음악 만들기- <퍼그스 러브 비트>의 이론과 재미

(Making Music: Pugs Luv Beats' Theory and Fun)

작성자: 얀 세즈넥(Yann Seznec)

작성일: 2012년 8월 1일

인디게임축제(IGF) 후보에 오른 게임 <퍼그스 러브 비트(Pugs Luv Beats)[1] >의 개발자가 음악 기술과 이론을 귀엽고 컬러풀한 게임플레이와 결합해서 아방가르드 음악 기교와 쉬운 즐거움을 조합해 놀라운 결과를 만들어낸 방법을 설명한다.

사람들은 종종 나에게 어떻게 게임을 만들게 됐느냐고 묻는다. 그럴때마다 나는 여전히 놀란다. 왜냐하면 내 배경이 음악가이며 공연자인만큼 나는 정말 나 자신이 게임을 만든다고 자각하고 있지 않아서다. 4년 전 럭키 프레임(Lucky Frame)을 설립하고 위 리모트 음악 조정 소프트웨어(Wii remote music controlling software[2])와 함께 명성(혹은 악명)을 좀 얻으면서부터 나와 내 회사가 인디 게임계에서 더 많이 알려졌다는 것을 깨달았다. 멋진 일이었다.

그건 아주 성공적이었다. 음악가로서 나는 굉장히 많은 기념비적인 업적과 게임에서 인터랙티브(interactive) 오디오와 음악계가 가진 흥미로운 잠재력을 보았기 때문이다. 언젠가 음악가인 내 친구가 지금 음악계에서 일어나는 가장 혁신적인 일은 음악 자체에서뿐만 아니라 음악 소프트웨어에서도 다양하게

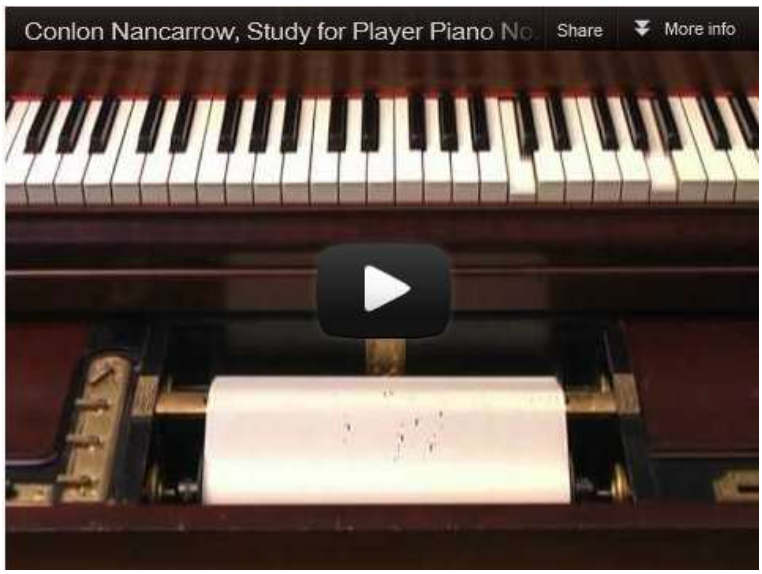
[1] 참조링크: http://www.pugsluvbeats.com/

[2]참조링크: http://www.theamazingrolo.net/wii

일어나고 있다고 알려주었다. 나는 오늘날 만약 바흐[3]나 콜트레인[4]이나 낸케로우[5]가 살아 있다면 그들이 음악 생성 시스템을 디자인하거나 새로운 모노메(Monome)[6] 패치를 프로그래밍할 것이라고 확신한다.

이는 새롭거나 혁명적인 이야기가 아니다. 창조적인 음악 도구로써의 기술을 지향하기 시작한 것은 귀에 거슬리는 소음을 만들어내기 위해 그들만의 비정상적인 악기를 만든 루솔로(Russolo) 형제(사진)와 같은 작곡가들과 함께 수년 전에 시작됐다고 본다. 그리고 데프니 오람(Dephne Oram)과 그녀의 작품을 필름에 그릴 수 있게 해준 놀라운 오라믹스[7] 기계와 함께 지속되었다고 말이다.

내가 방금 언급했던 콘론 낸캐로우(Conlon Nancarrow)는 기술을 사용하는 데 흥미를 느끼기 시작한 작곡가의 또 다른 흥미로운 예이다. 음악가들이 연주하기에는 그의 작품이 너무 복잡하다는 것이 입증되었을 때, 그는 수공으로 피아노 롤에 구멍을 뚫기 시작했고, 자동피아노로 연주를 해서 전혀 들어보지 못했던 (또한 그를 아마도 음악 프로그램을 짠 첫 번째 사람으로 만들어준) 엄청나게 복잡한 음악을 만들 수 있게 되었다.



[동영상: http://youtu.be/LFz2lCEkjFk]

---

[3] 참조링크 : http://vimeo.com/31179423

[4] 참조링크: http://www.heplaysjazz.btinternet.co.uk/giants.html

[5] 참조링크: http://www.telegraph.co.uk/culture/music/classicalmusic/9203463/Conlon-Nancarrow-maestro-of-the-automatic-piano.html

[6] 참조링크: http://monome.org/

[7] 참조링크: http://www.bbc.co.uk/news/technology-12953324

이런 경향은 아마 지난 20년 간 믿을 수 없을 만큼 폭발적인, 미디어 기술의 증가로 최근에 가속화되었을 것이다. 최고급으로 녹음과 제작을 하려면 비싸긴 하지만, 이제는 대규모 인원이 고품질의 소리를 매우 적은 금액으로 녹음하고, 믹싱하고, 작곡하고, 제작하는 것이 가능해졌다.

1980년에 신시사이저(synthesizer)를 원했다면? 음, 아르프(ARP) 한 대에 1300달러를 내야 했다. 이제는 무료 신시사이저 소프트웨어가 터무니없이 많아서 결정하기가 아주 힘들 것이다. (그리고 만약 하드웨어를 만드려고 한다면 부품들은 훨씬 더 저렴할 것이다.) 그 결과 상상할 수 있는 모든 소리를 무료 소프트웨어를 이용해 빠르고 쉽게 만들 수 있다.

매튜 하버트(Matthew Herbert[8])가 종종 지적했듯이 작품 측면에서 또 다른 거대한 혁명은 과거의 작곡가들은 무언가를 흉내내야만 했지만(나이팅게일[9]을 모방하여 쓴 베토벤(Beethoven)의 플루트 파트, 쿠티 윌리엄스(Cootie Williams)의 말하는 트럼펫[10]), 근래 작곡가들은 작품을 위해 밖에 나가서 새 소리를 녹음하거나 누군가 말하는 소리를 녹음해 사용할 수 있다는 점이다.

