Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorthesis

Tim Staats

Praxissemester bei portrix.net GmbH

Fakultät Technik und Informatik
Department Informations- und Elektrotechnik

Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

Tim Staats

Praxissemester bei portrix.net GmbH

Bachelorthesis based on the study regulations
for the Bachelor of Engineering degree programme
Informations- und Elektrotechnik
at the Department of Information and Electrical Engineering
of the Faculty of Engineering and Computer Science
of the Hamburg University of Applied Sciences

Supervising examiner : Prof. Dr. rer. nat. Henning Dierks
Second Examiner : Prof. Dr. Klaus Jünemann

Day of delivery 6. Mai 2019

Tim Staats

Title of the Bachelorthesis

Extending an Android App for a Multi-user Geodatabase of Historical Monuments

Keywords

Java, Android, Webapplication, Backendless, Geodata, Userservice, historical monuments, google map, Baas

Abstract

This thesis documents the development of an Android Application. The app offers the opportunity to registered users to capture and catalog historical Monuments. Furthermore the app displays the data within a google map and in form of a detailed list. The corresponding Backend manages the data and also display it within a webapplication. The necessary measures, concepts, problems and their solutions are described below.

Tim Staats

Titel der Arbeit

Weiterentwicklung einer Geodatenbasierten Multi-Nutzer Android App für historische Monumente

Stichworte

Java, Android, Webapplication, Backendless, Geodaten, Nutzerservice, historische Monumente, Google Map, BaaS

Kurzzusammenfassung

Diese Arbeit dokumentiert die Weiterentwicklung einer Android Applikation. Die App bietet registrierten Nutzern die Möglichkeit historische Monumente zu erfassen und zu katalogisieren. Des Weiteren visualisiert die App die katalogisierten Daten auf eine google Map und in Form von detailierten Listen. Im dazugehörigen Backend werden die Daten verwaltet und in einer Web Applikation visualisiert. Die dafür erforderlichen Maßnahmen, Konzepte, Probleme und deren Lösungen werden im folgenden beschrieben.

Contents

1. Introduction **6**

2. Theory **7**
- 2.1. Basics of Electric Guitars . . . 7
- 2.2. Basics of Audio Engineering . . . 9
- 2.3. Raspberry Pi and Audio-HAT . . . 15
- 2.4. Digital Signal Processing . . . 16
  - 2.4.1. Digital Signals . . . 17
  - 2.4.2. Digital Systems . . . 19
- 2.5. Guitar Effects . . . 21
  - 2.5.1. Delay . . . 22
  - 2.5.2. Distortion . . . 23

3. Requirements **27**
- 3.1. Hardware . . . 27
- 3.2. Software . . . 27

4. Design **29**
- 4.1. Hardware . . . 29
  - 4.1.1. Case . . . 29
  - 4.1.2. Preamp Module . . . 29
  - 4.1.3. User Interface Module . . . 31
  - 4.1.4. Power Supply . . . 32
- 4.2. Software . . . 33

5. Implementation **35**
- 5.1. Hardware Realisation . . . 35
  - 5.1.1. Preamp Module . . . 35
  - 5.1.2. User Interface Module . . . 42
  - 5.1.3. Construction of complete Unit . . . 45
- 5.2. Software Realisation . . . 48
  - 5.2.1. Audio Signal Processing . . . 49
  - 5.2.2. User Interaction . . . 54

6. Test **57**
6.1. Preamp Module Performance 57
6.2. Measurement of Guitar Effects 61
6.2.1. Delay 62
6.2.2. Distortion 64
6.3. Maximum Latency 68

7. Summary **70**

A. Appendix **71**
A.1. Attached CD 71
A.2. Total Schematic of Effect Unit 72
A.3. Delay Effect Measurements 73
A.4. Distortion Effect Measurements 83

1. Introduction

Since its market introduction in 2012, the Raspberry Pi has become one of the most distributed single-board computers for didactic usage. Due to many application possibilities, the Pi found its way into many projects in University. The development of an Audio-HAT, providing professional analog audio in- and outputs by Sebastian Albers [**?**], extends the Pi with the capability of digital signal processing.

Digital effect units for electric guitars are manipulating the audio signal to improve the users sound experience. These units are placed in the middle of the signal chain between the guitar output and amplifier input. A Pi in combination with the Audio-HAT can fulfil the task of digital signal processing. Due to the several interfaces, the audio processing can be controlled by the user without having a mouse or keyboard attached to the Pi.

The goal of this thesis is to develop a fully working multi-effects unit based on the Pi working as an embedded system. In addition to that, the unit should consist of an appropriate preamp module to adjust the guitar signal and a user interface module for the system controls.

After the theoretical part and the declaration of requirements, the design and implementation are described in a detailed way. Finally, the device is tested with regard to the requirements.

2. Theory

2.1. Basics of Electric Guitars

Guitar Components

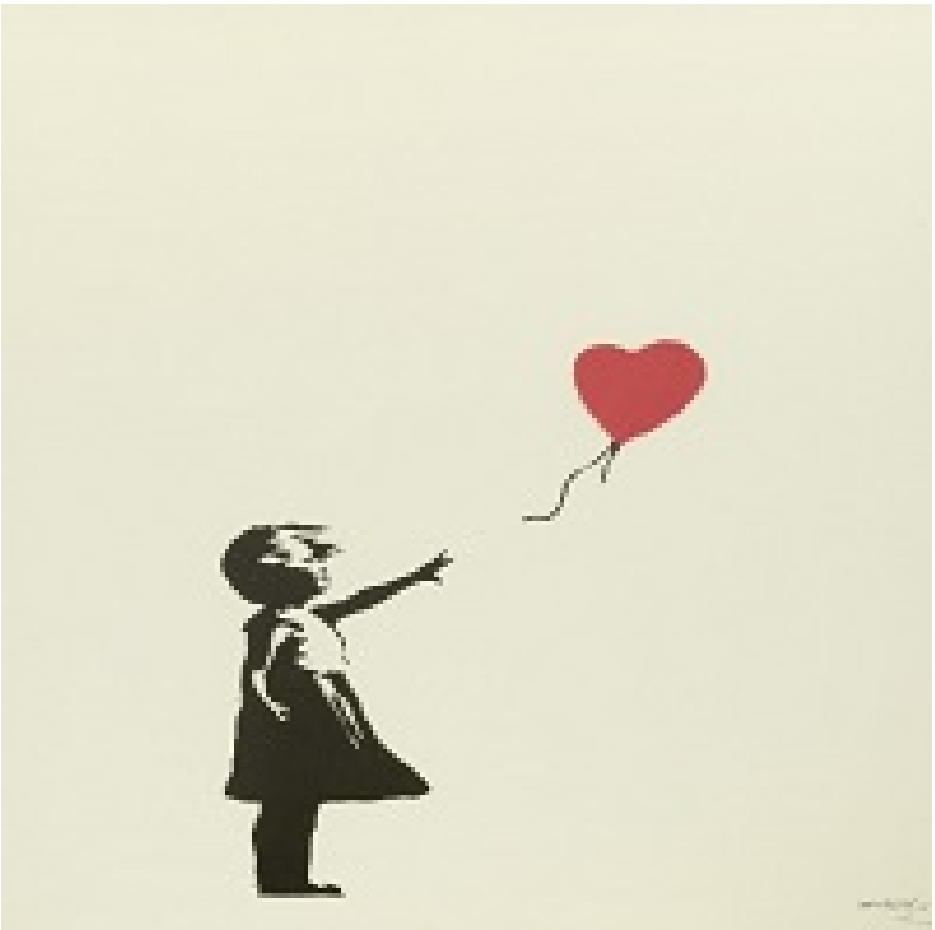

Figure 2.1.: Components of an electric guitar[1]

A regular electric guitar consists of six strings and two or three pickups (2.1). Acoustic guitars have a hollow soundboard in which the vibration of the strings resonates. As a consequence, the sound is transmitted to the air. Regular electric guitars have a solid-body and depend on electromagnetic pickups which are turning mechanical vibrations into an electric signal for further amplification.
Before the output signal is provided for the processing by external devices, it can be modified by the volume and tone controls. The volume controller is a potentiometer that regulates the output voltage. Tone controls consists of a potentiometer connected in series with a capacitor simply acting as a filter.

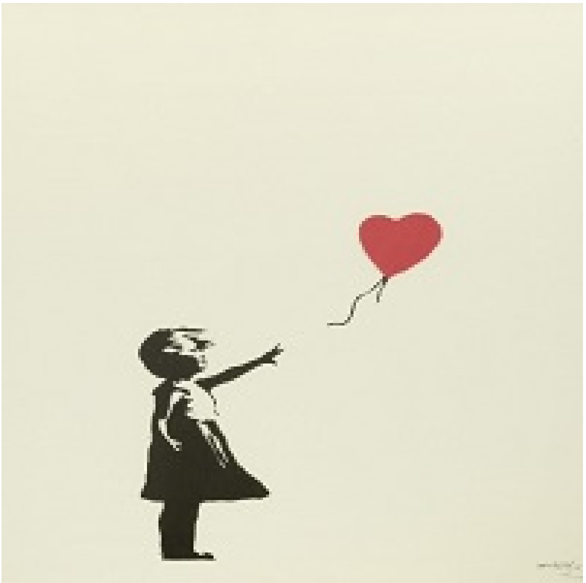

Figure 2.2.: Scheme of a single-coil pickup [**?** , p 29]

For this thesis only passive single coil pickups, as fitted to the majority of electric guitars, are described. A pickup basically consists of six bar magnets wrapped with a copper-wired coil, see figure (2.2). The magnets produce a stable magnetic field which is disturbed when a string is plucked. A vibrating string, made of ferromagnetic material, induces an electric Voltage into the coil.
According to Faraday's law of induction (see eq. 2.1) the Voltage u in a wound coil of wire with n turns is proportional to the change of the magnetic flux Φ over time.

$$u = n\frac{d\Phi}{dt} \tag{2.1}$$

[1]URL: https://www.guitarchalk.com/wp-content/uploads/2017/07/electric-guitar-parts.jpg [cited 22 August 2018]

Frequency Range

The sound from a guitar can be described as a note. Furthermore, every note corresponds to a fundamental frequency. The fundamental frequencies can be calculated (see eq. 2.2).
The hearable fundamental frequency $f_{\text{string/fret}}$ depends on the fret played and the frequency $f_{0,\text{string}}$ of the open string that is stroked. A regular six-string guitar with 24 frets in the standard-tuning (2.1) can generate fundamental frequencies between 82,41 Hz and 1318,5 Hz.
Besides the fundamental frequencies, an electric guitar produces harmonics at the same time. Harmonics are integer multiples of the fundamental frequency [**?** , p. 150]. These are overlapped with the fundamental frequency in the time domain which leads to the measurable signal form at the output.
The total frequency range of an electric guitar is extended by the harmonics up to several kilohertz. As a limiting factor the total hearable frequency range of a human ear, 16 Hz to 20 kHz [**?** , p. 28] should be taken into consideration.

$$f_{\text{string/fret}} = f_{0,\text{string}} \cdot \sqrt[12]{2}^{\,\text{fret}} \tag{2.2}$$

Open string	1(Low)	2	3	4	5	6(High)
Note	E	A	D	G	B	E
$f_{0,\text{string}}/\text{Hz}$	82,41	110	146,83	196,00	246,94	329,63

Table 2.1.: Standard tuning of a guitar[2]

2.2. Basics of Audio Engineering

Acoustical Level

The human ear is not designed for absolute loudness values. The principle of subjective loudness is based on stimuli-doubling and halving [**?** , p. 3]. Therefore the acoustic levels are described in logarithmic scale to represent human sensory perception. It is common to refer to a power of a signal or the intensity of the sound as a "level". To quantify the sound intensity the most common dimension used is the sound pressure level (SPL) L_{p0} measured in decibel. L_{p} is defined relative to the smallest sound pressure noticeable of $p_0 = 20\mu\text{Pa}$

[2]URL: https://en.wikipedia.org/wiki/Guitar_tunings [cited 22 August 2018]

as a reference level (see eq. 2.3). Table 2.2 shows a few exemplary sound sources and the resulting SPL at one meter distance.

$$L_\mathrm{p} = 20\log\frac{p}{p_0}\,\mathrm{dB} \tag{2.3}$$

Sound source	L_p/dB **at 1m**	**Sound source**	L_p/dB **at 1m**
Threshold of hearing	0	Street traffic	80
Quiet rural location at night	20	Pneumatic hammer	100
Library	40	Threshold of pain	120
Talking	60	Rifle	140

Table 2.2.: Example of SPL levels[3]

In audio engineering, the sound levels are defined by electrical levels. To have a corresponding translation from the acoustic level to the electrical level volume controls are often designed in a logarithmic scale [**?** , p. 34]. In addition to that, the electrical levels are labelled with a special index.
There are two important voltage levels used as a standard. The American and Japanese manufactures often use a reference voltage of u_0=1 V measured in decibel volts (dBV) (2.4). The European standard is defined by a reference voltage of u_0=0.775 V as a holdover from the early telephone standards. It is labelled decibel unloaded (dBu) (2.5) for a clear separation from decibel volts.

$$L_\mathrm{u} = 20\log\frac{u}{u_0}\,\mathrm{dBV} \tag{2.4}$$

$$L_\mathrm{u} = 20\log\frac{u}{u_0}\,\mathrm{dBu} \tag{2.5}$$

Voltage Bridging

In audio engineering, the transmission of analog audio signals usually happens by cable. The audio signals are in the baseband so they are transmitted without modulation. A transmission from an analog audio signal can be described as the transferring process from the source device towards the load device. Impedance matching is a requirement used in the field of

[3]URL: https://www.bsria.co.uk/news/article/acoustic-testing-what-is-actually-measured/ [cited 23 August 2018]

radio frequency transmissions to avoid reflections in the transmission line. The Goal is to gain maximum power transfer.
In audio engineering, the power matching principle is not applied. For the interconnection of two audio devices, a maximum voltage connection is necessary commonly known as voltage bridging.
The principle can be explained on the basis of a voltage divider (see figure 2.3). For a successful transmission, a much higher load impedance compared to the source impedance is required. In a worst-case scenario, the input impedance of the load is $Z_2 >> Z_1$ which would lead to no measurable voltage at the load. To have an efficient transmission of the signal, the voltage at the load device needs to be maximised. Most manufacturers adhere to a ratio of $Z_1 \geq 5 \cdot Z_2$ [**?** , p. 212].

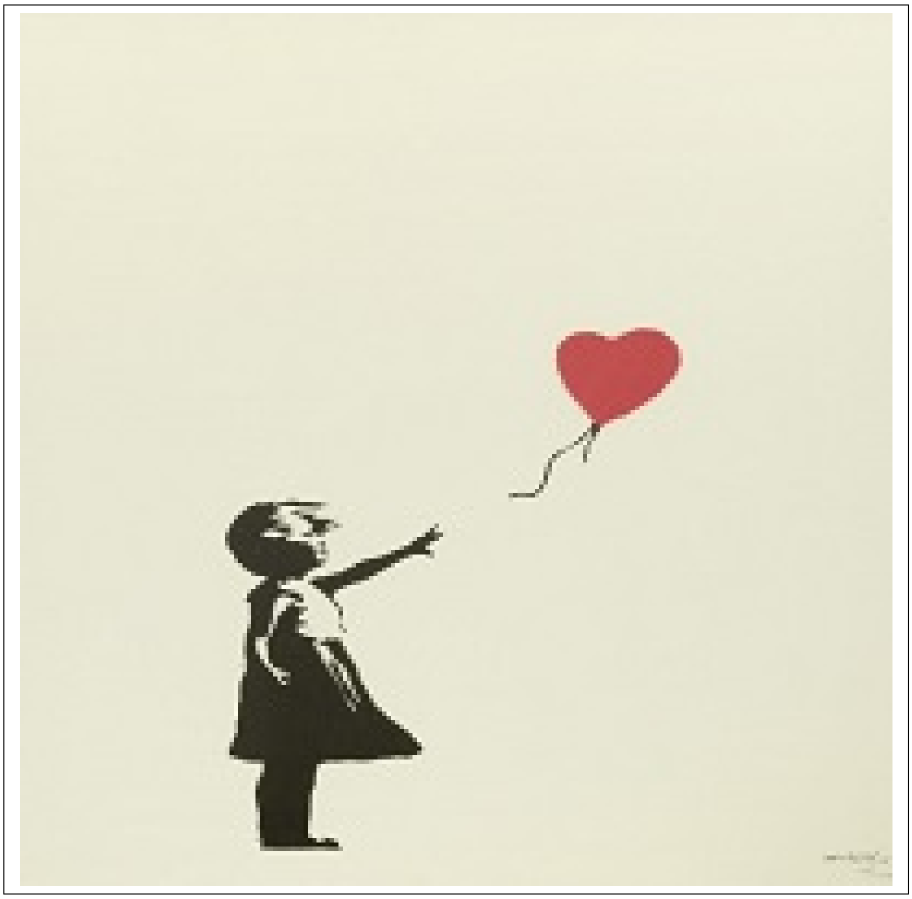

Figure 2.3.: Interconnection of two audio units (left) - voltage divider (right)[4]

[4]URL: http://www.sengpielaudio.com/calculator-bridging.htm [cited 23 August 2018]

Balanced and Unbalanced transmission

For the transmission of audio signals, two cable types are commonly used. Unbalanced cables consist of the centred signal wire and the shield, which is the conductor for the signal return as well. A shielded cable protects the signal from electromagnetic and radio interference. This noise rejection is only efficient up to a maximum length of 4,5 - 6 meter, due to the fact that the wire also acts as an antenna and picks up noise again[5]. Unbalanced cables are often used to connect an electric guitar or a microphone to an amplifier. In the domain of consumer audio, they are widely spread.

The balanced cable has in contrast to the unbalanced system one additional wire for the signal. Both wires transmit the same signal with a phase shift of 180 degrees. In other words, the second wire carries the inverse of the signal. Along the length of the cable, the two signal wires pick up the same additional noise. Since the noise parts of the two signals are in phase, both wires can be connected to a differential amplifier in the receiving device [**?** , p. 490]. By calculating the difference of both signals the noise can be eliminated.

Balanced audio systems can carry much longer cable runs compared to an unbalanced cable. They are used in professional sound systems or in recording studio environment for connections between amplifiers and mixing consoles.

Line Level

The line standard is distributed in a wide range of applications. It standardizes the audio transmission in many fields such as home entertainment, television broadcasting, and professional recording studios. It describes the nominal level as a ratio for a suitable interconnection between devices designed for line level.

For the consumer audio application (such as CD / DVD Players) the decibel volts (dBV) are used. The nominal level for consumer audio is specified as -10 dBV. In professional audio systems (e.g. mixing tables in television studios) the decibel unloaded (dBu) with a nominal level of +4 dBu is standard.

Table 2.2 shows the line levels with the corresponding nominal voltages which can be calculated with eq. 2.4 and eq. 2.5.

Input and output of the line connections present unequal impedances. Both designed to ensure a suitable interconnection from a line output to a line input. The impedance of an

[5] https://ask.audio/articles/music-studio-essentials-understanding-balanced-vs-unbalanced-xlr-cables [cited 25 August 2018]

input is typically around 10 kΩ while the output impedance is usual around 100 Ω to 600 Ω[6]. Due to the higher input impedance, the signal level is maximised and the current is kept low.

Application Field	**Nominal level**	**Nominal level,** V_{RMS}
Consumer audio	-10 dBV	0,316
Professional audio	+4 dBu	1,228

Table 2.3.: Line levels and their approximate nominal voltages

Instrument Level

Every passive instrument with an output connector for further amplification provides a different voltage level (see table 2.2). Depending on the instrument type, a variety of transducers are used to transform acoustical vibrations an electrical voltage. This is the reason why there is not one single voltage level valid for all instruments.

Type	**Output level**/ mV_{RMS}
Ribbon microphone	0,1
150-Ohm dynamic microphone	10
Fender Precision bass pickup	150
Humbucker guitar pickup	200
Piezo guitar pickup	0.5

Table 2.4.: Output levels for passive transducers [**?**]

Though for an electric guitar the range can be limited. Depending on several parameters like adjustment of the bridge, string cross section and strength of the string-stroke a pickup can produce an output signal between 10 mV_{RMS} and 1 V_{RMS}[7]. The strumming on an open A string of a *Fender Stratocaster* leads to a measured output-voltage about 72 mV_{RMS} (-22 dBV or -20 dBu) (see figure 2.4). Due to the additional harmonics, in this case the oscilloscope calculates the complex waveform as 480,4 Hz. These values can be taken as an assumption for a regular output voltage level of an electric guitar for further calculations.

[6] https://en.wikipedia.org/wiki/Line_level [cited 26 August 2018]
[7] http://www.muzique.com/lab/pick.htm [cited 26 August 2018]

The output impedance of electric guitars varies caused by the diverse guitar types and tone control settings. A guitar pickup is a high impedance inductive source commonly in the range of 10 kΩ to 25 kΩ[8].

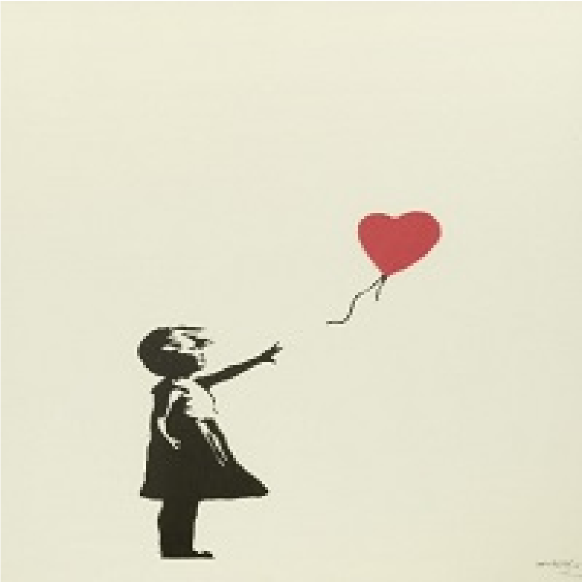

Figure 2.4.: Single-coil pickup waveform of an open A 110 Hz (Fender Stratocaster) [**?** , p. 27]

Analog Audio Connectors

For the interconnection of audio devices, several connectors and cables are used. In the area of analog audio engineering connectors such as XLR, RCA, and phone connectors (often called audio jacks) are to be mentioned. With regard to the following chapters, only the pinout of the relevant audio jacks (see figure 2.5) is shown in table 2.2.

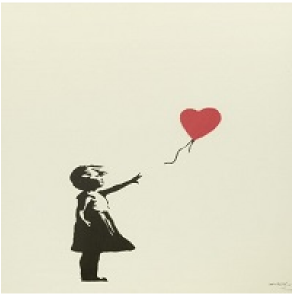

Figure 2.5.: Pinout diagram of a stereo audio jack in TRS standard[9]

[8] https://learn.sparkfun.com/tutorials/proto-pedal-assembly-and-theory-guide/theory-of-operations [cited 26 August 2018]

Audio jack	Pin 1	Pin 2	Pin 3
6.3 mm mono	SLEEVE: Ground	TIP: Signal	-
3.5 mm stereo	SLEEVE: Ground	TIP: Signal (Left)	RING: Signal (Right)

Table 2.5.: Pinout of audio jacks

2.3. Raspberry Pi and Audio-HAT

The Raspberry Pi is a single-board computer developed by the Raspberry Pi Foundation. By March 2017 over 12,5 million Raspberry Pis were sold, making it one of the best-selling "general purpose computer" behind Apple Macintosh and Microsoft Windows PCs[10] .

The first generation "Raspberry Pi 1 Model B" in 2012 came with a processor speed of 700 MHz, while the newer processors are running with up to 1,4 GHz. For this thesis the widely spread "Raspberry Pi 3 Model B" with a 1,2 GHz Quad Core Processor is used. A variety of operating systems like Android Things, Debian or the Debian-based Raspian can be installed via the SD-card reader. Equipped with many onboard interfaces, such as GPIO and display ports, the Pi can be used for a vast number of applications.
A major shortcoming is the analog audio output, which is designed as a 3.5mm audio jack. It provides a Signal-to-Noise ratio (SNR) of 65,5 dBA [**?** , pp. 72-74], which leads to a very weak sound quality. In addition to that, the Pi is not fitted with an analog audio input.

Therefore Sebastian Albers developed in the context of his bachelor thesis [**?**] an Audio-HAT (Hardware Attached on Top). The HAT extends the Raspberry Pi with professional analog stereo audio in- and output. Also placed on the HAT are the analog-to-digital conversion and the digital-to-analog conversion with a digital word size of 24 Bit and a maximum sampling frequency of 192 kHz.The SNR has been improved to 97 dB [**?** , p. 94]. Due to Albers achievements, the Raspberry Pi is therefore capable of digital signal processing.

