Deckblatt einer

w i s s e n s c h a f t l i c h e n Bachelorarbeit

Vor- und Familienname	Matrikelnummer
Benjamin Weis	0630744
Studienrichtung	Studienkennzahl
Elektrotechnik-Toningenieur	F 033 213

Thema der Arbeit:

Entwurf eines Moduls zur Distanzsimulation einer Schallquelle
in der grafischen Programmiersprache Pure Data

Angefertigt in der Lehrveranstaltung: Musikinformatik 01
(Name der Lehrveranstaltung)

Vorgelegt am: 29.06.2011
(Datum)

Beurteilt durch: Ao.Univ.Prof. Dipl.-Ing. Winfried Ritsch
(LeiterIn der Lehrveranstaltung)

Benjamin Weis
(Name in Blockbuchstaben)

06300744
(Matrikelnummer)

Erklärung

Hiermit bestätige ich, dass mir der *Leitfaden für schriftliche Arbeiten an der KUG* bekannt ist und ich diese Richtlinien eingehalten habe.

Graz, den 29.06.2011

Unterschrift der Verfasserin / des Verfassers

Entwurf eines Moduls zur Distanzsimulation einer Schallquelle in der grafischen Programmiersprache Pure Data

Benjamin Weis

25. Juni 2011

Inhaltsverzeichnis

1 Einleitung

Ziel dieser Arbeit ist, ein Programm zu entwerfen, mit dem ein Schallereignis in der Tiefe positioniert werden kann. Weiters kann diesem Ereignis noch ein Doppler-Effekt hinzugefügt werden. Der Schall wird an einer ideal reflektierenden Wand zurückgeworfen und tritt verzögert zum ursprünglichen Ereignis am Ort des Hörers ein. Zusätzlich können die Quelle oder die reflektierende Wand bewegt werden, welches eine Frequenzverschiebung durch den Doppler-Effekt hervorruft. Im Rahmen dieser Arbeit wird die für das Programm notwendige und zugrunde liegende Theorie erläutert sowie der Aufbau und die Programmierung erläutert.

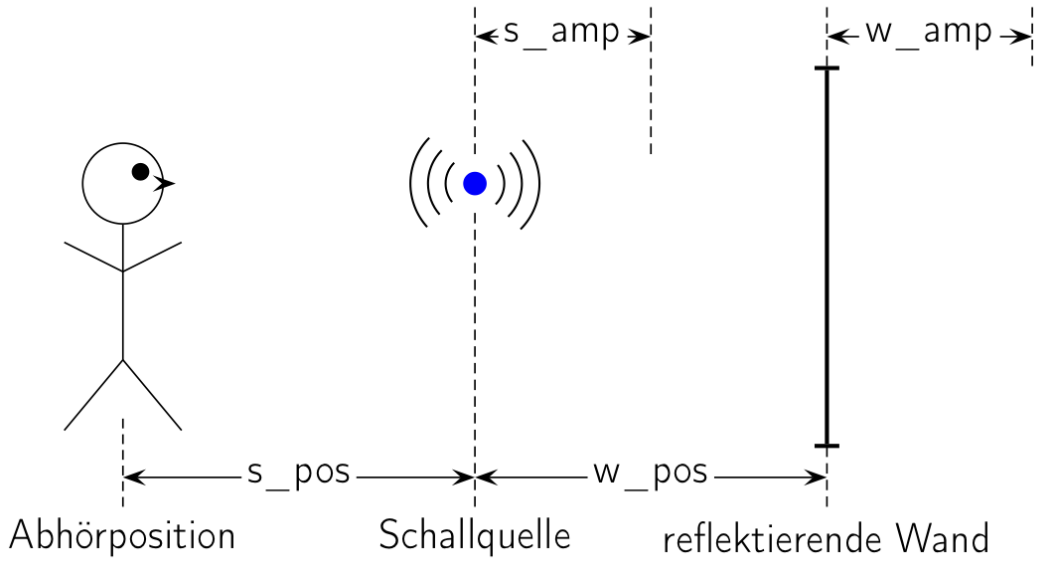


Abbildung 1.1: *Systemskizze*

1.1 Wahrnehmung

Frei nach [Dic97] und [MF10]: Im Gegensatz zu binauralen Signalunterschieden, welche die Richtungsortung von Schallereignissen ermöglichen, sind bei der Entfernungswahrnehmung die eigene Vertrautheit und Erfahrung mit dem Schallereignis maßgeblich. Am Weg von der Quelle zum Ohr ändert sich nicht nur die Schallintensität.

1.1.1 Entfernung

Bei allseitig gleichmäßiger Abstrahlung einer punktförmigen Quelle im Freifeld nimmt die Instensität bei Verdopplung der Entfernung um 6dB ab, das entspricht einer Abnahme mit dem Quadrat des Abstandes zwischen Quelle und Hörer ($\frac{1}{r^2}$). Bei gerichteter Abstrahlung ist dieser Wert entsprechend kleiner. Die Schallintensität ist eine rein physikalische Betrachtung. Die Lautheit hingegen berücksichtigt die Frequenzabhängigkeit des menschlichen Gehörs, was in den Kurven gleicher Lautstärke dargestellt wird. Sie sind genormt nach DIN 45 630.

Da höhere Frequenzen im allgemeinen gerichtet und tiefere Frequenzen ungerichtet abgestrahlt werden, erfolgt eine Klangfärbung bei entfernten Schallereignissen, die die tieferen Frequenzen stärker bedämpft werden. Bei geringer Distanz werden tiefe Frequenzen leicht angehoben. Hinzu kommt, dass das menschliche Gehör bei sinkender Lautstärke für tiefe Frequenzen unempfindlicher wird. Weiters erfolgt eine Abschwächung der höheren Frequenzen durch die Luft, welche von Lufttemperatur und -feuchtigkeit abhängt. Aufgrund der persönlichen Erfahrung können wir je nach Klangfärbung und Lautheit die Entfernung zur Schallquelle bestimmen. In unbekannter Umgebung oder mit unbekannten Schallereignissen fällt eine korrekte Zuordnung der Entfernung schwer.

1.1.2 Erste Reflexionen

Abgesehen vom idealen Freifeld (bzw. schalltoter Raum), in dem keine Reflexionen auftreten, trifft Schall überall sonst auf reflektierende Flächen und wird, abhängig von der Frequenz und Größe des Hindernisses, entsprechend zurückgeworfen bzw. gebeugt. Ist das Hindernis entsprechend groß und besitzt im Vergleich zur Wellenlänge eine glatte Oberfläche, erfolgt eine

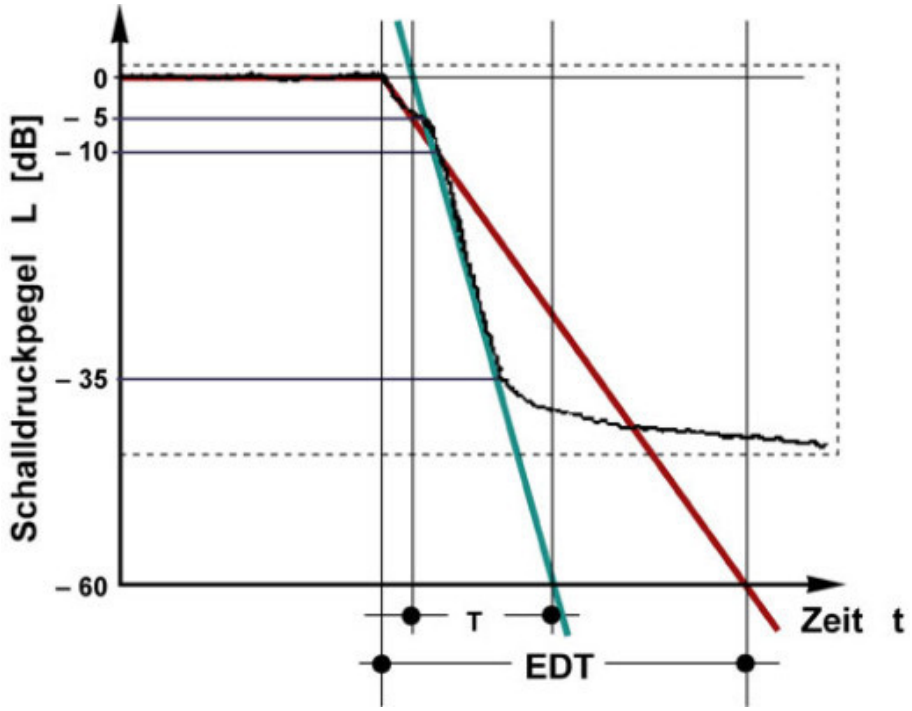


Abbildung 1.1: *Impulsantwort eines Raumes, Quelle: http://www.baunetzwissen.de, 07.02.2011*

geometrische Reflexion (Einfallswinkel = Reflexionswinkel). Ist die Oberfläche verhältnismäßig grob gegliedert, wird die Welle diffus (also in alle Richtungen) reflektiert. Bei akustisch gesehen kleinen Abmessungen wird die Schallwelle ums Hindernis gebeugt, die Welle wird in ihrer Ausbreitung kaum gestört. [WW07] Folgende Gütemaße dienen der Beschreibung des akustischen Verhaltens eines Raumes (siehe auch Abbildung 1.2; vgl. [Dic97], [UZ05]):

Nachhallzeit (decay time) ist die Zeit, in der das Schallfeld eines Raumes nach einer Anregung durch einen Impuls von 0dB auf -60dB SPL abfällt. Bei der Messung wird nur der Abfall auf -35dB betrachtet und auf -60dB extrapoliert. Dieses Maß ist die älteste und auffälligste Beschreibung eines Raumes. (siehe Abbildung 1.1 "T")

Anfangsnachhallzeit (EDT, early decay time) ist eine subjektivere Beschreibung des Nachhalls, jedoch stärker Ortsabhängig als die Nachhallzeit. Der Anfang des Ablingvorganges wird üblicherweise besser wahrgenommen. Daher wird der Abfall von 0db auf -10dB SPL gemessen und auf -60dB extrapoliert. (siehe Abbildung 1.1 "EDT")

Direktschall ist jener Anteil des Schallfeldes, der als erstes und auf direktem Weg am Hörort

eintrifft. Er trägt maßgeblich zur Ortung des Schallquelle bei.

