Manipulation der Abstandswahrnehmung von virtuellen Schallquellen in Ambisonics-Systemen

Technische Universität Berlin
Fakultät I, Geisteswissenschaften
Fachgebiet Audiokommunikation

Masterarbeit

Vorgelegt von Alexander Poletajev
[redacted]

Erstgutachter: Prof. Dr. Stefan Weinzierl
Zweitgutachter: Henrik von Coler
Datum: 30. Juni 2017

Die selbständige und eigenhändige Anfertigung dieser Arbeit versichere ich Eides statt.

Ort, Datum	Alexander Poletajev

Abstract

In Rahmen dieser Arbeit wurde eine Software entwickelt, mit der man die Distanzwahrnehmung von virtuellen Schallquellen in Ambisonics-Systemen manipulieren kann. Das Wirkungsprinzip beruht auf dem Direktschall-Hall-Verhältnis und wurde in Form eines Hybrid Reverb umgesetzt. Dafür werden die Erstreflexionen mit Hilfe von Rotationen im Bereich der Kugelflächenfunktionen erzeugt und der Nachhall auf Grundlage eines Feedback Delay Networks synthetisiert. Die Software wurde als „Pure Data External“ und als Audioplugin implementiert. Darüber hinaus wurde für die Steuerung des Plugins eine OSC-Schnittstelle zu dem Virtual-Reality Controller „Leap Motion“ eingerichtet.

Inhaltsverzeichnis

1 Grundlagen **7**
1.1 Higher Order Ambisonics . . . 7
1.1.1 Konventionen . . . 8
1.2 Rotation von Kugelflächenfunktionen . . . 10
1.3 Abstandswahrnehmung . . . 10
1.4 Hybrid Reverb . . . 11
1.5 Feedback Delay Network . . . 14
1.6 Widening-Algorithmus . . . 17

2 Software Design **20**
2.1 Synthese von Erstreflexionen . . . 21
2.1.1 Berechnung der Rotationsmatrix $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ . . . 22
2.1.2 Impulsantwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ . . . 22
2.1.3 Verwendung als Erstreflexionen . . . 24
2.1.4 Ressourcenverbrauch und Optimierung . . . 27
2.2 Synthese von Nachhall . . . 28
2.3 Überblendung . . . 30
2.4 Ergebnisse . . . 31
2.4.1 Höreindruck . . . 34

3 Implementierung **37**
3.1 Implementierung in Pure Data . . . 37
3.2 Audio Plugin . . . 38
3.3 Implementierung in JUCE . . . 39
3.4 OSC Interface . . . 43

4 Ausblick und offene Fragen **46**
4.1 Software Design . . . 46
4.2 Implementierung . . . 47

Abbildungsverzeichnis

1.1 ACN Nummerierung bis $\mathcal{O}(3)$ 9
1.2 Schematische Darstellung einer Raumimpulsantwort 11
1.3 Signalflussdiagramm eines FDN 4ter Ordnung 14
1.4 (a) Transformationspaar einer phasenmodulierten Cosinus-Funktion, (b) Schema der x-,y-,z-Drehungen, (c) Trajektorie der Drehung 17
1.5 Neue Dispersionsform durch mehrfache Faltung bei verschiedenen Q 19
2.1 Allgemeiner Signalfluss 20
2.2 Signalflussdiagramm der Erstreflexionen 21
2.3 Impulsantwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},\mathtt{m}_i}$ für verschiedene `Q` 23
2.4 Antwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}$ auf verschiedene Frequenzen 24
2.5 Impulsantwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}$ für verschiedene Raumwinkel . . . 25
2.6 Impulsantwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}$ ohne Einschwingphase 25
2.7 Zeitliche Varianz der Raumwinkelzuordnung für ein Sinussignal mit $f = 100Hz$ 26
2.8 Signalflussdiagramm der Nachhallsynthese, Signale fließen von links nach rechts 28
2.9 Impulsantwort des FDN mit und ohne Einschwingphase 29
2.10 Ergebnis der Überblendung, Amplitudengang im Zeitbereich . 30
2.11 EDR für unterschiedliche $\mathtt{T}_{60,ratio}$ 31
2.12 EDR für verschiedene `Roomsize` 32
2.13 EDR der Erstreflexionen und des Nachhalls 33
2.14 EDR eines Seminarraums und der Software bei gleicher Nachhallzeit 33
2.15 EDR für unterschiedliche Nachhallpegel 34
2.16 Bildlicher Vergleich der möglichen Abstandsempfindungen mit und ohne ER (In der xy-Ebene, Hörer im Ursprung); **Blau**: Eindeutige Empfindung der Position, **weiss**: Keine Empfindung der Position 35
3.1 AmbVerb-External in Pure Data 37
3.2 GUI von AmbVerb in Reaper 40
3.3 Vereinfachte Darstellung der C++ Klassen 41
3.4 Ambisonics Produktion in Reaper 43
3.5 LeapMotion Handerkennung 44

Tabellenverzeichnis

1 Aktuelle Forschungsarbeiten zu Hybrid Reverb 13
2 Die gängigsten Plugin-Formate 39

Abkürzungen

IR Impulsantwort

ER Erstreflexionen

Cue Anhaltspunkt für akustische Wahrnehmung

FDN Feedback Delay Network

EDR Energy Decay Relief

HOA Higher Order Ambisonics

PD Pure Data

GUI Graphische Benutzeroberfläche

OSC Open Sound Control

DSP Digitaler Signalprozessor

DAW Digital Audio Workstation

FFT Fast Fourier Transform

Einleitung

In den letzten Jahrzehnten ist die elektroakustische Synthese von Schallfeldern verstärkt in das Interesse der Forschung gerückt. Eines der wichtigsten Verfahren dafür ist „Higher Order Ambisonics“(HOA) [17][18][24]. HOA ermöglicht es, mithilfe sphärischer Lautsprecheranordnungen, virtuelle Schallquellen in allen Raumrichtungen zu verteilen. Die mathematischen Modelle bieten zwar auch die Möglichkeit, die genaue Position festzulegen, jedoch reicht es nicht aus, um dem Hörer einen Eindruck von der Distanz zur virtuellen Schallquelle zu vermitteln. Denn die zugrunde liegenden Modelle gehen von Freifeldbedingungen aus, während das menschliche Hörsystem Distanzeindrücke vor allem aus raumakustischen Informationen gewinnt. Der wichtigste Anhaltspunkt ist dabei das Direktschall-Hall-Verhältnis [1][21].
Im Rahmen dieser Arbeit wird ein Audioplugin vorgestellt, in dem das Direktschall-Hall-Verhältnis ausgenutzt wird, um die Abstandswahrnemung von virtuellen Schallquellen in Ambisonics-Systemen zu manipulieren. Das Ziel ist dabei nicht, möglichst hohe Ansprüche an physikalische Akkuratheit zu erfüllen, sondern einen auditiven Effekt zu erzeugen. Das Plugin, mit der Bezeichnung „**AmbVerb**“, beruht auf einem Hybrid Reverb [2][11][9], das die Erstreflexionen und den Nachhall auf Basis verschiedener Algorithmen erzeugt. Während der Nachhall mit Hilfe der altbewährten Technik eines Feedback-Delay-Network [27] synthetisiert wird, kommt zur Erzeugung der Erstreflexionen ein neues Prinzip zum Einsatz. Das „Phantom-Source-Widening“ [25] basiert auf Rotationen des virtuellen Schallfeldes im Bereich der Kugelflächenfunktionen und scheint für die Erzeugung von Erstreflexionen in Ambisonics geeignet zu sein.
AmbVerb dient dazu, virtuelle Schallquellen perzeptiv im dreidimensionalen Raum zu positionieren, deshalb entsteht der Bedarf nach einem Interface, mit dem sich die Position intuitiv steuern lässt. Zu diesem Zweck wird der Virtual-Reality Controller „Leap Motion[1]“ in Verbindung mit OSC-Nachrichen verwendet. Mit dessen Hilfe, lassen sich die Distanz und der Raumwinkel der virtuellen Schallquelle mit einer Handbewegung kontrollieren.

[1] www.leapmotion.de

1 Grundlagen

1.1 Higher Order Ambisonics

Es gibt unterschiedliche mathematische Formulierungen von HOA. Letztendlich basieren alle auf der Tatsache, dass ein Schallfeld im Innern eines beliebigen quellenfreien Volumens V, eindeutig über die Feldgrößen auf dem Rand δV beschrieben werden kann [17]. Demnach lassen sich theoretisch alle möglichen Schallfelder $P(\mathbf{x}, \omega)$, $\mathbf{x} \epsilon V$ durch gezielte Kontrolle der Randwerte $P(\delta V, \omega)$ erzeugen.
Unter Voraussetzung von Freifeldbedingungen ergibt sich ein Schallfeld, das durch die Abstrahlung von Monopolen auf δV erzeugt wird aus:

$$P(\mathbf{x}, \omega) = \oint_{\partial V} G(\mathbf{x}|\mathbf{x}_0, \omega) D(\mathbf{x}_0, \omega) dA(\mathbf{x}_0), [18] \tag{1}$$

mit der Kreisfrequenz ω, dem Flächendifferential dA, der GREENschen Funktion $G(\mathbf{x}|\mathbf{x}_0, \omega)$ eines Monopols und dessen Position $\mathbf{x}_0 \epsilon \partial V$. Um ein Schallfeld exakt nachzubilden, muss die passende Treiberfunktion $D(\mathbf{x}_0, \omega)$ gefunden werden. Im Rahmen von Ambisonics wird Gl.(1) explizit gelöst. Dazu wird das Schallfeld, von sphärischen Lautsprecheranordnungen ausgehend, in Kugelflächenfunktionen zerlegt.

$$P(\mathbf{r}, \omega) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} -ikj_n(kr)h_n(kr_L)Y_n^m(\theta)a_{nm}, \tag{2}$$

mit der Wellenzahl k, den Bessel- und Hankelfunktionen $j_n(kr)$ und $h_n(kr_L)$, den Kugelflächenfunktionen $Y_n^m(\theta)$, dem Raumwinkel θ sowie den Expansionskoeffizienten a_{nm}. Die Lösung ergibt sich dann mit Hilfe eines Koeffizientenvergleichs.

Bei der praktischen Umsetzung treten jedoch einige Schwierigkeiten auf. Zum einen wird eine hohe Anzahl von Lautsprechern benötigt[2], zum anderen treten Probleme wie „ill-posedness“, „non-uniqueness“ und hohe Ansprüche an die genauen Lautsprecherpositionen auf [3][17]. Eine HOA-Variante, die diese Probleme umgeht, ist das „All-Round Ambisonics Panning und Decoding“[24] (ALLRAP und ALLRAD) welches die Treiberfunktion mit Hilfe

[2] bei einem Lautsprecherabstand von 0.41m ensteht „spatial-aliasing“ ab einer Frequenz von 420Hz [18]

von „modal-source-strength-matching“[26] bestimmt. Bei der Verwendung von „modal-source-strength-matching“ wird nicht mehr der Versuch unternommen ein bestimmtes Schallfeld physikalisch nachzubilden, sondern ein perzeptiver Ansatz gewählt. Man beschränkt sich auf die Wiedergabe virtueller Quellen, die sich auf ∂V befinden. Dazu geht man davon aus, dass die Helmholtzgleichung[3] von einer kontinuierlichen Quellstärke $f(\theta)$ auf ∂V angeregt wird.

$$(\Delta + k^2)p = -f(\theta)\frac{\delta(r - r_{\partial V})}{r^2}$$

Diese Differentialgleichung wird durch Gl.(2) gelöst. Die Expansionskoeffizienten a_{nm} stammen dabei aus der Transformation von $f(\theta)$.

$$a_{nm} = \int_{\partial V} f(\theta) Y_n^m(\theta) d\theta, \tag{3}$$

Für die Wiedergabe auf einer realen Lautsprecheranordnung, wird die diskrete Quellstärke $\hat{f}(\theta) = \sum_{l=1}^{L} g_l \delta(1 - \theta^\intercal \theta_L)$ mit Hilfe der gains g_l an die kontinuierliche Quellstärke angepasst:

$$f(\theta) \stackrel{!}{=} \hat{f}(\theta)$$

$$\Leftrightarrow a_{nm} \stackrel{!}{=} \sum_{l=1}^{L} g_l Y_n^m(\theta_L),$$

1.1.1 Konventionen

Das bisher am weitesten verbreitete Format zur Standardisierung von Ambisonics war das B-Format (FuMA)[8]. Aufgrund der Beschränkung bis $\mathcal{O}(3)$ und maximalen Dateigrößen von 4GB, ist es mittlerweile veraltet. Das vorgeschlagene AmbiX-Format [10] vermeidet solche Restriktionen und setzt sich als Standard durch. Die wichtigsten Konventionen betreffen dabei die Normierungskoeffizienten der Kugelflächenfunktionen und die Kanalreihenfolge. Bei der Entwicklung eines Schallfeldes nach Kugelflächenfunktionen treten Normierungskoefizienten $N_n^{|m|}$ auf.

$$Y_n^m(\varphi, \vartheta) = N_n^{|m|} P_n^{|m|}(\sin(\vartheta)) \begin{cases} \sin(|m|\varphi) & m < 0 \\ \sin(|m|\varphi) & m \geq 0 \end{cases}$$

[3] Wellengleichung unter Annahme harmonischer Lösungen: $(\Delta + k^2)p = 0$

Für deren Wahl stehen verschiedene Möglichkeiten (FuMa, N3D, SN3D) bereit. Im Rahmen des Ambix-Formats wird die SN3D Normierung der Form

$$N_n^{|m|} = \sqrt{\frac{2-\delta_m}{4\pi}\frac{(n-|m|)!}{(n+|m|)!}}$$

gewählt. Der Vorteil dieser Normierung ist, dass der Expansionskoeffizient nullter Ordnung stets den höchsten Wert aufweist. Im Rahmen dieser Arbeit wird von dieser Tatsache noch Gebrauch gemacht.

Für die Kanalreihenfolge gab es bisher auch einige Varianten (FuMa, SID, ACN) zu Auswahl. Hier verwendet Ambix die ACN Reihenfolge. Diese ist konsistent mit der üblich verwendeten Reihenfolge der Kugelflächenfunktionen in der Mathematik und algorithmisch nummerierbar gemäß:

$$ACN = n^2 + n + m$$

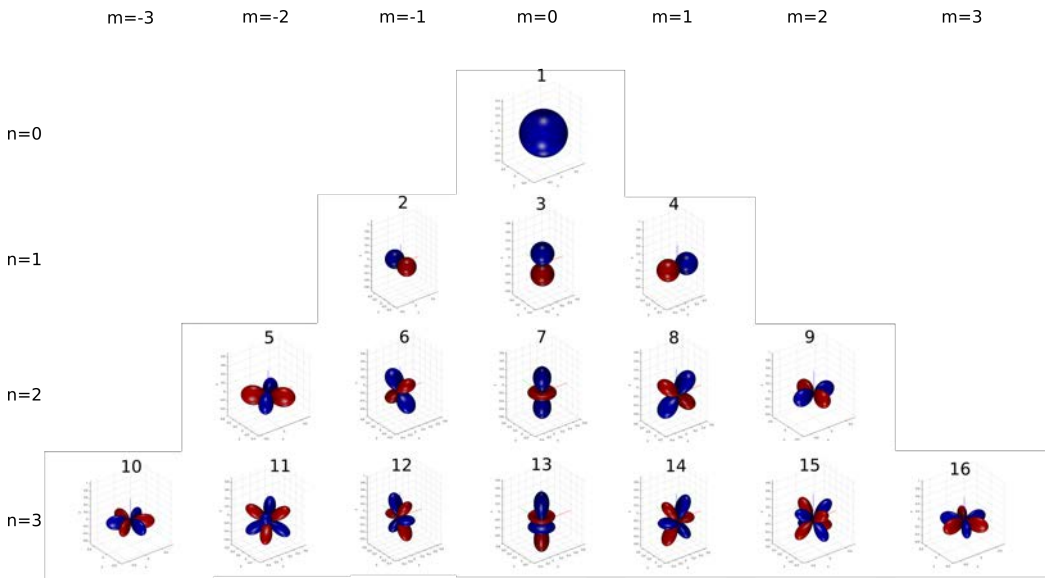

Abbildung 1.1: ACN Nummerierung bis $\mathcal{O}(3)$

1.2 Rotation von Kugelflächenfunktionen

Nach Zotter [23] betrifft die Rotation der Kugelflächenfunktionen um die z-Achse nur Expansionskoeffizienten mit gleichem Index m.

$$\begin{bmatrix} b_m[q] \\ b_{-m}[q] \end{bmatrix} = \begin{bmatrix} \cos m\psi & -\sin m\psi \\ \sin m\psi & \cos m\psi \end{bmatrix} \cdot \begin{bmatrix} a_m[q] \\ a_{-m}[q] \end{bmatrix} \tag{4}$$

