Tuomo Heikkinen

Virtuaali-instrumentin kehitys Unity-pelimoottorilla

Tradenomi

Tietojenkäsittely

Kevät 2021

KAMK • University of Applied Sciences

Tiivistelmä

Tekijä(t): Heikkinen Tuomo

Työn nimi: Virtuaali-instrumentin kehitys Unity-pelimoottorilla

Tutkintonimike: Tradenomi, tietojenkäsittely

Asiasanat: virtuaali-instrumentti, MIDI, Unity-pelimoottori, ohjelmointi, musiikki

Opinnäytetyön tavoitteena oli kehittää MIDI-koskettimilla soitettava virtuaali-instrumentti käyttäen pelinkehitykseen yleisimmin käytettyä Unity-pelimoottoria hyödyntäen sen tarjoamia ominaisuuksia.

MIDI on 1980-luvun alkupuolella kehitetty standardiprotokolla, jolla sitä hyödyntävät laitteet, kuten koskettimet, saadaan kommunikoimaan keskenään. MIDI-laitteilla voidaan nykyään ohjata myös virtuaali-instrumentteja, joilla MIDI-laitteilta saadut viestit muutetaan esimerkiksi ääneksi. Nämä äänet voivat olla erilaisilla aaltomuodoilla ja synteeseillä tuotettuja syntetisaattorimaisia ääniä tai oikeiden soittimien äänistä tehtyjä ääninäytteitä.

Opinnäytetyössä kehitetty virtuaali-instrumentti sisältää muutaman aaltomuodon sisältävän oskillaattorin sekä arpeggiaattorin. Lisäksi virtuaali-instrumentin ääntä voidaan muokata hyödyntämällä Unity-pelimoottorista löytyvillä efekteillä.

Virtuaali-instrumentin toimivuudesta voidaan päätellä Unity-pelimoottorin soveltuvan myös ääntä käyttävien sovellusten kehitykseen pelien lisäksi. Unity-pelimoottorista puuttuu MIDI-laitteiden tuki, mutta se voidaan lisätä kolmannen osapuolen liitännäisellä, kunnes Unity-pelimoottorin kehittäjät lisäävät sellaisen itse.

Abstract

Author(s): Heikkinen Tuomo

Title of the Publication: Developing a Virtual Instrument with Unity Game Engine

Degree Title: Bachelor of Business Administration, Business Information Technology

Keywords: virtual instrument, MIDI, Unity game engine, programming, music

The goal of this Bachelor's thesis was to develop a virtual instrument with Unity game engine that can be used with MIDI keyboard and that uses the features that Unity game engine provides.

MIDI is a standardized protocol that was made in early 1980s which can be used to make devices, such as keyboards, to communicate with each other. With MIDI devices, it is possible to control virtual instruments which translate data from MIDI devices to sounds, for example. These sounds can be synthesizer-like sounds which are made with different waveforms and syntheses or sound samples from real instruments.

The virtual instrument developed in this thesis contains an oscillator that can use various waveforms and an arpeggiator. The virtual instrument's sound can be modified using filter effects that can be found in Unity game engine.

Based on the functionality of the virtual instrument, it can be concluded that Unity game engine is also suitable for development of audio applications besides games. Unity game engine lacks support for MIDI devices but it can be added to the game engine using a third-party plugin until it is added by the developers of the game engine.

Sisällys

1 Johdanto 1

2 MIDI 2

2.1 MIDI:n kehitys 2

2.2 MIDI-laitteet 4

2.3 MIDI-koskettimet 4

3 Virtuaali-instrumentit 6

3.1 Syntetisaattorit 8

3.2 Sample-pohjaiset 9

4 Virtuaali-instrumentin kehitys Unity-pelimoottorissa 11

4.1 MIDI-koskettimet Unity-pelimoottoriin 12

4.2 Virtuaali-instrumentin tekeminen 14

4.3 Lisäominaisuudet 18

4.4 Graafisen ulkoasun suunnittelu 20

4.5 Jatkokehitys 22

5 Pohdinta 24

6 Yhteenveto 25

Lähteet 26

Liitteet

Symboliluettelo

aaltomuoto	äänisignaalin muoto tietyn aikavälin sisällä
arpeggiaattori	engl. arpeggiator, toistaa sarjan nuotteja tietyllä nopeudella
MIDI	Musical Instrument Digital Interface, välittää tietoa digitaalisten instrumenttien ja tietokoneiden välillä
oskillaattori	tuottaa jatkuvaa aaltomuotoa
virtuaali-instrumentti	ohjelma, joka tuottaa ääntä esimerkiksi MIDI:n avulla

1 Johdanto

Virtuaali-instrumentit kuuluvat nykyään kaikenlaisessa musiikissa, niin studionauhoituksissa kuin live-konserteissakin. Virtuaali-instrumentit voivat esimerkiksi korvata kokonaisen sinfoniaorkesterin, mikäli sellaisen käyttäminen ja äänittäminen ei ole mahdollista logistisista tai taloudellisista syistä. Virtuaali-instrumentit tuovat soittajan käsiin lähes rajattomat mahdollisuudet kokeilla erilaisia soittimia ja muokata niitä haluamansa kuuloisiksi.

Useimpia virtuaali-instrumentteja voidaan hallita MIDI-teknologialla, joka on 1980-luvun alusta lähtien mullistanut musiikkialaa luomalla standardin, jolla laitteet voivat kommunikoida keskenään laitevalmistajasta riippumatta [1]. Nykyään markkinat ovat täynnä MIDI-koskettimia, jotka voidaan yhdistää tietokoneeseen USB-liitännällä aikaisemmin käytettyjen MIDI-johtojen sijaan. Virtuaali-instrumenttien hallinta MIDI-koskettimilla missä ja milloin vain on helpompaa ja nopeampaa kuin koskaan.

Opinnäytetyön tavoitteena on valmistaa Unity Technologiesin kehittämällä Unity-pelimoottorilla virtuaali-instrumentti, jota pystytään käyttämään ulkoisten MIDI-koskettimien kanssa. Opinnäytetyössä pyritään selvittämään, mitä ominaisuuksia Unity-pelimoottori tarjoaa ja voidaanko niitä hyödyntää virtuaali-instrumentissa.

Opinnäytetyö on jaettu kahteen osaan, teorian ja käytännön osuuteen. Teoriaosuudessa käydään läpi mitä tarkoitetaan MIDI-termillä, mitä virtuaali-instrumentit ovat ja millä eri tavoilla ne tuottavat ääntä. Käytännön osuudessa käydään läpi prosessia, kuinka virtuaali-instrumentti luodaan Unity-pelimoottorissa ja miten sitä pystytään monipuolistamaan Unity-pelimoottorin tarjoamilla ominaisuuksilla.

2 MIDI

Musical Instrument Digital Interface, lyhennettynä MIDI, on musiikkiteknologian standardisoitu protokolla, jonka avulla luodaan yhteys digitaalisten soittimien, kuten digitaalisyntetisaattoreiden ja tietokoneiden välille. Muusikot ja säveltäjät ovat käyttäneet MIDI:ä jo 1980-luvun alusta lähtien, ja sen suosio on vain kasvanut vuosikymmenten kuluessa leviten myös laitteista tietokoneohjelmiin, kuten peleihin. [1.] Nykyään MIDI:ä voidaan käyttää jopa valaistuksen ohjaamiseen.

2.1 MIDI:n kehitys

Universaalin MIDI-protokollan syntyä edelsi aika, jolloin eri laitevalmistajien syntetisaattorit eivät sopineet toimimaan keskenään, koska standardisointia ei ollut. Roland Corporationin Ikutarō Kakehashi esitti vuonna 1981 idean standardoinnista Sequential Circuitsin toimitusjohtajalle, Dave Smithille. Eri laitevalmistajien ja yhteistyökumppaneiden kanssa käydyn ideoinnin ja suunnittelun jälkeen, vuoden 1983 soitin- ja musiikkilaitevalmistajien NAMM-messuilla Roland ja Sequential Circuits esittelivät uuden MIDI-standardin, jota demonstroitiin yhdistämällä Rolandin Jupiter-6- ja Sequential Circuitsin Prophet 600-syntetisaattorit keskenään, jotka näkyvät kuvassa 1. Saman vuoden elokuussa julkaistiin viimeistelty versio standardista, nimeltään MIDI 1.0. [2.]

Kuvat 1 ja 2. Roland Jupiter-6- [3.] ja Sequential Circuits Prophet-600 -syntetisaattorit. [4.]

MIDI:n käyttäjien kannalta tärkein lisäys siihen on ollut General MIDI, joka esiteltiin vuonna 1991. General MIDI:n avulla pystyttiin lisäämään 128 soitinta ja ääniefektiä MIDI-laitteeseen. (Taulukko 1.) General MIDI:ä pidettiin kuitenkin rajoittuneena, jonka seurauksena Roland loi myöhemmin

samana vuonna General Standard MIDI:n, jolla jokainen 128 MIDI-kanavasta voi sisältää saman verran variaatioita. [5, s. 50.] General MIDI:ä laajennettiin vielä vuonna 1999 lisäämällä siihen uusia ominaisuuksia ja päivittämällä vanhempia toimintoja.