Erste Reflexionen treffen kurz nach dem Direktschall am Hörort ein. Sie tragen durch Einfallswinkel, Stärke und Verzögerung maßgeblich zum Raumeindruck bei. Einerseits können sie die Deutlichkeit des Direktschalls erhöhen (falls sie um maximal 50ms verzögert sind), andererseits führen sie bei Aufnahmen zu Klangfärbungen durch Summierungen und Auslöschungen zwischen Direktschall und Reflexionen (Verzögerung von 0,8ms bis 20ms, das entspricht einem Umweg von 0,3m bis 7m). Reflexionen mit 20ms bis 50ms (7m bis 17m Umweg) sind bestimmend für den Raumeindruck. Treffen sie später beim Hörer ein, wirkt der Raum größer. Entsprechend kleiner wirkt der Raum bei früherem Eintreffen der ersten Reflexionen. Deutliche Reflexionen mit einer Verzögerung über 50ms werden als störende Echos wahrgenommen. Bei Sprache darf der Pegel der ersten Reflexionen bis zu 10 dB über dem des Direktschalls liegen, ohne dass die Richtungsortung der Schallquelle beeinträchtigt wird. Erste Reflexionen, welche aus gleicher Richtung wie der Direktschall kommen, werden kaum wahrgenommen. Seitlicher Schall hingegen wird deutlicher gehört und trägt stark zur empfundenen Räumlichkeit bei. Ob erste Reflexionen als störend empfunden werden oder ob sie der Hörsamkeit dienlich sind, hängt auch stark von der Art der Musik ab.

Nachhall bezeichnet die zahlreichen Reflexionen, die nach dem Direktschall auftreten.

Während Direktschall und die ersten Reflexionen noch über geometrische Modelle bestimmt werden können, ist eine Nachbildung des Nachhalls nur mehr in statistischen Modellen möglich. Hallalgorithmen verwenden zahlreiche kaskadierte IIR-Filter, um einen dichten Nachhall zu erzeugen. In diesem Programm wird lediglich der Direktschall und eine Reflexion berechnet.

1.1.3 Doppler Effekt

Der Doppler-Effekt beschreibt die Änderung der wahrgenommenen Frequenz, falls sich der Betrachter oder die abstrahlende Quelle bewegt. Er wurde 1842 vom österreichischen Physiker und Mathematiker Christian Doppler vorausgesagt und 1845 von Christoph Buys-Ballot in

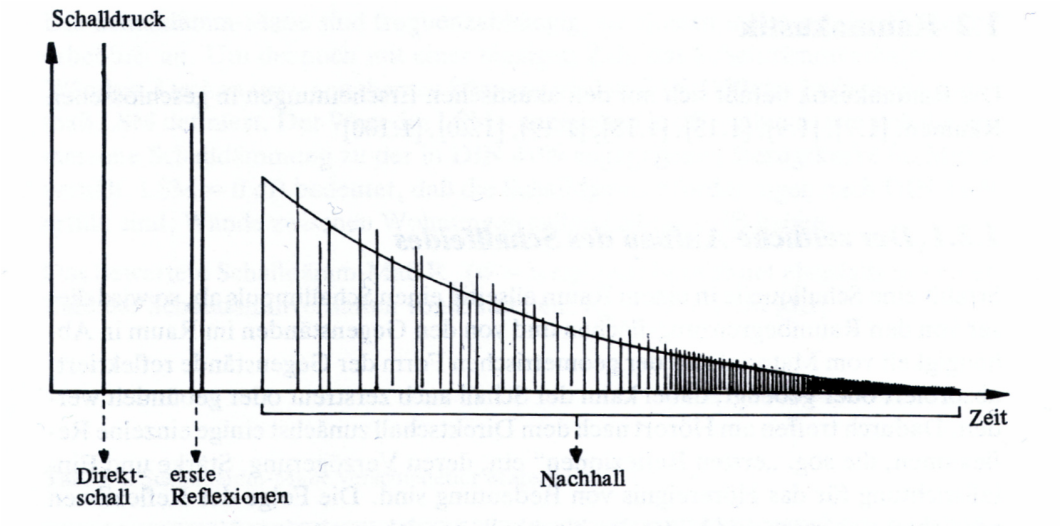


Abbildung 1.2: *Impulsantwort eines Raumes Quelle: [Dic97]*

der akustischen Domäne experimentiell bestätigt [dop]. Der Effekt tritt bei allen Formen von Wellen auf, in der Optik, bei Wasserwellen, bei Akustik etc. Im astronomischen Bereich beruhen viele Messungen auf dem Prinzip des Doppler-Effekt. Folgende Überlegungen gelten für beliebige Spektralbereiche und Geschwindigkeiten. Die auftretenden Phänomende werden jedoch in der akustischen Domäne erklärt (der Beobachter ist der Hörer, die abstrahlende Quelle ist die Schallquelle).

Beobachter in Ruhe, Quelle bewegt sich Bewegt sich die Schallquelle auf den Beobachter zu, wird die Wellenlänge gestaucht, das heißt sie wird kleiner (-). Entfernen sie sich, wird die Wellenlänge größer (+).

$$\lambda_{res} = (1 \pm \frac{v_0}{c}) \cdot \lambda_0 \tag{1.1}$$

Damit ergibt sich eine neue Frequenz an der Position des Beobachters:

$$f_{res} = \frac{f_0}{1 - \frac{v}{c}} s \tag{1.2}$$

Beobachter bewegt sich, Quelle in Ruhe Die Wellenlänge wird in diesem Fall nicht gestaucht oder gestreckt, da die Quelle sich nicht bewegt. Jedoch erscheint es dem Beobachter durch seine eigene Bewegung so, da er die Wellen schneller “passiert”.

Die sverschobene Frequenz ergibt sich zu

$$f_{res} = (1 \pm \frac{v}{c}) \cdot f_0 \tag{1.3}$$

Gleichung 1.2 und 1.3 stimmen nur für $v << c$ überein.

(vgl.[Far08]) In der Optik spricht man im ersten Fall (Entfernung zwischen Beobachter und Quelle wird kleiner) von einer Rotverschiebung, da das Licht in Richtung des roten Spektralbereichs verschoben wird. Umgekehrt spricht man von einer Blauverschiebung. In der Akustik macht sich diese Verschiebung durch eine Änderung der Tonhöhen bemerkbar. Bewegen sich Schallquelle und Hörer aufeinander zu, verschieben sich die Frequenzen nach oben, andernfalls nach unten (allgemein bekannt ist der vorbeifahrende Rettungswagen mit eingeschaltetem Folgetonhorn). An folgendem Beispiel ist zu erkennen, dass Intervalle dabei erhalten bleiben, das heißt Akkorde und Harmonien werden nicht verfälscht.

> Die Frequenzen $f_{0,1}$ und $f_{0,2}$ stehen in einem beliebigen Verhältnis. Bei einer Bewegung der Schallquelle mit der Geschwindigkeit v_0 entsteht an der (ruhenden) Abhörposition das Verhältnis
>
> $$\frac{f_{res,1}}{f_{res,2}} = \frac{\frac{c}{c \pm v_0} \cdot f_{0,1}}{\frac{c}{c \pm v_0} \cdot f_{0,2}} = \frac{f_{0,1}}{f_{0,2}}$$
>
> Das Verhältnis der resultierenden Frequenzen bleibt gleich dem der ursprünglichen.

1.2 Programmierung

Als Programmiersprache wurde Pure Data (kurz: Pd) gewählt. Kurz und prägnant ausgedrückt:

> Pd (aka Pure Data) is a real-time graphical programming environment for audio, video, and graphical processing. [Puc96]

Es wurde von Miller Puckette[1] entwickelt. Im Gegensatz zum kommerziellen Max/MSP[2], welches eine Abspaltung des ebenfalls von Puckette entwickeltem Max/FTS ist, steht Pd unter der GNU-Lizenz und wird von einer aktiven Entwicklergemeinde gewartet.

Pd ist für GNU/Linux, MacOS X, IRIX, BSD und Windows erhätlich. In kürzester Zeit lassen sich damit lauffähige Programme erstellen. Es eignet sich daher gut zum Entwerfen und Testen von Prototypen. Aber auch umfangreiche Projekte sind programmierbar, welche dank des objektorientieren Aufbaus übersichtlich bleiben. Ein wichtiger Aspekt ist die Möglichkeit, objektorientiert programmieren zu können. Durch geschickte Aufteilung der Programmteile lassen sich nachträgliche Änderungen und Erweiterungen leichter realisieren. Weiters ist eine Erweiterung von Pd mittels Bibliotheken möglich. Dabei werden Objekte als zusätzliche Funktionen dem Programm hinzugefügt.

