ARTIKKELIT 4/2022

Antti Laaksonen

Piipperin rajoitukset ja mahdollisuudet PC-pelien musiikissa:

Menetelmä piipperimusiikin transkriptioon ja musiikkianalyysit peleistä Alley Cat *ja* Stunts

Antti Laaksonen (antti.laaksonen@helsinki.fi) työskentelee yliopistonlehtorina Helsingin yliopiston tietojenkäsittelytieteen osastossa. Hänen erikoisalaansa ovat musiikkiaineistojen käsittelyyn ja automaattiseen transkriptioon liittyvät algoritmit.

DOI: 10.51816/musiikki.125644

Abstract

The article presents a method using which it is possible to accurately transcribe PC speaker music in games. The idea of the transcription method is to modify the DOSBox emulator so that it records PC speaker events (note onset times and frequencies) while playing the game. This results in an initial transcription which can be processed using music notation software.

Using the transcription method, music from two PC games are transcribed and analyzed. In Alley Cat (1984), an impression of polyphony is achieved by playing sounds in different registers close to each other. In Stunts (1990), a large number of PC speaker events are created to produce polyphony, vibrato and glissandos. The analyses show examples of restrictions and possibilities when composing PC speaker music. The article presents a method using which it is possible to accurately transcribe PC speaker music in games. The idea of the transcription method is to modify the DOSBox emulator so that it records PC speaker events (note onset times and frequencies) while playing the game. This results in an initial transcription which can be processed using music notation software.

Piipperin rajoitukset ja mahdollisuudet PC-pelien musiikissa:

Menetelmä piipperimusiikin transkriptioon ja musiikkianalyysit peleistä Alley Cat *ja* Stunts

Antti Laaksonen

Piipperi (*PC speaker*) on PC-tietokoneen yksinkertainen äänilähde, jota käytettiin PC-pelien musiikin tuottamiseen erityisesti 1980-luvulla ja 1990-luvun alussa ennen äänikorttien yleistymistä. Piipperissä on kaksi tilaa (päällä ja pois päältä), ja sitä voidaan ohjata tietokoneella. Piipperin tilaa muuttamalla voidaan tuottaa halutun taajuuden yksiäänistä kanttiaaltoa. Piipperin rajoitukset aiheuttivat haasteita PC-pelien musiikkien toteuttajille. Joissain peleissä musiikki toteutettiin ensin äänikorttia varten, minkä jälkeen musiikista tehtiin sovitus piipperille.

Tämä artikkeli esittelee artikkelin kirjoittajan kehittämän menetelmän, jonka avulla PC-pelissä olevasta piipperimusiikista voidaan tehdä tarkka transkriptio. Menetelmässä peli käynnistetään DOSBox-emulaattorissa ja tarkkaillaan komentoja, joita peli lähettää piipperille. Tämän avulla voidaan päätellä automaattisesti piipperillä soitettujen äänten taajuudet ja ajanhetket. Menetelmää käyttäen alustava transkriptio pelin musiikista voidaan tehdä automaattisesti, minkä jälkeen transkription työstämistä voidaan jatkaa eteenpäin nuotinnusohjelmassa.

Transkriptiomenetelmää käyttäen artikkelissa analysoidaan tarkemmin kahden PC-pelin musiikkia. Ensimmäinen analysoitava peli on *Alley Cat* (1984), jossa piipperin yksiäänisyys pyritään häivyttämään soittamalla eri taajuuden ääniä lähellä toisiaan. Tämän ansiosta kuulijalle tulee vaikutelma moniäänisyydestä. Toinen analysoitava peli on *Stunts* (1990), joka sisältää musiikkia sekä äänikortille että piipperille. Peli lähettää musiikkia soittaessaan piipperille suuren määrän lyhyitä komentoja. Käyttämällä piipperiä tällä tavalla pystytään tuottamaan samaan aikaan vaikutelma moniäänisyydestä, vibratosta ja glissandoista.

Artikkelin analyysit antavat kuvaa siitä, mitä rajoituksia piipperi aiheutti PC-pelien musiikin toteuttamisessa. Toisaalta analyyseista näkee,

miten taitavat ohjelmoijat pystyivät kiertämään näitä rajoituksia. Piipperi ei ole vain rajoittava tekijä vaan näyttäytyy myös kiehtovana haasteena: kuinka tuottaa mahdollisimman hyvän kuuloista musiikkia piipperin avulla. Siinä missä peleissä voi olla nykyään mitä tahansa musiikkia, piipperi on kuin oma soittimensa, joka loi alkuvuosien PC-peleille niiden tunnistettavan äänimaailman.

Artikkelin alussa tarkastellaan pelimusiikin historiallista kehitystä erityisesti PC-pelien näkökulmasta, piipperimusiikin asemaa osana chip-musiikkia sekä piipperin teknisiä ominaisuuksia. Tämän jälkeen artikkelissa kuvataan DOSBox-emulaattoriin perustuva piipperimusiikin transkriptiomenetelmä. Artikkelin lopussa analysoidaan kuvatun menetelmän avulla pelien *Alley Cat* ja *Stunts* musiikkia sekä tarkastellaan piipperin rajoituksia ja mahdollisuuksia näiden analyysien perusteella.

Pelimusiikin kehitys 1970–1990-luvuilla

Varhaisissa tietokonepeleissä 1950- ja 1960-luvuilla ei ollut yleensä ääniä laitteiston rajoitusten vuoksi. Ensimmäiset askeleet kohti pelimusiikkia olivat piippauksina toteutetut yksinkertaiset ääniefektit, jotka liittyivät pelin tapahtumiin. Historiallisesti tärkeä peli oli vuonna 1972 julkaistu *Pong*, jossa pallon osuminen mailaan aiheutti piippauksen (Kent 2001, 37–48). Vaikeutena pelimusiikin tutkimuksessa on kuitenkin määritellä, milloin pelin soittamia ääniä voidaan pitää varsinaisena musiikkina (Collins 2008, 8–12; Fritsch 2013, 11–12). Esimerkiksi ei ole selvää, ovatko *Pong*-pelin piippaukset musiikkia.

Tietokoneiden kehityksen myötä 1970-luvun lopusta lähtien markkinoille alkoi tulla edullisia kotitietokoneita, ja samaan aikaan pelaaminen siirtyi pelihalleista koteihin (Collins 2008, 28–33). Koneiden valmistajat suuntasivat koneita pelikäyttöön ja ottivat pelaamisen huomioon koneiden tekniikassa. Musiikin saaminen peliin oli kuitenkin teknisesti vaikeaa ja vaati perusteellista tietoa ohjelmoitavan laitteen ominaisuuksista. Pelin ohjelmoijan tuli suunnitella peli niin, että peli pystyi sekä soittamaan musiikin että toteuttamaan samaan aikaan muut tarvittavat toiminnot kuten grafiikan piirtämisen (Hytönen 1987). Teknisiä haasteita aiheutti myös tietokoneiden pieni tallennustilan määrä.

Merkittävä pelikone oli vuonna 1977 julkaistu Atari VCS (Video Computer System), jossa oli yksinkertainen kaksikanavainen äänigeneraattori. Atari VCS:n peleissä oli lähinnä ääniefektejä, koska mahdollisia äänen taajuuksia oli rajoitetusti. Tekniikka kehittyi nopeasti ja esimerkiksi vuon-

na 1982 julkaistussa Commodore 64:ssa oli monipuolinen SID-äänipiiri (Sound Interface Device), joka antoi paljon mahdollisuuksia pelimusiikin toteuttamiseen. Vuonna 1985 Nintendo julkaisi NES-järjestelmän (Nintendo Entertainment System), jossa oli Commodore 64:n tapaan monipuolinen äänipiiri. Tästä lähti liikkeelle japanilaisen pelimusiikin kulttuuri (Belinkie 1999), jonka yksi tunnettu edustaja on vuonna 1986 julkaistu *The Legend of Zelda*. Pelimusiikki alkoi siirtyä enemmän taiteelliseen suuntaan, ja mukana musiikkien luomisessa oli enemmän ammattisäveltäjiä.

Pelimusiikki erosi 1990-luvulle asti muusta musiikista koneiden teknisten rajoitusten ja tallennustilan pienen määrän vuoksi. Erityisesti tallennustilan määrä rajoitti musiikkia, koska pelit tallennettiin yleensä levykkeille ja kiintolevyt olivat pieniä. Tärkeä muutos oli CD-tekniikan yleistyminen 1990-luvun puolivälissä. CD-levyllä oli huomattavasti aiempaa enemmän tilaa, ja peliin saattoi liittää samantasoista musiikkia kuin musiikkia sisältävällä CD-levyllä. Tämä poisti monia rajoituksia pelien musiikin toteuttamisesta.

Nykyään tietokonepelin musiikin säveltäminen ei enää vaadi teknistä erikoisosaamista, ja kyseessä voi olla samantapainen tehtävä kuin elokuvan musiikin säveltäminen (Collins 2008, 85–89). Musiikin tulee usein mukautua pelin tapahtumiin sen perusteella, miten pelaaja pelaa peliä, mutta muuten säveltäjillä ei ole rajoituksia työssään. Tietokoneissa ei ole enää yksilöllistä äänimaailmaa, eikä transkription tekeminen nykypäivän pelin musiikista ole haaste, jota voisi lähestyä käytetyn laitteiston tekniikan kautta.

PC-koneiden pelien musiikki

IBM julkaisi vuonna 1981 tietokoneen nimellä IBM Personal Computer, jossa oli Intel 8088 -prosessori. Tästä alkoi PC-koneiden kehitys, joka jatkuu edelleen. PC oli suunnattu ammattikäyttöön eikä pelikoneeksi, ja sen ääniominaisuudet olivat vaatimattomat. Koneen ainoa äänilähde oli piipperi, jolla pystyi soittamaan yksiäänistä kanttiaaltoa halutulta taajuudelta. Piipperi oli huomattavasti alkeellisempi kuin monien 1980-luvun kotikoneiden äänilähteet, jotka mahdollistivat moniäänisen musiikin ja erilaisten soittimien luomisen.