1	Acoustic Grand Piano	33	Acoustic Bass	65	Soprano Sax	97	FX 1 (rain)
2	Bright Acoustic Piano	34	Electric Bass (finger)	66	Alto Sax	98	FX 2 (soundtrack)
3	Electric Grand Piano	35	Electric Bass (pick)	67	Tenor Sax	99	FX 3 (crystal)
4	Honky-tonk Piano	36	Fretless Bass	68	Baritone Sax	100	FX 4 (atmosphere)
5	Electric Piano 1	37	Slap Bass 1	69	Oboe	101	FX 5 (brightness)
6	Electric Piano 2	38	Slap Bass 2	70	English Horn	102	FX 6 (goblins)
7	Harpsichord	39	Synth Bass 1	71	Bassoon	103	FX 7 (echoes)
8	Clavinet	40	Synth Bass 2	72	Clarinet	104	FX 8 (sci-fi)
9	Celesta	41	Violin	73	Piccolo	105	Sitar
10	Glockenspiel	42	Viola	74	Flute	106	Banjo
11	Music Box	43	Cello	75	Recorder	107	Shamisen
12	Vibraphone	44	Contrabass	76	Pan Flute	108	Koto
13	Marimba	45	Tremolo Strings	77	Blown Bottle	109	Kalimba
14	Xylophone	46	Pizzicato Strings	78	Shakuhachi	110	Bagpipe
15	Tubular Bells	47	Orchestral Harp	79	Whistle	111	Fiddle
16	Dulcimer	48	Timpani	80	Ocarina	112	Shanai
17	Drawbar Organ	49	String Ensemble 1	81	Lead 1 (square)	113	Tinkle Bell
18	Percussive Organ	50	String Ensemble 2	82	Lead 2 (sawtooth)	114	Agogo
19	Rock Organ	51	Synth Strings 1	83	Lead 3 (calliope lead)	115	Steel Drums
20	Church Organ	52	Synth Strings 2	84	Lead 4 (chiff lead)	116	Woodblock
21	Reed Organ	53	Choir Aahs	85	Lead 5 (charang)	117	Taiko Drum
22	Accordion	54	Voice Oohs	86	Lead 6 (voice)	118	Melodic Tom
23	Harmonica	55	Synth Voice	87	Lead 7 (fifths)	119	Synth Drum
24	Tango Accordion	56	Orchestra Hit	88	Lead 8 (bass + lead)	120	Reverse Cymbal
25	Acoustic Guitar (nylon)	57	Trumpet	89	Pad 1 (new age)	121	Guitar Fret Noise
26	Acoustic Guitar (steel)	58	Trombone	90	Pad 2 (warm)	122	Breath Noise
27	Electric Guitar (jazz)	59	Tuba	91	Pad 3 (polysynth)	123	Seashore
28	Electric Guitar (clean)	60	Muted Trumpet	92	Pad 4 (choir)	124	Bird Tweet
29	Electric Guitar (muted)	61	French Horn	93	Pad 5 (bowed)	125	Telephone Ring
30	Overdriven Guitar	62	Brass Section	94	Pad 6 (metallic)	126	Helicopter
31	Distortion Guitar	63	Synth Brass 1	95	Pad 7 (halo)	127	Applause
32	Guitar Harmonics	64	Synth Brass 2	96	Pad 8 (sweep)	128	Gunshot

Taulukko 1. General MIDI:n soittimet ja niiden numeroinnit eriteltyinä soitinryhmittäin. [6.]

MIDI:n seuraajaa saatiin odottaa vuosikymmeniä, kunnes tammikuussa 2020 MIDI-valmistajien liitto hyväksyi MIDI:n 2.0-version. Uudessa versiossa MIDI-laitteet viestivät toistensa kanssa, eikä tieto mene johdossa vain yhteen suuntaan kuten aiemmin. Tällä laitteet voivat nyt esimerkiksi määrittää automaattisesti asetuksensa toisilleen sopivaksi. Vanhemmat MIDI 1.0 -laitteet toimivat edelleen taaksepäin yhteensopivuuden avulla, vaikka toinen laitteista olisi MIDI 2.0 -yhteensopiva. MIDI 2.0 ei korvaa aiempaa versiota vaan laajentaa sitä. [7.]

2.2 MIDI-laitteet

Erilaisia laitteita, jotka käyttävät MIDI:ä, on useita, mutta yleisimpiä näistä ovat MIDI-kontrollerit. MIDI-kontrollerit lähettävät MIDI-viestejä, jotka muutetaan vastaanottajan päässä toiminnoiksi, kuten esimerkiksi ääniksi. MIDI-kontrollereista yleisimpiä ovat MIDI-koskettimet, jotka muistuttavat kannettavaa pianoa tai syntetisaattoria, ja pad-kontrollerit, joissa on useita kosketusherkkiä painikkeita, joita käytetään myös esimerkiksi rumpukoneissa ja sampler-laitteissa. Yleensä MIDI-koskettimissa voi olla myös samanlaisia painikkeita kuin pad-kontrollereissa. Kokeellisimpia MIDI-kontrollereita edustaa esimerkiksi Donald Buchlan 1990-luvulla kehittämä Buchla Lightning, jota käytetään kahdella sauvalla, joiden sijaintia ja liikettä seurataan infrapunan avulla. [8.]

MIDI:ä voidaan ohjata myös kielisoittimilla, kuten kitaralla. Tämä kuitenkin vaatii erillisen mikrofonin ja joissain tapauksissa ohjausyksikön, jotka muuttavat kitaran jokaisen kielen värinän MIDI-muotoon tai MIDI-muuntajan, jolloin kitarasta tuleva äänisignaali muutetaan MIDI-muotoon yksittäisten kielien sijaan. [9.] Tämä voidaan tehdä myös ohjelmallisesti tietokoneella. Esimerkiksi Jam Originin kehittämä MIDI Guitar 2 -ohjelma muuttaa kitaran polyfoniseksi MIDI-kontrolleriksi, jolloin useampi kieli voi soida samaan aikaan ilman, että äänenseuranta kärsii, mikä on usein ongelmana MIDI-muuntajissa, kun erillistä MIDI-mikrofonia ei ole käytössä. [10.]

2.3 MIDI-koskettimet

MIDI-koskettimia on useita eri kokoja, joissa on yleisimmin 25–88 kosketinta. Koskettimien lisäksi MIDI-koskettimet voivat sisältää esimerkiksi erilaisia säätimiä, nuppi- ja liukusäätimistä pitch bend- ja modulaatioefektien säätöpyöriin. Vanhemmat MIDI-koskettimet käyttävät pelkästään 5-pinnisiä MIDI-johtoja siirtäessään dataa, mutta uudemmat MIDI-koskettimet sisältävät USB-liitännän, jonka kautta se voidaan yhdistää helposti tietokoneeseen eikä se tarvitse silloin erillistä virtalähdettä, koska USB-johdolla siirtyvät niin MIDI-viestit kuin tarvittava jännite. MIDI-johtoja käytettäessä tarvitaan kaksi johtoa, koska sisään- ja ulostuloille on omat liitännät.

MIDI-koskettimet voivat sisältää muitakin liitäntöjä, kuten esimerkiksi kuvassa 3 näkyy USB- ja MIDI-liitäntöjen lisäksi liitännät ekspressio- ja sustain-pedaaleille. Pianoistakin löytyvä, kaikupedaalin nimelläkin tunnettu, sustain-pedaali jättää äänen kuulumaan, vaikka kosketinta ei enää painettaisikaan. Ekspressiopedaalilla voidaan hallita tiettyjä arvoja, kuten vaikka äänenvoimakkuutta. Tätä liitäntää harvemmin löytyy pienemmistä MIDI-koskettimista.

Kuva 3. M-Audio Axiom 49-MIDI-koskettimien takapaneeli, jossa liitännät ja virtakytkin.

MIDI-koskettimien koskettimia on kolmea eri tyyppiä, ja ne jaotellaan painotuntuman mukaan painotettuihin, puolipainotettuihin tai painottamattomiin koskettimiin. Painottamattomat koskettimet toimivat pelkillä jousilla, kun taas painotetut ja puolipainotetut koskettimet pyrkivät jäljittelemään pianon koskettimia käyttämällä painoja ja jousia, koska MIDI-koskettimissa ei ole pianon kieliä tai vasaroita, joilla saadaan tietty pianon soittotuntuma. [11.]

MIDI-koskettimien hinnat liikkuvat noin viidenkymmenen euron ja yli tuhannen euron välillä riippuen koosta ja lisäominaisuuksista. Pienimmät 25-koskettimen MIDI-koskettimet ovat yleensä 50–300 euron välillä, kun taas isoimmat 88-koskettimen MIDI-koskettimet 200–1400 euron välillä. [12.] Hintaa nostavat koskettimien määrän lisäksi muun muassa koskettimien tyyppi, erilaisten säätimien määrä ja ominaisuudet, kuten musiikkiohjelmistojen ja virtuaali-instrumenttien hallinnointi suoraan MIDI-koskettimista.

3 Virtuaali-instrumentit

Virtuaali-instrumentteihin liitetään yleensä sana VST-liitännäinen, jonka lyhenne VST tulee sanoista Virtual Studio Technology, johon kuuluvat instrumenttien lisäksi efektit ja MIDI-efektit. VST-instrumenteilla luodaan ääntä ja VST-efekteillä muokataan ääntä, kun taas VST MIDI-efektit lähettävät tietoa MIDI:n kautta laitteille ja muille VST-instrumenteille. [13.] VST-liitännäisiä käytetään ensisijaisesti esimerkiksi äänituotantoon tarkoitetuissa ohjelmistoissa, mutta jotkin VST-instrumentit toimivat myös erillisinä sovelluksina, jolloin äänitysohjelmaa ei tarvitse avata. Tunnetuimpia VST-liitännäisten valmistajia ovat esimerkiksi Native Instruments ja Waves, mutta suosittuja liitännäisten valmistajia on kuitenkin lukuisia. VST-liitännäiset eivät ole kuitenkaan ainoa liitännäistyyppi, sillä esimerkiksi Apple-tietokoneiden Logic Pro -ohjelmisto käyttää vain Audio Units -liitännäisiä, ja Pro Tools -ohjelmistot käyttävät joko Avid Audio Extension- tai Real Time Audio Suite -liitännäisiä. [14.] Tämän takia virtuaali-instrumenteista onkin usein ladattavissa tai asennettavissa erilliset liitännäisversiot riippuen siitä, mitä ohjelmistoa käyttää.

Virtuaali-instrumentti voi olla yksittäinen soitin, jonka ääntä voidaan muokata tai sitten se voi olla useita erilaisia instrumentteja sisältävä kokonaisuus, kuten kuvassa 4 näkyvä Native Instrumentsin tekemä Kontakt-ohjelmisto. Vaikka ilmaisia ja myös laadukkaita virtuaali-instrumentteja löytyy internetistä ladattavana, laadukkaimmat virtuaali-instrumentit ovat yleensä maksullisia, joiden hinta voi nousta satoihin euroihin riippuen niiden laajuudesta ja niiden tuottaman äänen aitoudesta.

Virtuaali-instrumentteja myydään myös paketteina, kuten esimerkiksi Native Instrumentsin Komplete-kokoelma, joka sisältää erilaisten instrumenttien lisäksi myös VST-efektejä. Vuoden 2021 alussa uusimman Komplete 13 -kokoelman perusversio kustantaa 599 dollaria, jonka sisältämien VST-liitännäisten yhteisarvoksi erillisinä tuotteina tulisi 6888 dollaria. Laajin ja samalla kallein Komplete 13 -versio taas sen sijaan tuo 17440 dollarin arvoiset liitännäiset ja soitinkirjastot hintaan 1599 dollaria. [15.] Kuvan 4 virtuaali-instrumentit ovat Komplete 12 -perusversiosta, jotka myös seuraavat Komplete-versiot sisältävät, joihin päivitys on mahdollista alennettuun hintaan.