Ein Grundprinzip von Pd ist die Trennung von DSP und GUI (siehe Abbildung 1.1). Der DSP-Teil erledigt die Berechnungen in Echtzeit und arbeitet mit einem Message-Interpreter und Scheduler. Messages werden existieren nur beim Zeitpunkt des Auftretens. Sie werden vom Benutzer erzeugt bzw. vom Programm selbst generiert. Sie sind die Steuerparameter zur Verarbeitung. Die GUI hat die Aufgabe, mit dem Betriebssystem zu kommunizieren und die grafische Oberfläche aufzubereiten. Die Verbindung zum Betriebssystem erfolgt über das tk-Toolkit [Ous94]. [Puc96]

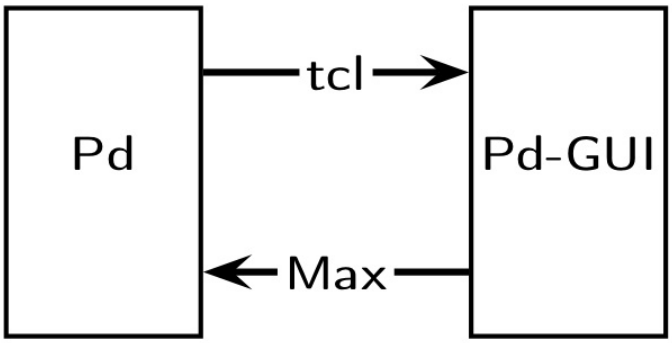


Abbildung 1.1: *Trennung von DSP und GUI als Grundprinzip von Pd*

[1] http://crca.ucsd.edu/~msp/
[2] http://cycling74.com/

1.2.1 Das User-Interface

Beim Entwurf eines Gerätes oder einer Software liegt ein großer Augenmerk auf der Bedienbarkeit. Dabei geht es nicht darum, alle möglichen Parameter veränderbar und steuerbar zu machen, sondern dem Benutzer eine einfache und intuitive *Benutzungsschnittstelle* ([Her05]) bereitzustellen, mit deren Hilfe das Gerät oder die Software problemlos und effizient benutzt werden kann. In [Iso06] ist der Begriff *Gebrauchstauglichkeit* (Usability) beschrieben als

> "das Ausmaß, in dem ein Produkt durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufrieden stellend zu erreichen."

Die Folgen einer schlechten Bedienbarkeit sind:

- physische und psychische Beeinträchtigung der Nutzer
- hoher Einarbeitungsaufwand
- Herausbildung "unentbehrlicher Experten", die die unnötig komplizierten Systeme zu bedienen gelernt haben
- die Systeme arbeiten gegen die gewohnte oder gewünschte Arbeitsweise
- Verschiebung in der Relevanz von Arbeitsvorgängen
- Verschwendung menschlicher Denkfähigkeit und Flexibilität

Neben einer Beeinträchtigung des Benutzers führen diese Auswirkungen auch zu einer Reduktion der Effizienz der Benutzer, was zu unnötigen Kosten in der Anwendung der Systeme führt. (Auszug aus [Her05])

1.2.2 DSP-Programmierung

Das Programm verläuft ähnlich einem Datenstrom und wirkt daher sehr intuitiv. Signale werden mittels synchronem Datenfluss (kurz SDF) miteinander verbunden. Sie laufen oben

bei einem Objekt hinein und unten laufen sie wieder heraus. Die Steuerung erfolgt wie bereits erwähnt über Messages, die aus Benutzereingaben generiert werden.

Aus der Systemskizze lässt sich folgender Signalfluss ableiten (Abbildung 1.2). Anhand dieses Überblicks wird deutlich, wo sich ähnliche Objekte befinden (zB. die Wegstrecken). Diese lassen sich abstrahieren, sodass als Resultat ein möglichst übersichtlicher Code bestehen bleibt.

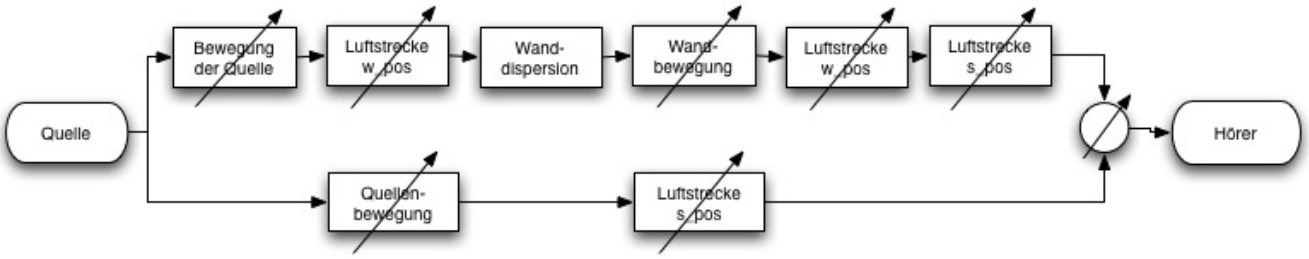


Abbildung 1.2: *Signalflussdiagramm der Problemstellung*

2 Die Objekte dopplerre~ und dopplerre_gui

Dieses Modul wird in Form einer Bibliothek mit zwei Objekten erstellt. Diese Bibliothek kann als Erweiterung zu Pd verwendet werden. Sie besteht aus einem GUI Objekt dopplerre_gui zur Bedienung und einem DSP Objekt dopplerre~, welches alle Berechnungen durchführt.

2.1 dopplerre_gui

Das User-Interface wird durch dopplerre_gui <channel> zur Verfügung gestellt. Es erscheint die GUI, mit der sämtliche Einstellungen vorgenommen werden (Abbildung 2.1). Mit <channel> erfolgt die Verlinkung mit dem entsprechenden DSP-Objekt. Zusammengehörige Module müssen am gleichen Kanal arbeiten. Die Bedienelemente sind aufs Notwendigste beschränkt, ohne jedoch unübersichtlich zu wirken. Es gibt einige globale Parameter: max_dist, filtering, damping und die Schallgeschwindigkeit sonci_speed. Die anderen Parameter werden der Wand (Präfix w) oder der Quelle (Präfix s) zugeteilt. spos gibt die Entfernung der Quelle zum Hörer an, wpos ist die Entfernung der Wand von der Quelle. Die Parameter wamp und samp geben an, in welchem Bereich sich die Quelle bzw. die Wand bewegen dürfen.

Das User-Interface gliedert sich in drei große Bereiche, welche auch farblich getrennt sind:

Globale Einstellungen sind ockerfarben hinterlegt. Sie befinden sich im oberen Bereich, damit die Bypass-Funktion gleich erkannt wird und weil max_dist eingestellt werden muss, um überhaupt einen Effekt zu erzielen. Die Regler für filtering und damping sind als vslider ausgeführt, um nicht den Eindruck zu erwecken, zwischen dem linken und

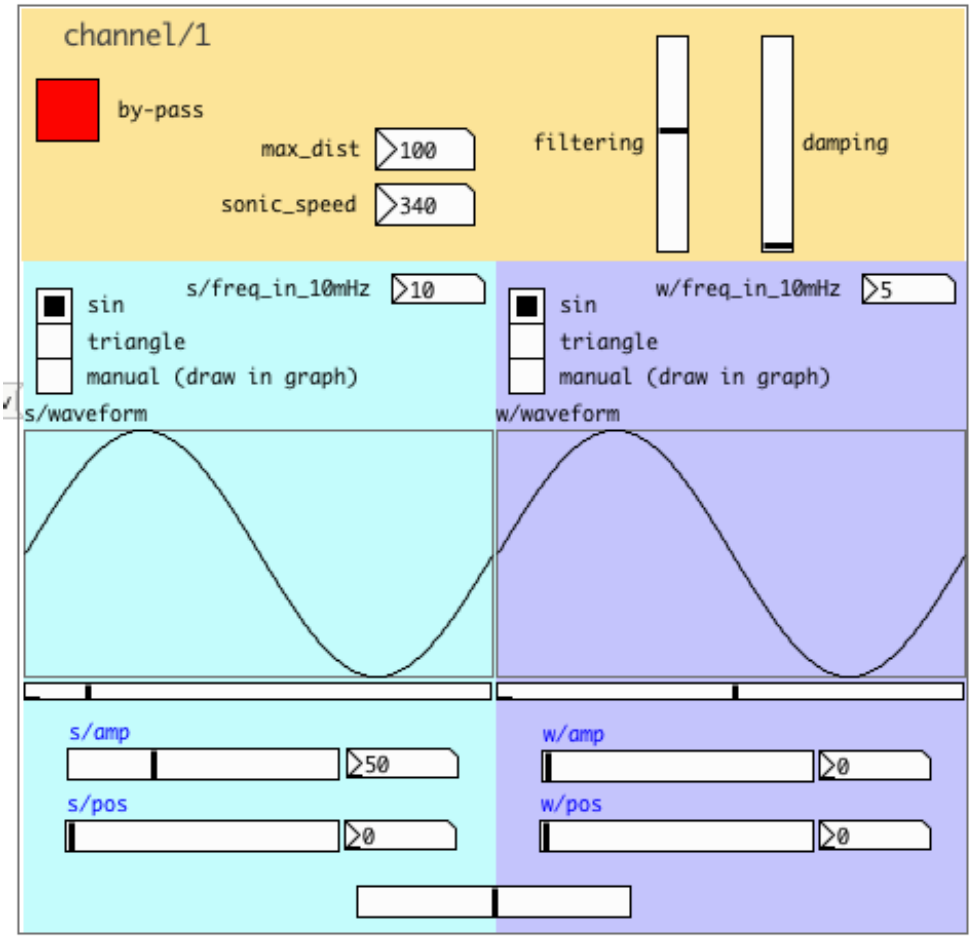


Abbildung 2.1: *Das User-Interface*

rechten Teil der GUI zu regeln, was bei Verwendung von `hslider` versehentlich so interpretiert werden könnte. Dafür wird der Mix mit einem `hslider` ohne Beschriftung eingestellt (Abbildung 2.1 unten Mitte).