[9] URL: https://robrobinette.com/images/Audio/TRS_Pinout.jpg [cited 25 August 2018]

[10] https://www.theverge.com/circuitbreaker/2017/3/17/14962170/raspberry-pi-sales-12-5-million-five-years-beats-commodore-64 [cited 26 August 2018]

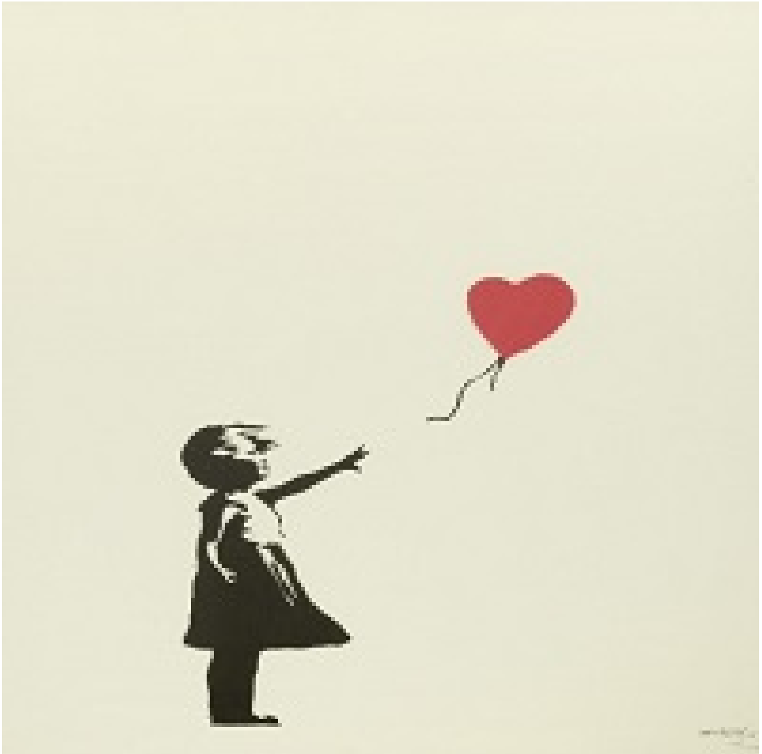

Figure 2.6.: Raspberry Pi Model 3B and Audio-HAT [**?** , p. 65]

2.4. Digital Signal Processing

For the manipulation or processing of analog signals, the digital signal processing is often used. Due to the greatly reduced prices for microelectronics and mass-production of digital circuits, the digital signal processing is widely distributed [**?** , p. 101].
Figure 2.7 visualizes the process in the context of digital audio effects (DAFX) which are part of this thesis. The analog signal $x(t)$ is a continuous-time signal carrying the information as electric pulses of varying amplitude. For the digital signal representation as a discrete-time signal $x(n)$ of the analog signal, an ADC is needed. The digitization consists of two steps: sampling and quantization.
The DAFX box, representing the digital system, performs a manipulation of the sequence of samples $x(n)$ in this case by halving the values. The processed digital signal $y(n)$ is then passed to the DAC for the reconstruction of the analog signal $y(t)$.

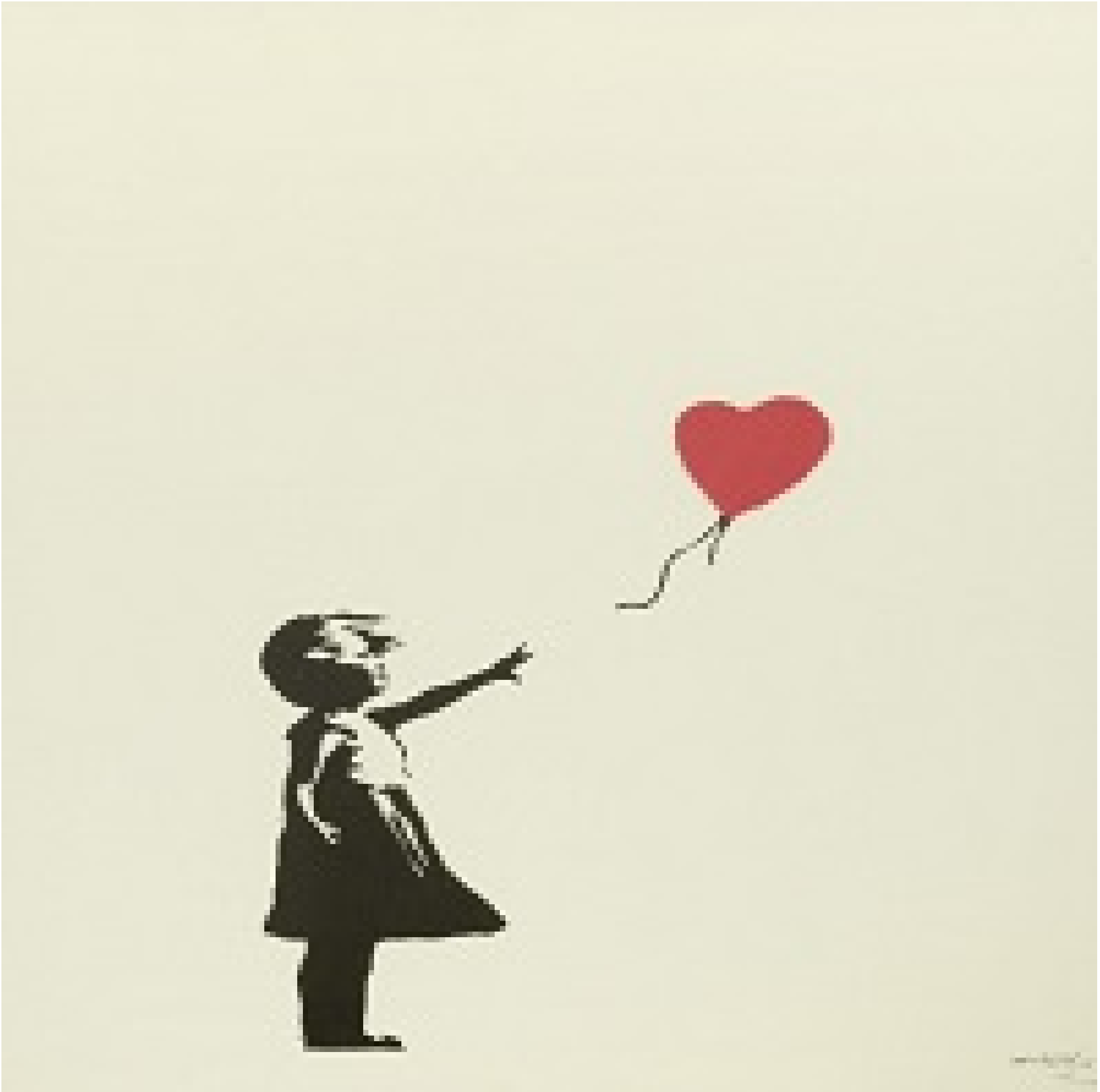

Figure 2.7.: Scheme of digital signal processing [**?** , p. 3]

2.4.1. Digital Signals

Sampling

The sampling rate f_s is the number of samples obtained per second in Hertz. For the sampling of the analog amplitudes on an equidistant grid along the horizontal time axis, the Nyquist-Shannon sampling theorem (2.6) must be respected. It defines that f_s must be higher than twice the maximum frequency f_{max} of the analog signal. If this condition is satisfied, the signal can be reconstructed from its samples. Not fulfilling the theorem, the sampling could lead to the aliasing effect [**?** , p. 103].

$$f_s > 2 \cdot f_{\text{max}} \tag{2.6}$$

Quantization

For the amplitudes of the analog signal, discrete values are allocated. They are represented by numbers $x(n)$ along the vertical axis symbolized by dots (see figure 2.7). The digital resolution, or the vertical scaling, depends on the defined bit depth. A higher bit depth leads to a minimized quantization error. For example, in a 24 bit representation of sample amplitudes the quantization is in the integer number range between -2^{24} to 2^{24}-1. The maximum digital value is defined as 0 decibels to full scale (dbFS). Based on that, the digital amplitude levels can be described in relation to a defined maximum.

Discrete Fourier transform

For the conversion of signals from the time-domain to the frequency-domain the discrete Fourier function (DFT) is used (eq. 2.7). Based on that the calculation of the Fourier transform of a signal can be done by a computer. The sequence $x(n)$ of N complex numbers is transformed to the frequency-domain $X(k)$. In addition to DFT, a more efficient way to perform the transformation is the fast Fourier transform (FFT). The algorithm is basically just an optimized DFT with a significant reduction of the number of multiplications. It is a mathematical technique of enormous technological importance because it allows powerful spectrum analysis on inexpensive microcomputers.

$$X(k) = \text{DFT}[x(n)] = \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/N} \tag{2.7}$$

$$k = 0, 1, ..., N-1$$

2.4.2. Digital Systems

The processing of the digital input signal $x(n)$, provided by the ADC, takes place in the digital system. An implemented algorithm processes every sample by performing mathematical operations. The result of the processing is the output signal $y(n)$.

Impulse Response

The reaction of a system triggered by a short-time signal is defined as the impulse response (see 2.8). The impulse can be modelled with the Dirac delta function $\delta(n)$ as a test signal. As a result, the output signal is the impulse response $h(n)$.

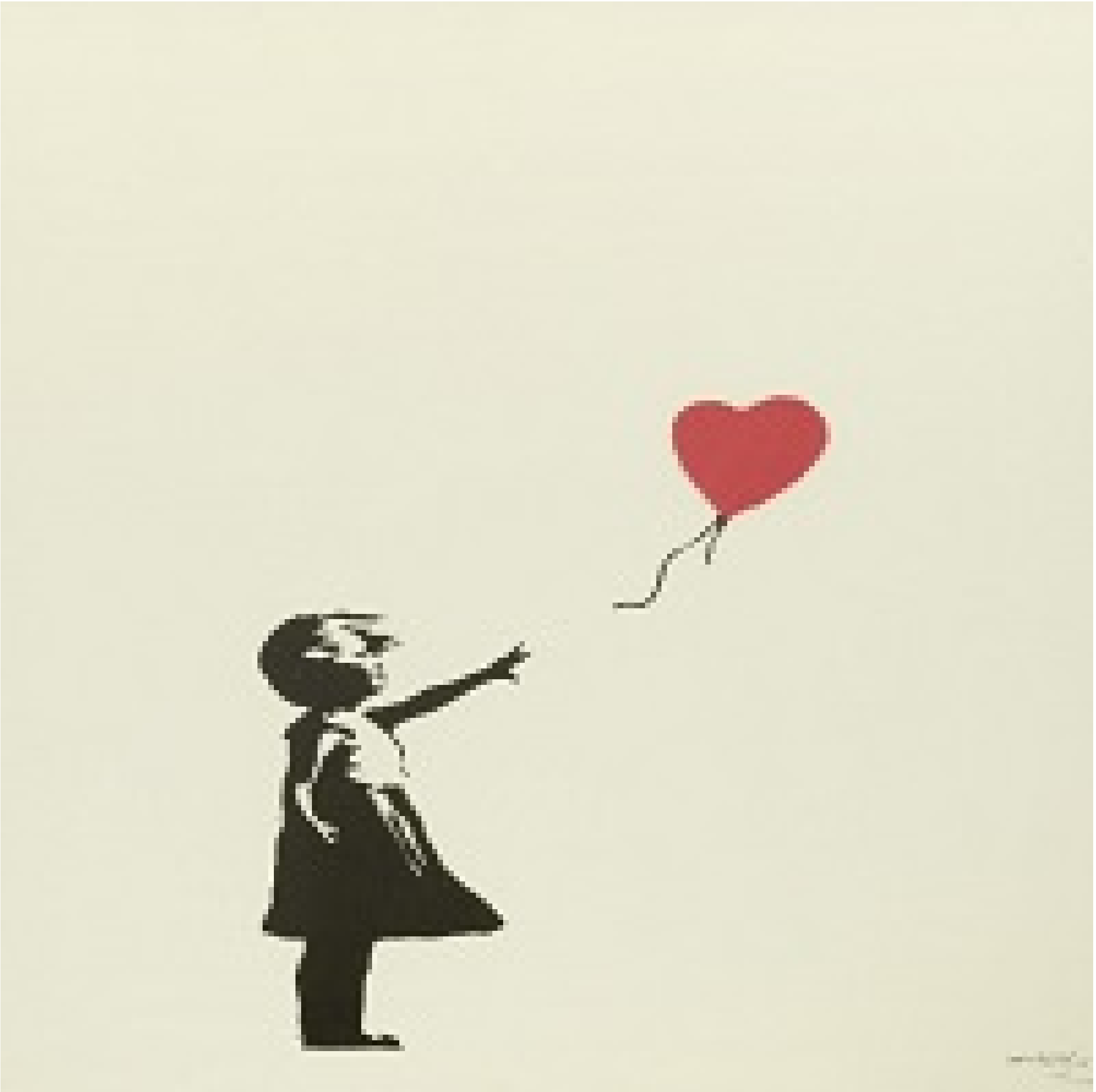

Figure 2.8.: Impulse response $h(n)$ as a time domain description [**?** , p. 18]

Transfer Function

In the time-domain, the behaviour of a digital system can be described with the impulse response. The reaction of the system in the frequency domain can be described as well. A system can reject, pass and enhance certain frequencies included in the input signal spectrum [**?** , p. 21]. The Transfer function (2.9) describes the frequency domain behaviour. It results by applying the Z-transform (2.8) which is mapping the discrete signal into a complex frequency domain representation. At this point, the complex Z plane is not explained in detail because it is not relevant for this thesis.

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) \cdot z^{-n} \tag{2.8}$$

$$H(z) = \sum_{n=-\infty}^{\infty} h(n) \cdot z^{-n} \tag{2.9}$$

2.5. Guitar Effects

Guitar players often want to highlight their own performance by applying specific sounds. The manipulation of the guitar sound via tone controls is very limited. Therefore to enhance the sound, effect units are widely spread. The units are connected between the guitar output and the input of an amplifier. Inside the unit the analog or digital signal processing takes place, changing the signal in the time-domain and/or in the frequency domain. A selection of popular guitar effect is depicted in table 2.6. In this section, the delay and distortion are described in more detail.

Effect	Description
Clean	No effect applied to the signal. Simple pass through
Equalizer	Variation of the total frequency spectrum. Adjusts specific frequency ranges
Delay	Time effect. Multiplication of the signal with a changed delayed copy of itself in the time domain. Creating an echoing sound
Chorus	Modulation effect. Comparable with the delay but in addition to that the frequency of each tone is modified
Tremolo	Manipulation of the amplitude. Varies volume of the sound very quickly over time
Boost	Increases the output signal of the guitar. Ideally used during solos
Overdrive	Signal amplitude is limited by using soft clipping. Creating a rounded but cropped wave and a distorted sound
Distortion	Signal amplitude is limited by using hard clipping. Leads to a full and saturated sound
Fuzz	Signal amplitude is amplified and cut off at the same time. Almost sounds like an overstrained speaker and leads to the most extreme distorted sound

Table 2.6.: List of popular guitar effects

2.5.1. Delay

In technology, a delay is often associated to be a negative effect in terms of signal transmission or processing. From an artistic point of view, the delay effect can improve the sound experience. In a comparable way that a sound benefits from a naturally reverberant space, a delay effect unit can create an echoing sound. A famous example of the usage in popular music is *Pink Floyd - Run like hell* [11]. The basic principle of a delay effect unit is shown in figure 2.9.

The resulting output signal is the addition of the two signal paths: The *Bypass Curcuit* and the *Delay Line*. Due to the *Bypass Curcuit* the guitar signal is passed forward to the output without any manipulation. Besides that, the *Delay Line* consists of recordings of the input signal and the recordings back from the *Feedback Line*. As a result, the sound is enhanced with a decaying echo.

[11] D.Gilmour/R.Waters (PinkFloyd) "Run like Hell". *The Wall.* Harvest Records,1980.LP

Figure 2.9.: Block diagram of the signal-flow for a typical simple delay-line tied to an electric guitar [**?** , p. 124]

2.5.2. Distortion

After the invention of the electric guitar, musicians looked for a way to create a louder and heavier sound. On one hand, they boosted the gain level of the guitar by experimenting with new pickups. On the other hand, they increased the volume inside on an amplifier until the vacuum tubes compressed and distorted the guitar signal. As a result, the distorted sound became one of the most important guitar effects and led to the birth of *Rock and Roll*. For instance, a typical distortion sound is used in the song *In Bloom* by the band *Nirvana*[12].

Any deviation of an output signal from the corresponding input signal is called distortion. There are two kinds of distortion to be distinguished: Linear and nonlinear (see 2.10).

[12]K.Cobain (Nirvana) "In Bloom".*Nevermind*. Geffen Records,1991.LP

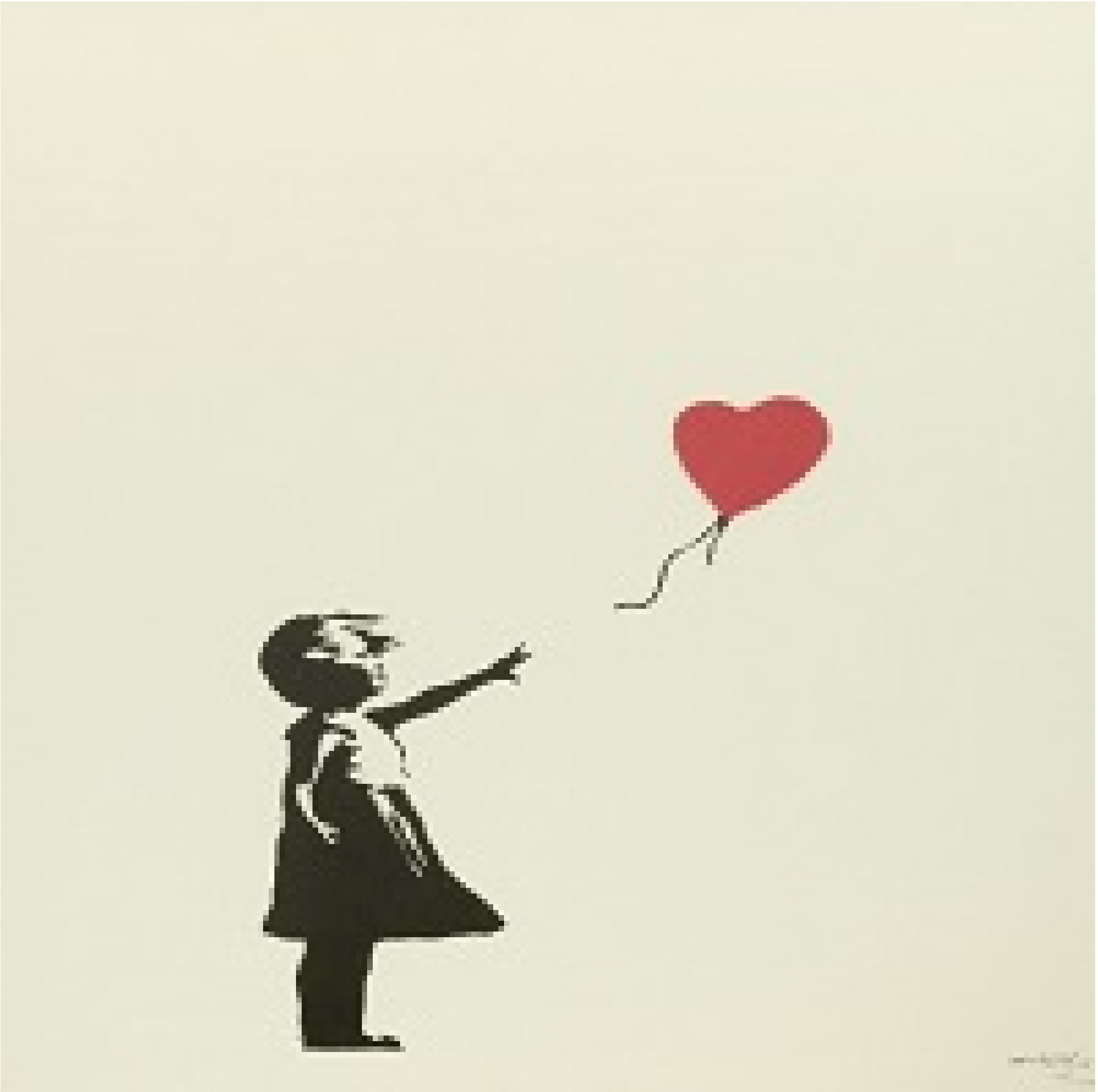

Figure 2.10.: Input and output signals of a linear and nonlinear system [**?** , p. 94]

Linear Distortion

Linear distortion appears when the processing does not affect the original waveform of the signal but a change in amplitude or phase. A simple example is a volume adjustment with no influence on the tone quality. Also, an equalizer for the amplification and/or attenuation of the frequency range is assigned to the linear systems. In addition to that, they occur by using frequency-dependent amplifiers and capacitive or inductive voltage dividers, which do not produce further harmonics [**?**].

Nonlinear Distortion

A nonlinear system affects an output signal that is strongly shaped. In regard to the distortion effect defined in table 2.6 the method of *hard clipping* is to be mentioned here. Nonlinear distortions are caused by curved characteristic lines of semiconductors and vacuum tubes [**?**]. According to that, these systems create harmonic frequency components that are not part of the input signal.

Depending on the symmetry of the components's characteristic lines, different parts of the harmonics are amplified.
A symmetrical cubic characteristic line highlights the odd integer multiples of the fundamental frequency f_1, thus $3 \cdot f_1, 5 \cdot f_1, 7 \cdot f_1$ and so on.
Unsymmetrical quadratic lines lead to even integer multiples such as $2 \cdot f_1, 4 \cdot f_1, 6 \cdot f_1$.

Total Harmonic Distortion

As an indication for the nonlinearity of a system, the total harmonic distortion (THD) is defined. It specifies the relative amount of harmonics expressed in percentage (2.11) or decibel (2.10) and describes the extent of distortion [**?**]. If the amplitude of the fundamental is U_1, and the amplitude of the n-th harmonic is U_i, then the THD for n harmonics can be defined. A lower THD means a more accurate reproduction of the input signal.

$$\mathrm{THD_{dB}} = 20 \cdot \log \frac{\sqrt{\sum_{i=2}^{n} \cdot {U_i}^2}}{\sqrt{\sum_{i=1}^{n} \cdot {U_i}^2}} \tag{2.10}$$

$$\mathrm{THD_{\%}} = 10^{\frac{\mathrm{THD/dB}}{20}} \cdot 100 \tag{2.11}$$

Besides the THD, a much more common method for the evaluation of an audio device performance is the total harmonic distortion plus noise (THD+N) measurement. In contrast to the THD, the harmonics are measured - by taking into account the resulting noise (see equation 2.12). As well as the THD measurement the THD+N is expressed as an RMS level.

$$\mathrm{THD}_{\mathrm{dB}} = 20 \cdot \log \frac{\sqrt{\sum_{i=2}^{n} \cdot {U_i}^2 + {U_{noise}}^2}}{\sqrt{\sum_{i=1}^{n} \cdot {U_i}^2}} \tag{2.12}$$

3. Requirements

The goal of this thesis is to develop a multi-effects unit for electric guitars. This device should be able to modify an electric guitar's signal according to the required effect. As a deliverable, the result should be hearable and measurable. Designed as a prototype development this project should also show potential improvements for further development. The requirements are split up into the hardware and software demands.

3.1. Hardware

All components of the unit should be placed in a 19-inch case designed for standardized mounting in a 19-inch rack. The unit is supposed to use only one power supply from the 230 V electricity grid. The digital signal processing shall be done by a Raspberry Pi in combination with the Audio-HAT developed by Sebastian Albers [**?**]. For the signal in- and output, the device should be equipped with two common standard 6,3 mm audio jacks on the front panel. That allows an easy interconnection with an electric guitar and a guitar amplifier. In regard to limit the application range, for this development cycle only input signals from passive six-string electric guitars are to be used. Other instruments like electric basses and keyboards are excluded from testing. In addition to that, a development of a preamp module is necessary for the adjustment from instrument level to the line level of the Audio-HAT's input. The effect unit itself shall not change the loudness of the signal from input to output. For the user control, a suitable user interface should be mounted on the front panel.

3.2. Software

To demonstrate the audio signal processing capabilities of the Pi in combination with the Audio-HAT, three exemplary guitar effects should be implemented: A clean, delay and distortion effect according to the hearable and measurable specifications (2.6). To control these effects a menu depending on the signals from the user interface should be implemented. Required features are:

- Visualize current settings on a display
- Switching to another effect
- Change three parameters of the effect

The unit is not supposed to provide a function to combine the effects with each other. In order to achieve a good stage performance, the maximum latency of the unit from input to output should not be more than $\Delta t = 10\,\mathrm{ms}$.

4. Design

In this chapter, the design process of the effect unit is described. Before the single parts of the device are developed, a thorough design phase is reasonable. The usage of a Raspberry Pi 3 Model B in combination with the Audio-HAT is foreseen (for further explanation see section 2.3). In the following sections, the choice of hardware and software in order to fulfil the requirements is explained and reasoned.

4.1. Hardware

4.1.1. Case

All components of the unit are supposed to be placed in one 19-inch case. 19-inch racks offer the advantage of mounting electronic devices into a standardized frame. In the field of recording studios or stage engineering, they are commonly used. Cases are available in different rack units U defining the height. One U is specified by 1,75inch. The used 19-inch case has $2U$ to provide enough space for the planned display (see 4.1.3).

4.1.2. Preamp Module

The input- and outputlevel of the Audio-HAT is designed for a reference voltage of $1\,V_{RMS}$ = 0 dBV [**?** , p. 31]. As a consequence, the Audio-HAT is developed for the interaction with devices on consumer audio line level (see section 2.2). Therefore a pre-amplifier (preamp) for the amplification from instrument level is reasonable to obtain a higher digital resolution and less quantization noise at the analog-digital conversion.

In addition to that, a suitable voltage bridging is necessary to gain a maximum voltage level on the HAT input. The existing $10\,k\Omega$ to $25\,k\Omega$ output impedance from a regular guitar does not ideally match with the $16{,}753\,k\Omega$ [**?** , p. 50] input impedance provided by the HAT.

The MXR-Micro Amp is a simple but efficient classical effect unit designed as a stompbox (figure 4.1). It is categorized as a *Boost* effect, which is a clean volume increaser without any modification of the sound. Taking the advantage of available schematics from the internet [**?**] the Micro Amp is used as a base for the preamp development.

