Bachelorarbeit

im Studiengang Audiovisuelle Medien

Prototypische Implementierung eines objekt-basierten Mastering-Kompressors für 3D-Musikproduktion

vorgelegt von

Benedikt Ernst

an der Hochschule der Medien Stuttgart

am 06. September 2022

zur Erlangung des akademischen Grades Bachelor of Engineering

Erstprüfer:	Prof. Dr. Frank Melchior
Zweitprüfer:	Prof. Oliver Curdt

Matrikelnummer: 37396

info@benedikt-ernst.com

Ehrenwörtliche Erklärung

Hiermit versichere ich, Benedikt Ernst, ehrenwörtlich, dass ich die vorliegende Bachelorarbeit mit dem Titel: „Prototypische Implementierung eines objekt-basierten Mastering-Kompressors für 3D-Musikproduktion“ selbstständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die mit dem Wortlaut oder dem Sinn nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quellen kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden.

Ich habe die Bedeutung der ehrenwörtlichen Versicherung und die prüfungsrechtlichen Folgen (§24 Abs. 2 Bachelor-SPO (7 Semester) der HdM) einer unrichtigen oder unvollständigen ehrenwörtlichen Versicherung zur Kenntnis genommen.“

Ort, Datum, Unterschrift

Kurzfassung

Durch die zunehmende Anzahl an objekt-basierten Audioformaten, wächst in der Musikbranche die Anzahl der dreidimensionalen Produktionen. Während immer mehr Tonstudios ihre Regieräume für immersive Audioformate aufrüsten, wird objekt-basierten Audioprozessoren bisher wenig Aufmerksamkeit gewidmet.

Die vorliegende Arbeit beschäftigt sich mit der Entwicklung eines objekt-basierten Audioprozessors, der eine einheitliche Bearbeitung mehrerer Audiosignale in Abhängigkeit ihrer Positionsdaten ermöglicht. Dabei ist das Ziel dieser Studie eine prototypische Implementierung eines objekt-basierten Mastering-Kompressors.

Um ein Konzept für objekt-basierte Audioprozessoren zu entwickeln, wurden die Funktionen von Kompressoren und objekt-basierten Audioformaten und deren Einsatz beim Mastering für Musik berücksichtigt. Das Konzept baut auf dem aktuellen Stand der Forschung auf.

Die prototypische Implementierung hat gezeigt, dass objekt-basierte Audioprozessoren für die einheitliche Bearbeitung technisch realisiert werden können und damit ein Mastering für objekt-basierte Audioformate möglich ist.

Abstract

With the increasing number of object-based audio formats, more three-dimensional music is produced. While music studios are upgrading their mixing rooms with immersive sound systems, there is a lack of development for object-based audio processors. The current study develops an object-based audio processor, which allows uniform processing of multiple audio signals depending on the position data contained in the metadata. Therefore, the purpose of this study is a prototypical implementation of an object-based compressor for mastering in music.

The functioning of audio compressors and object-based audio formats as well as their use in mastering music was considered when developing a concept for an object-based audio processor. This audio processor builds on the current state of research.

The prototypical implementation shows that object-based audio processors for uniform processing can be implemented. Thus, mastering object-based audio formats is possible.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

3D-Audio rückt seit einigen Jahren immer mehr in den Fokus der Tonschaffenden und seit Kurzem auch der Konsumenten/-innen. Insbesondere durch die Integration verschiedener 3D-Formate wie Sonys 360 Reality Audio und Dolby Atmos bei den Musik-Streaming-Diensten Tidal Hi-Fi, Amazon Prime Music HD und Deezer Hi-Fi (Bresler, 2021; Inc., 2020; Sony Electronics Inc., 2019, 2021) sowie der Release von Apples Spatial Audio (Apple Inc., 2021a) gibt der Entwicklung entsprechender Formate großen Aufwind. Zudem muss der Aufwand für die Konsumenten/-innen durch die flexiblen Wiedergabemöglichkeiten nicht größer sein als für Stereo-Musik, die über Kopfhörer konsumiert wird.

Obwohl 3D-Audio wie eine Erfindung des 21. Jahrhunderts vermarktet wird, gab es bereits 1940 den ersten Kinofilm mit 3D-Audio – Walt Disneys *Fantasia* (Boren, 2017). Und auch flexible Wiedergabemöglichkeiten einer 3D-Audio-Datei auf verschiedenen Endgeräten sind seit mehreren Jahrzehnten mit Ambisonics realisierbar. Warum bekommt 3D-Audio Musik dann erst jetzt die breite Aufmerksamkeit?

Wenngleich die Versuche, Musik für kanalbasierten Surround Sound zu etablieren, erfolglos waren, da die Hürden für die meisten Konsumenten zu großen waren (Braddock et al., 2021) und Upmixe für größere Systeme bzw. Downmixe für kleinere Systeme eher unzufriedenstellend waren (Buff, 2020), entwickelte sich vor allem im Bereich des Sende- und Kinotons das Hörerlebnis immer mehr in Richtung Immersion und Interaktion. Dafür wurde ein flexibleres als das bisherige kanal-basierte Format benötigt (EBU, 2014). Die objektbasierte Lösung für Fernsehen und Kino wurde mit etwas Verzögerung auch zur Lösung der Probleme, die für den ausbleibenden Erfolg von Musik für kanal-basierten Surround verantwortlich waren.

Bei objekt-basiertem Audio (z. B. Dolby Atmos) besteht, anders als bei kanal-basiertem Audio (z. B. Dolby Surround), nicht die Notwendigkeit beim Produzieren und Konsumieren über ein identisches Abhörsystem zu hören (Braddock et al., 2021). Unabhängig davon, ob die Produktion mit 16 Lautsprechern (z. B. 9.1.6), 14 Lautsprechern (z. B. 5.1.4) oder binaural (mit Kopfhörern) stattfand, kann der finale Master mit einem 5.1 Lautsprechersetup wiedergegeben werden, ohne dass es dabei zu Verlusten von Signalinformationen kommt. Das ist möglich, weil die Kanalzuordnung für die Lautsprecher nicht wie bei kanal-basiertem Audio bei der Produktion im Studio festgelegt wird, sondern erst bei den Konsumenten/-innen auf den entsprechenden Endgeräten.

Damit bietet objekt-basiertes Audio im immersiven Audiobereich eine Lösung für die Probleme von kanal-basiertem Surround Sound und ist skalierbar für neue Abspielsysteme, wie etwa Smart Speaker, Soundbars oder Hi-Fi-Anlagen.

Mit der wachsenden Aufmerksamkeit der letzten Jahre wurde auch die Entwicklung der Produktionsumgebungen beschleunigt, sodass neben Avid ProTools und Steinbergs Nuendo seit 2021 mit Logic Pro (Apple Inc., 2021b), Cubase (Steinberg Media Technologies GmbH, 2022) und Pyramix (Merging Technologies SA, 2022) die Anzahl digitaler Audio Workstations[1] (im Folgenden DAW), die ein objekt-basiertes Format nativ unterstützen, immer größer wird. Zeitgleich entwickeln auch Plug-in Hersteller vermehrt Mehrkanal-Produkte, die für die Produktion von immersivem Audio von Nutzen sind, wodurch einige der wichtigen Bearbeitungstools auch bei Mehrkanalaudio verwendet werden können. Dennoch gibt es bisher kaum Bearbeitungstools, die auf der Grundlage von objekt-basiertem Audio funktionieren.

Die vorliegende Arbeit entwickelt ein Konzept zur Umsetzung von akustischer Bearbeitung objekt-basierter Audiosignale mit besonderem Bezug auf die Arbeit beim Mastering und den Einsatz von Regelverstärkern. Daraus ergibt sich folgende Forschungsfrage:

Wie kann ein objekt-basierter Mastering-Kompressor realisiert werden?

Zur Beantwortung der Forschungsfrage wird das entwickelte Konzept in einer prototypischen Implementierung umgesetzt.

Dazu wird in Kapitel 2 das Mastering von Musik und die verwendeten Audioprozessoren zusammengefasst. In dem Zusammenhang wird die Funktionsweise eines Abwärts-Kompressors erklärt, die für die später folgende Konzeption die Grundlage bildet.

In Kapitel 3 wird 3D-Audio besprochen. Dabei wird auf verschiedene Formate eingegangen und deren Funktion erläutert. Ein besonderer Schwerpunkt wird auf die objekt-basierten Formate und ihre Infrastruktur gelegt. Die Erkenntnisse dieses Kapitels werden bei der Konzeption berücksichtigt.

Kapitel 4 fasst den aktuellen Forschungsstand zusammen und umreißt die Möglichkeiten von objekt-basiertem Mastering, bevor das darauffolgende Kapitel 5 die Frage zur Umsetzung eines objekt-basierten Mastering-Kompressor behandelt.

Zur Überprüfung der Lösungsansätze aus Kapitel 5 werden diese in Kapitel 6 innerhalb einer prototypischen Implementierung umgesetzt. Dazu werden der Aufbau des Prototyps mit der Metadatenverarbeitung und dem Signalfluss dargestellt.

Abschließend werden die Ergebnisse der Konzeptionierung und prototypischen Implementierung, in Kapitel 7 zusammengefasst, bewertet und eingeordnet.

[1] DAW: „Digital Audio Workstation“. Computerprogramm zur Bearbeitung von Audio

2 Mastering von Musik

Obwohl die Musikindustrie seit Beginn des 21. Jahrhunderts durch die wachsende Relevanz von Streaming-Diensten tiefgehende infrastrukturelle Veränderung durchlaufen hat, die den Prozess von aktuellen Produktionen und Veröffentlichungen maßgeblich bestimmen, ist das Mastering immer noch ein essenzieller Bestandteil in der Produktionskette (Cousins & Hepworth-Sawyer, 2013; Hiller & Walter, 2017).

Nüchtern betrachtet ist das Mastering ein Aufbereitungsschritt für die Distribution eines Werkes. Dieser Schritt ist nicht nur die letzte Möglichkeit Korrekturen vorzunehmen, sondern auch mit kreativen Eingriffen das finale Produkt noch einmal etwas aufzupolieren (Cousins & Hepworth-Sawyer, 2013; Owsinski, 2009). Deshalb ist das Mastering ebenso als eine Kunst zu verstehen wie das Produzieren oder Mischen, denn obwohl sich viele Mythen um diesen Produktionsschritt ranken, eine Wissenschaft ist dieser Schritt nicht und den *einen* richtigen Master gibt es auch nicht (Cousins & Hepworth-Sawyer, 2013).

2.1 Geschichte

Durch die Erfindung der ersten Bandmaschine (Ampex 200A) Mitte des 20. Jahrhunderts wurde das Mastering als letzter Schritt vor der Vervielfältigung ein notwendiger Bestandteil der Musikproduktion. Bis dahin wurden die wenigen Mikrofone, die zur Aufnahme verwendet wurden, direkt auf 10" Vinyl aufgenommen, die anschließend als Vinylmaster zur Vervielfältigung verwendet werden konnten. Die Bandmaschine ermöglichte neue Aufnahmetechniken – z. B. das *Overdubbing* – sodass standardmäßig auf Magnetband aufgenommen wurde. Dies erforderte vor der Vervielfältigung eine Übertragung von Magnetband auf einen Vinylmaster – das Mastering (Cousins & Hepworth-Sawyer, 2013; Owsinski, 2009).

Zu dieser Zeit bestand die Aufgabe der Mastering-Ingenieure/-innen darin, die Aufnahme vom Band so effektiv wie möglich auf das Vinyl zu übertragen und dabei gleichzeitig die akustische Integrität der Aufnahme zu bewahren (Cousins & Hepworth-Sawyer, 2013).

Erst mit der kommerziellen Verbreitung der Stereoschallplatte Ende der 1950er-Jahre begannen Mastering-Ingenieure/-innen langsam und wohl bedacht einen akustischen Fingerabdruck beim Mastering zu hinterlassen, indem sie beim Übertragen auf Vinyl Equalizer und Kompressoren verwendeten. Dies hatte aber nicht unbedingt kreative Gründe, sondern vor allem technische. Bei lauten Aufnahmen, oder Aufnahmen mit hohem Bassanteil, wird die Rille der Schallplatte breiter. Zum einen konnte bei zu hohen Pegeln der Schneidestichel kaputt gehen, zum anderen verringert sich der Platz auf der Schallplatte, was in einer verkürzten Spielzeit resultiert. Bei zu leisen Aufnahmen hingegen wird das Rauschen der Vinyl-

Platte deutlich hörbar. Deshalb ist es notwendig, den Dynamikbereich durch Kompression sowie den Frequenzbereich durch Filter zu begrenzen (Noltemeyer, 2012; Owsinski, 2009).

In den folgenden Jahrzehnten entwickelte sich die Arbeit der Mastering-Ingenieure/-innen mehr und mehr zu einem hochkomplexen Prozess, der sich deutlich von der eigentlichen Musikproduktion abgrenzte. Zeitgleich bekam das Mastering durch den Einsatz von Equalizern auch mehr kreativen Einfluss eingeräumt, sodass nicht nur Fehlerkorrekturen durchgeführt wurden, sondern auch die Tonalität eines Mixes beim Mastering angepasst wurde (Cousins & Hepworth-Sawyer, 2013).

Mit dem Einzug der Compact Disc (CD) Anfang der 1980er-Jahre begann für die Musikbranche die digitale Neuzeit, während die Mastering-Prozessoren immer noch identisch mit denen der Vinylzeiten waren. Die Eigenschaften der CD boten technisch bessere Voraussetzungen gegenüber einer Schallplatte: verlängerte Spielzeit, uneingeschränkter Frequenzbereich und eine geringere Relevanz der Monokompatibilität. Außerdem spielten die Pegel weder bei der Spielzeit noch bei der Erstellung des Mediums eine Rolle, sodass sich die Möglichkeit bot, deutlich lautere Pegel zu erzeugen (Noltemeyer, 2012; Owsinski, 2009). Stattdessen gab es eine obere Grenze für den Pegel und es traten neuen Schwierigkeiten wie Quantisierungsfehler bei leisen Pegeln auf (Cousins & Hepworth-Sawyer, 2013).

Mitte der 1990er-Jahre begann die Ära der digitalen Verbreitung über das Internet. Diese wurde ermöglicht durch die geringe Dateigröße des neuen Formats *MPEG-1 Audio Layer 3* (MP3), das bis heute noch Verwendung findet (Owsinski, 2009).

Im Zuge der Digitalisierung der Musikindustrie wurden auch für das Mastering neue Tools entwickelt, wie z. B. der digitale *Limiter*. Dieser ermöglichte durch Kompression eine Lautheit zu erreichen, die mit analogen Kompressoren für CD kaum erreichbar war. Dadurch wurde der Fokus beim Mastering vermehrt auf das Erreichen der größtmöglichen Lautheit gerichtet, während klangliche Qualität vernachlässigt wurde (Noltemeyer, 2012). Glücklicherweise entwickelt sich der Trend der maximalen Lautheit immer weiter zurück (Izhaki, 2008).

Während Mastering-Ingenieure/-innen zu Beginn ihrer Tätigkeit noch hauptsächlich mit dem Transfer von Magnetband auf Vinyl und dem korrekten Schneiden des Vinylmasters beschäftigt waren, arbeiten sie heutzutage mit technisch aufwendigen Plug-ins und erstellen digitale Master für die Verbreitung übers Internet. Mastering, wie es heute durchgeführt wird, entspricht eher dem Arbeitsschritt, der vor dem Schneiden auf Vinyl durchgeführt wurde. In diesem Schritt wurden das Audiomaterial für das Scheiden mit Hilfe von Regelverstärkern und Equalizern vorbereitet. Nichtsdestotrotz bildet Mastering immer noch den letzten Bearbeitungsschritt in der Produktionskette und ohne die technischen Einschränkungen und Entwicklungen der vorangegangen Jahre und die daraus entstandenen

Techniken, wäre das kreative Mastering wie es heute praktiziert wird nicht möglich (Cousins & Hepworth-Sawyer, 2013).

2.2 Was ist das Ziel vom Mastering?

Wie bereits oben erwähnt, wird beim Mastering ein Song oder eine Sammlung von Songs – ein Album – abschließend bearbeitet. Das bedeutet, dass hier die letzte Möglichkeit besteht Korrekturen durchzuführen sowie das Klangbild und die Lautheit für die Wiedergabe bei den Konsumenten/-innen zu optimieren. Bei einem ganzen Album wird der Hörfluss, das Klangbild aller Songs, das *Timing* und die Stille zwischen den Titeln so angepasst, dass alle Songs als Teil eines Gesamtwerkes verstanden werden (Owsinski, 2009).

Die Kunst besteht darin, den finalen Mix nochmal aufzuwerten und für den Konsumierenden noch eindrucksvoller zu gestalten und den Song damit zu verbessern. Dabei ist das Ergebnis nahezu ausschließlich von den Mastering-Ingenieuren/-innen, deren Können, Erfahrung und Geschmack abhängig (Owsinski, 2009). Die Ingenieure/-innen müssen bei jedem Song entscheiden, wie viel Bearbeitung ein Mix wirklich benötigt. In manchen Fällen sind lediglich minimale, in anderen wiederum große Eingriffe notwendig, um ein optimales Ergebnis zu erzielen. Dennoch kann ein Master nur in einem gewissen Rahmen einen Mix verbessern. Dementsprechend ist für einen exzellenten Master auch ein exzellenter Mix notwendig (Cousins & Hepworth-Sawyer, 2013).

Neben der kreativen Bearbeitung soll beim Mastering einerseits sichergestellt werden, dass das Endergebnis auch außerhalb seiner Produktionsumgebung bei den Endkonsumenten/-innen bestmögliche Qualität hat und andererseits, dass ein gewisser Industriestandard erreicht wird und das Endprodukt konkurrenzfähig ist. Dies trifft weniger auf Klassik- und Jazz-Musik zu, sondern hauptsächlich auf populäre Musikrichtungen wie Pop, Rock, R&B (Owsinski, 2009). Konkurrenzfähig bezieht sich in diesem Zusammenhang vor allem auf die Lautheit und den Ansatz, dass aufgrund des nicht-linearen Frequenzgangs des menschlichen Gehörs lautere Musik als besser empfunden wird, weil tiefere und höhere Frequenzen besser wahrgenommen werden (Taylor, 2017). Seitdem dies in den 1950er-Jahren entdeckt wurde, ist es bei Radioproduktionen zur Selbstverständlichkeit geworden, dass die Mastering-Ingenieure/-innen die Songs für eine maximale Lautheit optimieren (Owsinski, 2009).

Mit Hilfe von *Limitern* kann die wahrgenommene Lautheit erhöht und der durchschnittliche Pegel des Masters angehoben werden (Cousins & Hepworth-Sawyer, 2013). *Limiter* ermöglichen eine Abwärtskompression mit vergleichsweise wenig Klangverfärbung im Vergleich zu den meisten Kompressoren (Katz, 2010). Deshalb werden sie häufig als letzte

Instanz verwendet, um die größtmögliche Lautheit zu erreichen, ohne dass es dabei zu Verzerrungen kommt (Owsinski, 2009).

Trotz all der Möglichkeiten war und bleibt die Priorität beim Mastering, dass die akustische Integrität des finalen Mixes bewahrt wird und dass die Bearbeitungen der Mastering-Ingenieure/-innen „unsichtbar“ (Katz, 2010, S. 143) ist: „ist der Sound hörbar manipuliert worden, hat [der Mastering-Ingenieur] seinen Job nicht korrekt ausgeführt.“ (Katz, 2010, p. 143).

Dabei spielt die Abhörsituation in einem Mastering-Studio eine entscheidende Rolle, denn um Korrekturen und Verbesserungen durchzuführen, müssen zuerst Fehler gefunden werden. Können diese Fehler aufgrund der Raumbegebenheiten nicht gehört werden, oder werden durch Equalizer Überbetonung des Raums statt des Mixes entzerrt, verfehlt das Mastering seinen Zweck (Owsinski, 2009). Nicht zuletzt bringen die Mastering-Ingenieure/-innen, die das Projekt im optimalen Fall beim Mastering zum erste Mal hören, durch ihre unvoreingenommenen Ohren eine gewisse Objektivität mit und können mit ihrer Erfahrung und der optimierten Abhörsituation den Mix der Mixing-Ingenieure/-innen auf hohem Niveau beurteilen (Cousins & Hepworth-Sawyer, 2013).

Mastering-Ingenieur Bobby Owsinski fasst die Arbeitsschritte in wenigen in Funktionen zusammen:

> „die Pegel der verschiedenen Programmelemente maximieren, die Ausgeglichenheit des Frequenzbandes beibehalten und die Hauptfunktionen der DAW einsetzen, wie zum Beispiel editieren, Fades gestalten und Pausen zwischen den Songs festlegen sowie die Eingabe von PQ und ISRC.“ (Owsinski, 2009, S. 51)

2.3 Mastering Prozessoren

Beim Mastering werden eine Reihe unterschiedlicher Audio-Prozessoren verwendet, um den finalen Mix zu optimieren. Grundlegend sind zwei Arten von Prozessoren: Filter und Regelverstärker. Filter werden zur Entzerrung und verändern des Timbres verwendet, während Regelverstärker zur Bearbeitung des Dynamikumfangs dienen (Cousins & Hepworth-Sawyer, 2013).

2.3.1 Filter und Equalizer

Filter spielten seit der Übertragung von Band auf Vinyl eine Rolle beim Mastering. Jedoch nicht so ausgeprägt und in solcher Vielfalt wie heute. Heutzutage sind Filter aus dem

Mastering nicht mehr wegzudenken und es kommen die unterschiedlichsten Typen zum Einsatz (Cousins & Hepworth-Sawyer, 2013).

Mithilfe von Filtern lässt sich das Frequenzspektrum eines Audiosignals verändern. Dabei werden lineare Verzerrungen genutzt, um bestimmte Frequenzen oder Frequenzbereiche anzuheben oder abzusenken. Diese Funktion wird beim Mastering sowohl für Korrekturen im Frequenzspektrum, z. B. überbetonte Resonanzen, als auch zum Anpassen des Klangbilds verwendet (Cousins & Hepworth-Sawyer, 2013; Noltemeyer, 2012). Deshalb gehören Filter zu den wichtigsten Werkzeuge beim Mastering und bieten in verschiedenen Ausführungen abgeänderte Funktionen (Owsinski, 2009).

Auch wenn die meisten Arten von Filter gleichermaßen beim Mix und beim Mastering zum Einsatz kommen, zeichnen sich Mastering-Filter häufig dadurch aus, dass sie keine stufenlosen Regler haben. Stattdessen werden gerasterte Regler verwendet, die einen exakten Recall, also ein Wiederaufrufen einer Einstellung, ermöglichen. Das ist für Revision bei Hardware-Prozessoren relevant (Owsinski, 2009).

Cut-Off-Filter

Cut-Off-Filter beschneiden das Frequenzspektrum ab einer definierten Grenzfrequenz, der *Cut-Off-Frequenz*. Durch einen *Hochpass-Filter* bleiben alle Frequenzen oberhalb der *Cut-Off-Frequenz* unverändert, während die Frequenzen unterhalb der *Cut-Off-Frequenz* stark abgesenkt werden. Gegensätzlich arbeitet ein *Tiefpass-Filter*: Frequenzen unter der Grenzfrequenz bleiben unverändert; oberhalb werden sie abgesenkt (Izhaki, 2008).

Shelving-Filter

Ähnlich wie die Filter bearbeitet der *Shelving-Filter* auch ausschließlich die Frequenzen der oberen und unteren Grenze des Frequenzspektrums. Allerdings erlaubt er nicht nur das Absenken, sondern auch das Anheben der Frequenzbereiche oberhalb oder unterhalb der Grenzfrequenz (Cousins & Hepworth-Sawyer, 2013; Izhaki, 2008). So können z. B. tiefe Frequenzen abgesenkt und hohe Frequenzen angehoben werden, um das Lied offener und weniger dumpf klingen zu lassen (Noltemeyer, 2012).