이는 작곡가와 음악가들에게 거대한 의문을 제기한다. 만약 모든 게 그냥 이루어진다면, 그로 인한 즐거움도 사라지지 않을까? 나는 이것이 작곡가와 음악가들이, 특히 다양한 전자/디지털 장비를 사용하는 이들이 점점 더 인터페이스와 프로세스에 관심을 가지게 되는 이유라고 생각한다. 어떻게 음악이 만들어지는가, 그리고 어떻게 그것이 최종 결과에 영향을 미치는가? 이것이 지난 몇 년간 내가 해온 작업 대부분의 철학적 포커스였다. 그리고 그게 내가 영국 리얼리티 쇼 <용의 굴(Dragons' Den)>에 이상하게 출연하고 최근에는 살아 있는 버섯을 이용해[11] 전자 악기를 조종하는 실험을 하게 된 이유다.

---

[8] 참조링크: http://www.matthewherbert.com/

[9] 참조링크: http://www.youtube.com/watch?v=C3wJzecTJQY

[10] 참조링크: http://www.youtube.com/watch?feature=player_detailpage&v=fFut94LzW2M#t=103s

[11] 참조링크: http://www.theamazingrolo.net/sound-design/spores

[동영상: http://youtu.be/vehSdGmmEvY]

당연히 내가 음악에 게임을 이용하는 데 흥미를 갖게 되는 건 시간 문제였을 뿐이다. 비디오 게임은 아마 현재 가장 강력한 상호작용 인터페이스일 것이다. 음악을 만드는데 게임 상호작용의 힘을 이용하고 시도하지 않을 이유가 뭐겠는가?

게임 세계에서 오디오와 음악을 하는 사람들에게 흥미로운 시대가 왔다. 기타 히어로(Guitar Hero) 스타일의 게임은 아마 서서히 사라질 것이다. 나는 그게 바람직하다고 생각한다. 그런 게임 형태는 분명히 자기 자리가 있고(다시 돌아올 것이라고 확신한다), 그런 유형의 게임들은 플레이어가 기존의 음악을 가능한 한 정확하게 재창조하게 만든다.(보통 이상한 플라스틱 인터페이스를 통해 악기를 표현한다.).

물론 주목하지 않을 수 없는 뛰어난 게임 디자인이지만, 오디오나 음악의 디자인과 개발 관점에서 그렇게 흥미로운 것은 아니다. 참고로 말하면 내 생각에는 재미나 음악성, 창의성보다 정확성에 초점을 맞춘다는 점에서 이런 게임은 흥미롭게도 구식 음악 학교의 교육 방식과 유사점이 있다.

이게 우리가 최근 iOS용으로 발표한, 인디게임축제(IGF) 'Excellence in Audio' 부분 후보에 오른[12](흠흠-) <퍼그스 러브 비트>를 만든 출발점이었다. 우리는 사용자가 음악을 도전으로 보게 하기보다 음악 창작에 초점을 맞춘 음악 게임을 만들고 싶었다. 그래서 오디오 세대에게서 힘을 얻고자 했다.

---

[12] 참조링크: http://igf.com/2012/01/2012_independent_games_festiva_3.html

스페이스 슈터 드럼 머신(space shooter drum machine)으로 첫 시도를 했는데, 스페이스 슈터 드럼 머신은 럭키 프레임(Lucky Frame)의 프로그래머(coder)이자 디자이너인 존 브로드스키(Jon Brodsky)가 이전에 프로토타입을 만들어둔 것이었다. 스페이스 히어로(Space Hero)라고 명명한 이것은 우주선으로 무찔러야 하는 적(enemies) 시리즈를 설정하는 시스템이었다. 적이 스크린에 도착하면 드럼 소리를 내고, 적이 파괴되면 베이스 소리를 만들어냈다. 기본적으로 편집 가능하고 연주하기 쉬운 드럼 기계로, **비디오 경보의 멋진 원형(prototype)이다!**

[동영상: http://youtu.be/uOkqtpTRY9c]

이 원형은 게임 디자인 혹은 최소한 게임으로 상호작용하는 것이 다방면에서 음악 공연이나 작품과 유사하다는 것을 증명해준다. 둘 다 근본적으로 일련의 선택에 기반하며, 많은 경우에서 약간의 시작과 끝, 그리고 순환을 포함한다.

언젠가 대위법 선생님은 나에게 다음과 같이 설명해주었다. 일단 곡의 한 음을 쓰고 나면 오직 네 가지의 선택이 가능하다. 그 음을 다시 연주하거나, 그 음을 다르게 다시 연주하거나, 그 음을 바꾸거나, 아니면 어떤 음도 연주하지 않는 것이다. 유사한 선택들이 게임디자인 세계에도 존재한다. 그리고 과정을 단순화하는

유사 기술 또한 존재한다. 특히 좋은 예시는 푸가를 쓰는 데 사용되는 기술로, 멜로디 하나를 다양한 방식으로 중복해서 쓰는 것이다. 이 비디오를 통해 바흐(Bach) 푸가의 시각화를 확인해보자.

[동영상: http://youtu.be/pVadl4ocX0M]

첫 선율이 한 번 연주되고 전개되기 시작하면, 같은 선율이 다양하게 변주되는 것을 명확히 볼 수 있을 것이다. 이는 작품 내내 여러 번 더 일어난다.

나는 네 개의 푸가 선율이 마치 네 개의 서로 다른 캐릭터들이 속도를 바꿔고 힘을 배가하는 등 다양한 동력을 찾아 같은 공간을 탐험하는 듯 상상해보기를 즐긴다. 다른 비유들도 많이 만들 수 있다. 예를 들어, 바흐 시각화는 플랫폼 게임의 꽤 괜찮은 레벨 빌더(level builder)를 만들 것이다. (미래의 게임 아이디어로 기록해두자)

<퍼그스 러브 비트>는 우리가 이 철학을 탐구하여 만든 음악 게임 시리즈의 첫 작품이다. 우리는 음악과 게임이 동등한 입장에 있는 음악 게임을 만들고자 했다. 하나를 하는 것이 다른 것을 발생하게 하고, 그 역방향 역시 성립하도록. 다시 말해서, 우리의 목표는 게임을 만들기 위해 작곡 기술을, 음악을 만들기 위해

게임을 사용하는 것이었다.

이를 위해 우리는 창의적인 탐구에 적합한 게임플레이와 복잡하고 생산적이지만 모방할 수 있는 오디오를 제공하는 음악 생성 시스템을 모두 만들어내야 했다. 각각은 서로 다른 이유로 중요했다.