Demnach lassen sich leicht Rotationsmatrizen für beliebige Ordnungen definieren. Für $\mathcal{O}(2)$ zum Beispiel, hat die gesamte Rotationsmatrix folgende Form:

$$T^z(\psi) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \cos\psi & 0 & \sin\psi & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\sin\psi & 0 & \cos\psi & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cos 2\psi & 0 & 0 & 0 & \sin 2\psi \\ 0 & 0 & 0 & 0 & 0 & \cos\psi & 0 & \sin\psi & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\sin\psi & 0 & \cos\psi & 0 \\ 0 & 0 & 0 & 0 & -\sin 2\psi & 0 & 0 & 0 & \cos 2\psi \end{bmatrix} \tag{5}$$

Rotationsmatrizen um andere Raumachsen sind deutlich aufwendiger zu bestimmen. Deswegen schlägt Zotter vor, Drehungen um beliebige Raumachsen aus einer Kombination von festen 90° Drehungen um die y-Achse und variablen Drehungen um die z-Achse zusammenzusetzen. Zur Berechnung der $T^y(90°)$ Matrizen hat er ein Konzept hergeleitet, dass auf Wiederholungseigenschaften und Symmetrien der y-Rotationsmatrizen beruht.

$$T^{xyz}(\phi, \theta, \psi) = T^z(\phi + 45°)\, T^y(90°)\; T^z(\theta + 180°)\, T^y(90°)\, T^z(\psi + 45°)$$

1.3 Abstandswahrnehmung

Dem menschlichen Gehör stehen zur Wahrnehmung des Abstands von einem Schallereignis unterschiedliche Anhaltspunkte (Cues) zur Verfügung. Nach Zahorik [21] sind die wichtigsten Cues die Intensität, das Direktschall-Hall-Verhältnis, das Spektrum und binaurale Cues. Die Schallintensität als Cue ergibt sich aus dem Abstandsgesetz $L_2 = L_1 - 20\log\frac{r_2}{r_1}$ (Der Direktschallpegel nimmt um 6dB je Abstandsverdopplung ab). Das Spektrum als Cue basiert auf der Luftabsorbtion, vornehmlich bei hohen Frequenzen. Binaurale Cues

gewinnen im Nahfeld einer Quelle an Bedeutung, da dort größere Unterschiede zwischen den Direktschallsignalen in beiden Hörkanälen entstehen. Diese sind für Lautsprecherwiedergabe jedoch irrelevant. Im Fernfeld ist der wichtigste Anhaltspunkt das Direktschall-Hall-Verhältnis (direct-reverberant ratio, D/R). Es steht dem Menschen in reflektierenden Umgebungen zur Verfügung und sinkt mit größer werdendem Abstand.

$$D/R = \frac{\int_0^T h^2(t)dt}{\int_T^\infty h^2(t)dt},$$

mit der Integrationszeit $T = (2-3)ms$.
Albrecht und Lokki [1] haben gezeigt, dass es Sinnvoll sein kann eine Erweiterung des D/R zu verwenden. Um der Wahrnehmung von Erstreflexionen und Nachhall besser gerecht zu werden, schlagen sie das „early-to-late energy ratio“ vor. Bei diesem Cue wird die Integrationszeit auf $T = 80ms$ gesetzt. Dieser Cue ist verwandt mit dem Klarheitsmaß C_{80} aus der Raumakustik [19].

1.4 Hybrid Reverb

Nachhall entsteht in reflektierenden Umgebungen und ist die Grundlage für raumakustische Wahrnehmung. Es existieren viele unterschiedliche Ansätze zur Erzeugung künstlichen Nachhalls. Davon basieren manche auf Filter, andere beruhen auf Faltung mit Impulsantworten oder auf Raumsimulation. Allen haben jedoch miteinander gemein, dass sie die Reflexionseigenschaften von Räumen nachahmen.

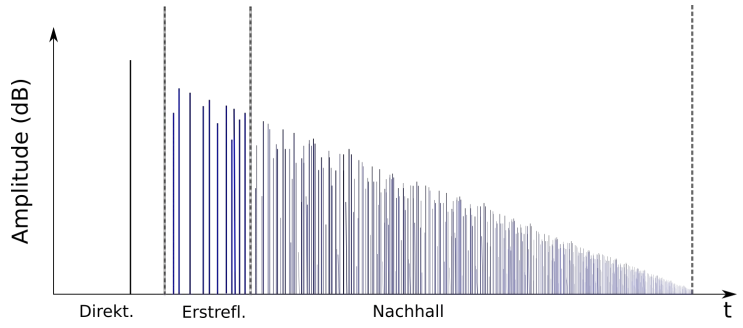

Abbildung 1.2: Schematische Darstellung einer Raumimpulsantwort

Ein räumliches Schallereignis unterteilt sich zeitlich in drei Abschnitte: Direktschall, Erstreflexionen und Nachhall (siehe Abb.1.2). Zuerst erreicht der

Direktschall den Hörer und kurz darauf folgen die ersten Reflexionen. Mit zunehmender Zeit steigt die Reflexionsdichte so weit an, bis die Schallenergie homogen im Raum verteilt ist, diesen Abschnitt nennt man Nachhall. In allen drei Zeitabschnitten sind unterschiedliche Informationen enthalten. Aus dem Direktschall erfährt der Hörer die Richtung, in der das Hörereignis stattfindet. Die Erstreflexionen lassen sich vom Hörsystem auflösen und enthalten Informationen über Geometrie und Materialien im Raum. Im Bereich des Nachhalls sind keine geometrischen Informationen mehr enthalten, das Schallfeld ist von einer normalverteilten Amplitudenverteilung geprägt und weist ein Leistungsspektrum auf, dessen Abklingverhalten von der Größe des Raums und der Absorbtionsfläche abhängt [15].

Durch moderne Echtzeitanwendungen, wie Computerspiele und Virtual-Reality, steigt der Bedarf nach real klingendem und flexiblen Nachhall mit minimalen Rechenaufwand. Da Erstreflexionen und Nachhall unterschiedliche Eigenschaften haben, stellen sie verschiedene Anforderungen an den Syntheseprozess. Um den Rechenaufwand zu minimieren, kann es deshalb sinnvoll sein, beide Teile mithilfe verschiedener Methoden zu synthetisieren. Ein solches Softwaredesign nennt man Hybrid Reverb.
Die am meisten verwendeten Methoden, die zur Erzeugung künstlichen Nachhalls zur Verfügung stehen, lassen sich in drei Gruppen unterteilen [15]:

Faltung mit Raumimpulsantworten

Durch Faltung kann die Akustik genau definierter Räume nachgebildet werden. Dafür müssen Impulsantworten(IR) bereit stehen, die entweder aus akustischen Messungen an realen Räumen oder aus Raumsimulationen stammen. Mit dieser Strategie lassen sich realistische Erstreflexionen erzeugen. Ein Nachteil liegt in hohem Rechenaufwand, insbesondere für IRs mit großer Nachhallzeit. Außerdem ist Faltung nicht sehr flexibel, da für neue Raumeigenschaften andere IRs benötigt werden. Deswegen eignet sich Faltung vor allem für die Erzeugung von Erstreflexionen.

Raumsimulation

Mit Hilfe von Raumsimulation kann Hall erzeugt werden, der genau definierten Raumeigenschaften und Positionen entspricht. Besonders Erstreflexionen lassen sich damit akkurat erzeugen. Auch diese Methode ist für große Nachhallzeiten mit einem sehr hohen Rechenaufwand verbunden und in einem Hybrid Reverb eher für die Erzeugung von

Erstreflexionen geeignet. Die wichtigsten Strategien zur akustischen Simulation sind Image-Source-Method, Beamtracing und Raytracing.

Feedback Delay Network

Feedback Delay Network (FDN) ist die am häufigsten verwendete Methode zur Synthese von künstlichem Nachhall. Die Vorteile von FDN sind Flexibilität und wenig Rechenaufwand. Nachhallzeit, sowie Raumgröße und Absorbtionsverhalten lassen sich mit FDN in Echtzeit modellieren und haben keinen Einfluss auf die benötigte Rechenkapazität. Der wesentliche Nachteil ist, dass die Qualität der Erstreflexionen aus FDN für viele Anforderungen nicht ausreicht. Im Falle eines Hybrid Reverb eignet sich FDN sehr gut zur Erzeugung des Nachhallteils, denn gerade hier liegen die Stärken der Methode.

Daraus ist ersichtlich, dass es sinnvoll ist die Erstreflexionen durch Faltung oder Simulation zu synthetisieren und den Nachhall aus einem FDN zu gewinnen. Dieser Sachverhalt wird auch bei einem Blick auf aktuelle Forschungsarbeiten (Tab.1.2) deutlich. Bei allen Arbeiten wird der Nachhall in einem FDN erzeugt, während die Erstreflexionen durch Faltung oder Simulation enstehen.

Wissenschaftliche Arbeit	**Erstreflexionen**	**Nachhall**
Primavera, 2013 [11]	Faltung IR	FDN
Carpentier, 2014 [2]	Faltung IR	FDN
Wendt, 2014 [20]	Image-Source-Method	FDN
Risheng, 2015 [14]	Faltung IR	FDN

Tabelle 1: Aktuelle Forschungsarbeiten zu Hybrid Reverb

Ein kritischer Teil im Design eines Hybrid Reverb beschäftigt sich damit, auf welche Weise die Signale der Erstreflexionen durch die Signale des Nachhalls abgelöst werden (Überblendung). Dazu muss ein Zeitpunkt(t_{mix}) sinnvoll gewählt werden, bei dem die Überblendung durchgeführt wird. Für t_{mix} stehen verschiedene Auswahlkriterien zur Verfügung, die von Lindau [7] evaluiert wurden. Alle Auswahlkriterien gehen davon aus, dass eine Impulsantwort eines realen oder simulierten Raums vorliegt. Der einfachste Weg ist die Annahme, t_{mix} hinge nur vom Raumvolumen ab. $t_{mix} = \sqrt{V}$. Eine andere Möglichkeit besteht darin, den Zeitpunkt zu wählen, ab dem die Impulsantwort eine normalverteilte Amplitudenverteilung angenommen hat.

Zusammengefasst wird ein Hybrid Reverb in drei Prozesse unterteilt: Die Erzeugung von Erstreflexionen, die Erzeugung von Nachhall und die Überblendung, die die beiden Signale bei der „mixing time“ t_{mix} verknüpft.

Im Rahmen dieser Arbeit wird für die Erzeugung von Erstreflexionen eine neue Idee verwendet, die insbesondere auf Ambisonics Systeme anwendbar ist. Dabei handelt es sich um den Widening-Algorithmus von Zotter und Kronlachner [22]. Der Algorithmus ist eigentlich zur Ausdehnung virtueller Schallquellen in Ambisonics bestimmt und wird zum ersten Mal für die Synthese von Erstreflexionen benutzt. (siehe Kapitel 1.6) Der Nachhall wird mit Hilfe eines FDN erzeugt. Die Wahl von t_{mix} wird durch die speziellen Eigenschaften des Widening-Algorithmus bestimmt (siehe Kapitel 2.3)

1.5 Feedback Delay Network

Feedback-Delay-Networks (FDN) haben ihren Ursprung in einer Arbeit von Gerzon (1967 [4]) und wurden seitdem weiterentwickelt bis zur verallgemeinerten Formulierung von Jot (1991 [5]).

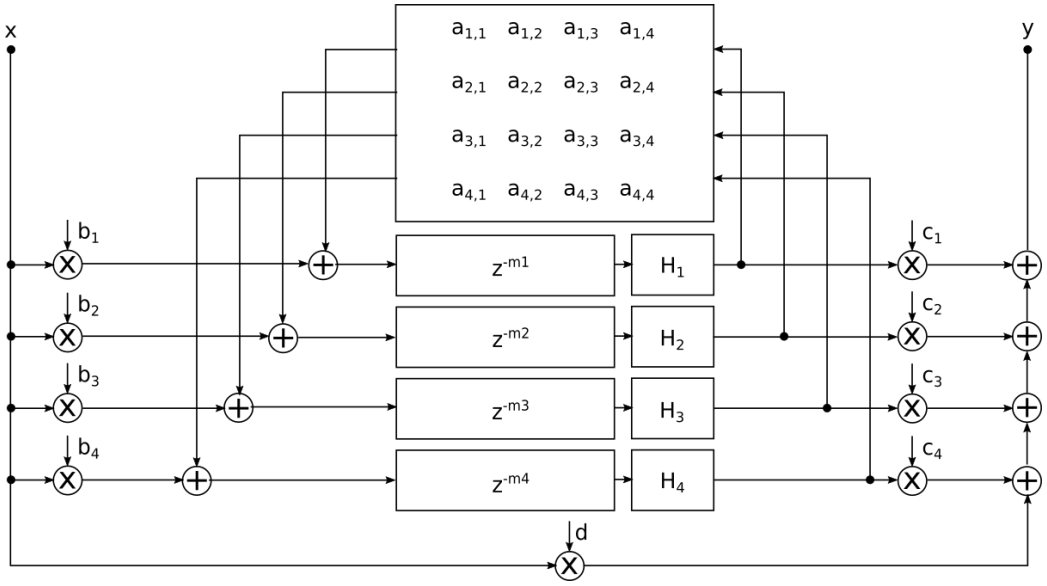

Abbildung 1.3: Signalflussdiagramm eines FDN 4ter Ordnung

Der Aufbau eines FDN N-ter Ordnung wie in Bild(1.3), basiert auf N Delayspuren, von denen jede um $\tau_i = m_i T_s$ mit $\mathbf{T}_s = 1/f_s$ verzögert und über

eine Feedbackmatrix auf alle Spuren zurückgeführt wird. Die Filterstruktur wird durch folgende Gleichungen beschrieben [27]:

$$y_i(n) = \sum_{i=1}^{N} c_i s_i(n) + dx(n)$$

$$s_i(n + m_i) = \sum_{j=1}^{N} a_{ij} s_j(n) + b_i x(n)$$

Dabei ist $s_i(n)$ das n-te Sample der i-ten Delayspur. Für $m_i = 1$ erhält man die ausführlich untersuchte Zustandsraumdarstellung eines diskreten, linearen Übertragungssystems. Daraus kann die Überragungsfunktion des FDN bestimmmt werden:

$$H(z) = \frac{Y(z)}{X(z)} = \mathbf{c}^T[\mathbf{D}(z^{-1}) - \mathbf{A}]^{-1}\mathbf{b} + d$$

mit der Delaymatrix $\mathbf{D}(z)$ =diag$(z^{-m_1}, z^{-m_2}, ..., z^{-m_N})$ und der Feedbackmatrix $\mathbf{A} = [a_{ij}]_{N\times N}$. Die Stabilitätseigenschaften der Filterstruktur hängen allein von der Feedbackmatrix ab. Wenn $\|A\|^n$ exponentiell abfällt ist die Stabilität des Systems gesichert. Die Pole des FDN ergeben sich als Lösungen von

$$\mathbf{det}[\mathbf{A} - \mathbf{D}(z^{-1})] = 0$$

Damit alle Pole auf dem Einheitskreis liegen, ist es sinnvoll eine unitäre Feedbackmatrix zu wählen. In diesem Fall ist das System verlustlos mit unendlicher Nachhallzeit. In der Praxis ist das ein guter Ausgangspunkt. Hat man erst ein stabiles, verlustloses FDN aufgebaut, kann man Dämpfungsfaktoren $g_i = \alpha_i^m$ einfügen um die Nachhallzeit festzulegen. Das entspricht einer Substitution von $\mathbf{D}(z)$ nach $\mathbf{D}(z/\alpha)$ und führt dazu, dass alle Pole gleichmäßig zum Ursprung gezogen werden. Dies garantiert ein gleichmäßiges Abklingverhalten für alle Frequenzen und bestimmt die Nachhallzeit zu

$$T_{60} = \frac{-3T_s}{\log(\alpha)}$$

Um der Tatsache gerecht zu werden, dass in der Realität hohe Frequenzen schneller Abklingen als tiefe, kann man anstelle der Dämpfungsfaktoren Tiefpassfilter in die Delayspuren einfügen. Durch ein Filterdesign entsprechend

$$H_i(z) = G^{m_i}(z) \tag{6}$$

bleibt dabei ein gleichmäßiges Abklingverhalten benachbarter Frequenzen erhalten. Mit der frequenzabhängigen Nachhallzeit $\mathbf{T}_{60}$ folgt aus Gl.(6):