Grafischer Equalizer

Equalizer vereinen verschiedene Filtertypen. Grafische Equalizer zeichnen sich durch ihre vordefinierten Frequenzbänder aus. Die Frequenzbänder unterteilen den Frequenzbereich in gleichgroße Abschnitte, sodass diese einzeln angehoben oder abgesenkt werden können. Eine gängige Unterteilung ist ein Terzabstand, die den Frequenzbereich zwischen 20 Hz und 20 kHz in 31 Bänder unterteilt (Izhaki, 2008).

Für das Mastering ist diese Art von Equalizer nicht gut geeignet, da er aufgrund seiner festgelegten Frequenzbänder und *Güte* (die Bandbreite), keine präzisen Einstellungen zulässt (Cousins & Hepworth-Sawyer, 2013; Noltemeyer, 2012).

Parametrischer Equalizer

Im Gegensatz zum grafischen Equalizer erlaubt der voll-parametrische Equalizer die Einstellung der Parameter und ist damit ein wichtiges Werkzeug beim Mastering. Bei einem voll-parametrischen Equalizer sind *Gain,* Frequenz und *Güte* für die einzelnen Frequenzbänder einstellbar. Das bietet die nötige Präzision, die beim Mastering gefordert wird, um das bereits bearbeitete Audiomaterial nochmals gezielt zu optimieren (Cousins & Hepworth-Sawyer, 2013).

Dynamischer Equalizer

Dynamische Equalizer erweitern die Funktionen von parametrischen Equalizern um eine dynamische Anpassung des *Gains*. Damit lassen sich Veränderungen des Frequenzgangs abhängig vom bestehenden Frequenzgang umsetzen. Die zusätzlichen Parameter (*Threshold*, *Ratio*, *Attack* und *Release*) erinnern an einen *Multiband-Kompressor* (siehe Kapitel 2.3.2). Mit diesen Parametern kann das Frequenzband abhängig vom Signal angehoben oder abgesenkt werden (Izhaki, 2008). Das ist eine beim Mastering häufig benötigte Funktion. Ändert sich im Verlauf eines Songs die Instrumentation, verändert sich gegebenenfalls auch der Frequenzgang, sodass die Equalizer-Einstellungen des einen Abschnitts nicht mehr für den Frequenzgang des nächsten oder übernächsten Abschnitts geeignet sind (Noltemeyer, 2012).

Phasen-Linearer-Equalizer

Ebenfalls eine Abwandlung von parametrischen Equalizern sind Phasen-Lineare-Equalizer. Während bei parametrischen Equalizern einstellungsabhängige Phasenverschiebungen entstehen, zeichnen sich Phasen-Lineare-Equalizer durch einen linearen Phasengang aus, der von den Einstellungen unabhängig ist. Dadurch können die entstehenden klanglichen Veränderungen auf die Frequenz beschränkt werden (Cousins & Hepworth-Sawyer, 2013).

Im Mastering kann diese Funktion hilfreich sein, um bei Korrekturen die Unversehrtheit des Klangbilds zu bewahren. Geht es allerdings um die Veränderung des Gesamtklangbilds, sind die Phasenverschiebungen von einfachen Equalizern teilweise sogar erwünscht, da sie einen bestimmten Klang erzeugen, der bei einem Phasen-Linearen-Equalizer nicht entsteht (Cousins & Hepworth-Sawyer, 2013).

M/S-Equalizer

Stereo-Equalizer besitzen notwendigerweise zwei Kanäle. Diese zwei Kanäle können bei einem M/S-Equalizer dazu genutzt werden, das Signal nicht in linkes und rechtes Signal zu unterteilen, sondern in ein Mitten- und ein Seiten-Signal.

Diese Technik bietet beim Mastering viele Vorteile, da so gezielter auf einzelne Elemente im fertigen Mix eingegriffen werden kann (Noltemeyer, 2012).

2.3.2 Regelverstärker

Die zweite Gruppe grundlegender Audioprozessoren beim Mastering sind die Regelverstärker (Owsinski, 2009). Während Filter das Timbre der Musik bearbeiten, wird mit Regelverstärkern primär die akustische Dynamik der Musik verändert (Cipriani & Giri, 2019). Als Dynamikbereich wird der Bereich zwischen dem leisesten und dem lautesten Pegel beschrieben (Izhaki, 2008). Dies geschieht sowohl aus technischen, als auch aus kreativen Gründen (Cipriani & Giri, 2019).

Bei Regelverstärkern wird die Spannung des Eingangssignals genutzt, um den Faktor der Verstärkung zu verändern. Das bedeutet, dass der Pegel des Eingangssignals die Verstärkung des Ausgangssignals bestimmt. Dadurch kann der Dynamikbereich eines Signals eingeschränkt oder vergrößert werden (Weinzierl, 2008). Werden die hohen Pegel abgesenkt und gleichzeitig die leisen Pegel angehoben, ist es theoretisch möglich, den Dynamikbereich maximal einzuschränken, sodass das Signal durchgehend den gleichen Pegel hat. Praktisch wird das komprimierte Resultat wahrscheinlich nicht überzeugen (Friesecke, 2014).

Die Idee hinter Regelverstärkern war eine automatische Kontrolle der Dynamik eines Signals. Dabei sollte die Bearbeitung ursprünglich möglichst unauffällig sein. Die hörbaren Auswirkungen der entstehenden Kompression sind allerdings ein willkommener Effekt, sodass Regelverstärker nicht nur technisch, sondern auch kreativ verwendet werden (Izhaki, 2008).

Mittlerweile ist nahezu jedes konsumierte Audiomaterial mindestens durch einen Kompressor, wahrscheinlich sogar durch mehrere komprimiert worden (Dickreiter et al., 2014). Insbesondere bei populären Musikstilen wie Pop, Techno, House etc. machen sich Produzent/-innen bei der Produktion Regelverstärker zunutze, um die Programmdynamik auf ein Minimum zu reduzieren. Zum einen verleiht es dem Mix einen bestimmten Klang, zum anderen entsteht dadurch eine größere Lautheit, die Konsumenten/-innen aus Fernsehen und Radio gewohnt sind (Cipriani & Giri, 2019).

Wird die entstehende Wirkung von korrekt genutzten Regelverstärkern, insbesondere von Kompressoren und *Limitern*, betrachtet kann erkannt werden, dass der Einsatz

gewünscht ist. Auf der einen Seite wird die Dynamik eines Signals automatisch begrenzt, wodurch das Signal kontrollierbarer wird. Dadurch ist ein manuelles Eingreifen durch Fader-Automationen nicht mehr unbedingt notwendig. (Dickreiter et al., 2014). Auf der anderen Seite bewirkt die Kompression ein lauteres, dichteres und druckvolleres Klangbild (Izhaki, 2008).

Dennoch kann sich Kompression auch negativ auswirken, denn die Begrenzung des Dynamikbereichs sorgt für eine Veränderung der Verhältnisse innerhalb eines Signals. Das führt dazu, dass Tiefe verloren geht, weil leise Signale ungewollt aus dem Hintergrund in den Vordergrund treten oder dass das Stereo-Bild zusammenfällt und kleiner wird (Cipriani & Giri, 2019; Katz, 2010).

Zu starke Kompression, z. B. um eine bestimmte Lautheit zu erreichen, führt meistens zu einem unnatürlichen Klang, der von Konsumenten/-innen sogar als langweilig oder anstrengend empfunden werden kann (Izhaki, 2008). Zudem verringert sich auf eine gewisse Weise mit der Begrenzung der Dynamik auch die künstlerische Freiheit, die eine reale akustische Umgebung bietet (Cipriani & Giri, 2019).

2.3.2.1 Parameter

Obwohl es Regelverstärker in vielen verschiedenen Ausführungen gibt, die unterschiedliche Bauarten aufweisen, teilen sie sich größtenteils die gleichen Parameter (Dickreiter et al., 2014). Diese gängigen Parameter werden im Folgenden aufgeführt:

Input-Gain

Mit dem Parameter *Input-Gain* ist eine Veränderung des Eingangspegels möglich. Bei Regelverstärkern mit einstellbarem *Threshold* dient der *Input-Gain* der Anpassung an den Arbeitspegel des Regelverstärkers. Bei Regelverstärkern, die einen festen *Threshold* haben, wird mit dem *Input-Gain* das Signal an den festen *Threshold* angepasst (Friesecke, 2014).

Threshold

Der *Threshold* ist der Schwellenwert, ab dem der Regelverstärker das anliegende Signal verändert (Katz, 2010). Dieser Wert kann je nach Anwendung ein unterer oder ein oberer Schwellenwert sein, sodass z. B. ein Regelverstärker eingreift, sobald der *Threshold* unterschritten wurde und ein anderer, wenn der *Threshold* überschritten wurde (Friesecke, 2014).

Einige Regelverstärker besitzen sogar zwei *Thresholds* – einen unteren und einen oberen. Der Bereich zwischen den beiden *Thresholds*, indem der Regelverstärker nicht aktiv ist, wird als *Hysterese* bezeichnet (Friesecke, 2014).

Ratio

Die *Ratio* stellt das Verhältnis der Veränderung des Ausgangssignal ein und bezieht sich dabei stets auf den Pegel. Sie wird häufig mit $Eingang : Ausgang$ angegeben. Im Fall eines Kompressors halbiert der eingestellte *Ratio*-Wert von $2:1$ den Pegel, der den *Threshold* überschreitet. Andersherum verdoppelt ein *Ratio*-Wert von $1 : 2$ bei einem *Expander* (siehe S. 16) den Pegel, der den *Threshold* unterschreitet (Friesecke, 2014; Noltemeyer, 2012).

Mit einer Kennlinie lässt sich der *Threshold* und die *Ratio* gut visualisieren (siehe Abbildung 2.1). Diese Linie stellt die Abhängigkeit des Ausgangspegels (y-Achse) vom Eingangspegel (x-Achse) dar (Dickreiter et al., 2014). Ist der *Ratio*-Wert $1:1$, verläuft die Linie linear, denn der Eingangspegel entspricht dem Ausgangspegel. Ist die *Ratio* größer oder kleiner als 1 ist der Verlauf der Kennlinie nicht-linear und besteht aus zwei Abschnitten: einem unveränderten Abschnitt und einem abgeflachten oder stark ansteigenden Abschnitt. Während der erste Teil der Kennlinien die Relation von Eingangspegel zu Ausgangspegel bis zum *Threshold* beschreibt, beschreibt der zweite Teil das Verhältnis nach dem Überschreiten des *Thresholds*. Dementsprechend kann der Knick, der die Kennlinie in zwei lineare Graphen unterteilt als der *Threshold* erkannt werden.

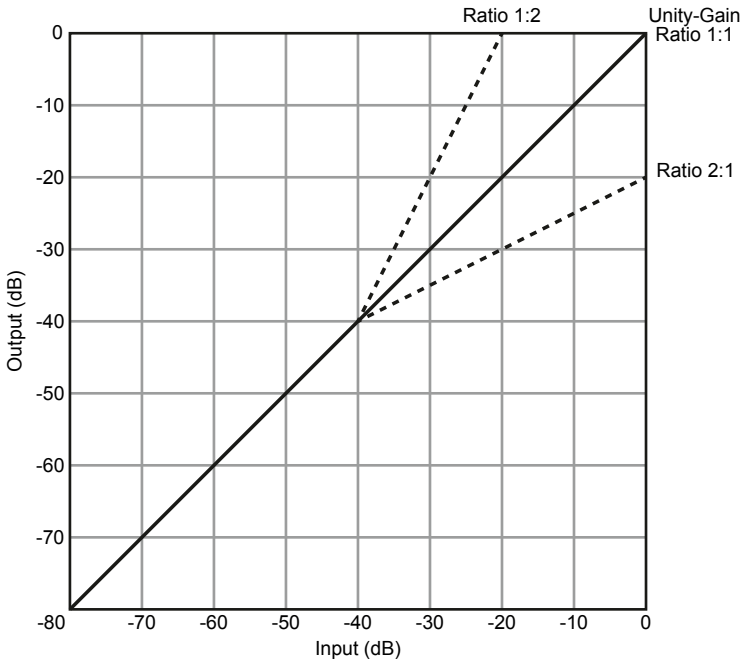

Abbildung 2.1: Kennlinie mit Ratio 1:2 und 2:1. Quelle: In Anlehnung an Izhaki, 2008, S. 265

Knee

Wie zuvor beschrieben ist der *Threshold* der Punkt, ab dem beim Überschreiten oder Unterschreiten der Regelverstärker aktiv wird und wird bei der Kennlinie durch einen Knick sichtbar. Diesen Knick kann man mit dem Parameter *Knee* verändern, sodass er eine Kurve beschreibt, die *Soft Knee* genannt wird (siehe Abbildung 2.2). Der Knick hingegen wird als

Hard Knee bezeichnet. Bei Verwendung eines *Soft Knees* wird im Bereich um den *Threshold* die *Ratio* langsam vergrößert. D. h. der Regelverstärker wird bereits unterhalb des *Threshold* aktiviert. Dadurch entsteht ein Ansteigen der *Ratio* und die Kompression wird sanfter und weniger abrupt (Katz, 2010).

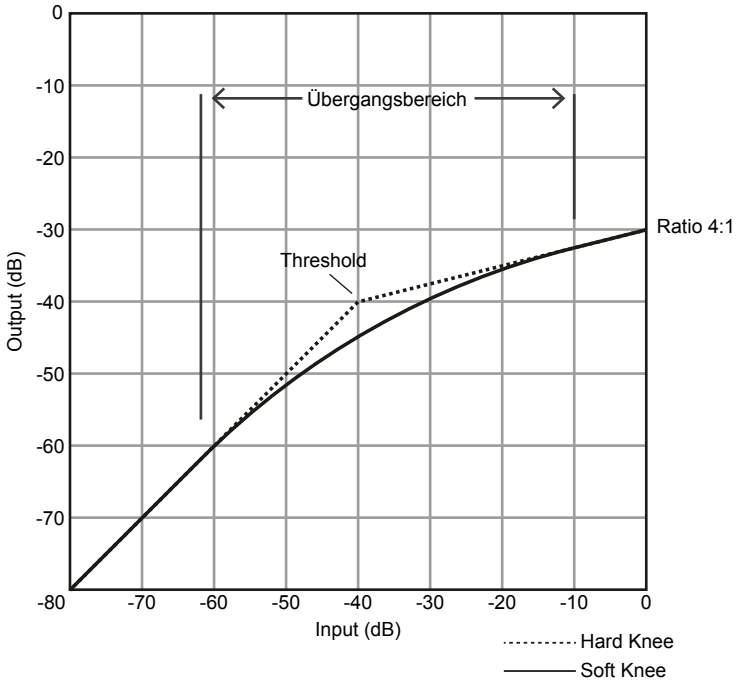

Abbildung 2.2: Kennlinie mit *Hard* und *Soft Knee*. Quelle: In Anlehnung an Izhaki, 2008, S. 289

Zeit-Konstanten

Die Parameter *Threshold*, *Ratio* und *Knee* bestimmen den Faktor der Verstärkung des Signals. Darüber hinaus gibt es Parameter, mit denen das zeitliche Verhalten eines Regelverstärkers eingestellt werden kann.

Die *Attack-Zeit* bestimmt die Reaktionszeit die der Regelverstärker benötigt, um das Eingangssignal mit der ganzen *Ratio* zu bearbeiten, nachdem der *Threshold* überschritten wurde (Katz, 2010). Tatsächlich arbeitet der Regelverstärker ab dem Zeitpunkt der Überschreitung/Unterschreitung des *Thresholds*. Dadurch beginnt die Dynamikbearbeitung nach Ablauf der *Attack-Zeit* nicht erst bei 0 Prozent, sondern bereits bei 63 Prozent und hat 100 Prozent nach etwa drei- bis fünffacher *Attack-Zeit* erreicht (Friesecke, 2014).

Mit der *Attack-Zeit* kann man den Klang des Regelverstärkers maßgeblich bestimmen. Insbesondere Transienten[2] können mit diesem Parameter beeinflusst werden. Bei einem Schlagzeug, das durch den perkussiven Anschlag Transienten erzeugt, können bei einer längeren *Attack-Zeit* die Transienten den Regelverstärker unverändert passieren, während sie bei einer kurzen *Attack*-Zeit stark verändert werden können (Noltemeyer, 2012).

[2] Transient: Einschwingvorgang eines Tons.

Der zweite zeitabhängige Parameter ist die *Release*- oder *Recovery*-Zeit. Diese gibt die Dauer an, die der Regelverstärker benötigt, damit das Ausgangssignal nach dem Unterschreiten/Überschreiten des *Thresholds* wieder denselben Pegel hat wie das Eingangssignal (Katz, 2010). Es ist die Dauer, die es braucht, bis der Regelverstärker abgeschaltet ist. Je kürzer diese Zeit ist, desto abrupter ist der Wechsel zurück zu *Unity-Gain*. Deshalb sind längere *Release*-Zeiten unauffälliger und entsprechen unserer Hörgewohnheit (Friesecke, 2014). Sind die Zeiten allerdings zu lang, können sie Einfluss auf darauffolgende Transienten haben und in einer stärkeren Dynamikbearbeitung resultieren (Noltemeyer, 2012).

Make-up-Gain

Da durch die Dynamikbearbeitung eines Regelverstärkers die Pegel des Ausgangssignals verändert werden, ist es logisch, dass das bearbeitete Signal im Vergleich zum Eingangssignal lautere oder leisere Pegel hat. Das *Make-up-Gain* ist dazu da, denn Unterschied auszugleichen und das bearbeitete Signal anzupassen (Noltemeyer, 2012).

2.3.2.2 Verschiedene Arten von Regelverstärkern

Regelverstärker weisen viele verschiedene Funktionen auf, weshalb eine anwendungsbezogene Abgrenzung nützlich ist. Dazu werden im Folgenden die gängigen Regelverstärker vorgestellt:

Kompressor

Allgemein verringert ein Kompressor in Abhängigkeit eines anliegenden Signals den Dynamikbereich des Ausgangssignals. Dies kann auf unterschiedliche Art und Weise passieren und hängt von den eingestellten Parametern ab (Friesecke, 2014).

Kompressoren lassen sich in zwei Gruppen unterteilen: *Abwärts-Kompressor* (siehe Abbildung 2.3) und *Aufwärts-Kompressor* (siehe Abbildung 2.3) (Cipriani & Giri, 2019). Sowohl die *Abwärts*- als auch die *Aufwärts-Kompression* verringert die Dynamik und erzeugt dadurch ein dichteres Klangbild und eine größere Lautheit (Noltemeyer, 2012).

Wenn von Kompression die Rede ist, ist in den meisten Fällen *Abwärts-Kompression* gemeint. Diese Art erlaubt ein Anheben des Gesamtpegels z. B. mit dem *Make-up-Gain* nachdem laute Pegel oberhalb des *Thresholds* reduziert wurden. Dadurch werden im Verhältnis leise Signale angehoben (Cousins & Hepworth-Sawyer, 2013; Izhaki, 2008).

Aufwärts-Kompression verringert den Dynamikumfang, indem Pegel, die unterhalb des *Thresholds* liegen, angehoben werden. Dabei bleiben die lauten Pegel unverändert, aber die untere Dynamikgrenze verschiebt sich nach oben, wodurch die Lautheit größer wird (Cipriani & Giri, 2019; Dickreiter et al., 2014).

Da die *Aufwärts-Kompression* unter anderem auch durch den Einsatz eines parallel eingesetzten *Abwärts-Kompressors* erzielt werden kann, kommen reine *Aufwärts-Kompressoren* selten vor (Cipriani & Giri, 2019).

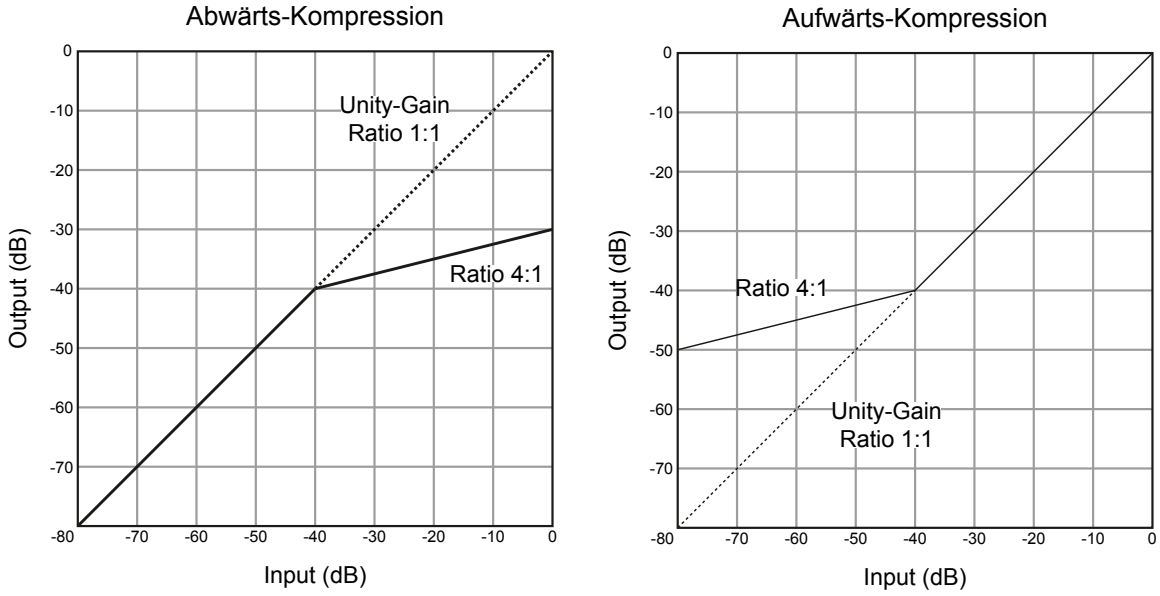

Abbildung 2.3: Kennlinien für Abwärts- und Aufwärts-Kompressoren. Quelle: In Anlehnung an Izhaki, 2008, S. 267

Multiband-Kompressor

Der *Multiband-Kompressor* ist eine Variation zum oben beschriebenen Breitband-Kompressor. Anders als bei diesem wird beim *Multiband-Kompressor* nicht das gesamte Frequenzspektrum gleichermaßen bearbeitet, sondern das Signal wird zunächst in mehrere Frequenzbereiche unterteilt und dann separat komprimiert. Dazu ist für jeden Frequenzabschnitt ein eigener Kompressor verfügbar, der unabhängig eingestellt werden kann (Dickreiter et al., 2014).

Das unabhängige Komprimieren der unterschiedlichen Frequenzbereiche ermöglicht beim Mastering einen gezielteren Eingriff, um störende Resonanzen zu bearbeiten oder das Klangbild zu verändern (Noltemeyer, 2012). Bei der Veränderung des Klangbilds ist es möglich, bestimmte Frequenzbereiche stark, andere nur leicht oder gar nicht zu komprimieren, wodurch ein finaler Mix im Mastering interessant gestaltet werden kann. Außerdem ist es möglich, mit einem *Multiband-Kompressor* eine höhere Lautheit zu erzielen als mit einem Breitband-Kompressor (Katz, 2010).

De-Esser

Der *De-Esser* basiert er auf einer ähnlichen Idee wie der Multiband-Kompressor, obwohl er nur ein Band besitzt (Cipriani & Giri, 2019). Mit ihm kann ein bestimmter Frequenzbereich gezielt komprimiert werden. Da es sich häufig um einen Frequenzbereich im mittleren bis

hohen Frequenzspektrum – dort wo die Sibilanten[3] hörbar sind – handelt und Sibilanten insbesondere bei Stimmaufnahmen auftreten, wird der Kompressor als *De-Esser* bezeichnet, denn er wird dazu eingesetzt, die Zischlaute zu reduzieren (Owsinski, 2009).