복잡성은 아마도 가장 분명한 것이거나(음악을 창조하는 게임에서 액션이 충분한 깊이를 가져야 하며, 반복이 구식이 되지 않도록 충분한 가치를 제공해야 한다는 생각이었다. 이건 좀 별난 과학이다. 왜냐하면 한 수준에서 음악은 반복과 패턴에 기반하지만, 게임에서 반복적인 음향은 환상을 깨기 때문이다) 혹은 최소한 성가신 것이었다.

여기에 더해서, 음악적 복잡성이 제한적이어야 한다. 그렇지 않으면 사용자들은 액션과 음악적 결과의 연결성을 잃을 것이다. 예를 들어, 플레이어의 캐릭터가 특정 타일을 횡단하여 소리를 발생하게 하면, 그것은 순식간에 구식이 된다.

하지만 만약 그 타일이 매번 다른 옥타브의 같은 음을 내면, 이는 음악적 복잡성을 한 층 더할 것이다. 균형(trade-off)은 액션과 소리의 연결성을 감소시킨다. 따라서 균형(balance)은 깨질 필요가 있다!

생성 혹은 절차상의 오디오는 꽤 전문 용어(buzz phrase)이며, 틀림없이 다수의 서로 다른 정의가 인터넷에 떠돌아다니고 있을 것이다. 좀 별도로 생성 음악의 틀에서 이에 대해 생각해보면, 이는 크세나키스(Xenakis)와 존 케이지(John cage) 같은 20세기 고전주의 음악 작곡가들(그리고 주장컨대 찰스 밍거스(Charles Mingus) 같은 재즈 작곡가들)의 세계에 더 가까이 결부된다.

그 세계에서 정신 집중은 공연자가 자신의 음악을 탐구하고 만드는 시스템을 만드는 데 관한 것일 수 있다. 다방면에서 이는 음악적 가능성이 무한하게 바뀔 수 있는 설정을 가능케 하는 제약과 경계, 프레임워크를 만드는 것과 관련이 있다.

예를 들어, 존 케이지의 파이프 오르간 작품 <가능한 한 가장 느리게(As Slow As Possible)>에서, 연주해야 할 것 같은 실제 박자는 확실하지 않다. 이는 공연 시간이 20분에서 639년 [13] 까지 다양해질 수 있음을 의미한다. 케이지의 <폰타나

[13] 참조링크: http://news.bbc.co.uk/2/hi/europe/7490776.stm

믹스(Fontana Mix) [14] >는 여기서 훨씬 더 나아가, 기악 편성을 "자기 테이프(magnetic tape)의 트랙이나 플레이어들을 얼마든지, 악기의 수나 종류 역시 얼마든지"라고까지 정의하고 있다. 악보는 단계를 이룰 때 플레이어 안내 패턴을 만드는 일련의 슬라이드와 종이이다. 잠재적으로 무한한 패턴이 창조된다.

아마도 모두 좀 더 학술적으로 보이겠지만, 이런 (종종 비결정적이고 변수가 있다고 불리는) 유형의 음악 작품과 게임 사이의 기본적 유사성에 주목하는 것은 중요하다. 게임과 스포츠는 몰론, 시스템을 설정하고 플레이어들이 자신만의 길을 만들게 하는 완벽한 본보기(epitome)이기 때문이다. 게임에서 각각의 플레이는 약간 다를 것이며, 어떤 경우에는 서로 다른 플레이(playthrough)들이 알아보기 힘든 결과를 만들 것이다. 20세기의 많은 작곡가들이 게임 이론에 매혹된 것은 우연이 아니다.

마지막으로, 재현성(reproducibility)은 사용자의 수익 측면에서 필수다. 앞서 묘사했듯 게임에서 생성된 음악은 깊이를 가지고, 사용자는 그 창작품 너머의 에이전시를 느껴야 하지만, 게임이 되기 위해서는 미스터리 요소가 필요하다. 이 미스터리는 재현할 수 있어야 한다. 그렇지 않으면 사용자는 패턴을 이해하지 못하고 결국 이용하지 않게 될 것이다. 이것이 소음을 내는 물체를 악기로 바꾸는 주 요소이다.

그러면 우리는 어떻게 이 모든 것들을 실행하였는가? 음, 우리는 모자를 쓴 개를 이용해 게임을 만들었다. <퍼그스 러브 비트>는 우리가 2011년 12월에 발표한[15] 전 세계적인 음악 작곡 게임이다. 플레이어는 이상한 품종의 퍼그를 조종한다. 한때 경이롭고 고도로 발전된 문명의 주인이었던 이 퍼그들은 그들 자신의 탐욕의 피해자들이다. 그들은 무엇보다도 특별한 종류의 "러브(luv)"를 맺은 비트를 모으는 것을 사랑한다. 그러나 가장 큰 비트를 키우기 위해 경솔한 전략을 펼쳐 제어할 수 없을 정도로 넓게 회전하는 바람에, 그들이 태어난 행성은 파괴되었다. 플레이어들은 퍼그들이 더 많은 비트들을 확대해서 새로운 행성들을 재발견하고, 집을 짓고, 잃어버린 기술을 회복하도록 해야 한다.

---

[14] 참조링크: http://www.johncage.info/workscage/fontana.html

[15] 참조링크: http://www.pugsluvbeats.com/

[동영상: http://youtu.be/auiY1oFcDC4]

우리는 이 게임이 매우 자랑스럽다. 무엇보다 우리가 단 세 사람이 모인 작은 회사라는 점에서 그렇다. 존 브로드스키(Jon Brodsky)는 모든 코딩과 루아(Lua)의 개방틀(openframeworks) 조합의 사용, 블러드(Blud)라 불리는 커스텀 빌드 게임 엔진을 다루었다. 나는 내가 쉽게 쓸 수 있는 libpd와 퓨어 데이터(Pure Data)[16]를 이용해 모든 오디오를 만들었다. 아티스트이자 디자이너인 션 매킬로이(Sean McIlroy)는 귀엽고 독창적이며 묘하게 감동적인 퍼그 캐릭터를 개발했다. 게다가 모든 게임 세계(gameworld)와 사용자 인터페이스(UI)를 만드는 놀라운 일을 했다. 이는 대단히 중요한 부분이다. 곧 알게 되겠지만, <퍼그스 러브 비트>는 다방면에서 게임으로 가장한 음악 시퀀서(sequencer)지만, 우리는 이것이 전혀 음악 소프트웨어처럼 보이지 않았으면 했다.