$$|H_i(e^{j\omega T})|^{\frac{T_{60}(\omega)}{m_i T}} = 0.001 \tag{7}$$

Tiefpass Filter Design

Ein einfaches Filterdesign, mit dem sich die Nachhallzeit für tiefe und hohe Frequenzen getrennt festlegen lässt, wird von Smith [16] beschrieben. Dabei werden die Koeffizienten eines Filters erster Ordnung so gewählt, dass sich die gewünschten Nachhallzeiten $\mathbf{T}_{60}(\omega = 0)$ und $\mathbf{T}_{60}(\omega = \pi/T)$ einstellen. Für einen Tiefpass erster Ordnung mit Übertragungsfunktion

$$H_i(z) = \frac{g_i}{1 - p_i z^{-1}} \tag{8}$$

ist die Verstärkung bei $(\omega = 0)$ gleich $H_i(1) = \frac{g_i}{1-p_i}$ und bei $(\omega = \pi/T)$ gleich $H_i(-1) = \frac{g_i}{1+p_i}$. Mit Gl.(7) ergibt sich daraus folgendes Gleichungssystem:

$$\frac{g_i}{1 - p_i} = 10^{-3m_i T/T_{60}(0)} \mathrel{\widehat{=}} R_0^{m_i} \tag{9}$$

$$\frac{g_i}{1 + p_i} = 10^{-3m_i T/T_{60}(\pi/T)} \mathrel{\widehat{=}} R_\pi^{m_i} \tag{10}$$

mit den Lösungen:

$$g_i = \frac{R_\pi^{m_i} - R_\pi^{m_i}}{R_\pi^{m_i} + R_\pi^{m_i}} \tag{11}$$

$$p_i = \frac{2R_\pi^{m_i} R_\pi^{m_i}}{R_\pi^{m_i} + R_\pi^{m_i}} \tag{12}$$

In der Praxis ist es sinnvoller die Nachhallzeit für tiefe Frequenzen festzulegen und die Nachhallzeit für hohe Frequenzen über ein Verhältnis auszudrücken:

$$T_{60,ratio} = \frac{T_{60}(\pi/T)}{T_{60}(0)} \tag{13}$$

1.6 Widening-Algorithmus

Der Widening-Algorithmus wurde von Zotter und Kronlachner [22] für die Ausdehnung virtueller Schallquellen in Ambisonics entwickelt. Das Wirkungsprinzip beruht auf frequenzabhängigen Rotationen des Schallfeldes im Bereich der Kugelflächenfunktionen.
Bei einer Drehung um die z-Achse, ergeben sich neue Expansionskoeffizienten durch folgende Operation (siehe Kapitel(1.2)):

$$\begin{bmatrix} b_m[q] \\ b_{-m}[q] \end{bmatrix} = \begin{bmatrix} \cos(m\zeta) & -\sin(m\zeta) \\ \sin(m\zeta) & \cos(m\zeta) \end{bmatrix} \cdot \begin{bmatrix} a_m[q] \\ a_{-m}[q] \end{bmatrix}$$

Im Falle eines frequenzabhängigen Drehwinkels $\zeta = \hat{\phi}\cos(\Omega)$, mit Modulationsamplitude $\hat{\phi}$ und normierten Kreisfrequenz $\Omega = \omega T$, wird die Operation wie folgt im Spektralbereich ausgedrückt.

$$\begin{bmatrix} \mathcal{F}(b_m)[\Omega] \\ \mathcal{F}(b_{-m})[\Omega] \end{bmatrix} = \begin{bmatrix} \cos(m\hat{\phi}\cos(\Omega)) & -\sin(m\hat{\phi}\cos(\Omega)) \\ \sin(m\hat{\phi}\cos(\Omega)) & \cos(m\hat{\phi}\cos(\Omega)) \end{bmatrix} \cdot \begin{bmatrix} \mathcal{F}(a_m)[\Omega] \\ \mathcal{F}(a_{-m})[\Omega] \end{bmatrix} \tag{14}$$

Um die Rotation im Zeitbereich zu verstehen, müssen wir zuerst das folgende Transformationspaar untersuchen. Definiert man einen zeitinvarianten

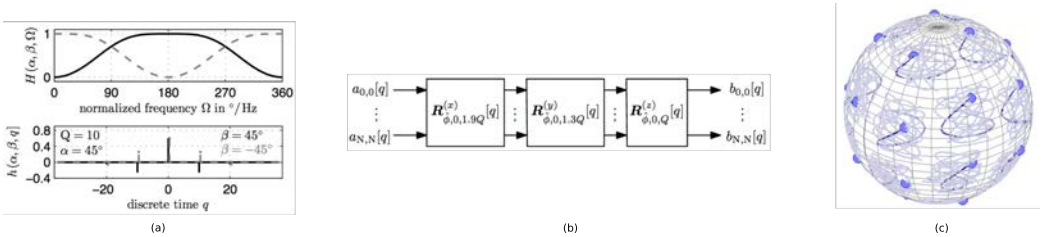

Abbildung 1.4: (a) Transformationspaar einer phasenmodulierten Cosinus-Funktion, (b) Schema der x-,y-,z-Drehungen, (c) Trajektorie der Drehung

Frequenzgang in Form eines phasenmodulierten Cosinus (siehe Abb.1.4(a)) mit normierter Kreisfrequenz $\Omega = \omega T$, dann erhält man eine dünn besetzte Impulsantwort im Zeitbereich.

$$\begin{aligned} H(\alpha,\beta,\Omega) &= \cos(\alpha\cos(\Omega)+\beta), \\ \Leftrightarrow \mathcal{F}^{-1}(H(\alpha,\beta,\Omega)) &= \mathcal{F}^{-1}(\cos(\alpha\cos(\Omega)+\beta)) \\ \Leftrightarrow \qquad h(\alpha,\beta,t) &= \sum_{\lambda=-\infty}^{\infty} \cos(\tfrac{\pi}{2}\,|\lambda|+\beta) J_\lambda(\alpha)\delta(t+\lambda) \end{aligned}$$

Die einfachste diskrete Implementierung $h(\alpha, \beta, q]$ (siehe Abb.1.4(a)) nutzt den Sampleindex q und ist ungleich null für $q = \lambda Q$.

$$h(\alpha, \beta, \lambda Q] = \cos(\tfrac{\pi}{2}\ |\lambda| + \beta) J_\lambda(\alpha) \tag{15}$$

Dieses achsensymmetrische IIR Filter ist nicht kausal. Durch Abschneiden der Impulsantwort bei $|\lambda| = \Lambda$ und Verschiebung des Wertebereichs zu $0 \leq \lambda \leq 2\Lambda$, erhalten wir ein kausales FIR Filter mit $2\Lambda + 1$ Einträgen im Zeitintervall $q \in [0, 2\Lambda Q]$.
Durch Vergleich von Gl.(15) mit Gl.(14) folgt, dass frequenzabhängige Drehungen im Zeitbereich, durch Faltung mit der Matrix $\mathbf{R}^m_{\hat{\phi},Q}[q]$, erreicht werden können.

$$\begin{bmatrix} b_{n,m}[q] \\ b_{n,-m}[q] \end{bmatrix} = \mathbf{R}^m_{\hat{\phi},Q}[q] * \begin{bmatrix} a_{n,m}[q] \\ a_{n,-m}[q] \end{bmatrix}$$

mit

$$\mathbf{R}^m_{\hat{\phi},Q}[q] = \begin{bmatrix} h(m\hat{\phi}, 0, (\lambda - \Lambda)Q] & h(m\hat{\phi}, \frac{\pi}{2}, (\lambda - \Lambda)Q] \\ h(m\hat{\phi}, -\frac{\pi}{2}, (\lambda - \Lambda)Q] & h(m\hat{\phi}, 0, (\lambda - \Lambda)Q] \end{bmatrix} \tag{16}$$

Bisher wurden nur Drehungen um die z-Achse beschrieben. Schallquellen, die nah an der z-Achse positioniert sind, werden von diesen Drehungen kaum beeinflusst. Mit Hilfe von Koordinatentransformationen lässt sich die Drehung jedoch auf alle Raumachsen anwenden. Dazu muss man nur die x- oder y-Achse auf die z-Achse abbilden und nach der Drehung wieder zurück transformieren. Unter Voraussetzung solcher Transformationsmatrizen $\mathbf{R}^{(z\cdot)}$ und $\mathbf{R}^{(\cdot z)}$ können wir Drehungen um x- und y-Achse folgendermaßen definieren:

$$\mathbf{R}^{(z)}_{\hat{\phi},0,Q}[q] = [\mathbf{R}^m_{\hat{\phi},0,Q}[q]]_{n,m} \tag{17}$$

$$\mathbf{R}^{(y)}_{\hat{\phi},0,Q}[q] = \mathbf{R}^{(yz)}[q]\ \mathbf{R}^{(z)}_{\hat{\phi},0,Q}[q]\ \mathbf{R}^{(zy)}[q] \tag{18}$$

$$\mathbf{R}^{(x)}_{\hat{\phi},0,Q}[q] = \mathbf{R}^{(xz)}[q]\ \mathbf{R}^{(z)}_{\hat{\phi},0,Q}[q]\ \mathbf{R}^{(zx)}[q] \tag{19}$$

$$\tag{20}$$

$$\mathbf{R}^{(x,y,z)}_{\hat{\phi},0,Q}[q] = \mathbf{R}^{(x)}_{\hat{\phi},0,1.9Q}[q] * \mathbf{R}^{(y)}_{\hat{\phi},0,1.3Q}[q] * \mathbf{R}^{(z)}_{\hat{\phi},0,Q}[q] \tag{21}$$

Diese Matrizen kann man gemäß Gl.(22) mit dem Eingangssignal Vektor falten und erhält Drehungen in 3 Dimensionen. Unterschiedliche Zeitraster $[1.9Q, 1.3Q, Q]$ der x,y,z Drehungen führen zu einer Drehung entlang einer frequenzabhängigen Trajektorie, die einen Raumwinkel von $\approx 2\hat{\phi}$ gleichmäßig ausfüllt.(Bild 1.4(b,c))

$$\mathbf{b}[q] = \mathbf{R}^{(x,y,z)}_{\hat{\phi},0,Q}[q] * \mathbf{a}[q] \tag{22}$$

Die Form der Dispersionskurve kann verändert werden, indem man $h(m\hat{\phi}, m\varphi_0, (\lambda - \Lambda)Q]$ für verschiedene Q miteinander faltet. (siehe Abb.1.5)

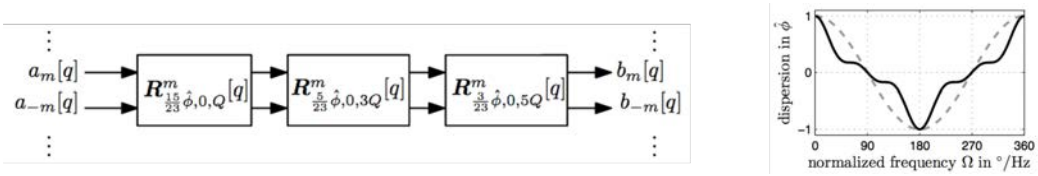

Abbildung 1.5: Neue Dispersionsform durch mehrfache Faltung bei verschiedenen Q

2 Software Design

Die Software folgt dem Ambix-Standard und wird zwischen Encoder und Decoder einer Ambisonics Produktionsumgebung eingebunden. Auf diese Weise ist sie von dem Lautsprechersystem entkoppelt und auf eine Vielzahl von Ambisonics-Systemen anwendbar. An den Eingängen werden entsprechend SN3D normierte Ambisonics-Signale in ACN Reihenfolge erwartet[4]. Das hy-

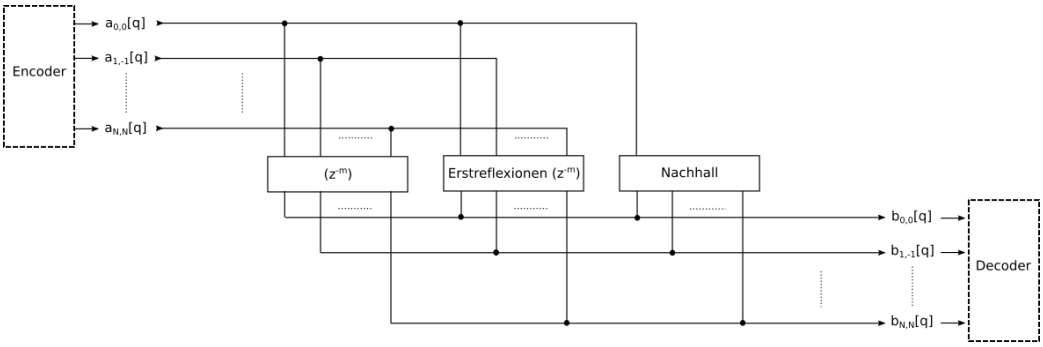

Abbildung 2.1: Allgemeiner Signalfluss, Signale fließen von links nach rechts

brid Reverb unterteilt sich in drei unterschiedliche Signalströme und die zugehörigen Algorithmen: Der Direktschall, die Erstreflexionen und der Nachhall. Der Direktschall bleibt in seiner Form unverändert. Die Erstreflexionen werden mit Hilfe des Widening-Algorithmus in einer „Multiple Input Multiple Output“(MIMO) Struktur auf Grundlage aller Eingangssignale erzeugt. Der Nachhall entsteht in einer „Single Input Multiple Output“(SIMO) Struktur, die als Eingang nur den Ambisonics-Kanal nullter Ordnung verwendet. Das macht Sinn, weil in SN3D Normierung $a_{00}[q]$ dem unkodierten Eingangssignal entspricht. Für die Überblendung wird die Einschwingphase des Nachhalls entfernt und die Signale des Direktschalls und der Erstreflexionen verzögert, damit am Ausgang des Systems eine einheitliche Impulsantwort anliegt.

Die Kontrollparameter der Software unterteilen sich in Parameter zur Festlegung der Raumeigenschaften und Parameter zur Manipulation der Distanzwahrnehmung. Die Raumeigenschaften werden durch folgende Parameter gesetzt: Die Nachhallzeiten $\mathtt{T}_{60}(0)$ und $\mathtt{T}_{60}(\pi/T)$ sind maßgeblich mit der

[4]Falls der verwendete Encoder/Decoder nicht auf dem Ambix-Standard basiert, müssen die Ein-/Ausgänge transformiert werden. Dazu existieren bereits frei erhältliche Plugins, zB. in der Ambix-PluginSuite [6].

theoretischen Absorbtionsfläche verknüpft. `Roomsize` kann als Raumgröße verstanden werden und legt die Reflexionsdichte fest. Die übrigen Parameter bestimmen den auditiven Abstand zur virtuellen Schallquelle, nämlich die Lautstärke der Erstreflexionen, des Nachhalls und des Direktschalls. Die Lautstärke des Direktschalls wird durch den inversen Parameter `Distance` ersetzt. Um so größer der Wert `Distance` ist, desto leiser der Direktschall und kleiner das Direktschall-Hall-Verhältnis. Die Lautstärken der Erstreflexionen und des Nachhalls können unabhängig voneinander eingestellt werden, so ist künstlerischer Freiraum gegeben um auch unrealistische Eindrücke erzeugen zu können.

2.1 Synthese von Erstreflexionen

Die Verwendung des Widening-Algorithmus zur Erzeugung von Erstreflexionen wurde bisher noch nicht getestet. Die Tatsache, dass das Widening die Bestandteile eines Eingangssignals auf verschiedene Raumwinkel verteilt und dabei eine dünn besetzte Impulsantwort aufweist lässt vermuten, dass es dafür geeignet ist. Trotzdem bringt es Eigenschaften mit, die für die Erzeugung von Erstreflexionen als hinderlich zu bewerten sind und muss für diese Anwendung angepasst werden.
Das Signalflussdiagramm der Erstreflexionen ist in Abb.2.2 zu sehen. Die

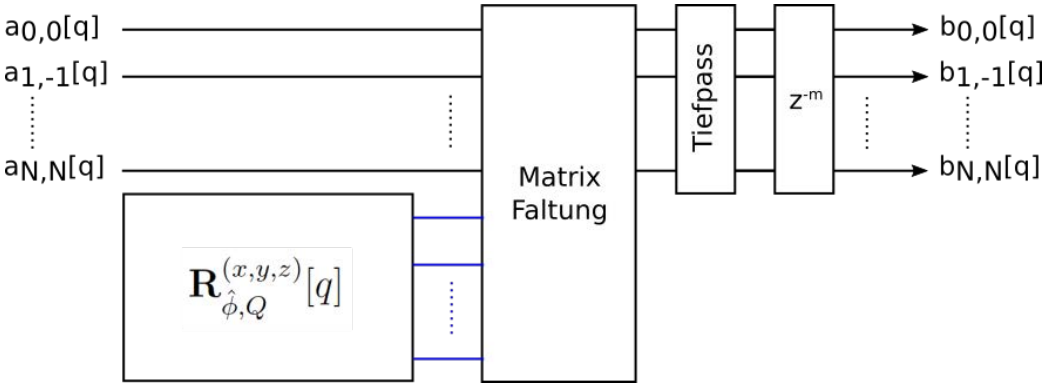

Abbildung 2.2: Signalflussdiagramm der Erstreflexionen, Signale fließen von links nach rechts

Ambisonics-Kanäle werden mit der dreidimensionalen Rotationsmatrix $\mathbf{R}_{\hat{\phi},Q}^{(x,y,z)}[q]$ gefaltet und danach, zur Vorbereitung auf die Überblendung, durch einen

Tiefpass geleitet und verzögert. Obwohl der Signalfluss sehr einfach dargestellt ist, verbraucht dieser Teil der Software die meisten Ressourcen und ist konzeptionell am anspruchsvollsten. Der kritische Teil des Algorithmus liegt in der Berechnung der Rotationsmatrix sowie in der Herausforderung die hohe Anzahl an Faltungen in Echtzeit durchzuführen. Außerdem müssen mehrfach heuristische Designentscheidungen getroffen werden, um die Impulsantwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ als Erstreflexionen für das Hybrid Reverb verwenden zu können. Die Modulationsamplitude $\hat{\phi}$ wird fest auf 80° gestellt, das führt zu Rotationen innerhalb eines Raumwinkels von 160° und wird als angemessen vorausgesetzt. Die Beschneidung wird auf $\Lambda = 8$ festgelegt. Dieser Wert wurde als guter Kompromiss zwischen Verlustfreiheit und Rechenaufwand bewertet.