Um den Regelverstärker nur zu aktivieren, wenn ein bestimmter Frequenzbereich den *Threshold* überschreitet, ist eine Bearbeitung des Steuersignals notwendig. Dieses kann gefiltert und somit in seinem Frequenzbereich eingeschränkt werden (Dickreiter et al., 2014).

Beim Mastering wird der *De-Esser* nicht nur verwendet, um die S-Laute des Gesangs zu kontrollieren, sondern häufig auch, um andere hochfrequente Signale, wie z. B. eine Hi-Hat zu komprimieren (Noltemeyer, 2012).

Limiter

Werden bei einem Kompressor die *Ratio* auf ein Maximum und die *Attack* auf ein Minimum gestellt, wird der Kompressor zu einem *Begrenzer* oder *Limiter* (Weinzierl, 2008). Wie ein Abwärts-Kompressor senkt der *Limiter* Pegel oberhalb des *Thresholds* ab, allerdings nicht im Verhältnis zum Eingangssignal, sondern auf den Pegel des *Thresholds*. Dadurch kann das Signal den *Threshold* nicht überschreiten, weshalb der *Threshold* beim *Limiter* auch als *Ceiling* bezeichnet wird (Izhaki, 2008).

Indem der *Limiter* die lautesten Pegel bis zu einem bestimmten Wert reduziert, können Übersteuerungen effektiv vermieden werden. Um diesen Schutz vor Übersteuerung gewährleisten zu können, benötigt der Regelverstärker nicht nur eine hohe *Ratio*, sondern ebenfalls eine schnelle *Attack*-Zeit von wenigen Millisekunden, die ausnahmslos alle Transienten einfängt. Durch diese kurzen *Attack*-Zeiten können jedoch ungewünschte Verzerrungen entstehen (Friesecke, 2014). Diesen Verzerrungen kann mit einem *Look-Ahead*, einer *Vorschau*, des Steuersignals entgegengewirkt werden. Im Vergleich zum Steuersignal wird das Hauptsignal verzögert. Da das Steuersignal den Kompressor nun aktiviert, bevor das Hauptsignal den *Threshold* überhaupt überschritten hat, können die *Attack*-Zeiten verlängert werden. Aufgrund des verzögerten Hauptsignals kann der Kompressor dennoch zum Zeitpunkt des maximalen Überschreitens des *Thresholds* die vollständige Pegelreduktion erreichen, allerdings mit einem weicheren Übergang als ohne *Look-Ahead* und mit einer *Attack*-Zeit gegen 0 Millisekunden (Cipriani & Giri, 2019).

Limiter werden vor allem im Mastering eingesetzt. Dort ermöglichen sie das Erzeugen einer größeren Lautheit, da das gesamte Audiosignal verstärkt werden kann, während die Spitzen-Pegel von dem *Limiter* abgefangen und reduziert werden (Owsinski, 2009).

[3] Sibilant: Zischlaut

Expander

Während die bisher genannten Regelverstärker zur Begrenzung des Dynamikumfangs verwendet werden, wird ein *Expander* für die Erweiterung eingesetzt. Er kann als Pendant zum Kompressor verstanden werden. Dennoch gleichen die Parameter denen eines Kompressors mit Ausnahme der *Ratio*, die beim *Expander* größer als 1 (z. B. $1:2$) ist. Ebenso wie der Kompressor gibt es auch hier gegensätzliche Arten: den *Aufwärts-Expander* und den *Abwärts-Expander* (siehe Abbildung 2.4) (Dickreiter et al., 2014; Izhaki, 2008).

Durch einen *Abwärts-Expander* wird das Signal abgesenkt, sobald das Steuersignal den *Threshold* unterschreitet. Der *Aufwärts-Expander* hingegen verstärkt das Signal, wenn das Steuersignal den *Threshold* überschreitet (Izhaki, 2008).

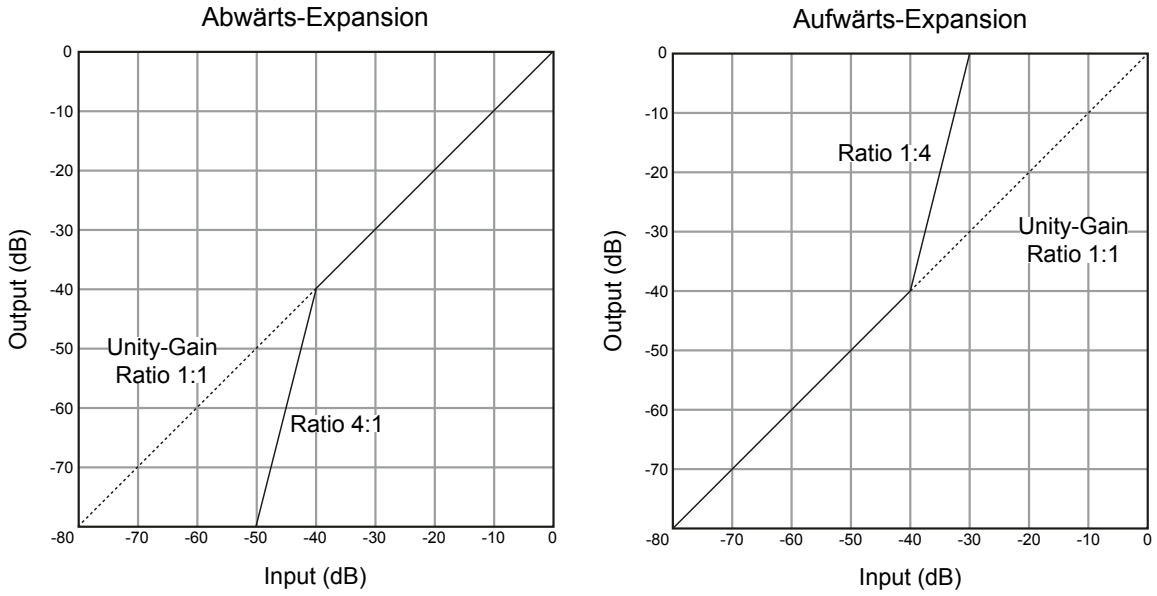

Abbildung 2.4: Kennlinien für *Abwärts-* und *Aufwärts-Expansion*. Quelle: In Anlehnung an Izhaki, 2008, S. 366

Gate

Das *Gate* verhält sich zum *Expander* ähnlich wie der *Limiter* zum Kompressor. Bei maximaler *Ratio* (gegen $\infty:1$) wird aus einem *Abwärts-Expander* ein *Gate* (Dickreiter et al., 2014). Diese Art eines Regelverstärkers kann Signale unterhalb des *Thresholds* nicht nur absenken, sondern auch stummschalten. Damit können Störgeräusche z. B. zwischen Gesangspassagen eliminiert werden (Friesecke, 2014).

Der wesentliche Unterschied zwischen einem *Gate* und einem *Expander* besteht in dem Umgang mit der *Ratio*. Beim *Expander* hängt die Reduzierung eines Pegels mit dem Pegel des Steuersignals zusammen. Beim *Gate* wird der Pegel immer auf einen festen Wert abgesenkt. Das Steuersignal bestimmt dabei nur, ob und wann das Signal abgesenkt wird (Izhaki, 2008).

2.4 Kompression beim Mastering

Früher wie heute ist es Aufgabe der Mastering-Ingenieure/-innen, die Dynamik des finalen Masters für ein Medium zu optimieren (Izhaki, 2008). Selbst wenn das Ziel einer Musikproduktion die möglichst natürliche Wiedergabe der Aufnahme sein soll, muss schlussendlich immer die Abhörsituation der Konsumenten/-innen in Betracht gezogen werden. Für eine bestmögliche Wiedergabe auf vielen unterschiedlichen Abspielsystemen ist es notwendig, die Dynamik eines Songs anzupassen. Neben der Anpassung kann der finale Mix durch gezielte Dynamikbearbeitung beim Mastering auch interessanter und lebendiger gestaltet, der Druck und die Lautheit erhöht und ein gewisser Klang erzeugt werden, der viele populäre Musikstile auszeichnet. (Katz, 2010). Dazu werden im Mastering vor allem drei Arten von Regelverstärkern mit unterschiedlichen Zielen verwendet. Um die Verhältnisse im Mix gezielt zu verändern, wird auf einen Multiband-Kompressor zurückgegriffen. Der *Limiter* kommt zum Einsatz, wenn es darum geht die Lautheit eines Songs anzuheben und der Breitband-Kompressor findet Verwendung in der allgemeinen Dynamikbearbeitung, aber auch indem er das Zusammenspiel verschiedener Elemente verändert, sodass sie mehr wie eine Einheit klingen. Dieser Effekt wird häufig als *Glue* bezeichnet (Cousins & Hepworth-Sawyer, 2013).

Mit *Glue* wird im Grunde genommen das Zusammenwirken von den einzelnen Song-Elementen beschrieben. Der bewusste Einsatz von Kompressoren auf einer Summe von Signalen erzeugt für alle Signale abhängig voneinander dieselbe Dynamikbearbeitung, Klangverfärbung und Verzerrung (Braddock et al., 2021). Dadurch entsteht eine einheitliche Bewegung in der Dynamik und eine Zusammengehörigkeit aller Signale (Cousins & Hepworth-Sawyer, 2013). Eine Studie zur Sprachverständlichkeit von Hörgeräten mit einem Breitband-Kompressor von Stone und Moore (2003) kommt zu dem Ergebnis, dass durch eine Kompression mit *Attack*-Zeiten von 1 bis 2 Millisekunden und *Release*-Zeiten um 360 Millisekunden mehrere Hörereignisse zu *einem* verschwimmen können und entsprechend als Einheit wahrgenommen werden. Obwohl die Praxisanwendung zur Erzeugung von *Glue* eher mittlere bis langsame *Attack*- und *Release*-Zeiten empfiehlt, zeigt die Studie, dass es einen Zusammenhang zwischen den Zeit-Konstanten eines Kompressors und der Zusammengehörigkeit mehrerer komprimierter Schallereignisse gibt (Braddock et al., 2021).

2.5 Funktion eines Kompressors

Wie bereits in Kapitel 2.3.2 erläutert, gibt es verschiedene Parameter innerhalb eines Regelverstärkers, die für die Bearbeitung des Signals erforderlich sind. Im Rahmen der Erläuterung sind lediglich die Parameter genannt und erklärt worden, nicht jedoch ihre technische Funktion. Dies soll im Folgenden anhand eines digitalen Breitband-Kompressors erfolgen.

Obwohl es nicht den *einen* Aufbau eines Breitband-Kompressors gibt, gibt es dennoch eine Grundlage, auf der die meisten Kompressoren dieser Art basieren. Da das Eingangssignal mit einem Steuersignal multipliziert wird und sich daraus das Ausgangssignal ergibt, muss es innerhalb des Kompressors mindestens zwei Signalpfade geben: einen für das Hauptsignal, der Signalweg, und einen für das Steuersignal – der Steuersignalweg oder auch *Side-Chain*. Das Steuersignal wird innerhalb des *Side-Chains* aus einem Audiosignal generiert (Friesecke, 2014).

Es ist möglich, das Steuersignal sowohl aus dem zu komprimierenden Eingangssignal als auch aus einem weiteren, externen Audiosignal zu erzeugen. Wird das Hauptsignal als Steuersignal genutzt, gibt es zwei Möglichkeiten das Signal abzugreifen. Einerseits kann das Signal direkt nach dem Eingang gesplittet und auf den Hauptweg und den *Side-Chain* geroutet werden. Andererseits kann es auch erst am Ausgang gesplittet werden, sprich nachdem das Steuersignal bereits mit dem Hauptsignal multipliziert wurde, und zurück in den *Side-Chain* geroutet werden. Wird das Signal am Eingang gesplittet, spricht man von einem *Feed-Forward-Kompressor*, andernfalls von einem *Feedback-Kompressor* (Friesecke, 2014).

Soll das Steuersignal aus einem externen Audiosignal generiert werden, muss dieses am *Key-Eingang* des Kompressors anliegen. Das externe Audiosignal wird bei Musikern häufig als *Side-Chain* bezeichnet. Auch wenn dieser Ausdruck nicht inkorrekt ist, sorgt er bei der Beschreibung des Signalflusses für Verwirrung, weshalb das externe Audiosignal im Folgenden als *Key-Input* als bezeichnet wird. Da in diesem Fall zwei Signale, das Hauptsignal und der *Key-Input*, am Kompressor anliegen, ist ein Splitten der Signale nicht mehr notwendig (Cipriani & Giri, 2019).

Um innerhalb des *Side-Chains* aus einem Audiosignal ein Steuersignal zu erzeugen, durchläuft das Audiosignal verschiedene Module, die es verändern (siehe Abbildung 2.5) und mit denen Einfluss auf das finale Steuersignal genommen werden kann.

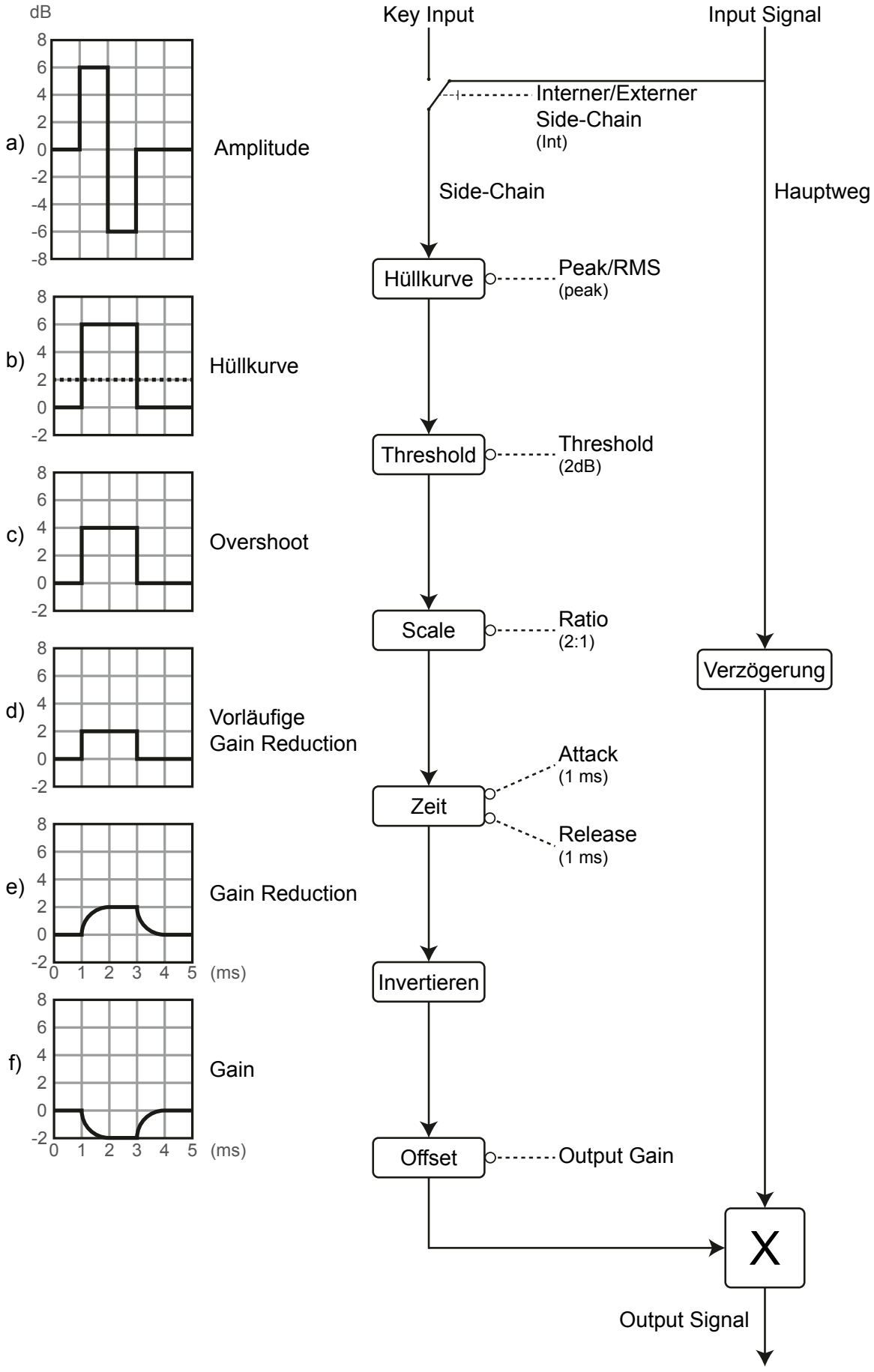

Abbildung 2.5: Signalfluss und Signalveränderung innerhalb eines Feed-Forward-Kompressors. Quelle: In Anlehnung an Cipriani und Giri, 2019, S. 378; Izhaki, 2008, S. 275

Hüllkurve

Zunächst wird mit dem Audiosignal eine Hüllkurve erzeugt. Diese Hüllkurve besteht im Gegensatz zum Audiosignal ausschließlich aus positiven Werten. Zum Generieren einer Hüllkurve gibt es verschiedene Optionen. Eine Möglichkeit ist die Bildung des Durchschnittswerts der positiven Amplituden über eine bestimmte Dauer an Samples (Cipriani & Giri, 2019). Bei einer Dauer von zwei Samples ergibt sich daraus:

$$y(n) = \frac{x(n) + x(n-1)}{2}$$

Formel 2.1 (Cipriani & Giri, 2019, S. 416)

Eine alternative Möglichkeit ist die *Gleichrichtung* der Amplituden-Werte. Dabei wird der Absolutbetrag der Amplituden-Werte bestimmt, wodurch alle negativen Werte positiv werden. Das Ergebnis der Betragsbildung wird im Anschluss mit einem Tiefpassfilter geglättet. Das ist erforderlich, um Verzerrungen des Ausgangssignals durch zu hohe Pegelsprünge innerhalb des Steuersignals zu verhindern (Cipriani & Giri, 2019).

Unabhängig von der gewählten Art der Hüllkurvenerzeugung gibt es zwei unterschiedliche Modi zur Pegelerkennung: *Peak* und *RMS*[4]. Die Art der Pegelerkennung bestimmt das Verhalten des Kompressors. Damit er auf die Spitzenwerte des Steuersignals regiert, ist der *Peak*-Modus der Pegelerkennung notwendig. Um auf den Effektivwert des Steuersignals zu reagieren, verwendet der Kompressor den *RMS*-Modus (Dickreiter et al., 2014). In vielen Fällen wird der *RMS*-Modus verwendet, da er mehr der menschlichen Wahrnehmung von Lautheit entspricht und weniger auffällige Effekte erzeugt (Self, 2015). Einige Kompressor-Modelle bieten auch ein Umschalten zwischen *Peak*- und *RMS*-Modus an (Izhaki, 2008).

Overshoot

Der *Overshoot* ist der Anteil der Hüllkurve der oberhalb des *Thresholds* liegt. Er hat einen direkten Einfluss auf die *Gain Reduction* (Izhaki, 2008). Für die Berechnung des *Overshoots*, wird zunächst entschieden, ob der Wert der Hüllkurve größer ist als der *Threshold*-Wert (beide Werte in Dezibel). Ist dies der Fall, wird der *Overshoot* os berechnet, indem der Hüllkurven-Wert env mit dem *Threshold*-Wert thr subtrahiert wird (Cipriani & Giri, 2019). Daraus ergibt sich für $(env > thr)$:

$$os(n) = env(n) - thr$$

Formel 2.2 (nach Cipriani und Giri, 2019, S. 423)

[4] RMS: „Root Mean Square“. Effektivwert, quadratischer Mittelwert

Berechnung der Gain Reduction

Aus dem *Overshoot*-Signal und der *Ratio* wird die *Gain Reduction* errechnet. Da die *Ratio* die prozentuale Absenkung des *Overshoots* angibt, berechnet sich die *Gain Reduction* at aus der Differenz von dem *Overshoot* und dem Quotienten von *Overshoot* und *Ratio* rt (Izhaki, 2008). Daraus ergibt sich die Formel:

$$at = os - \frac{os}{rt}$$

Formel 2.3 (Cipriani & Giri, 2019, S. 424)

Statische Gain Reduction

Mit einer Invertierung der bisher berechneten *Gain Reduction*, entsteht bei der Multiplikation von Steuersignal und Hauptsignal eine statische *Gain Reduction*. Diese ist ausschließlich vom Pegel des Steuersignals und der *Ratio* abhängig und lässt sich deshalb mit einer einfachen Kennlinie darstellen (siehe Abbildung 2.6) (Weinzierl, 2008).

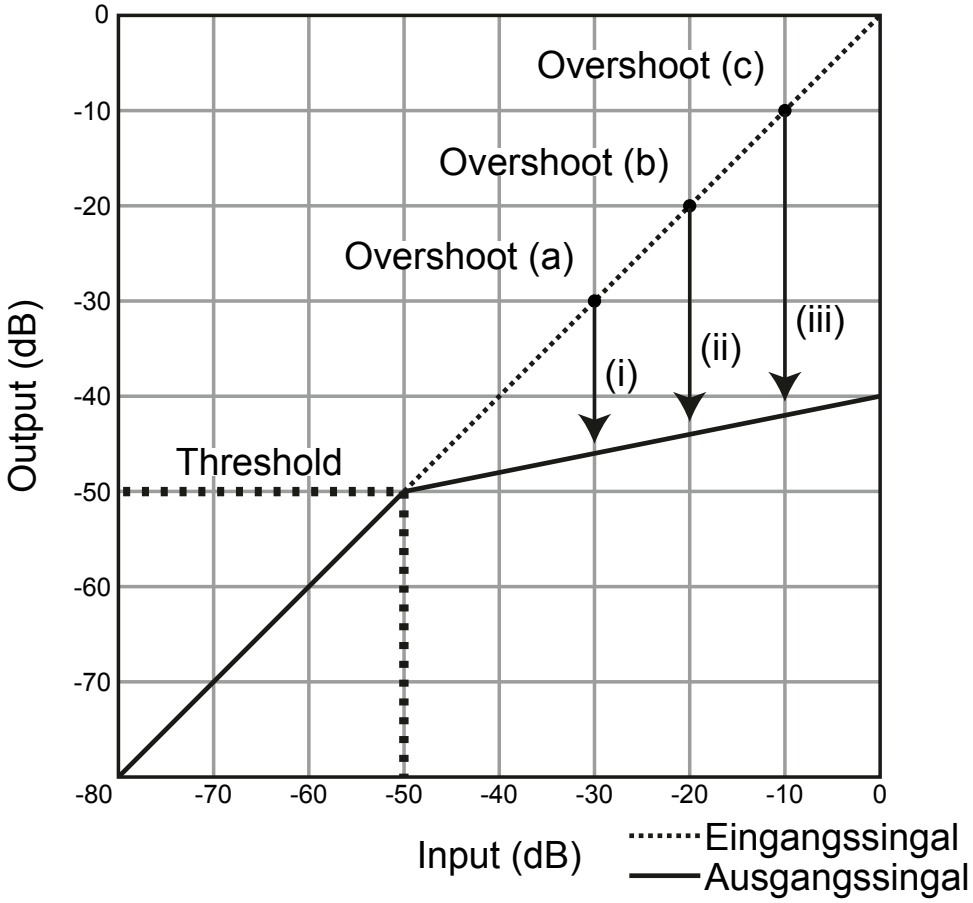

Abbildung 2.6: Kennlinie der statischen Kompression.