Kuva 4. Kuvakaappaus Analog Dreams 2.0-virtuaali-instrumentista Kontakt-ohjelmistossa.

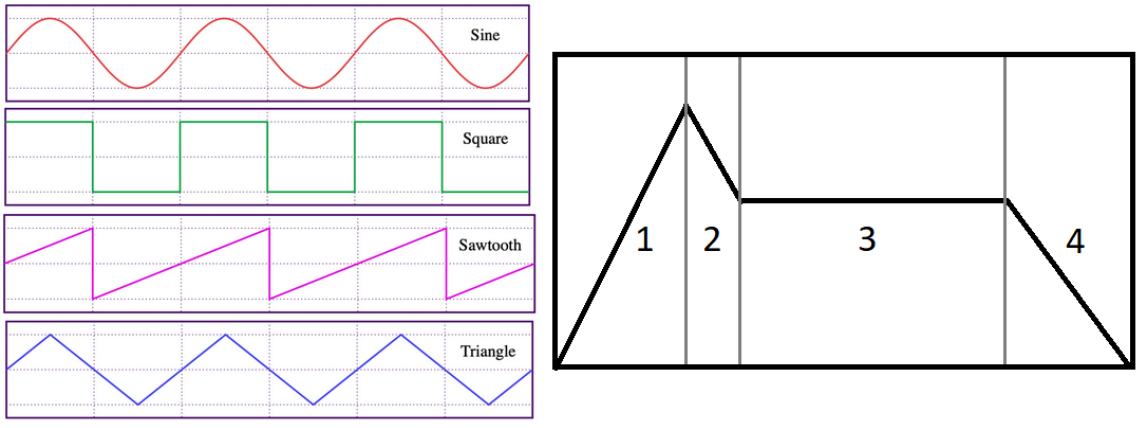
Virtuaali-instrumentit sisältävät usein suuren määrän valmistajan tekemiä valmiita esiasetuksia, jotka muuttavat virtuaali-instrumentin ääntä erilaisilla asetusten yhdistelmillä. Esiasetukset on myös yleensä kategorisoitu niiden tuottaman äänen perusteella, jotta käyttäjä voisi helpommin löytää etsimänsä esiasetuksen. Virtuaali-instrumentit antavat myös käyttäjän tallentaa tekemiään esiasetuksia, jolloin käyttäjä voi jakaa niitä eteenpäin esimerkiksi internetin kautta. Esiasetukset ovat hyvä keino tutustua uuden virtuaali-instrumentin ominaisuuksiin, ja ne voivat toimia jopa inspiraation lähteenä, jos käyttäjä löytää itseänsä miellyttävän äänen.

MIDI-koskettimien säätimet voidaan helposti asettaa muuttamaan tiettyjä arvoja joissakin virtuaali-instrumenteissa käyttämällä MIDI Learn- tai MIDI Assign -ominaisuutta, jolloin seurataan, mitä MIDI-tiedon seassa olevaa MIDI Control Change -viestiä muutetaan, kun säädintä käytetään ja tällöin yhdistetään säätimen ja virtuaali-instrumentin parametrin arvot toisiinsa. Sama toimii kaikkiin nuppi- ja liukusäätimiin, nappeihin ja pad-painikkeisiin. Joissakin tapauksissa muutettavaa MIDI Control Change -viestiä ei voida automaattisesti tunnistaa, jolloin se täytyy valita manuaalisesti listasta. [16.]

3.1 Syntetisaattorit

Virtuaali-instrumentteja, joiden tuottama ääni syntyy instrumentissa itsessään, voidaan kutsua syntetisaattoreiksi. Kuten fyysisissä syntetisaattoreissa, ääni luodaan erilaisten synteesien avulla, jolloin sähköisiä signaaleja muutetaan ääniaalloiksi, mutta digitaalisessa muodossa. Yleisimmistä synteeseistä additiivisessä synteesissä yhdistetään aaltomuotoja toisiinsa, vähentävässä synteesissä tiettyjä taajuuksia suodatetaan pois aaltomuodosta ja FM-synteesissä aaltomuotoa moduloidaan eli muutetaan sitä toisten ääniaaltojen taajuuksilla. Vaihtoehtoisesti synteesi voidaan tehdä myös valmiiksi äänitetyistä ääninäytteistä tai tallennetuista aaltomuodoista. [17.]

Synteesi tapahtuu oskillaattoreissa, joissa aaltomuotoa toistetaan halutulla taajuudella. Oskillaattoreissa on yleensä valittavina aaltomuotoina sini-, kantti-, sahalaita- ja kolmioaaltomuodot, jotka näkyvät kuvassa 5. Virtuaali-instrumenttien oskillaattoreissa on usein enemmän kuin yksi oskillaattori, ja niissä kaikissa pystyy säätämään useita arvoja, joilla niiden aaltomuotoa saadaan muokattua. Oskillaattorin ääntä voidaan muokata myös esimerkiksi suodattimilla, joilla vähennetään äänestä tiettyjä taajuuksia. Valmiina suodattimina on yleensä ali- ja ylipäästösuodattimet, joilla suodatetaan alemmat tai ylemmät taajuudet ja joissain tapauksissa myös kaistanpäästösuodatin, jolla voidaan jättää suodattamatta tietyt taajuudet. Muita keinoja muokata ääntä on LFO-oskillaattori, jonka tuottamaa todella matalaa taajuutta käytetään muuttamaan muita oskillaattorin arvoja sekä verhokäyrä, jolla erotetaan äänestä neljä eri vaihetta, joita voidaan muokata. Nämä vaiheet näkyvät kuvassa 6, ja ne ovat äänen huippuunsa nouseminen, normaaliin äänentasoon vaimeneminen, äänentason pysyminen samana ja äänen lopullinen vaimeneminen. [18.]

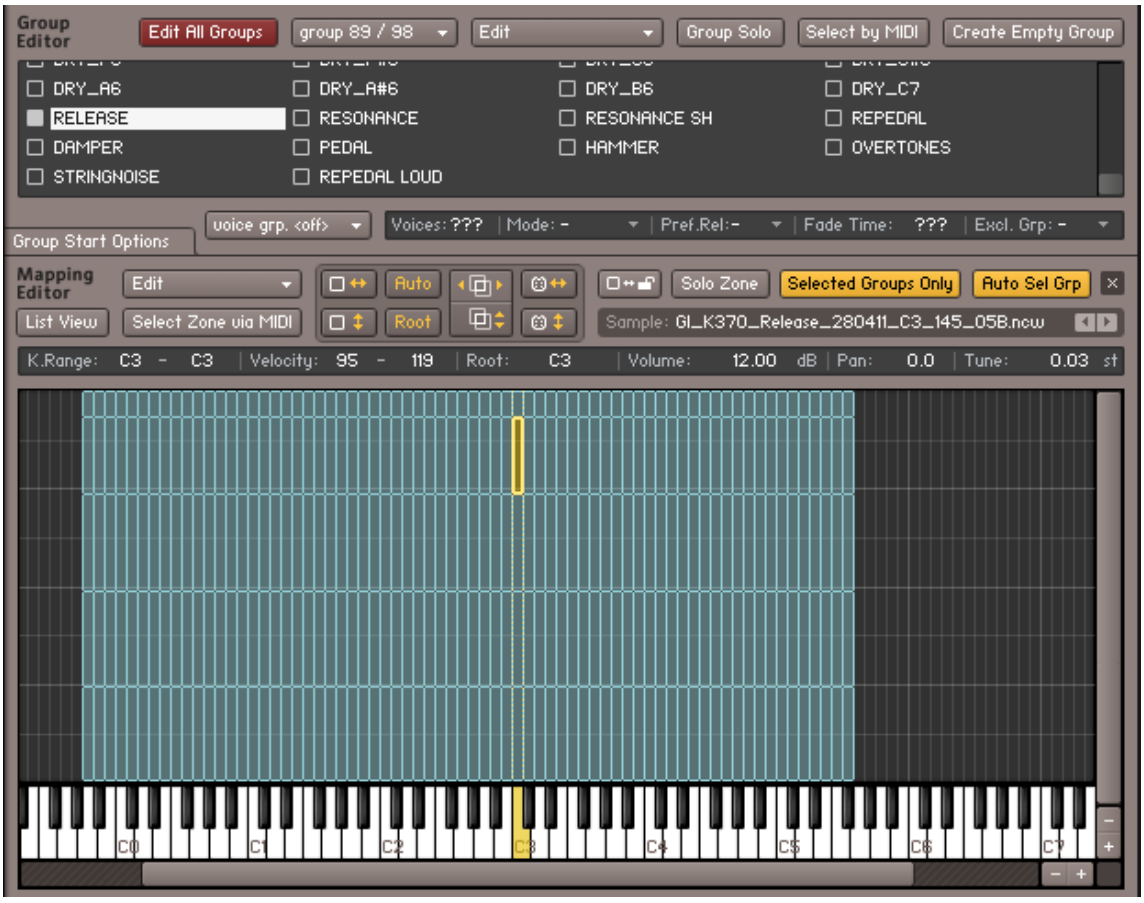


Kuvat 5 ja 6. Sini-, kantti-, sahalaita- ja kolmioaaltomuodot [18.] ja verhokäyrän vaiheet.

3.2 Sample-pohjaiset

Virtuaali-instrumentit, jotka esimerkiksi kuulostavat joltakin oikealta soittimelta, käyttävät sampleja tuottaakseen äänen. Samplet ovat ääninäytteitä, joita voidaan käyttää sellaisenaan tai muokata niitä. Esimerkiksi oikean soittimen äänittäminen virtuaali-instrumentiksi voidaan tehdä äänittämällä jokaisesta nuotista tai vaikka joka kolmannesta nuotista sample. Mikäli äänitetyt nuotit eivät ole vierekkäisiä nuotteja, voidaan nostaa tai laskea sampleja digitaalisesti, jolloin jokainen nuotti saadaan lisättyä virtuaali-instrumenttiin. Nuoteista voi olla eri versioita, jotka voivat olla soitettu eri tavalla, kuten hiljemmin ja voimakkaammin soitetut nuotit. [19.]