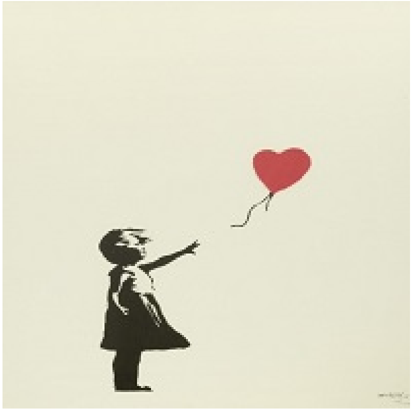

Figure 4.1.: MXR Micro-Amp stompbox[1]

The Micro Amp is designed in a non-inverting topology originally equipped with the operational amplifier TL061 [**?**] by Texas Instruments. The main advantage of the TL061 is the very low power consumption, necessary for a battery operating stompbox. A very low power consumption is not needed, so this part is replaced by the low noise TL071ACP, which is often used for high-fidelity and audio pre-amplifier applications [**?**].
Table 4.1.2 shows a comparison in terms of slew rate SR, equivalent input noise voltage V_n and THD to justify this choice.

operational amplifier	$SR/\frac{\mathrm{V}}{\mu\mathrm{s}}$	$V_\mathrm{n}/\frac{\mathrm{nV}}{\sqrt{\mathrm{Hz}}}$	$\mathrm{THD}_{\%}$
TL061	3,5	42	no information
TL071ACP	13	18	0,003

Table 4.1.: Comparsion of TL061 and TL071ACP operational amplifier

Besides the pre-amplification to adjust the guitar signal level for the Audio-HAT, the output signal of the HAT needs to be attenuated. The reduction of the voltage level from consumer audio line level back to instrument level is planned with a simple voltage divider.

[1] URL: https://www.constantinecruz.com/product/mxr-m133-micro-amp-pedal/ [cited 28 August 2018]

4.1.3. User Interface Module

On the front panel of the case, the user interaction takes places. The Raspberry provides a 40 pin header for the interaction with external components. 23 of the 40 pins are allocated for the communication with the Audio HAT [**?** , p. 99]. The remaining 17 pins can be used for the interconnecting with the user interface module via general purpose input/output (GPIO).

Display

In regard to an easy use, a suitable display is required to visualize the current state of the effect unit. There is a vast amount of liquid crystal displays (LCD) available for the Raspberry Pi such as graphical displays, touch displays or dot-matrix text modules.
The main selection criterion is the usability during live stage performances. Touch displays are a bad solution due to "sweaty" hands of a guitarist on stage. Graphical displays, with a resolution of 128x64 pixel or more, are not necessary because there are no complex graphics intend to visualize. For a simple navigation through the effects a text display is sufficient. Considering the word lengths to be displayed an appropriate resolution is 2x20 characters. Due to the difficult light conditions on stage, an LED background light is useful.

For an easy communication with the Raspberry Pi the chosen LCD 202A BL [**?**] by Display Visions provides an integrated HD44780 controller. The interaction via GPIO is possible and libraries including pre-developed functions are available on the internet. The 5 VDC supply voltage for the display is provided via GPIO.

Buttons and Switch

For the Power ON/OFF control of the effect unit a simple 230 V rocker switch with an imprinted I for ON and 0 for OFF is chosen. Besides that, two push buttons are required. One for the effect selection and the other one for shutting down the Pi. Both are foreseen to be connected via GPIO with the Pi, so the maximum switching voltage is in the extra-low voltage range (<120 VDC). For a convenient usage on stage, the chosen buttons provide a snap-action mechanism for a tactile and audible feedback provoking a hearable "click".

Rotary Encoders

As required, three parameters of each effect should be adjustable by the user. The modification of a parameter is simply interpreted as an increment or decrement of a value. The most user-friendly solutions are potentiometers or rotary encoders. The use of a potentiometer requires an extensive analog-to-digtal conversion. In addition to that, the potentiometer has an unavoidable memory effect caused by the turning position of the wiper. This would lead to implementation issues in terms of default values for the parameters.

Hence, KY-040 rotary encoders (see figure 4.2) are used to fulfil the requirements. Already soldered on a small printed circuit board (PCB) and equipped with pull-up resistors, the rotary encoders can be directly connected via GPIO.

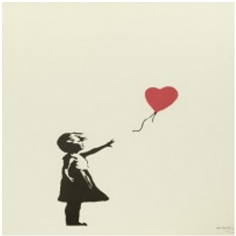

Figure 4.2.: KY-040 rotary encoder module[2]

4.1.4. Power Supply

According to the requirements stated before, the unit should be connected with the 230 V electricity grid. The unit requires a 230 V connection equipped with a protection-earth contact, caused by metallic electrically conductive parts (e.g. screws) touchable from the outside of the case.
For the design of the power switching supply, the significant criteria are the lower supply voltages needed for the built-in components. The Raspberry Pi has a micro USB connection for the 5.1 VDC operating voltage, covering the supply for the user interface via GPIO as well. The MXR Micro-Amp, on which the preamp module development is based, was originally designed for a 9-volt battery. The operational amplifier is running with a bias voltage (virtual ground). According to the specification of the TL071ACP [**?**] the recommended supply voltage for the total V_{CC} is between 10 VDC and 30 VDC. Therefore the desired power switching supply requires a minimum output voltage of 10 VDC. For the further reduction of the voltage level for the Raspberry Pi, a DC-to-DC converter can be used.

The electric power of the Raspberry Pi is about $5.1\,\mathrm{V} \cdot 2.5\,\mathrm{A} = 12.75\,\mathrm{W}$ [**?**], the low power consumption of the preamp module can be neglected. Therefore the power switching supply requires a minimum output power about 13 W.

[2]https://alltopnotch.co.uk/wp-content/uploads/imported/6/Rotary-Encoder-Module-KY-040-Brick-Sensor-Clickable-Switch-Arduino-ARM-PIC-UK-231884393106.JPG [cited 31 August 2018]

Table 4.1.4 shows the chosen power supplies. The power switching supply MW LRS-35-12 by Mean Well with an output power of 35 W is generously designed. This is reasonable in regard to possible further extensions such as the integration of a power amplifier. The step-down module LM2596 is designed for the direct connection to the power switching supply output, providing 5 VDC for the Raspberry Pi.

Type	**Input Voltage**	**Output Voltage**	**Output Current**
MW LRS-35-12	85 - 264 VAC	12 VDC	3 A
LM2596	3 - 35 VDC	Adjustable 1.5 -35 VDC	3 A

Table 4.2.: Specification of used power supplies [**?**], [**?**]

4.2. Software

The Raspberry Pi 3 B (plus Audio Hat) is the core component of the effect unit, responsible for the control of the main features such as audio processing and user interaction. It is an advantage that the interacting components are all available during the development phase. That allows an efficient hardware-near programming. Therefore it is planned that the whole software implementation is realised directly on the Raspberry Pi with mouse, keyboard and monitor attached to it.
Hence, cross-compiling from an external computer is not necessary.

Operating System

There are many different OS (Operating Systems) available for the Raspberry Pi. The official distribution *Raspbian OS* is in direct competition to other systems like *Windows 10 IoT Core* or *Ubuntu Mate*. For the choice of the ideal OS for this thesis, the main focus is on user-friendliness and the availability of relevant information and tutorials. In addition to that, the OS should provide the fundament for a necessary IDE. The Linux Debian based *Raspbian OS* is the most distributed operating system and supported by the Raspberry Pi foundation. Therefore it is chosen for the software developing process.

Programming Language

As a consequence of the required minimum latency of the audio signal, a fast programming language is reasonable. *C* or *C++* are common solutions in terms of time-critical tasks in close-to-hardware programming. The planned programming language for this project is *C*. The decision is made on the basis of significant advantages like available libraries for the display control and GPIO declaration. Moreover, the Audio-HAT demonstration program by Sebastian Albers [**?**], which provides a good basis for further implementations, is written in *C*.

Integrated Development Environment

As a suitable integrated development environment (IDE) the freeware *Geany* is chosen. Running on Raspbian OS, *Geany* is a simple but efficient program supporting a vast amount of programming languages including C. There are several plugins available, like a useful debugger or a syntax highlighting function. In fact, the program operates with very short load times, ideal for a hardware-near programming during test phases.

5. Implementation

In this chapter, the implementation phase is described in detail. After the fundamental design of the effect unit, all components are implemented separately in the first place. The following schematics and layouts are saved additionally on the attached CD (see A.1). During the realisation of the components, pre-testing approaches are reasonable to set adjustments for a good system performance.

5.1. Hardware Realisation

5.1.1. Preamp Module

Figure 5.1 shows the implemented circuit of the preamp module. The following descriptions and calculations are partially adopted from the available Analysis [**?**].

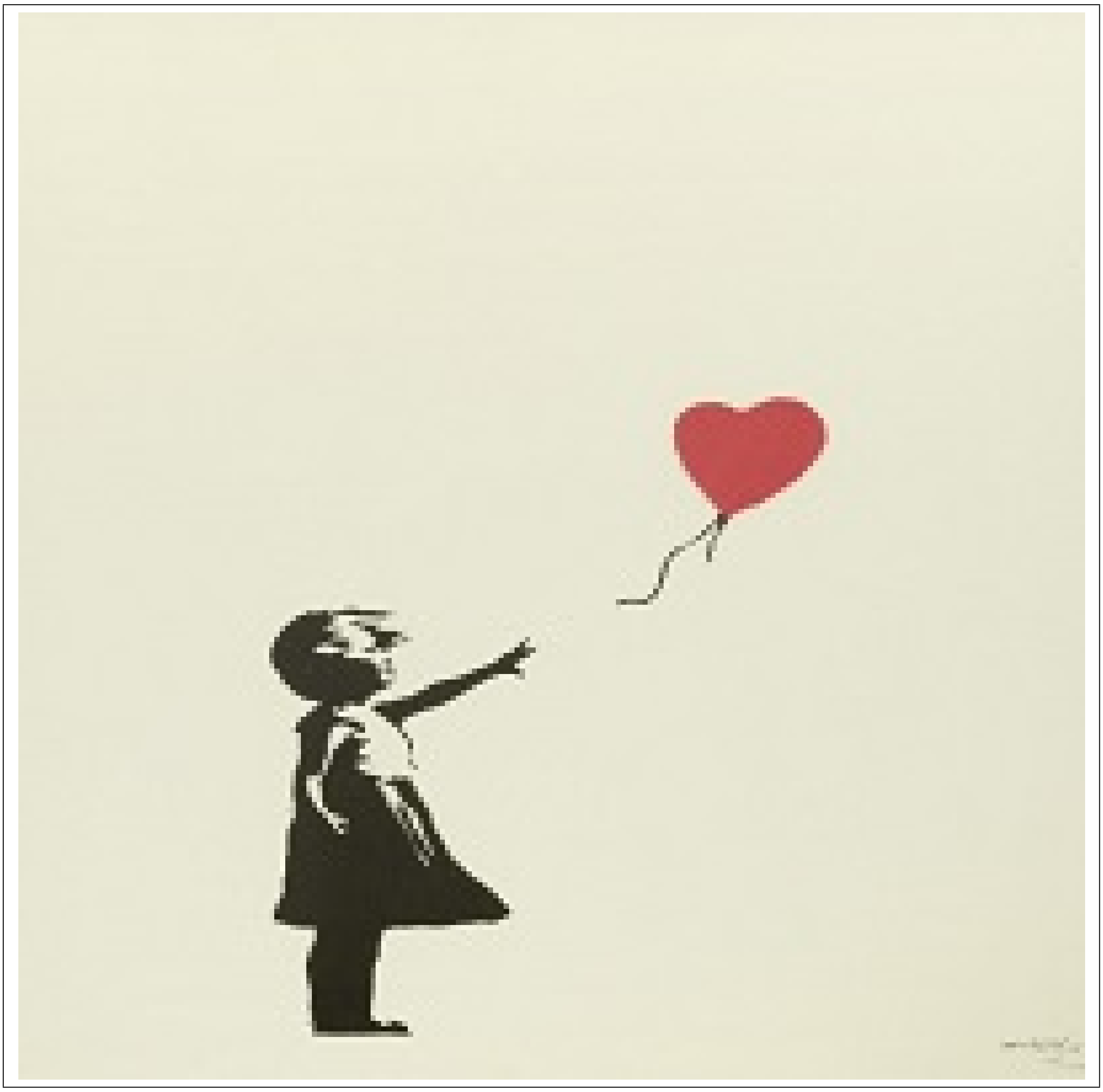

Figure 5.1.: Schematic of preamp module

Power Supply and Interfaces

Via a direct connection to the power switching supply output, the operating voltage of 12 VDC is established. The TL071ACP is supplied by the 12 VDC, whereas a voltage divider creates the needed 6 VDC bias voltage to ensure amplification of negative an positive signal amplitudes.

The preamp provides two 6.3 mm mono audio jacks for the external interconnection of the effect unit, designed as unbalanced connections. A guitar output is supposed to be connected with the *INSTRUMENT_IN* - a guitar amplifier via *INSTRUMENT_OUT*.
For the unbalanced connection with the Audio-HAT two 3.5 mm stereo audio jacks are implemented. Due to the connection of the TIP-Pin only the left channel is used (mono). The *LINE_OUT* is connected with the audio input of the HAT, *LINE_IN* receives the HAT's output signal.

Voltage Gain

The original Micro-Amp is equipped with a gain regulator for an amplification between 0 dB and 26,2 dB. For the preamp module a fixed amplification is used, suitable for different passive guitars. The goal is to raise the instrument level of the guitar to consumer audio line level. By a calculation done on a theoretical basis, the voltage gain can be assessed (5.1) assuming the RMS nominal levels (see section 2.2).
As a result of a practical pre-testing phase, the implemented voltage gain is set to 10,92 dB (5.2). This avoids hearable clipping caused by unexpected voltage peaks of different types of guitar pickups and/or heavy strumming styles.

$$\mathrm{Gain}_{\mathrm{theoretical}}/\mathrm{dB} = 20 \cdot \log\left(\frac{0,316\,\mathrm{V_{RMS}}}{0,072\,\mathrm{V_{RMS}}}\right)\mathrm{dB} = 12,85\,\mathrm{dB} \tag{5.1}$$

$$\mathrm{Gain}_{\mathrm{implemented}}/\mathrm{dB} = 20 \cdot \log\left(\left(1 + \frac{R_4}{R_5}\right) \cdot \left(\frac{R_7}{R_6 + R_7}\right)\right)\mathrm{dB} = 10,92\,\mathrm{dB} \tag{5.2}$$

The processed signal provided by the audio output from the HAT needs to be attenuated by the same factor. A simple voltage divider formed by *R8* and *R9* is used (see 5.3). The implemented attenuation differs from an expected loss of -10,92 dB, due to side effects of the Audio-HAT occurred in a practical test (see section 6.2.2).

$$\mathrm{Loss}_{/}\mathrm{dB} = 20 \cdot \log\left(\frac{R_9}{R_8 + R_9}\right)\mathrm{dB} = -11.04\,\mathrm{dB} \tag{5.3}$$

Voltage Bridging

In addition to the signal amplification, the required voltage bridging is guaranteed. For the interconnection with the guitar via *INSTRUMENT_IN* a suitable high input impedance is achieved (5.4). Provided by the $10^{12}\,\Omega$ high JFET Input Stage resistance of the TL071ACP [**?**].

At the *LINE_OUT* a lower impedance is implemented to ensure maximum voltage transfer towards the HAT's audio input (5.5), based on the TL071ACPs internal output resistance of $192\,\Omega$ according to the specification.
Both adjustments fulfil the required ratio between source and load impedance $Z_{\text{load}} \geq 5 \cdot Z_{\text{source}}$ (see section 2.2).

Guitar amplifiers commonly own a very high input impedance about $1\,\text{M}\Omega$[1]. As a consequence the voltage bridging for the attenuation path between *LINE_IN* and *INSTRUMENT_OUT* is not further analysed.

$$Z_{\text{INSTRUMENT\_IN}} = \left(R_1||R_2\right)\Big|\Big|\left(R_3||Z_{\text{in,TL071ACP}}\right) = 5\,\text{M}\Omega \tag{5.4}$$

$$Z_{\text{LINE\_OUT}} = R_7\Big|\Big|\left(R_6||Z_{\text{out,TL071ACP}}\right) = 657,64\,\Omega \tag{5.5}$$

Frequency Response

The preamp module is also equipped with four passive filters adopted from the MXR Micro-Amp to preserve its typical transparent and clean sound. These filters are not designed to cut-off relevant bass and treble frequency ranges of the characteristic guitar sound, but to eliminate low-frequency hum and harsh harmonics. Four RC networks are used, designed as high pass or low pass. The objectives of these filters are not further described in the context of this thesis.

$$f_{\text{cutoff},1} = \frac{1}{2\pi\Big(R_2||(R_3 + Z_{\text{in,TL071ACP}})\Big)} = 0,159\,\text{Hz} \tag{5.6}$$

$$f_{\text{cutoff},2} = \frac{1}{2\pi C_2 R_4} = 60,4\,\text{kHz} \tag{5.7}$$

[1] URL: https://www.soundonsound.com/sound-advice/q-what-are-correct-input-impedances-guitars-and-mics/ [cited 06 September 2018]

$$f_{\text{cutoff},3} = \frac{1}{2\pi C_3 R_5} = 1,532\,\text{Hz} \tag{5.8}$$

$$f_{\text{cutoff},4} = \frac{1}{2\pi C_4 R_7} = 0,159\,\text{Hz} \tag{5.9}$$

PCB Design

The layout of the preamp module is created with the Freeware version of *EAGLE (Easily Applicable Graphical Layout Editor)* by *Autodesk* [2]. The resulting printed circuit board (PCB) layout is depicted in figure 5.2. For the mechanical mounting of the preamp module four drill holes with the nominal diameter of 3 mm are foreseen. In addition to that, the 6.3 mm Audio jacks protrude over the edge for the direct mechanical coupling with the front panel of the case.
The assembled board is shown in figure 5.3.

[2]URL: https://www.autodesk.com/products/eagle/overview [cited 06 September 2018]

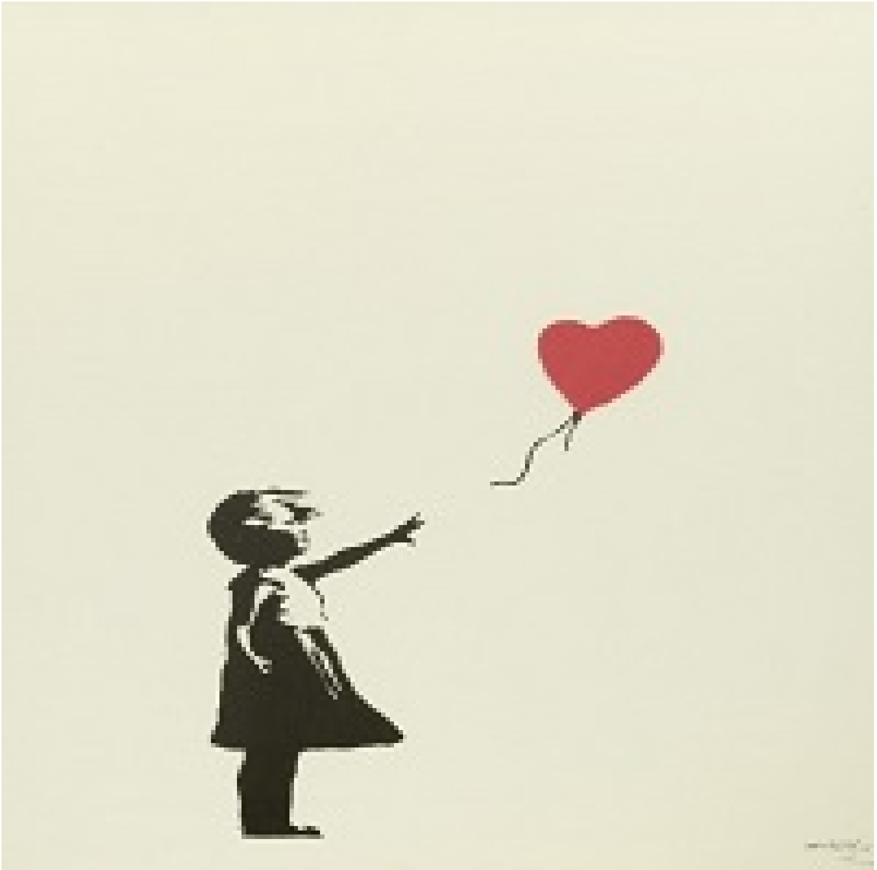

Figure 5.2.: PCB layout of preamp module

Figure 5.3.: Photo of preamp module

5.1.2. User Interface Module

For the connection of the chosen interface components (see section 4.1.3) the 40 pin-header of the Raspberry Pi is used. As mentioned before 23 of 40 Pins are allocated by the Audio-HAT, thus 17 pins are available.
Acting as an intermediate element a module is implemented to unify common voltage potentials and provide enough space for needed additional resistors. The Pi and the user interface periphery is connected via jumper cables with the module. Figure 5.4 shows the implemented board with the pin headers *X1* to *X6* for the periphery and *X7* for the direct forwarding towards the Raspberry Pi.

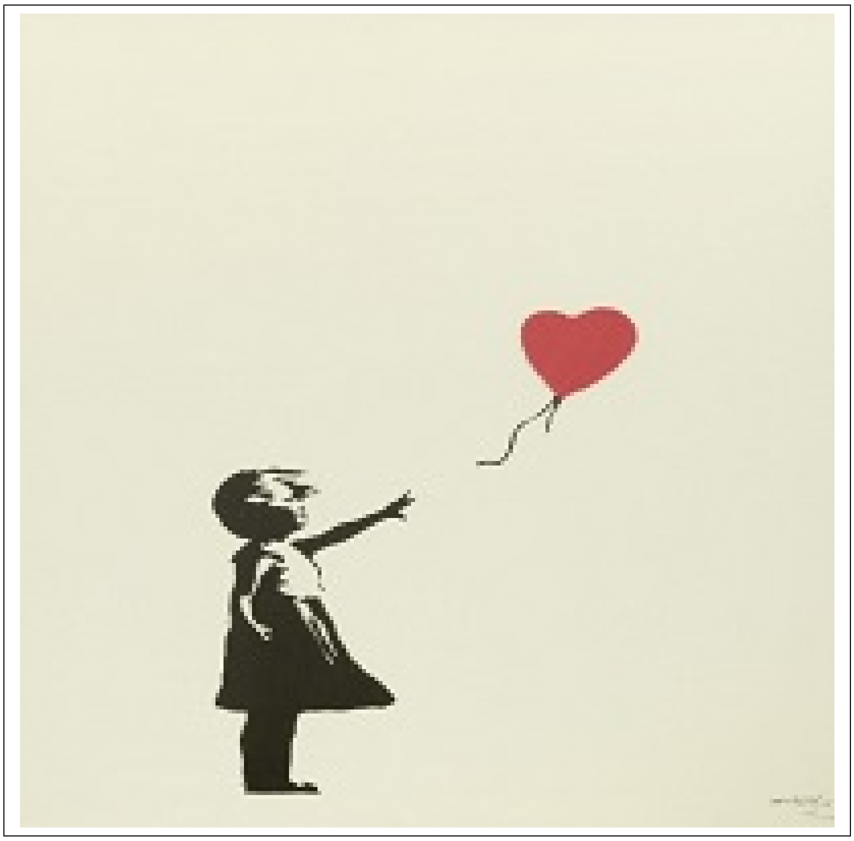

Figure 5.4.: Schematic of user interface module

Rotary Encoders and Buttons

The incremental rotary encoders KY-040 are equipped with five contacts. Besides the *+* and *GND* for the voltage supply, the two encoder signals *CLK* and *DT* are used to interpret the rotation direction. The usage of the *SW* contact, which is delivering a high signal when the encoder is pressed, is left out.

For the interconnection of the rotary encoders and the buttons, suitable pull-up and/or pull-down resistors are necessary. The appropriate debouncing is done on the software side. The onboard PCB of the rotary encoders already provides two 10 kΩ pull-up resistors for the encoder contacts *CLK* and *DT*. Hence, *X1* to *X3* can be directly connected with the desired GPIO pin of *X7*. For the buttons on *X4* and *X5* two 10 kΩ pull-down resistors are additionally implemented.

Display

The LCD 202A-Module is directly connected to the pin header *X6*. The display generally is operating in 4 bit or 8 bit mode. In the context of this effect unit, a high update rate for the display is not required, so the benefit of a faster 8 bit transfer can be neglected. Therefore, and due to the limited GPIO pins of the Raspberry Pi, the display is connected in 4 bit mode (see table 5.1).
For the adjustment of the LED backlight brightness, the resistor $R1$ is applied. As a result of an experimental test with a potentiometer, a value of 470 Ω delivers an appropriate brightness. For the LCD contrast adjustment $R2$ = 4,7 kΩ turned out to be a suitable choice.

Pin	Symbol	Level	Description
1	LED-	—	Back light cathode
2	LED+	—	Back light anode
3	D7	H/L	Data bit 7
4	D6	H/L	Data bit 6
5	D5	H/L	Data bit 5
6	D4	H/L	Data bit 4
7	E	H/L	Enable signal
8	RW	H/L	H : Read mode, L : Write mode
9	RS	H/L	H : Data signal, L : Instruction signal
10	VEE	—	Input Voltage for LCD contrast
11	VDD	5 V	Supply voltage for logic
12	VSS	0 V	Ground

Table 5.1.: Interface pin connections of LCD 202A module [**?**]

PCB Design

The development of the PCB for the user interface module is comparable to the construction of the preamp module. Four drill holes with the nominal diameter of 3 mm allow a tight mechanical coupling with the bottom plate of the case. Figure 5.5 shows the PCB layout created with *EAGLE*. The completely equipped board is depicted in Figure 5.6.