게임플레이 자체는 퍼그에 초점이 맞춰져 있다. 먼저 작은 집에 사는 퍼그 한 마리로 시작한다. 퍼그가 비트(beet)를 수확하게 만들어야 하는데, 단순히 퍼그가 갈 길을 태핑(tapping)하면 된다. 퍼그가 비트(beet)를 치면, 음악 비트(beat)를 만들어내고(하!) 비트(beet) 카운트가 올라간다. 비트가 많아지면 다른 퍼그와 집을 살 수 있다. 퍼그와 집을 더 많이 살수록 행성을 더 많이 발견할 수 있고, 따라서 더 많은 비트를 얻게 된다. 비트를 충분히 모으면 새로운 행성을 사고 다시 시작할 수 있다.

---

[16] 웹 프로그래밍 개발자 커뮤니티

[동영상: http://youtu.be/V0i18_--8Yc]

우주를 확장함에 따라 새로운 지형(terrain)을 발견하게 된다. 새로운 지형은 다른 소리들을 만들어내고 이로써 음악적 색깔들이 늘어날 것이다. 다만 이 지역들이 퍼그를 느리게 만드는데, 이건 퍼그에 모자와 의상을 찾아 착용시키면 문제를 해결할 수 있다. 눈 위에서 더 빨리 가고 싶다고? 산타 모자를 찾아라! 물에선? 상어 지느러미!

게임 디자인과 구조를 이런 식으로 접근했기 때문에 이 게임은 실질적으로 끝이 없다. 모든 모자와 의상을 발견하더라도 계속해서 우주를 탐험하고 새로운 지형의 조합을 찾을 수 있다. 또한 각각의 행성은 독립된 음악 세계로, 언제나 처음부터 시작할 수 있으며 이전의 행성으로 돌아가 리믹스하고 다시 만들 수도 있다. 의상을 입은 퍼그들을 모두 이용해서 말이다.

우리의 미학적 스타일과 게임 방법 모음 뒤에는 이 모든 이론과 디자인을 실행한 생산적 음악 엔진이 있다. 조금 살펴보자.

우리는 모든 오디오를 다루는 데에 퓨어 데이터(Pure Data)를 사용하기로 일찌감치 결정했다. 피터 브링크만(Peter Brinkmann)과 피터 컨(Peter Kirn)이 어려운 작업을 해준 덕분에 가능했다. 무엇보다도 그들은 libpd를 통합해줬다. (신간 사러 가세요!) 퓨어 데이터(Pure Data)는 무료 오픈 소스 그래픽프로그래밍 인터페이스로, 모든 것을 직접 프로그램하는 것을 원하지 않는 오디오와 음악 전문가(geek)를 위해 만들어진 것이다. 이 시점에서 내가 여기에 빠져 들었음을 인정하고 가야겠다.

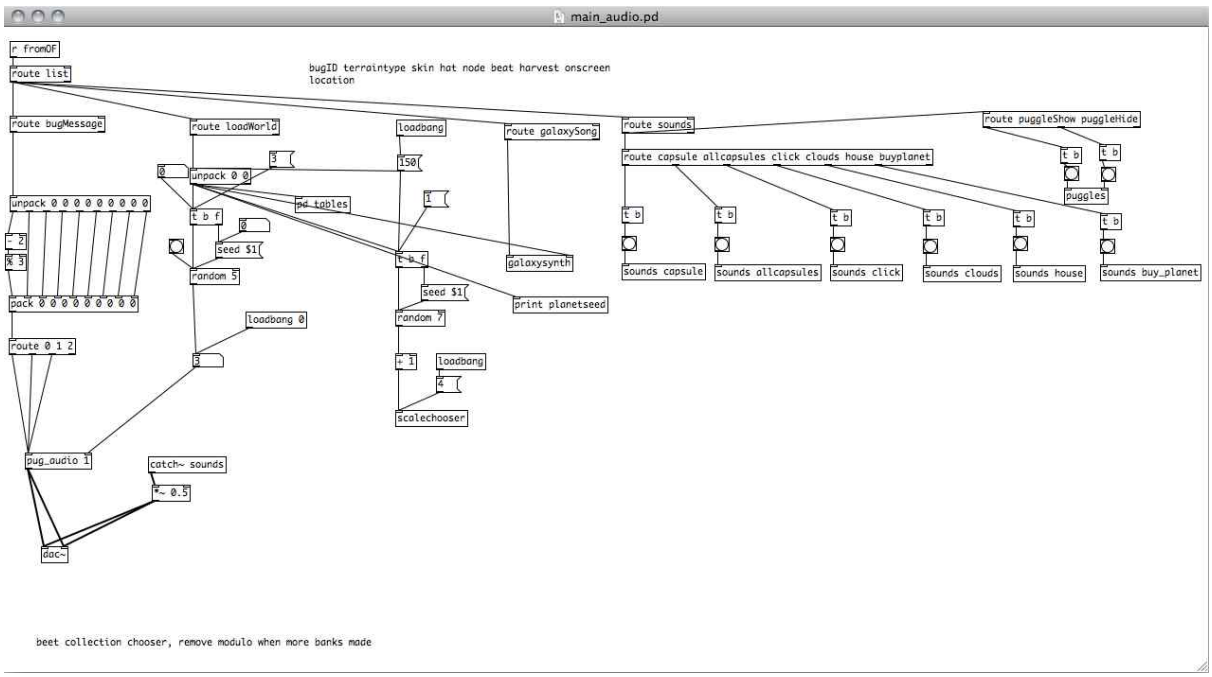
덧붙여서 나는 프로그래머(coder)가 아니며, 아마 앞으로도 절대 프로그래머가 되지 않을 것이다. 화면상의 텍스트는 정말 어려웠지만, 그래픽으로 재현된 코드는 훨씬 이해하기 쉬웠다.

그래서 나는 현재 미술계와 음악계에서 매우 유명한 상업 그래픽 프로그래밍 시스템인 Max/MSP에 손을 대었다. 나는 공부를 했고, 무엇보다 위 루프 기계(Wii Loop Machine)를 만드느라 프로그램을 쓰면서 Max/MSP에 더욱 익숙해졌다.