Esimerkiksi Native Instrumentsin Kontakt-ohjelmistossa virtuaali-instrumentille voidaan luoda ryhmiä, jolloin voidaan asettaa erilaiset samplet omiin ryhmiinsä, jotka käyttävät omia asetuksiaan erillään muista. Kuvassa 7 näkyy, kuinka ryhmät ja tietyn ryhmän samplet voidaan asetella Kontakt-ohjelmistossa. Vaaleansiniset osat ovat sampleja, jotka toistetaan, mikäli MIDI-kosketti-milla soitetun tai äänitysohjelmassa jo ennalta nauhoitetussa MIDI-raidassa toistuvan nuotin voimakkuus eli velocity on samplelle määritettyjen raja-arvojen välillä.

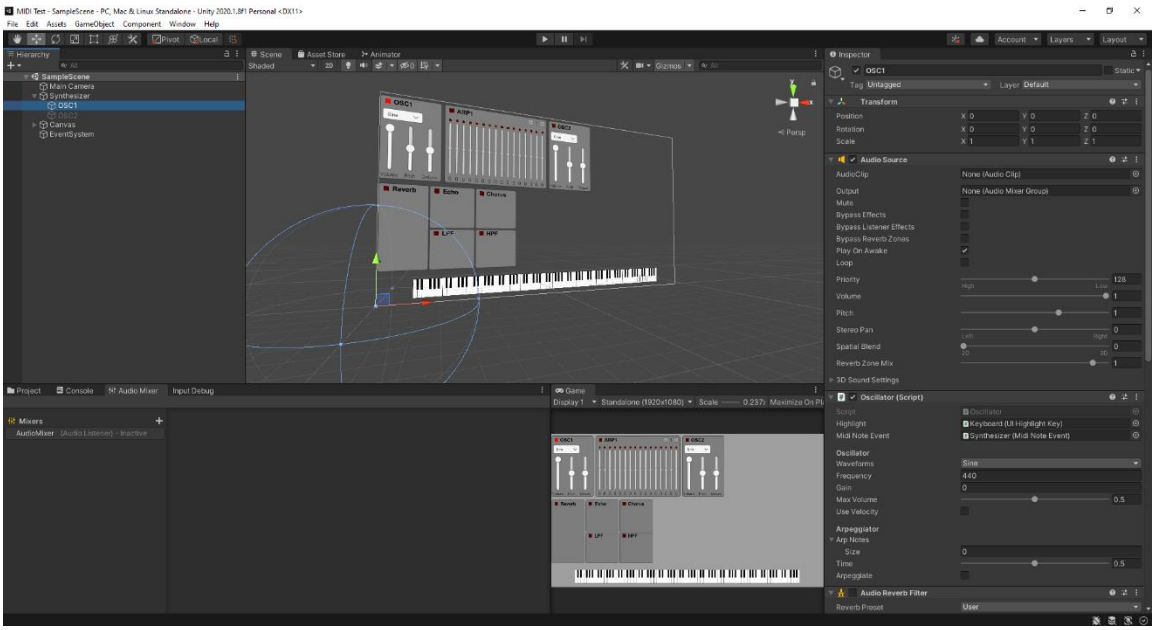


Kuva 7. Samplejen ryhmittely- ja asetteluikkunat Kontakt-ohjelmistossa.

Kuten syntetisaattorin tapaisissa virtuaali-instrumenteissa, sample-pohjaisissakin ääntä voidaan muuttaa erilaisten suodattimien ja muiden työkalujen avulla. Lisäksi usein on tarjolla myös erilaisia efektejä ja miksaustyökaluja. Joissakin virtuaali-instrumenteissa voi käyttää omia sampleja tai muokata olemassa olevia sampleja, kun taas toiset antavat vain mahdollisuuden toistaa valmiita sampleja ilman minkäänlaisia muokkaustyökaluja. [20.]

4 Virtuaali-instrumentin kehitys Unity-pelimoottorissa

Virtuaali-instrumentin kehitykseen valittiin kehitysympäristöksi Unity Technologiesin kehittämä Unity-pelimoottori, jonka 2020.1.8f1-versiota käytettiin tässä työssä. (Kuva 8.) Vaikka Unity-pelimoottori on ensisijaisesti tarkoitettu pelinkehitykseen, on sitä mahdollista myös käyttää erilaisten sovelluksien kehitykseen. Valintaan vaikutti myös henkilökohtainen kokemus Unity-pelimoottorista ja sen käyttämästä C#-ohjelmointikielestä. Muina työkaluina käytettiin Microsoftin kehittämää Visual Studio 2019 -kehitysohjelmistoa, jolla muokattiin Unity-pelimoottorissa luotuja C#-skriptitiedostoja, joihin kaikki virtuaali-instrumentin C#-ohjelmointikoodi on kirjoitettu.



Kuva 8. Kuvakaappaus Unity 2020.1.8f1-pelimoottorin muokkausnäkymästä.

Tässä työssä kehitettävän virtuaali-instrumentin tärkein osa on oskillaattori. Se toimii valitsemalla aaltomuodon, jota syötetään Unity-pelimoottorin OnAudioFilterRead()-metodille, jonka Audio Source -komponentti toistaa äänenä. Virtuaali-instrumenttia varten valittiin yleisimmät aaltomuodot, joita muista oskillaattoreista löytyy. Oskillaattorin lisäksi virtuaali-instrumenttiin saatiin lisättyä arpeggiaattori ja erilaisia efektejä, jotka löytyvät Unity-pelimoottorista suoraan.

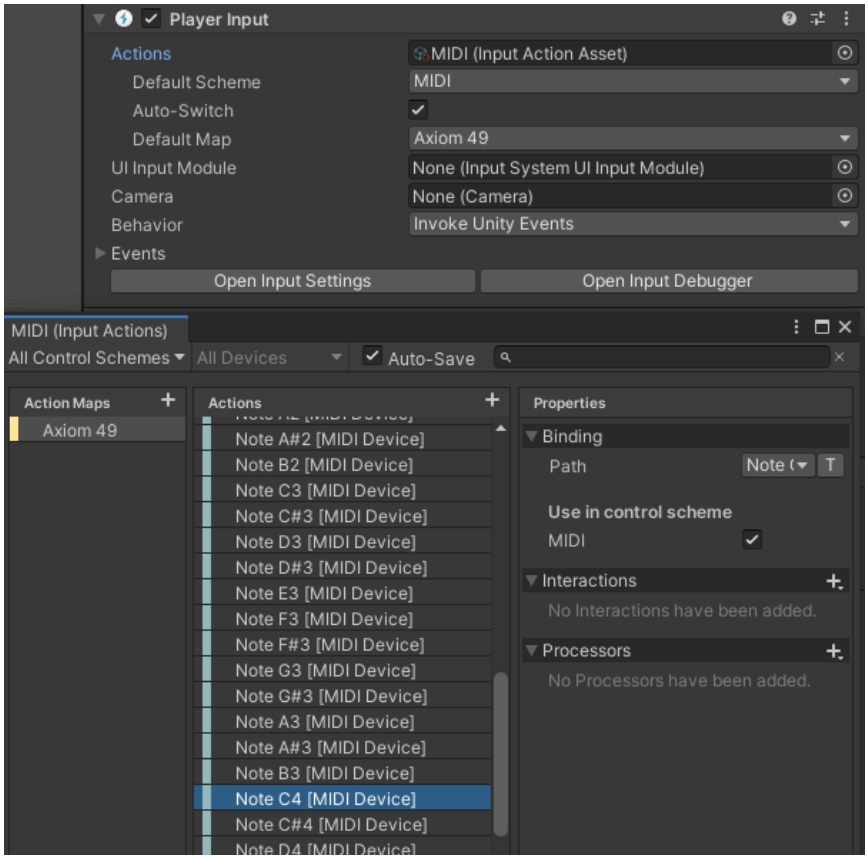
Virtuaali-instrumenttia kehitettäessä sitä käytettiin kahden eri valmistajan MIDI-koskettimilla, jotka olivat erikokoisia, jotta saatiin todettua virtuaali-instrumentin toimivuus edes kahdella laitteella. MIDI-koskettimina toimivat M-Audion Axiom 49, jossa on 49 kosketinta, ja Arturian Minilab MkII, jossa on 25 kosketinta. MIDI-koskettimien koot eivät vaikuttaneet virtuaali-instrumentin

toimintaan, koska niissä on sisäänrakennettuna ominaisuus, jolla niiden sävelkorkeutta voi vaihtaa.

4.1 MIDI-koskettimet Unity-pelimoottoriin

Unity-pelimoottori ei ainakaan vielä tue suoraan MIDI-laitteita, joten tuki niille on joko tehtävä itse tai käytettävä kolmannen osapuolen liitännäistä. Työn alussa harkittiin tuen tekemistä itse, mutta sen vaatiman ajan ja teknisen osaamisen takia päädyttiin lopulta käyttämään ohjelmakehitysprojektien versionhallintaan tarkoitetulta GitHub-sivustolta löytyvää Keijiro Takahashin tekemää Minis: MIDI Input for New Input System -liitännäistä, joka lisää tuen MIDI-syötteelle Unity-pelimoottoriin. [21.] Kyseinen liitännäinen nimensä mukaisesti käyttää Unity-pelimoottorin uudempaa syötejärjestelmää, jonka käyttöä täytyi opetella ensimmäisenä.

Liitännäisen asennuksen jälkeen Unity-pelimoottori tunnisti tietokoneeseen liitetyt MIDI-koskettimet, ja ne pystyttiin lisäämään käytettäviin laitteisiin. Uusi syötejärjestelmä vaati toimiakseen Player Input -komponentin, jota varten täytyi luoda myös uusi Input Action Asset -tiedosto. Tähän tiedostoon määritetään kaikki toiminnot, joita käytetään ja mikä syöte tekee kyseisen komennon. Tässä tapauksessa lisättiin kaikki 128 M-Audio Axiom 49-MIDI-koskettimien nuottia (kuva 9), jotka toimivat samalla tavalla myös muissa MIDI-koskettimissa, joten samoja toimintoja ei tarvitse lisätä uudestaan, vaikka laite vaihtuisi.



Kuva 9. Kuvakaappaus Player Input -komponentista ja Input Actions -ikkunasta Unity-pelimoottorissa.

