Bachelor's thesis

Information and Communications Technology

2022

Shane Wirkes

Environmentally Reactive Sound Design

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

Bachelor's Thesis | Abstract

Turku University of Applied Sciences

Information and Communications Technology

Completion year of the thesis: 2022 | number of pages 29

Shane Wirkes

Environmentally Reactive Sound Production

A prototype for an interactive music player was developed as a proof of concept for a startup, *(furniture music)*. Using the Raspberry Pi microcomputer and a collection of hardware extensions developed by the author, a system was created to play music that changes in reaction to the environment. In this case, light sensors controlled the volume levels of two different tracks playing simultaneously, while a motion sensor triggered a random, short piano recording from a list of many. The device was used as a visual/audio aid in pitching the startup idea during a startup accelerator program. As a result of the pitches, (furniture music) was asked to install the device at the Shift Business Festival in Turku in the summer of 2021. In addition to detailing the technical and artistic work required for the system, a summary of the public reaction to the device and areas for further development is included.

Keywords:

Microcomputers, interactive audio, generative music, environmental sensors, prototype, start-up

Content

Figures

Pictures

Functions

List of abbreviations

Abbreviation	Explanation of abbreviation [Source]
ADC	Analog-Digital Converter
DAW	Digital Audio Workstation
DSP	Digital Signal Processing
GPIO	General Purpose Input-Output
HAT	Hardware Attached on Top
IDE	Integrated Development Environment
LDR	Light Dependent Resistor
OSC	Open Sound Control
PIR	Passive Infrared
SBC-PIR	Motion sensor module by JOY-IT [1]
scide	SuperCollider’s integrated development environment
sclang	SuperCollider’s programming langage
scsynth	SuperCollider’s real-time audio server
SoC	System on a Chip
USB	Universal Serial Bus

1 Introduction

The goal of this thesis is to create a system for music and sound production that responds to the environment in which it is placed. This project was developed for *(furniture music)* as a proof-of-concept in tandem with Boost Turku's startup accelerator, Startup Journey. (furniture music) is a project conceived of by the author. In addition to using this device on stage during pitching competitions, (furniture music) was also invited to implement the device in a "relax room" at the Shift Business Festival in Turku, Finland, August 2021. The room's purpose was to provide a quiet, relaxing environment to juxtapose the frenetic, high-paced energy of the ongoing startup festival.

Throughout this project, inspiration and instruction were taken in the form of Eli Fieldsteel's SuperCollider Tutorial playlist on YouTube [2]. Fieldsteel is an assistant professor and director of the Experimental Music Studios at The University of Illinois at Urbana-Champaign. He currently uses YouTube as a platform for hosting online lectures on SuperCollider.

To this end, a set of hardware and software requirements were gathered to develop a system for mapping sensor data to musical parameters and composing a framework within which to play coherent audio based on the sensor data.

Many of the technological elements used in this project were found in a pre-existing and reliable form. From the programming language, sclang, to the python library for the motion sensors, there are a plethora of resources and writings to reference. The stitching together of these disparate parts to fit the project specification was where the bulk of the development took place.

A compositional technique pioneered by Brian Eno was used to structure the musical elements into a logical form, ensuring aural coherence regardless of the activities in the environment. Eno created tape loops of widely varying lengths with heavy use of silence. Playing multiple loops simultaneously, results in new musical combinations as the loops are playing asynchronously [3].

To adapt this idea to environmentally reactive sound design, a 7-minute loop of a guitar drone was paired with a 9-minute recording of birds singing. This is similar to the musical "polyrhythm" where two or more rhythms of different length, *(m, n),* repeat simultaneously, resulting in *mxn* different combinations [4]. This is an efficient way to create new sounds out of a limited amount of material.

Simultaneously, when the motion sensor detects motion, a random recording from a library of piano melodies is played. The volumes of the guitar drone and the bird samples are set with the data from the light-dependent resistor array. Essentially, when there is more light, the bird sounds are louder and the guitar sounds are quieter, and when there is less light, the bird sounds are quieter and the guitar sounds are louder.

This thesis lays out the process of designing and implementing the device. In chapter 2, the hardware and software requirements are discussed as well as the overall architecture and the physical enclosure. Chapter 3 delves into the implementation of

the software and hardware. Chapter 4 gives an account of the event in which the device was deployed, discussing successes, criticisms. In the conclusion, chapter 5, a summation is given as well as a brief discussion of areas for future development.

2 Device Specifications

The first step in determining the viability of an environmentally reactive sound system was creating a device specification. This included deciding what hardware and software to use as well as how to configure a system for mapping sensor data to musical parameters, and finally, composing a framework within which to play coherent audio based on the sensor data. While questions of productization and market have been considered, this work focuses on a prototypical proof-of-concept.