Piipperi oli tarkoitettu käyttäjän huomion kiinnittämiseen ja virhetilanteiden ilmaisemiseen, mutta sitä alettiin käyttää myös peleissä, koska muutakaan äänilähdettä ei ollut saatavilla. Pikkuhiljaa ohjelmoijat löysivät myös tapoja kiertää piipperin rajoituksia (Leonard 2016). Vaikka piip-

peri on yksiääninen, sillä pystyy tuottamaan moniääniseltä kuulostavaa musiikkia soittamalla eri taajuuden ääniä lähellä toisiaan. Myös vibrato ja glissandot ovat mahdollisia muuttamalla soitettavan äänen taajuutta nopeasti. Myöhemmin löydettiin jopa tapa soittaa piipperillä mitä tahansa ääniaaltoa pulssinleveyden modulaation avulla (Chappell 1991; McAlpine 2017). Tämän tekniikan ongelmia ovat kuitenkin sen vaatima prosessoriteho sekä huono äänen laatu.

Piipperin ominaisuudet olivat riittämättömät monipuolisen pelimusiikin tuottamiseen, ja 1980-luvun lopulla PC-pelaajien piirissä suosiota alkoivat saavuttaa erilliset koneeseen liitettävät äänikortit. Vuonna 1987 julkaistiin AdLib-äänikortti (AdLib Music Synthesizer Card), jonka pohjana oli FM-synteesiin perustuva Yamahan YM3812-siru. AdLib-kortin avulla pystyi soittamaan moniäänistä musiikkia, jossa ohjelmoija voi valita ääniaallon tyypin, voimakkuuden ja muita ominaisuuksia parametrien avulla. Samana vuonna tuli saataville myös MIDI-tekniikkaan perustunut Roland MT-32 -syntetisaattori (Multi-Timbre Sound Module), jota alettiin käyttää pelien musiikin tuottamiseen äänikortin kaltaisesti. MT-32 käytti digitoituja ääninäytteitä, joiden taajuutta ja muita ominaisuuksia pystyi muuttamaan musiikin tuottamiseksi. Siinä oli myös erillisiä ääninäytteitä rumpuäänten tuottamiseksi. (Leonard 2016.)

Äänikortit mullistivat PC-pelien musiikin. Erityisesti AdLib-äänikortti yleistyi pelaajien keskuudessa, ja monet pelit pystyivät soittamaan musiikkia sen kautta. Vaikeutena oli kuitenkin, että osalla pelaajista oli vain piipperi ja osalla taas jokin äänikortti. Käytännöksi tuli toteuttaa peli niin, että se pystyi soittamaan musiikkia usealla tavalla. Tällöin myös pelin sisällä täytyi olla musiikista eri versioita ja pelaajan täytyi valita pelin asetuksista musiikin soittotapa. Joissakin peleissä piipperimusiikki jäi pois ja vain äänikorttien omistajat kuulivat musiikkia.

Vuonna 1989 julkaistiin ensimmäinen Sound Blaster -äänikortti (Sound Blaster 1.0), joka sisälsi AdLib-kortin tavoin YM3812-sirun ja oli muutenkin yhteensopiva AdLib-kortin kanssa. Kuitenkin kortissa oli myös toinen moduuli, jonka avulla pystyi soittamaan mitä tahansa digitaalisessa muodossa olevaa ääniaaltoa. Tämän avulla voi toteuttaa esimerkiksi aidon kuuloisia ääniefektejä, jotka eivät olisi olleet mahdollisia AdLib-kortilla. Sound Blaster ja sen seuraajat nousivat valta-asemaan 1990-luvun alkuvuosina. (Leonard 2016.)

Noin 1990-luvun puolivälissä äänikortista tuli PC-koneen vakiovaruste. Samaan aikaan pelit alkoivat siirtyä DOS-käyttöjärjestelmästä Windowsiin, mikä helpotti musiikin soittamista sekä pelin ohjelmoijan että pelaajan näkökulmasta. Siinä missä DOS-pelin ohjelmoijan tuli luoda

musiikista erilaisia versioita eri äänikorteille tai käyttää ulkopuolista kirjastoa, Windows tarjosi yhtenäisen rajapinnan äänikortin käyttämiseen. Tämän myötä pelaajan ei enää tarvinnut tietää eikä ilmoittaa pelin asetuksissa, mikä äänikortti hänellä on. Äänikorttien tekniset ominaisuudet eivät myöskään enää rajoittaneet pelien musiikkien säveltäjien työtä.

Piipperimusiikki osana chip-musiikkia

Piipperimusiikki voidaan luokitella osaksi chip-musiikkia (*chipmusic*, *chiptune*), joka tarkoittaa 1980-luvun tietokoneiden äänilähteillä tuotettua musiikkia tai yleisemmin sen ajan pelimusiikilta kuulostavaa musiikkia (Driscoll ja Diaz 2009). Chip-musiikki on noussut uudestaan suosioon 2000-luvulla, ja sitä sävelletään nykyään myös moderneilla työkaluilla alkuperäisten laitteiden sijasta. Tämä on johtanut keskusteluun siitä, mitä tekniikoita chip-musiikin tuottamisessa tulisi käyttää ja millainen musiikki on autenttista (Polymeropoulou 2014).

Dittbrenner (2007, 85–96) mainitsee joukon chip-musiikin piirteitä, jotka johtuvat laitteiston teknisistä rajoituksista. Tavallisesti käytössä on vain vähän kanavia, mikä rajoittaa moniäänisyyttä. Äänilähteen ominaisuudet määrittävät, millaisia soittimia ja äänensävyjä voidaan käyttää. Lisäksi musiikin toteuttamista rajoittavat tallennustilan pieni määrä sekä tietokoneen prosessorin vaatimaton tehokkuus. Piipperimusiikissa rajoitukset ovat tavallaan äärimmäiset: käytössä on vain yksi kanava ja vain yksi mahdollinen soitin.

Vaikka piipperimusiikkia esiintyi paljon vanhoissa PC-peleissä, uutta piipperimusiikkia ei juurikaan sävelletä. Yksi poikkeus on nimellä "shiru8bit" toimivan retromuusikon vuonna 2019 ilmestynyt albumi *System Beeps*, joka on sävelletty kokonaan piipperille. Mahdollinen selitys piipperin vähäiseen suosioon on, että piipperi rajoittaa paljon säveltäjän työtä ja nykyään on aina saatavilla muitakin vaihtoehtoja. Piipperimusiikki on jäänyt harvinaiseksi chip-musiikin alalajiksi, jota on tähän mennessä tutkittu vain niukasti.

Piipperi ohjelmoijan kannalta

Piipperillä voidaan tuottaa halutun taajuuden kanttiaaltoa muuttamalla piipperin tilaa tietyin väliajoin. Esimerkiksi A4-viritysääni (440 Hz) voidaan tuottaa muuttamalla piipperin tilaa 440 kertaa sekunnissa. Tavalli-

nen tapa ohjelmoida piipperiä on kytkeä se tietokoneen ajastimeen, joka muuttaa piipperin tilaa halutulla tavalla ja tuottaa tietyn taajuuden ääntä. Piipperin ominaisuuksia ei ole määritelty tarkasti, vaan eri PC-koneissa on erilaisia piippereitä. Esimerkiksi äänen sävy ja voimakkuus vaihtelevat riippuen tietokoneesta. Ohjelmoijan näkökulmasta erilaisia piippereitä käytetään kuitenkin samalla tavalla.

Piipperi on varsin rajoittunut tapa tuottaa musiikkia: sillä voi tuottaa vain kanttiaaltoa ja käytössä on vain yksi kanava, mikä rajoittaa moniäänisyyttä. Lisäksi tietokoneen käyttäjällä ei ole yleensä mahdollisuutta vaikuttaa piipperin äänen voimakkuuteen tai estää äänen tulemista piipperistä. Rajoituksista huolimatta piipperiä käytettiin monissa peleissä ennen äänikorttien yleistymistä, koska parempia tapoja äänen tuottamiseen ei ollut saatavilla. Toisaalta piipperin alkeellisuuden voi nähdä myös kiinnostavana haasteena ohjelmoijalle: kuinka saada tuotettua mahdollisimman hyvää musiikkia piipperillä?

Ohjelmoija pystyy tuottamaan piipperillä ääntä lähettämällä komentoja, jotka muuttavat piipperin tilaa tai yhdistävät sen tietokoneen ajastimeen. Konekielen tasolla piipperiä ohjataan portin 61h kautta, missä ”h” viittaa siihen, että portin numero on annettu heksalukuna. Piipperin voi joko kytkeä päälle tai pois, tai sitten piipperin voi asettaa toimimaan tietokoneen ajastimen kautta. Jälkimmäinen tapa on yleensä hyvä piipperin ohjelmointiin, koska silloin ajastin huolehtii automaattisesti siitä, että piipperi kytketään päälle ja pois halutuin väliajoin tietyn taajuuden äänen tuottamiseksi.

Seuraavassa on esimerkkinä konekielinen ohjelma, joka soittaa A4-viritysääntä (440 Hz), kunnes käyttäjä painaa mitä tahansa näppäintä, jolloin ohjelma päättyy. Ohjelma on toteutettu niin, että sen voi kääntää NASM-kääntäjällä COM-tiedostoksi ja suorittaa MS-DOS-ympäristössä.

```
org 100h
; aseta ajastimen taajuudeksi 440 Hz
mov al, 0b6
out 43h, al
mov ax, 2711
out 42h, al
shr ax, 8
out 42h,al
; kytke piipperi ajastimeen
in al, 61h
or al, 3
```

```
out 61h, al
; odota näppäimen painallusta
mov ah, 0
int 16h
; sulje piipperi
in al, 61h
and al, 0fch
out 61h, al
; sulje ohjelma
mov ax, 4c00h
int 21h
```

Ohjelman ensimmäinen osa asettaa taajuuden 440 Hz ajastimeen. Ajastimessa on kolme kanavaa (numeroitu 0–2), joista kanava 2 voidaan kytkeä piipperiin. Ensin ohjelma lähettää porttiin 43h tiedon siitä, että se haluaa muuttaa kanavan 2 toimintaa ja että halutaan tuottaa kanttiaaltoa. Tämän jälkeen ohjelma lähettää porttiin 42h taajuuden, jolla ajastimen tulee toimia. Ajastin sykkii 1193182 kertaa sekunnissa, ja koska tavoitteena on tuottaa taajuus 440 Hz, ohjelma asettaa ajastimen aktivoitumaan 1193182 / 440 ≈ 2711 ajanhetken välein.

Ohjelman seuraava osa kytkee piipperin ajastimeen portin 61h kautta, jolloin ääni alkaa soida. Sitten ohjelma odottaa käyttäjältä näppäimen painallusta, mikä onnistuu keskeytyksen 16h avulla. Lopuksi ohjelma odottaa, että käyttäjä painaa jotain näppäintä, ja lopettaa sitten piipperin käyttämisen portin 61h kautta ja sulkeutuu keskeytyksen 21h avulla. Tarkempaa tietoa piipperin ohjaamisesta konekielen tasolla sekä ajastimen käyttämisestä on esimerkiksi PC-GPE-kokoelmassa (Feldman 1994a, 1994b).