2.1.1 Berechnung der Rotationsmatrix $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$

Um $\mathbf{R}^{(x,y,z)[q]}_{\hat{\phi},Q}$ zu berechnen, werden Gl.(18-21) ausgeführt. Als erstes wird die z-Rotationsmatrix $\mathbf{R}^{(z)}_{\hat{\phi},Q}[q]$ mit den Einträgen aus $\mathbf{R}^{m}_{\hat{\phi},Q}[q]$ (siehe Gl.(16)) für alle m gefüllt. Danach wird sie durch Multiplikation mit den Transformationsmatrizen $\mathbf{R}^{(z\cdot)}$ und $\mathbf{R}^{(\cdot z)}$ auf die x- und y-Achse abgebildet. Die endgültige Rotationsmatrix erhält man als Faltungsprodukt der drei Zwischenergebnisse.

2.1.2 Impulsantwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$

In Abb.2.3 sieht man die Impulsantworten von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ auf allen Ambisonics-Kanälen bei verschiedenen Parametern `Q`. Offensichtlich hat `Q` keinen Einfluss auf die Grundform der IR, sondern führt zu Streckung oder Stauchung entlang der Zeitachse. Interpretiert man die Einträge als Reflexionen, dann lässt sich mit Hilfe von `Q` die Reflexionsdichte festlegen. Deshalb wird `Q` an den Kontrollparameter `Roomsize` gekoppelt.

Durch Faltung eines Eingangssignal-Vektors mit $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ wird die virtuelle Schallquelle nach Frequenzen in verschiedenen Richtungen verteilt. In Abb.2.4 ist diese Zuordnung für einige Frequenzen dargestellt. Bemerkenswert ist dabei das augenscheinlich verlustfreie Ergebnis trotz $\Lambda = 8$. In Abb.2.5 sieht man die Impulsantwort für verschiedene Winkelpositionen der

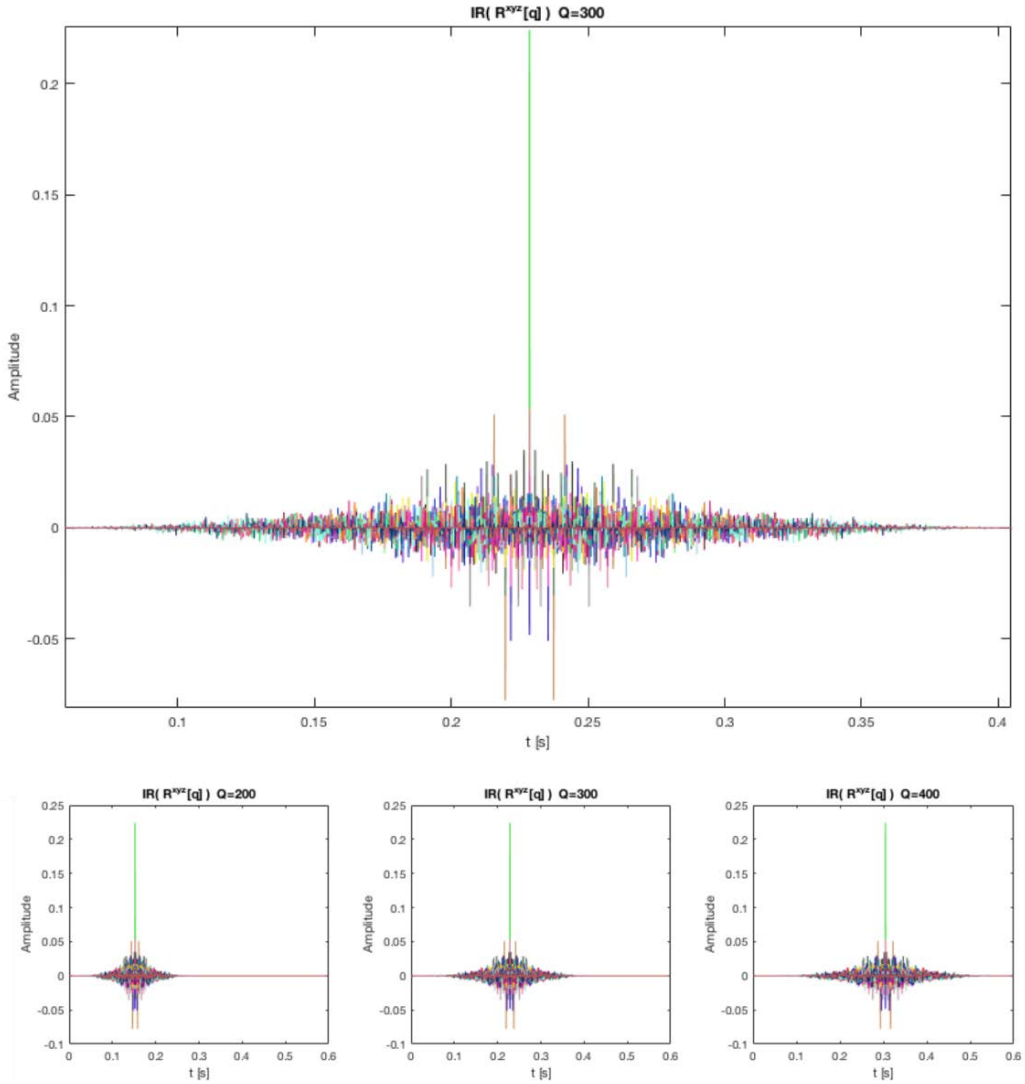

Abbildung 2.3: IR von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ für verschiedene `Q` ($\mathcal{O}(5)$, $\Lambda = 8$, $\phi = 80°$)

virtuellen Schallquelle. Der Algorithmus arbeitet zwar ohne die Information, in welchem Raumwinkel die Schallquelle kodiert wurde, jedoch führen unterschiedliche virtuelle Raumwinkel zu verschiedenen IRs. Das liegt daran, dass sich die Rotationen, je nach Position, anders zusammensezten. Dieses Verhalten ist für die Erzeugung von Erstreflexionen als positiv zu bewerten, da die Erstreflexionen in realen Räumen ebenfalls von der Sender-Empfänger-

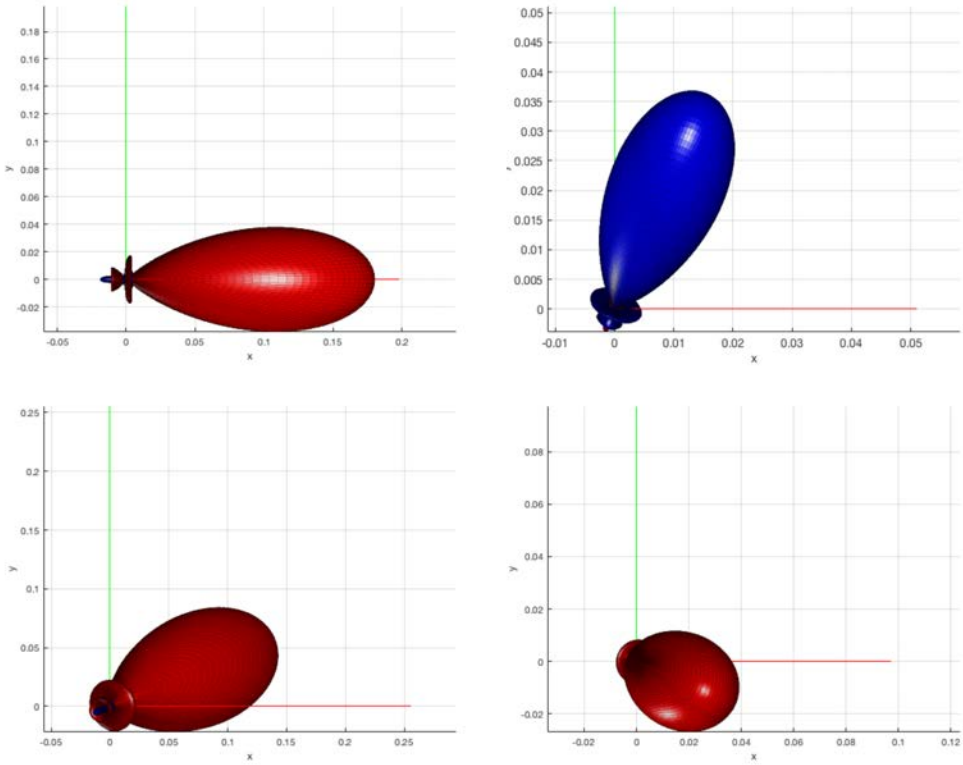

Abbildung 2.4: Antwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ auf verschiedene Frequenzen; Im Uhrzeigersinn: Eingangssignal ohne Faltung, 100Hz, 800Hz, 10kHz

Position abhängen. Der damit verbundene geometrische Eindruck ist jedoch unbekannt.

2.1.3 Verwendung als Erstreflexionen

Allen Impulsantworten ist die symmetrische Form mit gleichlangem Ein- und Ausschwingen gemeinsam. Das ist ein Problem, denn so ein Einschwingverhalten ist in realen Raumimpulsantworten nicht gegeben. Diese unerwünschte Symmetrie wird durch Beschneidung der IRs in $\mathbf{R}^{(xyz)}_{\hat{\phi},Q}[q]$ entfernt (siehe Abb.2.6). Dadurch ergibt sich die Frage, welchen Einfluss diese Beschneidung auf die Rotationen hat. Abb.2.7 zeigt, dass die Ausrichtung des Kegels in Abhängigkeit der Frequenz, als Folge der fehlenden Einschwingphase, nicht mehr zeitlich konstant ist, sondern mit der Periode der Signalfrequenz zwi-

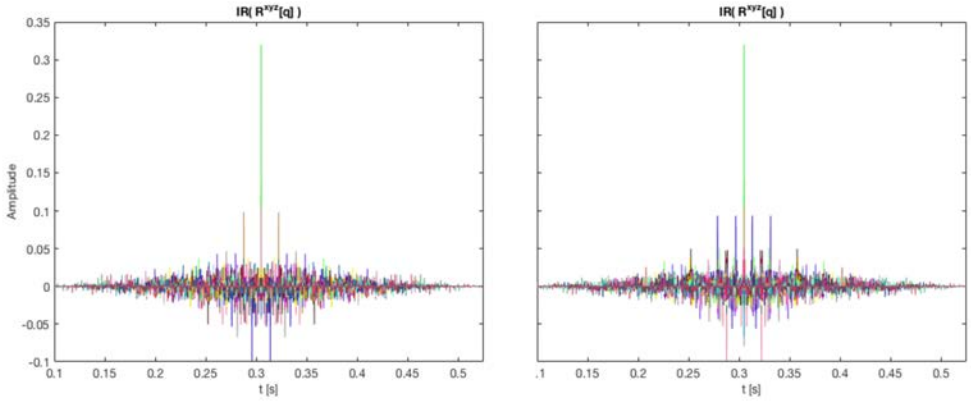

Abbildung 2.5: IR von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ (`Q` = 400) für verschiedene Raumwinkel; links: Azimuth=$\frac{\pi}{2}$, Elevation=0; rechts: Azimuth=0, Elevation=$\frac{\pi}{2}$

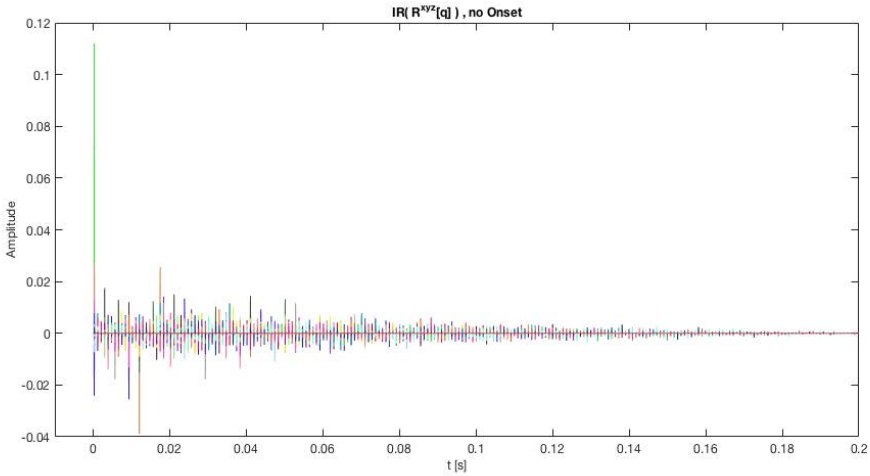

Abbildung 2.6: IR von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ ohne Einschwingphase, `Q` = 400

schen zwei verschiedenen Zuständen hin und her schwingt. Dieses Verhalten ist als negativ zu bewerten, weil dadurch die Einfallsrichtung der Reflexionen verschmiert. In wie weit das den Höreindruck beeinflusst, kann nicht ohne weiterführende Untersuchungen geklärt werden.