Threshold: -50 dB; Ratio 5:1;

Overshoot (a) 20 dB ergibt eine *Gain Reduction* (i) von –16 dB; Ausgangspegel –46 dB

Overshoot (b) 30 dB ergibt eine *Gain Reduction* (ii) von –24 dB; Ausgangspegel –44 dB

Overshoot (c) 40 dB ergibt eine *Gain Reduction* (iii) von –32 dB; Ausgangspegel –42 dB

Diese Kennlinie beschreibt die Abhängigkeit von Eingangs- und Ausgangssignal im Zusammenspiel mit der *Ratio*. Bei einer Abwärts-Kompression wird bei einem höheren Eingangspegel der *Overshoot* größer und damit auch die *Gain Reduction* (siehe Abbildung 2.6) (Dickreiter et al., 2014).

Dynamische Gain Reduction

Die Kennlinie in Abbildung 2.6 stellt die Kompression mit einer Reaktionszeit von 0 Millisekunden dar. In der Theorie bedeutet das, dass sobald das Steuersignal den *Threshold* überschreitet und es zu einer Pegelveränderung kommt, wird diese mit dem Ausgangssignal multipliziert (Dickreiter et al., 2014). Obwohl eine so kurze Reaktionszeit mit digitalen Regelverstärkern praktisch umsetzbar wäre, liefert es bei musikalischer Kompression in der Regel nicht das gewünschte Ergebnis. Oftmals sollen die charakteristischen Einschwingphasen der Instrumente weniger stark oder sogar unbearbeitet bleiben, während der Rest des Signals komprimiert wird. Zudem ist das Zeitverhalten charakteristisch für verschiedene Kompressoren. Um dieses Verhalten bei einem Kompressor umsetzen zu können, gibt es zeitabhängige Parameter, die das Verhalten zu Beginn und zum Ende einer Pegelveränderung separat beeinflussen (Izhaki, 2008). Diese Kompression in zeitlicher Abhängigkeit ist die *dynamische Gain Reduction* (Weinzierl, 2008).

Das Verhalten zu Beginn der Pegeländerung des *Overshoots* kann mit dem Parameter *Attack* beeinflusst werden. Das Verhalten zum Ende der Pegeländerung mit dem Parameter

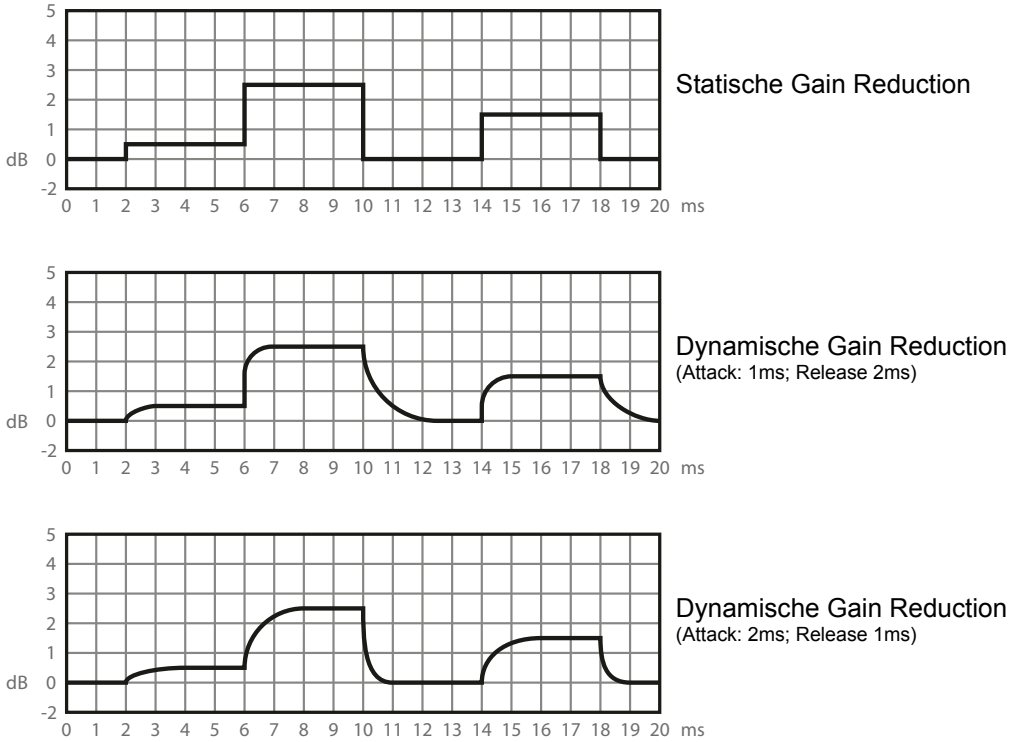

Abbildung 2.7: Zeitlicher Verlauf von statischer und dynamischer *Gain Reduction*. Quelle: In Anlehnung an Izhaki, 2008, S. 275

Release (Dickreiter et al., 2014). Die Interpolation zwischen den unterschiedlichen Pegeln erfolgt oftmals exponentiell (siehe Abbildung 2.7).

Im Gegensatz zu einer linearen Interpolation verfärbt die exponentielle Interpolation das Signal weniger und klingt dadurch natürlicher (Izhaki, 2008).

Die *Attack-Zeit* Δt_{att} drückt die Dauer aus, die der Kompressor benötigt, um 63 Prozent der *Gain Reduction* zu erreichen. Zeitpunkt t_1 entspricht dabei einer Pegelveränderung im *Overshoot*. Überschreitet der Pegel zu t_1 das erste Mal den *Threshold*, so gibt Δt_{att} die Zeit an, bis der Kompressor zu 63 Prozent aktiv ist. Bei Zeitpunkt t_2 sind 100 Prozent *Gain Reduction* erreicht (siehe Abbildung 2.8) (Weinzierl, 2008).

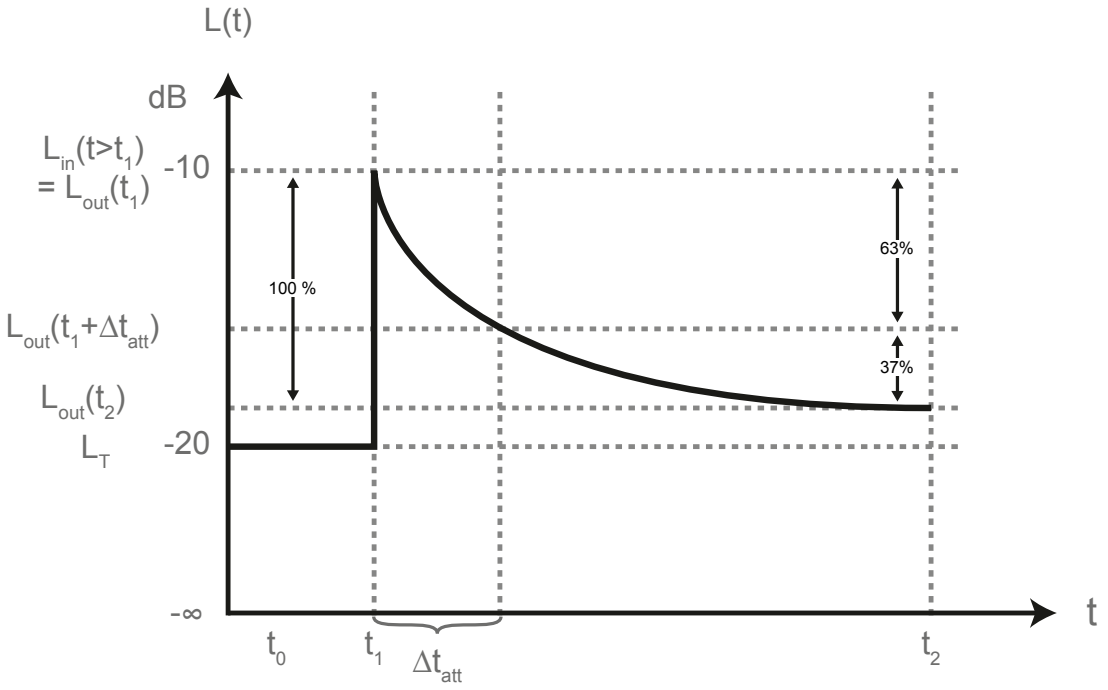

Abbildung 2.8: Attack-Verhalten bei dynamischer *Gain Reduction*. Quelle: In Anlehnung an Weinzierl. 2008, S. 733

Unabhängig von der *Attack*-Zeit gibt die *Release-Zeit* Δt_{rel} die Zeit an, die nach der *Gain Reduction* vergeht, bis 63 Prozent der Differenz zwischen dem letzten Pegel und dem nächsten Pegel erreicht sind. Unterschreitet das Steuersignal zu Zeitpunkt t_1 den *Threshold*, so gibt die *Release*-Zeit die Zeitspanne an, bis der Kompressor 63 Prozent der Differenz zwischen dem letzten Pegel oberhalb und dem ersten Pegel unterhalb des *Thresholds* erreicht hat (siehe Abbildung 2.9) (Weinzierl, 2008).

Zwar werden diese zeitabhängigen Parameter erst aktiviert, sobald das Steuersignal den *Threshold* überschreitet, doch verändern sie nicht nur das Zeitverhalten beim Überschreiten und Unterschreiten des *Thresholds*, sondern bei jeder Pegeländerung des *Overshoots*, also oberhalb des *Thresholds* (siehe Abbildung 2.7) (Izhaki, 2008).

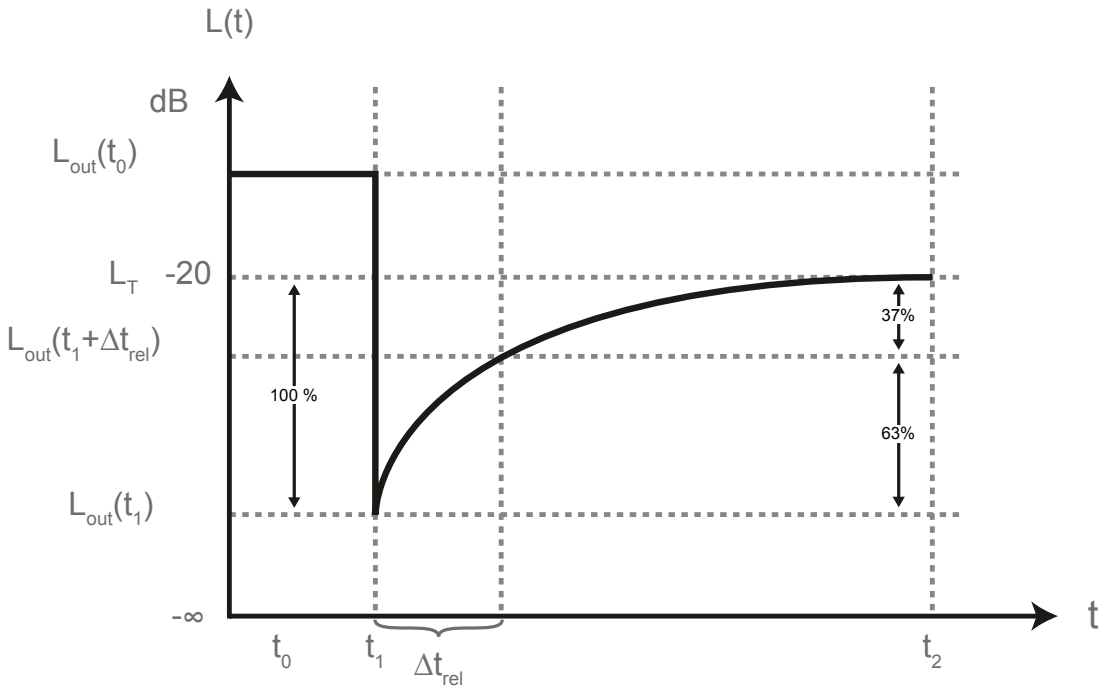

Abbildung 2.9: *Release*-Verhalten bei dynamischer *Gain Reduction*. Quelle: In Anlehnung an Weinzierl, 2008, S. 733

Invertieren

Nachdem die *Gain Reduction* mit den zeitabhängigen Parametern angepasst wurde, liegt das Steuersignal als modifizierte Hüllkurve vor. Diese besteht ausschließlich aus positiven Werten und muss vor der Multiplizierung mit dem Hauptsignal invertiert werden. Aus dem Vorzeichenwechsel ergibt sich eine umgedrehte Hüllkurve (siehe Abbildung 2.5 f) (Cipriani & Giri, 2019).

Make-up-Gain

Der letzte Schritt vor der Multiplikation mit dem Hauptsignal ist das *Make-up-Gain* oder *Offset*. Dieser Parameter ist notwendig, um den Verlust der Pegelspitzen durch die Kompression wieder auszugleichen und damit die wahrgenommene Lautheit zu erhöhen. Dazu wird das vollständige Signal, unabhängig ob es oberhalb oder unterhalb des *Thresholds* liegt, einheitlich verstärkt (siehe Abbildung 2.10) (Izhaki, 2008).

Delay/Lookahead

Das am Anfang gesplittete Eingangssignal, das bei der Multiplikation wieder zusammengeführt wird, durchläuft im *Side-Chain* Bearbeitungsschritte, die zu Verzögerungen führen können. Demgegenüber durchläuft das Hauptsignal bis zur Multiplikation mit dem Steuersignal keine weitere Bearbeitungsprozesse. Dadurch trifft das Steuersignal bei der Multiplikation später ein als das Hauptsignal, sodass Letzteres die *Gain Reduction* erst nach dem *Peak*, der eine *Gain Reduction* ausgelöst hat, erfährt, selbst wenn die *Attack*-Zeit

0 Millisekunden wäre. Um dem entgegenzuwirken, kann das Hauptsignal verzögert werden. Das *Delay* des Hauptsignals entspricht dabei der Zeit, um die das Steuersignal durch die Bearbeitung verzögert wird, damit beide Signale mit der gleichen Verzögerung multipliziert werden. Dadurch ist die vollständige Kontrolle der Transienten über die *Attack*-Zeit einstellbar (Izhaki, 2008).

Wenn kurze *Attack*-Zeiten zu unmusikalisch klingen und trotzdem die Transienten komprimiert werden sollen, kann die *Delay*-Funktion als *Lookahead* genutzt werden. Dabei wird das Hauptsignal nicht nur um die Zeit verzögert, die das Steuersignal zum Durchlaufen der Bearbeitungsprozesse benötigt, sondern darüber hinaus, so dass das Steuersignal vor dem Hauptsignal den Multiplikator erreicht. Dadurch erfährt das Hauptsignal eine *Gain Reduction* noch vor der auslösenden Pegelveränderung. Das ermöglicht den Einsatz längerer *Attack*-Zeiten, um Transienten vollständig abfangen zu können (Dickreiter et al., 2014).

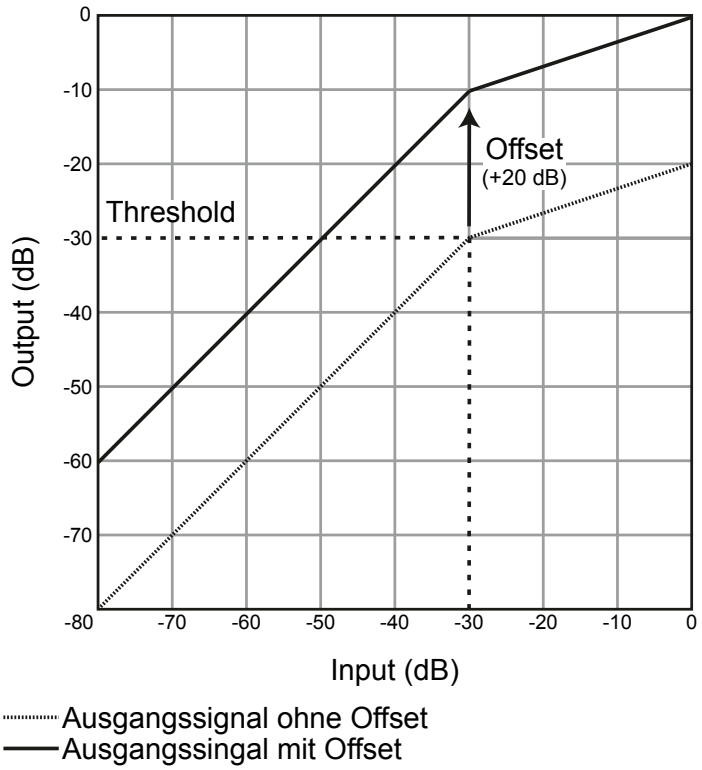

Abbildung 2.10: Kennlinie: Make-up-Gain. Quelle: In Anlehnung an Cipriani und Giri, 2019, S. 377

3 3D-Audio

Die Fähigkeit dreidimensional zu hören, ist für den Menschen und von essenzieller Bedeutung. Durch die Ortung von Schallquellen und Reflektionen, die uns tagtäglich einhüllen, verschaffen wir uns einen Überblick unserer Umgebung. In der Tontechnik war das Einfangen und Reproduzieren der realen Schallwelt aufgrund eingeschränkter Aufnahme- und Wiedergabetechniken seit jeher eine Herausforderung. Obwohl den räumlichen Dimensionen in der Tontechnik schon seit über einem Jahrhundert Aufmerksamkeit gewidmet wird, konnten erst im letzten halben Jahrhundert Fortschritte gemacht werden, die zu zufriedenstellenden Ergebnissen führen (Rumsey, 2013).

Mit *3D-Audio* oder *Immersive Sound* kann ein Klangerlebnis geschaffen werden, dass den Zuhörern/-innen eine neue Realität eröffnet. Dabei sollen die Hörer/-innen nicht nur einen Einblick in diese Realität bekommen, sondern das Gefühl haben, dort zu sein (Roginska & Geluso, 2017). Auf der einen Seite bringt es einen neuen kreativen Freiraum, der weit über die Grenzen von Stereo hinausgeht (Braddock et al., 2021), auf der anderen Seite kann es den Konsum, die Interaktion und den Umgang mit Musik revolutionieren (Roginska & Geluso, 2017).

Mit dem seit den 1950er Jahren vorherrschendem Stereo-Standard konnte, dem Stand der Technik entsprechend, eine Illusion erzeugt werden, die bis heute die Hörgewohnheiten der Konsumenten/-innen prägt. Obwohl ein guter Stereo-Mix eine beeindruckende Tiefenstaffelung haben kann, muss das Gehirn durch Interpretation der akustischen Ereignisse für die Wahrnehmung räumlichen Dimension sorgen. Das steht im Gegensatz zu 3D-Audio. Dort entsteht die räumliche Dimension nicht erst im Kopf des/der Zuhörers/-in, sondern bereits im Wiedergabesystem und kann direkt ohne Interpretation als solche wahrgenommen werden. Dadurch ist das Erlebnis der Konsumenten/-innen oftmals intensiver (Braddock et al., 2021).

Der Schritt von Stereo zu 3D-Audio kann mit der Erweiterung von Mono zu Stereo verglichen werden. Dementsprechend ist 3D-Audio nur eine Erweiterung des bisher dominierenden Formats (Lawrence, 2019), die jedoch nach Braddock et al. (2021) gegenüber Stereo eine größerer Verbesserung ist, als es Stereo gegenüber Mono war.

Aufgrund der Vervielfachung der möglichen Audio-Kanäle bzw. der flächenmäßigen Vergrößerung des Klangbilds ist bei der Produktion für 3D-Audio deutlich weniger Klangbearbeitung des Materials notwendig, als es bei Stereo der Fall ist. Auch das ist ein Grund dafür, dass eine gute 3D-Produktion natürlicher und realistischer klingen kann (Braddock et al., 2021).

Dass bei 3D-Audio eine Klangumgebung geschaffen werden soll, die die Konsumenten/-innen umgibt, ist ein offensichtliches Ziel. Jedoch gibt es unterschiedliche Ansätze wie

dieses Ziel erreicht werden kann (Roginska & Geluso, 2017). Dabei spielt die Erwartungshaltung der Hörenden eine entscheidende Rolle (Lawrence, 2019). Gerade bei klassischer Musik ist ein konservativer Mix zielführend, der die natürliche Umgebung bestmöglich wiedergibt. Für die Zuhörer/-innen soll die Illusion entstehen, dass sie sich direkt im Konzertsaal befinden. Die Reproduktion einer realen Klangumgebung ist nur bei entsprechenden Aufnahmen, wie eben bei einem klassischen Orchesterkonzert möglichen. Andernfalls ist eine anderer Mixansatz notwendig, der deutlich abstrakter werden kann und statt eines natürlichen Klangbilds eher eine erweiterte Realität erzeugt. Dies ist vor allem für kommerzielle, nicht linear produzierte Musikstile ein sinnvoller Ansatz (Rumsey, 2013). Je abstrakter die Musik, desto abstrakter darf auch der Einsatz der dreidimensionalen Umgebung sein (Braddock et al., 2021). Nichtsdestotrotz sollte eine 3D-Produktion die Musik in ihrer Aussage unterstützen und den Hörer überzeugen und nicht bloß als Demonstration der Möglichkeiten dienen (Lawrence, 2019).

3.1 Geschichte von 3D-Audio

Obwohl 3D-Audio wie eine Erfindung des 21. Jahrhunderts scheint, zeigt die Geschichte, dass die Idee alles andere als neu ist. Vielmehr ist die Umsetzung der benötigten Abspielsysteme eine Errungenschaft des letzten Jahrzehnts (Boren, 2017).

Nachdem Steinberg und Snow schon in den 1930er-Jahren bei großen Kinosälen eine Verbesserung der Stereo-Wiedergabesysteme durch einen weiteren Kanal zwischen dem rechten und dem linken Lautsprecher, dem *Center*, erzielen konnten (Rumsey, 2013), gab es mit der Uraufführung des Zeichentrickfilms *Fantasia* 1940 den ersten Film mit Mehrkanalton. Dabei wurden drei Audiokanäle und zehn Lautsprecher eingesetzt. Einer der drei Audiokanäle war für den Center bestimmt, die anderen beiden Kanäle konnten über die restlichen Lautsprecher verteilt werden. Neun der Lautsprecher umgaben das Publikum auf der horizontalen Ebene, während ein zusätzlicher Lautsprecher an der Decke hing (siehe Abbildung 3.1). Anfangs konnten die Audiokanäle nur über eine manuelle Überblendung zwischen den Lautsprechern bewegt werden. Mithilfe eines aufgezeichneten Steuersignals konnte das *Panning* später automatisiert werden (Boren, 2017; Torick, 1998).

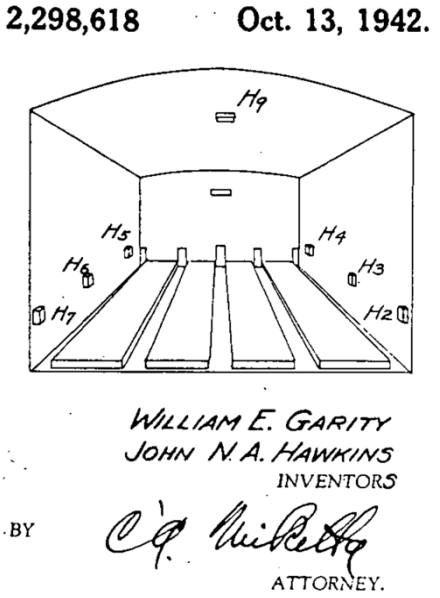

Abbildung 3.1: Patent von Garrity und Hawkins. Quelle: Torick (1998, S. 29)

Zu dieser Zeit wurde die Entwicklung hauptsächlich von der Filmindustrie vorangetrieben, die unter anderem mit einem verbesserten Audiosystem mehr Publikum in die Säle locken wollten (Boren, 2017; Rumsey, 2013).