Quellenparameter befinden sich an der linken Seite mit dem Gedanken, dass die Wege gemäß der Systemskizze von links nach rechts verlaufen. Die Wellenform wurde zuerst mit reinen Buttons ausgewählt. Die nun verwendeten Radio-Buttons haben jedoch die Vorteile, dass die aktuelle Auswahl angezeigt wird und ein gleichzeitiges Auswählen mehrerer Elemente nicht möglich ist. Die aktiven Wellenform werden sofort als Graph dargestellt. Im manuellen Modus kann eine eigene Wellenform eingezeichnet werden. Unter den Graphen wandert jeweils ein `hslider` mit der Indexposition der Arrays mit.

Dort lässt sich also die aktuelle Position an der Bewegungskurve ablesen. Das gilt jedoch nur für niedrige Frequenzen bis ca. 3 Hz. Darüber hinaus lässt sich die Bewegung des Sliders nicht mehr verfolgen und ist auch nur von untergeordnetem Interesse.

Wandparameter sind analog zu den Quellenparametern ausgeführt und liegen auf der rechten Seite.

Die **Bypass-Funktion** (Abbildung 2.1) erlaubt, das Modul zu deaktivieren. Im DSP-Teil in Abbildung 2.2 ist am rechten Pfad zu erkennen, dass im Bypass-Modus keine Manipulation des Signals erfolgt; die linken zwei Pfade werden nicht durchlaufen.

Über den **Mix** lässt sich Einstellen, welchen Anteil des Gesamtsignals das von der Wand reflektierte Signal einnimmt. Der Verlauf erfolgt nach einer Cosinus-Funktion. Bei einem linearen Fader ist bei der Mittenstellung des Faders die Lautstärke um 3dB geringer, da beide Lautsprecher jeweils nur mit halber Amplitude angesteuert werden. Beim Faden wäre dieser Einbruch deutlich hörbar. Der Cosinus-Fader hingegen bringt bei der Mittenstellung insgesamt die volle Leistung. An den Endpositionen klingt das entsprechende Signal gleichmäßig aus. Bei einem Fader, der mit der Quadratwurzel gewichtet ist, würde das Signal am Rand aprupt ausgeblendet werden (vgl. Abbildung 2.2).

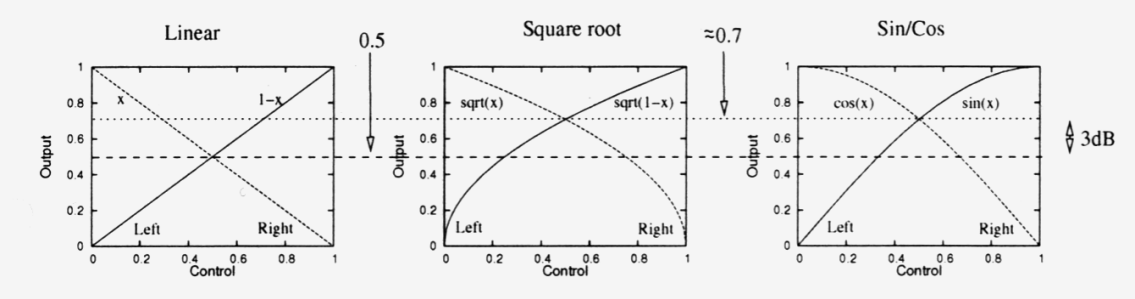


Abbildung 2.2: *Übertragungsfunktionen für Linear-, Wurzel- und Sinus/Cosinus-Fader. Quelle: [Far08]*

2.2 dopplerre~

Das Struktur des Objekts ist sehr modular. Das erhöht die Übersicht und erleichtert eine weitere Bearbeitung. Es gibt lediglich einen Ein- und einen Ausgang für die Audiodaten, da

dem Signal nur eine eindimensionale Entfernungsinformation hinzugefügt wird. Eine Richtungsinformation (z.B. links-rechts) wird nicht berechnet. Um das Modul in ein Programm einzubinden, wird ein Objekt mit dem Namen `dopplerre~ <channel>` instanziert. Mit dem Übergabeparameter `<channel>` wird jede Instanzierung identifiziert, somit können mehrere Objekte nebeneinander verwendet werden. Beispielsweise kann für Mehrkanal-Anwendungen in jeden Kanal ein Objekt `dopplerre~` eingebunden werden, welche unabhängig voneinander arbeiten.

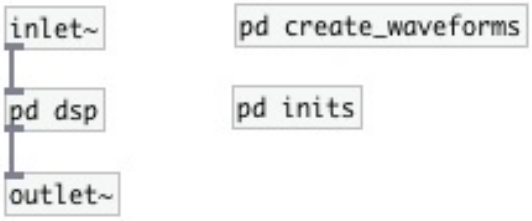


Abbildung 2.1: *Aufbau von dopplerre~*

2.2.1 dsp

Im DSP-Teil befindet sich der eigentliche Hauptteil, nämlich der gesamte Signalpfad mit allen Klangmanipulationen (Abbildung 2.2). Der linke Pfad ist die direkte Strecke von der Schallquelle zum Ort des Höreres, der rechte Pfad führt von der Schallquelle zur Wand und anschließend zum Hörer.

Der Signalpfad (Abbildung 2.2) beinhaltet folgende Objekte:

`path` wird mit dem Paramter `s` oder `w` der Quelle oder der Wand zugewiesen und dient zur Berechnung von Laufzeit und Dämpfung der jeweiligen Strecken

`move` wird ebenfalls mit den Parametern `s` oder `w` der Quelle oder der Wand zugewiesen und generiert mit einem variablen Delay (`vd~`) den Dopplereffekt. Als zweiter, optionaler Parameter kann hier `b` für "back" angegeben werden. Betrachtet man die Systemskizze, wird deutlich, dass bei einer Bewegung der Schallquelle zwei Dopplerfeffekte berücksichtigt werden müssen: Einerseits in Richtung des Hörers, und andererseits in Richtung der

Wand. Diese zwei Effekte sind invers, eine Bewegung zum Hörer hin bedeutet nämlich eine Bewegung von der Wand weg. Somit tritt in die eine Richtung eine Frequenzerhöhung auf, in die andere jedoch eine wird die Frequenz kleiner. Der Steuerparameter `b` berücksichtigt diese Abstrahlung "nach hinten". Programmtechnisch bewirkt dies eine Spiegelung der Wellenform an der x-Achse, also einen Vorzeichenwechsel der Werte aus dem Array, mehr dazu in Kapitel 4.4.2.

`w_disp` ist vorgesehen, um der Wand z.B. einen frequenzabhängigen Reflexionsfaktor zuordnen zu können, besitzt aber keine manipulative Funktion

2.2.2 `create_waveforms`

Hier werden die Bewegungsmuster erzeugt, welche die Schallquelle und die Wand annehmen können. Zur Auswahl stehen eine Sinus- und eine Dreieckschwingung und eine manuelle Vorgabe. Die Werte werden bei Anklicken der jeweiligen Funktion erzeugt und in Arrays gespeichert.

2.2.3 `inits`

Um das Modul sofort nach der Instanzierung verwenden zu können, werden sämtliche Werte initialisert, zum Beispiel die Schallgeschwindigkeit $c = 340m/s$, die Dämpfungscharakteristik der Luftstrecke oder Entfernungsangaben. Auch beim Erstellen des GUI-Objekts wird die Initialisierung noch einmal aufgerufen, damit auch dieses mit den korrekten Werten beschrieben wird.

2.3 Parameter und Wertebereiche

Die `hslider` `w_amp`, `w_pos`, `s_amp` und `s_pos` haben einen Wertebereich von 0-127. Somit ist eine Steuerung mit MIDI-Controllern ohne weiteres Mappen der Bereiche möglich. Die volle Aussteuerung entspricht hier dem Wert von `max_dist`. Es werden mit den Reglern also Verhältnisse eingestellt. Erst durch die Vorgabe einer Maximaldistanz werden konkrete