Minis: MIDI Input for New Input System -liitännäisen mukana tulee MIDI Device Assigner -skriptitiedosto, jota tarvitaan MIDI-laitteiden tunnistukseen. Sen avulla pystytään lukemaan viimeisin MIDI-koskettimista painettu nuotti ja sen painamisen voimakkuus. Tätä varten luotiin MidiNoteEvent-skriptitiedosto, joka lukee nämä tiedot ja säilöö ne currentNote- ja currentVelocity-muuttujiin. (Kuva 10.)

```
using UnityEngine;
using UnityEngine.InputSystem;

public class MidiNoteEvent : MonoBehaviour
{
    public int currentNote = 0;
    public float currentVelocity = 0f;

    private void Start()
    {
        InputSystem.onDeviceChange += (device, change) =>
        {
            if (change != InputDeviceChange.Added)
            {
                return;
            }

            var midiDevice = device as Minis.MidiDevice;

            if (midiDevice == null)
            {
                return;
            }

            // Tunnistaa soitetun nuotin ja sen voimakkuuden
            // ja asettaa ne ylläoleviin muuttujiin
            midiDevice.onWillNoteOn += (note, velocity) =>
            {
                currentNote = note.noteNumber;
                currentVelocity = velocity;
            };

            // Tunnistaa nuotin, jonka soittaminen lopetetaan
            midiDevice.onWillNoteOff += (note) =>
            {
                // Tee jotain, kun kosketin päästetään irti
            };
        };
    }
}
```

Kuva 10. Kuvakaappaus MidiNoteEvent-skriptitiedostosta Visual Studio 2019 -ohjelmistossa.

4.2 Virtuaali-instrumentin tekeminen

Varsinaisen virtuaali-instrumentin kehitys aloitettiin miettimällä, kuinka sen käyttämien nuottien taajuudet säilöttäisiin ja haettaisiin. Lopulta päädyttiin ratkaisuun, jolla nuottien taajuudet on tekstitiedostossa eroteltu puolipistettä käyttäen. Tekstitiedosto sijaitsee Unity-pelimoottorin luoman projektikansion Resources-alikansiossa. Unity-pelimoottorissa voidaan hakea koodilla erilaisia tiedostoja, jos ne sijaitsevat tässä kyseisessä kansiossa.

Tässä tapauksessa virtuaali-instrumentin käynnistyessä Oscillator-skriptitiedostossa haetaan GetFrequencies()-metodilla Frequencies-tekstitiedosto, joka sisältää taajuudet puolipisteillä eroteltuina ja luodaan väliaikainen tekstitaulukko, jossa jokainen taulukon teksteistä on yksi taajuuk-

sista. Sen jälkeen viedään jokainen taajuuksista numeroksi muutettuna pysyvään frequencies-numerotaulukkoon. (Kuva 11.) Taulukossa on täsmälleen saman verran taajuuksia kuin mahdollisia nuotteja MIDI-koskettimissa eli 128. Näin ollen ensimmäinen taajuus vastaa nuottia C-1 ja viimeinen taajuus nuottia G9.

```
private void GetFrequencies()
{
    // Haetaan taajuudet tekstitiedostosta
    var freqList = Resources.Load<TextAsset>("Values/Frequencies");
    // Erotellaan taajuudet puolipisteen avulla
    // ja lisätään ne väliaikaiseen tekstitaulukkoon
    string[] values = freqList.ToString().Split(';');

    for (int i = 0; i < frequencies.Length; i++)
    {
        // Lisätään taajuudet yksi kerrallaan
        // varsinaiseen taajuustaulukkoon
        frequencies[i] = double.Parse(values[i]);
    }
}
```

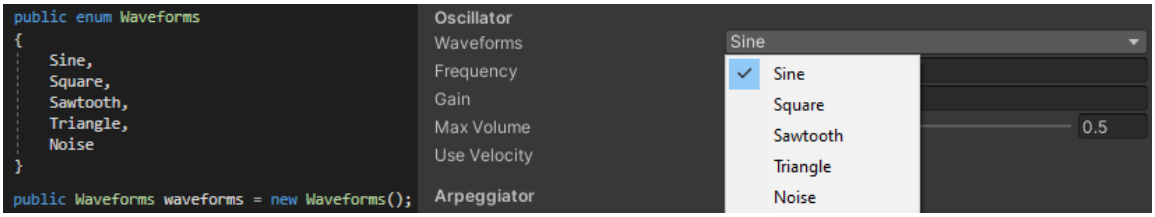
Kuva 11. Kuvakaappaus GetFrequencies()-metodista Oscillator-skriptitiedostossa Visual Studio 2019 -ohjelmistossa.

Minis: MIDI Input for New Input System -liitännäisen avulla voidaan tehdä if-ehtolause, joka seuraa IsPressed()-metodin kautta, onko MIDI-koskettimista painettu jokin nuotti. Jos vastaus on kyllä, määritetään nykyiseksi taajuudeksi painettua nuottia vastaava taajuus. Sama voidaan tehdä voimakkuuden kanssa, jolloin äänen voimakkuus on sama kuin painetun nuotin voimakkuus. Nämä taajuuden ja voimakkuuden määritykset on havainnollistettu esimerkissä kuvassa 12. Vaihtelevaa äänen voimakkuutta ei kuitenkaan laiteta tässä tapauksessa oletuksena päälle, vaan sen voi ottaa käyttöön halutessaan. Tällöin voimakkuus on aina suurin mahdollinen, jonka MIDI-koskettimista tuleva nuotti voi saada.

```
// Jos MIDI-koskettimia painetaan, tee jotain...
if (Minis.MidiDevice.current.IsPressed())
{
    // Toistettava taajuus on taajuustaulukosta taajuus X, joka vastaa painettua nuottia X
    frequency = frequencies[midiNoteEvent.currentNote];
    // Toistettavan äänen voimakkuus on painetun nuotin voimakkuus
    gain = midiNoteEvent.currentVelocity;
}
```

Kuva 12. Kuvakaappaus IsPressed()-metodin käyttöesimerkistä Visual Studio 2019 -ohjelmistossa.

Seuraavaksi oskillaattorin koodiin täytyy lisätä sini-, kantti-, sahalaita- ja kolmioaaltomuodot, joilla saadaan muutettua ääntä, jota oskillaattorilla tuotetaan. Lisätään myös kokeellisempi satunnaisgeneroinnilla kohinaa tuottava vaihtoehto. Nämä aaltomuodot lisätään julkiseen Waveforms-arvojoukkoon, josta luodaan uusi arvojoukko oskillattorin koodissa, jonka arvoa voidaan muuttaa pudotusvalikosta Unity-pelimoottorin puolella. (Kuva 13.)



Kuva 13. Kuvakaappaus Waveforms-arvojoukosta Visual Studio 2019 -ohjelmistossa ja sen muodostamasta pudotusvalikosta Unity-pelimoottorissa.

Aaltomuotojen toteutus tehdään Waveform()-metodissa, jossa katsotaan switch-case-rakenteella aikaisemmin luodun Waveforms-arvojoukon arvoa, joka kertoo, mitä aaltomuotoa käytetään tällä hetkellä. Näin saadaan laskettua oikea aaltomuoto oskillaattorille. (Kuva 14.)

```
private float Waveform()
{
    // Luodaan uusi aaltomuoto
    float waveform = 0f;

    // Katsotaan mikä aaltomuoto on valittuna arvojoukosta
    switch (waveforms)
    {
        // Jos valittuna on siniaalto, laske aaltomuodoksi tämä
        case Waveforms.Sine:
            waveform = gain * Mathf.Sin((float)phase);
            break;
```

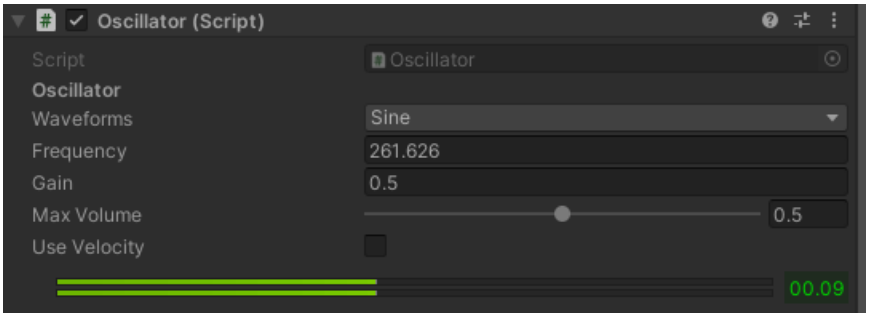
Kuva 14. Kuvakaappaus Waveform()-metodista ja siniaallon laskemisesta Visual Studio 2019 -ohjelmistossa.

Jotta oskillaattorista saadaan ääntä, täytyy käyttää Unity-pelimoottorin omaa OnAudioFilterRead()-metodia, jossa määritetään ulostuleva ääni käyttäen taajuutta ja aaltomuotoa. Äänisignaalia laskettaessa käytetään jakajana näytteenottotaajuutta, jolla määritetään, kuinka monta kertaa sekunnissa äänestä otetaan näytteitä [5, s. 186]. (Kuva 15.) Se voidaan määrittää itse, mutta tässä tapauksessa se haetaan Unity-pelimoottorin mikserin ääniasetuksien outputSampleRate-arvosta, kun virtuaali-instrumentti käynnistetään. Näytteenottotaajuuden oletusarvo Unity-pelimoottorissa on 48 kilohertsiä, joten virtuaali-instrumentin äänenlaatu on hieman parempi, mutta ei huomattava verrattuna yleiseen 44,1 kilohertsiin, jota käytetään esimerkiksi CD-levyissä.

```
public void OnAudioFilterRead(float[] data, int channels)
{
    // Lasketaan arvo ensin käyttämällä kaavaa:
    // koskettimien nuotin taajuus * 2π / näytteenottotaajuus
    double value = frequency * (2.0 * Mathf.PI) / sampleRate;

    for (int i = 0; i < data.Length; i += channels)
    {
        // Lisätään saatu arvo mukaan nykyiseen vaiheeseen
        phase += value;
        // Lasketaan data käyttäen aaltomuodoissa
        // äänenvoimakkuutta ja vaihetta
        data[i] = Waveform();

        if (channels == 2)
        {
            // Jos käytössä on 2 kanavaa eli stereoääni,
            // lisätään sama data myös toiselle kanavalle
            data[i + 1] = data[i];
        }

        if (phase > (2.0 * Mathf.PI))
        {
            // Hiljennetään ääntä vähitellen, jos
            // ääntä ei haluta toistaa enää
            phase -= (2.0 * Mathf.PI);
        }
    }
}
```

Kuva 15. Kuvakaappaus OnAudioFilterRead-metodista Visual Studio 2019 -ohjelmistossa.