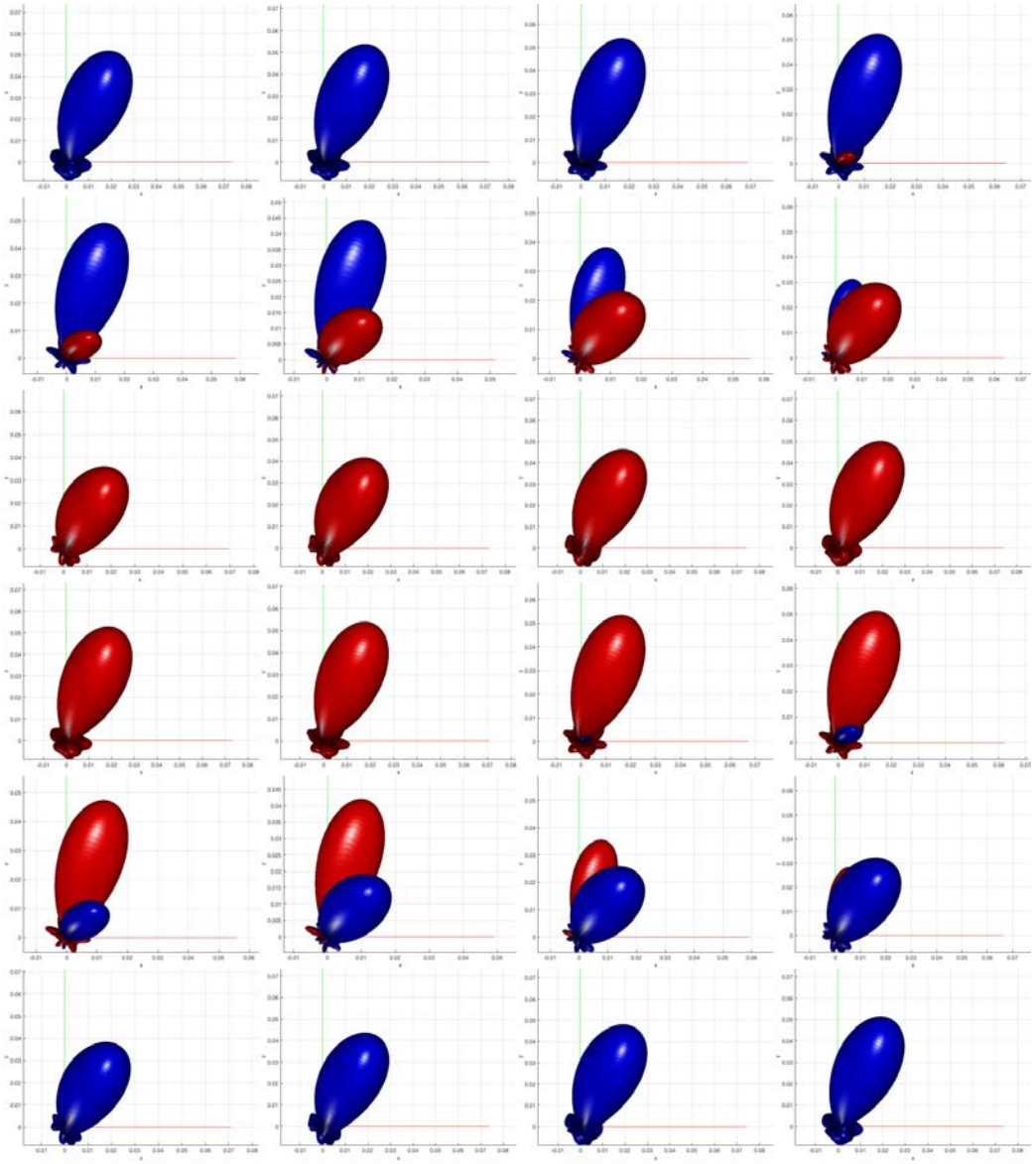

Abbildung 2.7: Zeitliche Varianz der Raumwinkelzuordnung für ein Sinussignal mit $f = 100Hz$, von links oben bis rechts unten vergeht eine Signalperiode

2.1.4 Ressourcenverbrauch und Optimierung

Die meiste Rechenleistung und Speicherkapazität wird für die Erzeugung der Erstreflexionen benötigt. Im Fall eines Ambisonics-Systems 5ter Ordnung besteht die Rotationsmatrix aus 36x36 Impulsantworten. Das führt zu 1296 Faltungen pro DSP Zyklus. Ein weiteres Problem ist die benötigte Puffergröße, mit der die Signale in die Faltung eingehen. Für das vorgeschlagene Zeitraster $[1.9\mathtt{Q}, 1.3\mathtt{Q}, \mathtt{Q}]$ der x,y,z-Drehungen hat die Implusantwort ohne Einschwingphase eine Länge von

$$N_{ER}(\mathtt{Q}, \Lambda) = (1 + 1.3 + 1.9)\mathtt{Q}\Lambda \tag{23}$$

Für $\mathtt{Q} = 400$ und $\Lambda = 8$ ergibt sich demnach eine Puffergröße von ca. 2^{14} Samples. Bei diesen Verhältnissen steigt die benötigte Blockgröße schnell an. Um Ressourcen zu sparen, werden alle Faltungen der Software im Spektralbereich als Multiplikationen durchgeführt(Schnelle Faltung). Außerdem zeigt sich bei näherer Analyse, dass nicht alle 36x36 IRs Einträge $\neq 0$ besitzen. Durch Auslassen dieser Faltungen wird der Rechenaufwand deutlich verringert. Zur Orientierung dient der Studiorechner[5] der TU-Berlin als Referenz. Um bei $\mathcal{O}(5)$ eine maximale Blockgröße von 8192 Samples nicht zu überschreiten, muss $\mathtt{Q} \leq 400$ gelten. Als untere Grenze hat sich $\mathtt{Q} = 200$ als geeignet gezeigt. Deshalb wird `Q` über den Parameter `Roomsize` aus einem Bereich von [200, 400] ausgewählt. Es gibt weitere Möglichkeiten den Ressourcenverbrauch zu senken, dazu mehr in Kapitel 4.

[5] Mac Pro 5.1, Quad-Core Intel Xeon, 2.4 GHz, 8GB RAM

2.2 Synthese von Nachhall

Abb.2.8 zeigt das Signalflussdiagramm der Nachhallsynthese. Diese unterteilt sich in zwei Abschnitte. Der Hauptsignalstrom(schwarz) fließt in Echtzeit und reichert das Eingangssignal mit Nachhall an. Der Blaue Teil dient dazu, die Impulsantwort des FDN von der Einschwingphase zu befreien. Die Ordnung des FDN wird im folgenden fest auf 16 gestellt, dieser Wert hat sich bei subjektiver Beurteilung als ausreichend erwiesen.(für Ambisonics $\mathcal{O}(5)$)

Das FDN entspricht den Ausführungen in Kapitel (1.5), stellt jedoch soviele Ausgänge bereit, wie bei jeweiliger Ambisonics-Ordnung benötigt werden. Diese werden direkt an den Delayspuren der Feedbackschleife abgegriffen. In

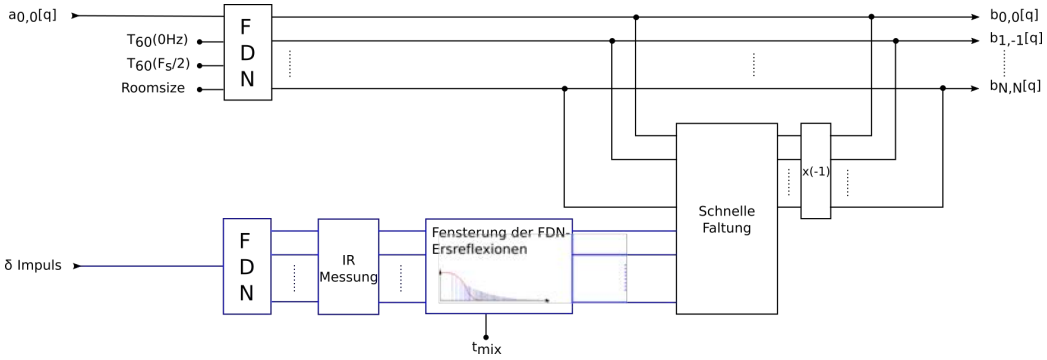

Abbildung 2.8: Signalflussdiagramm der Nachhallsynthese, Signale fließen von links nach rechts

dem Fall, dass die Anzahl der Ausgänge die Ordnung des FDN übersteigt, werden die verfügbaren Delayspuren mehrfach auf die Ausgänge verteilt. Bei einer sehr großen Anzahl von Ambisonicskanälen ist es möglich, dass die Nachhallsignale auf den Lautsprechern korreliert sind, in diesem Fall sollte die Ordnung des FDN erhöht werden. Als Feedbackmatrix dient eine Hadamardmatrix, die in Matlab mit der Funktion hadamard() berechnet wurde.
Die Delayzeiten $\mathtt{m}_i$ der jeweiligen Feedbackspur werden über den Kontrollparameter `Roomsize` eingestellt. Sie werden aus den Primzahlen im Intervall [`Roomsize`, `2.5Roomsize`] mit möglichst gleichmäßigem Abstand ausgewählt. Der Faktor 2.5 sorgt dafür, dass sich die wiederkehrenden Feedbackintervalle ohne zeitliche Lücken einander anschließen. Der Wertebereich von `Roomsize` wird auf 1000 bis 2000 Samples begrenzt. Die Dauer der Einschwingphase verhält sich proportional zur `Roomsize`.

Abb.2.9 zeigt eine Impulsantwort des FDN bei $\mathcal{O}16$, das FDN ist nach ca. 200ms eingeschwungen. Der Eingangsimpuls und der überwiegende Teil der

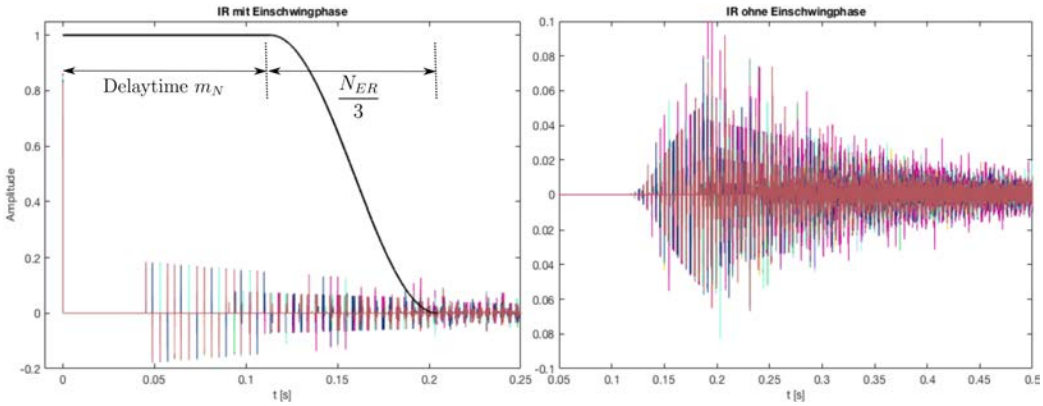

Abbildung 2.9: links: IR des FDN bei $\mathcal{O}(16)$, $\mathtt{T}_{60} = 2s$; rechts: IR ohne Einschwingphase

Einschwingphase werden zur Vorbereitung auf die Überblendung entfernt. Dieser Prozess ist im Signalflussdiagramm(Abb.2.8) blau dargestellt. Die Impulsantwort des FDN wird gemessen und der Anfangsteil, bestehend aus Eingangssignal und Einschwingphase, gefenstert[6]. Dieser Teil wird im Anschluss mit dem Hauptsignal gefaltet und davon subtrahiert. Auf diese Weise liegt am Ausgang des Systems nur der Nachhall ohne Einschwingphase an (siehe Abb.2.9, rechts).
Die Impulsantworten des FDN müssen immer dann neu gemessen werden, wenn sich die Kotrollparameter ($\mathtt{T}_{60}(0)$, $\mathtt{T}_{60}(\pi/T)$, `Roomsize`) ändern.

[6]Die Form des Fensters wurde in Anlehnung an ein Von Hann Fenster gewählt

2.3 Überblendung

Die Verwendung von Widening in einem Hybrid Reverb birgt große Herausforderungen für die Überblendung. Die gewöhnlichen Auswahlkriterien für t_{mis} orientieren sich an der physikalischen Realität. Die Signale, die normalerweise als Erstreflexionen in einem Hybrid Reverb verwendet werden, stammen aus Messungen oder Simulationen und erfüllen die Definitionen von Raumimpulsantworten. In diesen Fällen lässt sich t_{mix} zum Beispiel anhand der steigenden Reflexionsdichte oder statistischen Amplitudenverteilung bestimmen.
Die IR des Widening-Alorithmus erfüllt diese Eigenschaften nicht. Die Reflexionsdichte ist konstant und die Amplitudenverteilung nimmt keine Gaussche Form an. Zudem klingt sie von selbst je nach `Roomsize` innerhalb von 100 bis 200 ms ab und lässt deshalb nicht viel Spielraum für das Einsetzen des Nachhalls. Aus diesem Grund wird für die Überblendung ein pragmatischer Weg verfolgt.

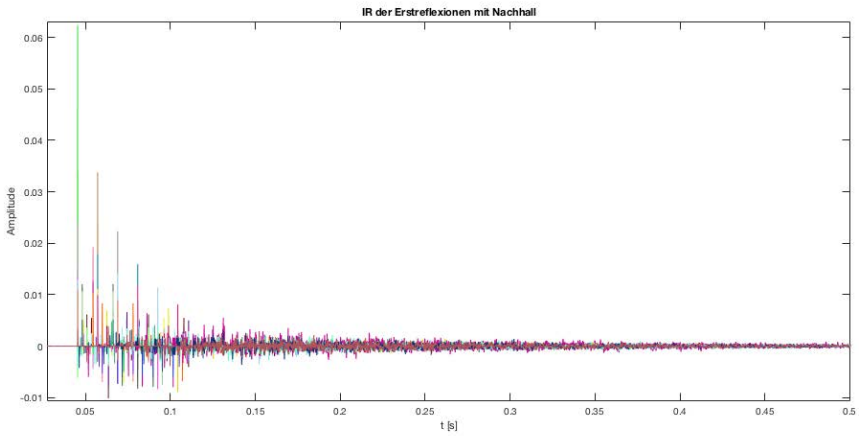

Abbildung 2.10: Ergebnis der Überblendung bei $\mathtt{T}_{60}(0) = 2s$, $\mathtt{T}_{60,ratio} = 0.15$ und $\mathtt{Roomsize} = 2000$ ($\mathtt{m}_i \in [2000, 4000]$; $\mathtt{Q} = 400$)

Die IR von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ geht nach dem Entfernen der Einschwingphase ganz, mit einer Länge von N_{ER} Samples (siehe Gl.(23)), in die Gesamtimpulsantwort des Hybrid Reverb ein. Der Nachhall wird von der Einschwingphase befreit und mit den anderen Teilen zu dem Gesamtsignal zusammengesetzt. Die Grenzen der Fensterung der FDN-Einschwingphase werden so gewählt,

dass der erste Feedbackdurchlauf ganz entfernt wird und das übrige Signal seinen vollen Pegel nach $\frac{N_{ER}}{3}$ erreicht (siehe Abb.2.9). Das Lautstärkeverhältnis zwischen Erstreflexionen und Nachhall ist dabei offen gelassen und wird vom Anwender seinen Wünschen entsprechend eingestellt. Ein Beispielergebnis ist in Abb.2.10 dargestellt. Dem Anschein nach liefert die Software annehmbare Ergebnisse.

2.4 Ergebnisse

Die Parameter $\mathtt{T}_{60}(0)$ und $\mathtt{T}_{60,ratio}$ prägen maßgeblich das spektrale Abklingverhalten („Energy Decay Relief“(EDR)) der endgültigen Impulsantwort. In Abb.2.11 ist das EDR für einige Nachhallzeiten zusammengestellt. Man sieht, dass die eingestellten Parameter $\mathtt{T}_{60}(0)$ und $\mathtt{T}_{60,ratio}$ erfüllt werden und benachbarte Frequenzen gleichmäßig Abklingen.

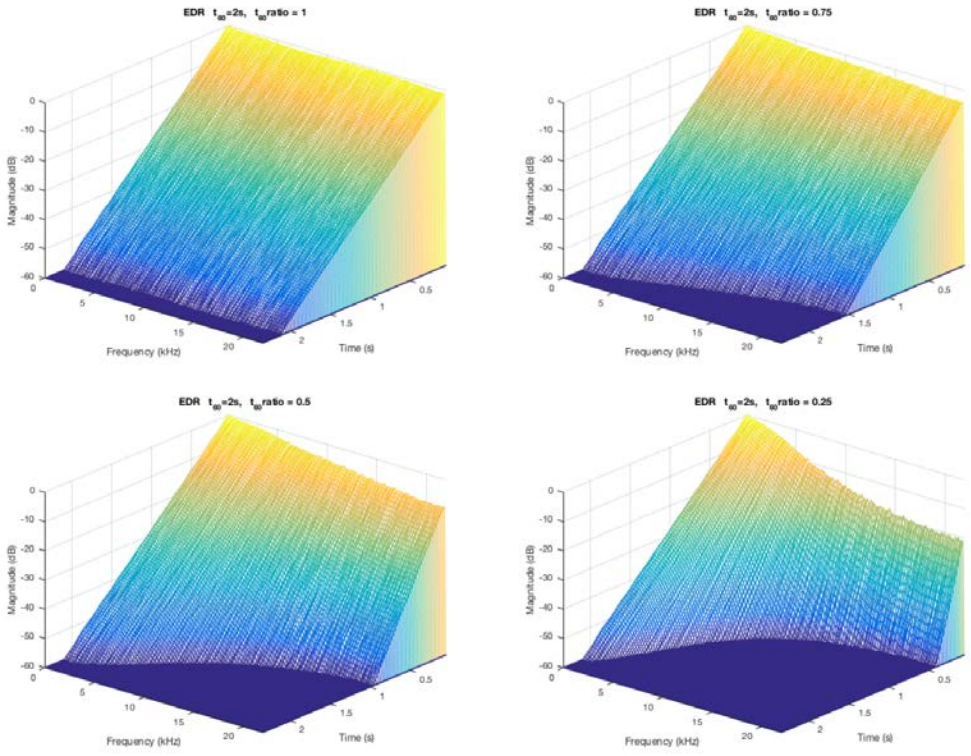

Abbildung 2.11: IR bei $\mathtt{T}_{60}(0) = 2s$ und unterschiedlichen $\mathtt{T}_{60,ratio}$;
$\mathtt{Roomsize} = 2000$ ($\mathtt{m}_i \in [2000, 4000]$; $\mathtt{Q} = 400$)

Der Parameter Roomsize hat keinen Einfluss auf das EDR. Dessen Wirkung ist im Zeitbereich deutlicher zu sehen(Abb.2.12). Mit `Roomsize` verändert sich die Reflexionsdichte, aber das Amplitudenverhältnis zwischen Erstreflexionen und Nachhall bleibt bei verändertem `Roomsize` nicht gleich. Das liegt daran, dass sich auch die Filterkoeffizienten der FDN-Tiefpässe verändern und damit eine andere Amplitudendämpfung einhergeht.