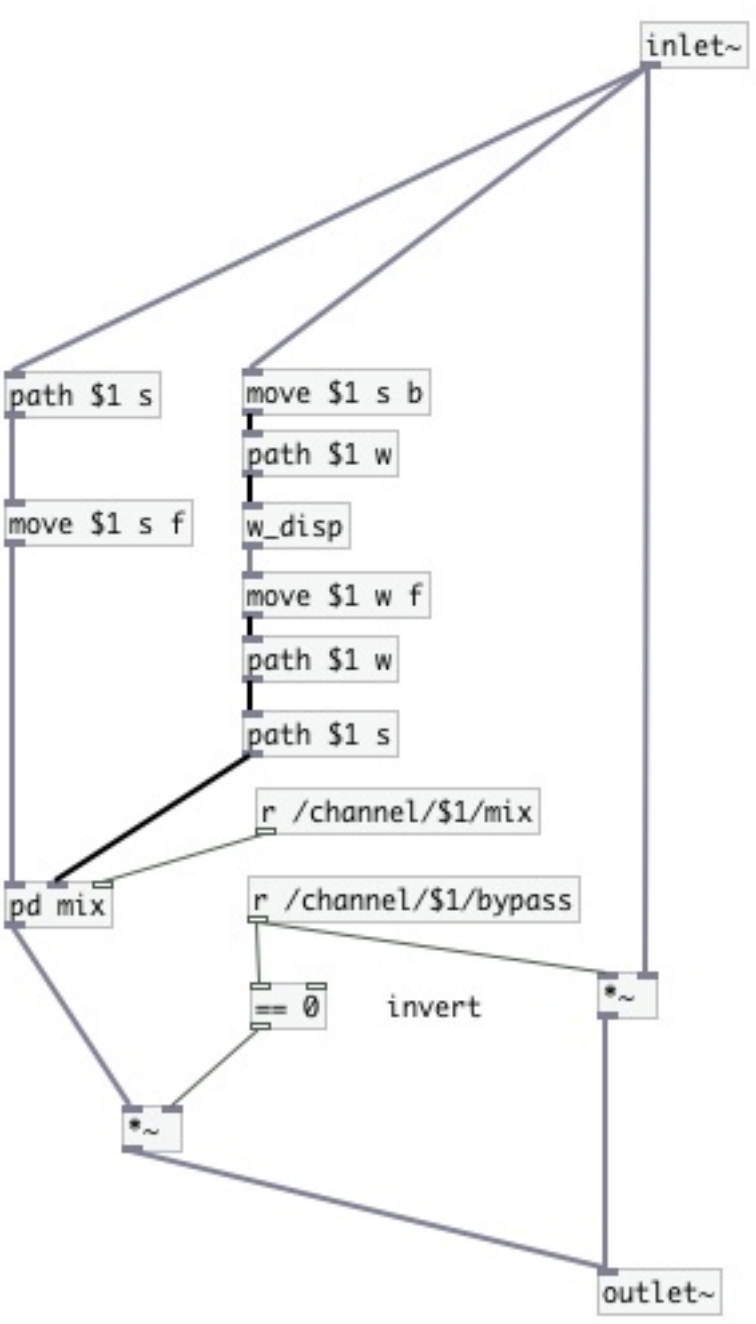


Abbildung 2.2: *Signalpfade*

Werte geliefert, welche auch rechts der Regler angezeigt werden. Der Wertebereich des Mix reicht von 0 (dry) bis 1 (wet). Eine Einstellung auf 0 (Regler ganz links) bedeutet, dass nur das Quellensignal mit entsprechender Laufzeit mit eventuellem Dopplereffekt durch die

Quellenbewegung hörbar ist (linker Signalpfad). In mittiger Positionierung des Reglers sind das verzögerte, reflektierte Signal und das Quellensignal gleich laut zu hören. Befindet sich der Regler rechts, ist nur das reflektierte Signal der Wand mit entsprechendem Dopplereffekt zu hören. Ebenso hat der Filter einen Wertebereich von 0 bis 1, welche jedoch für den Benutzer nicht wichtig ist. Anhand der Position des Reglers soll rein gefühlsmäßig die Filterwirkung der Luft eingestellt werden. Im Hintergrund verbirgt sich dazu ein Tiefpass 1. Ordnung. Der Reglerwert wird mit 10000 multipliziert und als Grenzfrequenz-Parameter einem `lop~`-Objekt zugeführt, welche somit von 0Hz bis 10kHz reicht. Damit wird die Stärke der Filterwirkung der Luft (annähernd Tiefpass 1. Ordnung) eingestellt, wodurch durchaus brauchbare Ergebnisse erzielt werden. Weiters wird mit `damping` die Dämpfung der Lautstärke berücksichtigt. Auch hier sind die Bereiche des Reglers ohne Beschriftung. Unten befindet sich das Minimum, die Lautstärke wird also durch Wegstrecken nicht beeinflusst, und am oberen Rand des Reglers befindet sich die Maximaleinstellung. Hier wird die Lautstärke am meisten beeinflusst.

Die Frequenz, mit der sich Quelle bzw. Wand bewegen, wird in 10mHz Schritten eingegeben. In diesem Größenbereich herrscht eine angenehme Empfindlichkeit. 1Hz Schritte wären zu ungenau, 1mHz Schritte wären in der Einstellung zu langsam.

3 Ähnliche Realisierungen

Es gibt bereits Geräte bzw. Plugins, die mit dem Doppler-Effekt arbeiten. Einige davon sind nachfolgend kurz Beschrieben.

3.1 Leslie-Lautsprecher

Eine sehr bekanntes Gerät, welches mit dem Doppler-Effekt arbeitet, ist der Leslie-Lautsprecher (Rotationslautsprecher, Leslie-Kabinett, Leslie-Box), benannt nach seinem Erfinder Donald Leslie (1911-2004). Dieser Lautsprecher verleiht der Hammond Orgel ihren typisch modulierten Klang. Mit Hilfe eines rotierenden Lautsprechers wird ein Vibrato erzeugt, welches viel komplexer ist als eine gewöhnliche Tonhöhenmodulation.

3.2 Doppler Plugins

Plugins für den Doppler-Effekt werden von einigen Herstellern angeboten.

3.2.1 Waves

Im Plugin von Waves lässt sich die Bewegung der Schallquelle mit Startpunkt, Endpunkt, Pfad und Bewegungsrichtung festlegen. Weiters ist die Dämpfung der Luft einstellbar sowie einige Faktoren für Pitch, Gain, Reverb etc. (Abb. 3.1)

Abbildung 3.1: *GUI des Doppler-Plugins von Waves. Quelle: http://www.waves.com/objects/html/plug-ins_ html/doppler.html, 07.02.2011*

3.2.2 Oli Larkin

Oli Larkin's Spacestation ist ein kleines Plugin ähnlich dem zu Waves. Im Fenster wird die Bewegung der Quelle dargestellt und es lassen sich die wichtigsten Parameter für Bewegung und Doppler-Effekt einstellen (Abb. 3.2).

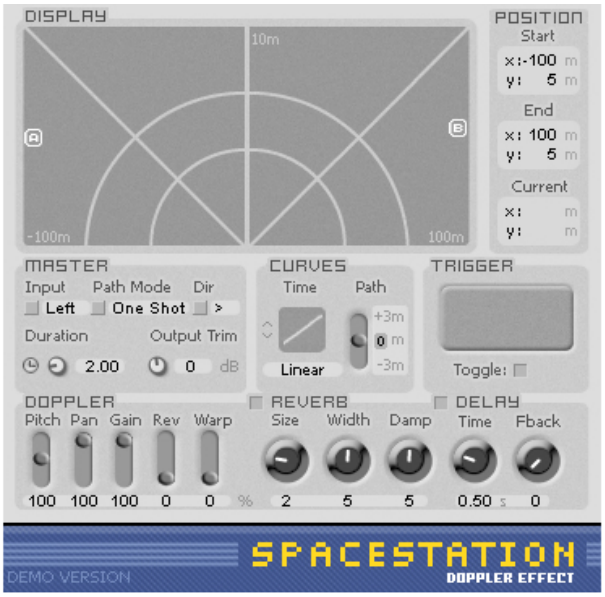


Abbildung 3.2: *Spacestation von Oli Larkin. Quelle: http://www.olilarkin.co.uk/index.php?p=spacestation, 07.02.2011*

3.2.3 GRM tools

Auch hier finden sich die gängigen Einstellungen. Zudem lässt sich die Amplitudenänderung von der Änderung der Entfernung entkoppeln, um auch unrealistische Szenarien darstellen zu können (Abb. 3.3).

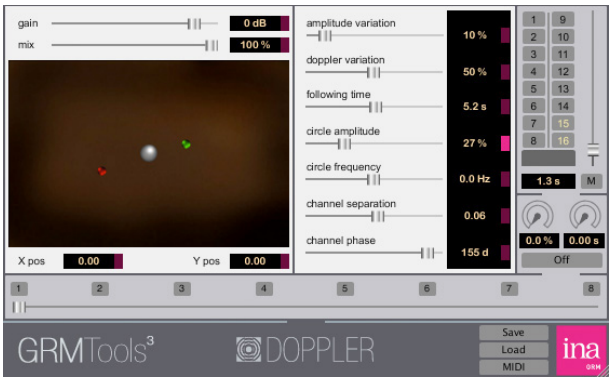


Abbildung 3.3: *Doppler-Plugin von GRM tools. Quelle: http://www.inagrm.com/doppler, 07.02.2011*

3.2.4 SpinAudio

Das Plugin 3D Panner Motion Effects von SpinAudio erlaubt im Gegensatz zu Waves eine Bewegung im dreidimensionalen Raum. Der Bewegungspfad der Quelle und die Gesschwindigkeit lassen sich vorgeben (Abb. 3.4).

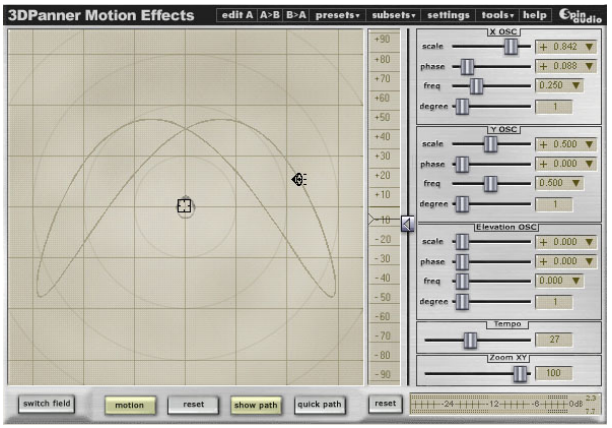


Abbildung 3.4: *Doppler-Plugin von SpinAudio. Quelle: http://www.kvraudio.com/get/566.html, 07.02.2011*

3.2.5 Unterscheidung zu `dopplerre~`

Die Bewegungskurven obiger Plugins können wesentlich komplexer eingestellt werden, mitunter bis in die dritte Dimension. Allerdings bietet keines die Möglichkeit der Reflexion an einer Wand. Ein gängiges Beispiel ist ein vorbeifahrendes Einsatzauto mit Folgetonhorn. Der Dopplereffekt, der dabei auftritt, ist allgemein bekannt und kann mit jedem der genannten Plugins simuliert werden. Mit dopplerre~ kann jedoch auch noch eine Reflexion erzeugt werden, die sehr belebend auf den Klang wirkt.