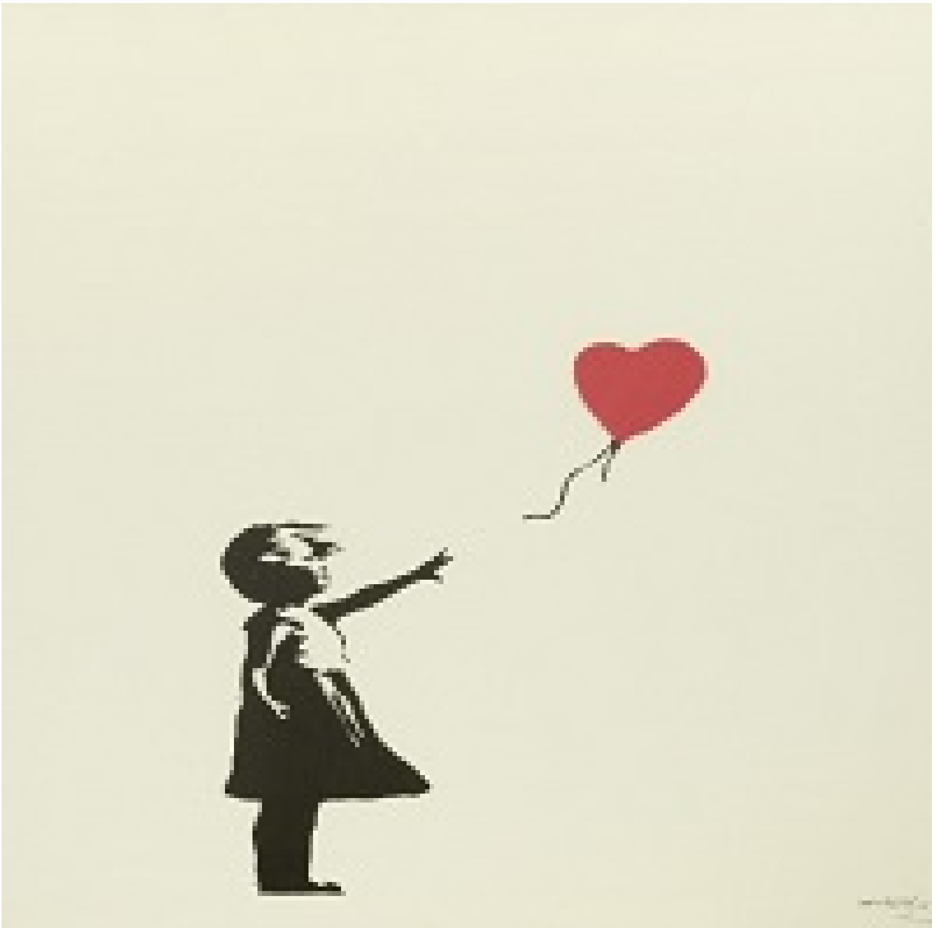

Figure 5.5.: PCB layout of user interface module.

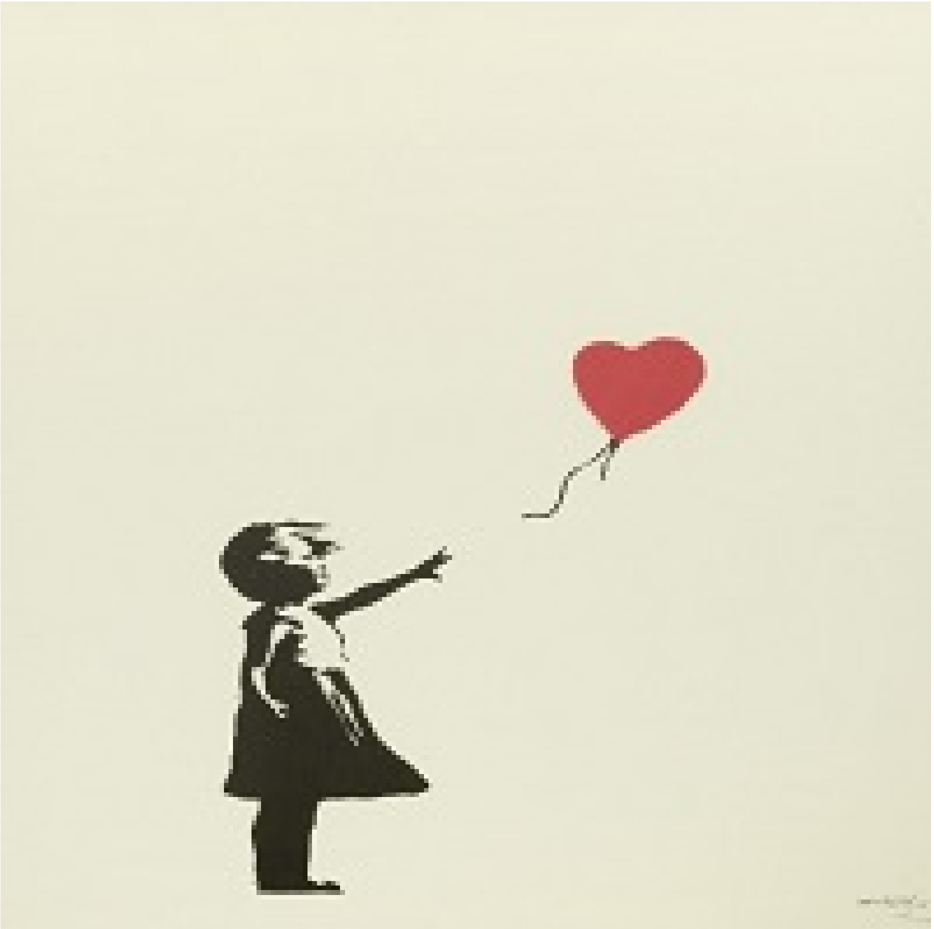

Figure 5.6.: Photo of user interface module

5.1.3. Construction of complete Unit

All modules and components are placed inside the 19-inch case. The mounting is realised with fitting screws and nuts, except for the 230 V power switch and the display which are fixed by a *Snap-In* mechanism.
A total schematic of the effect unit is placed in the appendix (see figure A.1).

Wiring of Power Supply

For a secure and reliable electrical wiring of the 230 V components, only *H07V-K <VDE>* stranded wires with a diameter of 0,75 mm^2 are used. In addition to that, suitable cable lugs and ferrules are applied as a standard in electrical enclosures. For the power supply wiring of the extra low voltage components, these security regulations are not mandatory.

Wiring of Signals

The control signals are realized with the use of jumper cables, ideally for a fast and easy connection. The resulting cable harness at pin headers $X1$ to $X6$ of the user interface module is bind together by cable ties. For the interconnection of the 40 pin header $X7$ with the Raspberry Pi, a ribbon cable is used as an ideal solution.

For the transmission of the audio signal from the preamp module to the Audio-HAT and back, unbalanced stereo audio cables are used. Suitable with the 3,5 mm stereo audio jacks of the preamp module and the HAT its a forward-thinking choice for possible future extensions. Figure 5.7 shows the total wiring of the effect unit.

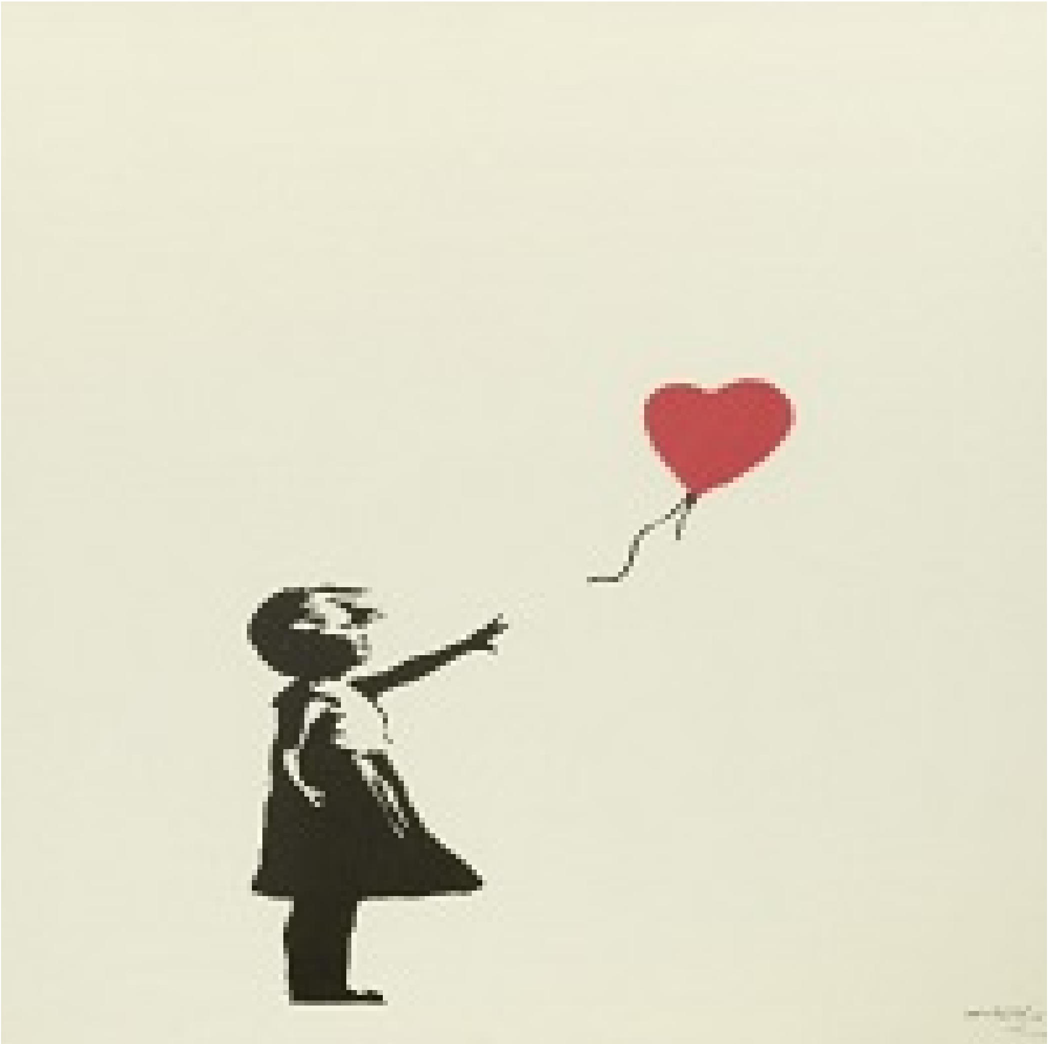

Figure 5.7.: Top view of effect unit (without cover)

Front Panel

The placement of the components on the front panel is designed to gain maximum usability (figure 5.8). The guitar signal input and output jacks are isolated on the left side. The display is centred on the front panel with a symmetrical formation of the rotary encoders below. As a consequence, the user is able to identify the corresponding rotary encoder for the variation of a parameter value. The remaining components for the user controls are placed on the right side.

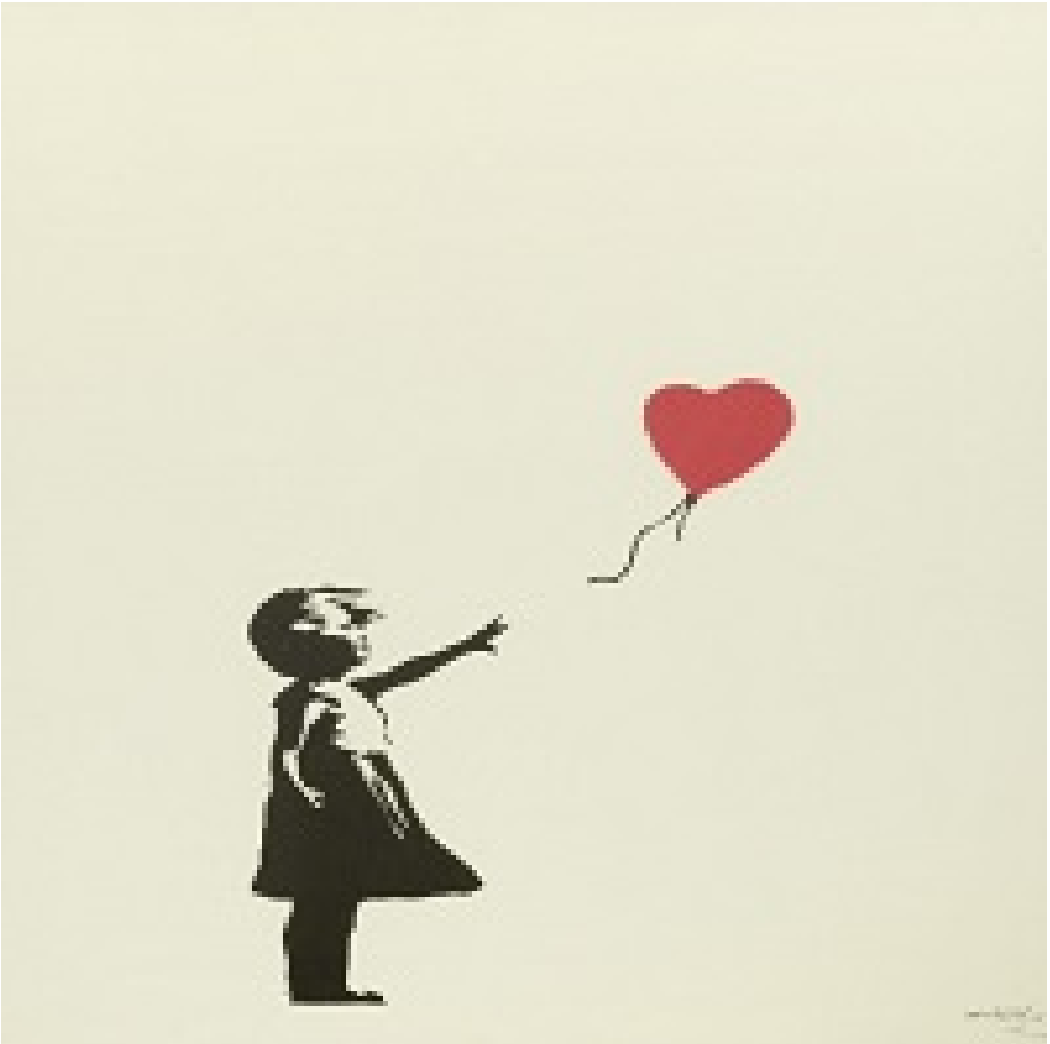

Figure 5.8.: Frontal view of effect unit (without cover)

5.2. Software Realisation

In order to create a better overview, the implemented software can be separated into two elements: user interaction and audio signal processing. Both elements in combination are important to reach the goal of a fully working effect unit. The final program *guitarEffects.c* (provided on the attached CD A.1) is called via the *rc.local* file of the Raspberry Pi, allowing the execution of the program while the Pi boots. Thus the effect unit is fully controlled via the user controls placed on the front panel.

5.2.1. Audio Signal Processing

The *i2cDemo.c* program provided by Sebastian Albers [**?**] was originally developed for the demonstration of the signal processing capabilities of the HAT. The program forms a good basis for the implementation for the required guitar effects, by initializing ADC and DAC communication and setting up the audio performance parameters.
For the audio processing, the sampling rate f_s is set to 48 kHz, fulfilling the demand of the Nyquist-Shannon sampling theorem (see equation 2.6). To ensure a high digital resolution, the bit depth is defined with 24 bit.
The implementation of the guitar effects extends the *i2cDemo.c* program by hearable and measurable sound changes. The manipulation of single samples is realised by taking advantage of the implemented FIFO (first in, first out) data buffer. The communication between the ADC, DAC and the Raspberry Pi is based on the I^2S (Inter-IC Sound) serial bus. For the left and right audio channel, a Pulse-code modulation (PCM) stream is provided.

Clean

For the clean effect, the samples are not manipulated. The input samples are simply passed forward to the output.

Delay

As depicted before in the block diagram (see 2.9) the delay a time-effect. An appropriate storage of input samples and the delayed addition with current input samples lead to the resulting output signal. In order to the requirements, a delay effect with three exemplary parameters is implemented. Figure 5.9 shows the signal-flow graph to visualize the algorithm.

The *level* parameter controls the general influence of the delay effect. As a factor placed at the end of the delay line it adjusts the total loudness of the delayed signal part.
For the variation of the fall time of the delayed samples, the *decay* parameter is used. Placed before the storage of input and delayed samples in the *delayBuffer[]*, the *decay* parameter can take the influence of the feedback path into account. The actual delay time for every sample is controlled by parameter *time*, defining the distance between the input and delayed sample in the time domain.

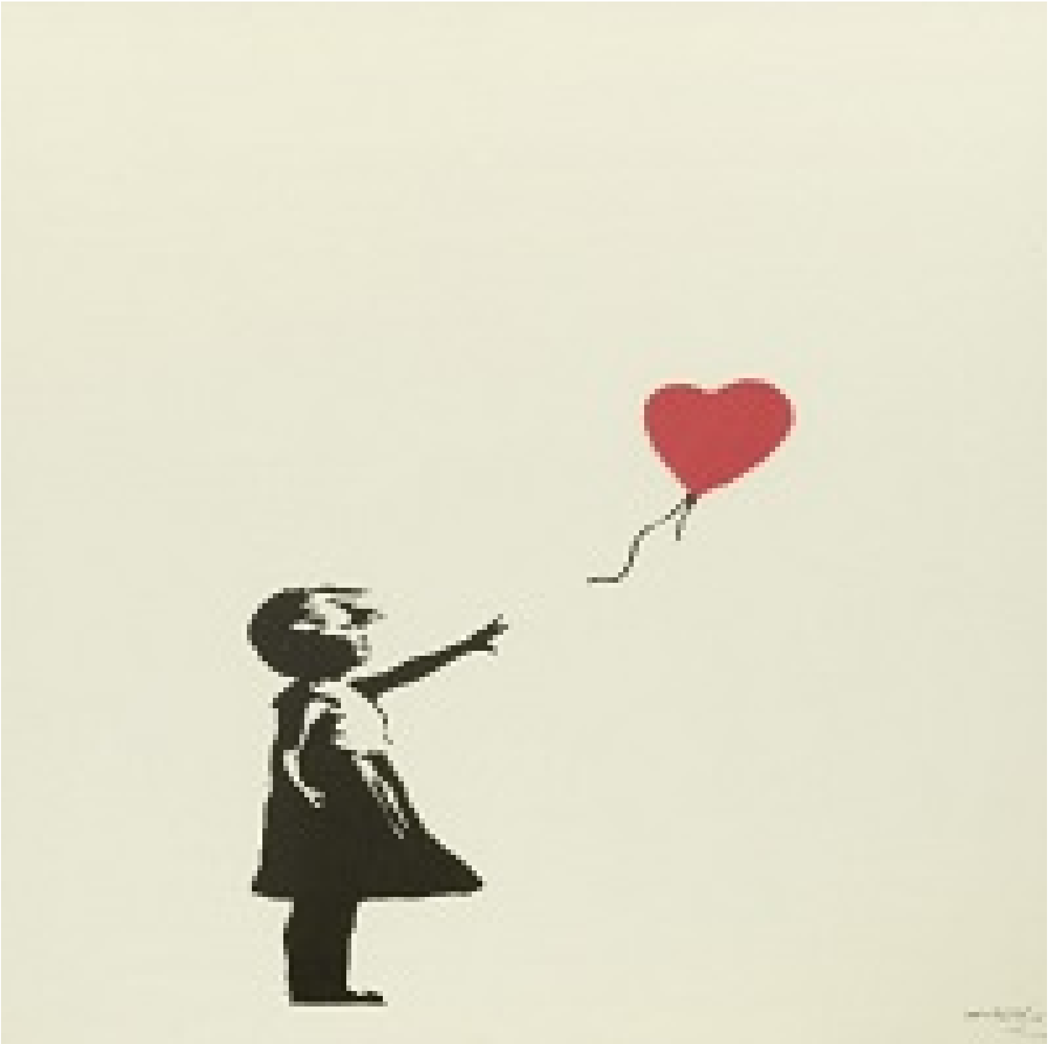

Figure 5.9.: Signal-flow graph of implemented delay effect

Listing 5.1 shows an source code excerpt of the implemented function *i2sDelayEffect()*. The parameter values are controlled by the variables *globalCounterParameter* set by the rotary encoders. For a user-friendly control, the adjustable and displayed values are in the range of 1 to 10. As a consequence, calculations are necessary for a suitable translation. Parameter *level* and *decay* are implemented as factors, controlling the height of the amplitude in percentage between 0 % and 100 % (stepsize = 10 %). The *time* parameter is calculated with the sampling rate f_s and the fixed value 5000 (see equation 5.10), which leads to a total value range of 0 ms to 1040 ms (stepsize = 104 ms).

$$\Delta t = \frac{\text{globalCounterParameter} \cdot 5000}{f_s} \tag{5.10}$$

```
// Calculate effect parameters
level = ((double)globalCounterParameter1 / 10);
decay = ((double)globalCounterParameter2 / 10);
time = globalCounterParameter3* 5000;

// Read from FIFO
inputSample = (long) p_mmap[PCM_FIFO_A];

// Fill delay fuffer and manipulate by decay factor
delayedSample = (inputSample + delayBuffer[delayCounter]);
delayedSample = (long)(delayedSample* decay);
delayBuffer[delayCounter] = delayedSample;

// Inkrement index to cause delay
delayCounter++;

// Clip Delay counter
if(delayCounter >= time){
        delayCounter = 0;
}

// vary gain level of delayed sample
delayBuffer[delayCounter] = (long)(delayBuffer[delayCounter]* level);
outputSample = (delayBuffer[delayCounter]+inputSample);

// write to FIFO
p_mmap[PCM_FIFO_A] = outputSample;
```

Listing 5.1: Excerpt of delay implementation

Distortion

There is a wide range of possibilities to achieve a typical distorted guitar sound. This distortion effect is implemented according to the description in table 2.6 and uses hard clipping. It is based on the analog method of placing two diodes antiparallel behind an amplifier connected to the ground. The amplitude of the guitar signal gets cut off by the time it reaches the threshold voltage of the diodes. As a consequence, the hard clipping leads to a very aggressive distorted guitar desired from many guitarists.
For this thesis, only one adjustable parameter for the threshold value is implemented for demonstration purposes.

The Audio-HAT is configurated with a bit depth of 24 bit. According to the specification, the

ADC *PCM1864* [3] is equipped with a full-Scale input of 2,1 V_{RMS}. So for the digital signal processing, the 0 dBFS level is assigned to a maximum analog level of 2,969 V_{Peak}. The DAC *PCM514* provided the same value as a ground centred output. Based on that, the U_{LSB}(voltage of the least significant bit) can be defined with equation 5.11.

$$U_{\mathrm{LSB}} = \frac{U_{\mathrm{FS}}}{2^n} = \frac{2 \cdot 2,969\,V}{2^{24}} = 3,539x10^{-7}\,\mathrm{V} \tag{5.11}$$

In a pre-testing phase with electric guitars for a suitable clipping-threshold, the hearable result was the most crucial factor. The distortion stages should vary in the range of 0 to 10 (from clean to slightly distorted and up to strongly distorted). A low threshold produces more hearable distortion by becoming a more square-wave-type signal. Unfortunately, a linear relation between the thresholds led to a more unbalanced subdivision, in terms of subjective perception.

