이 반사되어 물고기의 색을 선명하게 만들지 못한다.

b. 피사체의 선정 :색으로만 구분을 하기 때문에 같은 색을 피하여야 하며, 크기 또한 비슷하여야 한다.

c. 어항의 선정 :어항 양면은 정면에서 바깥쪽으로 갈수록 사다리 꼴 모양으로 구조를 조정하여, 벽면에 반사되는 영상이 잡히지 않도록 해야 한다.

d. 비트 음악 : 비트음을 내는 규칙이 음악적 조화를 고려하지 않아, 자칫 노이즈처럼 들리는 경우가 있다.

e. 컴퓨터 비젼 : 물고기의 다양한 형태변화를 위해서는 3차원 좌표를 만들어야 하며, 많은 계산량은 하드웨어로 해결해야 한다.

3. 결론 및 향후 과제

물고기의 다양한 움직임이 음악으로 표현될 수 있다는 가능성을 검증해 보았다. 위치, 속도, 상호간에 거리 정보만을 가지고 비트들을 생성하고, 조합하여 들려주는 것만으로도 음악이 갖는 경쾌함을 느낄 수 있었다.

눈으로 보는 물고기들의 움직임과 실제 들리는 음악간에 약간의 괴리감은 있었지만, 혼란을 가져오는 다른 물고기의 종들을 제거하고, 각 종에게 하나의 악기 소리를 부여하여 , 그 종의 위치, 속도, 방향으로 각 음계를 표현한다면 화음을 갖는 음악을 만들 수 있을 것이다. 3가지 요소(위치, 속도, 방향)로 표현 가능한 음악적 요소를 찾아내어 규칙을 만드는 것이 반드시 수반되어야 할 작업이다.

4. 참고문헌

[1] Ben Recht, "Audiopad: A Tag-based Interface for Musical Performance," Proc.NIME 2002, p. 1, May 2002,

[2] Rafael C.Gonzalez, Digital Image Processing, 2nd, p.p 295-302, 2002.

[3] Oliver Chormat, "Recognizing Goldfish? Or Local Scale Selection For Recognition Techniques," SISR99, p. 1, july 1999.

[4] Midi Manufacturer' s Association, General MIDI 2 Specification, http://www.midi.org/about-midi/gm/gm2_spec.shtml, nov' 1999