4 Implementierung

4.1 Reflexion

Die Wand ist schallhart, das heißt der Schall wird zur Gänze reflektiert. Sie steht normal zur Ausbreitungsrichtung des Schalls und ist so groß, dass alle Frequenzen mit einem Reflexionsfaktor $r = 1$ reflektiert werden. Betrachtet man den Signalverlauf in Abbildung 4.1 (Ausschnitt aus Abbildung 2.2), erkennt man die logische Folge des Weges: Das Audiosignal erfährt zuerst den umgekehrten Dopplereffekt, danach durchläuft es die Wegstrecke bis zur Wand (`path w`), welche frequenzabhängige Reflexionsfaktoren besitzen kann (`w_disp`). Durch die Bewegung der Wand erfährt das Signal hier einen weitern Dopplereffekt durch `move w f` und gelangt anschließend durch die zwei `path`-Objekte zum Hörort.

Abbildung 4.1: *DSP-Teil der Wandreflexion*

4.2 Arrays

Die Arrays werden wie oben erwähnt mit einer Sinus-, Dreieck- oder Freihandkurve beschrieben. Die Funktion dazu befindet sich im Subpatch `create_waveforms`. Die Parameter des Befehls `sinesum` geben Anzahl und Amplitude der Oberschwingungen an, woraus durch Summation eine beliebige periodische Wellenform gebildet werden kann. Konkret wird mit der Auswahl der Sinuskurve über `sinesum 2051 1` eine Sinusschwingung mit 2051 Samples erzeugt.

Die Dreiecksfunktion besitzt mehrere Oberschwingungen und wird mit `sinesum 2051 1 0 -0.111111 0 0.04 0 -0.0204082 0 0.0123457 0 -0.00826446 0` gebildet. Anschließend wird die Kurve mit `normalize 1` auf den Wertebereich ± 1 begrenzt. Bei der Auswahl, die Kurve selbst einzuzeichnen, wird lediglich der Gleichanteil 0 geschrieben (`sinesum 2051 0`).

4.3 Einfluss der Luft und Entfernung

Aus dem Alltag ist bekannt, dass sich der Klang einer Schallquelle mit der Entfernung ändert. Dabei wird er nicht nur leiser, sondern auch das Spektrum verändert sich. Somit gibt es zwei wesentliche Aspekte, die im Programm berücksichtigt wurden. Die spektrale Veränderung wurde mit einem Tiefpass 1. Ordnung realisiert. Der Benutzer kann dabei durch die Wahl der Grenzfrequenz dessen Stärke verändern. Eine Implementierung der Lautstärkeänderung war anfangs nicht vorgesehen, jedoch trägt deren Wirkung wesentlich zur Qualität des Programms bei. Die Lautstärke nimmt bei Verdopplung der Entfernung um 6dB ab, das entspricht also einem Verhalten, welches indirekt proportional ist zum Abstand ($\frac{1}{r}$). Dieser Teil wurde in einem eigenen Modul abstrahiert (Abbildung 4.1). Die reale Entfernung wird mit einem Faktor `damping` multipliziert. Dieser Faktor hat einen Wertebereich von 0.1 bis 1 und ist als logarithmischer Regler ausgeführt. Dadurch lässt sich das Verhalten angenehm einstellen. Das Audiosignal wird durch das Ergebnis dieser Multiplikation dividiert. Dabei muss natürlich darauf geachtet werden, dass der Divisor größer oder gleich null bleibt, was mit dem Objekt `max 1` gewährleistet wird.

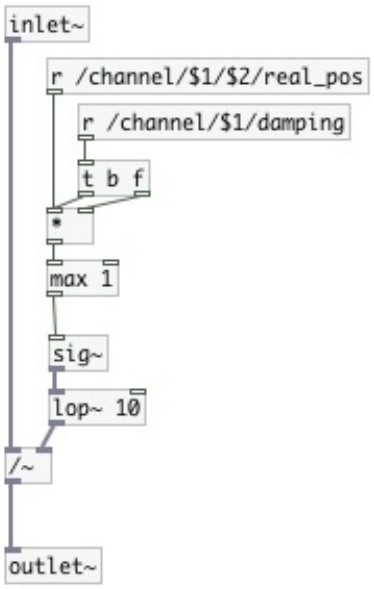


Abbildung 4.1: *Lautstärkedämpfung der Luft*

4.4 Doppler Effekt

4.4.1 Realisierung

Prinzipiell gibt es mehrere Möglichkeiten, einen Dopplereffekt zu erzeugen. Einerseits mit einem Pitch-Shifter oder mit einem Ringbuffer, der mit erhöhter oder verringerter Geschwindigkeit ausgelesen wird. In diesem Programm wird ein variables Delay (`vdelay~`) verwendet, was der Variante mit dem Ringbuffer entspricht. Eine Delay-Line wird dabei beschrieben und anschließend mit ändernder Verzögerung ausgelesen. Die Delay-Line ist mit 6000ms begrenzt, was einer Strecke von ca. 2000m entspricht. Eine Überschreitung der Verzögerungszeit macht sich durch kurze Aussetzer bemerkbar. Der Einfachheit halber wurde der Dopplereffekt gemäß Gleichung 1.3 realisiert, obwohl sie die Frequenzverschiebung beschreibt, wenn der Beobachter in Bewegung ist und die Quelle ruht. Dieses Modul wird seine Verwendung nicht dort finden, wo eine exakte Imitation der Umwelt gefordert wird, sondern wo Effekte nachgebildet werden. Bei geringen Geschwindigkeiten ist das Ergebnis noch sehr nahe an der "korrekten" Variante und hohe Geschwindigkeiten sind in der Realität ohnehin utopisch. Zudem entfällt das Problem der Polstelle, wenn die Bewegungsgeschwindigkeit gleich der Schallgeschwindigkeit ist, also im Bereich der Schallmauer.

4.4.2 Quellen-/Wandbewegung

Als Bewegungsmuster sind wie oben bereits erwähnt eine Sinuskurve, eine Dreieckskurve oder eine selbst gezeichnete Vorgabe auszuwählen. Die Werte dieser Kurven sind in einem Array gespeichert, welches anschließend sampleweise ausgelesen (Abbildung 4.1, rote Kurve) wird. Diese Werte werden durch Multiplikation mit 0.5 und anschließender Verschiebung in einen positiven Wertebereich gebracht (blaue Kurve). Damit erfolgt lediglich eine Auslenkung rechts von der Fixposition (`spos` bzw. `wpos`) aus betrachtet. Mit der Frequenz wird die Auslesegeschwindigkeit beeinflusst und mit `samp` bzw. `wamp` wird die maximale Auslenkung verändert. Bewegt sich die Quelle, erfährt der Hörer einen anderen Dopplereffekt als die Wand, was im Programm mit dem Parameter `b` angegeben wird. Dadurch werden die Arraywerte an der x-Achse gespiegelt (grüne Kurve), entsprechend wird die blaue kurve an $x = 0.5$

gespiegelt (pinke Kurve).

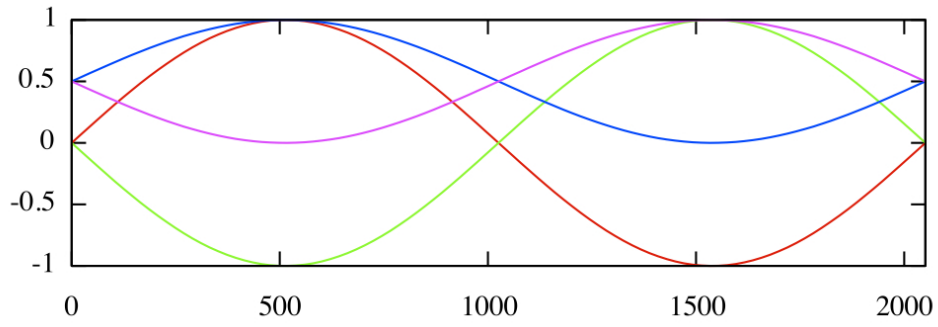


Abbildung 4.1: *Werte aus dem Array der Sinusfunktion*

4.4.3 Änderung der fixen Entfernungen

Beim Verstellen von `wpos` und `spos` muss rein intuitiv auch ein Dopplereffekt auftreten. Das stellte anfangs ein kleines Problem dar. Diese beiden Parameter werden vom Objekt `path` verarbeitet. In diesem Objekt wurde das Signal in einer Delayline gespeichert und um $t = \frac{w\_pos \cdot max\_dist}{c} ms$ später über `delread~` ausgelesen. Dieses Objektist jedoch nicht für dynamische Veränderungen der Delay-Zeiten ausgelegt ist (Abbildung 4.2). Bei einer Veränderung der Delayzeit tritt bei diesem Objekt keine Interpolation zwischen den Samples auf. Es treten Artefakte auf, die den ursprünglichen Klang völlig zerstören. Somit kommt ebenso wie bei der Realisierung des Doppler-Effekts auch hier ein variables Delay zum Einsatz. Das Steuersignal gibt die Verzögerung in Millisekunden an, der Benutzer gibt jedoch eine Wegstrecke vor. Zur Umrechnung von der Strecke zur entsprechenden Zeit wird die Abstraktion `m2ms`

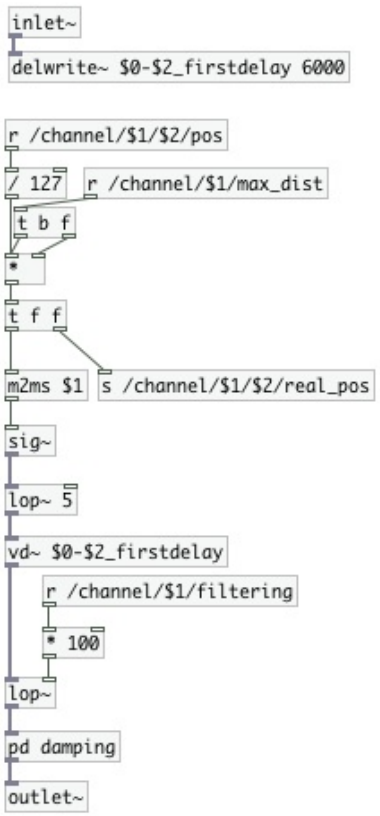


Abbildung 4.2: *`path`-Objekt*

verwendet. Diese errechnet nach $t[ms] = \frac{x\_pos[m]}{1000 \cdot c[\frac{m}{s}]}$ die Verzögerungszeit in Millisekunden. Dieses Steuersignal wird zur Glättung über `line~` in ein Audiosignal verwandelt und und anschließend mit einem Tiefpass gefiltert.