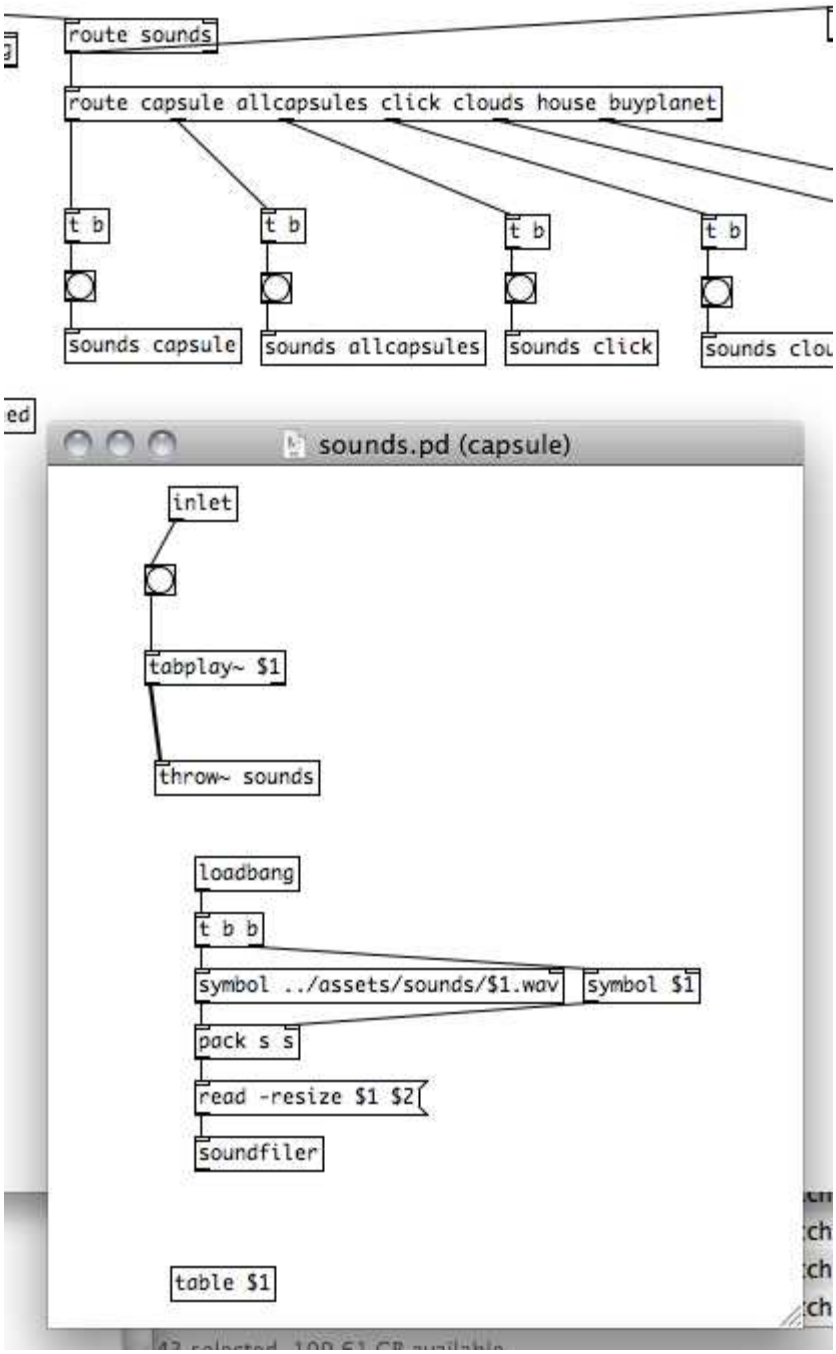
퓨어 데이터는 실질적으로 Max/MSP(여기에 대해서도 할 말이 많지만 별로 재미는 없다) 의 무료 버전이고, 그래서 퓨어 데이터를 우리 프로젝트에 통합시킬 가능성이 있을 때 얼른 기회를 잡았다.

퓨어 데이터로 일하는 것은 (무료 이상의) 몇 가지 큰 이점이 있다. 우선, 개발의 속도를 높여준다. 우리는 프로그래머(coder)가 단 한 명뿐인 세 명으로 구성된 작은 팀이다. 퓨어 데이터로 나는 시사회, 프로토타이핑, 실험을 비롯해 많은 경우에 모든 오디오 개발을 다룰 수 있었다. 나는 오디오 엔진을 여러모로 활용하여, 존을 전혀 방해하지 않고 정말 복잡한 변형을 만들 수 있었다.

두 번째로, 우리는 직접 개발하지 않고도 20년 간의 오디오 조사와 개발의 이점을 취할 수 있었다. 우리가 웨이브테이블 재생 시스템(wavetable playback system)의 정확한 샘플을 원하면? 시스템을 직접 만들 필요 없이, 퓨어 데이터를 이용해서 패키지로 받았다. 에코 시스템(reverb system)은 어떨까? 구글 검색을 통해, 퓨어 데이터에서 세계 각지의 연구원과 개발자들이 만든 수많은 에코(reverb)를 빠르게 찾을 수 있었고, 이를 우리 프로젝트에 맞게 재설정 및 적용 할 수 있었다.