2.1 Hardware

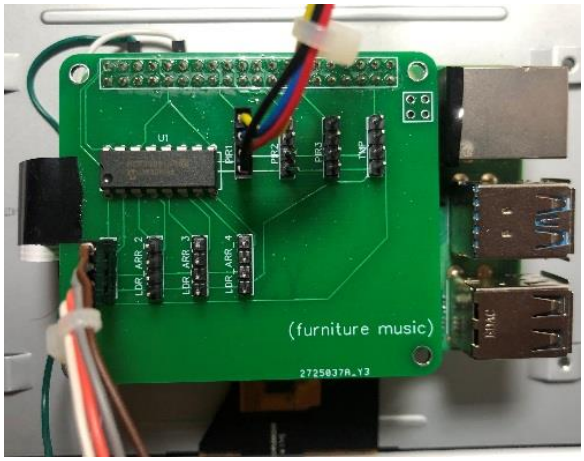
The system would need to be composed of several modular pieces of hardware that could communicate in a specific way. Although in most cases, the best decision was to use an off-the-shelf product, there were some instances where bespoke circuit design was more efficient.

2.1.1 Raspberry Pi 4

The Raspberry Pi 4 is a powerful microcomputer capable of processing sound synthesis via SuperCollider. The computer features a powerful quad-core Cortex-A72 (ARM v8) 64-bit SoC with a 1.5GHz clock [5].
Although there are specifically audio-oriented microcomputers available, the Raspberry Pi was chosen largely for its familiarity and availability to the author. A second, equally important factor was the robust documentation and active community revolving around SuperCollider and its compatibility with Raspberry Pis.

2.1.2 Raspberry Pi HAT

A circuit board or HAT (Hardware Attached on Top) was developed for this project to simplify the connection of sensors to the Raspberry Pi (Picture 1). A HAT is "an add-on board for [Raspberry Pis] that conforms to a specific set of rules [6]," or hardware specifications to be more precise. This allows the user to plug the hat into the Raspberry Pi and start working with it immediately.



Picture 1. PCB implementation.

The HAT was designed using EasyEDA (Figure 1), a free, web-based EDA program. It was fabricated by the company's partner, LCSC Electronics.

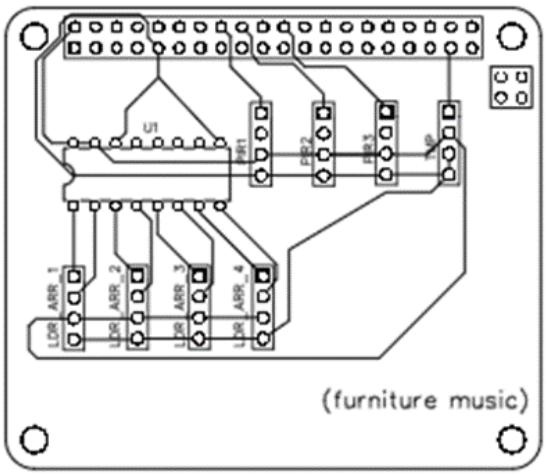


Figure 1. Hat installed on Raspberry Pi 4.

This hat consists of an MCP3008 [7] analog-digital converter (ADC) and several inputs for electronic sensors. There are four inputs for light-dependent resistor (LDR) arrays, three inputs for SBC-PIR motion sensors, and one input for a DHT-11 temperature and humidity sensor [8] (unused in this project). The MCP3008 ADC converts the analogue voltage data from the LDR array into discrete, digital information that can be read by the Raspberry Pi 4's general-purpose input/output (GPIO) pins [9]. The SBC-PIR motion sensors as well as the DHT-11 temperature and humidity sensor are each connected to a single GPIO as they already output digital information. Finally, the HAT routes each sensor's voltage and ground pins to the appropriate pins on the Raspberry Pi 4 (Figure 2).