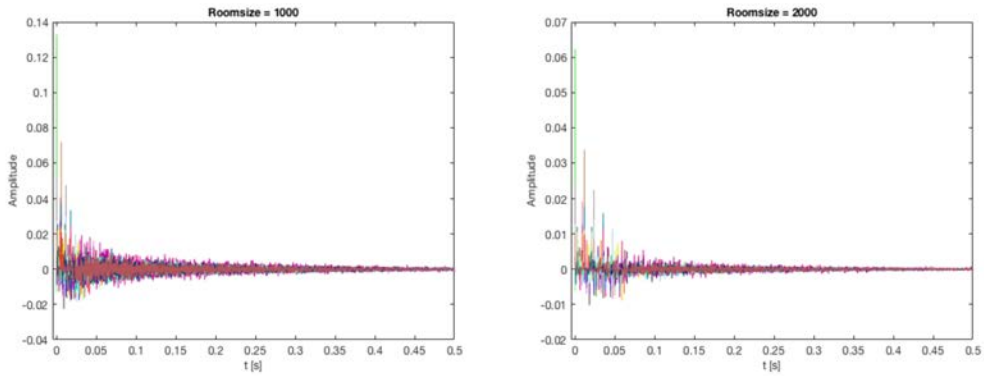

Abbildung 2.12: IR bei $\mathtt{T}_{60}(0) = 2s$, $\mathtt{T}_{60,ratio} = 0.15$ und verschiedene `Roomsize`; $\mathtt{Q} = 400$

Die EDR der Erstreflexionen und des Nachhalls werden getrennt in Abb.2.13 dargestellt. Dort ist erkennbar, dass der Nachhall zum Zeitpunkt 0s bis zu den hohen Frequenzen um mehr als 20dB abfällt. Dieser starke Abfall kommt dadurch Zustande, dass der Nachhall um die Einschwingphase des FDN bereinigt wurde und die übrigen Signale mindestens einmal durch den Tiefpass der Feedbackschleife gelaufen sind. Damit die beiden Signale besser zusammenpassen, werden die Erstreflexionen mit dem selben Tiefpass gefiltert, der in dem FDN[7] enthalten ist. Das reicht zwar nicht aus, damit sie in allen Frequenzen fließend ineinander übergehen, ein stärkeres Filter wird jedoch vermieden.
In Abb.2.15 ist das Gesamtsignal für verschiedene Nachhallpegel zusammengetragen. Dort wird deutlich wie die beiden Signalteile zusammenfließen.
In Abb.2.14 ist ein Vergleich mit der IR eines Seminarraums der TU-Berlin dargestellt. Dafür wurden die Nachhallzeiten für tiefe und hohe Frequenzen aus der EDR abgelesen und in der Software eingestellt. Das Ergebnis weist

[7]Es werden die Koeffizienten des stärksten Tiefpass (mit der größten Delayzeit m_N) gewählt.

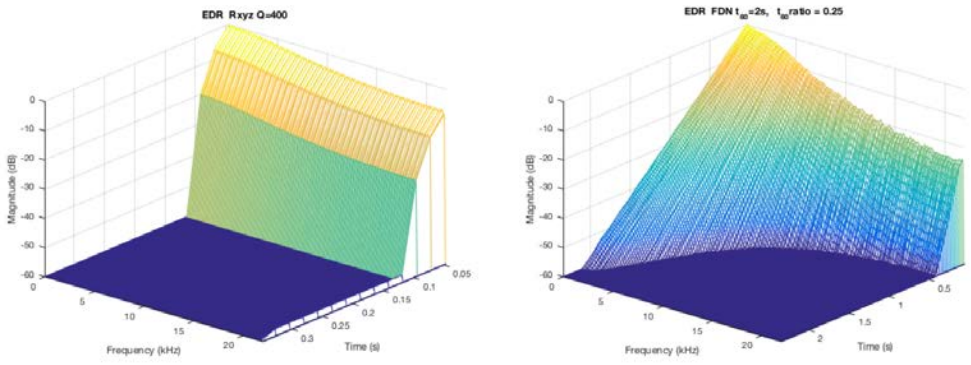

Abbildung 2.13: links: EDR der Erstreflexionen bei $\mathtt{Q} = 400$
rechts: EDR des Nachhalls bei $\mathtt{T}_{60}(0) = 2s$, $\mathtt{T}_{60,ratio} = 0.25$ und $\mathtt{m}_i \in [2000, 4000]$

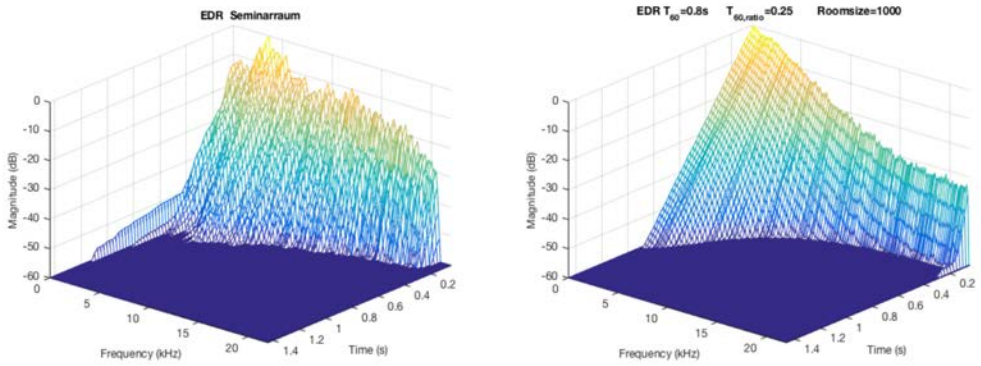

Abbildung 2.14: EDR eines Seminarraums und der Software bei gleicher Nachhallzeit

große Unterschiede auf. Die IR des Seminarraums ist komplexer strukturiert und die Nachhallzeit scheint mit steigender Frequenz linear abzunehmen. Das sind Absorbtionseigenschaften, die nicht mithilfe eines einfachen Filter erster Ordnung rekonstruierbar sind.

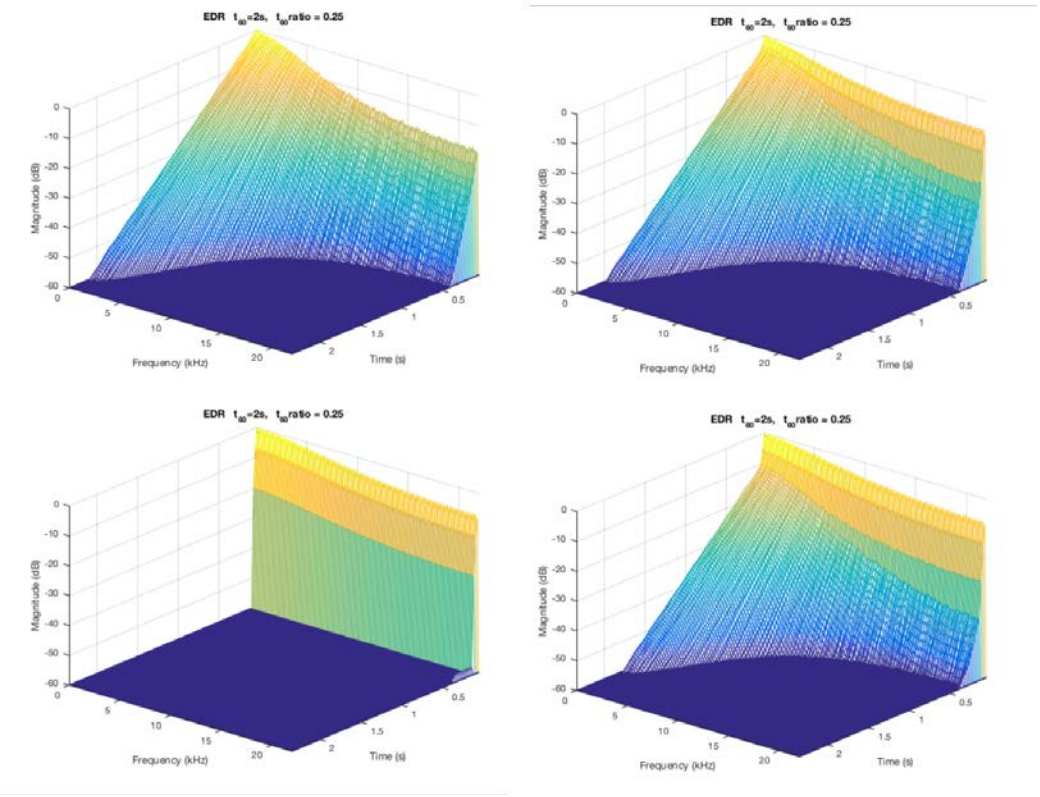

Abbildung 2.15: EDR bei $\mathtt{T}_{60}(0) = 2s$ und $\mathtt{T}_{60,ratio} = 0.25$ für unterschiedliche Nachhallpegel; `Roomsize`=2000 ($\mathtt{m}_i \in [2000, 4000]$ $\mathtt{Q} = 400$)

2.4.1 Höreindruck

Im Rahmen dieser Arbeit wurde kein repräsentativer Hörversuch durchgeführt um das Plugin zu beurteilen. Allerdings wurde die Software eingehend auf dem 21.2 System der TU-Berlin getestet und subjektiv beurteilt, davon wird im Folgenden unter Vorbehalt berichtet.
Aufgrund des hohen Ressourcenverbrauchs bei der Synthese der Erstreflexionen stellt sich die Frage, welchen Einfluss diese auf den Höreindruck haben und ob sich der Aufwand lohnt. Um die Wirkung der Erstreflexionen auf die Abstandswahrnehmung zu prüfen, wurde ein qualitativer Vergleich zwischen zwei Konfigurationen durchgeführt. In der ersten Konfiguration wurde das Plugin mit Erstreflexionen und Nachhall verwendet. In der anderen wurde auf Erstreflexionen verzichtet und nur der Nachhall des FDN inklusive Einschwingphase verwendet. Die Pegel der Erstreflexionen und des Nachhalls

wurden so eingestellt, dass sie dem Gehör nach gleichmäßig ineinander übergehen und neben einem eingehenden Sprachsignal bei `Distance` $= 0$, $\mathtt{T}_{60} = 2s$ und $\mathtt{T}_{60,ratio} = 0.25$ noch hörbar sind. (**D/R** ca. 60dB)
Dabei hat sich gezeigt, dass der Unterschied für `Distance`=0 nicht eindeutig erkennbar ist. Die Abstandswahrnehmung ist bei diesem Wert für beide Konfigurationen gleich und markiert ungefähr den kleinsten auditiven Abstand, der auf Grundlage des **D/R** in diesem System möglich ist. (siehe Abb.2.16) Bei größer werdendem `Distance` wird der Einfluss der Erstreflexionen aber deutlich. In beiden Konfigurationen steigt der empfundene Abstand zwar an, jedoch fällt es mit ER einfacher die virtuelle Schallquelle räumlich einzuordnen. Die Empfindung wirkt authentischer und die wahrgenommene Distanz ist größer. Sobald der Direktschall den gleichen Pegel erreicht wie

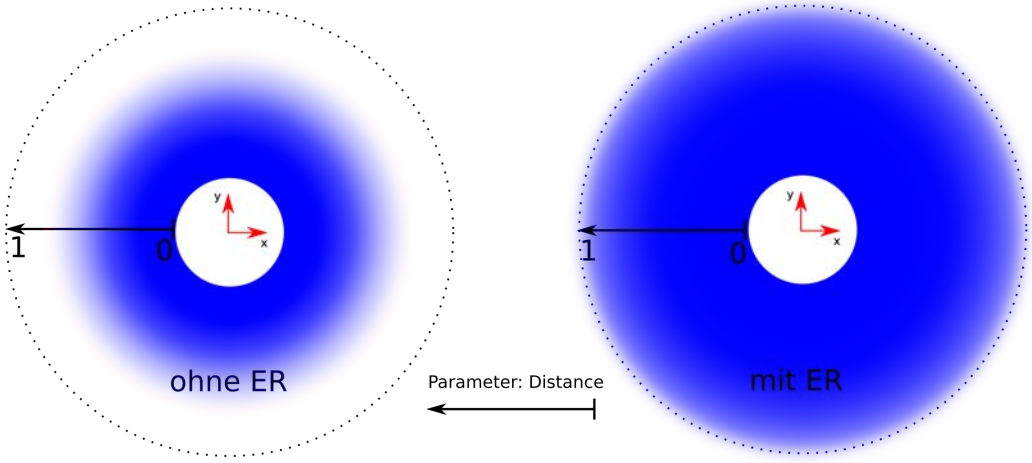

Abbildung 2.16: Bildlicher Vergleich der möglichen Abstandsempfindungen mit und ohne ER (In der xy-Ebene, Hörer im Ursprung); **Blau**: Eindeutige Empfindung der Position, **weiss**: Keine Empfindung der Position

der Hall, beginnt die Wahrnehmung der virtuellen Schallquelle ohne ER zusammenzubrechen. Mit ER bleibt sie stabil mit eindeutiger Einfallsrichtung und wachsendem Abstand. Steigt der Parameter `Distance` noch weiter, dann geht ohne ER die Empfindung der Schallquelle ganz verloren. Der Hörer ist umgeben von einem gleichmäßigem Hall, aus dem keine Informationen über den Ort der Quelle hervorgeht. Mit ER wird der räumliche Eindruck von der Einfallsrichtung und dem Abstand bis `Distance`=1 zwar ungenau, bleibt aber erhalten. Übrig bleibt die Empfindung der Raumgeometrie und der ungefähren Richtung einer weit entfernten Schallquelle.

Zusammengefasst verhelfen die Erstreflexionen zu einer stabilen, mit Veränderungen von `Distance` konsistenten, räumlichen Einordnung in einem virtuellen Raum, der sich deutlich größer anfühlt. Deshalb kann gesagt werden, dass die Erstreflexionen bei geeigneter Einstellung der Parameter einen eindeutigen Beitrag für die Manipulation der Abstandswahrnehmung virtueller Schallquellen in Ambisonics-Systemen leisten.

3 Implementierung

Die Implementierung von **AmbVerb** unterteilt sich zeitlich in zwei Abschnitte. Zuerst wurde die Software als „Pure Data External“ implementiert. Dabei wurden Grundkenntnisse der DSP Programmierung gesammelt und das Softwaredesign erarbeitet. Auf dieser Grundlage wurde im Anschluss ein Audioplugin erstellt. Das Softwaredesign und die Mehrheit der implementierten Funktionen blieben zwar unverändert, jedoch wurde das Plugin weiter ausgearbeitet und ist konzeptionell besser durchdacht. Deshalb wird im folgenden auf die Implementierung in Pure Data nur kurz eingegangen und im Anschluss, ein genauerer und representativer Blick auf die Implementierung des Audioplugins geworfen.

3.1 Implementierung in Pure Data

Pure Data ist eine visuelle, datenstromorientierte Programmiersprache mit modularen Aufbau [12][13]. Dem Prinzip nach ähnelt ein Pure Data Programm dem Aufbau analoger Studios bei dem der Signalfluss durch Verkabelung verschiedener Geräte hergestellt wird. Dem Anwender stehen diverse Objekte zur Verfügung, welche mithilfe von Linien verbunden werden, die den Datenfluss symbolisieren. Außer Audiosignalen können auch Nachrich-

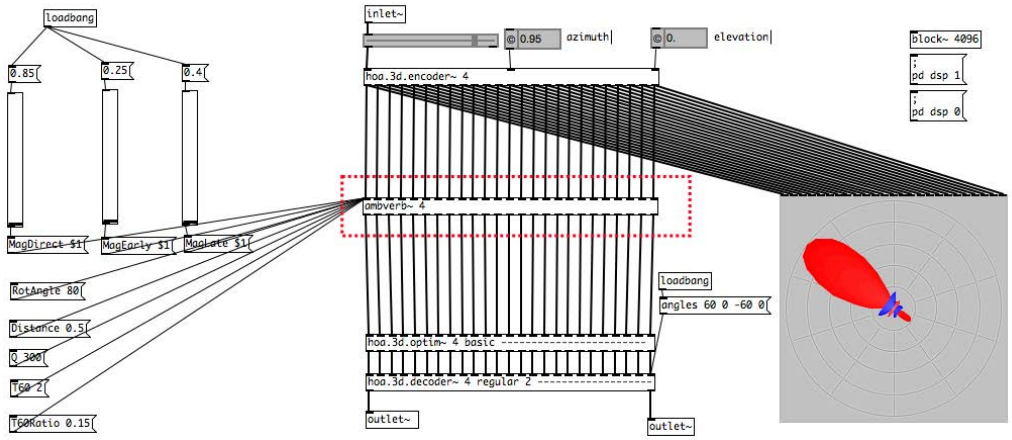

Abbildung 3.1: AmbVerb-External(rot markiert) im Einsatz, die Objekte der Encoder/Decoder und Visualisierung stammen aus der „HOA-Library“ (https://github.com/CICM/HoaLibrary)

ten zwischen PD-Objekten übertragen werden. Ein Objekt, dass von Dritten programmiert wurde, nennt man PD-External. PD-Externals werden in der Programmiersprache C geschrieben und basieren auf gleichen Prinzipien wie Audioplugins. Auch hier gibt es eine Callback-Funktion und eine Schnittstelle für Kontrolldaten. Aber wegen der einfacheren Programmiersprache C im Vergleich zu C++, fehlender GUI und keinem Multithreading, bietet es einen einfacheren Einstieg in DSP-Programmierung. Aus diesem Grund wurde die Software zuerst in PD implementiert und im Nachhinein auf ein Audioplugin übersetzt. Anstelle der Klassen `FDN` und `Earlyref` wurden die Strukturen `FDN` und `Earlyref` implementiert. Ein wesentlicher Nachteil im Vergleich zum Audioplugin ist, dass die Berechnungen nicht auf Threads verschiedener Priorität verteilt werden. Aus diesem Grund führen Änderungen von Kontrollparametern, bei denen rechenaufwendige Funktionen aufgerufen werden, zum Stillstand(lock) des DSP.