Vaikka tavallinen tapa käyttää piipperiä on tuottaa kanttiaaltoa tietokoneen ajastimen avulla, piipperiä on mahdollista käyttää myös muilla tavoilla, mikä monipuolistaa sen mahdollisuuksia. Pulssinleveyden modulaatio on tekniikka, joka perustuu siihen, että piipperi on fyysinen laite ja sillä kuluu pieni hetki aikaa siirtyä pois-tilasta päälle-tilaan ja päälle-tilasta pois-tilaan. Jos piipperin tilaa vaihtaa hyvin nopeasti päälle ja pois, piipperi ei ehdi kytkeytyä kokonaan päälle vaan jää välitilaan. Tämän ansiosta on mahdollista soittaa piipperin kautta mitä tahansa digitaalista ääniaaltoa ajastamalla tällaiset muutokset tarkasti (Chappell 1991; McAlpine 2017). Tätä tekniikkaa on käytetty jonkin verran peleissä (Leonard 2016).

Pulssinleveyden modulaatiossa on kuitenkin ongelmia, jotka rajoittavat sen käyttämistä. Ensimmäinen ongelma on, että tekniikka vaatii hyvin tarkkaa ajastusta ja sen käyttäminen voi viedä suuren osan prosessorin te-

hosta. Niinpä jos tekniikkaa käytetään pelissä, prosessorille jää vähemmän aikaa pelin muille toiminnoille kuten grafiikan piirtämiselle. Toinen ongelma on, että vaikka piipperin kautta voi soittaa mitä tahansa digitaalista ääniaaltoa, äänen laatu ei ole hyvä vaan selvästi huonompi kuin äänikortissa. Digitaalisen ääniaallon soittaminen piipperillä onkin enemmänkin ohjelmoijan taidonnäyte kuin todellisuudessa käyttökelpoinen menetelmä.

Seuraavaksi kuvattava piipperimusiikin transkriptiomenetelmä olettaa, että piipperiä käytetään tavalliseen tapaan kanttiaallon tuottamiseen. Menetelmä ei anna mielekkäitä tuloksia, jos käytetään pulssinleveyden modulaatiota tai jotain muuta erikoistekniikkaa.

Piipperimusiikin transkriptiomenetelmä

Musiikin transkriptio on prosessi, jonka tavoitteena on tuottaa nuottikirjoitusta tai jotain muuta kirjallista esitysmuotoa äänitetystä tai soitetusta musiikista. Perinteisesti transkription tekeminen on perustunut kuuntelemiseen: transkription tekijä koettaa kuuntelemalla hahmottaa, mitä musiikissa tapahtuu. Transkriptiota voi helpottaa musiikin hidastaminen, taajuuksien muuttaminen tai suodattaminen ja muut vastaavat keinot.

Yleensä transkriptiossa on monia tulkinnanvaraisia asioita. Transkription tekijän tausta ja näkemykset vaikuttavat lopputulokseen. Jos kaksi henkilöä laatii transkription samasta musiikista, tuloksena olevat transkriptiot eivät yleensä ole samanlaisia. Toisaalta transkription laatimiseen vaikuttaa myös transkription käyttötarkoitus: millä tarkkuudella musiikin ilmiöitä halutaan esittää ja kenelle transkriptio on suunnattu. (Lilja 2014, 164–168.)

Automaattisessa transkriptiossa transkription tekee ihmisen sijasta tietokoneohjelma, joka pyrkii analysoimaan digitaalisessa muodossa annettua musiikkia. Tyypillinen lähestymistapa automaattiseen transkriptioon on käyttää Fourier-muunnosta tai vastaavaa tekniikkaa äänisignaalissa olevien taajuuksien analysointiin. Jos musiikki on yksiäänistä, automaattinen transkriptio on suoraviivaista, mutta moniäänisessä musiikissa tehtävä on huomattavasti vaikeampi eikä tällä hetkellä ole olemassa menetelmää, joka tekisi transkription ihmisen veroisesti. (Benetos et al. 2013.)

Piipperimusiikista olisi mahdollista tehdä transkriptio yllä kuvatuilla menetelmillä joko kuuntelemalla pelissä soivaa musiikkia tai analysoimalla äänisignaalia. Tämä artikkeli keskittyy kuitenkin tarkempaan menetelmään, joka liittyy erityisesti PC-tietokoneen piipperin ominaisuuksiin. Ideana on tarkkailla pelin suorituksen aikana, mitä komentoja peli lähettää piipperille. Tällä tavalla voidaan merkitä muistiin piipperiin liittyvät

komennot ja muodostaa pelin suorituksen jälkeen näiden merkintöjen avulla transkriptio pelin musiikista.

Seuraavaksi kuvataan, miten piipperimusiikin transkriptio voidaan toteuttaa DOSBox-emulaattorin avulla. Emulaattoria käyttämällä vanhoja PC-pelejä voidaan suorittaa nykyaikaisessa käyttöjärjestelmässä. Tärkeä seikka DOSBoxissa on, että se on avoimen lähdekoodin ohjelma, minkä ansiosta on mahdollista muuttaa emulaattoria itse ja lisätä siihen uusia ominaisuuksia. Tässä tapauksessa emulaattoria muutetaan niin, että se näyttää tekstimuodossa piipperiin liittyvät komennot pelin suorituksen aikana. Kuvaus näistä komennoista voidaan edelleen tallentaa tiedostoon, jolloin niitä voidaan analysoida tarkemmin pelin suorituksen jälkeen.

DOSBoxin C++-kielinen lähdekoodi on saatavilla Sourceforge-sivustolla (*https://sourceforge.net/projects/dosbox/*). Seuraavassa kuvauksessa oletuksena on, että pohjana on emulaattorin revisio 4326. Piipperikomentojen tarkkailua varten muutetaan tiedostoja pcspeaker.cpp ja timer.cpp. Tiedosto pcspeaker.cpp sisältää piipperin toteutuksen, kun taas tiedosto timer.cpp toteuttaa ajastimen, jonka avulla piipperillä voidaan soittaa tietyn taajuuden ääntä. Ideana on lisätä tiedostoihin printf-komentoja, jotka näyttävät piipperiin liittyviä tietoja pelin suorituksen aikana. Lisätään ensin tiedostoon timer.cpp rivi (1):

```
        break;
    case 0x02:                          /* Timer hooked to PC-Speaker */
        printf("start %i %f\n", SDL_GetTicks(),PIT_TICK_RATE/(double)
p->cntr); // (1)
        PCSPEAKER_SetCounter(p->cntr,p->mode);
        break;
    default:
```

Tämän rivin ansiosta emulaattori ilmoittaa aina, kun piipperille asetetaan ajastin eli piipperi alkaa soittaa tietyn taajuuden ääntä. Tässä tilanteessa näytetään funktion SDL_GetTicks antama ajanhetki (millisekunteina) emulaattorin käynnistämisestä sekä äänen taajuus, joka saadaan jakamalla vakio PIT_TICK_RATE (montako kertaa ajastin sykkii sekunnissa) arvolla p->cntr. Lisätään sitten tiedostoon pcspeaker.cpp rivit (2) ja (3):

```
        ForwardPIT(newindex);
        switch (mode) {
        case 0:
                printf("stop %i\n",SDL_GetTicks()); // (2)
                spkr.mode=SPKR_OFF;
```

```
                AddDelayEntry(newindex,-SPKR_VOLUME);
                break;
...
        case 2:
                printf("stop %i\n",SDL_GetTicks()); // (3)
                sprk.mode=SPKR_ON;
                AddDelayEntry(newindex,SPRK_VOLUME);
                break;
```

Näiden rivien lisäämisen jälkeen emulaattori ilmoittaa aina, kun piipperi kytketään pois päältä tai uudestaan päälle. Näissä tilanteissa aiemmin aloitettu soitettu ääni päättyy. Kuten muutoksessa (1), ajanhetki haetaan funktion SDL_GetTicks avulla.

Tarkastellaan esimerkkinä tilannetta, jossa peli soittaa Ukko-Nooan melodian. Tässä tapauksessa emulaattorin näyttämät piipperikomennot voisivat olla seuraavat:

```
start 65227 1047.569798
stop 65677
start 65741 1047.569798
stop 66190
start 66254 1047.569798
stop 66704
start 66767 1319.891593
stop 67218
start 67280 1175.548768
stop 67730
start 67794 1175.548768
stop 68243
start 68307 1175.548768
stop 68756
start 68820 1397.168618
stop 69269
start 69334 1319.891593
stop 69783
start 69847 1319.891593
stop 70296
start 70359 1175.548768
stop 70809
start 70874 1175.548768
stop 71322
start 71386 1047.569798
stop 71836
```

Ensimmäisen äänen taajuus on noin 1047 hertsiä, mikä vastaa C6-säveltä. Ääni alkaa ajanhetkellä 65227 ja päättyy ajanhetkellä 65677, joten sen kesto on 450 millisekuntia. Kahdella seuraavalla äänellä on sama taajuus kuin ensimmäisellä äänellä, eli ne vastaavat myös C6-säveliä. Neljännen äänen taajuus puolestaan on noin 1320 hertsiä, mikä vastaa E6-säveltä. Tällä tavalla on mahdollista selvittää yksi kerrallaan piipperillä soitettuun musiikkiin kuuluvat sävelet. Emulaattorin tuottamasta tulostuksesta näkee myös, että tässä tapauksessa emulaattori on ollut käynnissä noin 65 sekuntia (65000 millisekuntia) musiikin alkaessa.