OnAudioFilterRead-metodin käyttäminen vaatii Unity-pelimoottorin oman Audio Source -komponentin, jotta oskillaattorilla muodostettu ääni voidaan toistaa. Audio Source -komponentin avulla voidaan hallita myös esimerkiksi oskillaattorin äänenvoimakkuutta ja sävelkorkeutta. Kuvassa 16 nähdään kuinka OnAudioFilterRead-metodi luo Oscillator-skriptiin äänenvoimakkuusmittarin.

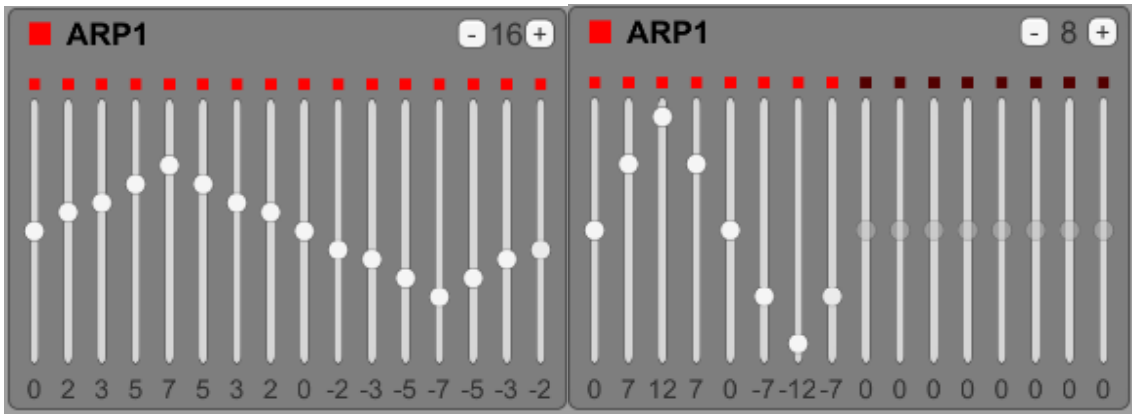


Kuva 16. Kuvakaappaus oskillaattorin skriptistä ja sen äänenvoimakkuusmittarista Unity-pelimoottorissa.

Virtuaali-instrumentin oskillaattori on nyt perusominaisuuksiltaan valmis. Pystytään vaihtamaan aaltomuotoa, säätää äänenvoimakkuutta ja sävelkorkeutta sekä voidaan ottaa huomioon halutessa koskettimen painamisvoimakkuus. Koska oskillaattorin koodi on yhdessä skriptitiedostossa, voidaan lisätä helposti toinenkin oskillaattori kopioimalla Unity-pelimoottorissa peliobjekti, joka sisältää Audio Source -komponentin ja Oscillator-skriptitiedoston. Tällöin saadaan kaksi oskillaattoria, joilla molemmilla voi olla erilaiset aaltomuodot ja ne toimivat samalla MIDI-koskettimien syötteellä.

4.3 Lisäominaisuudet

Oskillaattorien lisäksi virtuaali-instrumenttiin lisättiin arpeggiaattori, jolloin yhdellä koskettimen painalluksella saadaan toistettua useampia nuotteja tietyn ajan sisällä luoden arpeggion. Arpeggiaattoriin saa enimmillään 16 nuottia, joista jokaista voidaan säätää enimmillään 12 sävelaskelta eli oktaavin ylös tai alas. Kuvan 17 väliaikaiseen grafiikkaan ei ole vielä lisätty säädintä ajan säätöön ja sitä voidaan muuttaa tällä hetkellä vain Unity-pelimoottorista.



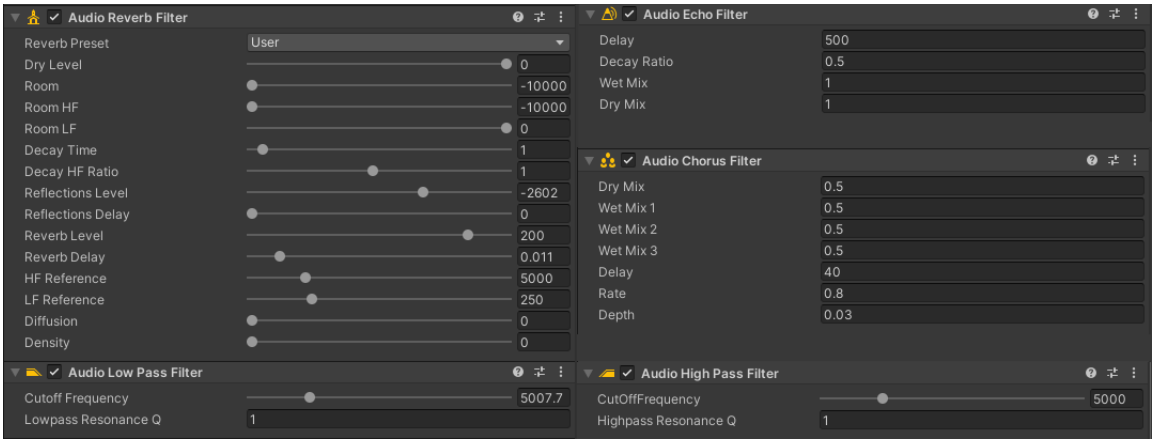
Kuva 17. Kuvakaappauksia virtuaali-instrumentin arpeggiaattorista Unity-pelimoottorissa.

Arpeggiaattorin toiminnallisuus on lisätty oskillaattorin skriptitiedostoon, johon säilötään nuottien määrä, niiden sävelkorkeudet ja aika, kuinka pitkään kestää, kunnes viimeinen nuotti on toistettu. Arpeggiaattori toimii IEnumerator-rajapinnan sisällä asettaen foreach-silmukassa taajuudeksi nykyisen nuotin, johon lisätään sävelkorkeuden muutos. (Kuva 18.) Rajapinta laitetaan jonoon, jonka kooksi on määritetty yksi, jolloin päällekkäistä arpeggiointia ei tapahdu.

```
private IEnumerator Arpeggiate()
{
    // Aseta tämä jonoon, johon mahtuu vain yksi IEnumerator-rajapinta,
    // jolloin useampaa päällekkäistä arpeggiointia ei tapahdu
    queue.Enqueue(Arpeggiate());

    while (Minis.MidiDevice.current.IsPressed())
    {
        // Tee tämä jokaiselle nuotille arpeggiossa
        foreach (int note in arpNotes)
        {
            // Mikäli nuotti on alempi kuin on mahdollista,
            // valitse alin taajuus
            if (midiNoteEvent.currentNote + note < 0)
            {
                frequency = frequencies[0];
            }

            // Mikäli nuotti on ylempi kuin on mahdollista,
            // valitse ylin taajuus
            else if (midiNoteEvent.currentNote + note > 127)
            {
                frequency = frequencies[127];
            }

            else
            {
                // Taajuus on nykyinen nuotti + arpeggiaattoriin
                // asetettu muutos alkuperäiselle nuotille
                frequency = frequencies[midiNoteEvent.currentNote + note];
            }

            if (!Minis.MidiDevice.current.IsPressed())
            {
                // Kun kosketinta ei paineta,
                // palautetaan taajuus alkuperäiseen
                frequency = frequencies[midiNoteEvent.currentNote];
                yield break;
            }

            // Odotetaan arpeggion nuottien välissä määritetty aika
            yield return new WaitForSeconds(time);
        }
    }

    yield break;
}
```

Kuva 18. Arpeggiaattorin toiminta Visual Studio 2019-ohjelmistossa.

Arpeggiaattorin lisäksi oskillaattoriin pystyy lisäämään efektejä, jotka löytyvät Unity-pelimoottorista jo valmiina. Yksi efekteistä on reverb-kaiku, jolla saadaan tuntumaan kuin ääni kuuluisi jossakin tilassa. Valmiista esiasetuksista löytyy tiloja, kuten auditorio, kylpyhuone ja parkkihalli, mutta kaiun voi säätää haluamakseen neljällätoista säätimellä, jotka muuttavat kaiun arvoja. Toinen kaiuista on delay-kaiku, jolla ääni toistuu useamman kerran jälkikäteen valitulla aikavälillä. Chorus-efekti saa äänen kuulostamaan siltä, että ääni kuuluisi useasti samaan aikaan pienillä eroilla luoden kuoromaisen efektin. Viimeiset efektit ovat ali- ja ylipäästösuodattimia, joilla voidaan poistaa matalia tai korkeita taajuuksia äänestä. (Kuva 19.)

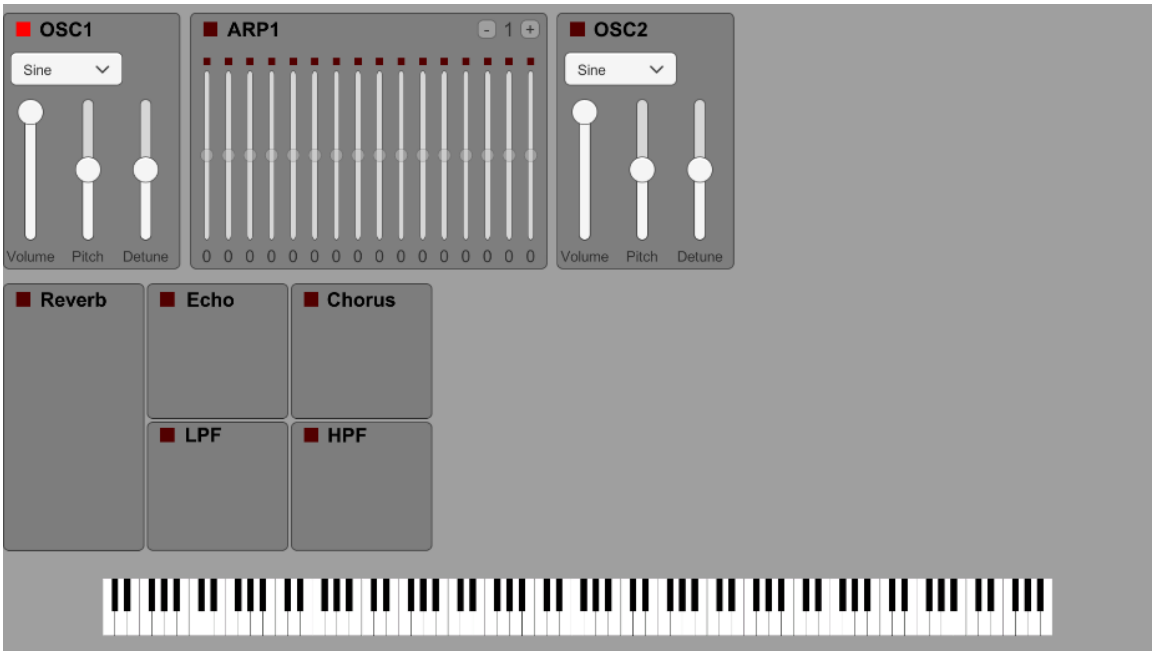


Kuva 19. Efektien komponentit ja niiden säädettävät arvot Unity-pelimoottorissa.