Erste kommerzielle Surroundsysteme für die Heimanwendung wurden in den 1970er-Jahren entwickelt. Das quadrofonische Wiedergabesystem konnte vier Audiokanäle über vier auf der Horizontalebene im Quadrat angeordnete Lautsprecher wiedergeben. Obwohl einige Produktionen das neue Format in einer Studioumgebung erfolgreich anwenden konnten, war die Reproduktion bei den Konsumenten/-innen schwierig (Braddock et al., 2021). Das lag zum einen an einer großen Anzahl an verschiedenen Codierungsalgorithmen die unterschiedliche Wiedergabeeinstellungen benötigten und zum anderen an den Voraussetzungen für das Wiedergabesystem. Die rechtwinklige Anordnung der Lautsprecher war nicht mit einer herkömmlichen Stereopositionierung der Lautsprecher, die üblicherweise in einem 60-Grad-Winkel zueinanderstehen, kompatibel, sodass die Lautsprecher des quadrofonischen Systems nicht für eine standardmäßige Stereowiedergabe verwendet werden konnten (Rumsey, 2013). Durch den daraus resultierenden kommerziellen Misserfolg wurde das quadrofonische Wiedergabeformat bereits Ende der 70er-Jahre nicht weiter verfolgt (Braddock et al., 2021). Die bei der Quadrofonie verwendeten Verfahren zur Matrizierung wurden jedoch weiterhin für audiovisuellen *Content* z. B. im Rahmen von Dolby Stereo oder Dolby Surround verwendet (Weinzierl, 2008).

Ambisonics, ein weiteres Format für Surround Sound, das eine dreidimensionale Wiedergabe ermöglicht, wurde ebenfalls in den 70er-Jahren entwickelt. Dieses Format basiert auf einem anderen Ansatz als die kanal-basierte Quadrofonie. Statt einer kanalabhängigen wird eine richtungsabhängige Reproduktion, die unabhängig von der Kanalanzahl ist, verfolgt. Dadurch kann das Material für unterschiedliche Lautsprecheranordnungen angepasst und wiedergegeben werden. Trotz der großen Flexibilität der benötigten Wiedergabesysteme konnte auch *Ambisonics* nicht kommerziell etabliert werden und wurde durch die Kombination verschiedener Ansätze insbesondere für Forschungszwecke verwendet (Rumsey, 2013).

Anfang der 1980er-Jahre folgte auf die Quadrofonie das erste Dolby Surround Format, das auch aufgrund von erschwinglichen Endgeräten für die Heimnutzung an Relevanz gewann. Zu diesem Zeitpunkt bot Dolby Surround drei Audiokanäle, mit denen Signale für ein Stereopärchen und Mono-Surround-Lautsprecher vorhanden waren. Später, gegen Ende der 80er-Jahre, enthielt das Nachfolgeformat Dolby ProLogic bereits vier Audiokanäle. Richtig etablieren konnte sich der Mehrkanalton erst durch die DVD Mitte der 1990er-Jahre (Weinzierl, 2008), als das Thema Heimkino für die Konsumenten/-innen zu einer bezahlbaren Vorstellung wurde und die Veränderung oder Einschränkung des Mobiliar dem Zweck der Unterhaltung angemessen schien (Rumsey, 2013).

Zu diesem Zeitpunkt wurden auch die ersten Musik-Produktionen für ein 5.1 Surround System erstellt, die zunächst auf DVD-Video den Weg auf die Heimanlagen finden mussten (Braddock et al., 2021).

In den folgenden Jahren wurde die Entwicklung des Überträgermediums fortgesetzt und von *Super Audio Compact Disc* (SACD) bis zur DVD-Audio wurde die Anzahl der Audiokanäle, der Dynamik- und Frequenzumfang, die Spielzeit und die Audiocodierung erweitert und verbessert (Braddock et al., 2021). Demzufolge mussten die Mehrkanal-Spuren nicht mehr matriziert werden, sondern konnte als diskrete Kanäle gespeichert und übertragen werden (Rumsey, 2013). Dennoch konnten sich weder SACD noch DVD-Audio erfolgreich durchsetzen (Braddock et al., 2021).

Zum Zeitpunkt der Einführung der Blu-Ray Disc 2006 waren in Deutschland, Österreich und der Schweiz ein gutes Drittel der Haushalte mit einem mehrkanalfähigen Wiedergabesystem ausgestattet (Braddock et al., 2021; Weinzierl, 2008). Blu-Ray war den vorangegangen Überträgermedien technisch überlegen und konnte mehr als sechs unkomprimierte Audiokanäle speichern (Braddock et al., 2021).

Die Entwicklung führte in der Folge zu einem einfacheren Produktionsvorgang für Mehrkanalaudio, da die aufwendigen Matrizierungsalgorithmen nicht mehr beachtet werden mussten, weshalb Anfang der 2010er der Fokus auf die Erweiterung vom zwei-dimensionalen Surround auf drei-dimensionalen *Immersive Sound* gelegt wurde und neue Formate entstanden (Braddock et al., 2021; Oramus & Neubauer, 2019). Diese neuen Formate erweiterten die horizontale Ebene nicht nur um eine weitere vertikale Ebene, sondern beinhalteten viele Verbesserungen gegenüber den vorangegangen Surround Formaten. Diese Verbesserungen sollten vor allem die Schwierigkeiten und Verwirrungen auf der Seite der Konsumenten/-innen beheben (Braddock et al., 2021).

Mittlerweile ersetzt 3D-Audio das herkömmliche Stereo nicht nur in der Filmindustrie, sondern auch immer häufiger bei Übertragungen und Musikproduktionen (Braddock et al., 2021).

3.2 Aktualität von 3D-Musik

Da Surround Sound und mittlerweile auch Immersive Sound seit bereits über einem Jahrzehnt als Standard in Kinosälen und ebenso in der Gaming-Industrie etabliert sind, besteht bereits eine Infrastruktur für Formate wie Dolby Atmos, DTS-X und Auro-3D/AuroMax, die innerhalb der Musikindustrie zunehmend an Relevanz gewinnen (Braddock et al., 2021; Lawrence, 2019). Die objekt-basierte Infrastruktur, auf dem die neuen immersiven Formate basieren, bietet nicht nur eine bessere Anpassung an aktuelle Wiedergabesysteme, sondern ist auch zukunftssicher (Tsingos, 2017). Damit einher geht eine unter

Wissenschaftlern/-innen, Forschern/-innen und Musik-Produzenten/-innen verbreitete Ansicht, dass 3D-Audio ein großer Schritt Richtung hochqualitativer Audio-Reproduktion ist (Lawrence, 2019). Spätestens seit der Implementierung von 3D-Audio bei Musikstreaming-Diensten wie Amazon Music (Sony Electronics Inc., 2021), Apple Music (Apple Inc., 2021a), Deezer und Tidal (Inc., 2020) ist 3D-Musik auch für normale Konsumenten/-innen in ein greifbares Thema geworden.

Innerhalb der Musikindustrie wird das Potenzial von Immersive Sound in Kombination mit objekt-basiertem Audio durch eine neue Art von Produktion und ein neues Verständnis von Raum erkannt und geschätzt, was sich vor allem darin äußert, dass immer mehr Tonstudios mindestens eine immersive Mix-Regie zur Verfügung haben. Gerade weil sich 3D-Musik für den kommerziellen Konsum noch in einem Anfangsstadium befindet, sind viele innovative und kreative Herangehensweisen notwendig, um ein Format, das vor allem für audiovisuellen *Content* ausgelegt ist, ausschließlich für Audio zu verwenden. Dadurch bietet sich in vielerlei Hinsicht eine besondere Möglichkeit für Musiker/-innen und Musikproduzenten/-innen, die Grenzen der bisherigen Stereo-Produktion zu überschreiten und die neue Klangumgebung kreativ zu nutzen (Lawrence, 2019).

3.3 Formate

Über den langen Zeitraum der Entwicklung für 3D-Audio sind mehrere unterschiedliche Formate entstanden, die in drei Kategorien aufgeteilt werden können: kanal-basiert, szenen-basiert und objekt-basiert. Während kanal-basiertes Audio auf dem gleichen Wiedergabesystem abgespielt werden muss, auf dem es produziert wurde, sind szenen-basiertes und objekt-basiertes Audio frei von dieser Einschränkung. Bei diesen beiden Formaten kann das Audiomaterial auf das Abspielgerät der Konsumenten/-innen angepasst werden (Braddock et al., 2021). Dies wird vor allem durch die Prozessorleistungen auf den Abspielgeräten ermöglicht, da szenen-basiertes und objekt-basiertes Audio ein vielfaches an einzelnen Audiokanälen im Vergleich zu kanal-basiertem Audio haben und aus diesen Audiokanälen erst auf dem finalen Abspielgerät die Signale für die vorhandenen Lautsprecher berechnet werden (Weinzierl, 2008).

Für eine bestmögliche Wiedergabe sind bei allen Formaten im besten Fall *Full-Range*-Lautsprecher, also Lautsprecher, die das gesamte Frequenzspektrum wiedergeben können, notwendig. Dadurch kann ein ausgewogenes Klangbild im gesamten Raum gewährleistet werden (Weinzierl, 2008).

Mehrkanal-Formate besitzen neben den Audiokanälen zusätzliche Metadaten, die für eine korrekte Wiedergabe sorgen sollen. Diese Metadaten beinhalten unterschiedliche Informationen von der Beschreibung des Datenstroms über den erforderlichen

Dynamikbereich bis hin zum Downmix-Verhalten (Weinzierl, 2008). Objekt-basierte Formate benötigen zudem Positionsdaten der einzelnen Audiokanäle, um diese, wie bei der Produktion festgelegt, wiedergeben zu können (Tsingos, 2017).

Einige Lautsprecher Formate verwenden sowohl kanal-basiertes als auch objekt-basiertes Audio, um das Beste aus beiden Formaten nutzen zu können (Lawrence, 2019).

Oramus und Neubauer (2019) konnten in ihrer Studie zur Wahrnehmung von kanal-basierten und objekt-basierten Mehrkanal-Formaten kein bevorzugtes Format ermitteln, sondern kamen zu dem Ergebnis, dass die beiden Formate unterschiedliche Klangerlebnisse erzeugen, die vom inhaltlichen Material abhängig sind. Auch in einer weiteren Studie zur räumlichen Lokalisation bei kanal-basiertem und objekt-basiertem Audio stellten Oramus und Neubauer (2020) fest, dass objekt-basiertes Audio gegenüber einem Surroundsystem mit vier Surroudlaustsprechern (7.1) keinen signifikanten Vorteil hat und es folglich keine genauere Ortung bietet.

Diese Erkenntnis eröffnet den Produzenten die Möglichkeit, das Format je nach Bedarf zu wählen, ohne dabei räumliche Präzision im Zielformat einbüßen zu müssen (Lawrence, 2019).

3.3.1 Kanal-basiertes 3D-Audio

Kanal-basierte oder auch diskrete Systeme zeichnen sich dadurch aus, dass bei der Wiedergabe eine Reihe von unabhängigen Signalen einer vordefinierten Anzahl und Positionierung von Lautsprechern zugeordnet ist. Die einfachsten Beispiele dafür sind Stereo-Systeme, die aus zwei Audiokanälen bestehen, wovon einer für den linken Lautsprecher und einer für den rechten Lautsprecher gedacht ist (Kim, 2017).

Bei kanal-basierten 3D-Formaten ist das Wiedergabesystem und die Lautsprecher Anordnung herstellerabhängig standardisiert. Obwohl die beiden gängigen Formate Auro-3D 13.1 und NHK 22.2 jeweils drei Ebenen haben, sind ihre Lautsprecher Anordnungen nur bedingt kompatibel (Braddock et al., 2021).

Die drei Lautsprecherebenen bei NHK 22.2 bestehen aus einer mittleren Ebene mit zehn Lautsprechern, einer oberen Ebene mit neun Lautsprechern und einer unteren Ebene mit drei Lautsprechern und 2 *LFEs*[5] (siehe Abbildung 3.2). Die Lautsprecher der mittleren Ebene befinden sich auf Ohrhöhe des Publikums und besteht aus fünf Front-Lautsprechern und 5 Surround-Lautsprechern. Die obere Lautsprecher-Ebene ist von der Hörposition idealerweise 45 Grad oberhalb der mittleren Ebene. Dabei bilden acht Lautsprecher, drei Front-Lautsprecher und 5 Surround-Lautsprecher, einen Kreis um die Zuhörer/-innen. Der

[5] LFE: „Low Frequency Effect“. Bei Kinofilmen nur für tieffrequente Effekte eingesetzt

neunte Lautsprecher kommt direkt von oben und weist damit einen Winkel von 90 Grad zu der mittleren Ebene auf. Die untere Ebene besteht ausschließlich aus Front-Lautsprechern zwischen denen die *LFEs* positioniert sind (Hamasaki et al., 2005).

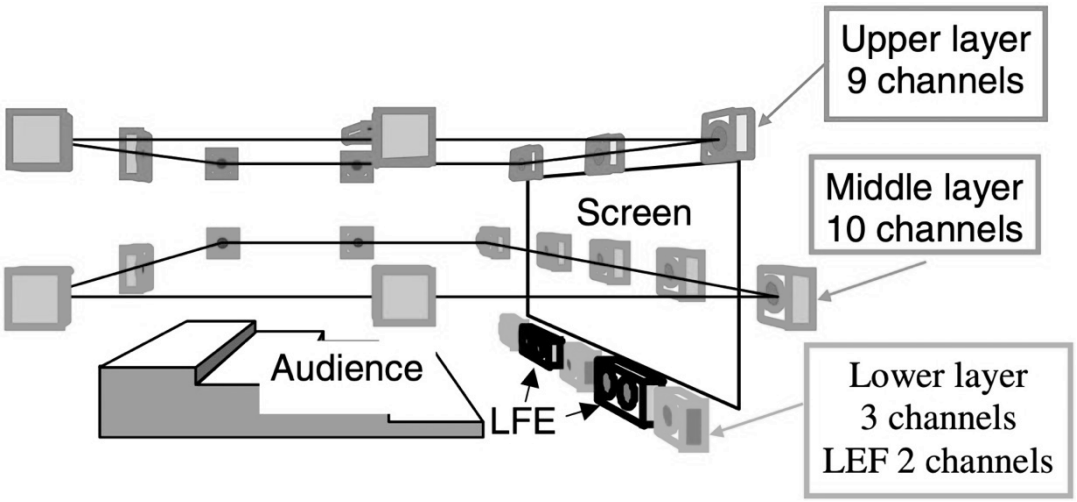

Abbildung 3.2: Lautsprecheranordnung NHK 22.2. Quelle: Hamasaki et al. (2005, S. 2)

Auro-3D 13.1 besitzt im Gegensatz zu NHK 22.2 keine untere Ebene. Die Ebenen befinden sich auf Ohrhöhe, bei 30 Grad und bei 90 Grad. Die Lautsprecher der unteren Ebene sind im 7.1-Standard angeordnet, wobei die Surround-Lautsprecher 110 Grad und 150 Grad zum Center aufweisen. Die mittlere Ebene ist besteht aus fünf Lautsprechern, die genau oberhalb der Lautsprecher der unteren Ebene positioniert sind: drei über den Front-Lautsprechern und zwei über den vorderen (110 Grad) Surround-Lautsprechern. Die obere Ebene besteht aus einem einzigen Lautsprecher, dem sogenannten *Voice-Of-God*, der sich ähnlich wie bei NHK 22.2 mittig über der Abhörposition befindet (siehe Abbildung 3.3) (Auro Technologies, 2015).

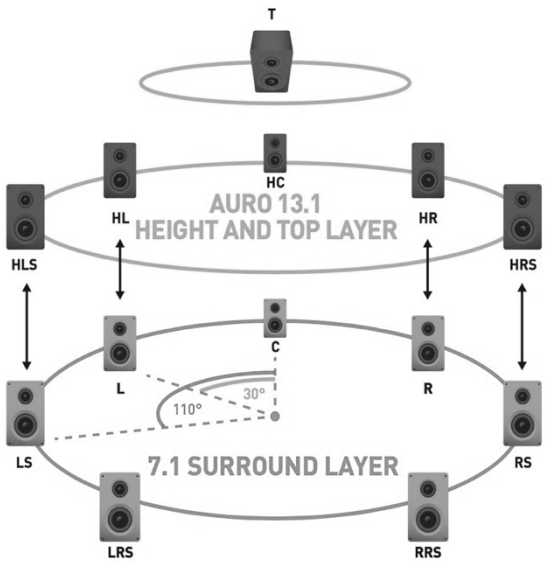

Abbildung 3.3: Lautsprecheranordnung Auro-3D 13.1. Quelle: Auro Technologies (2015, S. 14)

Durch die festgelegten Lautsprecher Positionen und die Bindung an das jeweilige Abspielsystem kann das Audio-Material für diese diskreten Systeme als einfach Mehrkanalsignal gespeichert werden. Die einzelnen Signale entsprechen den Lautsprechersignalen, die das Format benötigt (Friesecke, 2014). Eine Audio-Datei für NHK 22.2 besteht folglich aus 24 Audiokanälen.

Obwohl kanal-basierte Formate für eine bestimmte Lautsprecher-Konfiguration vorgesehen sind, ist es möglich davon abzuweichen. Für diesen Upmix oder Downmix werden verschiedene Algorithmen angewendet, die material- und systemabhängig sein können (Rumsey, 2013).

Upmix-Algorithmen werden insbesondere bei Stereo-Material verwendet, das auf einem größeren Lautsprecher-System wiedergegeben werden soll. Die beiden gängigsten Ansätze bestehen darin, aus dem vorhandenen Signal Informationen für die zusätzlichen Lautsprecher zu extrahieren oder Effekte zu erzeugen, die verschiedene Räume simulieren sollen, zwischen denen die Nutzer/-innen wählen können (Rumsey, 2013). Das Ergebnis eines Upmixes von Stereo für Mehrkanal-Formate ist für den geübten Hörer allerdings eher ernüchternd. Zwar wird der räumliche Eindruck verbessert, aber das ursprünglich vorhandene Stereobild leidet unter der automatisch erzeugten Erweiterung (Rumsey, 1999).

Downmix-Algorithmen beruhen auf Regeln, die entweder von den in den Metadaten vorhandenen Downmix-Koeffizienten, einem Standard oder dem Wiedergabegerät abhängig sind. Diese Regeln legen fest, welche Kanäle wie zusammengefasst werden. Dabei werden den Kanälen bestimmte Werten zum Zusammenmischen gegeben. Surround-Kanäle, die auf die Front-Lautsprecher summiert werden, erfahren eine andere Absenkung als der Center-Lautsprecher (International Telecommunication Union, 2012). Durch dieses Zusammenfassen von eigentlich unabhängigen Kanälen können Phasenprobleme entstehen, die zu einer ungewollten Veränderung des Audiomaterials führen und das Hörerlebnis negativ beeinflussen (Zielinski et al., 2003).

3.3.2 Objekt-basiertes 3D-Audio

Wie schon erwähnt, ist das Positionieren von Audiosignalen innerhalb eines dreidimensionalen Lautsprechersystems bereits in der ersten Hälfte des 20. Jahrhunderts umgesetzt worden. Zuerst manuell in Echtzeit, dann automatisiert (Torick, 1998). Bei kanal-basierten Formaten findet die Positionierung während der Produktion statt, sodass beim finalen Mix die Audiosignale entsprechend ihrer Positionen auf diskreten Lautsprechersignalen zusammengefasst werden. Danach können diese Lautsprechersignale ohne einen weiteren Bearbeitungsprozess auf einem passenden Lautsprechersystem wiedergegeben werden (Tsingos, 2017). Objekt-basiertes Audio wird ebenfalls bei der Produktion positioniert,

jedoch werden die Signale beim finalen Mix nicht zu Lautsprechersignalen zusammengefasst. Das Rendern der Lautsprechersignale geschieht auf dem jeweiligen Endgerät (Braddock et al., 2021). Die einzelnen Audiosignalsignale werden keinem Lautsprecherkanal mehr zugeordnet, sondern nur noch Positionen im Raum (Friesecke, 2014). Dadurch ist der finale Mix losgelöst von der Produktionsumgebung und kann optimal an die Abhörumgebung angepasst werden. Das führt bei korrekt kalibrierten Abhörsystem zu einem personalisierbaren, immersiveren Klangerlebnis gegenüber kanal-basierten Formaten. Allerdings erfordert diese Technik komplexere Verfahren in der Bearbeitung, Codierung und Übertragung (Tsingos, 2017).

Eine objekt-basierte Audiodatei benötigt zur Interpretation der einzelnen Audiokanäle Metadaten. Diese beschreiben unter anderem Lautstärke und die Position im Raum (Braddock et al., 2021). Audiosignale, die in Kombination mit Positionsdaten auftreten, werden als Audio-Objekte bezeichnet. Diese bestehen aus Mono-Signalen. Innerhalb einer DAW können sie beliebig bearbeitet werden, bevor sie über einen *Panner*[6] die gewünschte räumliche Position zugewiesen bekommen. Diese Position wird in den Metadaten festgehalten und kann sich im Verlauf des Songs verändern (Tsingos, 2017).

3.3.3 Metadaten

Spätestens seit den digitalen Dolby Mehrkanal-Formaten sind Metadaten, die die Audiosignale beschreiben, für eine korrekte Reproduktion notwendig. Diese ermöglichen sowohl eine technische Anpassung an das Abspielgerät als auch eine Personalisierung durch die Nutzer/-innen (Weinzierl, 2008). Die European Broadcast Union (EBU) hat für eine breitere Kompatibilität den *Audio Definition Model* Standard, kurz *ADM*, entwickelt (EBU, 2014). Da dieser Standard heute in vielen objekt-basierten 3D-Audioformaten angewendet wird, soll er im Folgenden erläutert werden:

Das ADM besteht aus zwei Teilen: dem *Content*- und dem Format-Teil. Letztere beinhaltet die nötigen technischen Informationen, die für die Decodierung und das Rendering benötigt werden. Der *Content*-Teil beschreibt die Eigenschaften der Audiosignale. Hier kann beispielsweise die Sprache definiert oder die Lautheit gemessen werden. Das Model verwendet für die Zuordnung der Kanäle und den Audiodateien das *Broadcast Wave Format* (BWF). Dafür enthält eine BWF-Datei eine Liste der vorhandenen Kanäle und die korrespondierenden Audio-Dateien sowie die Beschreibung des Audiokanal-Formats (z. B. Mono, Stereo, 5.1) (EBU, 2014). Abbildung 3.4 erläutert die verschiedenen Komponenten des *Audio Definition Models* und stellt ihre Beziehungen untereinander dar.