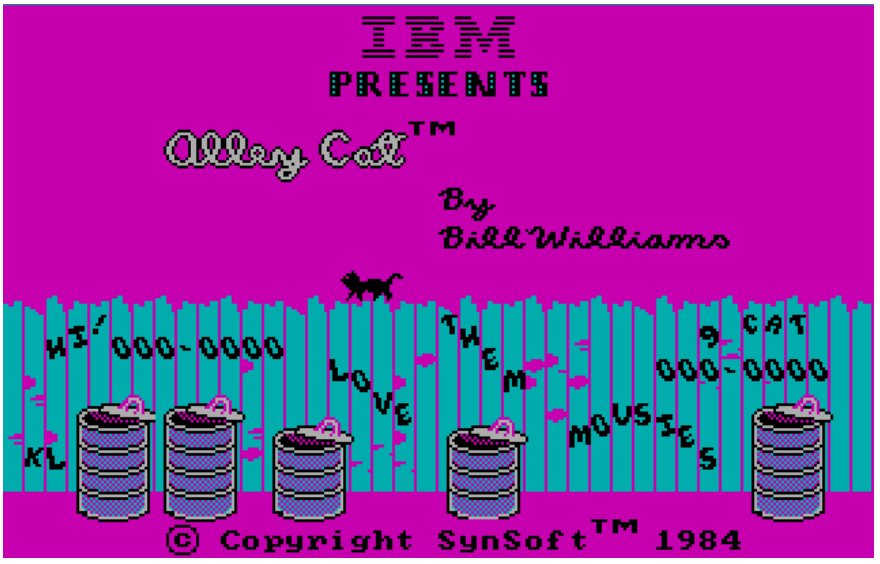
Hertseinä annettu taajuus saadaan muutettua sävelasteikolle kaavalla $12 \cdot \log2(x / f)$, missä x on äänen taajuus hertseinä ja f on A4-sävelen viritystaajuus hertseinä. Tällä kaavalla saadaan laskettua puolisävelaskelten määrä A4-sävelestä. Esimerkiksi Ukko-Nooan alussa esiintyvän C-sävelen taajuus on 1047.569798. Jos oletetaan, että viritystaajuus on usein käytetty 440 hertsiä, kaava antaa tuloksen $12 \cdot \log2(1047.569798 / 440) \approx 15.017651$ eli sävel on noin 15 puolisävelaskelta A4:n yläpuolella eli se on C6. Piipperimusiikissa viritystaajuus ei ole välttämättä 440 hertsiä, koska musiikin säveltäjä tai pelin ohjelmoija voi valita viritystaajuuden haluamallaan tavalla. Kun tiedossa ovat melodiaan kuuluvien sävelten taajuudet, niiden perusteella voidaan kuitenkin koettaa arvioida käytettyä viritystaajuutta.

Emulaattorin antama tekstimuotoinen kuvaus piipperikomennoista on jo eräänlainen transkriptio, koska siitä voidaan päätellä, milloin mikäkin sävel alkaa ja päättyy sekä mikä on sävelen taajuus. Tästä on kuitenkin vielä pitkä matka perinteiseen nuottikirjoitukseen. Transkription työstämistä voidaan jatkaa perinteisessä nuotinnusohjelmassa, johon se voidaan siirtää esimerkiksi MIDI- tai MusicXML-muodossa. Tekstimuotoinen transkriptio soveltuu toisaalta musiikin automaattiseen analyysiin, koska sitä on helppoa käsitellä ohjelmallisesti. Tällä tavalla voitaisiin koostaa esimerkiksi aineisto piipperiä käyttävien pelien musiikista ja tutkia aineiston musiikissa esiintyviä ominaisuuksia. Seuraavaksi kuitenkin tarkastellaan erityisesti kahden pelin musiikkia nuottikirjoitukseen asti vietyjen transkriptioiden avulla.

Alley Cat *-pelin musiikin analyysi*

Alley Cat on vuonna 1984 julkaistu peli, jossa pelaaja ratkoo erilaisia tehtäviä kissana. Pelaajan tulee esimerkiksi ottaa kiinni karannut lintu sekä juoda maitoa koirien vartioimista kupeista. Pelin ohjelmoija ja musiikin säveltäjä on amerikkalainen Bill Williams, joka oli omaperäisistä peleis-

tään tunnettu itseoppinut ohjelmoija (Olafson 1998a, 1998b). Pelin alkuruudussa soiva musiikki alkaa yksinkertaisella teemalla, jota sitten muunnellaan eri tavoin. Musiikki on esimerkki siitä, miten piipperin avulla voidaan toteuttaa moniääniseltä kuulostavaa musiikkia, vaikka todellisuudessa kerrallaan soi vain yksi ääni.



Kuva 1: Alley Cat *-pelin alkuruutu. Synapse Software / IBM 1984*

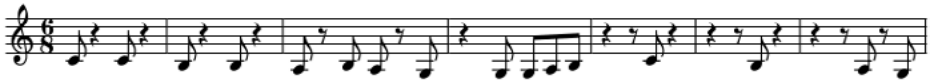
Alley Cat -musiikista saadaan muodostettua emulaattorin avulla aineisto, jossa on 400 start-komentoa ja 168 stop-komentoa. Musiikki alkaa yksiäänisenä, mutta myöhemmin musiikissa esiintyy kaksiäänisyyttä. Tämä on toteutettu soittamalla piipperistä eri rekistereissä olevia ääniä hyvin lähellä toisiaan, jolloin syntyy vaikutelma moniäänisyydestä. Esimerkiksi kaksiäänisen teeman alussa soitetaan ensin 259 hertsin (C4) ääni ja sitten 55 millisekunnin jälkeen 1307 hertsin (E6) ääni. Koska äänet soitetaan niin lähellä toisiaan, kuulostaa siltä kuin ne soisivat samaan aikaan.

Emulaattorin tuottamasta aineistosta voidaan havaita, että kaikki *Alley Cat* -musiikin sävelten ja taukojen pituudet millisekunteina ovat suunnilleen jaollisia 55:llä. Tästä voidaan päätellä, että peli käyttää musiikin tahdistamiseen PC:n ajastimen kanavaa 0. Ajastimen avulla voidaan suorittaa pelin musiikkia ohjaavaa koodia 18.2 kertaa sekunnissa eli noin 55 millisekunnin välein. Musiikki jakautuu 55 millisekunnin pituisiin jaksoihin, ja sävel voi alkaa tai päättyä tällaisen jakson rajalla. Kokeilemalla voidaan

huomata, että peli käyttää viritystä, jossa A4 on 436 Hz. Niinpä kaava $12 \cdot \log2(x / 436)$ muuttaa hertseinä annetun taajuuden x sitä vastaavaksi säveleksi.

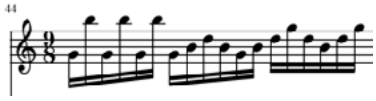
Alley Cat -musiikin transkriptio on esitetty liitteessä 1. Transkriptiossa kuudestoistaosanuotti ja sen jälkeinen tauko on esitetty kahdeksasosanuottina. Lisäksi kohdissa, joissa on kaksi viivastoa, päällekkäiset kahdeksasosanuotit ilmaisevat, että ensimmäinen kuudestoistaosanuotti on bassossa ja toinen kuudestoistaosanuotti on melodiassa.

Musiikin aloittaa seuraava teema:

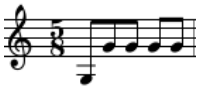


Tämän jälkeen tulee seitsemän teemaan perustuvaa muunnelmaa, jotka monimutkaistuvat pikkuhiljaa. Ensimmäiset muunnelmat ovat yksiäänisiä ja myöhemmät kaksiäänisiä. Muunnelmien jälkeen musiikissa on vielä neljän tahdin kadenssin kaltainen osuus. Teeman, muunnelmien ja kadenssin sävellaji on C-duuri.

Transkriptiossa musiikin tahtilaji on valittu olettaen, että joka tahdissa on kuusi kahdeksasosanuottia. Tämä oletus ei kuitenkaan päde kahdessa tahdissa. Ensinnäkin tahti 44 muodostuu niin, että siinä on yhdeksän kahdeksasosanuottia:

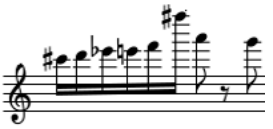


Tahdissa 60 on puolestaan vain viisi kahdeksasosanuottia:



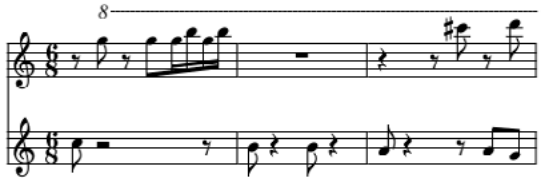
Tässä tapauksessa olisi mahdollista myös jakaa tahdit toisella tavalla niin, että tahdissa 57 olisi viisi kahdeksasosanuottia ja tahti 60 olisi tavallisen pituinen. Transkriptioon valittu tulkinta tuntuu kuitenkin luontevammalta, koska silloin tahdin 59 bassosävelet osuvat iskuille.

Tahdissa 31 vaikuttaa siltä, että nopean kuvion osana oleva fis-sävel on oktaavin liian korkealla:

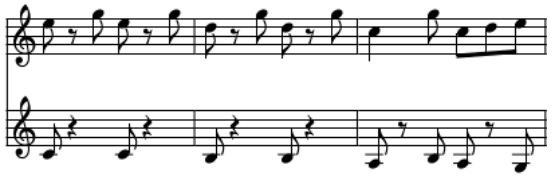


Luultavasti tarkoituksena on ollut, että fis-sävel olisi samassa oktaavissa kuin muutkin tahdissa olevat sävelet. Kuitenkin vaikuttaa siltä, että kun musiikki on siirretty pelin osaksi, kyseiseen säveleen on tullut väärä oktaavi. Tämä ei kuitenkaan häiritse pelin musiikin kuuntelijaa, koska kuvio on niin nopea, että asiaa tuskin huomaa. Tämän vuoksi virhettä ei luultavasti ole huomattu myöskään peliä ohjelmoitaessa.