Therefore the look-up table $\mathrm{factor\_table[n]}$ according to equation 5.12 is implemented to get thresholds with a non-linear relation. As a result of a multiplication (5.13) of these factors with the fixed number 50000, sample values can be calculated. These values are representing the threshold voltage (see eq. 5.14). Table 5.2 shows the translation of the $\mathrm{globalCounterParameter}$ controlled by the rotary encoder. Figure 5.10 illustrates the performed hard clipping for every parameter in the time domain.

$$\mathrm{factor\_table[n]} = 30 - 9.25\sqrt{\mathrm{globalCounterParameter}} \tag{5.12}$$

$$\mathrm{sampleValue} = \mathrm{factor\_table[n]} \cdot 50000 \tag{5.13}$$

$$U_{\mathrm{Threshold}} = U_{\mathrm{LSB}} \cdot \mathrm{sampleValue} \tag{5.14}$$

[3] http://www.ti.com/lit/ds/symlink/pcm1864.pdf [cited 10 September 2018]

globalCounterParameter	**factor_table[n]**	**sampleValue**	$\pm U_{\text{Threshold}}/\text{mV}$
1	20,75	1037500	367
2	16,62	846000	299
3	13,98	699000	247
4	11,5	575000	203
5	9,32	466000	165
6	7,34	367000	130
7	5,53	276500	98
8	3,84	192000	68
9	2,25	112500	40
10	0,75	37500	13

Table 5.2.: Implemented distortion threshold values

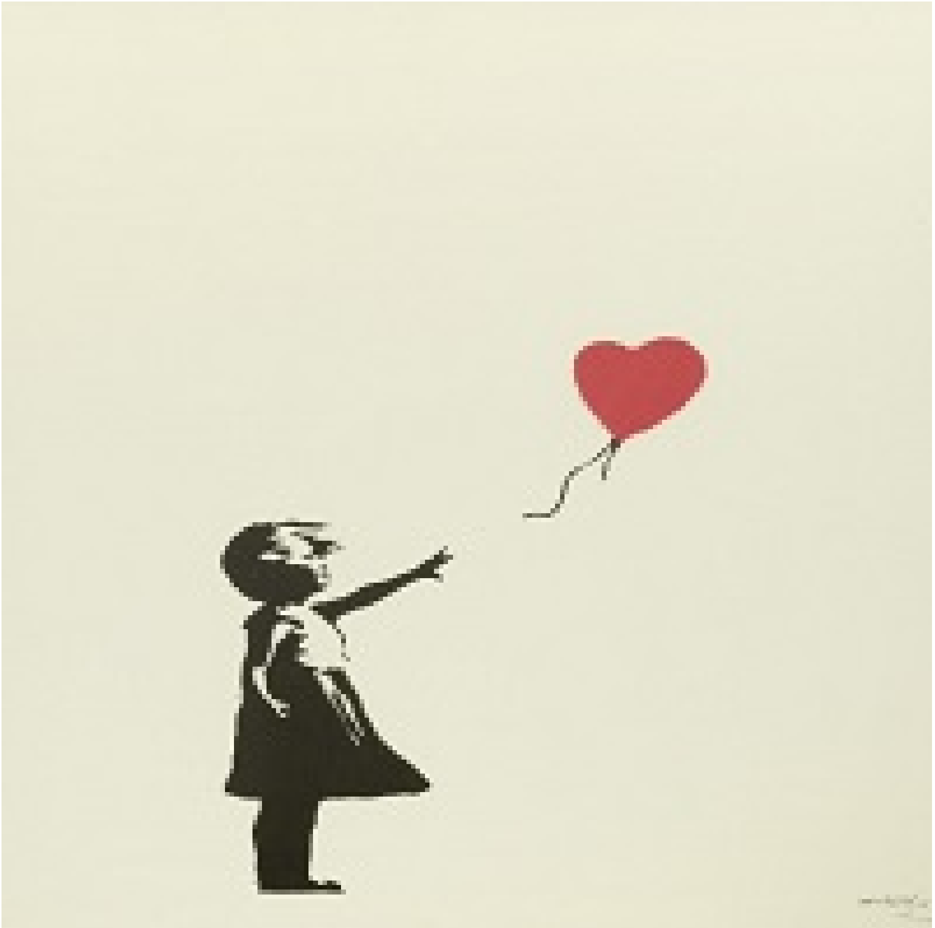

Figure 5.10.: Implemented hard clipping for globalCounterParameter 1 to 10

After removing all values above the desired threshold, an adjustment is implemented to set the amplitudes to the same voltage level. The implement common level is lower than the original amplitude of the clean signal. This is necessary because the generated harmonics lead to a subjective louder sound.
In a pre-testing phase the adjustment-level is set to $\pm$ 160 mV_{Peak}. As a result, the distortion effect has a similar subjective loudness compared with the other effects.

5.2.2. User Interaction

As described in 5.1.2 all user interface components are connected via GPIO with the Raspberry Pi. Instead of an elaborate *Direct Register Access*, the *WiringPI* [4] GPIO access library

[4]https://github.com/WiringPi [cited 12 September 2018]

is used for an easy way of setting up, reading and writing the pins.
Moreover, *WiringPI* includes an LCD library with helpful functions for the display communication.

The pins allocated to the buttons and rotary encoders are set up as interrupts. Once requested, the audio processing is changed by an user interaction. As a consequence, the display is updated with the current status.

In regard to the requirements, a user-friendly menu for the effect navigation is implemented (5.11). After the effect unit is turned on with the *Power ON/OFF* switch, the clean effect is executed. For this effect, no parameters are foreseen. By pressing the *Switch effect* button a new effect is called. The display is showing the effect name and the three parameter values starting with the default value 5. These values are changeable by turning the corresponding rotary encoder. For the delay effect the following parameters are assigned (from left to right): *level - decay - time*.
Only the left parameter of the distortion effect is used for the threshold adjustment. The remaining can be interpreted as dummies. By pressing the *Shut down* button at any time, the Pi can start shutting down. After about 5 seconds the voltage of the power switching supply needs to the interrupted manually by pressing the *Power ON/OFF* switch.

Figure 5.11.: Diagram of user menu

6. Test

After the implementation, the components and the complete unit itself need to be tested. The results are evaluated particularly with regard to the required features. Table 6.1 shows the laboratory instruments primary used during the testing phase. Besides a function generator and an audio analyzer as common sources for input signals, the unit is tested with an electric guitar (Stratocaster type - single coil pickups with 6 strings). In combination with a guitar amplifier the subjective auditory impression can be evaluated.

Instrument	**Type**
Peaktech 3500FG	Function Generator
Tektronik MSO2024	Oscilloscope
Rohde & Schwarz UPV	Audio Analyzer
Harley Benton ST-20 BK	Electric Guitar
Marshall Park G10	Combo Guitar Amplifier

Table 6.1.: Test equipment

6.1. Preamp Module Performance

The preamp module is tested to the two major demands given in the requirements in the first place: The voltage level adjustment and the voltage bridging. Measured in the time domain with the oscilloscope, the transition of a sine wave is measured. The input signal is provided by the function generator, representing a signal within the voltage- and fundamental frequency-range of an electric guitar. For the inspection of the voltage bridging, the preamp module is tested with and without a connection to the active Audio-HAT (unloaded and loaded).

Instrument Level to Line Level

For the amplification path of the preamp module, an input signal with 108 $\mathrm{mV}_{\mathrm{Peak}}$ at 200 Hz is used. The measured transition (unloaded) from instrument level to line level has a deviation of 3,12 % (6.2), which can be reasoned by production-related deviations of the used parts. For the interconnection with the Audio-HAT, a high voltage transfer is ensured. The signal amplitude is only slightly attenuated by 4,35 % as result of suitable voltage bridging(see figure 6.1 and figure 6.2).

$U_{\mathrm{INSTRUMENT\_IN}}/\mathrm{mV}$	$U_{\mathrm{LINE\_OUT}}/\mathrm{mV}$	$\mathrm{Gain}_{\mathrm{measured}}$	$\mathrm{Gain}_{\mathrm{implemented}}$	deviation/%
108	368	3,407	3,517	3,12

Table 6.2.: Measurement of pre-amplification

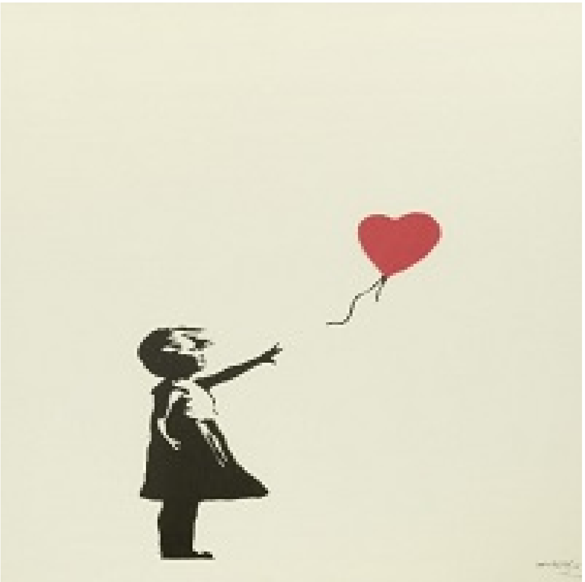

Figure 6.1.: Pre-amplification (unloaded)

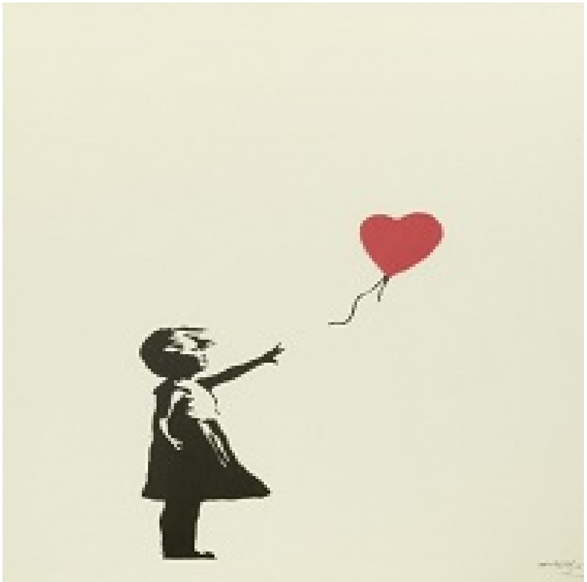

Figure 6.2.: Pre-amplification (loaded)

Line Level to Instrument Level

As an exemplary input signal for the attenuation path 360 $\mathrm{mV_{Peak}}$ at 200 Hz is suitable. For the adjustment of the signal back to instrument level, a minimal deviation of 0.714 % (6.3) is achieved. This is reasonable due to the non-complex used solution as a voltage divider. An oscilloscope measurement is depicted in figure 6.3.

$U_{\mathrm{LINE\_IN}}/\mathrm{mV}$	$U_{\mathrm{INSTRUMENT\_OUT}}/\mathrm{mV}$	$\mathrm{Gain}_{\mathrm{measured}}$	$\mathrm{Gain}_{\mathrm{implemented}}$	deviation/%
360	100	0,278	0,280	0,714

Table 6.3.: Measurement of attenuation

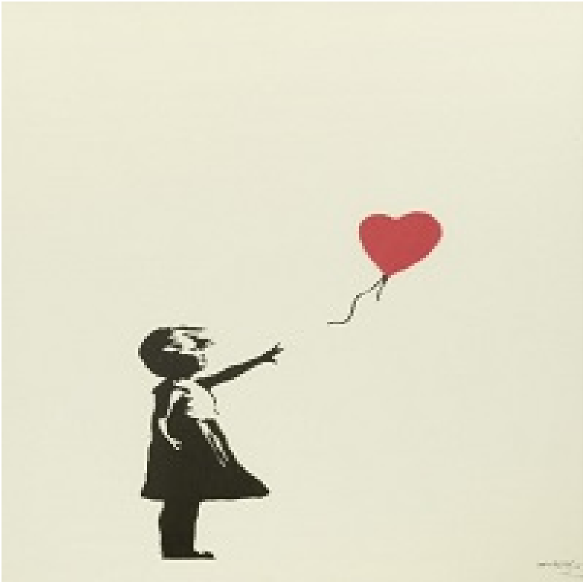

Figure 6.3.: Attenuation

Frequency Response

The Audio Analyzer is used for the measurements of the preamp module in the frequency domain. Figure 6.4 shows the frequency response of the amplification path using a sine sweep between 10 Hz and 20 kHz. The characteristic flat frequency response of adopted MXR Micro-Amp design is ensured within the range of the fundamental frequencies between 82,41 Hz and 1318,5 Hz (see section 2.1). The curve is damped in the range of higher frequencies. The minimal reduction of 4,5 % ($\widehat{=}$ 16 mV) can be neglected, so the guitar harmonics are appropriately amplified.

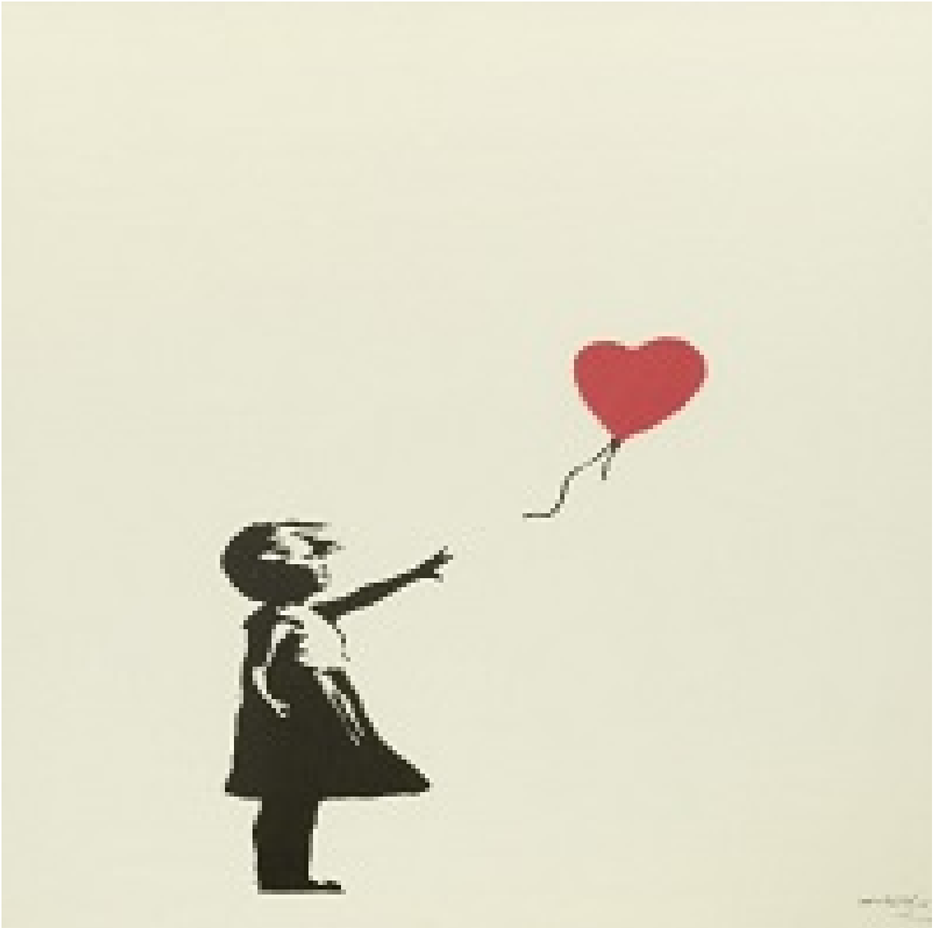

Figure 6.4.: Preamp frequency response

Total Harmonic Distortion plus Noise

For the evaluation of the audio performance, the THD+N is measured with the Audio Analyzer. As a standard for THD+N measurements, a sine-wave signal of 1 kHz is used. The amplitude is set to 100 mV_{RMS} representing a realistic output of an electric guitar. The resulting THD+N of -85.003 dB (0.006 %) indicates a very good performance in regard to the desired application field.

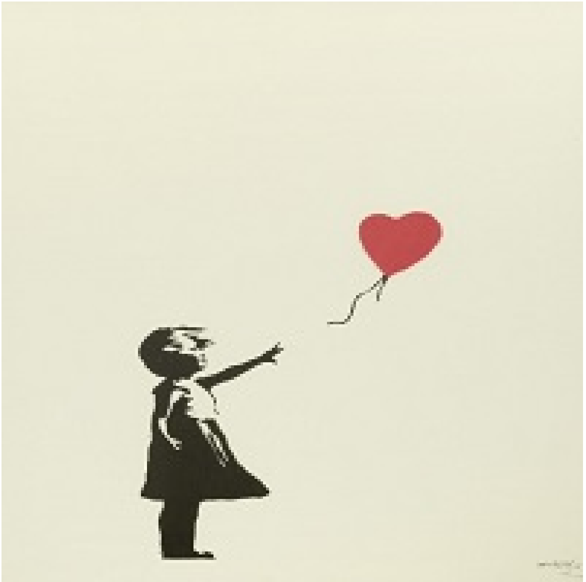

Figure 6.5.: Preamp THD+N

Hearable Results

In a practical test, the isolated preamp module is verified with the usage of the electric guitar signal. The amplification path is tested in the first place by connecting the *LINE_OUT* with an AUX(auxiliary) input of HIFI-Amplifier and with the input of a PC-Soundcard. In both cases, the hearable result is a clean undistorted sound from the subjective point of view. A possible extension for further developments might be a usage of the consumer audio line level outside of the effect unit.

In a further test, the *LINE_OUT* is connected directly with *the LINE_IN* to test the preamp module as a whole. According to the defined requirements, the total signal processing shall not change the loudness of the guitar signal. The module is connected via the *INSTRUMENT_OUT* jack with the Input of the guitar amplifier. As a hearable result, the guitar-amp gains the same loudness compared to the direct connection of the guitar (at the same amplifier- and guitar settings).

6.2. Measurement of Guitar Effects

In the following section, the implemented guitar effects are tested. During the tests, the unit is exclusively controlled by the user interface module. Due to the combined usage of all in-build components, these tests can be interpreted as a total system validation. The clean effect does not need to be measured in a separate test. The exact clean sound is included in the distortion test by setting the parameter to 0.

6.2.1. Delay

Based on a practical test using a guitar signal, the delay effect is validated. The effected signal is measured in the time domain for a direct comparison with the original signal. The measurements do not include all possible permutations of the three parameter values from 0 to 10. To visualize and verify the effect, only one parameter is changed while the others are set to 5 as a default value. Figure 6.6 shows the measurement for one exemplary parameter setting. The oscilloscope screenshots of the entire measurements are placed in the appendix (see A.3).

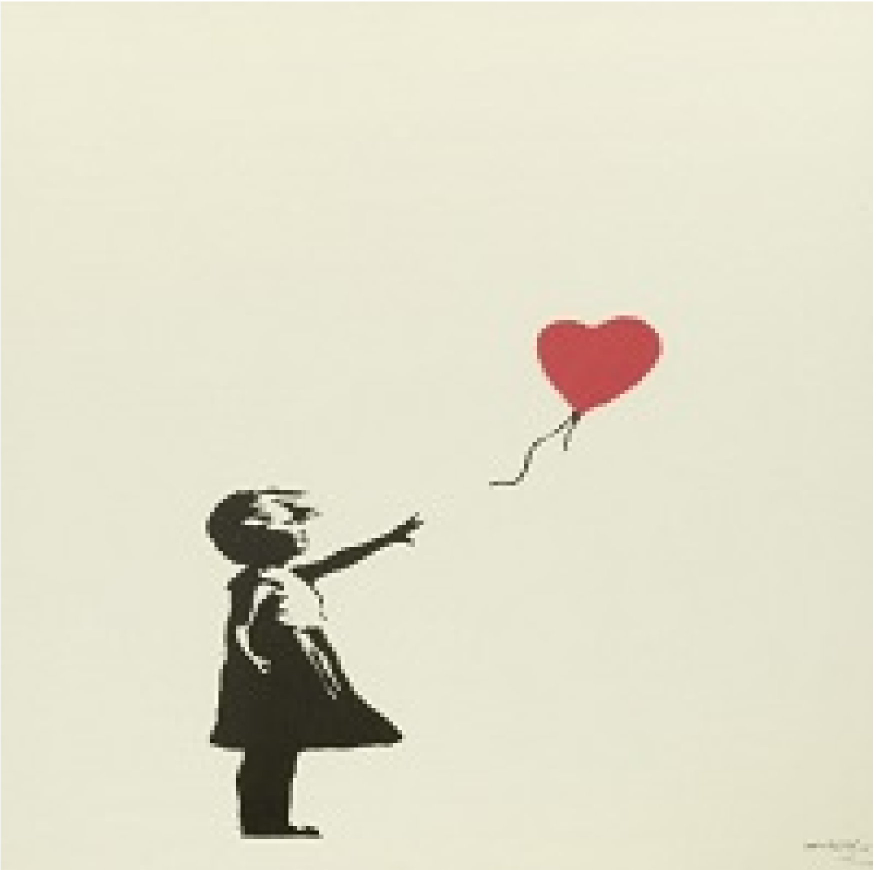

Figure 6.6.: Delay-measurement with *level=5*; *decay=10*; *time=5*

Time Parameter

Based on equation 5.10 using the sampling rate, the *time* parameter is implemented as the repetition time between the original and the delayed sample. Table 6.4 shows the measured delays. As a result, the exact implemented stepsize of Δt=104 ms is verified.

level	**decay**	**time**	Δt/ms
5	5	0	0
5	5	5	520
5	5	10	1040

Table 6.4.: Measurement of delay effect - *time* parameter

Decay Parameter

Acting as a factor within the *feedback line* (see figure 2.9) the *decay* parameter has a direct measurable influence at the fall time. The horizontal oscilloscope cursors capture the time span between the original sample and the last recordable delay. The three measured values depicted in Table 6.5 are in an expected linear relation.

level	**decay**	**time**	Δt/ms
5	0	5	0
5	5	5	1040
5	10	5	2080

Table 6.5.: Measurement of delay effect - *decay* parameter

Level Parameter

For the validation of the *level* parameter, the function generator is used for the production of rectangle pulses. That allows a more efficient detection of the actual peaks contained in the delayed signal. For a realistic guitar signal, a pulse length of 100 ms with 0.11 V_{Peak} is configurated. The Task of the *level* parameter is to control the total influence of the delay effect. The measurement (6.6) shows a direct impact of the parameter on the height of the first delayed peak. It can be observed a reduction of the subsequent delayed by halving the peaks.

level	**decay**	**time**	1stPeak/mV	2ndPeak/mV	3rdPeak/mV	4thPeak/mV
0	5	5	-	-	-	-
5	5	5	24	14	-	-
10	5	5	48	24	12	6

Table 6.6.: Measurement of delay effect - *level* parameter

Hearable Results

The delay effect tested with the electric guitar in combination with the amplifier leads to the desired echoing sound. The original clean sound gets enhanced by the delayed signal - adjustable by the three rotary encoders after the preferences of the guitarist. Audio recordings of the delay effect are placed on the attached CD (A.1) as WAV files.

6.2.2. Distortion

The following tests contain measurements in the time and frequency-domain to validate the implemented distortion effect. Due to the implementation of only one adjustable parameter, all parameter values from 1 to 10 are tested. The total set of measurement-screenshots are presented in the appendix (see section A.4).

Clipping Thresholds

According to a useful and playable distortion effect, the signal amplitude at every distortion stage is adjusted to the same level to achieve the same loudness. Therefore after the implemented clipping of the signal amplitude takes place, the signal amplitudes are adjusted. Figure 6.7 shows the results of the exemplary measurement for distortion stages 0, 5 and 10 to visualize the clipping and adjustment process.

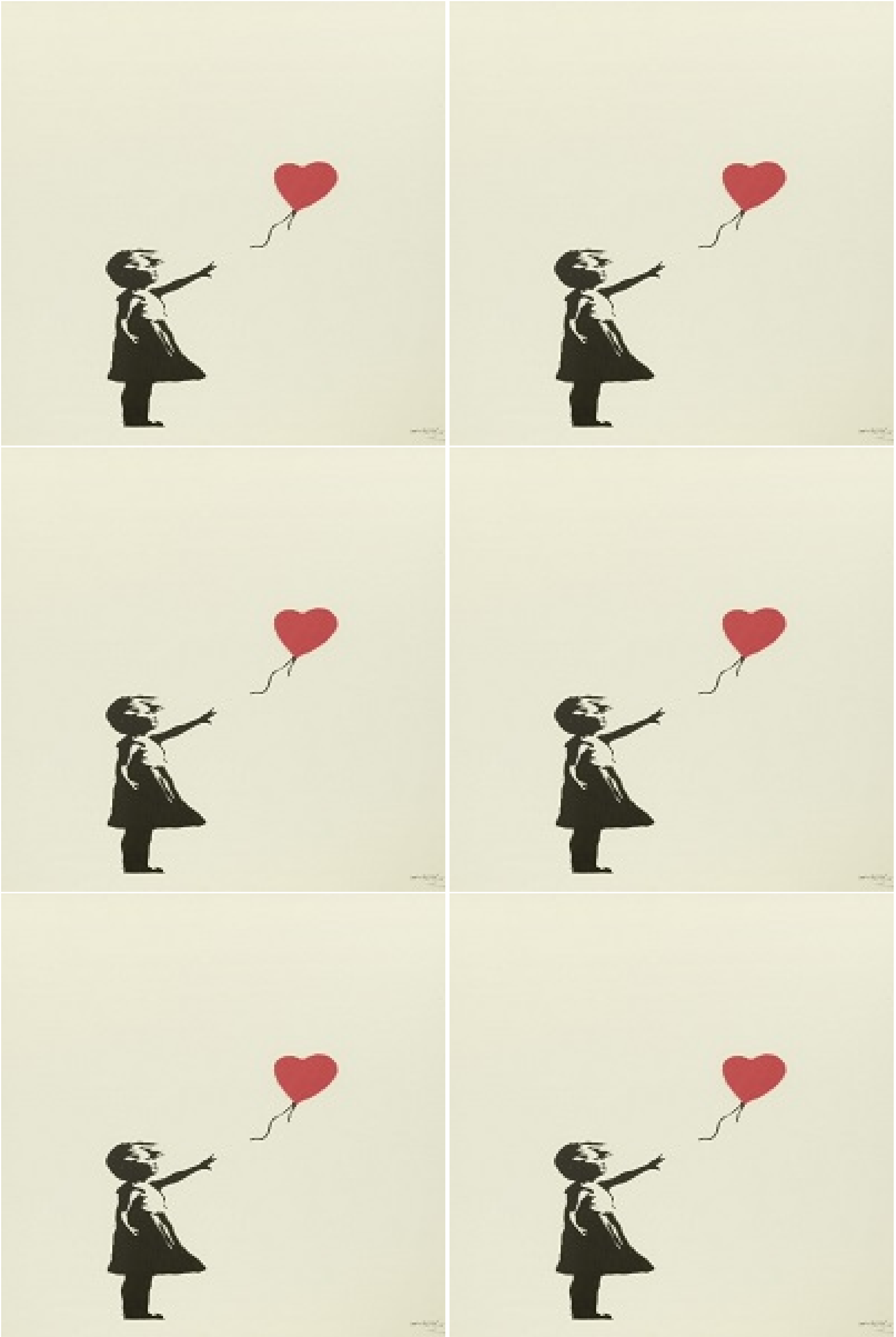

Figure 6.7.: Measurement of thresholds (left) and adjusted levels (right)

For the verification of the implemented clipping thresholds, the part that is responsible for the subsequent signal adjustment is commented out. In addition to that, the oscilloscope is directly connected to the *AUDIO_IN* and *AUDIO_OUT* of the HAT to gain unchanged values from the ADC and DAC. Table 6.7 depicts the measured clipping threshold in comparison to the implemented values.
As a result, an increasing deviation of ΔU is observed. The most probable cause is the unfavourable choice of a constant scaling of the y-axis during all measurements. In spite of the inaccuracies caused by the incorrect measuring method, the thresholds could be verified.