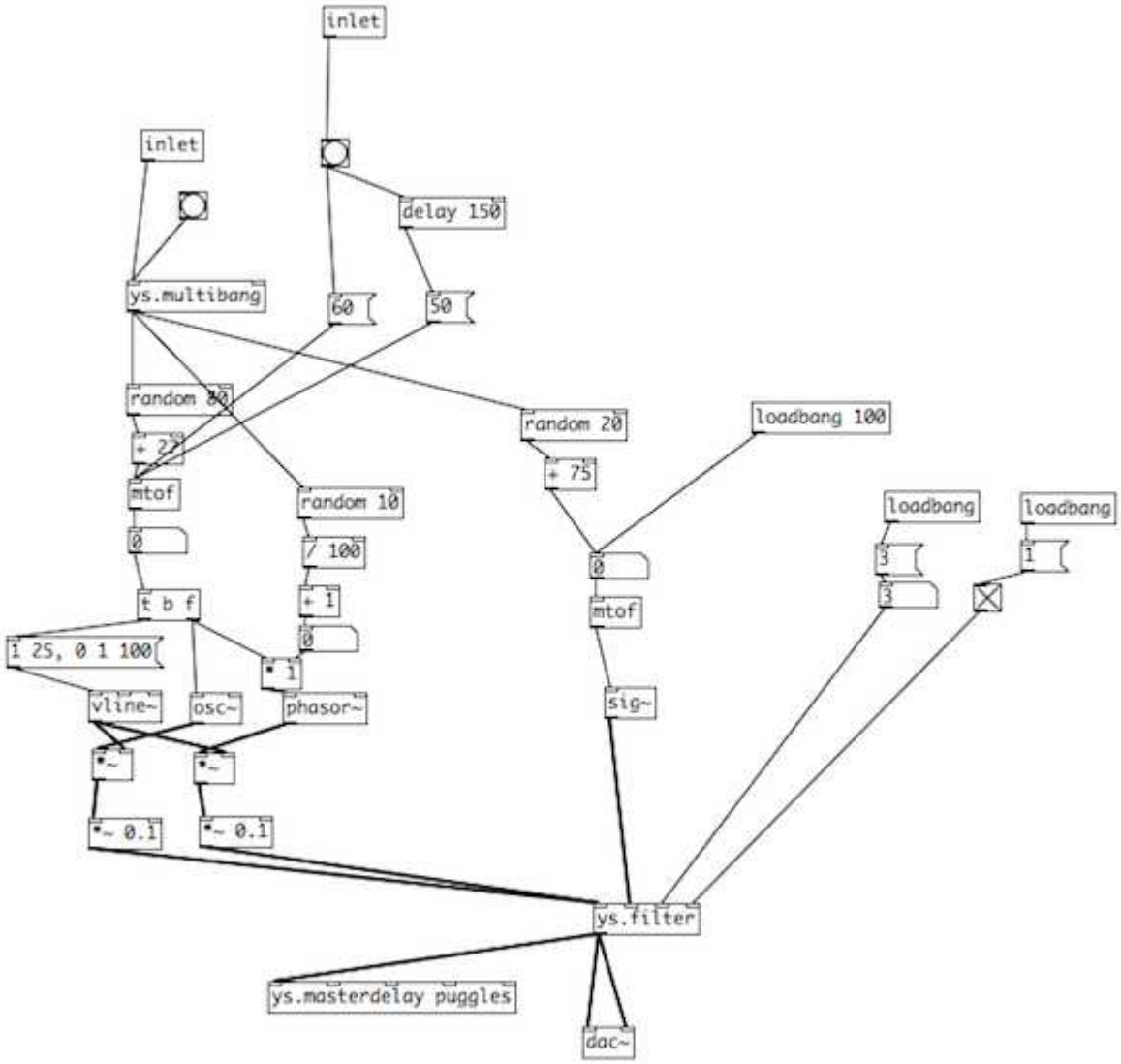


이것이 어떻게 작용하는지 설명하기 위해 <퍼그스 러브 비트>의 전체 음향 엔진을 이 패치에 담았다. 존이 합쳐놓은 루아(Lua)/개방틀 코드에서 메시지를 받고, 또 그 메시지를 서브패치 주위에 보내 음향을 생성하고 수정했다. 가장 간단한 것은 메뉴 동작, 물체 쌓기에 음향을 작동시키는 것이다. 맨 오른쪽의 "경로 음향(route sounds)"을 볼 수 있을 텐데 이는 모든 표준 음향을 다룬다. 각각의 "음향(sounds)" 물체는 음향을 연주하는 약간의 서브패치를 포함하고 있다.

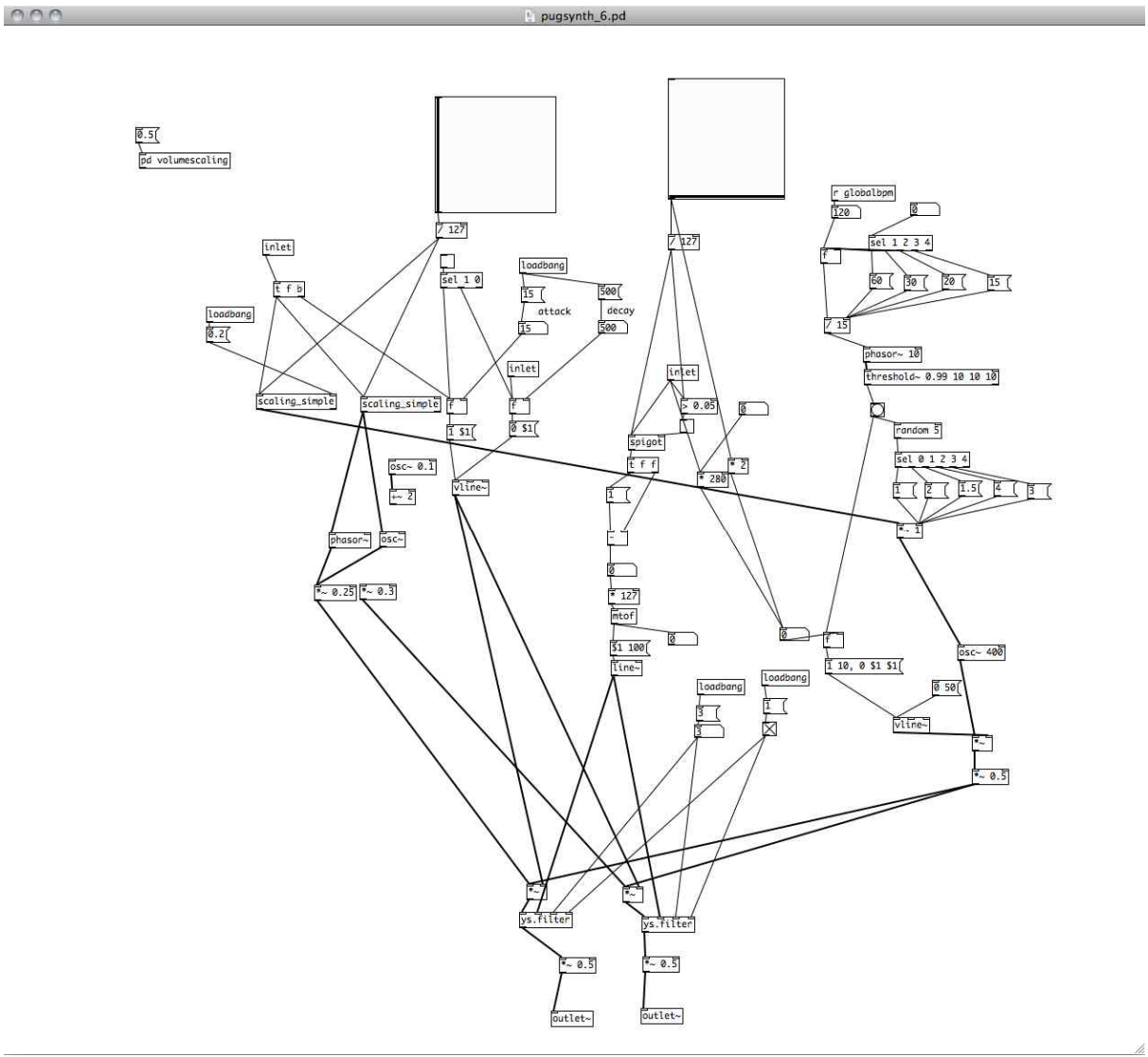


음향을 플레이만을 위해서 이 시스템을 사용하는 것이 불필요하게 복잡해 보일지 모르지만 이렇게 하면 유연성이 훨씬 커진다. 더 중요한 것은 내(음향 디자이너)가 (아마도 멀티스레딩(multithreading)이나 그 비슷한 것으로 유별나게 복잡한 작업을 하고 있을) 존을 방해하지 않고 음향에 관련된 것을 무엇이든 바꿀 수 있다는 점이다.