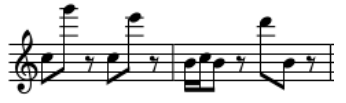
Tahdista 45 alkaen kuulijalle tulee vaikutelma moniäänisyydestä, koska eri rekistereissä olevia ääniä soitetaan nopeasti peräkkäin. Tämän vuoksi nuotit on merkitty transkriptiossa kahdelle eri riville. Tahdeissa 45–47 alarekisterissä on alkuperäisestä teemasta karsittu kuvio, kun taas ylärekisterissä on siihen yhdistyvä vastamelodia:



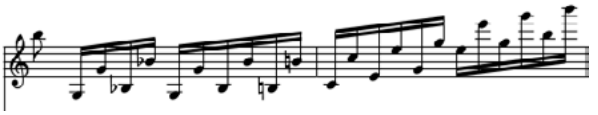
Tämän jälkeen tahdista 49 alkaa kaksiääninen teeman muunnelma, jossa alarekisterissä on bassokulku ja ylärekisterissä on muunneltu teema:



Koska moniäänisyys on toteutettu soittamalla säveliä nopeasti peräkkäin, joskus ei ole selvää, ovatko vierekkäiset sävelet tarkoitettu samanaikaisiksi. Esimerkiksi tahdeissa 41–42 on vierekkäisiä säveliä, jotka ovat selkeästi eri rekistereissä:



Kuitenkaan transkriptiossa näitä säveliä ei ole merkitty samanaikaisiksi, koska kummankin sävelen pituus on kahdeksasosanuotti. Jos tarkoituksena olisi ollut antaa vaikutelma kaksiäänisyydestä, sävelet olisi luultavasti soitettu lähempänä toisiaan. Lisäksi kadenssissa olevassa kuviossa on peräkkäisiä oktaaveja, jotka olisi myös mahdollista tulkita samanaikaisina:

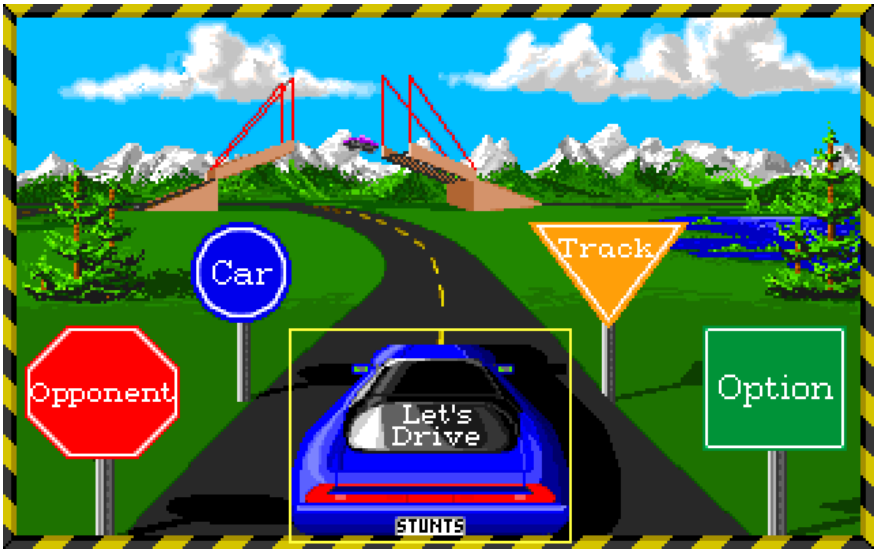


Näin ei ole kuitenkaan tehty transkriptiossa, koska tässä kuuntelijalle ei tule selkeää vaikutelmaa moniäänisyydestä, vaan enemmänkin lopussa oleva kuvio tuntuu arpeggion tyyppiseltä. Vastaavanlaista kuviota on myös tahdeissa 40 ja 43.

Stunts-pelin musiikin analyysi

Stunts (tunnetaan myös nimellä *4D Sports Driving*) on vuonna 1990 julkaistu kolmiulotteinen autopeli, jossa pelaaja ajaa kilpaa tietokonepelaajaa vastaan. Pelin musiikin säveltäjät ovat Krisjan Hatlelid, Brian Plank ja Michael Sokyrka. Pelin asetusohjelma mahdollistaa viisi eri vaihtoehtoa musiikin soittamiseen (piipperi, Tandy, AdLib, Sound Blaster, Roland MT-32). Kolme viimeistä vaihtoehtoa ovat monipuolisia äänikortteja. Pelin julkaisun aikaan äänikortit alkoivat yleistyä PC-koneissa, ja erityisesti aktiivisilla pelaajilla oli usein äänikortti.

Stunts-pelin alkuruudussa soiva musiikki on melko samanlainen eri vaihtoehdoissa. Musiikista erottuu selkeästi bassoraita ja melodiaraita. Piipperiversiossa äänet soitetaan piippauksilla, kun taas äänikorttiversioissa ne soitetaan aidomman kuuloisilla soittimilla. Lisäksi äänikorttiversiossa musiikin taustalla on rummut, mikä puuttuu piipperiversiosta. Tässä artikkelissa esitetty transkriptio perustuu musiikin piipperiversioon, mutta transkription viimeistelyn apuna on kuunneltu myös musiikin äänikorttiversioita.



Kuva 2: Stunts*-pelin alkuruutu. Distinctive Software / Broderbund 1990*

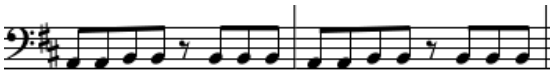
Stunts käyttää piipperiä eri tavalla kuin *Alley Cat*, eikä piipperikomennoista pysty päättelemään suoraan, miten ne vastaavat transkriptioon tulevia säveliä. Musiikin soidessa peli lähettää jatkuvasti (noin kymmenen millisekunnin välein) piipperikomentoja, jotka soittavat lyhyitä ääniä. Musiikissa esiintyvät sävelet muodostuvat useista peräkkäisistä komennoista. Tämä piipperin käyttötapa saattaa johtua siitä, että musiikin äänikorttiversio on tehty ensin ja sen perusteella on muodostettu piipperiversio.

Alley Cat ja *Stunts* eroavat musiikissaan myös siinä, että jokainen ääni *Alley Cat*-musiikissa kuuluu tavalliseen 12-säveliseen asteikkoon, mutta *Stunts*-musiikissa sävelkorkeudessa esiintyy puolisävelaskelta pienempiä muutoksia, joiden avulla toteutetaan vibratoa ja glissandoja. Useimmat pelin soittamat sävelet osuvat silti 12-sävelisen asteikon alueelle. Sävelten taajuudet vastaavat 440 hertsin viritystä, mikä voidaan ottaa virityksen lähtökohdaksi.

Liitteessä 2 on graafinen esitys emulaattorin näyttämistä piipperikomennoista *Stunts*-pelin suorituksen aikana. Jokainen start-komento esitetään mustana pallona. Tästä esityksestä pystyy näkemään vibraton ja glissandot melodiassa. Tästä voidaan myös havaita, että musiikki jakautuu tahteihin niin, että jokaisen tahdin kesto on noin sekunti. Tahtien rajat on merkitty harmailla pystyviivoilla. Liitteessä 3 on puolestaan *Stunts*-musiikin transkriptio nuottikirjoituksena, jossa pyritään esittämään tarkasti musiikissa

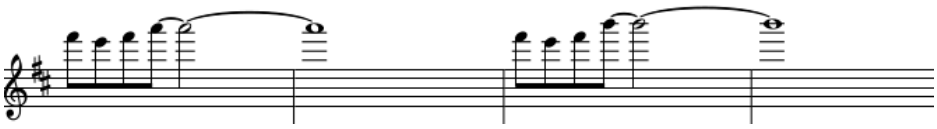
olevat sävelet. Tässä transkriptiossa ei ole kuitenkaan tietoa puolisävelaskelta pienemmistä muutoksista, joiden avulla vibrato ja glissandot on tuotettu.

Stunts-musiikissa on kaksi osaa, jotka molemmat on esitetty 24 tahtina transkriptiossa. Kummassakin osassa on ensin 8 tahdin bassokuvio ja sitten 16 tahtia melodiaa basson säestyksellä. Musiikin sävellajissa on h-mollin piirteitä, mutta sävellaji on häilyvä esimerkiksi alun bassokuviossa. Molemmat bassokuviot alkavat seuraavasti:

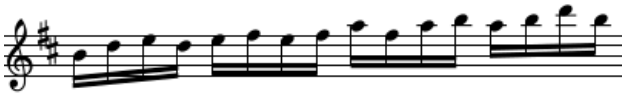


Tämän bassokuvion hahmottaminen piipperin tuottaman äänen perusteella ei ole helppoa. Mahdollinen toinen tulkinta olisi, että kaksi ensimmäistä a-säveltä olisivat kohotahdilla. Kuitenkin kuuntelemalla pelin musiikin äänikorttiversiot, joissa on mukana rummut ja painotuksia äänillä, tulee selväksi, että tämä on tarkoitettu tulkinta.

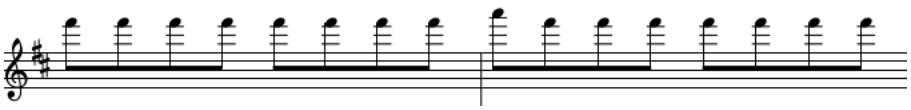
Tahdissa 9 alkaa ensimmäinen melodiaosuus:



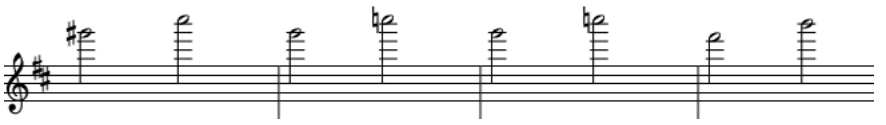
Tässä kuten muuallakin melodiassa pitkillä äänillä on vibratoa. Tämän jälkeen melodiassa on tiheämpi kuvio, jonka jälkeen on jälleen yksittäinen pitkä ääni, joka päättyy alaspäin kulkevaan glissandoon. Melodia huipentuu nopeaan ylöspäin etenevään kulkuun:



Tahdissa 33 alkaa toinen melodiaosuus:



Tämän nopean kuvion jälkeen tulee pitkiä ääniä ja seuraava alaspäin etenevä glissandokulku tahdista 41 alkaen:



Tällainen transkriptio ei ole kuitenkaan tarkka, koska tässä kohtaa melodiassa esiintyy mikrointervalleja äänteen aloituskorkeuksissa. Alemmassa linjassa kulku on gis-g(+)-g(-)-fis ja ylemmässä linjassa kulku on cis-c(+)-c(-)-h, missä (+) ja (-) viittaavat vähän perusviritystä korkeampaan ja matalampaan taajuuteen. Lopuksi musiikissa tulee osuus, jossa basso ja melodia soittavat samankaltaista kuviota, minkä jälkeen musiikki palaa alkuun uudestaan.

Yhteenveto

Artikkelissa on kuvattu tapa tehdä transkriptio PC-pelin piipperimusiikista laajentamalla DOSBox-emulaattorin toimintaa. Emulattoria muutetaan niin, että se antaa pelin suorituksen aikana tietoa pelin piipperille antamista komennoista. Näiden komentojen perusteella voidaan pelin suorituksen jälkeen selvittää tarkasti, miten peli käyttää piipperiä.

Artikkelissa esitetyt transkriptiot on toteutettu siirtämällä piipperikomentoja vastaavat sävelet nuotinnusohjelmaan, jossa on tehty lopullinen nuottikirjoituksena esitetty transkriptio. Tehdyt transkriptiot osoittavat, että artikkelissa kuvattu menetelmä on käytännössä toimiva tapa luoda tarkka transkriptio PC-pelin piipperimusiikista. Menetelmän etuna on, että se helpottaa transkription tekijän työtä verrattuna siihen, että transkriptio tehtäisiin kuuntelemalla tai analysoimalla äänisignaalia.