ParameterValue	0	1	2	3	4	5	6	7	8	9	10
$U_{\mathrm{Thres,implemented}}$ / mV	400	367	299	247	203	165	130	98	68	40	13
$U_{\mathrm{Thres,measured}}$ / mV	400	392	328	272	232	200	160	136	104	80	56
ΔU / mV	0	25	29	25	29	35	30	38	36	40	43

Table 6.7.: Distortion threshold measurements

THD+N

For a more quantitative valuation of the resulting distorted sound, the signal needs to be examined on resulting harmonics. The chosen input voltage of $100\,\mathrm{mV_{Peak}}$ is comparable with a guitar output.
Figure 6.8 and 6.9 show the two extrema of the implemented distortion. Within the FFT plot, all harmonics are labelled.

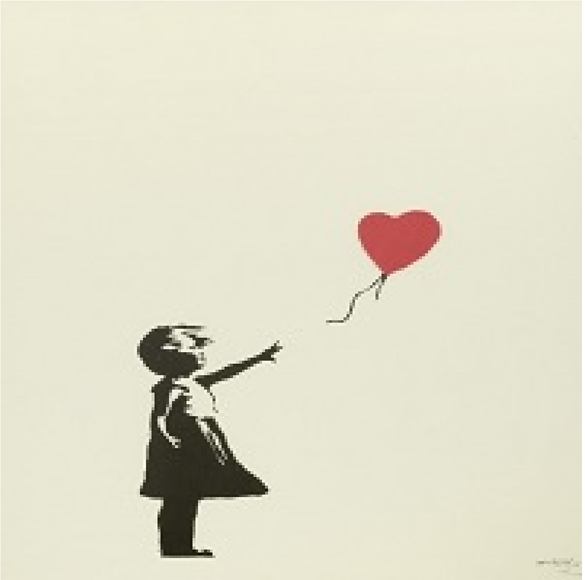

Figure 6.8.: FFT - ParameterValue 0

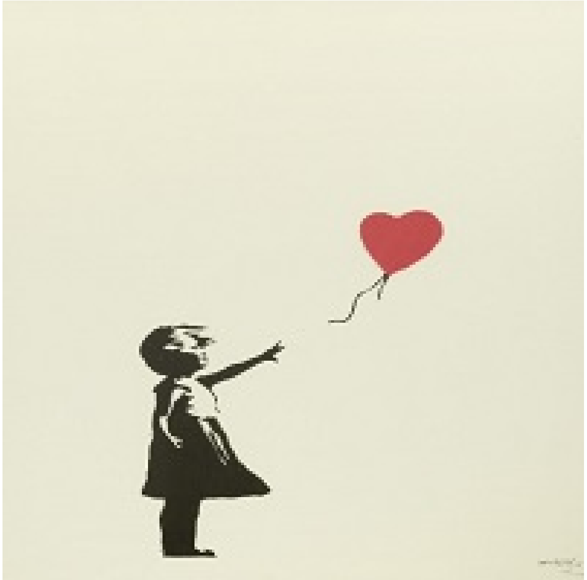

Figure 6.9.: FFT - ParameterValue 10

Since the original waveform of the input signal is changed by hard clipping, the effect performs a non-linear distortion. The implemented symmetrical clipping affects the positive and negative amplitude equally. As a consequence, the cubical parts of the harmonics are heightened. Thus the odd integer multiples of the fundamental frequency.

The measured THD+N values for every distortion stage are shown in table 6.8. For Parameter value 0 a THD+N of -78,38 dB results. This value was also identified by Sebastian Albers [**?** , p. 81]. Based on that, the THD+N increases according to the incremental distortion stages.

ParameterValue	0	1	2	3	4	5
THD+N / dB	-78,38	-18,88	-15,08	-12,94	-11,62	-10,5
THD+N / %	0,012	11,37	17,62	22,54	26,24	29,85
ParameterValue	6	7	8	9	10	
THD+N / dB	-9,76	-9,08	-8,53	-7,96	-7,66	
THD+N / %	32,5	35,16	37,45	39,99	41,4	

Table 6.8.: THD+N measurements of every distortion stage

Hearable Results

In the practical scenario, all distortion stages are tested. The division of the single stages offers the guitarist a variety of distorted sounds. The slightly distorted sound at parameterValue 1 is suitable for softer song-passages, while the distortion at parameterValue 10 is going a little over the top, from a subjective point of view. The default value at parameterValue 5 produces a full and saturated sound appropriate for most rock songs. In addition to that, the adjusted volume level for all distortion stages is similar to the subjective loudness of the other effects. Audio recordings of the distortion effect are placed on the attached CD (A.1) as well.

6.3. Maximum Latency

The measurement of the maximum latency includes all relevant hardware components in the total signal chain. Thus, it is the time span measured between *INSTRUMENT_IN* and *INSTRUMENT_OUT*. In the testing phase, all guitar effects lead to the same result. That is reasonable because all implemented effects are forwarding one input sample to the output as a first step. As a result, a total signal latency of Δt=1.75 ms is achieved (6.10).
That is a very satisfying result, comparable with the physical delay provoked of an amplifier distance of Δd= 0,6 m calculated at the assumption of dry air at 20° celsius (see equation 6.1).

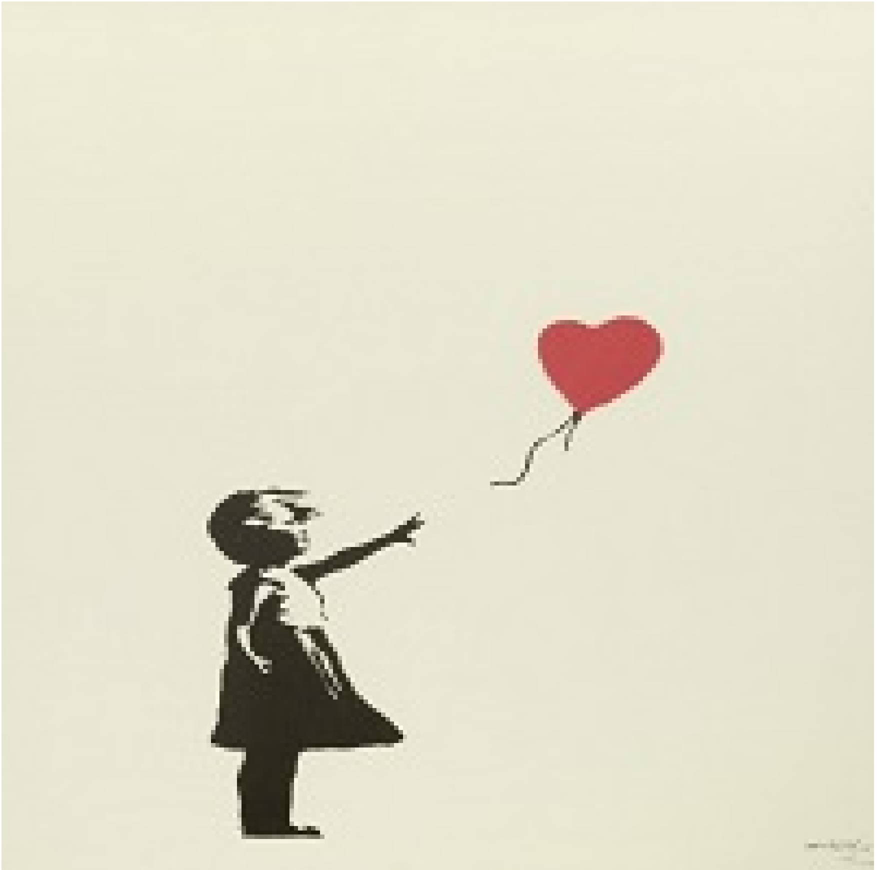

Figure 6.10.: Maximum latency

$$\Delta d = c \cdot \Delta t = 343 \frac{\mathrm{m}}{\mathrm{s}} \cdot 1,75\,\mathrm{ms} = 0,6\,\mathrm{m} \tag{6.1}$$

7. Summary

In this thesis, the goal of a fully working multi-effect unit is accomplished. A Raspberry Pi used in combination with the Audio-HAT performs the task of digital signal processing.
Three exemplary guitar effects are implemented: A clean effect for an unchanged guitar sound, a delay effect with three adjustable parameters for a variety of echoing sounds and a distortion effect based on hard clipping. All guitar effects are verified on the basis of measurable and hearable results.

Due to the development of a preamp module, the guitar signal is adjusted in an appropriate way. Hence, the input signal is amplified for an optimal interconnection with the Audio-HAT providing a THD+N of 0.006 %. The other way around the signal is attenuated for further processing by a guitar amplifier.

The user interface allows the guitar player the total system control. Consisting of a bright LCD-Text display, rotary encoders and buttons the interface is suitable for stage usage.

Outlook

The concept of the effect unit offers a vast amount of possible extensions.
On the hardware side, the unit can be equipped with a power amplifier, due to the generously designed power switching supply. The back panel leaves enough space for speaker connectors. For the interconnection with other audio devices like HIFI-Amplifiers or PC-Soundcards, a Line-Out via RCA connector sockets is reasonable. Suitable nominal voltages on consumer audio line level and impedances are available. Furthermore, the SW contact of the rotary encoders (high signal when the encoder is pressed) could be used for extensions of the user controls.

The implemented software can be extended by an arbitrary number of guitar effects. For further developments the source-code is commented at the relevant lines, highlighting good entry points. To provide a greater user-friendliness, it is reasonable to extend the user menu by displayed parameter names. As a final step, the combination of different effects might be a good idea to achieve an effect chain.

A. Appendix

A.1. Attached CD

Additional attachment is stored on a CD.
The CD is deposited with examiner Prof. Dr. Robert Hess and contains the following material:

- PDF version of Bachelorthesis
- EAGLE project files of all PCB layouts and schematics
- Collection of all implemented source codes
- Audio recordings of clean, delay and distortion effects

A.2. Total Schematic of Effect Unit

Figure A.1 shows the total schematic of the device. The detailed pin assignment of the Audio-HAT and Raspberry Pi is documented in the bachelor thesis of Sebastian Albers [**?** , p. 99].