엔진의 다소 더 복잡한 부분은 '퍼글신시사이저(Pugglesynth)'라고 부르는 것에서 볼 수 있다. 여기서 힘들었던 것은 튜토리얼에서 플레이어를 안내해줄 친절한 퍼그, 퍼글씨(Mr. Puggles)를 위한 음향 라이브러리를 만드는 것이었다. 우리는 퍼글씨가 뛰어오를 때마다 어떤 소리를 만들어주었으면 했지만, 음향 파일을 더 많이 사용하고 싶지는 않았다. 반복 없이 재미있고 매력적인 소리여야 했다. 그래서 두 개의 오실레이터 서브트랙티브 신시사이저(oscillator subtractive synth)를 만들어, 퍼그가 나타날 때마다 음의 높이과 여과 빈도를 조정하면서 일련의 무작위 수가 발생하게 했다. 반면에 퍼글씨가 사라질 때마다, 똑같은 두 개의 음표가 연주되었다. 그리하여 퍼글씨가 짜증나지 않고, 쉽게 알아볼 수 있는 캐릭터가 되었다.



신시사이저에 대해 말하자면, <퍼그스 러브 비트)>의 갤럭시 신시사이저(galaxy synth)는 매우 유명해져서 우리가 완전히 개별 어플리케이션으로 분리해 버렸다. 퍼그 신시사이저(Pug Synth)! 모두가 가장 흥미를 느끼는 건 아무래도 x-y 패드 신시사이저(x-y pad synthesizer)다. 강도(pitch)와 속도(rate)가 x축과 연관된 덥스텝 스타일(dubstep-style)의 필터가 특징인 스왐프 신시사이저(Swamp synth) 패치의 스크린샷을 보자.



하지만 부드럽게 연결되지를 않고, 속도가 여러 마스터 BPM(master BPM)에 고정되어 있다. 이 마스터 BPM은 드럼 머신도 조정한다. 즉, 멋진 퍼그 걸음걸이가 언제나 플레이어의 비트에 맞춰진다는 뜻이다. 또한 우리가 짓궂은 제너레이티브 드럼 머신(generative drum machine)을 만들 수도 있었으며, 모든 것을 강력한 음악적 틀 안에 두면서도 정말 쉽게 다양성을 더할 수 있다는 것을 의미한다.

[동영상: http://youtu.be/XrkHBTu3JCM]

특정 마스터 매개변수에 모든 것을 고정하는 접근법은 우리가 <퍼그스 러브 비트>에서 처음 시도한 방법이다. 창조적인 음악 게임을 기획할 때 큰 도전 과제 중 하나가 음향이 끔찍하지 않게 충분히 통제하면서, 플레이어가 음악적 결과물 너머의 진짜 힘을 느낄 자유를 충분히 보장해야 한다는 것이다. 이는 내가 이전에 언급했던 "복잡성" 및 "재현성"과 강력한 연관이 있으며, 해결책은 없었다.

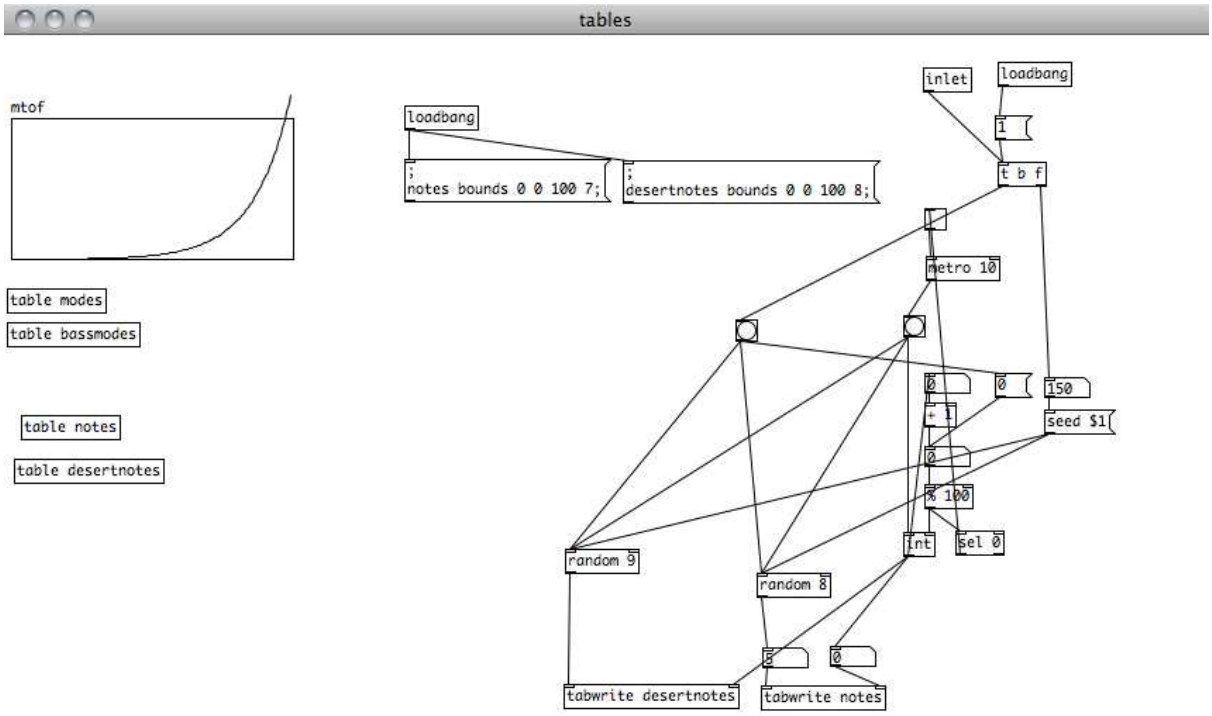
결국 우리는 언제 음악이 발생하든, 박자(tempo)와 조성(key)을 포함해 특정한 글로벌 매개 변수가 만들어지고 이를 바꿀 수 없도록 했다. 이전 것은 꽤 간단했지만, 나중의 것은 더 복잡한 접근법이 필요했다. 놀라지 마세요!