Piipperin perusrajoituksena on, että se voi soittaa kerrallaan vain yksiäänistä tietyn taajuuden kanttiaaltoa. Molemmissa artikkelissa käsitellyissä peleissä tätä rajoitusta on kierretty soittamalla eri taajuuksia hyvin lähekkäin, jolloin kuulijalle tulee vaikutelma, että äänet soivat samanaikaisesti. *Alley Cat* -pelin musiikissa jokainen uuden äänen aloittava piipperikomento vastaa yhtä nuotinnoksen säveltä. *Stunts*-peli puolestaan tuottaa koko ajan lyhyitä komentoja, joiden ansiosta saadaan esimerkiksi vaikutelma yhtä aikaa soivasta vibratoa käyttävästä melodiasta ja bassokuviosta.

Transkriptiot paljastavat pelien musiikeista myös joitakin sellaisia asioita, joihin ei kiinnittäisi huomiota kuulonvaraisesti. *Alley Cat* -musiikissa on yksi selkeältä tuntuva virhe (eri oktaavissa oleva ääni nopeassa kuviossa) sekä

horjuvuutta tahtien pituuksissa. *Stunts*-musiikin transkription avulla pystyy puolestaan saamaan tarkkaa tietoa melodiassa esiintyvistä mikrointervalleista. Tähän liittyen osoittautuu, että yhtä musiikin osuutta ei ole mahdollista merkitä muistiin tarkasti perinteisen nuottikirjoituksen keinoin.

Alley Cat -musiikki on sävelletty vain piipperille, eikä pelissä ole mahdollisuutta valita muuta tapaa musiikin soittamiseen. Pelissä käytetty tapa luoda vaikutelma moniäänisyydestä soittamalla ääniä lähekkäin toisiaan on yksinkertainen mutta käytännössä hyvin toimiva. Kuulijalle tuloksena oleva musiikki kuulostaa siltä, että siinä soisi samaan aikaan useita ääniä. Tämän perusteella piipperin käytössä oleva yksi kanava ei käytännössä rajoita moniäänisen musiikin tuottamista. *Alley Cat* -pelissä on tyydytty kaksiäänisyyteen, mutta olisi mahdollista soittaa myös useampia ääniä samanaikaisen kuuloisesti. Toisaalta kaksiäänisyys tuntuu olevan tietoinen valinta musiikin ohjelmoijalta eikä pelin musiikki tunnu liian niukalta nykyisellään.

Stunts-musiikissa tilanne on toinen, koska musiikkia on mahdollista soittaa myös äänikortin kautta. Piipperiversio on lisätty luultavasti peliin mukaan vain vaihtoehtona niitä pelaajia varten, joiden koneessa ei ole äänikorttia. Vaikuttaa myös, että piipperiversio on tuotettu ottamalla lähtökohdaksi äänikorttiversiot ja karsimalla niiden aineksia. Esimerkiksi äänikorttiversioissa on rummut, joiden tuottaminen piipperin kautta olisi vaikeaa ja jotka on jätetty pois piipperiversiosta. Joidenkin pelin musiikkiin kuuluvien asioiden hahmottaminen kuuntelemalla piipperiversiota on vaikeaa, koska jokainen piipperin kautta soitettu ääni on yhtä voimakas eikä ole mahdollista käyttää eri sävyjä. Niinpä kuulijan on vaikeaa saada käsitystä esimerkiksi kuvioissa esiintyvistä painotuksista. Voidaankin havaita, että *Stunts*-musiikin tapauksessa piipperi on rajoittanut mahdollisuuksia musiikissa. Toisaalta jos musiikki olisi sävelletty alun alkaen vain piipperille, nämä ongelmat olisi mahdollisesti voitu välttää.

Transkription tekeminen DOSBox-emulaattorin avulla vaikuttaa toimivalta tekniikalta. Sen avulla saadaan tarkasti tietoon sävelten ajankohdat ja taajuudet, minkä jälkeen transkriptio voidaan viimeistellä nuotinnusohjelmassa. Esitetyn menetelmän avulla voitaisiin tehdä vastaavanlaisia transkriptioita muidenkin pelien musiikeista sekä analysoida niissä esiintyviä ilmiöitä. Piipperimusiikkia käyttäviä pelejä tuotettiin paljon 1980- ja 1990-luvuilla, eli mahdollista analysoitavaa on paljon. Esimerkiksi Universal Videogames List (*https://www.uvlist.net/groups/info/spu-pcspeaker*) mainitsee tämän artikkelin kirjoitushetkellä yhteensä 679 PC-peliä, joissa on piipperimusiikkia.

Transkriptiomenetelmän puutteena on toistaiseksi, että musiikki täytyy ”kuunnella” käynnistämällä peli emulaattorissa ja odottamalla, että

musiikki on soitettu. Lisäksi jos peli soittaa transkription kohteena olevan musiikin lisäksi muitakin ääniä, haluttu musiikki täytyy ensin eristää muista piipperikomennoista. Tämän artikkelin transkriptioissa tämä ei ollut kuitenkaan ongelma, koska musiikit soivat pelien alkuruuduissa. Transkription tekemistä olisi mahdollista helpottaa muokkaamalla emulaattorin koodia lisää niin, että transkription voisi aloittaa ja lopettaa esimerkiksi tietyllä näppäinyhdistelmällä pelin suorituksen aikana.

Mahdollinen tulevaisuuden tutkimusaihe olisi mennä syvemmälle pelien toimintaan ja koettaa selvittää, missä muodossa musiikki on tallennettu pelien sisällä ja miten pelit tuottavat piipperiin liittyviä komentoja. Tämä antaisi mahdollisesti lisää tietoa musiikin sävellysprosessista ja selittäisi musiikissa esiintyviä ilmiöitä. Periaatteessa olisi myös mahdollista automatisoida piipperikomentojen etsiminen pelin sisältä, jolloin transkription voisi tehdä käynnistämättä peliä ja odottamatta sen musiikin soittamista. Tämä olisi kuitenkin luultavasti vaikeaa, koska on monia mahdollisia tapoja suorittaa piipperikomentoja pelin aikana.

Piipperimusiikkia on tutkittu tähän mennessä yllättävän vähän, kun otetaan huomioon, miten yleistä se oli vanhoissa PC-peleissä. Piipperin rajoittuneisuudesta huolimatta pelien kehittäjät käyttivät aikaa piipperimusiikin luomiseen ja pelien pelaajat kuuntelivat sitä, minkä vuoksi kyseessä on tärkeä ilmiö pelimusiikin historiassa. Piipperimusiikkia tutkimalla saa käsitystä siitä, millaista pelimusiikkia voi saada aikaan alkeellisella äänilähteellä, jossa on vain yksi kanava ja yksi mahdollinen soitin. PC-pelien musiikit muodostavat kiinnostavan aineiston, jota olisi mahdollista tutkia laajemminkin tässä artikkelissa esitetyllä menetelmällä.

Lähdeluettelo

Belinkie, Matthew. 1999. “Video game music: not just kid stuff”. *https://www.vgmusic.com/information/vgpaper.html* (viitattu 15.11.2022)

Benetos, Emmanouil, Simon Dixon, Dimitrios Giannouis, Holger Kirchhoff ja Anssi Klapuri. 2013. ”Automatic music transcription: challenges and future directions”. *Journal of Intelligent Information Systems* 41(3): 407–434. *https://doi.org/10.1007/s10844-013-0258-3*

Chappell, David. 1991. ”Software-based digital audio on PCs”. Proceedings of the 19th Annual Conference on Computer Science. ACM. *https://doi.org/10.1145/327164.328775*

Collins, Karen. 2008. *Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design*. MIT Press.

Dittbrenner, Nils. 2007. *Chip-Musik: Computer- und Videospielmusik von 1977–1994*. Universität Osnabrück.

Driscoll, Kevin ja Joshua Diaz. 2009. “Endless loop: a brief history of chiptunes”. *Transformative Works and Cultures* 2. *https://doi.org/10.3983/twc.2009.096*

Feldman, Mark. 1994a. “Programming the Intel 8253” (PC-GPE Collection). *http://qzx.com/pc-gpe/pit.txt* (viitattu 15.11.2022)

Feldman, Mark. 1994b. “Programming the PC Speaker” (PC-GPE Collection). *http://qzx.com/pc-gpe/speaker.txt* (viitattu 15.11.2022)

Fritsch, Melanie. 2013. ”History of Video Game Music”. Teoksessa Moorman, Peter (toim.). *Music and Game: Perspectives on a Popular Alliance*. Springer.

Hytönen, Pasi. 1987. ”Näin syntyi Uuno-peli”. *MikroBITTI* 4/1987: 36–38.

Kent, Steven. 2001. *The Ultimate History of Video Games: From Pong to Pokemon*. Three Rivers Press.

Leonard, Jim. 2016. “IBM PC Ramblings”. *http://www.oldskool.org/sound/pc* (viitattu 15.11.2022)

Lilja, Esa. 2014. ”Populaarimusiikin transkriptio ja analyyttisen kuuntelun kehittäminen”. *Etnomusikologian vuosikirja* 26: 162–191. *https://doi.org/10.23985/evk.66793*