4.4 Graafisen ulkoasun suunnittelu

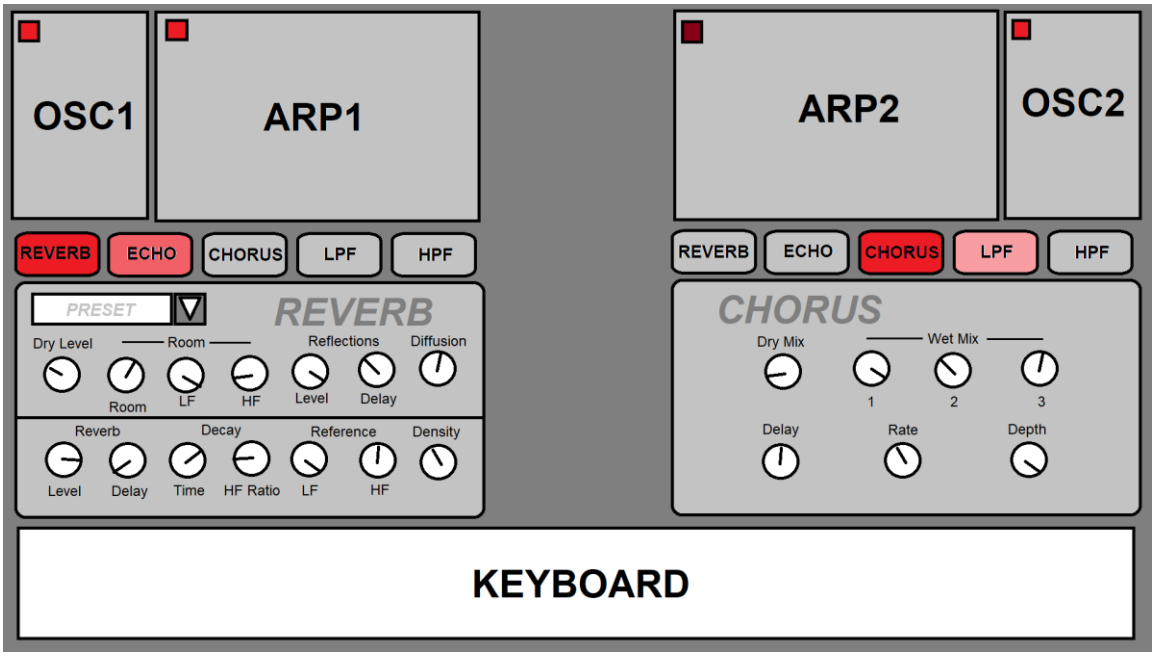
Graafista ulkoasua suunnitellessa käytiin läpi usean valmistajan virtuaali-instrumenttien ulkoasuja. Koska virtuaali-instrumentti on vielä kehitysvaiheessa, mitään lopullista ulkoasua ei voida lyödä vielä lukkoon. Nykyisellään virtuaali-instrumentti käyttää suurimmaksi osaksi Unity-pelimoottorilla tehtyjä väliaikaisia eli placeholder-grafiikoita, jotka tullaan korvaamaan tulevaisuudessa oikeilla vektorigrafiikoilla, joiden kokoa voidaan muokata menettämättä alkuperäistä kuvanlaatua. Myös virtuaali-instrumentin lopulliseen versioon sen värejä tullaan todennäköisesti vielä muuttamaan nykyisistä harmaan sävyistä.

Nykyinen ulkoasu sisältää kaksi oskillaattoria, arpeggiaattori ja toisen oskillaattorin efektit. Oskillaattoreissa on virtapainike ja kolme liukusäädintä, jotka ovat äänenvoimakkuudelle, oktaavikorkeudelle ja nuotin vireen vaihtamiselle. Arpeggiaattorissa on virtapainike ja 16 liukusäädintä nuoteille ja nuottien määrän lisäykseen ja poistamisen tarkoitetut painikkeet. Efekteillä ei ole tällä hetkellä muuta kuin virtapainikeet. Kaikkia säätimiä ei ole lisätty, koska virtuaali-instrumentista ei vielä ole tarkoitus tehdä Unity-pelimoottorin ulkopuolella käytettäviä itsenäisiä versioita, joten arvojen säätäminen tapahtuu tällä hetkellä Unity-pelimoottorin Editor-ikkunassa. Lisäksi virtuaali-instrumentin alaosassa on pianon koskettimistoa esittävä kuva, johon ilmestyy punainen piste sen koskettimen tai nuotin kohdalle, joka kuuluu tällä hetkellä oskillaattorista. Kuvassa 20 on nähtävissä tämänhetkinen virtuaali-instrumentin keskeneräinen ulkoasu.



Kuva 20. Virtuaali-instrumentin nykyinen ulkoasu Unity-pelimoottorissa.

Vaikka lopullista virtuaali-instrumentin ulkoasua ei ole päätetty vielä, tullaan nykyistä ulkoasua päivittämään sitä mukaa, kun uusia ominaisuuksia saadaan lisättyä. Ulkoasun seuraavaa versiota varten kaikki liukusäätimet tullaan korvaamaan pyöritettävillä nuppisäätimillä, mutta ne vaativat erillistä ohjelmointia, koska Unity-pelimoottorissa ei ole valmiita, pyöriteltäviä grafiikkaobjekteja. Lisäksi efektille tullaan lisäämään niiden arvoja muuttavat säätimet, joita ei ole vielä nykyisessä ulkoasussa. Sen sijaan, että kaikki efektit ovat näkyvillä, tullaan ne tekemään niin, että oskillaattorilla on vain yksi efekti näkyvänä, vaikka useampi efekti olisikin aktiivisena. Alaosassa sijaitsevan koskettimiston kuvaa tullaan muokkaamaan niin, että painettu kosketin muuttaa väriään sen sijaan, että siihen ilmestyy punainen piste. (Kuva 21.)



Kuva 21. Virtuaali-instrumentin ulkoasun seuraavan version luonnostelukuva.

4.5 Jatkokehitys

Opinnäytetyötä varten saatiin valmiiksi syntetisaattorityyppinen virtuaali-instrumentti, jossa on monofonisia eli yksiäänisiä oskillaattoreita, joissa on toimivat arpeggiaattorit ja mahdollisuus lisätä efektejä. Virtuaali-instrumentin nykyisien ominaisuuksien lisäksi siihen aiotaan tehdä uusia ominaisuuksia jatkossa. Myös pieniä bugeja eli ohjelmointivirheitä pyritään korjaamaan, joista yhtenä esimerkkinä on arpeggiaattorin virheellinen toiminta, kun sen toiminta-aika on asetettu todella pieneksi.

Yksi virtuaali-instrumentin uusista ominaisuuksista tulee olemaan polyfoninen-tila, jossa soivia nuotteja voi olla yhtä aikaa enemmän kuin yksi. Tämä mahdollistaisi soittamisen kaikilla sormilla samaan aikaan yhden sormen sijaan. Tämä tullaan toteuttamaan todennäköisimmin käyttämällä suurempaa määrää oskillaattoreita, jotka varattaan sitä mukaa nuottien käyttöön, kun koskettimia painettaan ja vapautetaan, kun koskettimen tuottama ääni on hiljentynyt kokonaan. Toinen uusi ominaisuus tulee olemaan samplejen lisäys virtuaali-instrumenttiin, jolloin käyttäjä voi asettaa jokaiselle koskettimelle tietyn samplen, joka toistetaan, kun kyseistä kosketinta painettaan. Sampleille tullaan tekemään muokkaustyökalu, joka mahdollistaa samplen sävelkorkeuden muut-

tamisen, jolloin käyttäjä voi käyttää samaa samplea useammalle koskettimelle halutessaan. Lisäksi mahdollisuus muuttaa virtuaali-instrumentin säätimien arvoja MIDI-koskettimien säätimillä on yksi mahdollinen lisäominaisuus.

Kun tulevaisuudessa virtuaali-instrumentin kaikki toiminnot ja graafinen ulkoasu ovat valmiita, virtuaali-instrumentti ja sen luomiseen käytetyt tiedostot tullaan laittamaan julkisiksi GitHub-sivustolle. Samaiselta sivustolta voi tällöin ladata toimivan version virtuaali-instrumentista sekä projektitiedostot Unity-pelimoottoriin, jolloin kuka tahansa voi muokata virtuaali-instrumenttia halutessaan. Projektitiedostot täytyy kuitenkin lisensoida niin, ettei kukaan voi julkaista virtuaali-instrumenttia tai sen lähdekoodia omanaan tai rahallisiin tarkoituksiin. Mikäli projektin ulkopuolinen kehittäjä haluaisi korjata virtuaali-instrumentin ohjelmointiskriptejä niin, että siitä olisi hyötyä virtuaali-instrumentin toiminnan osalta, korjaus tarkistettaisiin. Jos korjaus olisi aiheellinen, se lisättäisiin projektitiedostoihin ja korjauksen tekijän nimi mainittaisiin virtuaali-instrumentin tekijätiedoissa, mutta korjauksen tekijä ei saisi minkäänlaista käyttöoikeutta virtuaali-instrumentin GitHub-sivulle tai oikeutta julkaista virtuaali-instrumenttia omissa nimissään.