3.2 Audio Plugin

Audio Plugins sind ein fester Bestandteil digitaler Audioproduktion. Als Synthesizer, Audio- oder MIDI-Effekte dienen sie dazu, die Funktionalität einer DAW(Digital Audio Workstation) zu erweitern. Dabei fungiert ein Audio Plugin als Blackbox, die von der DAW bereitgestellte Signale verarbeitet und das Ergebnis an die DAW zurück gibt. Außer Audiosignalen können auch MIDI Befehle ausgetauscht werden. Die Anzahl von Ein- und Ausgängen ist beliebig und hängt von der spezifischen Funktionalität des Plugins ab. Die Steuerung von Plugins läuft über Kontrollparameter, die von der DAW organisiert werden und dem Anwender über eine Graphische Benutzeroberfläche(GUI) zugriff gewähren. Wenn das Plugin keine eigene GUI mitliefert, stellt die DAW eine GUI mit ihrem eigenen Design bereit. Heute existiert eine Reihe von Standard Protokollen, die den Austausch von Audio- MIDI- und Kontrolldaten zwischen DAW und Plugin regeln. In Tabelle 2 sind die gängigsten Plugin-Formate zusammengestellt.
Um sich bei der Programmierung eines Plugins nicht auf ein bestimmtes Format festlegen zu müssen, existieren sogenannte Wrapper, die generalisierte Funktionen bereitstellen und es ermöglichen, das Plugin in verschiedenen Formaten zu kompilieren. Im Rahmen dieser Arbeit wurde dafür das JUCE Framework verwendet.

Protokoll	Entwickler	Kompatibilität
Virtual Studio Technology (VST)	Steinberg	Mac OS X, Windows
Audio Units (AU)	Apple	OS X
Real Time AudioSuite (RTAS)	Avid	OS X, Windows
LV2	lv2plug.in	Linux, OS X, Windows

Tabelle 2: Die gängigsten Plugin-Formate

3.3 Implementierung in JUCE

Das JUCE[8] Framework ist eine Sammlung von C++ Klassen zur plattformübergreifenden Programmierung und beinhaltet Funktionalitäten für Audioverarbeitung, GUI Entwicklung, Multithreading und vieles mehr. Da in dieser Arbeit ein Audioplugin entsteht, wird im folgenden nur über diesen Teil des Frameworks berichtet.
In einem JUCE Project lassen sich Grundeinstellungen wie die Art des Plugins, die Kanalanzahl und benötigte Funktionalitäten vornehmen. Anschließend kann das Projekt für verschiedene Entwicklungsumgebungen wie z.B. Visual Studio für Windows und Xcode für Mac OS X eingerichtet werden. Ein Audioplugin wird dabei in zwei Klassen organisiert, die `PluginEditor` Klasse ist für die GUI und die Verarbeitung von Kontrollparametern verantwortlich. In der `PluginProcessor` Klasse liegt der DSP-Teil des Plugins, die wichtigste Funktion ist dabei die Callback-Funktion „`processBlock()`“. Diese Funktion wird von der DAW aufgerufen, sobald ein neuer Block gepufferter Eingangssignale bereit steht. Von dort aus werden die Audio- und MIDI-Signale verarbeitet und Ausgangssignale, zur Rückgabe an die DAW, in den Puffer geschrieben. Beide Klassen arbeiten auf verschiedenen Threads, wobei die `PluginProcessor` Klasse auf dem Audio-Thread mit sehr hoher Priorität arbeitet. Deshalb ist es besonders wichtig, Berechnungen die nicht zum DSP gehören, zum Beispiel GUI Funktionalität und Neuberechnungen von Kontrollparametern, von der `PluginEditor` Klasse aus durchzuführen. Andernfalls können Glitches und andere Störungen nicht ausgeschlossen werden. Für die Implementierung des AmbVerb-Plugins wurden zusätzlich die zwei Klassen `EarlyRef` und `FDN` erstellt, die in `PluginProcessor` initialisiert werden und in denen die Funktionen und Variablen zur Erzeugung der Erstreflexionen und des Nachhalls organisiert sind. Auch diese Klassen haben

[8] www.JUCE.com

jeweils eine Funktion namens „`processBlock()`“, die von `PluginProcessor::processBlock()` in den Callback eingebunden wird.
In Abb.3.3 ist die Arbeitsweise der vier Klassen vereinfacht dargestellt. Die DAW ruft `PluginProcessor::processBlock()` auf und übergibt einen Block gepufferter Eingangssignale. Der Eingangspuffer wird für die Synthese von Erstreflexionen und Nachhall an `EarlyRef::processBlock()` und `FDN::processBlock()` weitergegeben. Sobald die Synthese abgeschlossen ist, werden die erzeugten Signale mit dem Eingangssignal zusammengesetzt und in den Ausgangspuffer geschrieben.
Währenddessen wartet `PluginEditor` auf Änderungen der Kontrollparameter. Sobald in der GUI ein Regler bewegt wird, kommt es zum Aufruf von `PluginEditor::SliderValueChanged()`. In dieser Funktion werden die Kontrollparameter in `PluginProcessor` neu gesetzt und Methoden von `FDN` und `Eearlyref` aufgerufen, um die Werte der betroffenen Variablen (Filterkoeffizienten $\mathtt{g}_i$ und $\mathtt{p}_i$, Delayzeiten $\mathtt{m}_i$, IR und $\mathbf{R}^{(xyz)}_{\hat{\phi},Q}[q]$) neu zu berechnen. Dabei ist zu beachten, dass solche Methoden nicht auf Speicherplatz zugreifen sollten, auf den vom Audio-Thread aus zugegriffen wird. Sonst ist es wahrscheinlich, dass unvollständig errechnete Werte in den Signalfluss eingebunden werden. Das führt zu Artefakten oder sogar zum Absturz des DSP. Deswegen werden die neuen Werte auf temporären Variablen berechnet und erst im Anschluss für den Signalfluss freigegeben.
Im Folgenden werden einige wichtige Funktionen kurz erläutert. Einen ge-

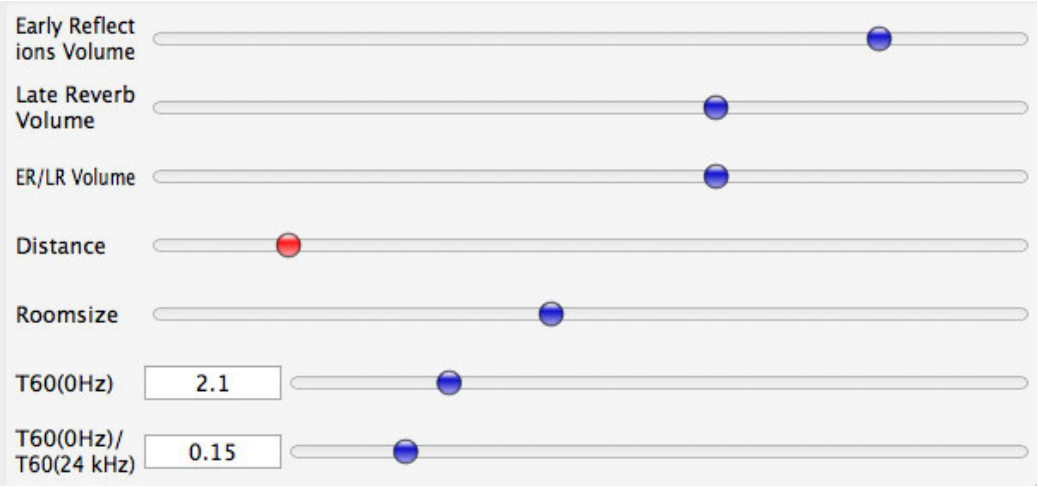

Abbildung 3.2: GUI von AmbVerb in Reaper

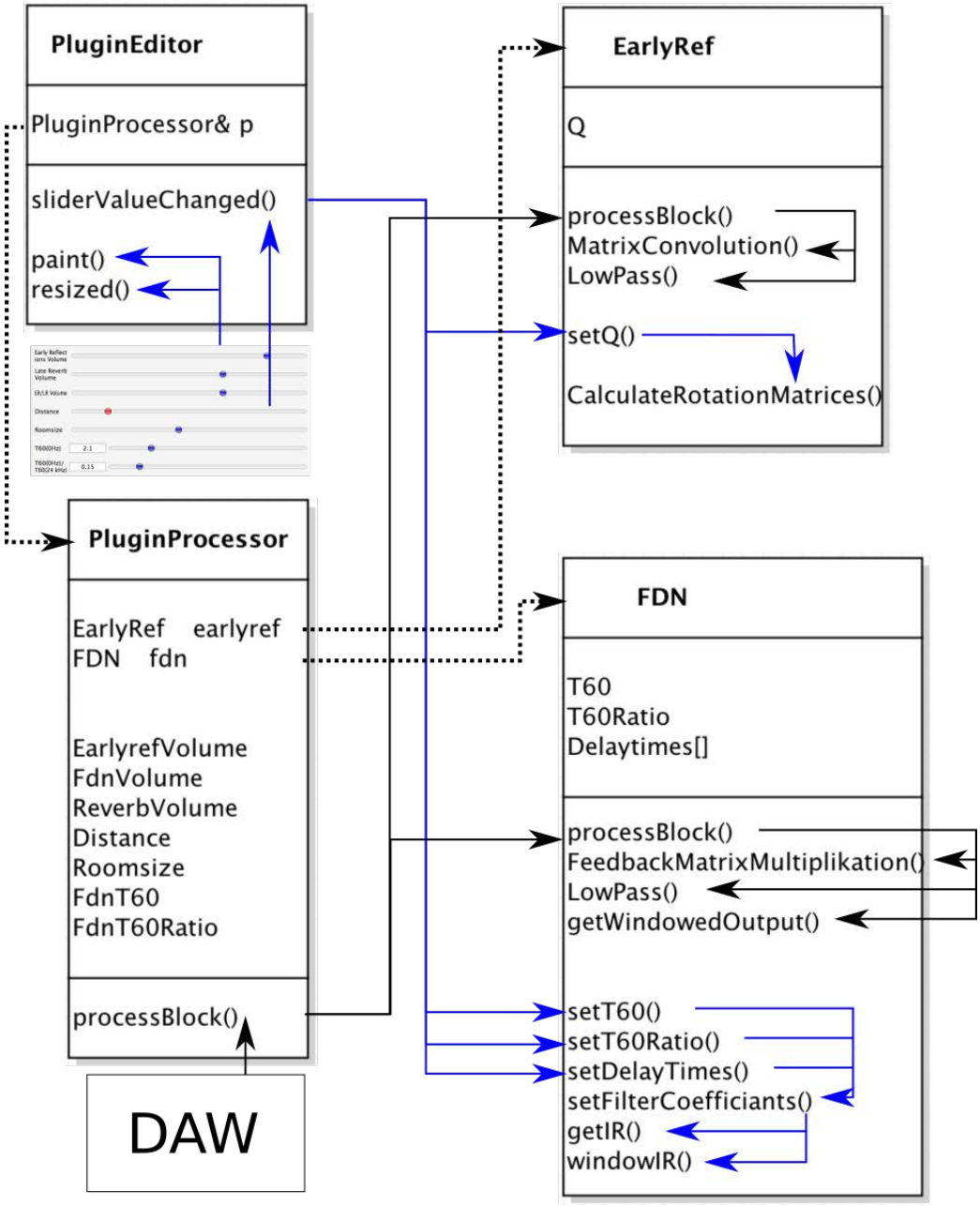

Abbildung 3.3: Vereinfachte Arbeitsweise von **AmbVerb**. Es sind nur Variablen und Methoden angegeben die zum grundlegenden Verständnis beitragen. schwarz: Funktionsaufruf auf Audio-Thread, blau: Funktionsaufruf auf PluginEditor-Thread, gestrichelt: Referenz

naueren Einblick kann man sich in den Quelltexten der Software verschaffen.

`FDN::`

`setDelayTimes()`
Auswahl der Delayzeiten $\mathtt{m}_i$ aus den Primzahlen im Bereich $[\mathtt{Roomsize}, 2.5\mathtt{Roomsize}]$. Wird aufgerufen, sobald sich `Roomsize` ändert.

`CalculateFilterCoefficiants()`
Berechnung der Filterkoeffizienten der Tiefpässe. Wird aufgerufen sobald sich `Roomsize`, $\mathtt{T}_{60}$ oder $\mathtt{T}_{60,Ratio}$ ändern.

`getIR()`
Messung der gegenwärtigen Impulsantwort des FDN.

`WindowIR()`
Fensterung der Einschwingphase der gemessenen IR.

`processBlock()`
Grundstruktur des FDN. Das Eingangssignal wird in die Delayspuren der Feedbackschleife geschrieben, verschiedene Funktionen zur Umsetzung des Signalflusses aufgerufen und die Beschneidung von der Einschwingphase organisiert.

`FeedBackMatrixMultiplikation()`
In dieser Funktion werden die aktuellen Blöcke der Delayspuren mit der Feedbackmatrix multipliziert und das Ergebnis zu dem Anfang der Feedbackschleife addiert.

`getWindowedOutput()`
Die gefensterte IR wird mit dem Ausgangssignal gefaltet.

`EarlyRef::`

`processBlock()`
Grundstruktur des Widening-Algorithmus. Von hier aus wird der Eingangssignal-Vektor mit $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ gefaltet und durch einen Tiefpass geleitet.

`CalculateRotationMatrices()`
$\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ wird gemäß Gl.(21) berechnet und von der Einschwingphase befreit. Wird aufgerufen sobald sich `Roomsize` ändert.

`MatrixConvolution()`
Faltung eines DSP Blocks mit $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$. (siehe Gl.(22))

Das Audioplugin kann bis zu Ambisonics $\mathcal{O}(10)$ und FDN $\mathcal{O}(32)$ kompiliert werden. Die Rotationsmatrizen $\mathbf{R}^{(z\cdot)}$ und $\mathbf{R}^{(\cdot z)}$ wurden in Matlab mit Hilfe der „Spherical-Harmonic-Transform“-Library[9] berechnet.