[6] Panner: In diesem Fall eine grafische Oberfläche zur Festlegung der Position von Audiosignalen in der Audioszene

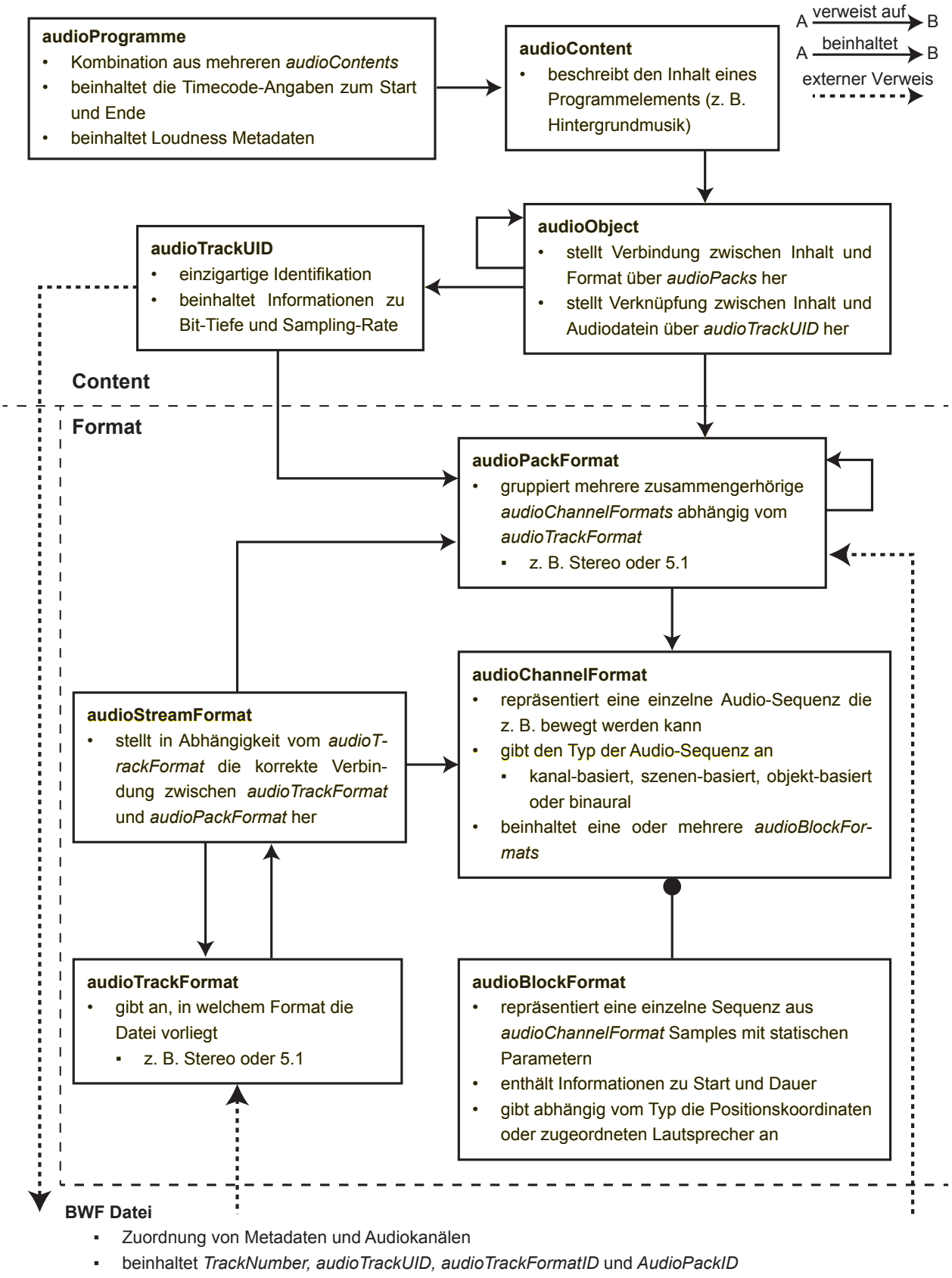

Abbildung 3.4: Audio Definition Model. Verbindungen der Komponente. Quelle: In Anlehnung an EBU, 2014, S. 9

Für die räumliche Positionierung einzelner Audio-Objekte bei einer dreidimensionalen Audio-Produktion sind die Positionskoordinaten notwendig. Diese Koordinaten werden z. B. von einer DAW geschrieben und können von einem Renderer interpretiert werden. Damit kann ein Audiosignal dem Abspielgerät entsprechend an einer beliebigen Stelle im Raum positioniert werden.

Die verschiedenen Möglichkeiten zur Festlegung einer Position im Raum können in zwei Kategorien eingeteilt werden: egozentrisch und allozentrisch. Diese Kategorien beziehen sich auf die Wahrnehmung. Wird zur Positionierung im Raum ein egozentrischer Ansatz gewählt, wird das Audio-Objekt relativ zur Position der/des Hörers/-in positioniert. Dazu bieten sich Polarkoordinaten an. Bei einem allozentrischen Ansatz werden die Audio-Objekte in einem definierten Raum unabhängig von den Hörern/-innen positioniert. Dieser Ansatz verwendet in der Regel kartesische Koordinaten (Tsingos, 2017).

Beim Rendering kann durch die verschiedenen Koordinatensysteme, abhängig von dem Renderer ggf. eine Konvertierung notwendigen sein. (Tsingos). Für eine große Kompatibilität hat das EBU mit dem *Audio Definition Model* einen offenen Standard für Metadaten entwickelt (EBU, 2014).

Rendering

Das auf das Endgerät optimierten Rendering ist bei objekt-basierten Formaten unumgänglich. Damit bietet es gegenüber kanal-basierten Formaten einen großen Vorteil, da das Abhörsystem der Konsument/-innen normalerweise nicht bekannt ist. Durch das angepasste Rendering wird die Wiedergabe immer bestmöglich reproduziert (Braddock et al., 2021).

Beim Rendering werden Audiosignale der Audio-Objekte auf die während der Produktion beabsichtigte Position im Raum gepannt. Dazu werden, abhängig von den vorhandenen Lautsprechern und dessen Position, Lautsprechersignale generiert. Bewegen sich Audio-Objekte, wird die Lautstärke der Audiosignalen auf den verwendeten Lautsprechern über einen Zeitbereich von wenigen Frames interpoliert (Tsingos, 2017).

Für die Berechnung der einzelnen Lautsprechersignale gibt es unterschiedliche Algorithmen, die bereits beim kanal-basierten Mehrkanal zum Einsatz kamen (Rumsey, 2013; Tsingos, 2017). Ein häufig verwendeter Algorithmus ist der *Amplitude Panning Algorithmus*. Das Lautsprechersignal $x_i(t)$ wird mit dem *Gain*-Faktor g_i, der Anzahl der Lautsprecher $i = 1, \dots, \mathrm{N}$ und dem Audiosignals $x(t)$ berechnet (Pulkki, 2001; Thomas & Robinson, 2017):

$$x_i(t) = g_i x(t), \qquad i = 1, \dots, N$$

Formel 3.1 (Pulkki, 2001, S. 11)

Dabei wird von einem normalisierten *Gain*-Faktor ausgegangen, sodass sich daraus die Formel 3.2 ergibt.

$$\sum_{i=1}^{N} g_i^2 = 1$$

Formel 3.2 (nach Thomas und Robinson, 2017, S. 2)

Der *Vector-Based Amplitude Panning* Algorithmus (VBAP) basiert auf dem *Amplitude Panning* und nutzt für die Positionierung der Audio-Objekte zwei bis drei Lautsprecher (Pulkki, 2001). Bei diesem Algorithmus kann nicht zwischen Objekten, die auf unterschiedlichen Position in der selben Richtung liegen, unterschieden werden, da nur die Richtung des Audio-Objekts im Verhältnis zu einer Referenzposition genutzt wird (Tsingos, 2017).

Ein weiterer Algorithmus, der *Position-Based Panning*-Algorithmus, bezieht sich ausschließlich auf die Position der Audio-Objekte und nicht auf deren Richtung. Dazu werden drei eindimensionale *Gain*-Werte gebildet, die den Achsen eines dreidimensionalen kartesischen Koordinatensystems x, y, z und damit den Richtungen links–rechts, vorne–hinten, oben–unten entsprechen. Das Produkt dieser Werte ergibt den *Gain* für das Lautsprechersignal G_i (Tsingos, 2017):

$$G_i(x, y, z) = Gx_i(x) \times Gy_i(y) \times Gz_i(z)$$

Formel 3.3 (Tsingos, 2017, S. 250)

Dieser *Panning*-Algorithmus erlaubt kontinuierliche Bewegungen durch den gesamten dreidimensionalen Raum. Dazu wird die Anzahl der verwendeten Lautsprecher nicht begrenzt. Auf der einen Seite führt das zu einem weicheren *Panning*, auf der anderen Seite zu klanglichen Artefakten (Tsingos, 2017).

Bei dreidimensionalen *Panning*-Algorithmen müssen Kompromisse zwischen den klanglichen Veränderungen des Audiomaterials und der Eigenschaften des *Pannings*, wie z. B. der Stabilität an der Abhörposition oder der Lokalisationsschärfe, eingegangen werden (Tsingos, 2017).

Durch das Verschieben des Renderingvorgangs vom Produktionsgerät auf das Abspielgerät kann der Renderer die benötigten Lautsprechersignale berechnen. Idealerweise können somit alle verfügbaren Lautsprecher unabhängig von ihrer Position genutzt werden, unter der Voraussetzung, dass der Renderer von ihnen weiß und ihre Position kennt. Rendering auf dem Endgerät sorgt für eine größere Flexibilität und Skalierung von dreidimensionalen Multikanal-Formaten, sodass diese auf großen Kinosystemen mit 64 Lautsprechern ebenso wiedergegeben werden können, wie auf einem kleinen 5.1 Heimkinosystem oder binaural auf Kopfhörern. Dabei kommt es nicht zu einem Downmix wie bei kanal-basiertem Audio, beim dem Lautsprechersignale summiert werden, sondern zu einer Neuberechnung jedes einzelnen Lautsprechersignals. Das verhindert z. B., dass Phantomschallquellen zwischen zwei Lautsprechern bei der Summierung der Lautsprechersignale Artefakte wie Phasenprobleme produzieren (Tsingos, 2017).

4 Objekt-basiertes Mastering

Mastering für immersive Audioformate findet unter der gleichen Prämisse statt wie Mastering für Stereo. Audioprozessoren zur Frequenz- und Dynamikbearbeitung kommen hier genauso zum Einsatz wie bei Stereo-Produktionen. Der Umfang der Bearbeitung ist meistens vergleichsweise geringer. Während bei der Produktion für Stereo auf eine Monokompatibilität geachtet wird, wird bei 3D-Audio auf die Kompatibilität mit herkömmlichen Heimkinosystem wie 5.1 und Stereo geachtet (Braddock et al., 2021).

Wie in Kapitel 3.3 erläutert, gibt es verschiedene Formate für 3D-Audio. Beim Mastering für kanal-basiertes 3D-Audio können die jeweiligen Lautsprechersignale wie beim Stereo bearbeitet werden. Da die Summierung der einzelnen Signale bereits bei der Produktion stattgefunden hat, kann der gesamte Klang des Lautsprechersignals bearbeitet werden. Bei objekt-basierten 3D-Audio werden die Lautsprechersignale erst auf dem Abspielgerät berechnet. In diesem Fall ist es nicht möglich, die Lautsprechersignale im Mastering zu bearbeiten.

Melchior et al. (2011) stellen eine Möglichkeit vor, um die Positionsdaten zur Steuerung von Parametern der Audioprozessoren zu verwenden. Hestermann et al. (2018) schlagen die Verwendung von Mastering-Objekten vor. Diese können mit Audio-Objekten verknüpft werden und Bereiche und Winkel festlegen in denen Audio-Objekte bearbeitet werden.

Beide Ansätze zeigen, dass metadatenabhängige Audio-Bearbeitung beim Mastering für objekt-basierte Formate realisierbar ist. Dazu wird für jedes Audio-Objekt eine Audioprozessor-Instanz benötigt, die das Audiosignal individuell verändert. Die Parameter werden entweder von einem metadatenabhängigen Steuerparameter (Melchior et al., 2011) oder von den Einstellungen des Mastering-Objekts gesteuert (Hestermann et al., 2018). Das ermöglicht eine einheitliche Bearbeitung statischer Audioprozessoren, für die betreffenden Audio-Objekte. Bei statische Audioprozessoren hat das Audiosignal keinen Einfluss auf das Verhalten des Audioprozessors. Ein Beispiel für einen statischen Audioprozessor ist der Equalizer.

In Abbildung 4.1 wird eine Taxonomie zu objekt-basiertem Mastering präsentiert. Hier wird die Bearbeitung nach einem finalen Mixdown in zwei Kategorien geteilt: *Metadaten-Bearbeitung* und *metadatenabhängige Bearbeitung*.

Bei der *Metadaten-Bearbeitung* werden die Metadaten ausgelesen und verändert. Das kann inhaltliche Metadaten betreffen, die z. B. ein Audio-Objekt als *Sprache* beschreiben, oder formale Metadaten wie die *Gain*- oder Positionsdaten. Durch die Veränderung der inhaltlichen Metadaten kann das Wiedergabeverhalten und die Interaktion der Konsumenten/-innen mit dem Programm verändert werden.

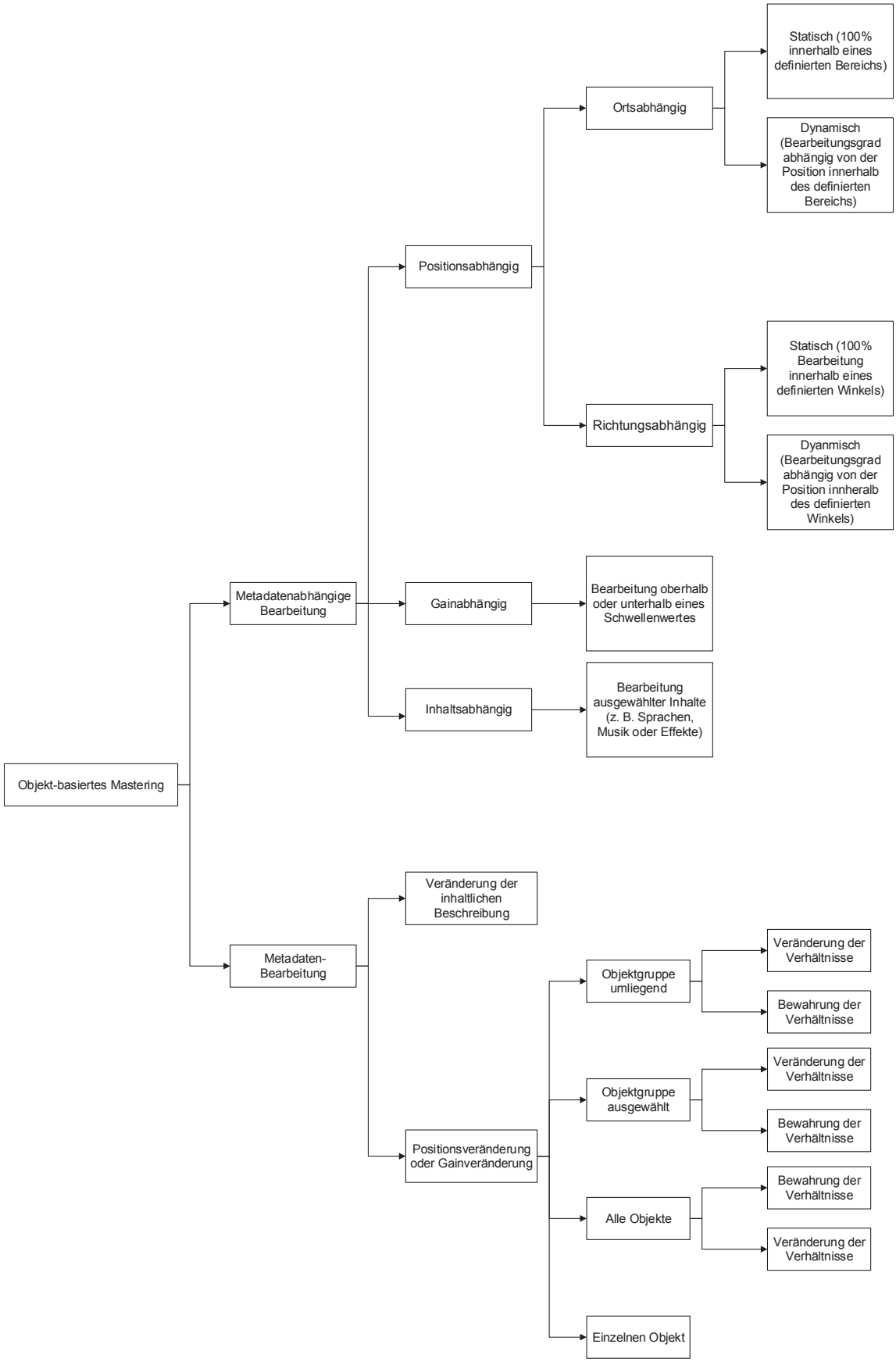

Abbildung 4.1: Taxonomie: objekt-basiertes Mastering

Die Manipulation der Positions- oder *Gain*-Informationen können wie von Hestermann et al. (2018) vorgeschlagen auf mehrere Audio-Objekte gleichzeitig angewandt werden. Dabei kann das Verhältnis zwischen den zu bearbeitenden Audio-Objekten bewahrt werden, indem die Veränderung in relativen Werten umgesetzt wird. Durch eine Veränderung mit absoluten Werten kann das Verhältnis geändert werden. Mit der relativen Bearbeitung kann z. B. eine ganze Szene gedreht werden. Mit der absoluten Bearbeitung können alle Audio-Objekte zu einem Punkt bewegt werden.

Die *metadatenabhängige Bearbeitung* nutzt die Metadaten wie von Melchior et al. (2011) vorgeschlagen zur Steuerung der Audioprozessoren. Bei dieser Bearbeitung bleiben die Metadaten unverändert. Die Bearbeitung kann von verschiedenen Metadaten, wie z. B. den inhaltlichen oder den formalen abhängig sein. Es ist möglich, inhaltlich ausgewählte Audio-Objekte zu bearbeiten, oder in Abhängigkeit ihrer Position oder ihres *Gains* Bearbeitungen zu automatisieren. In einem Beispiel können alle Audio-Objekte, die als Sprache beschrieben werden, mit einer bestimmten Equalizer-Einstellung bearbeitet werden. In einem anderen Beispiel können alle Audio-Objekte, die einen bestimmten *Gain*-Wert haben, mit einem *Limiter* begrenzt werden.

Bei einer positionsabhängigen Bearbeitung kann zwischen zwei Positionsbestimmungen unterschieden werden. Bei der richtungsabhängigen Bearbeitung durchlaufen alle Audio-Objekte, die sich innerhalb einer Richtung befinden, die Audioprozessoren. Die Richtung wird über einen Winkel der Polarkoordinaten definiert. Ortsabhängige Bearbeitung betrifft alle Audio-Objekte, die sich in einem definierten Bereich befinden. Dieser Bereich ist entweder ein Radius um ein ausgewähltes Audio-Objekt oder ein definierbares Polygon.

Sowohl bei der richtungsabhängigen als auch bei der ortsabhängigen Bearbeitung gibt es ein statisches und ein dynamisches Verhalten. Während bei einem statischen Verhalten der Grad der Bearbeitung innerhalb eines Winkels oder eines Bereichs gleichbleibt, variiert dieser beim dynamischen Verhalten abhängig von der Position des Audio-Objektes. Je näher sich dieses am geometrischen Schwerpunkt der Polygone oder an der Winkelhalbierenden befinden, desto größer ist der Grad der Bearbeitung.

Richtungsabhängige Bearbeitung kann zum Beispiel für Korrekturen bezüglich falsch kalibrierter Lautsprecher in der Mix-Regie verwendet werden. Auf einem Lautsprecher, der weniger Höhen hat als die anderen Lautsprecher im Abhörsystem, werden beim Mixing ggf. die Höhen angehoben. In einer korrekt eingemessenen Mastering-Regie werden die Audio-Objekte aus der Richtung des falsch kalibrierten Lautsprechers zu viele Höhen haben. Für ein gleichmäßiges Klangbild kann dies mit einer richtungsabhängigen Bearbeitung korrigiert werden.

Ortsabhängige Bearbeitung findet unter anderem Verwendung, wenn sich mehrere Audio-Objekte innerhalb eines Bereichs klanglich überlagern. Durch die Verknüpfung des

Bereichs an ein Audio-Objekt können alle umliegenden Audio-Objekte bearbeitet werden. Sobald sich die überlagernden Instrumente annähern, werden ein oder gegebenenfalls mehrere Instrumente mit einem Equalizer bearbeitet. Das dynamische Verhalten kann dazu beitragen, dass die Bearbeitung stärker wird, je näher sich die Audio-Objekte sind. Dadurch wirkt Bearbeitung unsichtbarer.

5 Objekt-basierter Kompressor

Während Funktionen von statischen Audioprozessoren über eine mehrfache Instanziierung und instanzübergreifenden Einstellung der Parameter verwendet werden können, stellt eine einheitliche dynamische Bearbeitung von mehreren Audio-Objekten eine Herausforderung dar.

In die Kategorie dynamische Audioprozessoren fallen alle Bearbeitungswerkzeuge, die abhängig vom Audiomaterial arbeiten. Typische dynamische Audioprozessoren sind dynamische Equalizer und Regelverstärker. Für eine einheitliche Dynamikbearbeitung von mehreren Regelverstärkerinstanzen zu erreichen, müssen diese von demselben *Key-Input* gesteuert werden. In einem kanal-basierten 7.1 Mix könnte eine *Bassdrum*, die ausschließlich auf dem *Center* positioniert ist mehr Durchsetzungsvermögen bekommen, indem das Lautsprechersignal des *Centers* als *Key-Input* die Kompressoren auf den restlichen sieben Lautsprechern steuern. Dadurch wird gleichzeitig auf allen Lautsprechern – der *Center* ausgenommen – ein Ducking-Effekt erzeugt. Dieser tritt immer dann auf, wenn das *Center*-Signal den *Threshold* überschreitet. Dieser Ansatz ist bei objekt-basierten Formaten nicht möglich.

5.1 Problemstellung

In populären Musikstilen ist die Kompression ein häufig verwendetes Stilmittel, das nicht nur auf einzelnen Spuren angewandt wird, sondern auch auf Subgruppen und Summen. Durch die Anwendung auf Summen wird ein gewünschter Effekt erzielt, der aus den einzelnen Signalen eine Einheit macht. Besonders bei modernen Produktionen, bei denen vermehrt Samples oder digital erzeugte Instrumente verwendet werden und die Aufnahme der Instrumente asynchron verläuft, ist dies ein bewährtes Werkzeug, um den einzelnen Songelementen einen Zusammenhalt zu geben. Während bei Stereo-Produktionen die Stereosumme einen optimalen Ansatzpunkt für solche einheitlichen Bearbeitungen bietet, gibt es bei objekt-basierten Produktionen bis zum Rendering auf den Endgeräten keine Summierung von Signalen. Die fehlende Summierung verhindert eine einheitliche Bearbeitung aller oder mehrerer Signale bei der Produktion.

Um beim objekt-basierten Mastering einen *Glue*-Effekt (siehe Kapitel 2.4) mit einem Kompressor zu erzeugen, muss für jede Kompressor-Instanz dasselbe Key-Input-Signal verwendet werden. Dieses ist die Summe aller vorhandenen Audiosignale. Dadurch erfolgt eine einheitliche Kompression aller Audio-Objekte.

3D-Audioformate bieten im Gegensatz zu Stereo-Formaten nicht nur *eine* Dimension, sondern *drei*. Deshalb kann es erstrebenswert sein, den *Glue*-Effekt nur in bestimmten

Richtungen oder Bereichen zu erzeugen. Während an diesen Stellen ein kompakterer Klang erzeugt wird, bleibt das restliche Klangbild offen.

Dieser Fall lässt sich mit herkömmlichen Audioprozessoren und einem *Key-Input*-Signal umsetzen. Die Kompression darf dabei nur auf die Audio-Objekte, die sich in dem gewünschten Bereich befinden, angewendet werden. Das *Key-Input*-Signal besteht ausschließlich aus der Summe dieser Audio-Objekte. Das setzt voraus, dass die räumlichen Positionen der DAW-Spuren bekannt sind oder die Audio-Objekte in der Rendering-Darstellung den Spuren in der DAW zugeordnet werden können. Dieser Workaround wird komplexer, sobald sich Audio-Objekte bewegen und sie definierten Bereiche betreten und wieder verlassen oder der Bearbeitungsgrad positionsabhängig sein soll. Dann müssen die Kompressoren und die *Sends* für den *Key-Input* für jedes Audio-Objekt einzeln angepasst und für bewegende Audio-Objekte automatisiert werden.