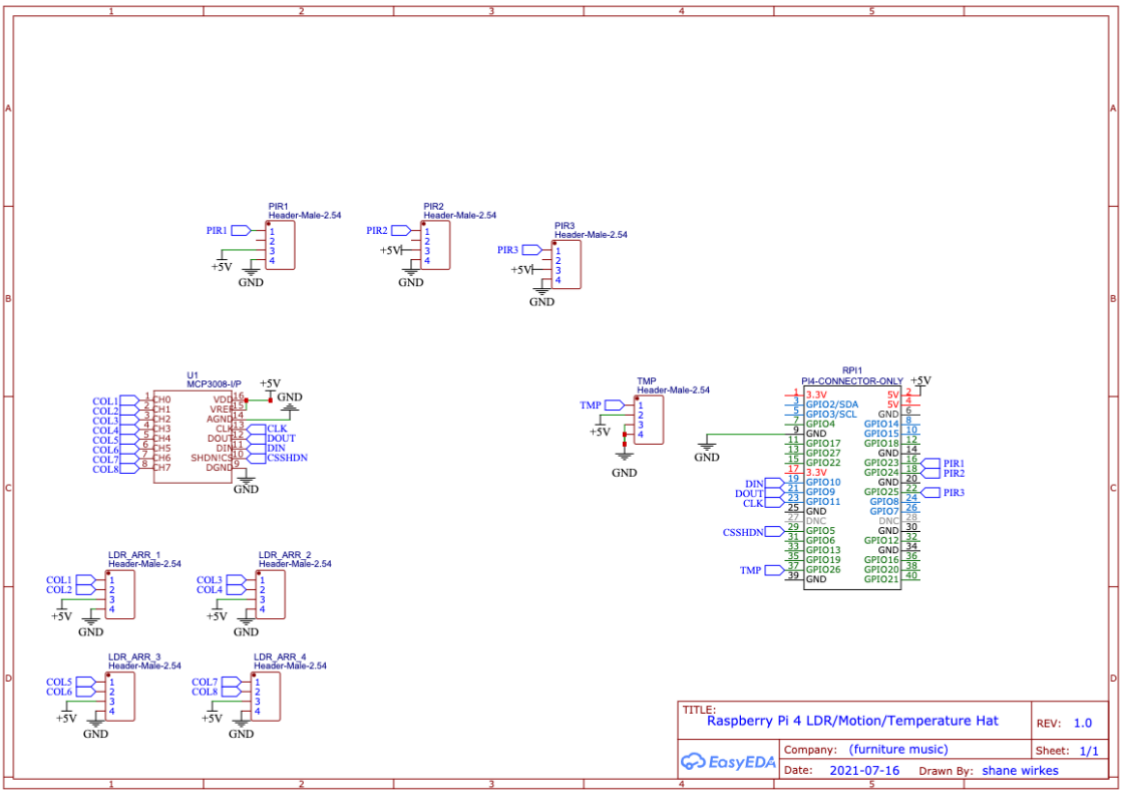


Figure 2. HAT Schematic.

2.1.3 Light-Dependent Resistor (LDR) Array

An array of light-dependent resistors (LDRs) was implemented based on the design of Dusan Grujic [10]. For this device, a small panel was designed on a rectangular grid with two arrays of eight LDRs. Although it looks like a matrix, the top two rows of four LDRs are wired in series. The bottom two rows are wired similarly (Figure 3).
Each LDR has a 10kohm resistor wired in parallel with a node between them to pass the light-controlled voltage. The first array of eight LDRs sends its voltage to the first channel of the ADC where it is converted to a digital signal that can be read by SuperCollider through the Raspberry Pi's GPIO.

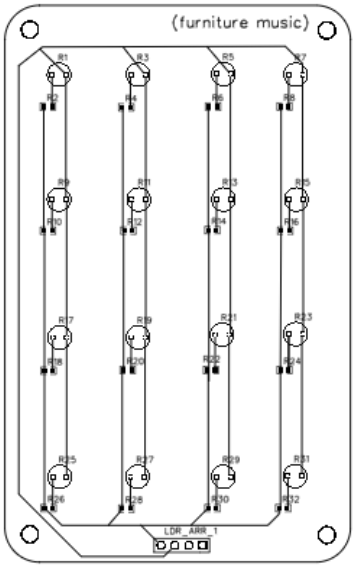


Figure 3. LDR Array PCB Layout.

The same setup is used for the second array of eight LDRs, although the voltage information is sent to channel 2 of the ADC (Figure 4).