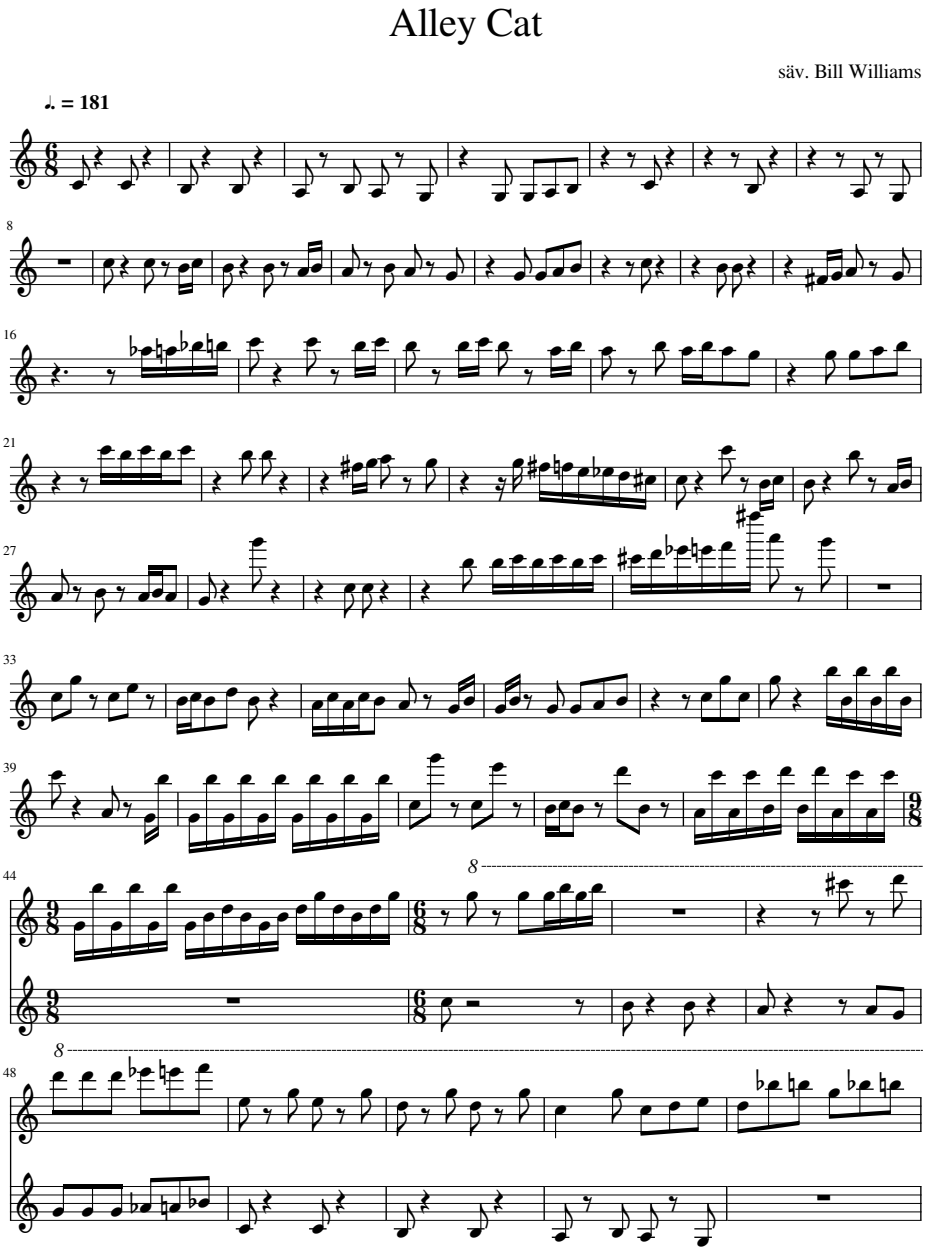
McAlpine, Kenneth. 2017. ”The Sound of 1-bit: Technical Constraint and Musical Creativity on the 48k Sinclair ZX Spectrum”. *The Italian Journal of Game Studies* 6.

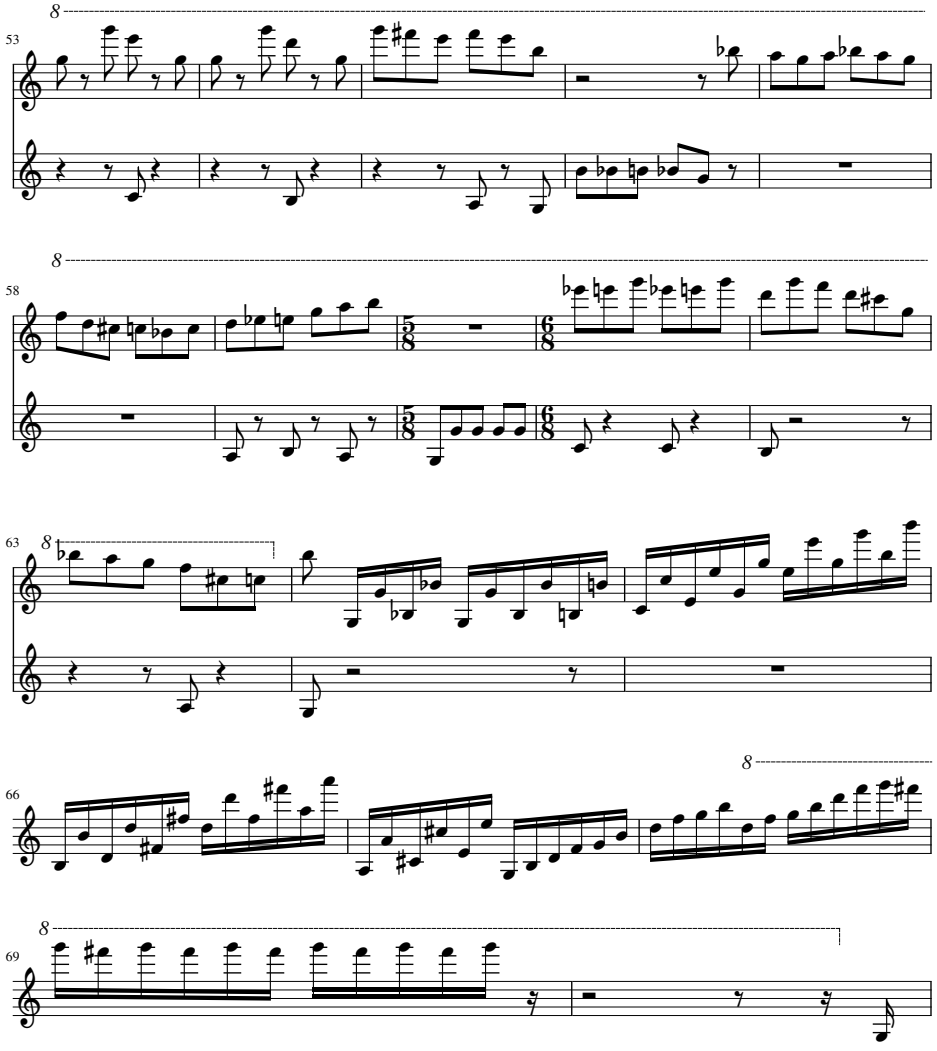
Olafson, Peter. 1998a. “A Tribute to the Work of Bill Williams: Part 1”. *Amazing Computing* 13(2).

Olafson, Peter. 1998b. “A Tribute to the Work of Bill Williams: Part 2”. *Amazing Computing* 13(3).

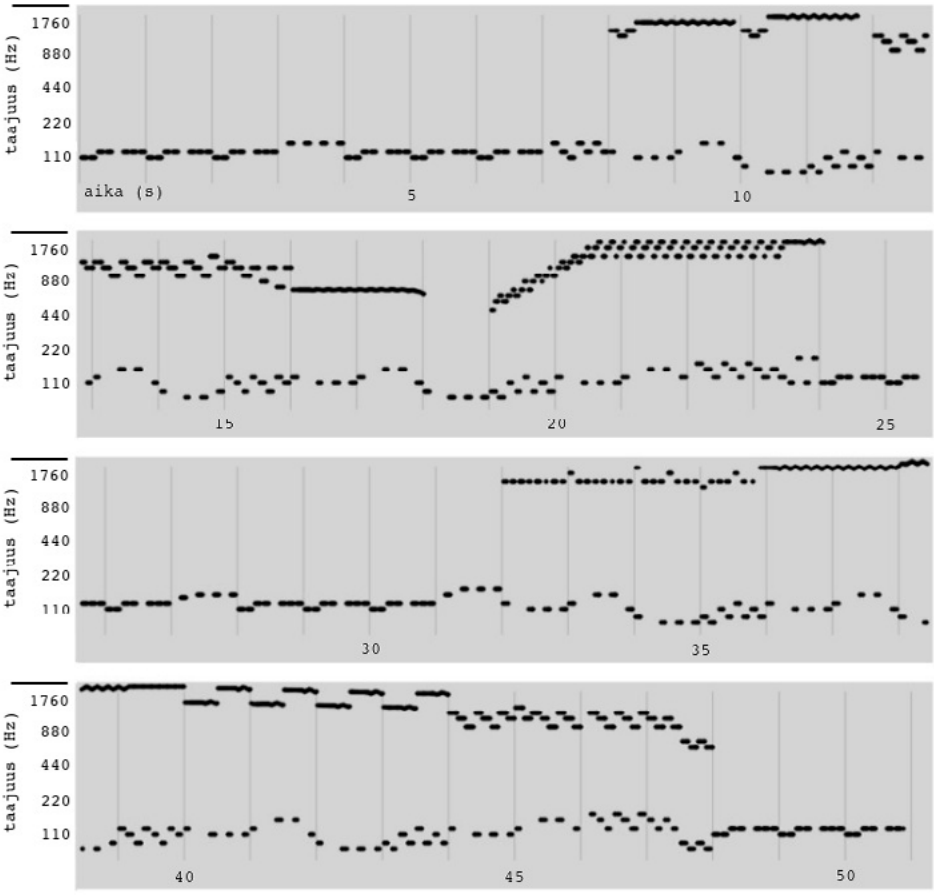
Polymeropoulou, Marilou. 2014. “Chipmusic, Fakebit and the Discourse of Authenticity in the Chipscene”. *WiderScreen* 1–2/2014.