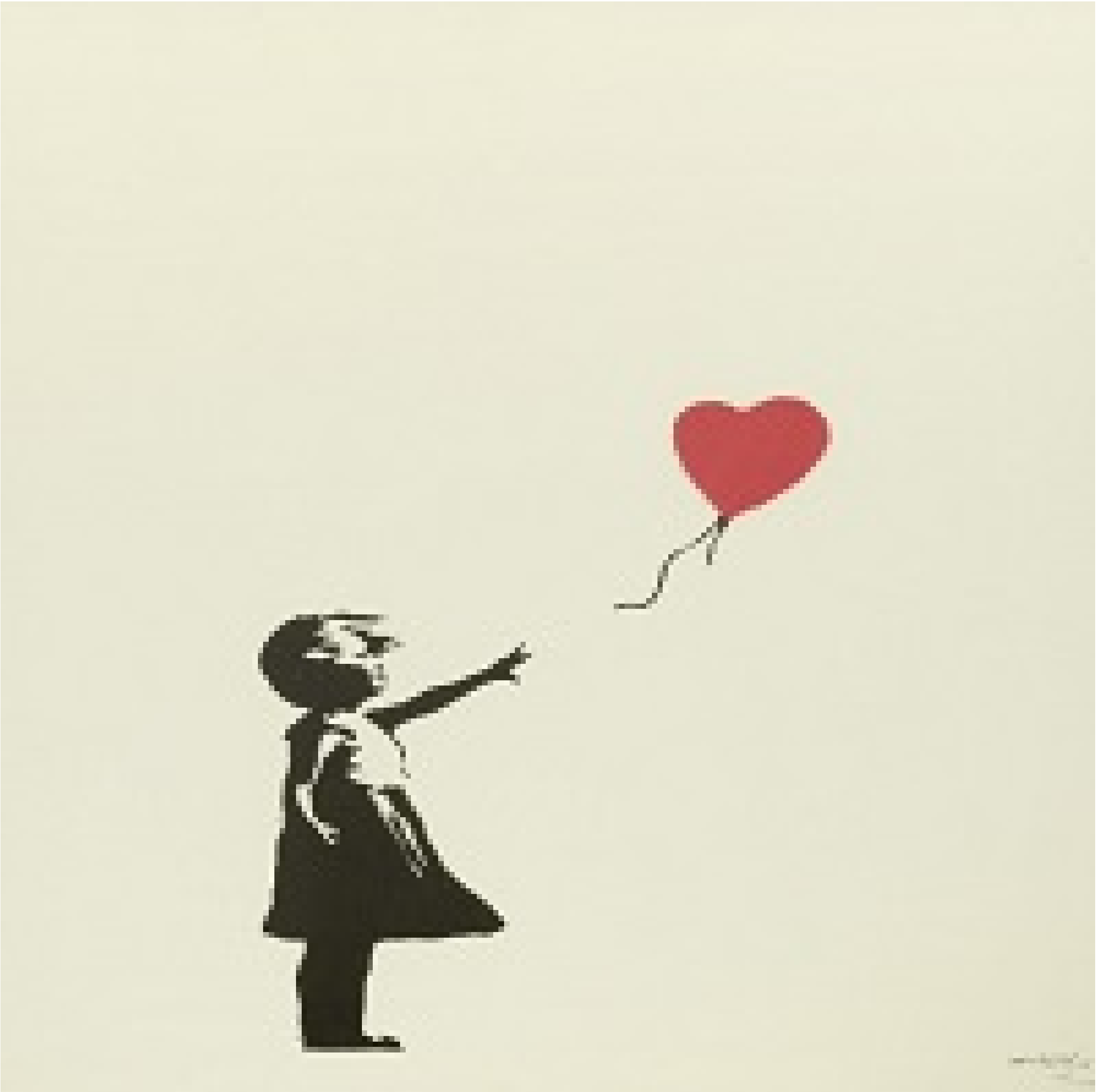

Figure A.1.: Schematic of total effect unit

A.3. Delay Effect Measurements

Influence of Time Parameter

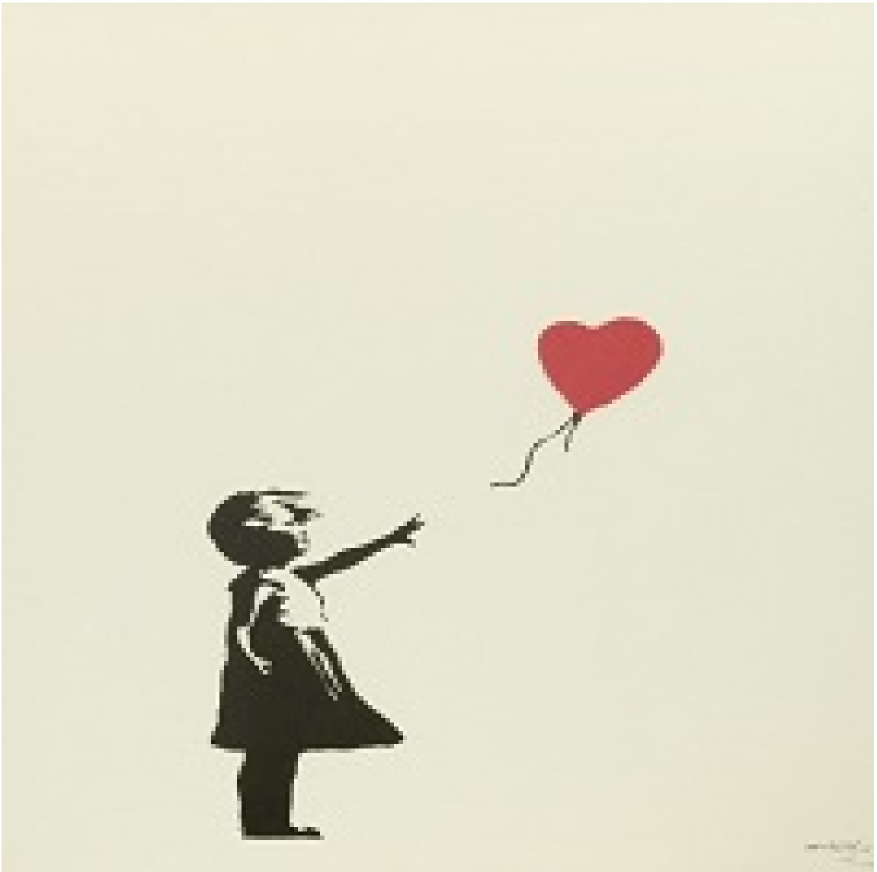

Figure A.2.: Delay-measurement with *level=5*; *decay=5*; *time=0*

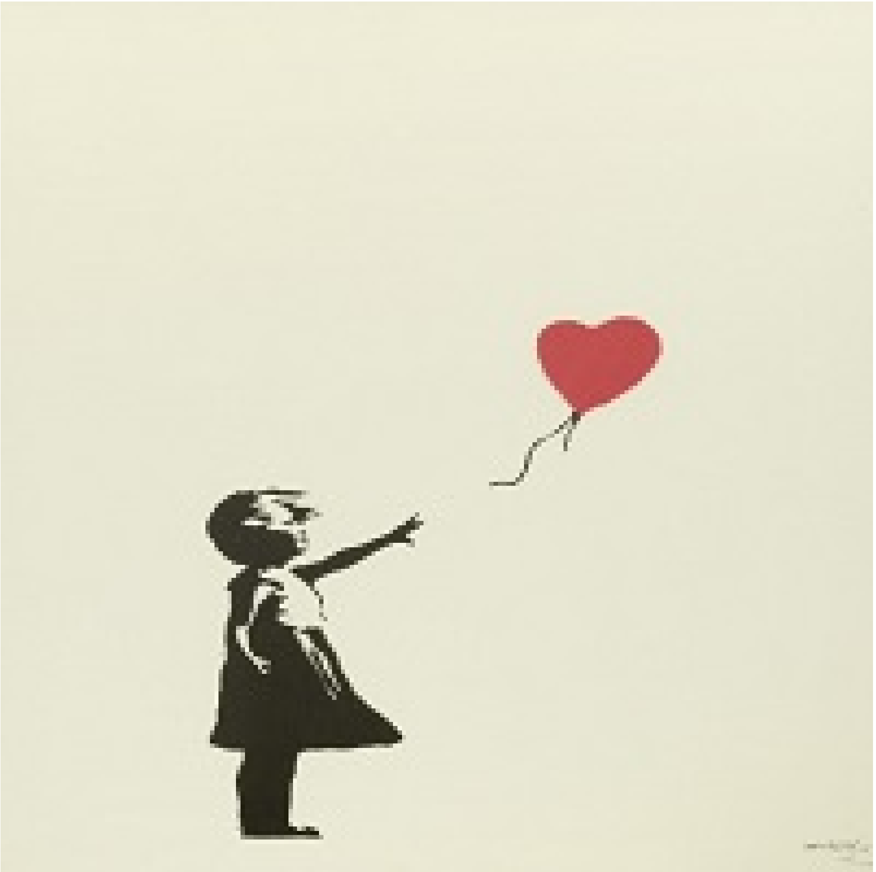

Figure A.3.: Delay-measurement with *level=5*; *decay=5*; *time=5*

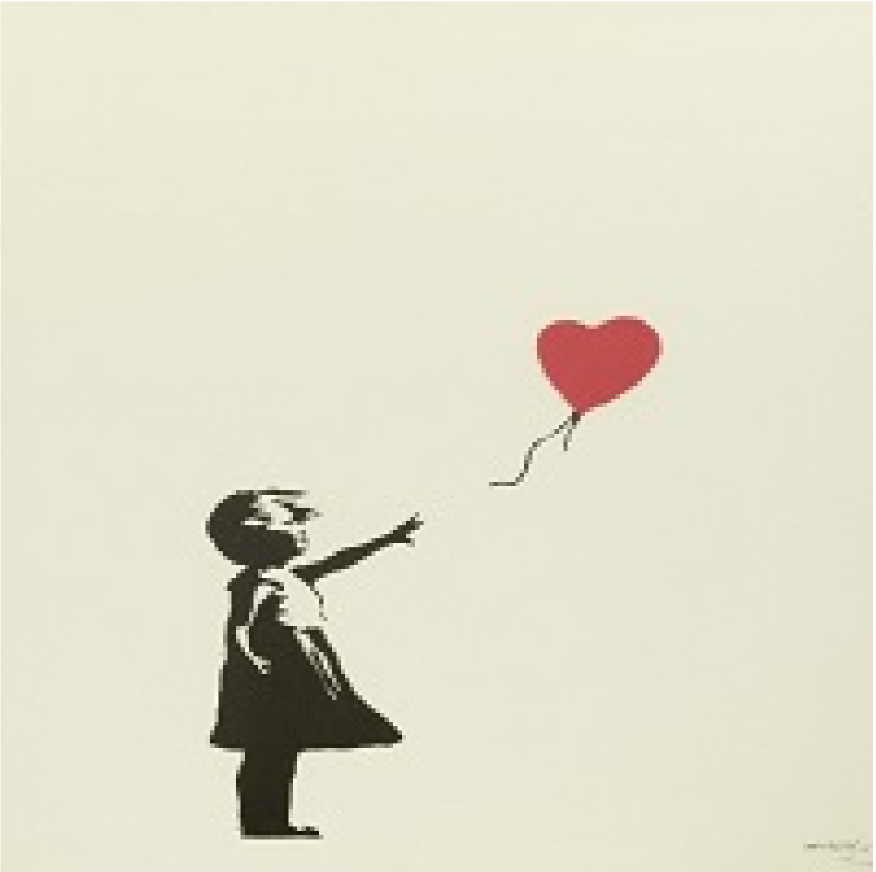

Figure A.4.: Delay-measurement with *level=5*; *decay=5*; *time=10*

Influence of Decay Parameter

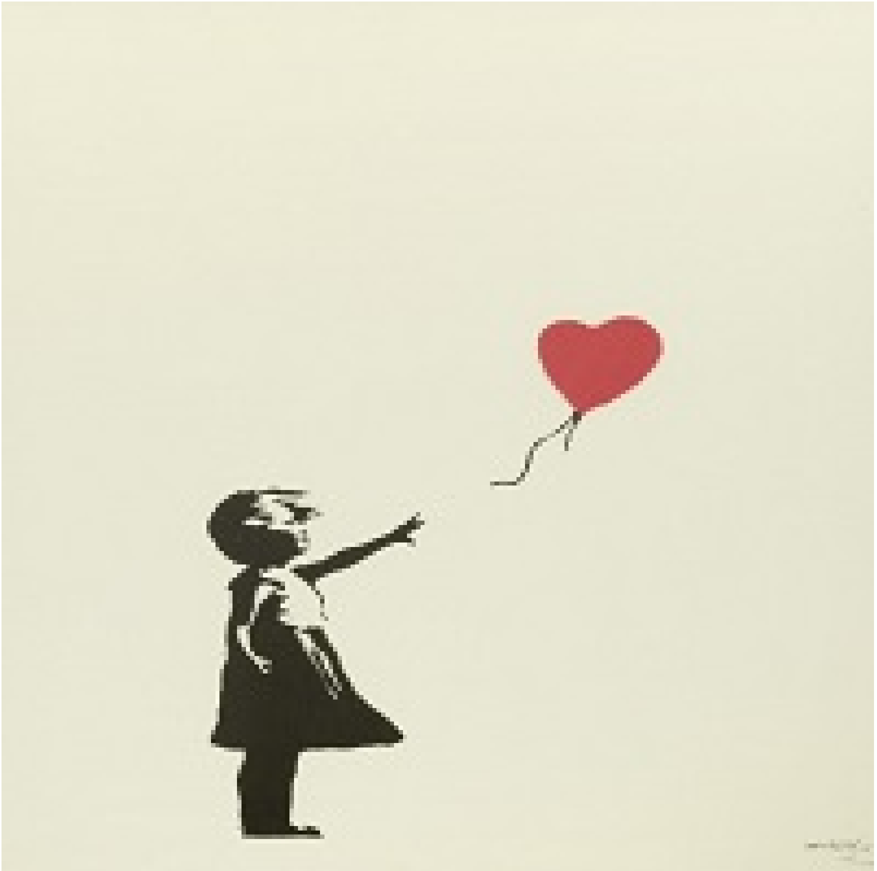

Figure A.5.: Delay-measurement with *level=5*; *decay=0*; *time=5*

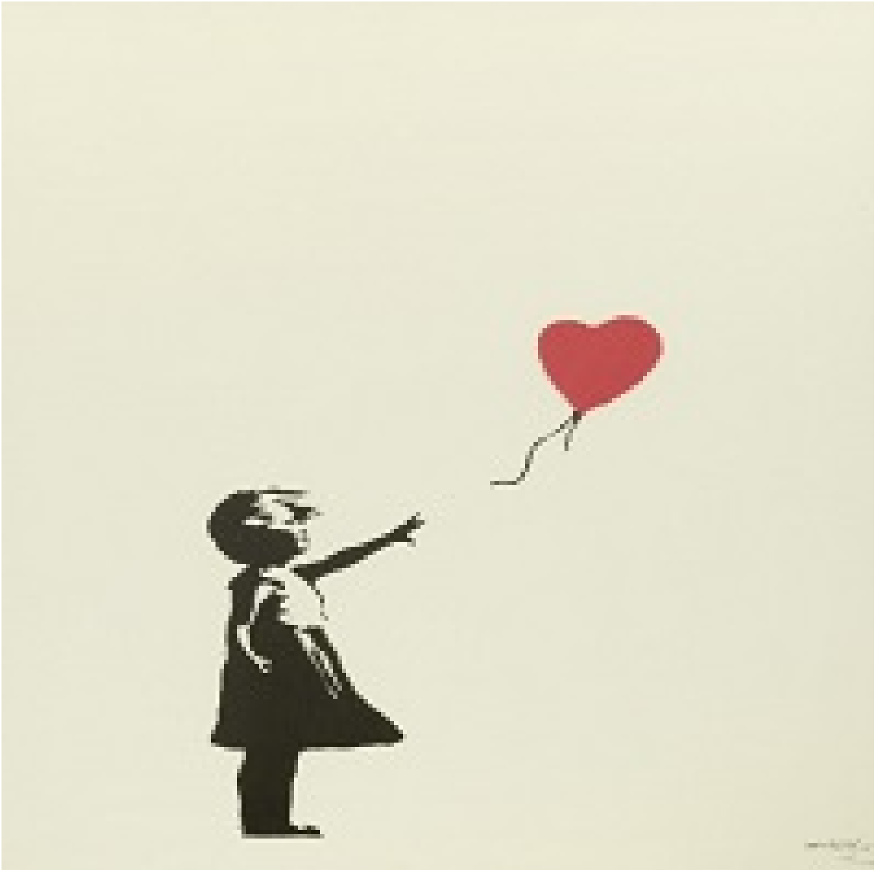

Figure A.6.: Delay-measurement with *level=5*; *decay=5*; *time=5*

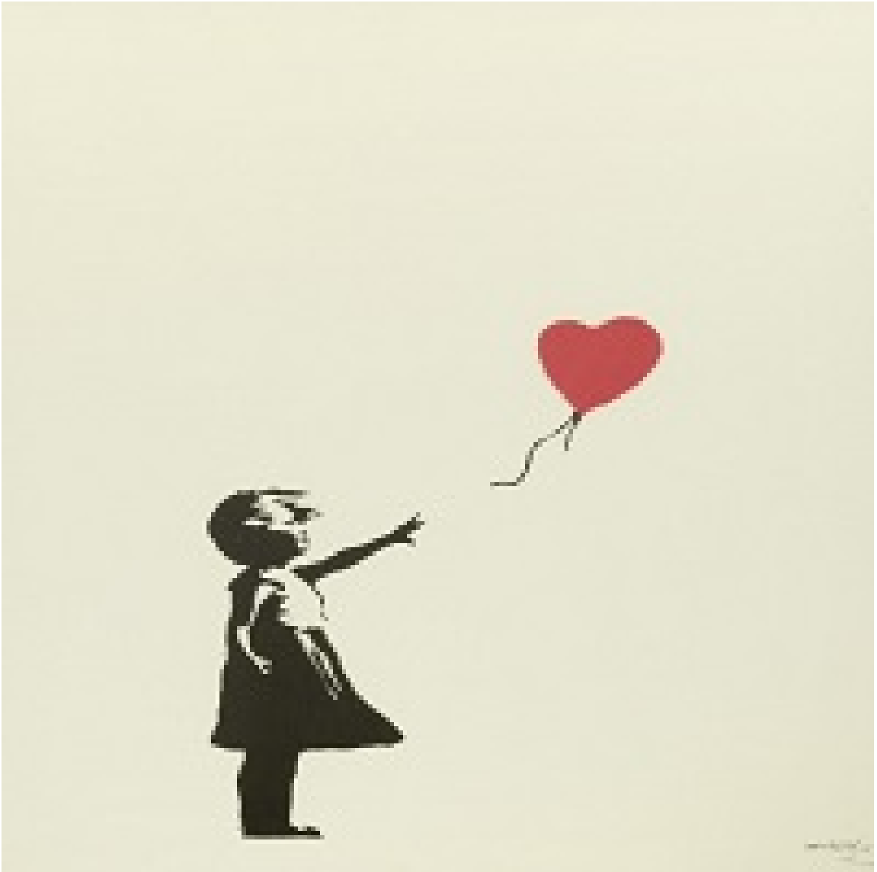

Figure A.7.: Delay-measurement with *level=5*; *decay=10*; *time=5*

Influence of Level Parameter

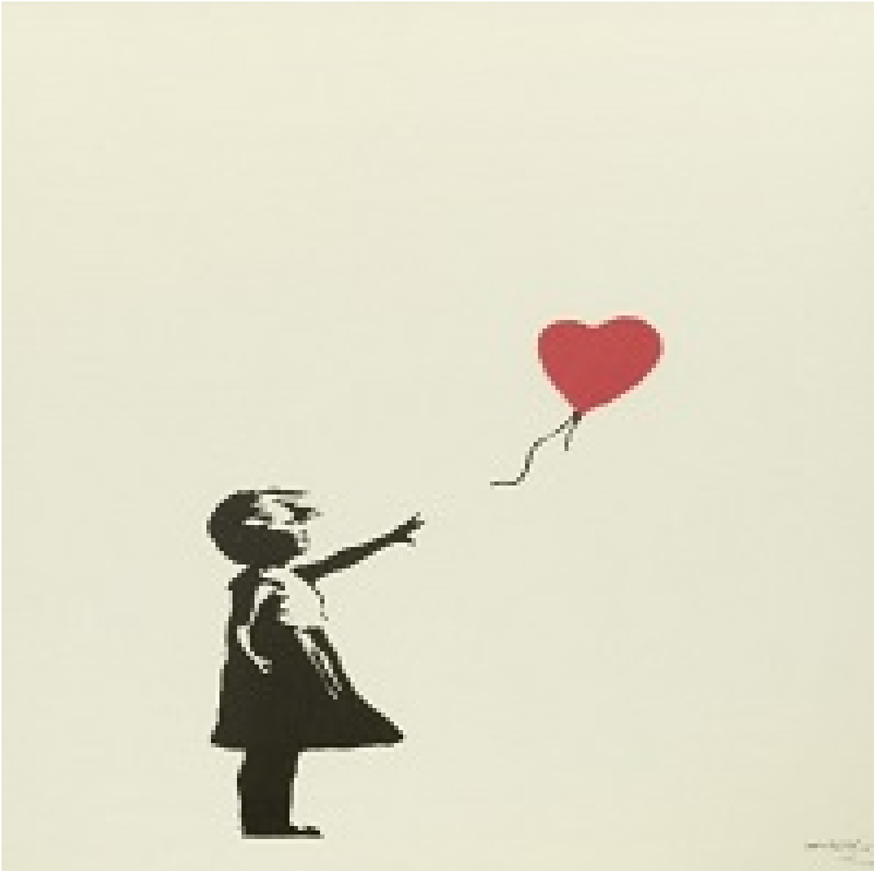

Figure A.8.: Delay-measurement with *level=0*; *decay=5*; *time=5*

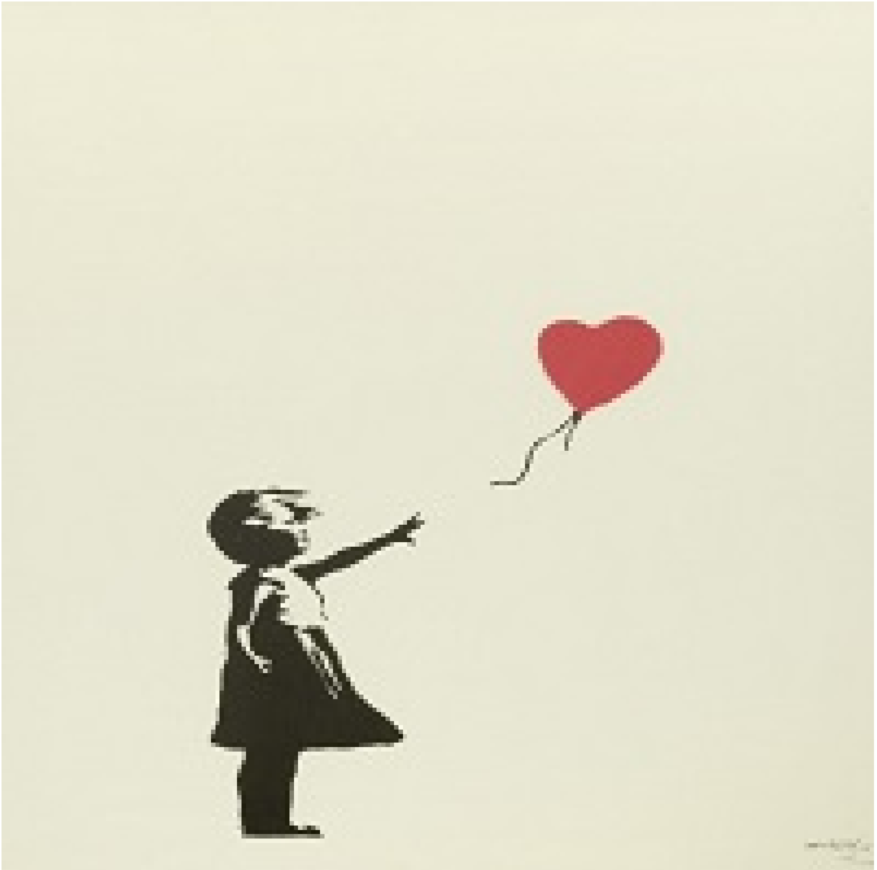

Figure A.9.: Delay-measurement with *level=5*; *decay=5*; *time=5*

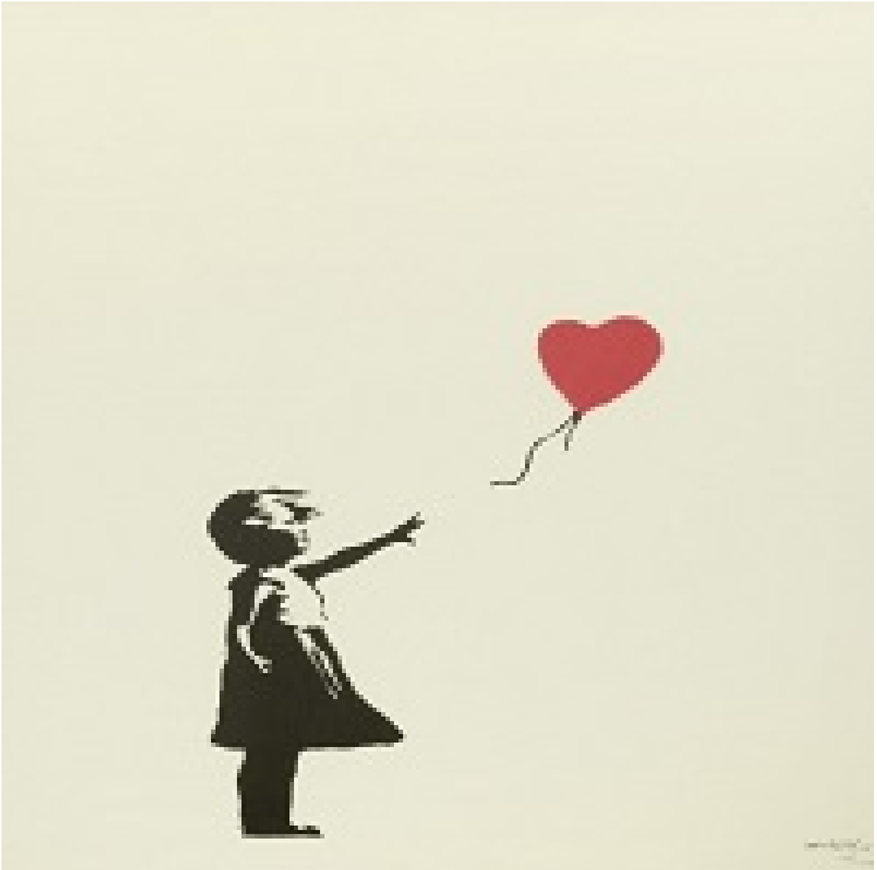

Figure A.10.: Delay-measurement(1) with *level=10*; *decay=5*; *time=5*

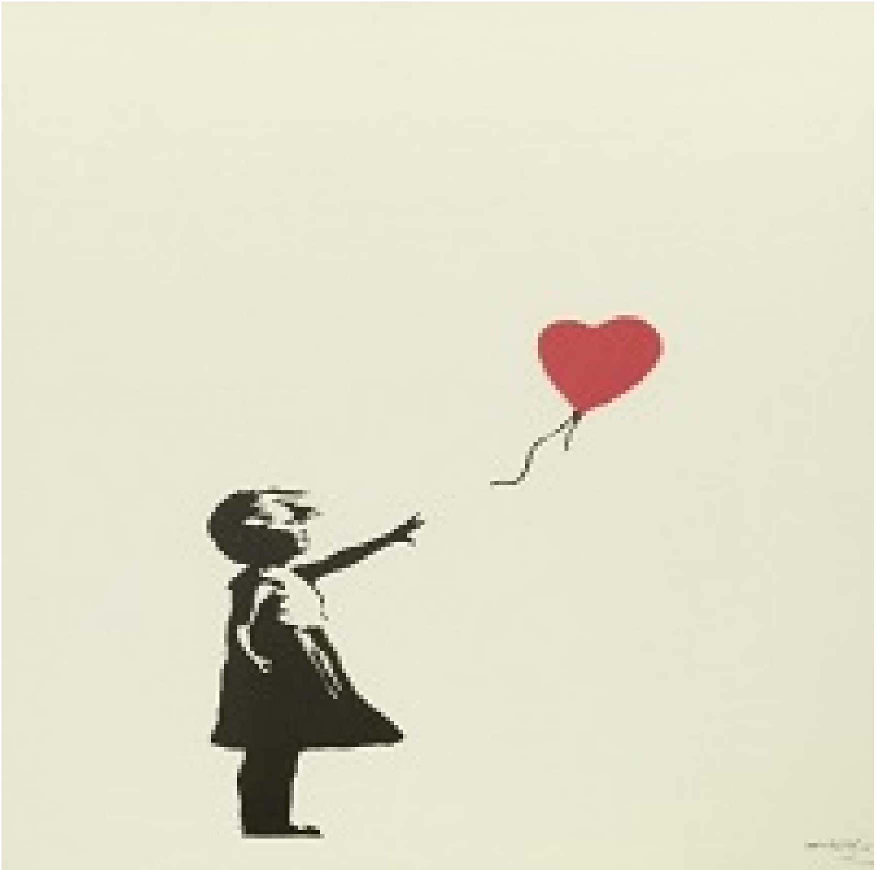

Figure A.11.: Delay-measurement(2) with *level=10*; *decay=5*; *time=5*

A.4. Distortion Effect Measurements

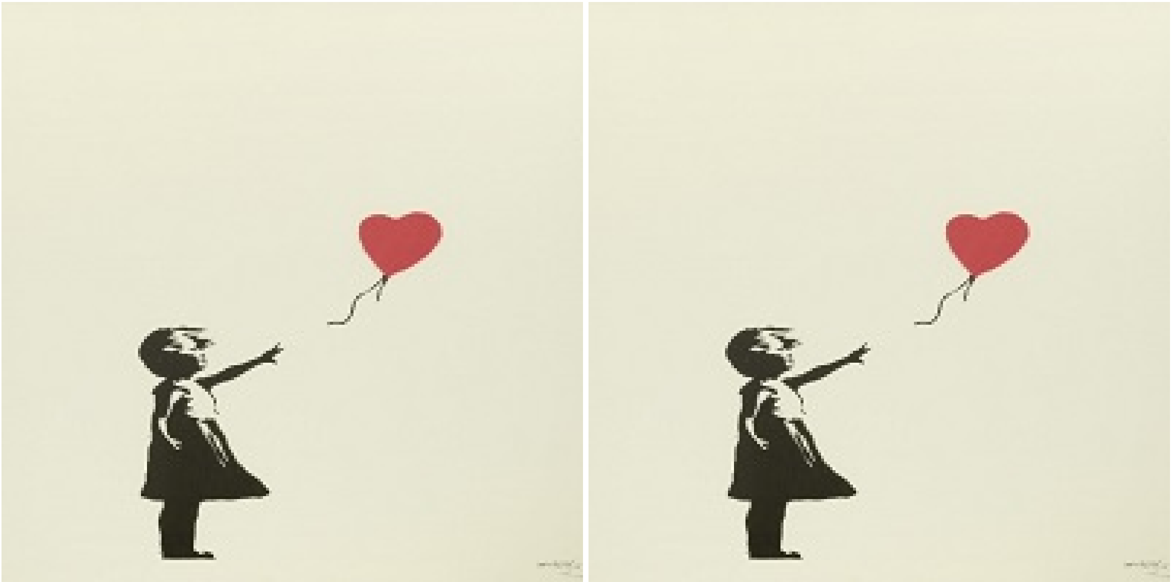

Figure A.12.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 0.

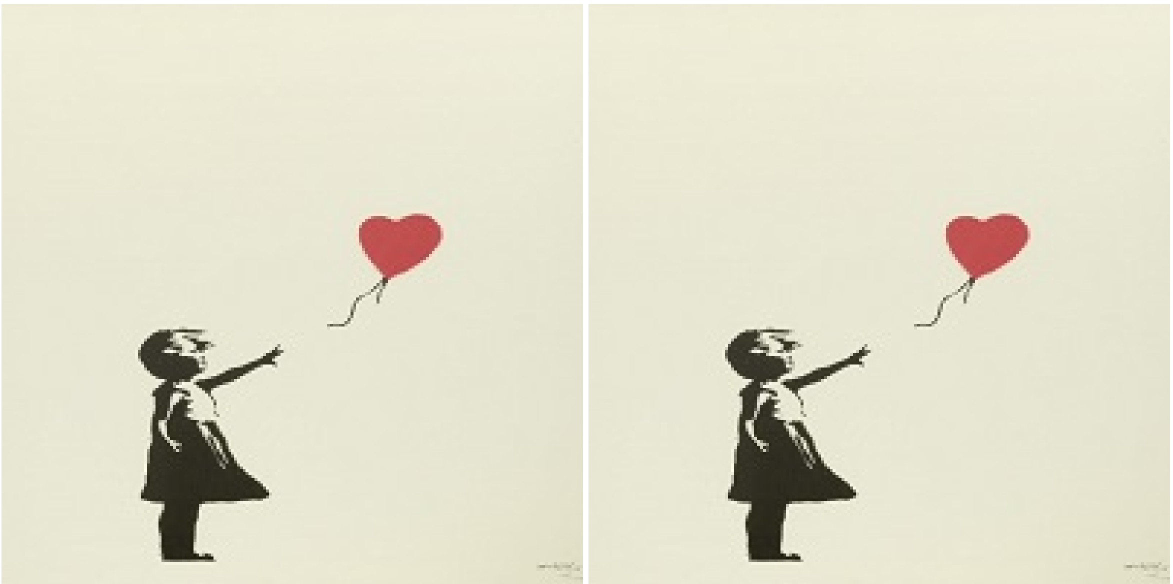

Figure A.13.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 1.

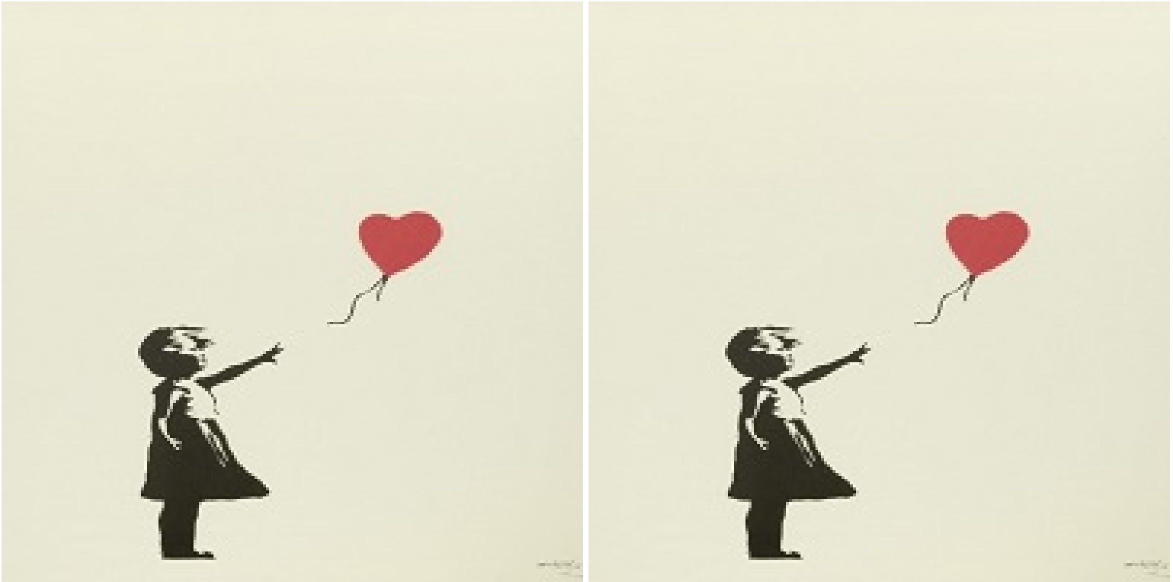

Figure A.14.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 2.

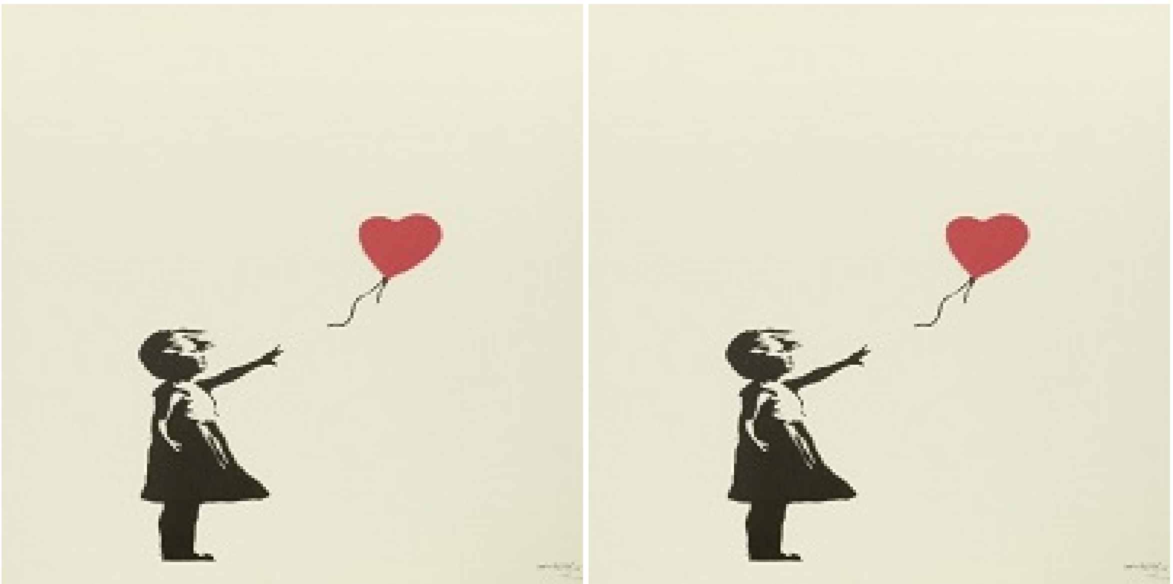

Figure A.15.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 3.

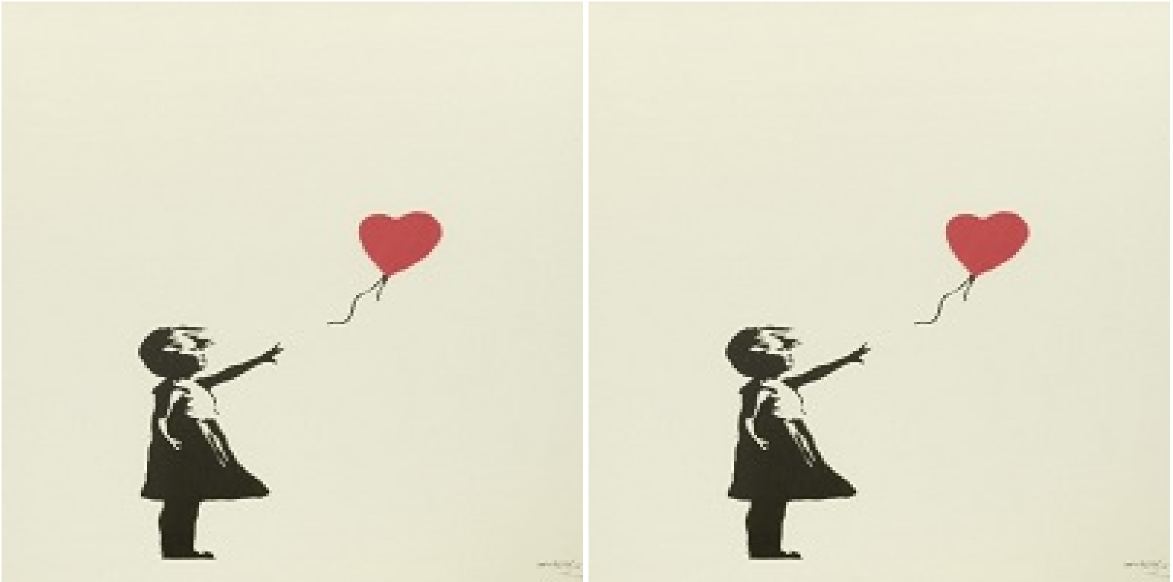

Figure A.16.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 4.