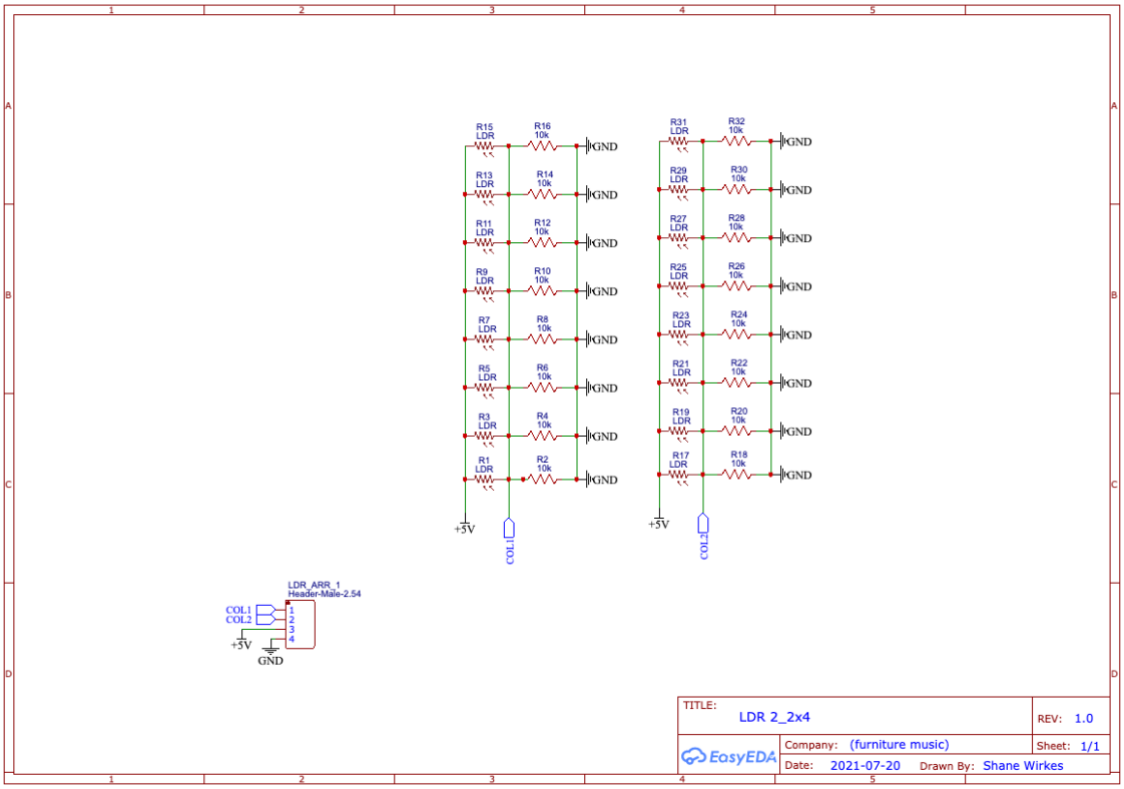


Figure 4. LDR Array Schematic.

2.1.4 SBC-PIR Motion Sensor

The SBC-PIR sensor (Picture 2) is a "module with a pyroelectric infrared sensor which detects movement in the detection area of the Fresnel lens and then outputs a 'HIGH' signal at the sensor output pin 'SIG' for 3 seconds. Afterwards [sic] the signal changes back to the 'LOW' state [11]." It is used in this project to detect observers moving around the device and trigger short, melodic piano samples.

Picture 2. Motion Sensor.

2.1.5 Komplete Audio 6 Audio Interface

The Komplete Audio 6 [8] USB audio interface is made by Native Instruments. Relevant to this project, it provides 24-bit analog-digital / digital-to-analog converters and four analog outputs. Additionally, it provides up to a 96kHz sampling rate as well as Audio Stream Input/Output (ASIO) and Core Audio low-latency drivers [12]. The 24-bit digital-to-analog converter in concert with a high sampling rate and low-latency drivers results in a complete, out-of-the-box solution for sending audio data from the Raspberry Pi to the speaker (Figure 5).

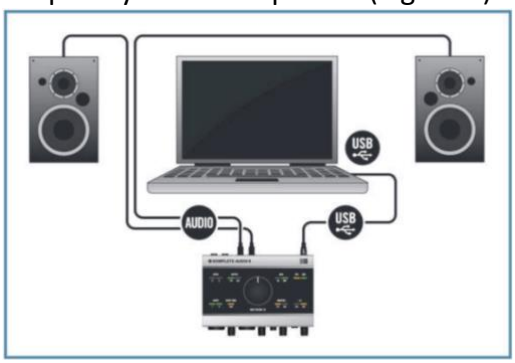


Figure 5. Typical Application for Komplete Audio 6.

2.1.6 Genelec 8030A Monitor

The Genelec 8030A [9] is a compact bi-amplified monitoring system with "performance comparable to much larger systems [13]." This loudspeaker's accurate sound imaging and compact size make it perfect for implementation in this project (Picture 3).

Picture 3. Genelec 8030A.

2.2 Software

There are myriad solutions for real-time audio processing, however, very few of them will run on the computationally-limited Raspberry Pi 4. Additionally, the audio processor need a way to accept voltage signals from the environmental sensors. With these requirements in mind, SuperCollider was chosen as the audio processing language and Python was used to communicate the analog signals from the sensors into the audio processor.

2.2.1 SuperCollider

> "SuperCollider is a platform for audio synthesis and algorithmic composition, used by musicians, artists, and researchers working with sound. It is free and open source [sic] software available for Windows, macOS, and Linux [14]."

SuperCollider consists of two main parts inside a single application:
scsynth is a real-time audio server. This is the element of SuperCollider that is responsible for the digital signal processing (DSP) and resultant sounds produced by the platform.
sclang is an interpreted programming language focused on audio production. It controls scsynth via Open Sound Control [11]. Open Sound Control (OSC) is a data transport specification (an encoding) for realtime [sic] message communication among applications and hardware. Its extreme accuracy in the time domain makes it essential for audio production over a server (Figure 6). scide is the editor for the SuperCollider language which also contains an integrated help system.