Der *Ducking*-Effekt, der von einem sich bewegenden Audio-Objekt ausgeht, ist ein weiteres Beispiel. Dieses kann mit den bisher verfügbaren Audioprozessoren nur mit viel Aufwand umgesetzt werden. Das Audio-Objekt könnte etwa die erste Gesangsstimme sein, die sich im Uhrzeigersinn um die Hörposition bewegt. An einer Position, z. B. im hinteren Bereich wird die Stimme von einem Klavier maskiert. An einer anderen Position, z. B. im seitlichen Bereich, wird sie von einer Gitarre maskiert. Im vorderen Bereich befinden sich Bass und Schlagzeug. Hier kann sich die Stimme gut durchsetzen. Um dieselbe Durchsetzungskraft der Stimme auch im seitlichen und hinteren Bereich zu erreichen, muss entweder die Stimme in den entsprechenden Bereichen lauter automatisiert werden oder die Instrumente leiser, sobald die Stimme in diese Bereiche kommt. Das ist ein aufwendiger Arbeitsschritt, der bei einer Stereo-Produktion durch einen Regelverstärker effektiv und bequem erledigt werden kann.

Die individuelle Dynamikbearbeitung von Audio-Objekten in Abhängigkeit ihrer Metadaten gestaltet den Mastering-Prozess effektiver. Hat ein sich bewegendendes Audio-Objekt einen hohen Spitzenpegelwert nahe $0\ dB$, kann es bei der Summierung für das Lautsprechersignal mit einem weiteren lauten Signal zu einem ungewollten *Limiting* seitens des Renderes kommen. Um das *Limiting* zu verhindern, werden die Audio-Objekte komprimiert, sobald sie zu einem Lautsprechersignal summiert werden. Sobald sich beide Audio-Objekte in einem bestimmten Bereich befinden, müssen die Kompressor-Parameter beider Audio-Objekte automatisiert werden.

Die Beispiele zeigen, dass Mastering von objekt-basierten Formaten auch jetzt schon möglich ist, aber durch die aufwendigen Einstellungen nicht effektiv ist. Durch das aufwendige Routing und Anpassen der Send-Levels kann der Fluss beim Arbeiten verloren gehen, sodass die Mastering-Ingenieure/-innen nicht direkt auf Probleme reagieren

können. Das kreative Mastering, wie es bei Stereo praktiziert wird, ist bei objekt-basiertem Audio nicht intuitiv und bietet der Kreativität weniger Freiraum.

Mit objekt-basierten dynamischen Audioprozessoren wird objekt-basiertes Mastering deutlich effektiver und schafft zeitgleich neue kreative Möglichkeiten. Dadurch bekommt das Mastering für objekt-basierte Audioformate mehr Aufmerksamkeit und wird wie bei Stereo ein fester Bestandteil einer professionellen Produktionskette. Schlussendlich verbessert sich die Qualität von objekt-basierten Produktionen.

5.2 Lösungsansatz

Ein effektives Mastering für objekt-basiertes Audio benötigt Audioprozessoren, die die Metadaten der Audio-Objekte mit einbeziehen. Damit können die Nutzer/-innen präzise Korrekturen diverser Fehler eines Mixes durchführen und effektiv kreative Eingriffe in einem Mastering-Prozess vornehmen. Im Folgenden wird ein Lösungsansatz für einen objekt-basierten Kompressor vorgestellt.

In diesem Lösungsansatz werden die Metadaten nicht verändert, sondern ausschließlich ausgelesen, um Einfluss auf die Audio-Prozessoren zu nehmen (siehe Abbildung 5.1). Der Lösungsansatz kann auf jede Art von Audiobearbeitung angewandt werden, wird im Folgenden aber nur für einen Abwärts-Kompressor erforscht. Er bezieht sich ausschließlich auf Audio-Objekte. Bei hybriden Formaten werden kanal-basierte und szenen-basierte Audiosignale nicht berücksichtigt.

In Abbildung 5.2 werden einige Funktionen eines objekt-basierten Regelverstärkers dargestellt. Die Funktionen bauen auf die positionsabhängige Bearbeitung aus Abbildung 4.1 auf.

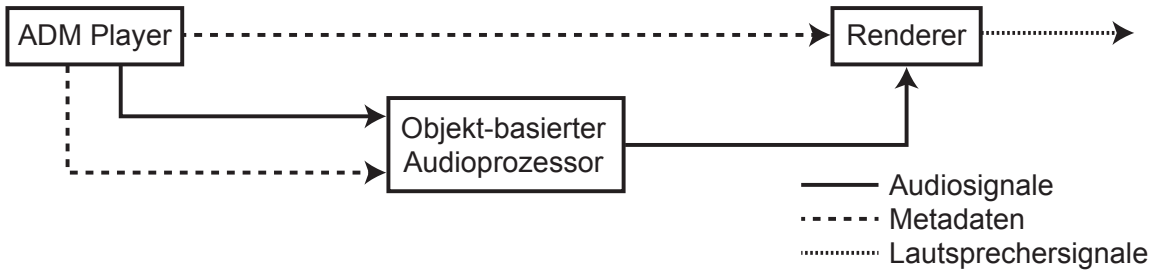

Abbildung 5.1: Blockdiagram metadatenabhäniger objekt-basierter Audioprozessor

5.2.1 Einzelbearbeitung

Die Einzelbearbeitung entspricht den Vorschlägen von Melchior et al. (2011) und Hestermann et al. (2018). Die einzeln vorliegenden Audiosignale der Audio-Objekte werden mit jeweils einer unabhängig agierenden Instanz eines Regelverstärkers manipuliert.

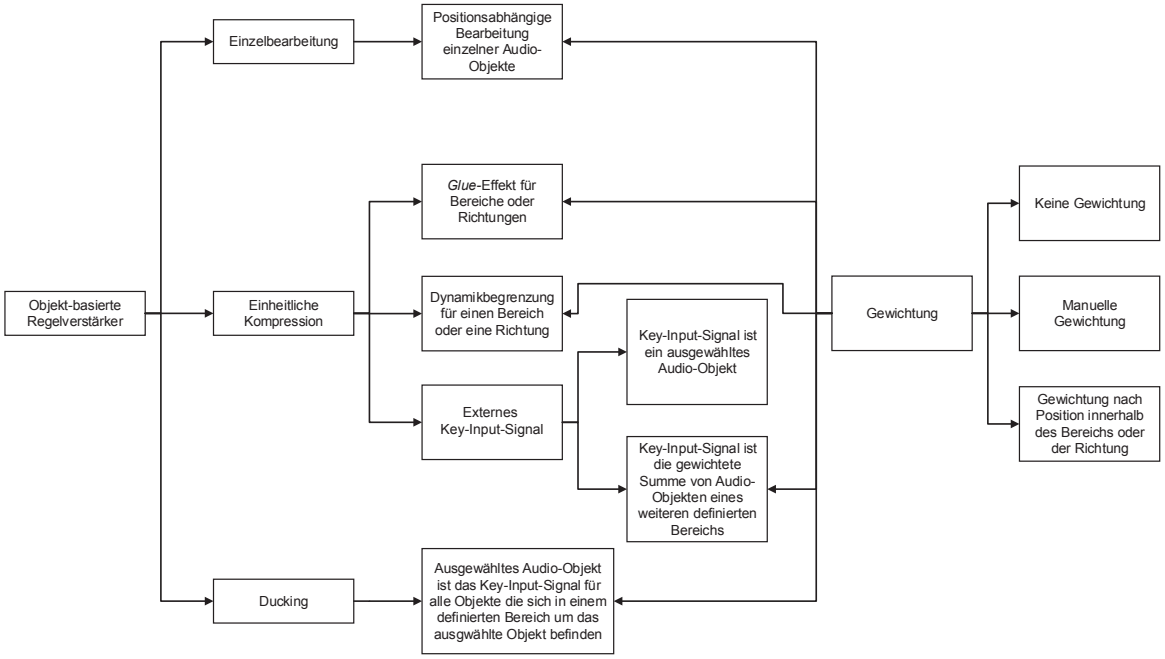

Abbildung 5.3: Ausgewählte Funktionen eines objekt-basierten Kompressors

Der Grad der Bearbeitung oder die Parametereinstellungen hängen dabei von der Position des Audio-Objekts ab (siehe Abbildung 5.3). Der Bearbeitungsgrad innerhalb des definierten Bereichs kann statisch oder wie in Kapitel 4 beschrieben positionsabhängig gewichtet werden. Die Gewichtung kann zum Beispiel das Verhältnis zwischen unbearbeitetem und bearbeitetem Signal steuern. Daraus resultiert eine Parallel-Kompression der Audio-Objekte innerhalb des definierten Bereichs. An dessen Rand besteht die Parallel-Kompression hauptsächlich aus unbearbeitetem Audiosignal. An dessen Mittelpunkt besteht sie hauptsächlich aus bearbeitetem Audiosignal. Alternativ können Parameter des Kompressors in Abhängigkeit der Position verändert werden.

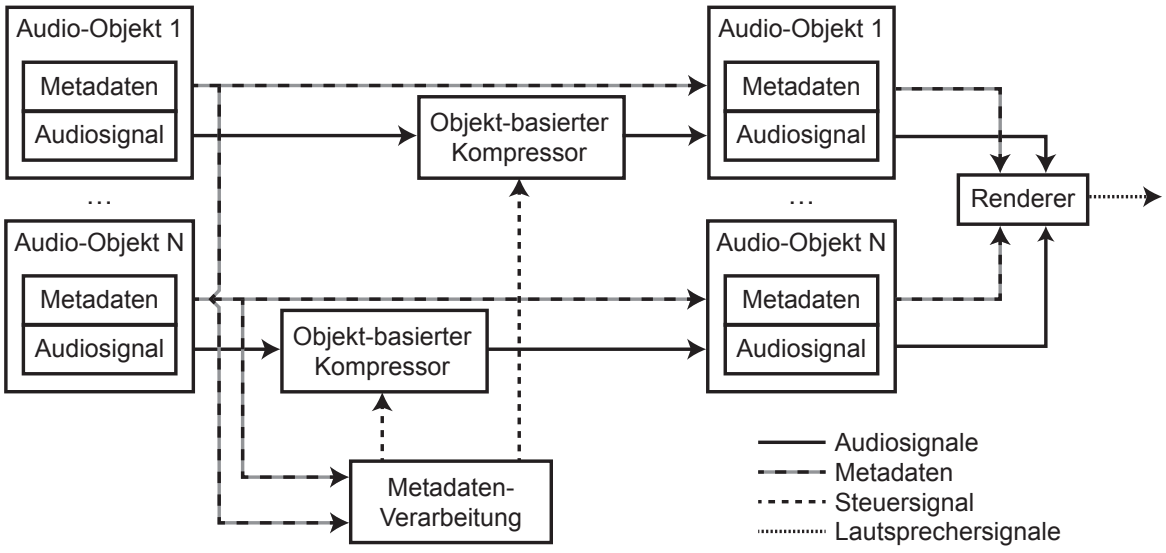

Abbildung 5.2: Signalfluss bei Einzelbearbeitung mit einem objekt-basierten Kompressor. Quelle: In Anlehnung an Melchior et al., 2011, S. 7

Zum Beispiel kann die *Ratio* zur Winkelhalbierenden größer werden oder der *Threshold* niedriger. Dieses dynamische Verhalten verhindert bei bewegten Audio-Objekten eine abrupte Veränderung des Audiosignals und ist deutlich unauffälliger.

5.2.2 Ducking

Abbildung 5.4 stellt den Signalfluss eines objekt-basierten Kompressors beim *Ducking*-Effekt dar. Beim *Ducking* wird ein Audio-Objekt ausgewählt, an das der Bearbeitungsbereich geknüpft wird. Der Bearbeitungsbereich entspricht einem Radius r um das Audio-Objekt und kann manuell eingestellt werden. Das Audiosignal des ausgewählten Audio-Objekts dient als *Key-Input*-Signal für die Kompressor-Instanzen der anderen Audio-Objekte. Wenn sich ein Audio-Objekt *B* in dem definierten Bereich um das auswählte Audio-Objekts *A* befindet, wird *B* mit dem *Key-Input*-Signal, bestehend aus dem Audiosignal von Audio-Objekt *A*, komprimiert. Immer wenn Audio-Objekt *A* den *Threshold* überschreitet, wird Audio-Objekt *B* komprimiert. Dieser Effekt wird auch als *Ducking* bezeichnet.

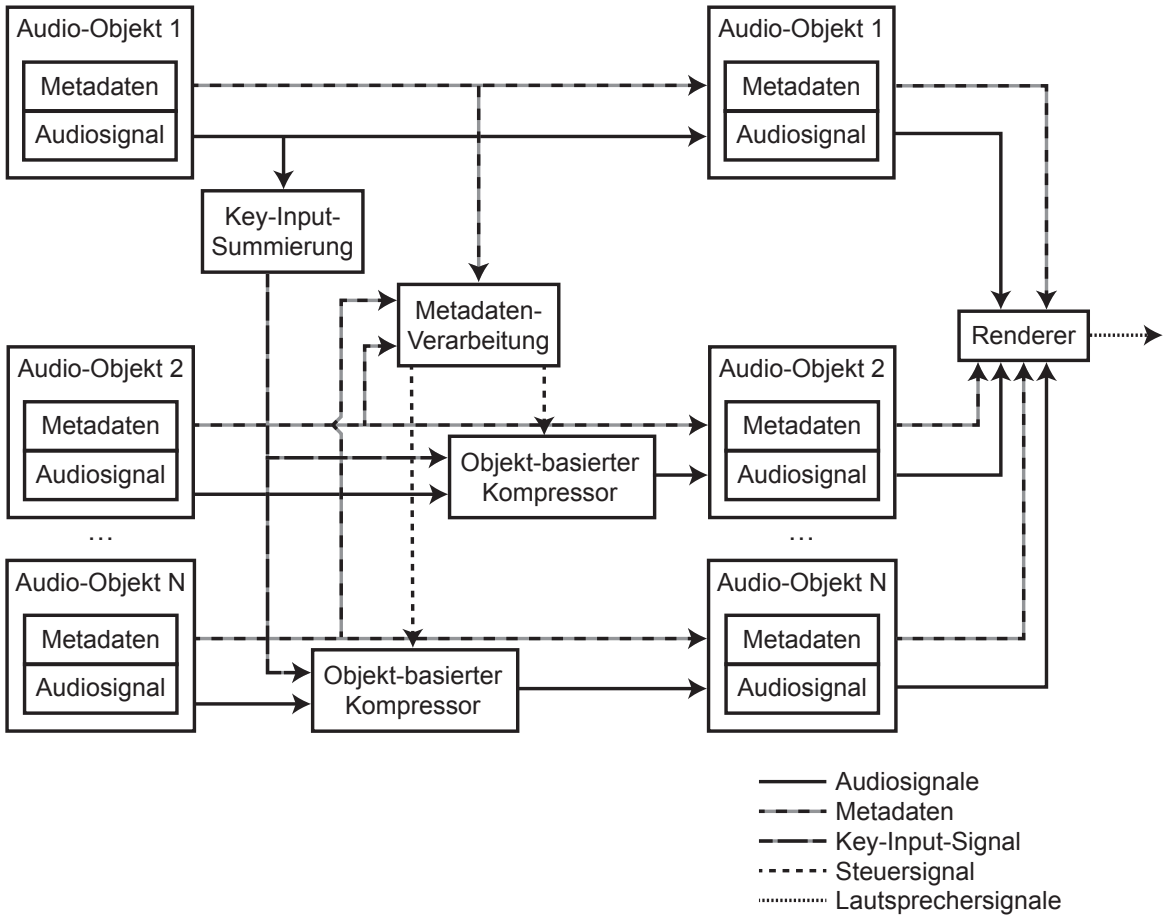

Abbildung 5.4: Signalfluss: *Ducking* mit objekt-basiertem Kompressor. Quelle: In Anlehnung an Melchior et al., 2011, S. 7

Auch beim *Ducking* gibt es statische und dynamische Gewichtung. Zusätzlich kann eine manuelle Gewichtung erfolgen. Diese wird von den Nutzern/-innen über einen dafür vorgesehen Parameter eingestellt. Der Gewichtungsfaktor ändert z. B. den Pegel des *Side-Chain*-Signals innerhalb der einzelnen Kompressor-Instanzen. Dadurch können Audio-Objekte unterschiedliche stark komprimiert werden.

5.2.3 Einheitliche Kompression

Im Gegensatz zu den zuvor beschriebenen Funktionen erfordert eine einheitliche Kompression mehrerer Audio-Objekte ein komplexeres Routing. Abbildung 5.5 stellt ein solches Routing dar. In diesem Fall wird das *Key-Input*-Signal aus den im definierten Bereich liegenden Audio-Objekten generiert. Die Audiosignale werden mit gleicher Gewichtung summiert. Das *Key-Input*-Signal wird nach der Summierung an die einzelnen Kompressor-Instanzen geroutet. Die aus den Metadaten gewonnenen Steuersignale steuern Kompressor-Parameter wie z. B. den Mix-Parameter, der das Verhältnis zwischen unbearbeitetem und bearbeitetem Signal kontrolliert. Auch in diesem Fall kann eine manuelle oder positionsabhängige Gewichtung angewendet werden.

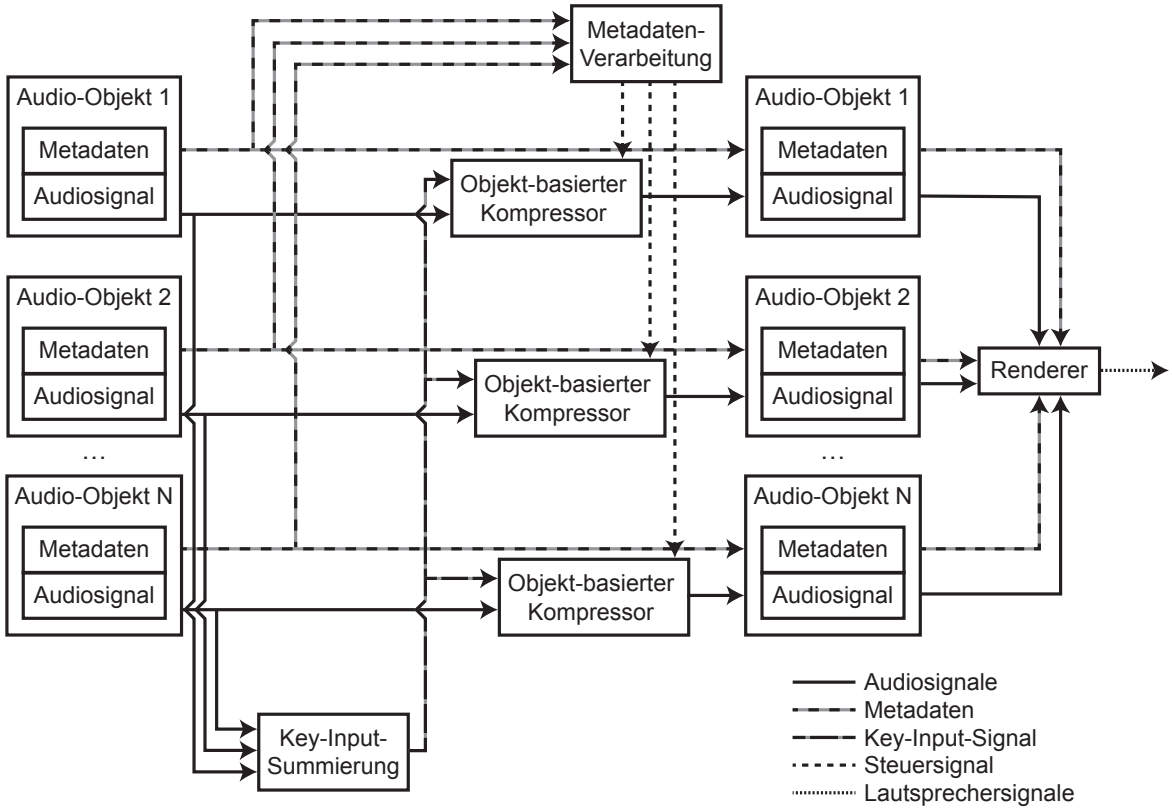

Abbildung 5.5: Signalfluss: Einheitliche Kompression mit einem objekt-basiertem Kompressor. Quelle: In Anlehnung an Melchior et al., 2011, S. 7

Die Metadaten können neben der Gewichtung der Parameter auch die Gewichtung der Audiosignale im *Key-Input*-Signal bestimmen (siehe Abbildung 5.6). Die Audiosignale werden dafür vor der *Key-Input*-Summierung mit dem aus den Positionsdaten generierten Steuersignal multipliziert. Das Ergebnis ist eine dynamische Veränderung des *Key-Input*-Signals. Das Audiosignal, das am nächsten zum Mittelpunkt des definierten Bereichs ist, beeinflusst die Kompression am stärksten, da es den größten Anteil des *Key-Input*-Signals ausmacht. Dies ermöglicht eine natürliche Kompression eines Bereichs, die sich nicht abrupt verändert, sobald ein sich bewegendes Audio-Objekt mit hohem Pegel in den definierten Bereich ein- oder austritt.

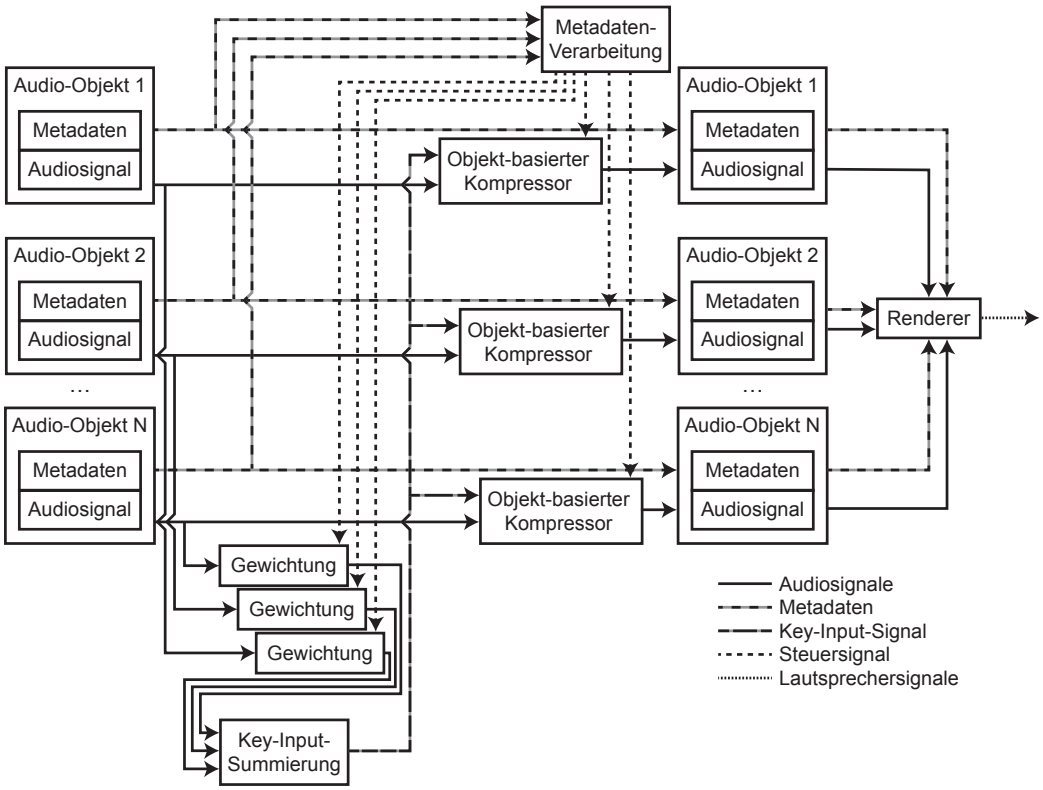

Abbildung 5.6: Signalfluss: Einheitliche Kompression mit objekt-basiertem Kompressor und Key Input Gewichtung. Quelle: In Anlehnung an Melchior et al., 2011, S. 7

Die einheitliche Kompression erweitert die beschriebene Funktion um ein externes *Key-Input*-Signal. Das externe *Key-Input*-Signal besteht nicht aus den Audiosignalen der Audio-Objekte innerhalb des definierten Bereichs, sondern aus externen Audio-Objekten. Ein zweiter definierbarer Bereich bestimmt, welche Audio-Objekte zu einem *Key-Input*-Signal summiert werden. Der Signalfluss ist in Abbildung 5.7 dargestellt.