조성(tonality)을 다루는 방식은 내가 가장 자랑스러워하는 오디오 엔진 요소 중 하나이다. 게임에는 퍼그들이 발견한 서로 다른 지형과 관련된 많은 멜로디 라이브러리들(melodic sound libraries)이 있다. 용암은 전자 기타를 작동시키고 모래는 이란식 산투르(Iranian santoor)를 작동시키는 등등. 연주되는 음표는 행성 위 타일의 위치와 연관이 있다.

이는 근본적으로 각각의 행성이 거대한 음악 프로덕션 센터(MPC), 혹은 샘플링 건반임을 의미한다. 타일에 착륙한 퍼그는 언제나 특정 강도(pitch)의 특정 음향을 발생시킬 것이다.

이 음향은 아주 간단하지만, 실제로 이루어내는 것은 좀 까다롭다. 왜냐하면 각각의 행성이 다르기 때문이다. 게임에 있는 수많은 잠재된 행성이 개별적인 데이터베이스를 갖는 것은 불가능할 것이다. 대신에, 우리는 음표를 제어하는 테이블을 생성하기 위해 각각의 행성에 특정한 감별 번호를 붙였다.

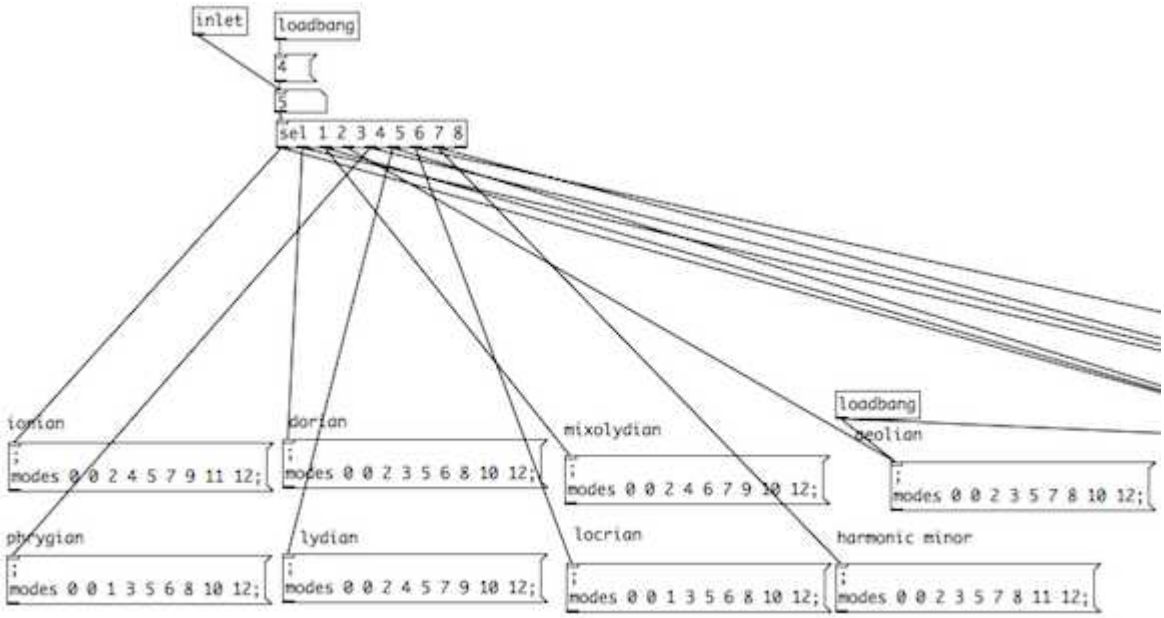
행성 ID는 세계의 모든 타일에 음표 값을 부여하는 색인표를 생성하는 무작위 숫자 시드(seed)로 이용된다. 이는 내가 윤곽을 잡은 이론을 응용한 것으로, 음표 방식을 행성마다 다르게 하여 행성에 복잡성을 주는 것이다. 하지만 이는 특정 행성으로 돌아갈 때마다 음표 방식이 같으므로 재현 가능할 것이다.



하지만, 내 안의 음악 이론 전문가는 만족하지 못했다. 나는 확실하게 모든 행성이 음악적 특징을 갖게 하고 싶었다. 그렇게 하면 정확히 같은 지형의 행성이 두 개 있다 하더라도, 선천적으로 다른 소리를 낼 것이다. 우리는 각 행성에 무작위 숫자 시드의 행성 ID를 사용하여 핵심 음계(mode)와 박자(tempo)를 줌으로써 이를 해냈다.

따라서 모드(mode)나 스케일(scale)은 행성에서 연주되는 모든 음표에 적용된다. 전형적인 장조나 전형적인 화성 단조, 혹은 (서양인에게) 매우 이상한 로크리안 모드[17]를 이용할 수 것이다. 모든 것이 같이 작용하려면 갤럭시 신시사이저 역시 같은 모드의 색인표를 이용해야 하므로, 자신의 행성에서 같은 조성의 즉흥 연주를 할 수 있을 것이다.

[17] 참조링크 http://en.wikipedia.org/wiki/Locrian_mode



이렇게 해서 우리가 지난 몇 년간 개발해온 대화형 음악에 대한 몇 가지 이론들을 포함한 iOS 음악 게임에 정말 강력한 음악 생성 엔진을 만들 수 있었던 과정을 약간 들여다보았다. 우리는 지금 다른 유형의 게임을 이용해 이 이론들을 다른 방식으로 적용시키고자 음악 게임을 더 개발하고 있다.

때때로 이론과 실제 사이에서 연관성을 찾기 힘들 수도 있지만, 나는 이 기사가 럭키 프레임(Lucky Frame)에서 시작한 사고 과정을 이해하는 데에 도움이 되길 바란다. 우리는 다양한 프로젝트에 착수했고, 바깥에서 보기에 그것들은 관련이 없어 보일 수도 있다. 예를 들어 현재, 우리는 <퍼그스 러브 비트 같은 iOS 게임을 만들고 있다. 또 나는 대화형 음악 돼지우리(interactive musical pigsty)를 만들기 위해 <한 마리 돼지 라이브(One Pig Live)> 쇼에서 매튜 하버트(Matthew Herbert)와 함께 공연을 하고 있다.

무엇이 이 프로젝트들을 연결해주는가? 사실 우리의 철학적 관심사는 상호작용과 인터페이스, 특히 음악과 관련된 것에 있다. 플레이어와 게임 혹은 공연자와 관객 혹은 다수의 서로 다른 동력들(dynamics) 사이의 상호작용을 의미한다. 궁극적으로 우리는 이해와 업무를 심화하여 지식과 재미로 이끌고자 한다. 이것은 게임을 만드는 꽤 훌륭한 이유가 된다.