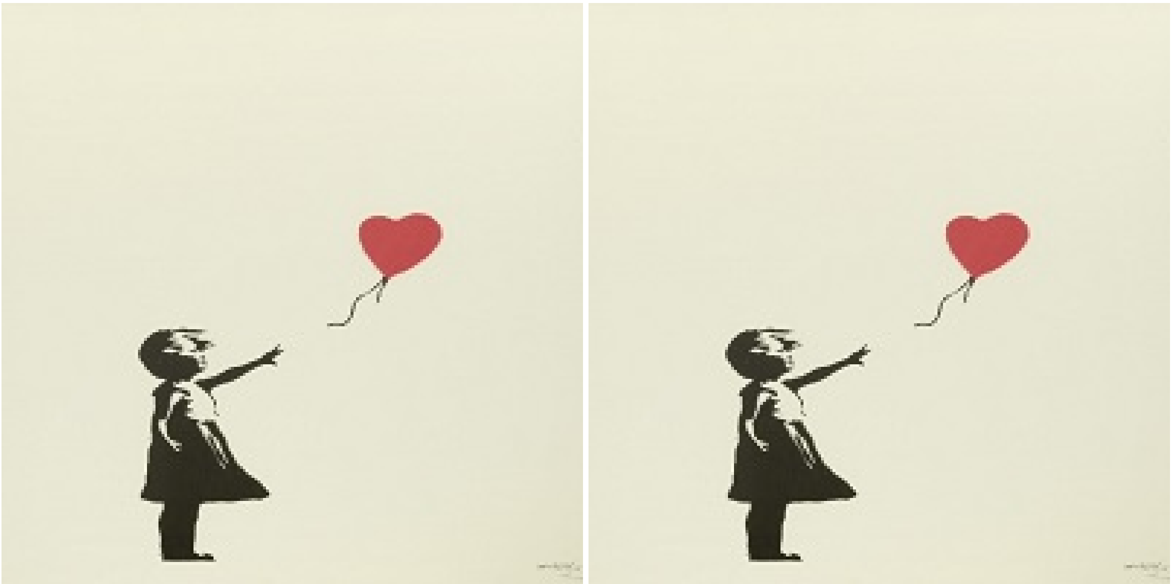

Figure A.17.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 5.

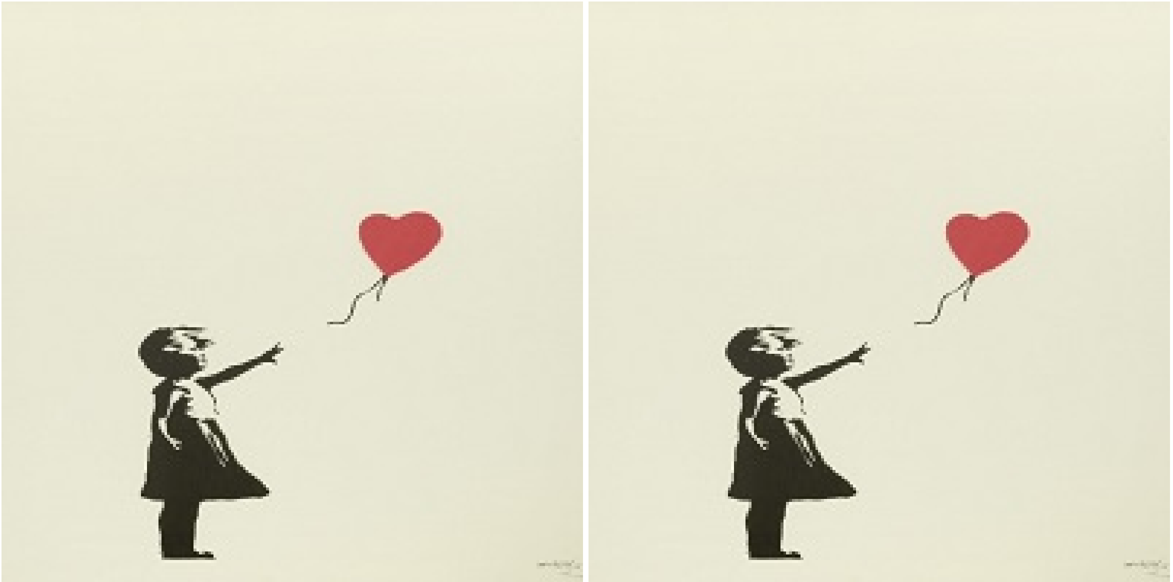

Figure A.18.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 6.

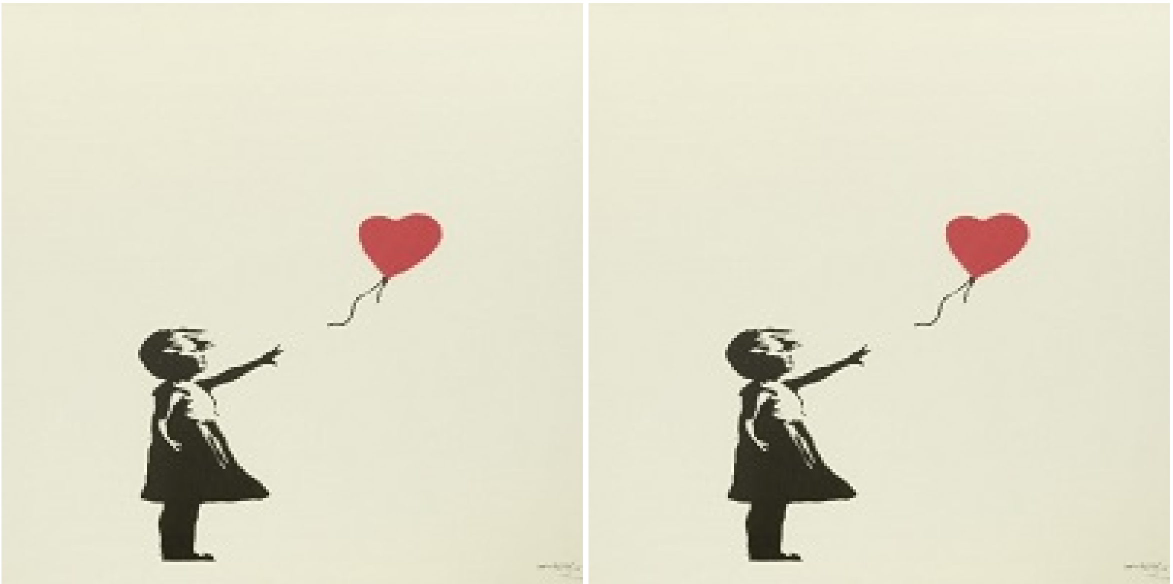

Figure A.19.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 7.

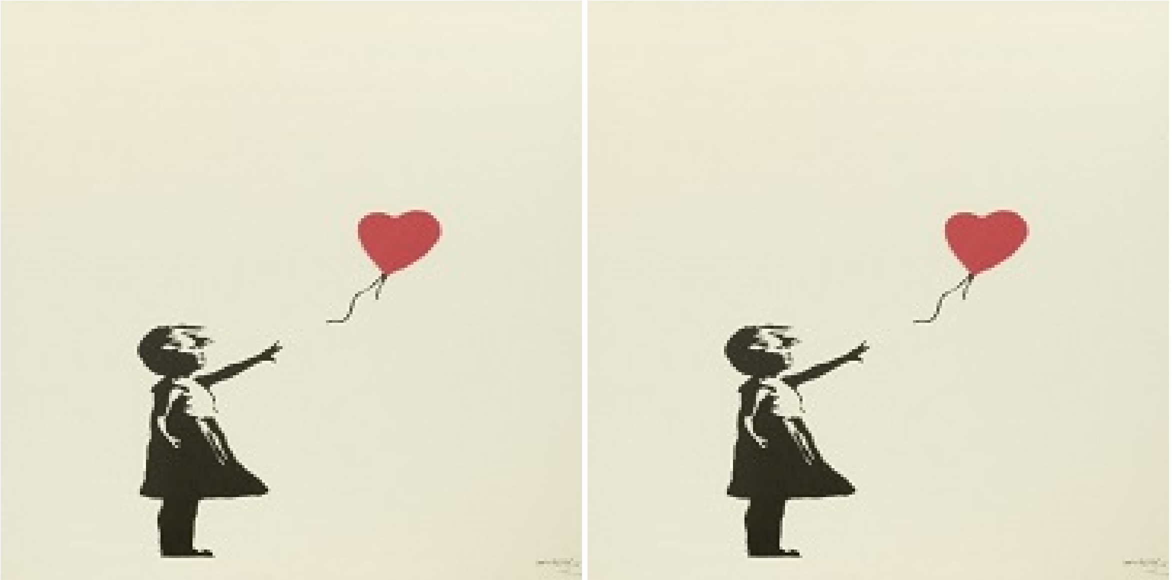

Figure A.20.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 8.

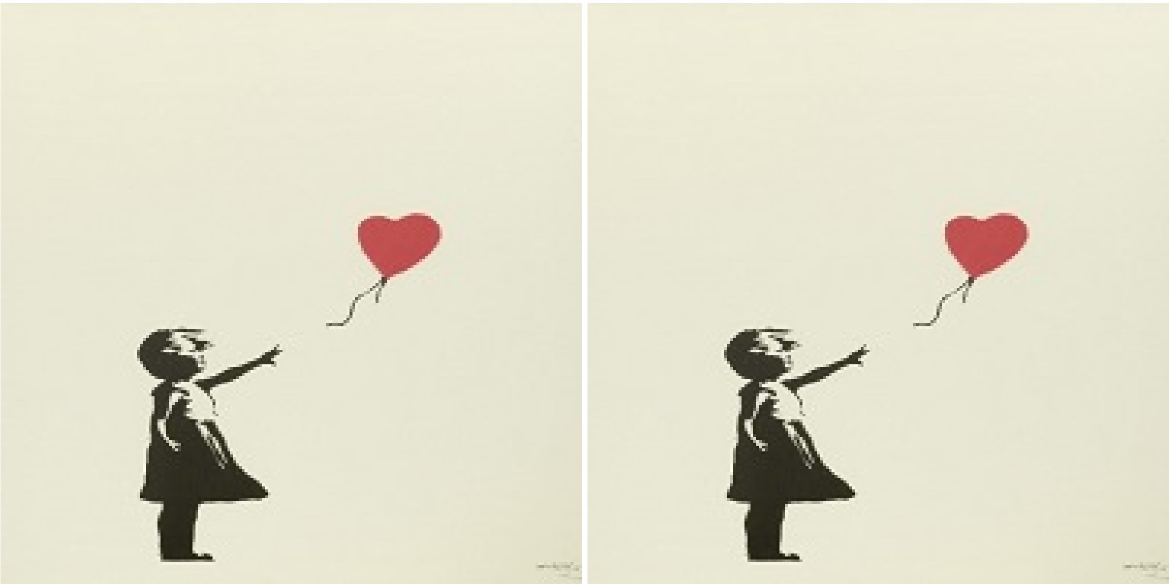

Figure A.21.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 9.

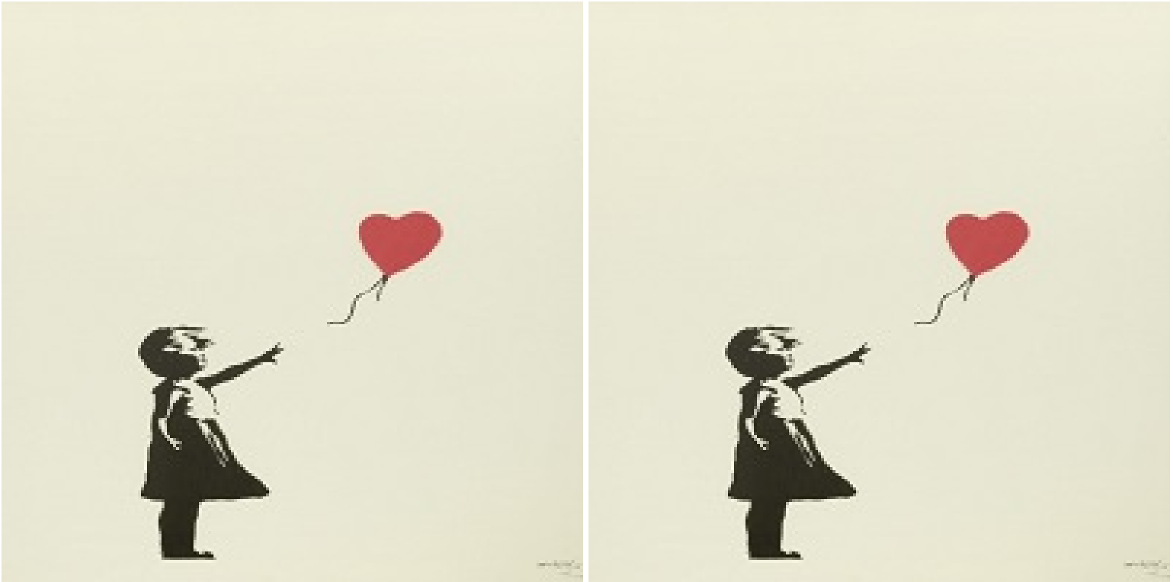

Figure A.22.: Measurement of threshold(left) and adjusted level(right) - ParameterValue 10.

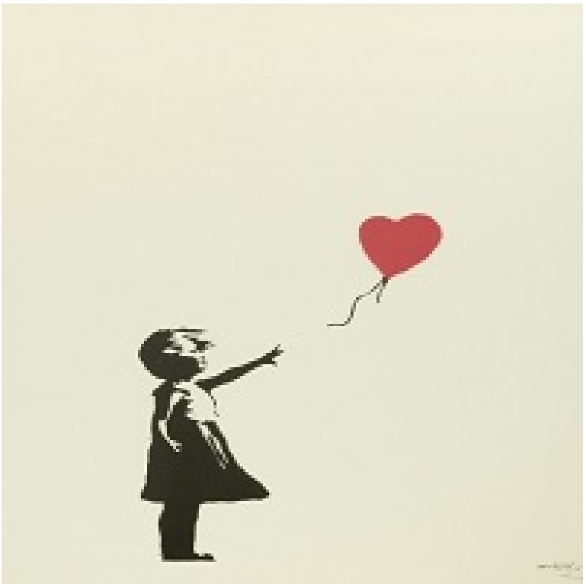

Figure A.23.: FFT - ParameterValue 0

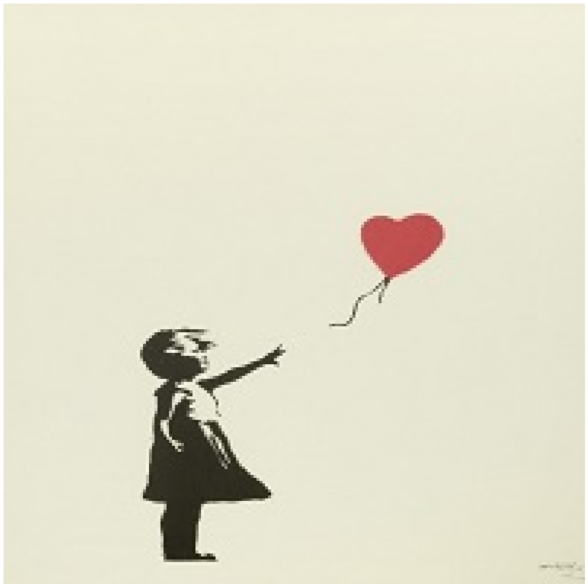

Figure A.24.: FFT - ParameterValue 1

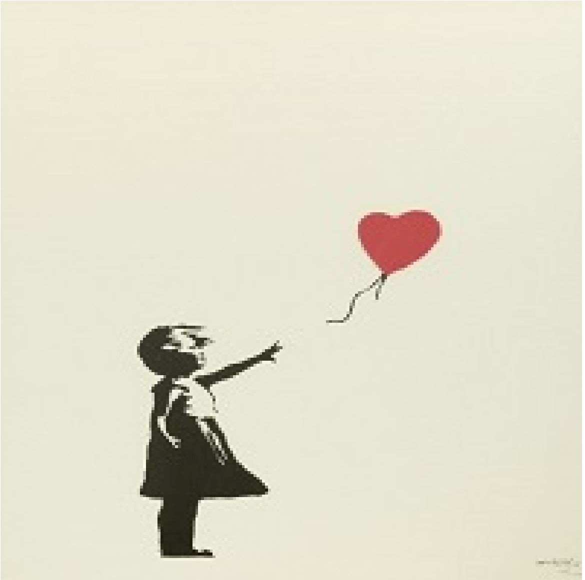

Figure A.25.: FFT - ParameterValue 2

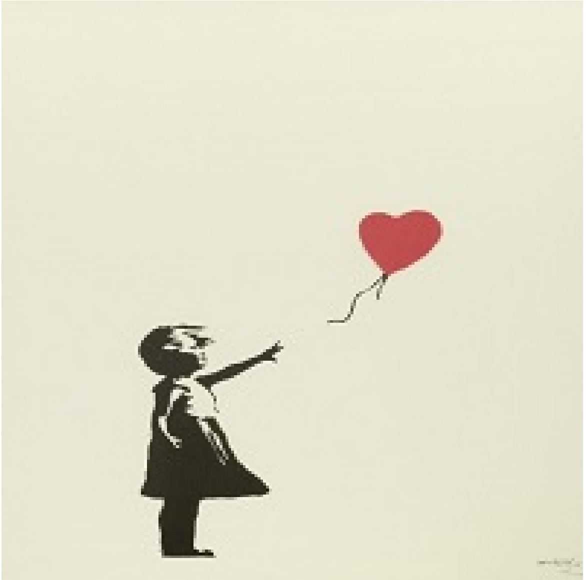

Figure A.26.: FFT - ParameterValue 3

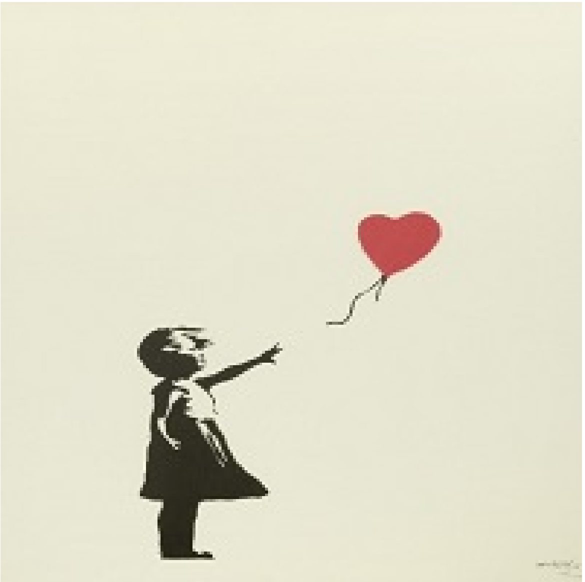

Figure A.27.: FFT - ParameterValue 4

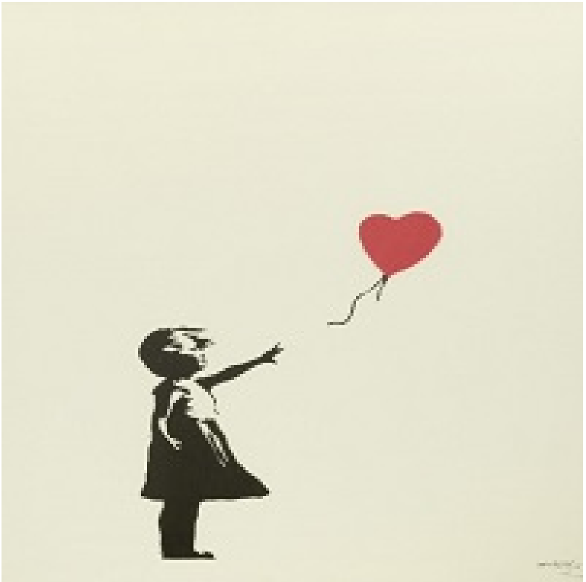

Figure A.28.: FFT - ParameterValue 5

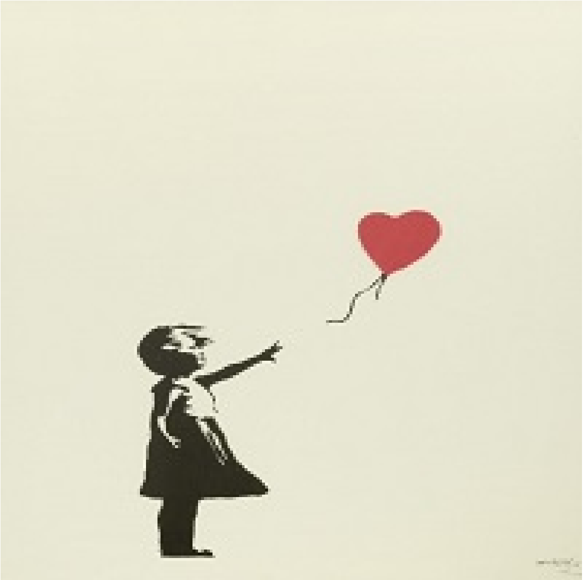

Figure A.29.: FFT - ParameterValue 6

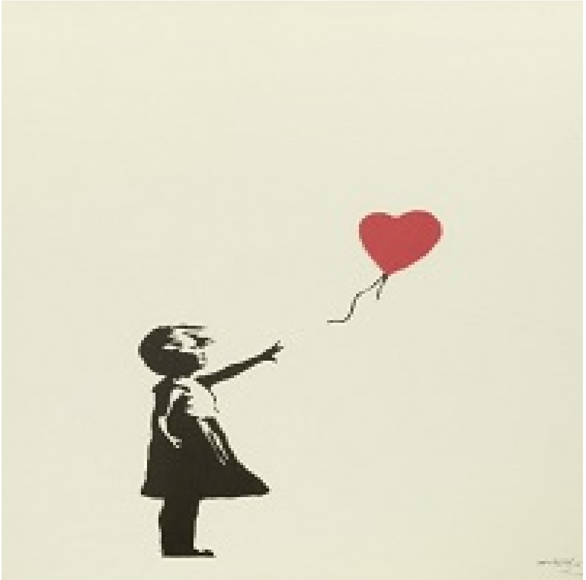

Figure A.30.: FFT - ParameterValue 7

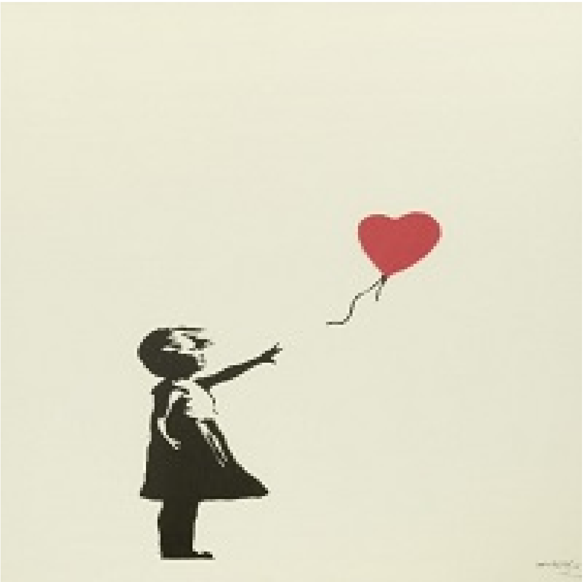

Figure A.31.: FFT - ParameterValue 8

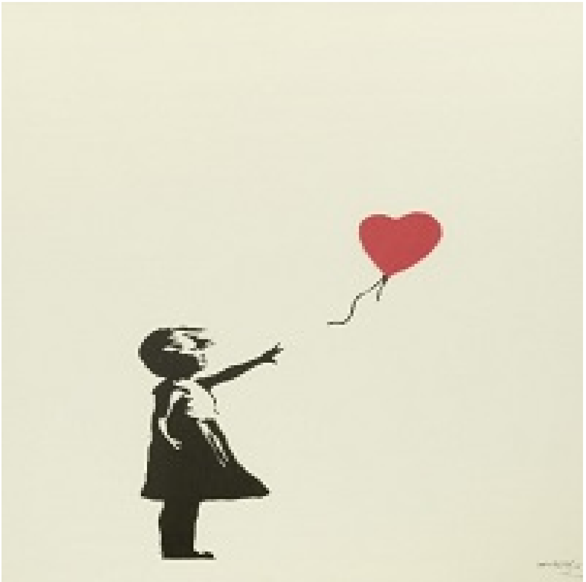

Figure A.32.: FFT - ParameterValue 9

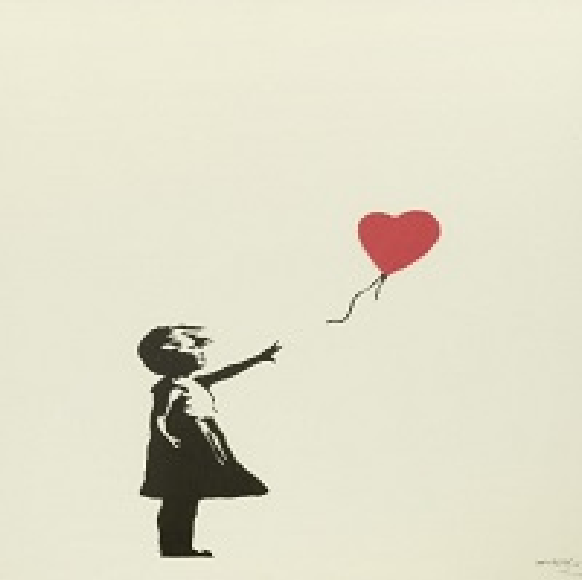

Figure A.33.: FFT - ParameterValue 10

Declaration

I declare within the meaning of section 25(4) of the Ex-amination and Study Regulations of the International De-gree Course Information Engineering that: this Bachelor report has been completed by myself inde-pendently without outside help and only the defined sources and study aids were used. Sections that reflect the thoughts or works of others are made known through the definition of sources.

Hamburg, May 6, 2019

City, Date

sign