5 Ergebnisse

Dieser Arbeit ist eine CD mit Hörbeispielen beigelegt. Die Reihenfolge der Tracks ist analog zur Beschreibung der Grafiken. Um auch in Textform die Klangveränderungen beschreiben zu können wurde eine grafische Auswertung gewählt.

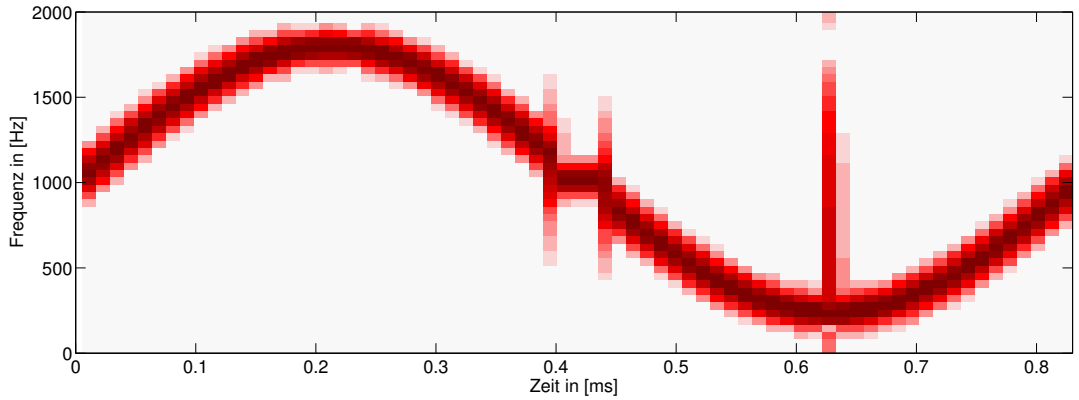
5.1 Grafische Auswertung

Nachfolgend werden einige Spektrogramme dargestellt, welche die Wirkungsweise des Moduls beschreiben. Dabei wird auch ein wesentlicher Nebeneffekt gut sichtbar, nämlich das Aliasing.

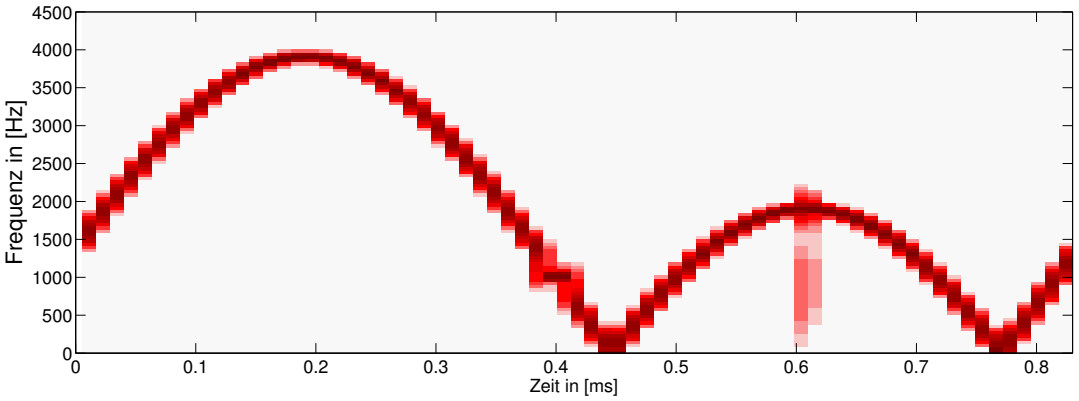
Am Eingang liegt ein Sinus-Ton mit 1000Hz. Die Wand schwingt sinusförmig mit einer Frequenz von 1.2Hz und hat eine maximale Auslenkung von 70m. Um nach Gleichung 1.1 die Frequenzen berechnen zu können, wird die Maximalgeschwindigkeit der Wand benötigt. Die Wand bewegt sich mit der Funktion $x = w/amp \cdot sin(2\pi \cdot w/freq \cdot t)$. Die Ableitung dieser (Orts)funktion führt zur Geschwindigkeit, welche lautet $x = (2\pi \cdot w/freq \cdot w/amp \cdot sin(2\pi \cdot w/freq \cdot t)$. Die Frequenz $w/freq$ ist vorgegeben mit 1,2Hz und die maximale Auslenkung w/amp beträgt 70m. Diese Werte eingesetzt ergeben eine maximale Geschwindigkeit von $527\frac{m}{s}$. Die Wellenlängen, die sich daraus ergeben, sind $0.8678m$ und $0.1878m$, was Frequenzen von $391Hz$ und $1810.6Hz$ entspricht. In Abbildung 5.1(a) ist sehr schön der Grundton bei 1000Hz erkennbar sowie die Reflexion an der Wand, welche symmetrisch zu den 1000Hz zwischen ca. 300Hz und 1800Hz schwingt. Die Berechnung stimmt mit der Beobachtung überein.

In Abbildung 5.1(b) ist sehr deutlich Aliasing zu erkennen. Die maximale Auslenkung der Wand wurde von 70m auf 260m vergrößert. Dadurch wird eine höhere Geschwindigkeit

durch die Bewegung erreicht, was eine stärkere Verschiebung der Frequenz zur Folge hat. Frequenzen, welche laut Berechnung in den negativen Bereich wandern, werden an 0Hz gespiegelt. Betrachtet man die positive Auslenkung, ist deren Differenz im Spitzenwert ca. 3000Hz zur Grundschwingung bei 1000Hz. Die negative Auslenkung reicht somit von 1000Hz bis 0Hz und wird anschließend bis 2000Hz zurückgespiegelt.



(a) ohne Aliasing



(b) auftretendes Aliasing

Abbildung 5.1: *Verschiebung eines 1000Hz-Tones bei sinusförmiger Wandbewegung*

Bei dreiecksförmiger Bewegung, bei der die Bewegung pro halber Persiode also mit gleicher Geschwindigkeit abläuft, werden oben beschriebene Effekte ebenfalls sehr gut sichtbar. Die

Ableitung einer linearen Funktion ergibt die Steigung, welche hier die Geschwindigkeit angibt. In 5.2(a) beträgt die maximale Auslenkung 20m. Die Wand entfernt sich innerhalb einer halben Periode

$$\frac{T}{2} = \frac{1}{2 \cdot f} = \frac{1}{2 \cdot 1,2Hz} = 0,42s$$

20m weit von der Position des Hörers weg und in der nächsten halben Periodendauer wieder zurück. Das ergibt eine Geschwindigkeit von

$$v = \frac{20m}{0,42s} = 47,6\frac{m}{s}$$

Die durch den Dopplereffekt verschobene Frequenz ergibt sich aus

$$f_{oben} = 1000 \cdot (1 + \frac{47,6}{340}) = 1140Hz$$

und

$$f_{unten} = 1000 \cdot (1 + \frac{47,6}{340}) = 860Hz$$

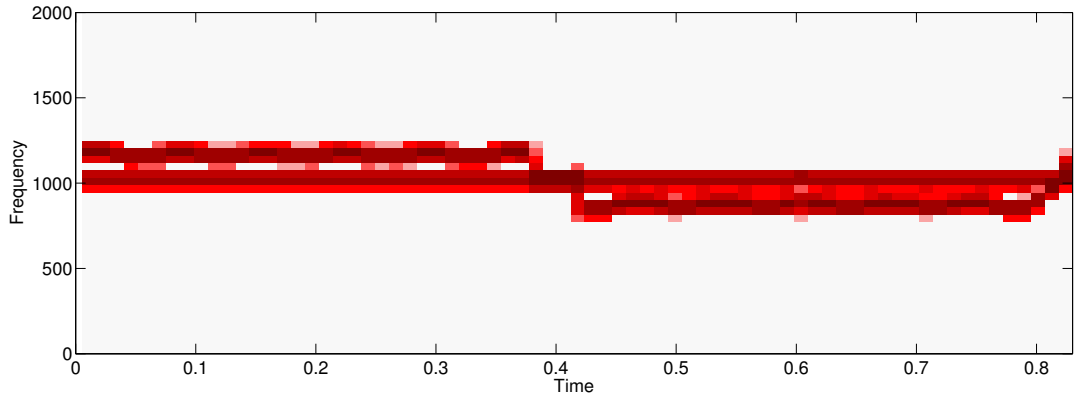
Diese Werte stimmen gut mit der Grafik überein. In 5.2(b) beträgt die maximale Entfernung 600m, woraus sich eine Frequenzverschiebung von

$$f = 1000 \cdot (1 + \frac{1440}{340}) = 5235Hz$$

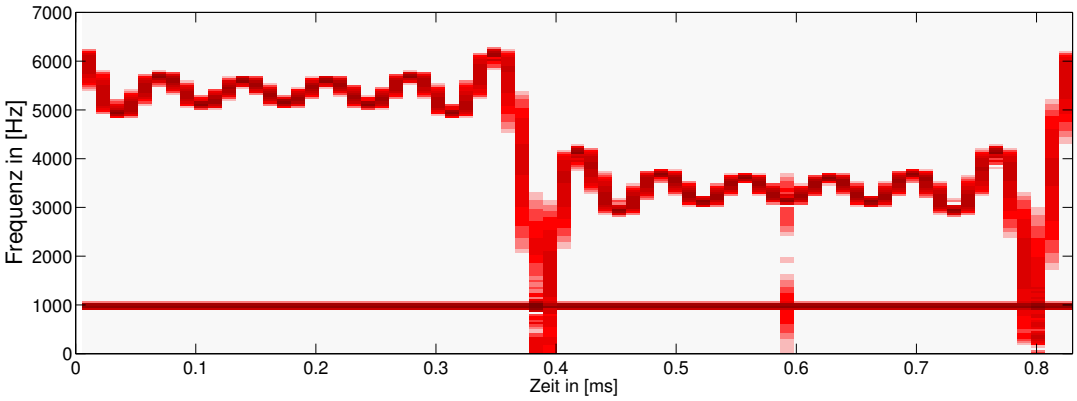
bzw.

$$f_{unten} = 1000 \cdot (1 - \frac{1440}{340}) = (-)3235Hz$$

Auch diese Berechnungen stimmten gut mit den aufgenommenen Spektrogrammen überein. In Abbildung 5.2(b) beträgt die Frequenz in der ersten Halbperiode ca. 5300Hz. In der zweiten Periode sind die berechneten (-)3235Hz deutlich zu erkennen. Negative Frequenzen werden an 0Hz gespiegelt, was rein grafisch mit einer Gleichrichtung vergleichbar ist.