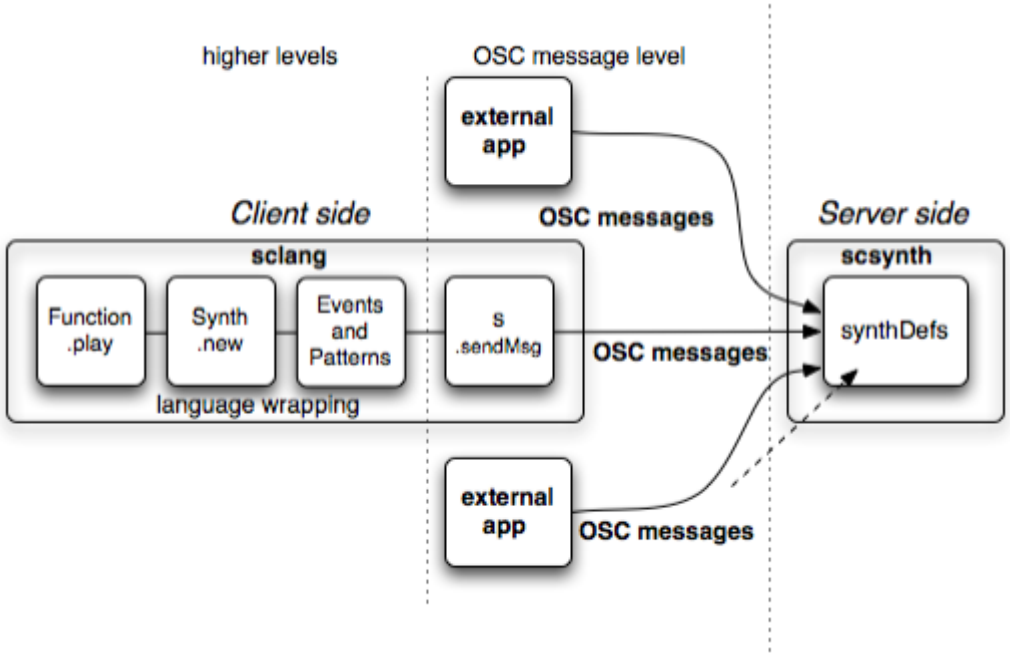


Figure 6. Sclang as a high-level client..

2.2.2 Python

"Python is a high-level general purpose [sic] programming language [16]." Because of Python's ability to send and receive OSC messages using the python-osc library and its integration into the Raspberry Pi OS, it is an ideal middle-ware between electronic sensors and SuperCollider's scsynth. There are Python libraries for controlling both the analog-digital converters and motion sensors used in this project.

2.2.3 Raspberry Pi OS

Raspberry Pi OS is the operating system on which the hardware device runs. The full graphical installation of this operating system was chosen because of the ease with which one can access scide, as well as the included Geany IDE to write and edit python files. The Raspberry Pi 4's quad-core SoC makes it easy to process audio on top of the graphic install of Raspberry Pi OS. The version used for this project is Raspbian 10 Buster based on DebianBuster.

2.2.4 Ableton Live

The sound recordings were produced and recorded using the digital audio workstation (DAW) software, Ableton Live [17].

2.3 Architecture

The essential function of this system is to convert analog and digital sensor signals into digital audio signals that can be multiplied with the preloaded audio signals (recordings) before being converted back into an audible analog signal.
More specifically, data from the environmental sensors is sent into SuperCollider through a Python script, initiating the playback of a recording or manipulating it in some way (Figure 7).

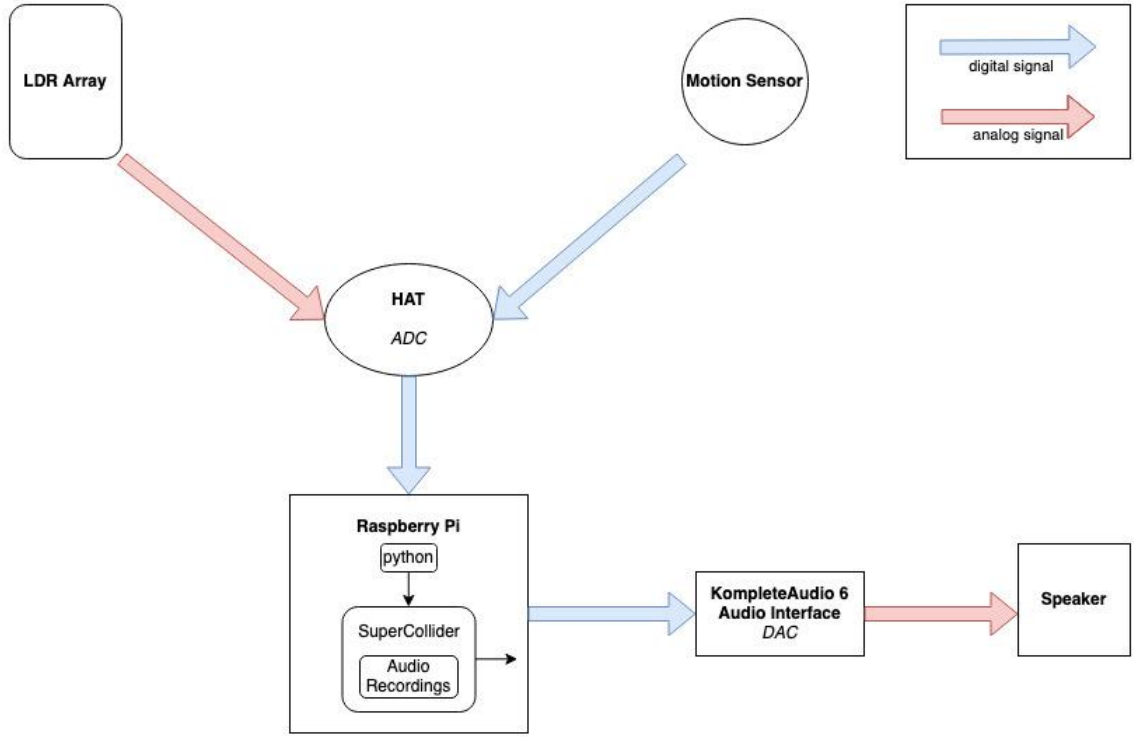
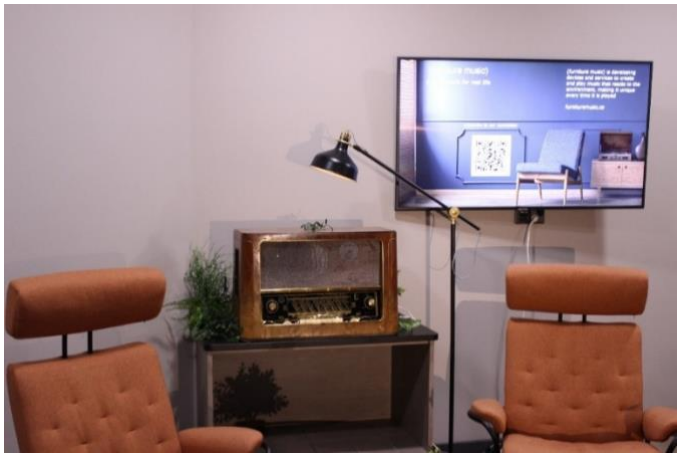