Liite 1: Alley Cat *-musiikin transkriptio. Musiikin säveltäjä Bill Williams. Synapse Software / IBM 1984*

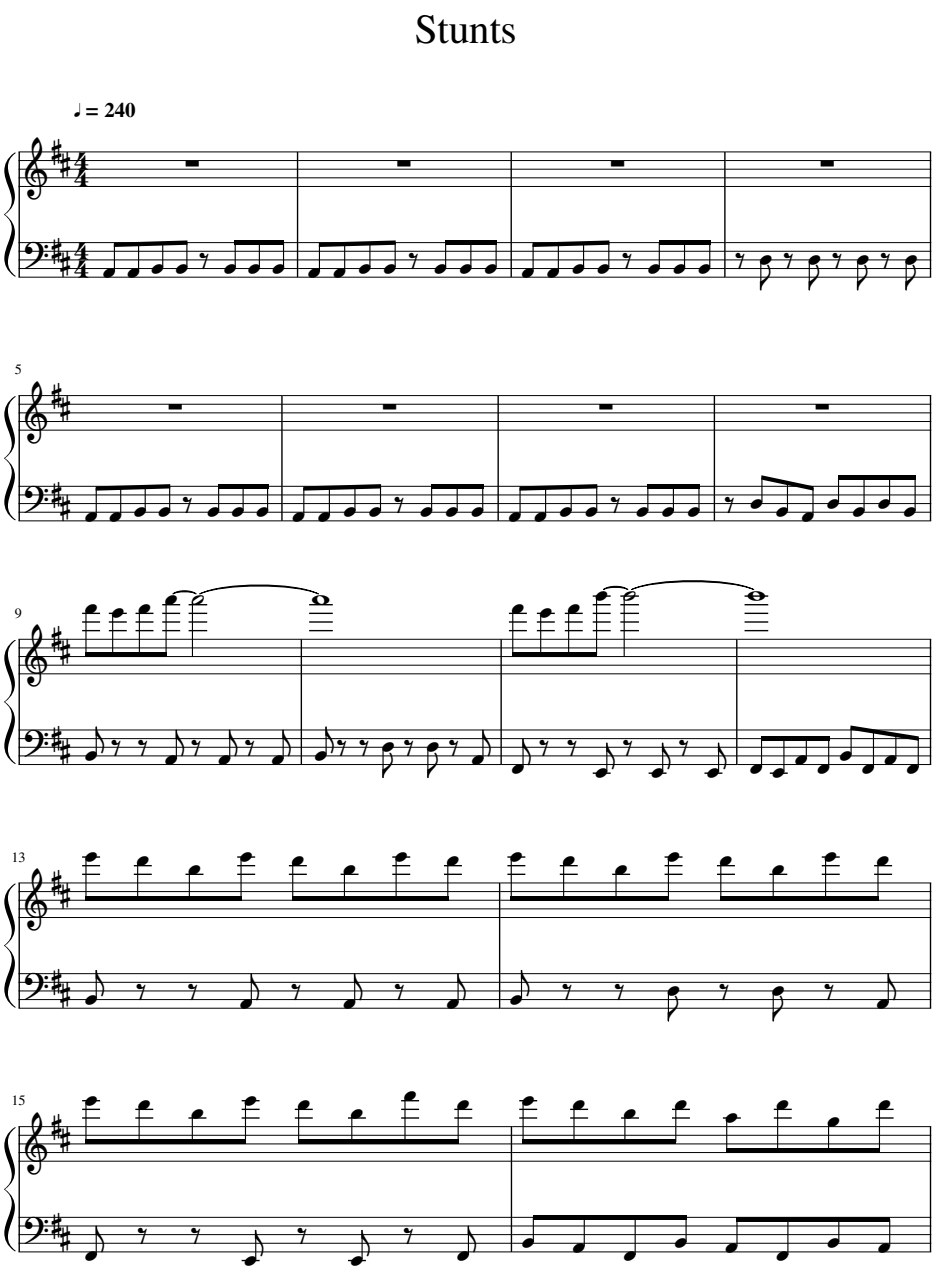


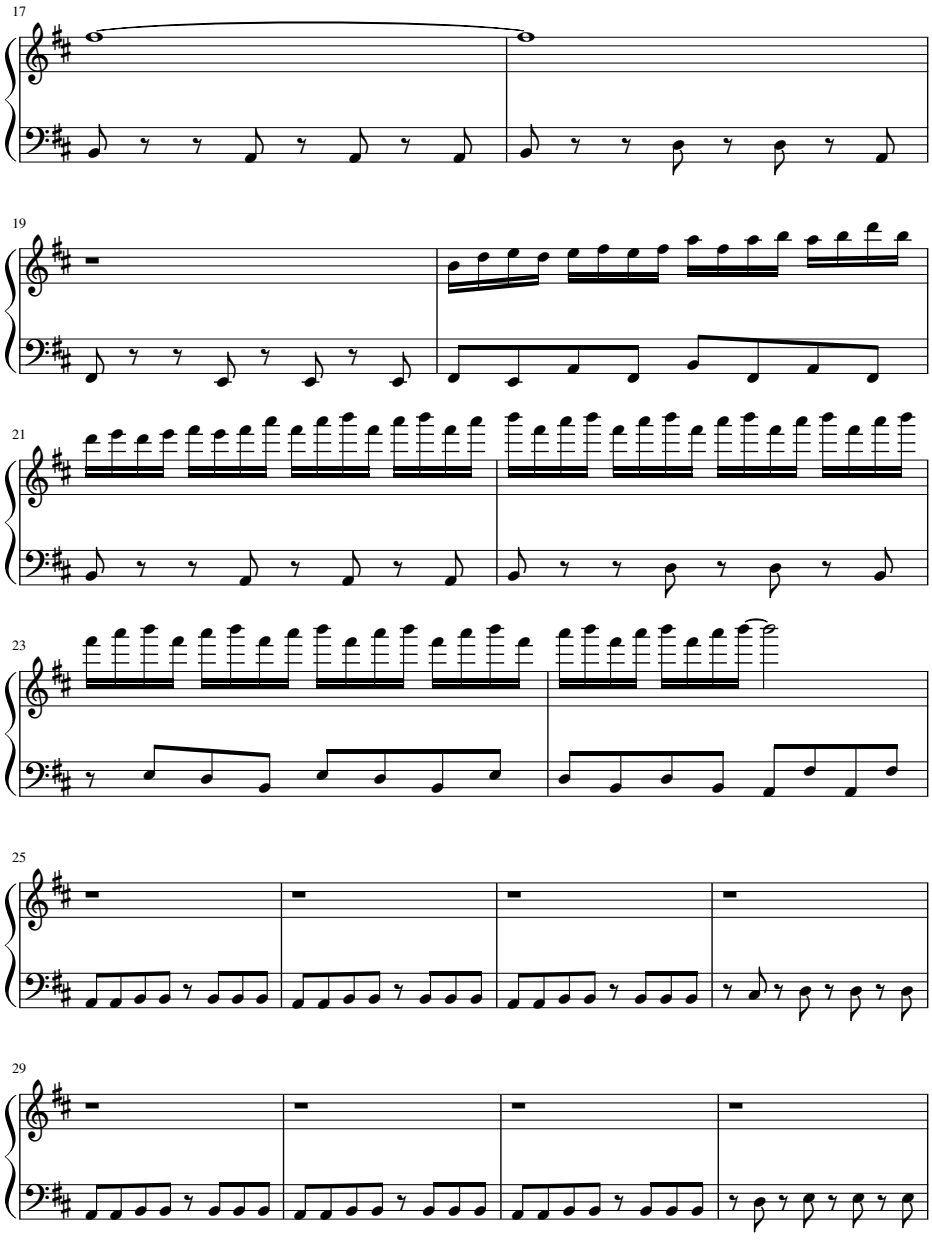

53
8
58
8
63
8
66
8
69
8

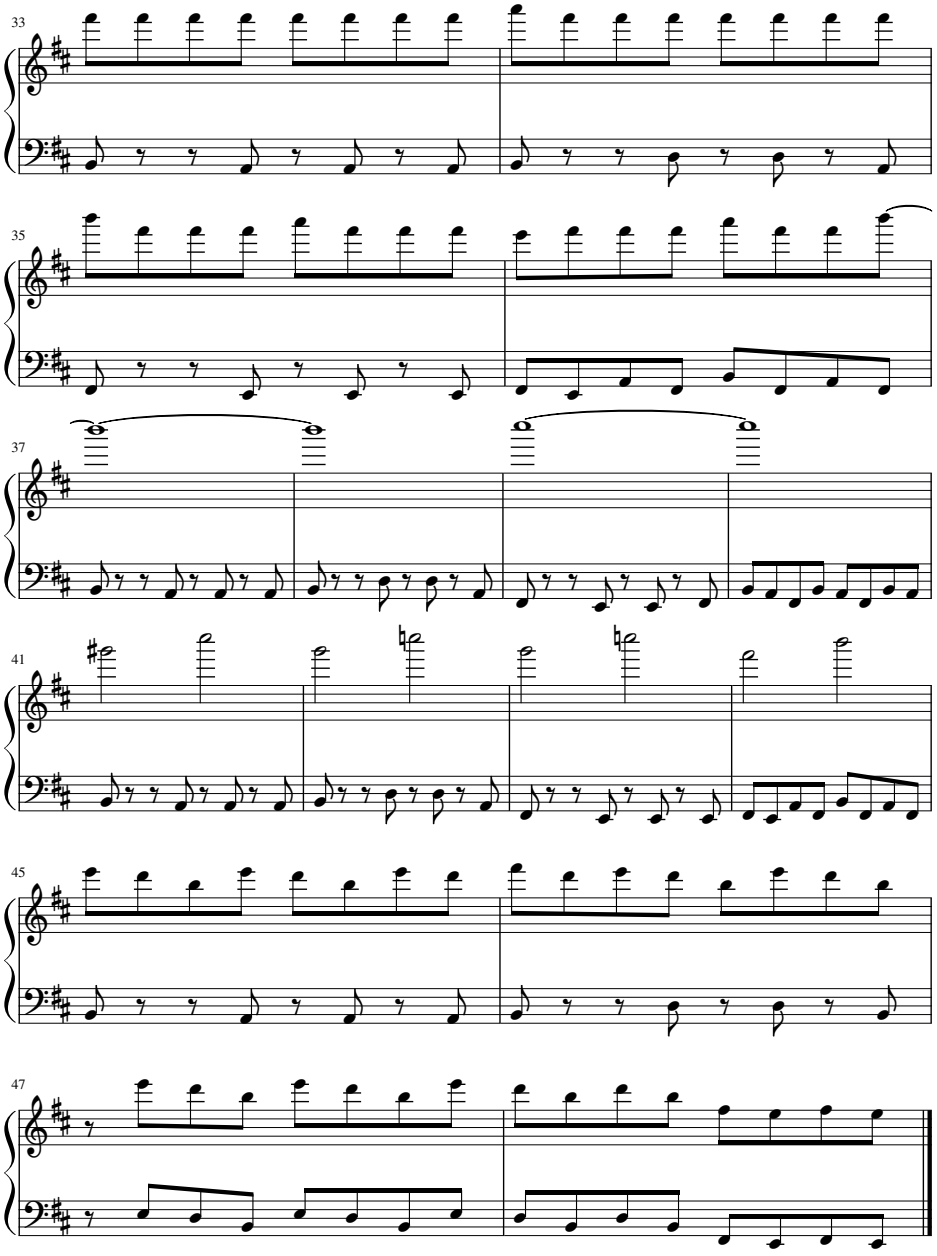
Liite 2: Stunts*-pelin tuottamat piipperikomennot esitettynä graafisesti.*



Liite 3: Stunts*-musiikin transkriptio. Musiikin säveltäjät Krisjan Hatlelid, Brian Plank ja Michael Sokyrka. Distinctive Software / Broderbund 1990*




17
19
21
23
25
29


33
35
37
41
45
47