3.4 OSC Interface

Das Plugin wurde bisher in der DAW Reaper in Verbindung mit dem Encoder und Decoder aus der AmbiX-PluginSuite[10] von Matthias Kronlachner getestet. AmbiX ist eine Sammlung von Audioplugins zur Audioproduktion in Ambisonics. Die GUI des Encoders bietet die Möglichkeit, eine virtuelle Schallquelle auf einer gedachten Kugeloberfläche zu positionieren. Dabei werden die Parameter `Azimuth` und `Elevation` festgelegt. Dies kann zum Beispiel mit einer Bewegung der Maus geschehen (siehe Abb.3.4). Durch

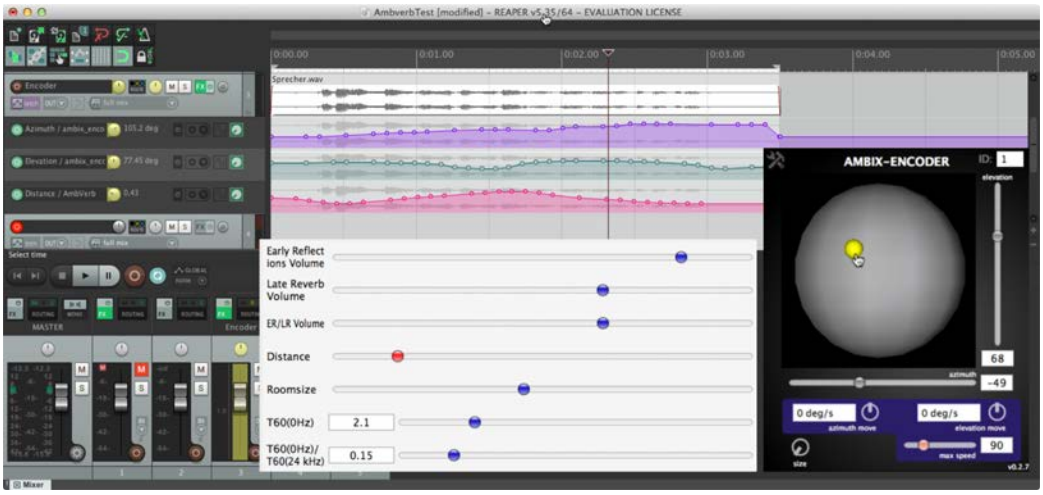

Abbildung 3.4: Ambisonics Produktion in Reaper

[9] https://github.com/polarch/Spherical-Harmonic-Transform
[10] http://www.matthiaskronlachner.com

Einbindung des AmbVerb-Plugins wird die mögliche Position der Schallquelle, mit dem Parameter `Distance` um eine weitere Dimension erweitert. Für die Produktion einer Klangszene, bei der eine Bewegung in drei Dimensionen erreicht werden soll, ist es sehr umständlich `Distance` unabhängig von dem Raumwinkel nach seinen Wünschen einzustellen. Aus diesem Grund entsteht der Bedarf nach einem Interface, mit dem sich alle drei Dimensionen gemeinsam kontrollieren lassen. Die Wahl fiel dabei auf den Virtual-Reality Controller **Leap Motion**. Der Controller basiert auf Infrarot Sensoren und ermöglicht die Erkennung von absoluten Positionen und Gesten der Hand. Für die Kommunikation zwischen Leap Motion und Reaper wurde das Pd-

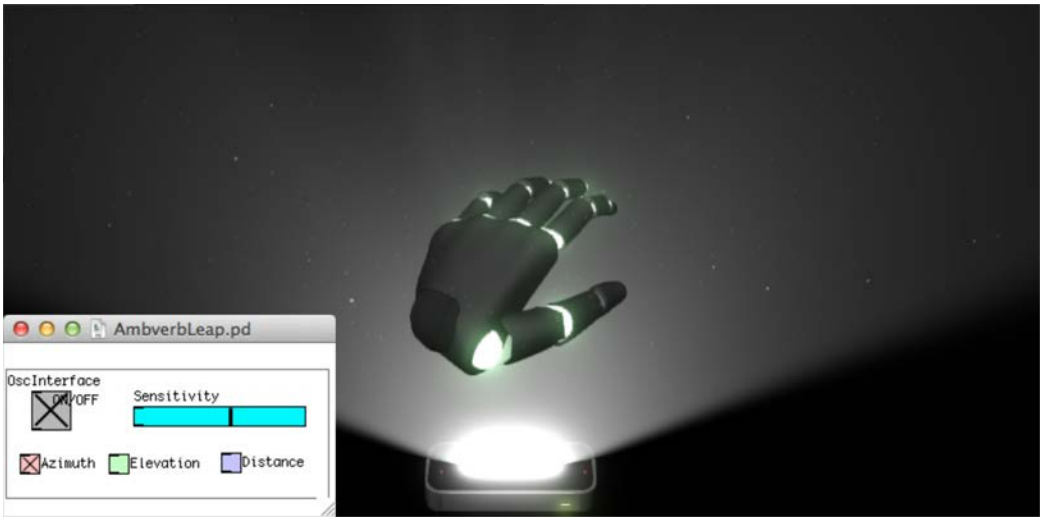

Abbildung 3.5: LeapMotion Handerkennung

Patch „AmbverbLeap“ erstellt. Dort werden die Daten aus der Leap Motion-Schnittstelle mithilfe des Pd-Externals „Leapmotion“ von Miyama Chikashi ausgelesen. Im Anschluss werden die kartesischen Koordinaten in Kugelkoordinaten umgerechnet, auf den Wertebereich der Audioparameter abgebildet und per OSC an Reaper gesendet. In Reaper müssen die OSC Nachrichten den richtigen Parametern zugewiesen werden. Dafür können die Positionsdaten für `Azimuth`, `Elevation` und `Distance` einzeln aktiviert werden um sie mithilfe von „OSC-learn“ an den richtigen Audioparameter zu koppeln.

Der Erkennungsbereich der Handposition liegt innerhalb eines Abstands von ca. 60cm zum Controller. Deshalb muss kurz geübt werden um ein Gefühl

dafür zu entwickeln. Danach lassen sich alle drei Audioparameter bequem und flüssig steuern. Die Empfindlichkeit(Sensitivity) von `Distance` kann in dem Patch zusätzlich eingestellt werden.

4 Ausblick und offene Fragen

Während der Entwicklung und Implementierung von **AmbVerb** wurde deutlich, dass die Verwendung des Widening-Algorithmus innerhalb eines Hybrid Reverb viele Freiheitsgrade mit sich bringt. Im Rahmen der vorliegenden Arbeit konnte die Anzahl an möglichen Designentscheidungen und der Arbeitsaufwand, der mit der Implementierung verbunden ist, nur mit Hilfe pragmatischer Zielgerichtetheit und heuristischer Methoden bewältigt werden. Im Folgenden werden einige Verbesserungsmöglichkeiten und Fragen erläutert, die dabei nicht behandelt werden konnten.

4.1 Software Design

Ein wichtiger Punkt betrifft den Ressourcenverbrauch der Erstreflexionen. Im gegenwärtigen Design geht die Impulsantwort von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ ganz in das Hybrid Reverb ein, obwohl die Überblendung schon nach einem Drittel der Dauer vollständig ausgeführt ist. Es ist zu bezweifeln, dass das Ausschwingen von $\mathbf{R}^{(x,y,z)}_{\hat{\phi},Q}[q]$ unter dem eingeschwungenen Nachhall noch wahrnehmbar ist. Durch weglassen dieses Teils kann die benötigte Puffergröße und Rechenkapazität halbiert werden. Bisher wurde davon abgesehen, weil die Auswirkung auf die Frequenzzuordnung der Rotationen noch nicht untersucht wurde. Eine weitere Idee, der man nachgehen kann, ist die Erstreflexionen bei geringerer Abtastrate durchzuführen. Auch in diesem Fall müssen noch die Auswirkungen auf den Höreindruck geprüft werden.

Bisher wurde noch kein Versuch unternommen, die Form und Reflexionsdichte der Impulsantwort zu kontrollieren. Zotter und Kronlachner haben bemerkt, das neue Formen entstehen, wenn man $h(m\hat{\phi}, m\varphi_0, (\lambda - \Lambda)Q]$ bei verschiedenen Q mehrmals mit sich selbst faltet. Andere Formen ließen sich auch erzeugen indem man den Rotationswinkel nicht mit $\zeta = \hat{\phi}\cos(\Omega)$, sondern mit anderen Funktionen moduliert. Dafür wären mathematische Untersuchungen notwendig. Darüber hinaus ist unklar, welche geometrischen Eindrücke durch die Form der Widening-Erstreflexionen erzeugt werden können. Aber auch ohne die Form der IR kontrollieren zu können, kann die Qualität des Hybrid Reverb verbessert werden. Die Parametrisierung der Überblendung wurde heuristisch durchgeführt. Durch eindringlichere Analyse kann der Übergang zwischen Erstreflexionen und Nachhall bezüglich Frequenzgang so-

wie dem zeitlichen Amplitudenverlauf sehr wahrscheinlich verbessert werden. Ferner kann man die Grenzen der Fensterung an analytische Schwellenwerte koppeln und die Zuordnung von `Q` und $\mathtt{m}_i$ an den Parameter `Roomsize` verbessern.

Eine weitere Designvariante ist, das FDN mit den Signalen des Widening-Algorithmus zu speisen. Diese Herangehensweise wurde im Rahmen dieser Arbeit zwar ausprobiert, aber nicht weiter verfolgt.

Des Weiteren wurde über den Parameter `Distance` bisher nur das Direktschall-Hall-Verhältnis mit der Integrationszeit $T = (2-3)ms$ beeinflusst. Die Auswirkung anderer Integrationszeiten wurde nicht untersucht.

$$D/R = \frac{\int_0^T h^2(t)dt}{\int_T^\infty h^2(t)dt}$$

Zusammenfassend lässt sich sagen, dass die Software noch viel Potential zur Forschung und Verbesserung bietet. Der aktuelle Zustand kann als guter Ausgangspunkt gesehen werden, von dem aus die Eignung des Widening-Algorithmus zur Erzeugung realitätstreuer Erstreflexionen untersucht werden kann.

4.2 Implementierung

Obwohl alle Vorsichtsmaßnahmen eingehalten wurden um Störgeräusche bei der Bedienung der GUI zu vermeiden, kommt es bei Änderungen der Parameter `Roomsize`, $\mathtt{T}_{60}$ und $\mathtt{T}_{60,Ratio}$ zu störenden Artefakten. Die Ursache liegt in der Fensterung der FDN-Einschwingphase und wurde noch nicht behoben. Wie in Kapitel2 beschrieben ist, wird die Einschwingphase des FDN entfernt, indem die gefensterte Impulsantwort mit dem Ausgangssignal des FDN gefaltet und davon subtrahiert wird. Wenn im Verlauf eines `processBlock()` die IR geändert wird, dann setzt sich der Ausgangspuffer mit hoher Wahrscheinlichkeit aus zwei Teilen zusammen. Der erste Teil enthält noch Werte die im vorhergehenden Systemzustand berechnet wurden, der andere Teil besteht aus Werten die dem aktuellen Zustand entsprechen. Wenn mit diesem Puffer die Routine zur Entfernung der Einschwingphase durchgeführt wird,

dann wird die Einschwingphase nur in dem Teil des Puffers entfernt, der aus dem neuen Systemzustand heraus berechnet wurde. Die Artefakte sind umso stärker, je größer die Änderung der IR ist. Folglich ist `Roomsize` am stärksten betroffen, denn mit ihm ändern sich die Delayzeiten, die Filterkoeffizienten und die Grenzen der Fensterung. Dieses Problem kann gelöst werden, indem der Wechsel in einen neuen Systemzustand nur am Ende oder Anfang des Callbacks passiert. Dafür kann man eine „Flag "-Variable einführen, die innerhalb von `FDN::processBlock()` zum Wechsel zwischen zwei Systemzuständen auffordert.

Der wichtigste Kritikpunkt bezieht sich auf die Kompatibilität mit anderen Betriebssystemen. Obwohl man das Plugin in den Formaten kompilieren kann, die von JUCE unterstützt werden, ist es im gegenwärtigen Zustand nur lauffähig auf **Mac OSX**. Dass liegt daran, dass die FFT-Funktionen, die für die Schnelle Faltung gebraucht werden, aus dem **Accelerate Framework**[11] von **Apple** stammen. Dieser Umstand lässt sich beheben, in dem die betroffenen Stellen durch Funktionen der plattformunabhängigen **FFTW**[12] Library ersetzt werden. Auch das PD-External ist von diesen Mängeln betroffen und muss überarbeitet werden, bevor es veröffentlicht wird. Diese Verbesserungen sind im Anschluss an diese Arbeit in Aussicht gestellt.

[11] https://developer.apple.com/documentation/accelerate
[12] www.fftw.org

Literatur

[1] Robert Albrecht and Tapio Lokki. Adjusting the perceived distance of virtual speech sources by modifying binaural room impulse responses. 2013. (document), 1.3

[2] Thibaut Carpentier, Markus Noisternig, and Olivier Warusfel. Hybrid reverberation processor with perceptual control. In *17th Int. Conference on Digital Audio Effects (DAFx-14), Erlangen*, 2014. (document), **??**

[3] Filippo M. Fazi and Philip A. Nelson. The ill-conditioning problem in sound field reconstruction. In *Audio Engineering Society Convention 123*. Audio Engineering Society, 2007. 1.1

[4] Michael Gerzon. Synthetic stereo reverberation, parts i and ii. *Part*, 1:632–635, 1971. 1.5

[5] Jean-Marc Jot and Antoine Chaigne. Digital delay networks for designing artificial reverberators. In *Audio Engineering Society Convention 90*. Audio Engineering Society, 1991. 1.5

[6] Matthias Kronlachner. Ambisonics plug-in suite for production and performance usage. In *Linux Audio Conference*, pages 49–54. Citeseer, 2013. 4

[7] Alexander Lindau, Linda Kosanke, and Stefan Weinzierl. Perceptual evaluation of model-and signal-based predictors of the mixing time in binaural room impulse responses. *Journal of the Audio Engineering Society*, 60(11):887–898, 2012. 1.4

[8] Dave Malham. Higher order ambisonic systems. *abstracted from'Space in Music-Music in Space', an Mphil thesis by Dave Malham, submitted to the University of York in April*, 2003. 1.1.1

[9] Damian Murphy, Mark Beeson, Simon Shelley, Alastair Moore, and Alex Southern. Hybrid room impulse response synthesis in digital waveguide mesh based room acoustics simulation. In *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx-08)*, pages 129–136, 2008. (document)

[10] Christian Nachbar, Franz Zotter, Etienne Deleflie, and Alois Sontacchi. Ambix-a suggested ambisonics format. In *Ambisonics Symposium, Lexington*, 2011. 1.1.1

[11] Andrea Primavera, Stefania Cecchi, Francesco Piazza, Junfeng Li, and Yonghong Yan. Hybrid Reverberator Using Multiple Impulse Responses for Audio Rendering Improvement. pages 314–317. IEEE, October 2013. (document), **??**

[12] Miller Puckette et al. Pure data: another integrated computer music environment. *Proceedings of the second intercollege computer music concerts*, pages 37–41, 1996. 3.1

[13] Miller S. Puckette and others. Pure Data. In *ICMC*, 1997. 3.1

[14] XIA Risheng, LI Junfeng, Andrea Primavera, Stefania Cecchi, Yôiti Suzuki, and YAN Yonghong. A hybrid approach for reverberation simulation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 98(10):2101–2108, 2015. **??**

[15] Lauri Savioja and U. Peter Svensson. Overview of geometrical room acoustic modeling techniques. *The Journal of the Acoustical Society of America*, 138(2):708–730, August 2015. 1.4

[16] Julius O Smith. Physical audio signal processing. *Linear Predictive*, 2010. 1.5

[17] Sascha Spors and Jens Ahrens. A comparison of wave field synthesis and higher-order ambisonics with respect to physical properties and spatial sampling. In *Audio Engineering Society Convention 125*. Audio Engineering Society, 2008. (document), 1.1, 1.1

[18] Sascha Spors, Hagen Wierstorf, Alexander Raake, Frank Melchior, Matthias Frank, and Franz Zotter. Spatial Sound With Loudspeakers and Its Perception: A Review of the Current State. *Proceedings of the IEEE*, 101(9):1920–1938, September 2013. (document), 1, 2

[19] Stefan Weinzierl, editor. *Handbuch der Audiotechnik*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. 1.3

[20] Torben Wendt, Steven van de Par, and Stephan D. Ewert. Perceptual and room acoustical evaluation of a computational efficient binaural room impulse response simulation method. 2014. **??**

[21] Pavel Zahorik. Auditory display of sound source distance. In *Proc. Int. Conf. on Auditory Display*, pages 326–332, 2002. (document), 1.3

[22] Frank Zotter, Matthias Frank, Matthias Kronlachner, and Jung-Woo Choi. Efficient phantom source widening and diffuseness in ambisonics. 2014. 1.4, 1.6

[23] Franz Zotter. *Analysis and synthesis of sound-radiation with spherical arrays*. Franz Zotter, 2009. 1.2

[24] Franz Zotter and Matthias Frank. All-Round Ambisonic Panning and Decoding. *J. Audio Eng. Soc*, 60(10):807–820, 2012. (document), 1.1

[25] Franz Zotter, Matthias Frank, Georgios Marentakis, and Alois Sontacchi. Phantom source widening with deterministic frequency dependent time delays. In *International conference on digital audio effects (DAFx). Paris, France*, 2011. (document)

[26] Franz Zotter, Hannes Pomberger, and Matthias Frank. An alternative ambisonics formulation: Modal source strength matching and the effect of spatial aliasing. In *Audio Engineering Society Convention 126*. Audio Engineering Society, 2009. 1.1

[27] Udo Zölzer and Xavier Amatriain, editors. *DAFX: digital audio effects*. Wiley, Chichester ; New York, N.Y, 2002. (document), 1.5