Figure 7. Analog/Digital Signal Flow.

2.4 Enclosure

In order to create a visually pleasing aesthetic in which to house the electronics, a vintage radio cabinet was selected (Picture 4). The existing electronics in the radio cabinet were removed (including the speakers) and the speaker cloth was replaced.



Picture 4. System set up at Shift Business Festival.

A small groove was chiseled out from the top of the cabinet in order to install the LDR array and feed the wire into the system (Picture 5).

Picture 5. LDR Array installed in cabinet.

3 Implementation

A folder of audio recordings were created to form the basis of the audio content. The audio recordings were created in Ableton Live with a mix of synthesizers, sampled instruments, and field recordings. The recordings were then triggered by SuperCollider to take advantage of the many different methods for manipulating and triggering audio recordings in a musically logical way.

3.1 Audio Recordings

The audio recordings, created by the author, are divided into three layers:

1. A bowed guitar drone
2. A recording of bird sounds
3. An array of ten recordings of short piano gestures, ranging in length from 5 to 15 seconds

3.2 Synth Definitions in SuperCollider

In SuperCollider, "a Synth represents a single sound producing unit [15]." Two different Synth definitions, or SynthDefs, were created to control the playback of the audio recordings. The first SynthDef (Function 1) is simply designed to load the audio file as a buffer and play it back on a single channel at a static amplitude.

Function 1. Synth definition \bufrd.

```
SynthDef.new(\bufrd, {
            arg amp=1, out=0, buf;
            var sig, ptr;
            ptr = Line.ar(0, BufFrames.kr(buf) - 1,
            BufDur.kr(buf), doneAction:2);
            sig = BufRd.ar(2, buf, ptr);
            sig = sig * amp;
            Out.ar(out, sig);
}).add;
```

The other SynthDef (Function 2) is designed to take two separate audio files as a buffer and play them back simultaneously in an asynchronous loop. Additionally, and most importantly, the SuperCollider method XFade2 has been implemented to allow inversely proportional volume control between the two sound files. This type of sound manipulation is known as crossfade.

Function 2. Synth definition \twobufrdphasor.

```
SynthDef.new(\twobufrdphasor, {
            arg amp=1, out=0, xfade=0, buf1, buf2, start1,
            start2, end1, end2, rate=1;
            var sig1, sig2, ptr1, ptr2;
            ptr1 = Phasor.ar(0, BufRateScale.kr(buf1)*rate, start1, end1);
            ptr2 = Phasor.ar(0, BufRateScale.kr(buf2)*rate, start2, end2);
            sig1 = BufRd.ar(2, buf1, ptr1);
            sig2 = BufRd.ar(2, buf2, ptr2);
            sig1 = sig1 * amp;
            sig2 = sig2 * amp;
            Out.ar(out, Xfade2.ar(sig1, sig2, xfade));
}).add;
```

3.3 Motion sensor to piano trigger

The SBC-PIR motion sensor is used to trigger a random piano sample from the array of ten samples.
In order to send messages from the motion sensor to SuperCollider, a Python script (Function 3) was written to wait for a motion sensor event and then transmit an Open Sound Control (OSC) message.
Function 3. Python motion sensor function.

```
Def send_motion_message():
    msg = osc_message_builder.OscMessageBuilder(address = ‘/motion’)
    msg.add_arg(1, arg_type=’’')
    msg = msg.build()
    client.send(msg)
    print('motion detected’)
    return
```

By listening for an OSC message with a pre-determined name, SuperCollider is able to receive a new message every time the python script sends one. Additionally, a ten-second wait is initiated after a sample is triggered to avoid excessive overlapping or distorting of the signal (Function 4).