(a) ohne Aliasing



(b) auftretendes Aliasing

Abbildung 5.2: *Verschiebung eines 1000Hz-Tones bei konstanter (dreieckförmiger) Wandbewegung*

5.2 Anwendungsbeispiele

5.2.1 Modellierung der Phsyik

Das Modul eignet sich gut, um die Physik zu modellieren. Der Dopplereffekt wirkt jedoch nur als solcher, wenn er bei Geräuschen, die aus dem Alltag bekannt sind, auftritt. Hier eignet sich das Beispiel des Martinshorns sehr gut. Der Dopplereffekt kann bereits bei sehr kleinen Frequenzverschiebungen erkannt werden. Anders ist dies bei Musik. Hier werden kleine Frequenzverschiebungen nicht wahrgenommen. Wird der Dopplereffekt hörbar, denkt man eher an eine Schallplatte, die nicht gleichmäßig läuft.

5.2.2 Klangverzerrung

Obwohl auch die Modellierung der Physik eine Verzerrung des Klanges darstellt, wird dieser Punkt extra angeführt. Im künstlerischen Bereich kann das Modul vielfältig eingesetzt werden, da die Effekte eine große Bandbreite haben. Diese reicht von einer mäßigen Verfremdung des Klanges bis zu seiner kompletten Umformung mit dem Verlust sämtlicher klanglicher Eigenschaften, sodass er nicht mehr wiedererkannt werden kann. Wie vorhin beschrieben, wirkt der Doppler-Effekt bei Musikstücken anders. Der Effekt der ungleichmäßig laufenden Schallplatte wird bei einer Erhöhung der Frequenz der Bewegung (von Schallquelle oder Wand) zu einem Effekt, wie er vom Scratchen bei DJs bekannt ist. Interessant ist hier der Wechsel zwischen aktiviertem und deaktiviertem Bypass im Takt zur Musik. Bei Live-Anwendung ist dies ein einfacher, aber sehr wirkungsvoller Effekt. Bei einer weiteren Erhöhung der Frequenz wird der Klang sehr stark verfremdet und der Effekt ist sehr abhängig vom Ausgangsmaterial. Wird eine sehr hohe Frequenz gewählt (>1000*100mHz) und die Amplitude klein gehalten, bleibt weiterhin der ursprüngliche Klang bzw. das Musikstück erkennbar. Der Effekt könnte am ehesten mit “Zittern” bzw. “Wobble” beschrieben werden.

6 Ausblick

6.1 Ideen zu Erweiterungen

6.1.1 Flaches Stereobild

Das entworfene Modul bietet derzeit sehr rudimentäre Funktionen und Darstellungen. Nimmt man ähnliche Plugins zum Vorbild, ist die nächste Erweiterungsstufe eine Implementierung des flachen Stereoklangs, also eine reine links-rechts Bewegung der Schallquelle mit einer Reflexion in der horizontalen Ebene. Das heißt, die Wand kann an beliebiger Stelle zwischen linkem und rechtem Rand positioniert werden. Das wäre somit eine Drehung der aktuellen Anordnung um 90°. Die Tiefe wird derzeit durch die Filterwirkung der Luft simuliert, diese müsste also weggelassen werden.

6.1.2 Stereobild mit Raumtiefe

Der nächste Schritt wäre, die Schallquelle nicht nur zwischen links und rechts, sondern auch in die Tiefe wandern zu lassen. Das ist mit einer erheblichen Veränderung der GUI verbunden, weil nun unendlich viele Pfade zur Auswahl stehen. Auch die Positionierung der Wand muss hierbei überlegt werden. Theoretisch kann diese nun beliebige Winkel einnehmen und den Schall spiegelbildlich (zB. von rechts hinten nach links vorne) zurückwerfen.

6.1.3 Surround Sound

Der Vollständigkeit halber wird hier noch die logische letzte Stufe erwähnt, nämlich eine Erweiterung zum Surround Sound. Die Quelle kann sich nun beliebig im Raum bewegen.

Die Umsetzung wird sich als sehr aufwändig erweisen und es ist wohl keine Erweiterung des bestehenden Moduls, sondern eher eine Neuentwicklung.

6.1.4 Entwicklung eines VST-Plugins

Um dieses Modul nicht nur in Pd, sondern in beliebigen anderen Programmen verwenden zu können, wäre eine Portierung in ein VST-Plugin nötig. Dort könnte die GUI auch umfangreicher und optisch ansprechender gestaltet werden.

6.2 Konkrete Verbesserungsmöglichkeiten

Bei dem Modul gibt es einige Probleme, die ich möglicherweise mit mehr Programmiererfahrung besser lösen hätte können.

6.2.1 Anzeige der aktuellen Positionen

Die Anzeige wird derzeit mit einem `hslider` angezeigt, welcher mit der Indexposition des Arrays mitfährt, in dem die jeweilige Kurvenform gespeichert ist. Gewünscht wäre eine Anzeige am Graphen gewesen, dass beispielsweise ein roter Punkt an der Kurve entlangfährt und die aktuelle Position der Wand oder der Quelle anzeigt. Die optisch ansprechendste Darstellung wäre eine Systemskizze ähnlich zu Abbildung 1.1, die sich in Echtzeit den entsprechenden Abständen anpasst. Hier stellt sich jedoch wieder die Frage, ob die Kosten für die aufgewendete Rechenleistung, welche einer komplexere grafische Anzeige erfordert, akzeptabel sind. Zweifelsohne ist die aktuelle Realisierung diejenige mit dem geringsten Rechenaufwand.

6.2.2 Artefakte

Beim Test mit einem reinen Sinus-Ton sind Artefakte (kurze Aussetzer) deutlich hörbar geworden. Mit großer Wahrscheinlicheit hängen sie mit dem variablen Delay zusammen, eine Korrektur war jedoch nicht möglich. Wie in Kapitel 2.3 erwähnt, ist beim Ändern der Positionen auch ein Dopplereffekt zu hören, wobei leider auch Störgeräusche auftreten. Mit

einer entsprechenden Interpolation der Steuer- oder Audiosignale wäre das Problem bestimmt lösbar. Mit Tiefpass- und Rampenfunktionen (`line`) konnte der Fehler jedoch nicht behoben werden.

6.2.3 Presets

Die Möglichkeit von Presets ist sehr attraktiv für Anwender, die an den Parametern feilen, um ganz bestimmte Effekte zu erzeugen und favorisierte Einstellungen öfters benutzen wollen. Ließen sich aktuelle Einstellungen beispielsweise in Textdateien speichern und daraus wieder einlesen, könnten sie in verschiedenen Instanzierungen umgehend wieder aufgerufen werden. Diese ließen sich auch per eMail oder USB-Stick einfach mit anderen Anwendern teilen. Weiters könnte das Modul selbst einige Presets mitliefern, um dem Anwender einige Beispiele vorzuführen und das Modul schneller einsatzbereit zu haben bzw. leicht konfigurieren zu können.

Literaturverzeichnis

[Dic97] Dickreiter, Michael: *Handbuch der Tonstudio Technik*. Nummer 1. K. G. Saur Verlag KG, München, 1997.

[dop] *http://de.wikipedia.org/wiki/Dopplereffekt*. 15.01.2011.

[Far08] Farnell, Andy: *Designing Sound*. Applied Scientific Press, 2008.

[Her05] Herczeg, Michael: *Software-Ergonomie. Grundlagen der Mensch-Computer-Kommunikation*. 2. Oldenbourg, 2005.

[Iso06] *EN ISO 9241 Ergonomie der Mensch-System-Interaktion*. 2006.

[MF10] Maria Fellner, Robert Höldrich: *Physiologische und Psychoakustische Grundlagen des räumlichen Hörens*. http://iem.at/projekte/publications/iem_report/report03_98/, 02.06.2010.

[Ous94] Ousterhout, J.K.: *Tcl and the Tk toolkit*. Addison-Wesley professional computing series. Addison-Wesley, 1994.

[Puc96] Puckette, Miller: *Pure Data*. International Computer Music Conference. San Francisco: International Computer Music Association, pp. 224-227, 1996.

[UZ05] Udo Zölzer, Martin Bossert, Norbert Fliege: *Digitale Audiosignalverarbeitung*. Vieweg+Teubner, 2005.

[WW07] Werner Weselak, Gerhard Graber: *Skript zur Vorlesung Raumakustik (LV-Nr. 441.150)*. Technische Universität Graz, 2007.

Abbildungsverzeichnis