5 Pohdinta

Opinnäytetyötä varten kehitettyä virtuaali-instrumenttia tehtäessä nousi esiin muutamia pienempiä ongelmia, joista virtuaali-instrumentin toiminnalle kriittisimmät saatiin lopulta ratkaistua. Suurin ongelma liittyi kuitenkin virtuaali-instrumentin kehityksen alkuvaiheeseen. Kehitysprosessin alussa oli ajatus tehdä yksinkertainen tuki Unity-pelimoottoriin MIDI-laitteille, jossa MIDI-viesteistä olisi saatu vain halutut tiedot, kuten painetun koskettimen numero ja voimakkuus. Lyhyen tutkimisen jälkeen kuitenkin todettiin, että MIDI-laitetuen tekeminen veisi liikaa aikaa ja vaatisi enemmän osaamista, jolloin se olisi voinut olla jo opinnäytetyön aihe itsessään. Tähän ongelmaan kuitenkin löytyi ratkaisu, joka oli kolmannen osapuolen tekemän liitännäisen käyttäminen, jolloin MIDI-laitteita voidaan käyttää Unity-pelimoottorissa. Liitännäisen avulla saatiin käyttöön tarvittavat tiedot MIDI-koskettimilta tulevista MIDI-viesteistä, mutta myös paljon ylimääräistä tietoa, jota voidaan hyödyntää jatkokehityksen aikana uusiin ominaisuuksia tehdessä.

Muita pienempiä ongelmia ilmeni tehdessä aaltomuotoja, koska jotkin aaltomuodoista eivät tuottaneet haluttua muotoa. Tämä saatiin osittain korjattua, kun vääränlaisen aaltomuodon tuottavaa ohjelmointikoodia muutettiin hieman. Kolmioaaltomuoto ei ole edelleenkään aivan oikean kuuloinen, johon tullaan keskittymään jatkokehityksen aikana, kun aletaan korjaamaan pieniä ohjelmointivirheitä. Toinen pienempi ongelma oli tehdessä pyöritettäviä säätimiä, jotka toimivat hieman eri tavalla, kun käytetään Unity-pelimoottorin uutta syötejärjestelmää. Näitä säätimiä ei saatu tarpeeksi toimiviksi niiden ohjelmointiin varatun ajan puitteissa, joten niiden lopullinen ohjelmointi ja hiominen jätettiin myös myöhempään kehitysvaiheeseen, jolloin opinnäytetyön valmistumisen asettamia aikarajoitteita ei ole.

Nykyinen virtuaali-instrumentin versio osoittaa, että sen jatkokehitys olisi kannattavaa, koska siinä on potentiaalia olla hyödyllinen työkalu monipuolisempana kokonaisuutena. Yksittäisenä komponenttina oskillaattori on tehty tarpeeksi modulaariseksi, jolloin sitä voisi käyttää uudelleen muissakin projekteissa. Esimerkiksi peliprojektissa voisi ohjata ohjelmointikoodilla useampaa oskillaattoria samaan aikaan, jolloin pelin musiikki syntyisi reaaliajassa itse pelissä eikä se käyttäisi valmista äänitiedostoa kuten peleihin musiikki nykyään yleensä lisätään.

6 Yhteenveto

Opinnäytetyön tavoitteena oli kehittää MIDI-koskettimilla soitettava virtuaali-instrumentti Unity-pelimoottorilla, joka hyödyntää Unity-pelimoottorin ominaisuuksia. Virtuaali-instrumentin kehitys aloitettiin lisäämällä Unity-pelimoottoriin tuki MIDI-laitteille kolmannen osapuolen liitännäisellä, jolla saatiin aikaiseksi kommunikaatio MIDI-koskettimilta Unity-pelimoottoriin.

Tätä seurasi varsinaisen virtuaali-instrumentin ohjelmointi käyttäen MIDI-koskettimilta saatuja MIDI-viestejä, jolloin näistä viesteistä saadulla tiedolla yhdistettiin oikea taajuus painettuun koskettimeen. Taajuudella saatiin viestitettyä oskillaattorille miltä korkeudelta ääni toistetaan. Oskillaattoriin lisättiin muutama aaltomuoto, jotka muuttavat ääntä, joka lähetetään OnAudioFilterRead()-metodilla Unity-pelimoottorin Audio Source -komponenttiin.

Opinnäytetyön käytännön osuuden loppuvaiheessa virtuaali-instrumenttiin lisäominaisuudeksi tehtiin arpeggiaattori, jolla saatiin oskillaattorin yksittäinen ääni muutettua useamman äänen sarjaksi. Lisäksi oskillaattorille lisättiin Unity-pelimoottorista löytyvät efektit. Ulkonäöllisesti virtuaali-instrumentti tehtiin alkeellinen ulkoasu käyttäen Unity-pelimoottorin omia työkaluja, kuten säätimiä ja painikkeita.

Kehitetyn virtuaali-instrumentin nykyinen versio antaa hyvän pohjan jatkokehitykselle, jolloin siihen tullaan lisäämään monia uusia ominaisuuksia, kuten moniäänisen tilan ja mahdollisuuden käyttää omia sampleja oskillaattorin tuottaman äänen lisäksi sekä korjaamaan pieniä ongelmia. Myös ulkoasua päivitetään virtuaali-instrumentin muuttuessa sitä mukaa, kun uusia ominaisuuksia lisätään.

Lähteet

1 Heckroth J. Tutorial on MIDI and Music Synthesis. Saatavilla: https://www.midi.org/specifications/developer-white-papers/tutorial-on-midi-and-music-synthesis-2. Viitattu 9.2.2021.

2 Chadabe J. The Electronic Century Part IV: The Seeds of the Future. Saatavilla: https://web.archive.org/web/20120928230435/http://www.emusician.com/gear/0769/the-electronic-century-part-iv-the-seeds-of-the-future/145415. Viitattu 9.2.2021.

3 Vintage Synth Explorer. Roland Jupiter-6. Saatavilla: http://www.vintagesynth.com/roland/jup6.php. Viitattu 19.5.2021.

4 Vintage Synth Explorer. Sequential Circuits Prophet 600. Saatavilla: http://www.vintagesynth.com/sci/p600.php. Viitattu 19.5.2021.

5 Collins K. Game sound: An introduction to the history, theory, and practice of video game music and sound design. Cambridge: MIT Press. 2008.

6 The MIDI Association. General MIDI 1 System Level 1. Saatavilla: https://www.midi.org/specifications/midi1-specifications/general-midi-specifications/general-midi-1. Viitattu 16.3.2021.

7 The MIDI Association. Details about MIDI 2.0, MIDI-CI, Profiles and Property Exchange. Saatavilla: https://www.midi.org/midi-articles/details-about-midi-2-0-midi-ci-profiles-and-property-exchange. Viitattu 15.3.2021.

8 Buchla. Lightning II Description. Saatavilla: https://web.archive.org/web/20060824224042/http://www.buchla.com/lightning/descript.html. Viitattu 10.2.2021.

9 Matthies A. Ultimate Guide to MIDI Guitar: Gear, Apps & Plugins. Saatavilla: https://guitargearfinder.com/guides/how-to/ultimate-guide-midi-guitar-gear-apps-plugins/. Viitattu 27.4.2021.

10 Jam Origin. MIDI Guitar 2. Saatavilla: https://www.jamorigin.com/. Viitattu 27.4.2021.

11 Sweetwater. MIDI Controller Buying Guide. Saatavilla: https://www.sweetwater.com/in-sync/midi-controller-buying-guide/. Viitattu 28.4.2021.

12 Thomann GmbH. Kontrollerit. Saatavilla: https://www.thomann.de/fi/kontrollerit1.html. Viitattu 30.4.2021.

13 Shambro J. VST Plug-Ins: What They Are and How to Use Them. Saatavilla: https://www.liveabout.com/what-are-vst-plug-ins-1817748. Viitattu 25.3.2021.

14 Sweetwater. Which Plug-in Format Do I Need For My DAW?. Saatavilla: https://www.sweetwater.com/sweetcare/articles/which-plug-format-need-for-my-daw/. Viitattu 29.4.2021.

15 Native Instruments. Komplete. Saatavilla: https://www.native-instruments.com/en/catalog/komplete/. Viitattu 25.3.2021.

16 Yelton G. MIDI Learn Explained. Saatavilla: https://en.audiofanzine.com/midi-controller/editorial/articles/midi-learn-explained.html. Viitattu 28.4.2021.

17 Sound Training College. Synthesis: A Basic Understanding. Saatavilla: https://soundtraining.com/synthesis-a-basic-understanding/. Viitattu 22.4.2021.

18 Kipersztok J. Synth VST Essentials: Oscillators, Envelopes, and Filters. Saatavilla: https://www.izotope.com/en/learn/soft-synth-essentials-osillators-envelopes-and-filters.html. Viitattu 29.4.2021.

19 Brown G. Making a Custom Sampler Instrument. Saatavilla: https://www.izotope.com/en/learn/making-a-custom-sampler-instrument.html. Viitattu 22.4.2021.

20 Albano J. Understanding The Architecture Of Sample-Based Virtual Instruments. Saatavilla: https://ask.audio/articles/understanding-the-architecture-of-sample-based-virtual-instruments. Viitattu 30.4.2021.

21 Takahashi K. Minis: MIDI Input for New Input System. Saatavilla: https://github.com/keijiro/Minis. Viitattu 17.3.2021.

Liitteet

M-Audio Axiom 49

- Eight fully programmable endless rotary encoders.
- Eight fully programmable sample trigger pads.
- Nine fully assignable 40mm sliders
- USB 1.1 port to connect the Axiom to a computer. The keyboard can also be powered from the USB port, so no additional power supply is required.
- MIDI IN and MIDI OUT ports for connecting external MIDI gear.
- Expression pedal socket (expression pedal not included).
- Sustain pedal socket (sustain pedal not included).
- Fully programmable monophonic aftertouch.
- Null mode – for full parameter recall with each preset.
- Controller Mute – reposition controllers without affecting your software.
- Easy to program controls can be assigned to a vast range of MIDI messages including MIDI controller messages, GM/GS/XG SysEx messages, NRPN/RPN messages, channel aftertouch, program/bank changes, note messages and more.
- Rotary encoders can be programmed to use any of six popular increment/decrement methods for compatibility with virtually all software with encoder support. The encoders can also function as standard MIDI controllers with a range of 0 to 127.
- A range of different acceleration curves for the encoders for realistic dial control.
- Trigger pads respond to velocity or pressure and can be programmed to either send MIDI note data or controller messages, allowing for full control over all your software samplers, for triggering loops and much more.

Arturia Minilab MkII

- 25 note velocity-sensitive slim keyboard
- 2 banks of 8 high quality velocity & pressure sensitive pads with RGB backlighting
- 16 rotary encoders (2 of them are clickable)
- 2 capacitive touch sensors for pitch bend and modulation wheel
- 8 user presets
- Sustain pedal jack
- Octave up and octave down buttons for full range
- USB powered
- USB/MIDI class compliant no drivers needed.
- Mac or PC
- Kensington Security Slot