Function 4. SuperCollider motion sensor function.

```
~motion_message = "motion_message";
~motion_OSC = OSCFunc({arg msg, time, addr, recvPort;
    ~motion_message = msg; },'/motion');
~control_motion = Routine.new({
    {
        if(~motion_message[1].booleanValue and:{ ~p.isPlaying.not })
        {
            ~p.play;
            10.wait;
        }
        {~p.stop};
        0.1.wait;
    }.loop
}).play;
```

3.4 Light data to crossfade

Voltage data from the LDR arrays are sent to SuperCollider in a similar fashion to the motion sensor (Function 5). However, the information is sent as a floating-point number rather than a boolean value. It is notable that the message is sending the raw ADC value from the light sensor, not the converted voltage message. The idea here is simply to avoid unnecessary conversions as will be seen next.

Function 5. Python LDR array function.

```
def send_light_message():
   msg = osc_message_builder.OscMessageBuilder(address = '/light')
   msg.add_arg(chan.value, arg_type='f')
   msg = msg.build()
   client.send(msg)
```

When SuperCollider receives these messages, it converts the raw ADC value to a decimal value between -1.0 and 1.0 using a linear scale. This decimal value is used to set the cross-fade between the bowed guitar sample and the recording of birds (Function 6). The effect is that when there is more light, the birds are louder and the guitar is quieter. When there is less light detected, the inverse happens (e.g. the birds are quieter and the guitar is louder). Because the light conditions in different spaces vary so widely, the minimum and maximum light values are evaluated and changed on the fly by the operator; a process for which SuperCollider is built.

Function 6. SuperCollider LDR array to crossfade function.

```
(
~control_light = Routine.new({
    {
        ~y.set(\xfade, ~message_light[1].linlin(25000, 55000, -1.0, 1.0));
        0.1.wait;
    }.loop
}).play;
)
```

4 Demonstration

The device was used throughout the summer of 2021 as a proof of concept for (furniture music) in the context of Hub Turku's Startup Journey. Startup Journey was a 2-month intensive startup accelerator with a major emphasis on pitching to investors and other potentially interested parties. The device worked very well as a visual and audio aid in the pitching presentations. In collaboration with the facilitators, the ambient light in the presentation space was raised and lowered at the author's direction in order to demonstrate in real-time the light sensitivity of the device. By walking in front of the device during the presentation, the motion detecting aspect of the system was featured. Positive feedback was received from the judges of the final pitch contest. Additionally, several people asked to be updated on when they could purchase such a device. In reaction to the pitch/demonstration, (furniture music) was invited to feature the device at Shift Business Festival in Turku.

The device was set up in a "relax room," designed for attendees to unwind in a quiet, tranquil space juxtaposed with the frenetic energy of the festival. In lieu of formal surveys or interviews, observation was the main way of procuring feedback, watching guests interact with the device and rest in the room. The reactions were again, generally positive. Many people sat in the room at length, closing their eyes, checking their email, or discussing quietly. It became apparent that the light sensitive aspect of the device did not function very meaningfully in an indoor space where the lighting was almost completely static. This aspect of the device had to be pointed out directly in most cases for the observers to realize it was happening. The most interesting results came when someone who had been sitting for a longer time finally got up to leave and triggered the motion sensors. Overall it was a very subtle effect, but it completely achieved the goal of creating a relaxing environment for festival goers.

5 Conclusion

The subtlety of the device is an overall strength. By not drawing direct attention to itself, the device is able to foster a more relaxing environment. However, there are opportunities to make the environmental reactivity more recognizable to the casual observer without being obtrusive. As mentioned previously, the light sensors had little effect in a room where ambient lighting never changed. Perhaps creating a larger array of light sensors and raising the sensitivity would allow for more subtle light changes, such as shadows, to register in the system. Another way to react meaningfully to the environment would be through the use of microphones. SuperCollider allows for the live monitoring and manipulation of audio signals, so the same system could be used to respond directly to sound already in the space, either complimenting it or attempting to soften it. The main focus for further development would be either replacing or enhancing the light sensing mechanism to measure much smaller changes in light and shadow and installing a microphone to monitor the audio landscape of the environment in which it is placed.

Another area for development is in the creation of sound content for the device. This prototype used only one set of sounds, but adding different combinations that could be controlled by the user would allow for a much richer experience and open up new possibilities of environmental reactivity. The addition of microphones into this system could also enable tuning the sound to the particular space.

The main goal of this thesis was to create a self-contained system that plays sound in response to environmental conditions. By developing a set of specifications to facilitate this goal, a prototype device was created and demonstrated publicly. The research involved along the way confirmed the possibility of creating the system while also revealing areas for further development that could result in a more immersive and memorable experience for the observer.

References

[1] SIMAC Electronics GmbH, "JOY-IT," 26 January 2021. [Online]. Available: https://joy-it.net/files/files/Produkte/SBC-PIR/SBC-PIR_Manual_20210521.pdf. [Accessed 1 February 2022].

[2] E. Fieldsteel, "YouTube," 2021. [Online]. Available: https://www.youtube.com/playlist?list=PLPYzvS8A_rTaNDweXe6PX4CXSGq4iEWYC. [Accessed 28 Jan. 2022].

[3] G. O'Brien, "Eno at the Edge of Rock," 1 June 2016. [Online]. Available: https://www.interviewmagazine.com/music/new-again-brian-eno. [Accessed 8 Jan. 2022].

[4] D. DeSantis, "Problems of Progressing: Asynchronous or Polyrhythmic Loops," in *74 Creative Strategies for Electronic Music Producers*, Berlin, Ableton AG, 2015, p. 151.

[5] Raspberry Pi Trading Ltd., "Raspberry Pi," 1 Jan. 2021. [Online]. Available: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf. [Accessed 8 Jan. 2022].

[6] J. Adams, "Introducing Raspberry Pi HATs," 31 July 2014. [Online]. Available: https://www.raspberrypi.com/news/introducing-raspberry-pi-hats/. [Accessed 7 Jan. 2022].

[7] Microchip Technology Inc., "2.7V 4-Channel/8-Channel 10-Bit A/D Converters," Dec. 2008. [Online]. Available: https://www.farnell.com/datasheets/808965.pdf. [Accessed 1 Feb. 2022].

[8] L. Fried, "DHT11, DHT22 and AM2302 Sensors," adafruit, 29 July 2012. [Online]. Available: https://learn.adafruit.com/dht. [Accessed 1 Feb. 2022].

[9] K. Rembor, "MCP3008 - 8-Channel 10-Bit ADC With SPI Interface," 25 Oct. 2018. [Online]. Available: https://learn.adafruit.com/mcp3008-spi-adc/python-circuitpython. [Accessed 7 Jan. 2022].

[10] D. Grujic, "100 x 100 Photo Resistor LED Matrix," June 1999. [Online]. Available: http://www.seattlerobotics.org/encoder/199906/dusan.html. [Accessed 22 June 2021].

[11] JOY-IT, "JOY-IT," 26 January 2021. [Online]. Available: https://joy-it.net/files/files/Produkte/SBC-PIR/SBC-PIR_Manual_20210521.pdf. [Accessed 8 January 2022].

[12] Native Instruments GmbH, "manualslib," 1 6 2016. [Online]. Available: https://www.manualslib.com/manual/1575095/Native-Instruments-Komplete-Audio-6.html?page=2#manual. [Accessed 8 Jan. 2022].

[13] Genelec OY, "Genelec," 1 Apr. 2007. [Online]. Available: https://assets.ctfassets.net/4zjnzn055a4v/xjY6MIRReKqGu48sSkgEe/390d8f322adbae6773953b1c121078b8/8030A_datasheet.pdf. [Accessed 8 Jan. 2022].

[14] SuperCollider, "SuperCollider Repository," 2018. [Online]. Available: https://github.com/supercollider/supercollider. [Accessed 8 Jan. 2022].

[15] SuperCollider Documentation, "SuperCollider Documentation," 14 May 2021. [Online]. Available: https://doc.sccode.org/Classes/Synth.html. [Accessed 8 Jan. 2022].

[16] python.org, "BeginnersGuide," 18 Sept. 2018. [Online]. Available: https://wiki.python.org/moin/BeginnersGuide/Overview. [Accessed 7 Jan. 2022].

[17] Ableton, "What is Live," Ableton AG, 18 June 2018. [Online]. Available: https://www.ableton.com/en/live/what-is-live/. [Accessed 1 Feb. 2022].

[18] S. 3. d. Contributors, "https://docs.sccode.org," 20 October 2021. [Online]. Available: https://doc.sccode.org/Classes/Synth.html.

[19] W. Pirkle, "Designing Audio Effect Plugins in C++," New York, Routledge, 2019, p. 1.

[20] D. C. N. C. Scott Wilson, "The SuperCollider Book," Boston, Massachusetts Institute of Technology, 2011.

[21] S. M. Paul Scherz, "Practical Electronics for Inventors," New York, McGraw Hill Education, 2016, p. 512.

[22] Debian Wiki, "ALSA," 7 Dec. 2021. [Online]. Available: https://wiki.debian.org/ALSA. [Accessed 7 Jan. 2022].

[23] Debian Wiki, "JACK," 19 Dec. 2019. [Online]. Available: https://wiki.debian.org/JACK. [Accessed 7 Jan. 2022].

[24] JOY-IT, "JOY-IT," 2021. [Online]. Available: https://joy-it.net/en/products/SBC-PIR. [Accessed 8 Jan. 2022].

[25] opensoundcontrol.org, "OSC is an Encoding," 23 Apr. 2021. [Online]. Available: https://opensoundcontrol.stanford.edu/. [Accessed 7 Jan. 2022].

[26] B. Croston, "RPi.GPIO 0.7.0," 21 July 2019. [Online]. Available: https://pypi.org/project/RPi.GPIO/. [Accessed 7 Jan. 2022].