Die Gewichtung der *Key-Input*-Signale kann statisch oder dynamisch sein. Eine dynamische Gewichtung ist abhängig von der Position der Audio-Objekte innerhalb des für das *Key-Input*-Signal definierten Bereichs. Die Positionsdaten der Audio-Objekte innerhalb des Bereichs, auf den die Kompression angewendet wird, steuern die Parameter der Kompressor-Instanzen.

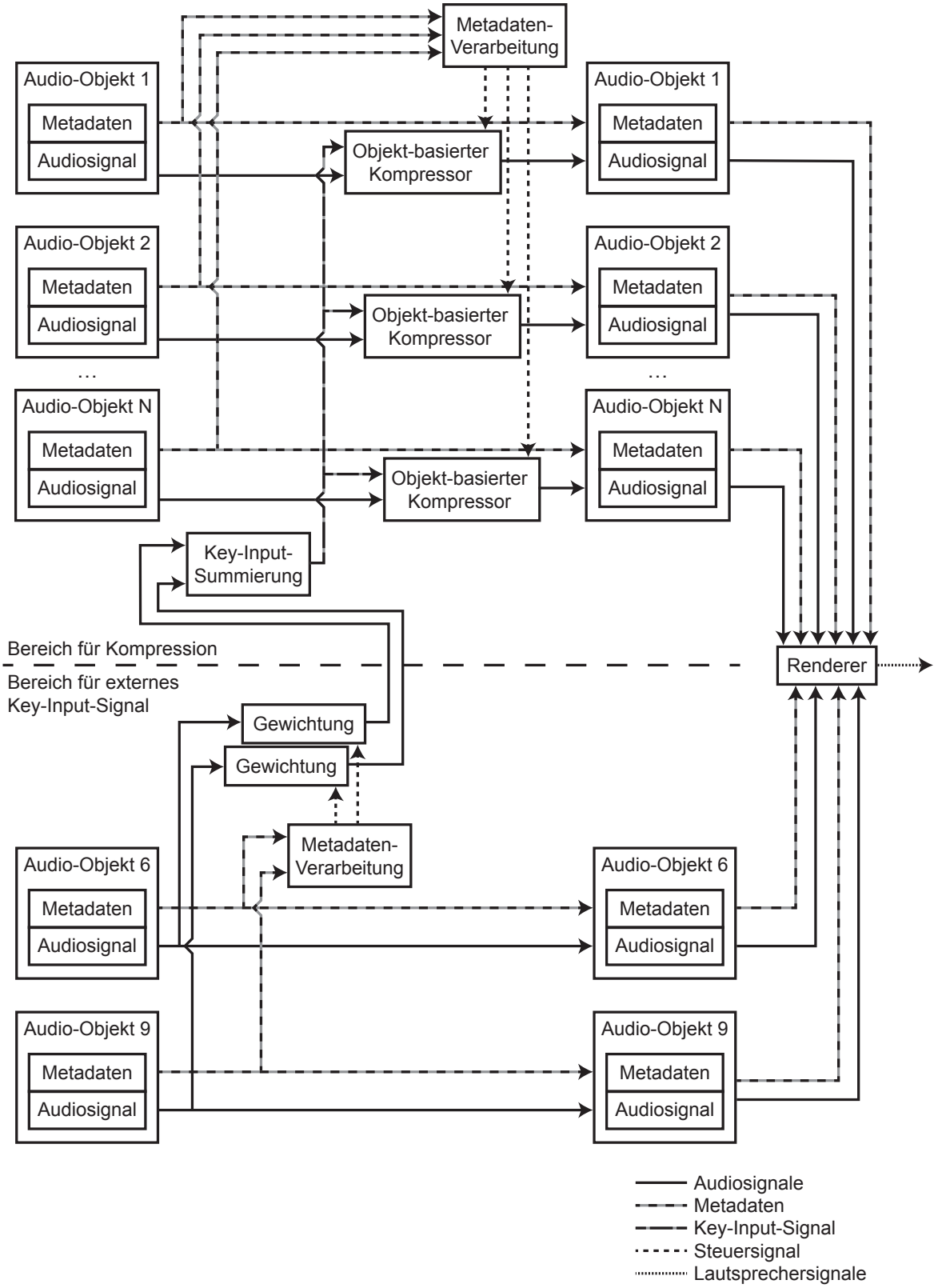

Abbildung 5.7: Signalfluss: Einheitliche Kompression mit objekt-basiertem Kompressor und exter-nem Key-Input-Signal. Quelle: In Anlehnung an Melchior et al., 2011, S. 7

5.2.4 Benutzeroberfläche

Die in Abbildung 5.8 dargestellte grafische Benutzeroberfläche ermöglicht den Nutzern/-innen, die Bereiche für Kompression und externen *Key-Input* zu definieren und die Parameter des Kompressors einzustellen. Die Audio-Objekte werden im Raum visualisiert und die einzelnen Audiosignale mit einem *Metering* für Eingangs- und Ausgangssignal und *Gain Reduction* dargestellt.

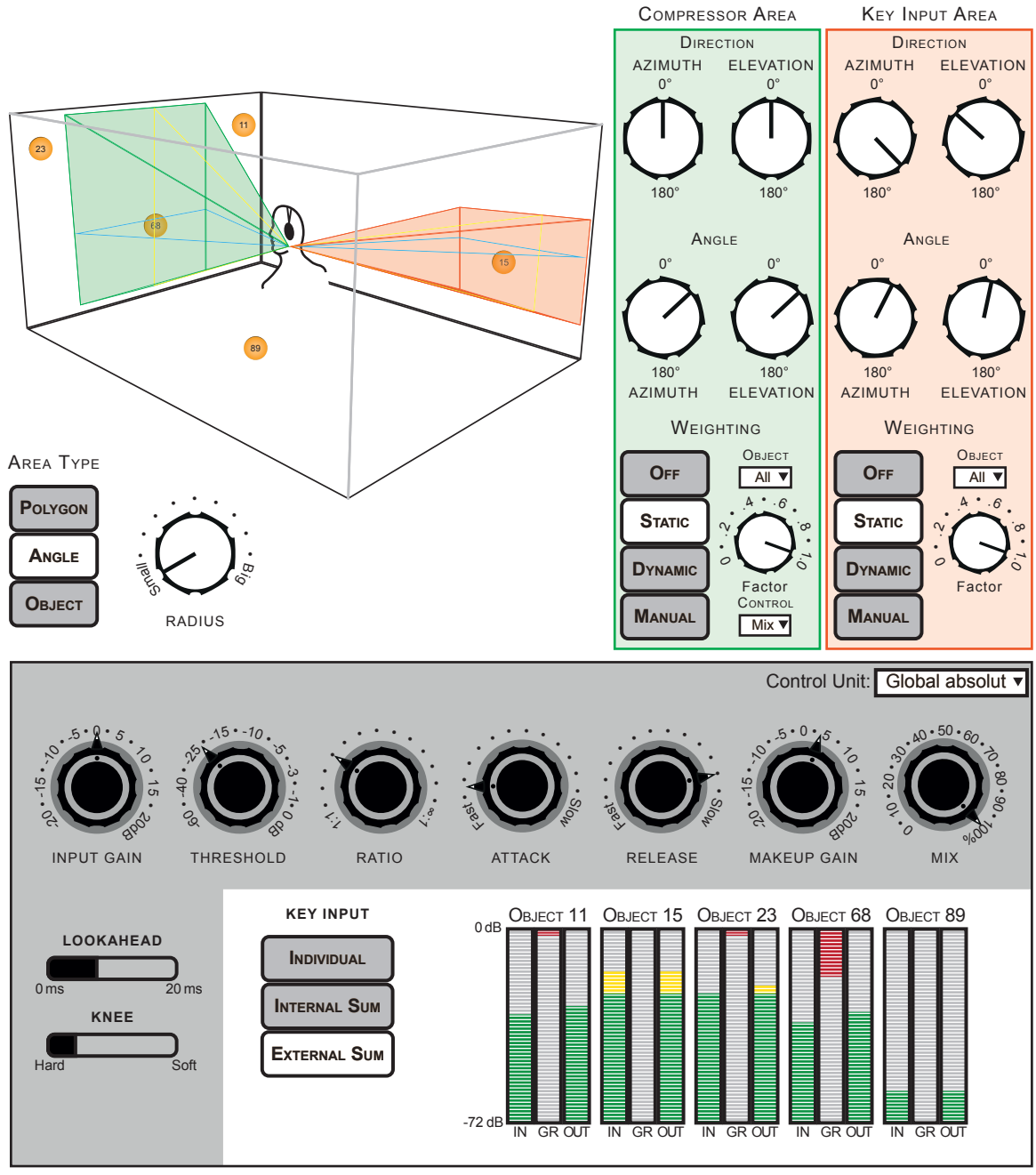

Abbildung 5.8: Grafische Benutzeroberfläche für einen objekt-basierten Kompressor

Auf der linken Seite befindet sich oben die Visualisierung der Audioszene. Dort werden die Audio-Objekte mit Nummern als orangene Kugeln dargestellt. Die grüne Fläche stellt

den definierten Bereich dar, in dem die Audio-Objekte komprimiert werden. Die hellrote Fläche ist der definierte Bereich für das externe *Key-Input*-Signal. In dem dargestellten Beispiel wird Audio-Objekt 68 von Audio-Objekt 15 komprimiert. Das blaue und gelbe Dreieck innerhalb der grünen und hellroten Flächen sind Darstellungen der Winkel, mit denen die Flächen definiert werden. Das blaue Dreieck visualisiert den Azimuth-Winkel; das gelbe Dreieck den Elevation-Winkel.

Rechts neben der Visualisierung der Audioszene befinden sich die Parameter für die Bereichsdefinitionen. Mit den Parametern im grünen Rechteck kann der Bereich für die Kompression und die Gewichtung bestimmt werden. Zur Bereichsdefinition gibt es vier Drehregler. Zwei davon legen die Richtung (*Direction*) fest, die anderen zwei die Winkelöffnung (*Angle*). Im unteren Teil des grünen Rechtecks befinden sich die Parameter zur Gewichtung. Mit den Schaltknöpfen *Off*, *Static*, *Dynamic* und *Manual* können die verschiedenen Gewichtungsmodi angewählt werden. Der danebenliegende Drehregler *Factor* bestimmt den Gewichtungsfaktor. Mit dem Dropdown-Menü oberhalb dieses Drehreglers kann für jedes Audio-Objekt ein individueller Gewichtungsfaktor festgelegt werden. Im dargestellten Fall ist die Gewichtung statisch und der Drehregler *Factor* stellt den Gewichtungsfaktor für alle Audio-Objekte ein. Unterhalb davon befindet sich das *Control*-Dropdown-Menü. Hiermit wird festgelegt, welchen Parameter die Gewichtung steuert. In der Darstellung wird der Mix-Parameter durch die Gewichtung verändert.

Die Parameter im hellroten Rechteck sind für die Definition des *Key-Input*-Bereichs vorgesehen. Sie sind identisch zu denen im grünen Rechteckt mit der Ausnahme, dass kein *Control*-Dropdown-Menü vorhanden ist. Bei dem *Key-Input*-Bereich wird die Gewichtung der Audio-Objekte innerhalb des *Key-Input*-Bereichs ausschließlich für den Pegel der Audio-Signale verwendet.

Unterhalb der Visualisierung der Audioszene auf der linken Seite sind die Schaltknöpfe für die Auswahl des Bereichstyps dargestellt. Mit der Wahl der Bereichstypen verändern sich die definierten Bereiche. Bei *Polygon* können die Bereiche mit Polygonpunkten definiert werden. Bei *Object* wird ein Audio-Objekt ausgewählt. Darum wird in Abhängigkeit des eingestellten Radius ein kugelförmiger Bereich definiert, der sich mit dem Audio-Objekt bewegt. Der Radius wird mit dem Drehregler rechts neben den Schaltknöpfen eingestellt. Mit dem in der Visualisierung ausgewählten Bereichstypen *Angle* werden die Bereiche über die Winkel bestimmt.

In der unteren Hälfte der Abbildung sind die Kompressor-Parameter dargestellt. Mit den sieben Drehreglern *Input Gain*, *Threshold*, *Ratio*, *Attack*, *Release*, *Makeup Gain* und *Mix* können die gängigen Parameter eines Kompressors eingestellt werden. Auf der linken Seite unterhalb der Drehregler wird ein *Lookahead*- und ein *Knee*-Schieberegler dargestellt.

Alle Parameter innerhalb der grauen Fläche können für jedes Audio-Objekt separat oder für alle mit absoluten oder relativen Werten eingestellt werden. Im Dropdown-Menü *Control Unit* oben rechts, wird das Audio-Objekt ausgewählt, für das die Parameter-Einstellungen getroffen werden sollen. Im dargestellten Fall ist *Global absolut* ausgewählt, sodass die Parameter-Einstellungen mit dem absoluten Wert auf alle vorhandenen Kompressor-Instanzen angewendet werden.

Mit den Schaltknöpfen zu *Key Input* wird das Steuersignal für den Kompressor ausgewählt. *Individual* bedeutet, dass jedes Audio-Objekt mit sich selbst komprimiert wird. *Internal Sum* bildet für das *Key-Input*-Signal die Summe der Audiosignale dessen Audio-Objekte im Kompressor-Bereich liegen. *External Sum* summiert für das *Key-Input*-Signal alle Audiosignale dessen Audio-Objekte im *Key-Input*-Bereich liegen.

Das *Metering* rechts unten zeigt für jedes Audio-Objekt den Eingangspegel, die *Gain Reduction* und den Ausgangspegel.

5.2.5 Systemumgebung

Der beschriebene Lösungsansatz für einen objekt-basierten Kompressor kann in verschiedenen Systemumgebungen implementiert werden. Als Plug-in für DAWs kann der objekt-basierte Kompressor auf jeder beliebige mono Audio-Objekt-Spur eingefügt werden. Die Plug-ins kommunizieren untereinander und können dadurch die beschriebenen Funktionen umsetzen. Dabei können die Plug-ins nur auf die Audio-Objekte zugreifen, auf deren Spur sie eingefügt wurden.

Eine hohe Anzahl an Plug-in-Instanzen ist häufig eine Herausforderung für DAWs. Deshalb ist eine separate Umsetzung des objekt-basierten Kompressors insbesondere beim Mastering eine Alternative zum Plug-in. In diesem Fall ist der objekt-basierte Kompressor eine Stand-alone-Software. Die Audio-Objekte (Audiosignale und Metadaten) werden aus der DAW an diese Software geroutet. Dort werden die Metadaten ausgelesen, die Einstellungen für Bereiche und Kompression getroffen und die Audiosignale bearbeitet. Die unveränderten Metadaten und die bearbeiteten Audiosignale werden an den finalen Renderer weitergegeben.

Im besten Fall führt eine Stand-alone-Software auch die Wiedergabe und das Rendering einer objekt-basierten Masterdatei aus. Die Prozessorleistung für die DAW und den Renderer kann damit gespart werden.

5.3 Anforderungen

Im Folgenden sollen die Anforderungen (abgekürzt mit Anf.) an einen objekt-basierten Audioprozessor für die einheitliche Bearbeitung in Abhängigkeit der Metadaten definiert werden.

Allgemein sollen objekt-basierte Audioprozessoren die Bearbeitung in Abhängigkeit der Metadaten vereinfachen und effektiver gestalten. Es muss möglich sein, auf das Gehörte zu reagieren, ohne dabei die Spurenreihenfolge oder das *Panning* der einzelnen Spuren zu kennen (Anf. 1). Tritt bei einer 3D-Produktion ein Problem im hinteren Drittel der rechten Raumseite auf, muss es möglich sein, diesen Bereich zu bearbeiten. Dazu ist für die Toningenieure/-innen irrelevant, welche Audio-Objekte sich dort befinden und welche Kanalnummern sie haben. Die Arbeitsweise wird effektiver, indem die Metadaten genutzt werden, um durch den Computer zu identifizieren, welche Audio-Objekte im Problembereich sind, sodass die Toningenieure/-innen nicht die notwendigen Spuren ausmachen müssen. Die Verwendung von positionsabhängigen Metadaten macht das Schreiben von Automationen überflüssig (Anf. 2). Anhand der Positionsdaten erkennt der Audioprozessor, wenn Audio-Objekte den definierten Bereich betreten oder verlassen. Durch diese Vorteile von objekt-basierten Audioprozessoren können sich die Toningenieure/-innen auf die wesentliche Arbeit – das Bearbeiten des Klangs – fokussieren.

Daneben sollte eine metadatenabhängige gewichtete Bearbeitung der einzelnen Audio-Objekte möglich sein. Positionsdaten können z. B. dazu genutzt werden, um die Distanz der Audio-Objekte zu einem definierten Punkt zu berechnen. Je näher sich ein Audio-Objekt an diesem Punkt befindet, desto höher ist die Gewichtung der Bearbeitung. Die Nutzer/-innen können die Gewichtungen den Parametern der Audioprozessoren zuordnen (Anf. 3). Beispielsweise kann bei einem Equalizer die Gewichtung dem *Gain* eines mittleren Frequenzbands zugeordnet werden. Bei einer höheren Gewichtung erfolgt z. B. eine größere Absenkung des Frequenzbereiches.

Diese Anforderungen gelten für statische Audioprozessoren und bilden nur die Basis für dynamische Audioprozessoren. Bei letzteren Audioprozessoren ist eine Erweiterung der Anforderungen aufgrund des Signalflusses nötig.

Soll der Audioprozessor von Audio-Objekt *A* auf das Audiosignal von Audio-Objekt *B* reagieren, muss das Audiosignal des Audio-Objekts *B* an den Audioprozessor *A* geroutet werden. Soll eine Gruppe von Audio-Objekten eine andere Gruppe steuern, müssen die Audiosignale der ersten Gruppe summiert werden, bevor sie als Steuersignal zu den Audioprozessoren der zweiten Gruppe geroutet werden (Anf. 4). Die daraus entstehenden Routings sind dynamisch und lassen sich von den Metadaten der Audio-Objekte kontrollieren (Anf. 5). Die Gewichtung der Audiosignale, die dem Audioprozessor als *Key-Input*-Signal

dienen, müssen anhand der Positionsdaten der Audio-Objekte gesteuert werden können. Für das Festlegen von Audio-Objekten, die zur Steuerung der Audioprozessoren dienen, wird eine intuitive Auswahl benötigt, wie bei der Definition des Bearbeitungsbereichs (Anf. 6). Das bedeutet, dass sich die Toningenieure/-innen auf ihr Gehör verlassen können und die Auswahl der Audio-Objekte anhand ihrer gehörten Position und nicht ihrer Kanalnummer bestimmt werden kann.

6 Prototypische Implementierung

Um das in Kapitel 5.2 beschriebene Konzept zu testen und zu validieren, wurde ein Prototyp des objekt-basierten Mastering Kompressors entwickelt. Mit diesem Prototyp können 32 Audio-Objekte zeitgleich bearbeitet werden. Die Bearbeitungsmöglichkeiten sind dabei auf die separate und die einheitliche Kompression der Audio-Objekte beschränkt. Die Kompressor-Instanzen können sowohl mit einer internen als auch externen *Key-Input*-Summe gesteuert werden. Dafür können die zwei notwendigen Bereiche (Kompressions- und *Key-Input*-Bereich) definiert werden. Dies geschieht in einem Koordinatensystem über zwei Polygone mit je vier Eckpunkten. Zur besseren Orientierung werden in dem Koordinatensystem auch die Audio-Objekte visualisiert.

Der Kompressor bietet die wichtigsten Parameter (*Input-Gain*, *Threshold*, *Ratio*, *Attack*, *Release*, *Make-up-Gain*) sowie ein *Metering* für Input- und Output-Level und die *Gain Reduction*. Diese Parameter sind absolute Werte für alle Kompressor-Instanzen, sodass nur globale und keine individuellen Einstellungen getroffen werden können.

Der Prototyp ist innerhalb seiner Entwicklungsumgebung eine Stand-alone-Software die sowohl den Player als auch den Renderer beinhaltet. Es ist keine zusätzliche Software notwendig.

6.1 Entwicklungsumgebung und Format

Die prototypische Implementierung des objekt-basierten Kompressors wurde in der grafischen Entwicklungsumgebung *Max 8* von Cycling ‘74 umgesetzt. Die Software bietet verschiedene Bausteine (sogenannte *Objekte*) und grafische Bedienoberflächen zur Entwicklung von interaktiven Anwendungen an. Die Objekte können über sogenannte *Patchcords* verbunden werden. Drittanbieter können eigene Erweiterungspakete mit weiteren Objekten anbieten (Cycling ‘74, n.d.-b). Alle Nutzer/-innen können auch ihre eigenen Objekte programmieren und die Programmiersprache *JavaScript* über Objekte in die Entwicklungsumgebung einbinden (Cycling ‘74, n.d.-a).

Zur Entwicklung des Prototyps wurden Max-Objekte, programmierte *JavaScript*-Objekte und das *Spat5*-Erweiterungspaket von Ircam (Ircam, 2022) genutzt. Dieses bietet die notwendige Infrastruktur für einen Echtzeit 3D-Audio Prozessor. Mit den entsprechenden Objekten können ADM-Dateien abgespielt und für Kopfhörer oder ein beliebiges Lautsprechersetup gerendert werden (Ircam Forum, n.d.).

Für die Produktion der Audiobeispiele wurde das Format Dolby Atmos verwendet. Dieses ist ein immersives Audioformat, das die Reproduktion auf einer großen Bandbreite von Geräten und Systemen ermöglicht (Dolby Laboratories Inc., 2021a).

Dolby Atmos ist ein hybrides Format, das 128 Kanäle aus einer Kombination von diskreten und objekt-basierten Kanäle nutzt. Die diskreten Kanäle beschränken sich auf die sogenannten *Bed*-Kanäle. Diese entsprechen einer 7.1.2 Lautsprecheranordnung mit *Left*, *Right*, *Center*, *LFE*, *Left Surround*, *Right Surround*, *Left Rear Surround*, *Right Rear Surround*, *Left Top* und *Right Top*. Die Master einer Dolby Atmos Produktion können als ADM BWF WAV Datei exportiert werden (Dolby Laboratories Inc., 2021b). Damit beinhalten sie alle notwendigen Informationen für eine Reproduktion mit einem anderen Wiedergabegerät.

6.2 Benutzeroberfläche

Die grafische Benutzeroberfläche des Prototyps ist für eine einfache Übersicht auf seine wesentlichen Funktionen beschränkt und in verschiedene Sektionen unterteilt (siehe Abbildung 6.1). Die Unterteilung dient ausschließlich der visuellen Trennung der Parameter. Einstellungen innerhalb einer Sektion haben Auswirkungen auf den gesamten Signalfluss, der bereichsübergreifend ist (siehe Kapitel 6.3).

Abbildung 6.1: Benutzeroberfläche des Prototypen: Überblick

Die Wiedergabesteuerung (siehe Abbildung 6.2) ermöglicht das Laden und Abspielen von ADM-Dateien. Die vorhandenen Schaltflächen führen entweder einmalige Befehle aus (*Buttons)* oder können umgeschaltet werden (*Toggle-Buttons*). In das Drag-and-drop-Feld können ADM-Dateien gezogen werden. Der Fortschrittsbalken informiert über die Spielzeit und kann zum Vor- und Zurückspulen genutzt werden.

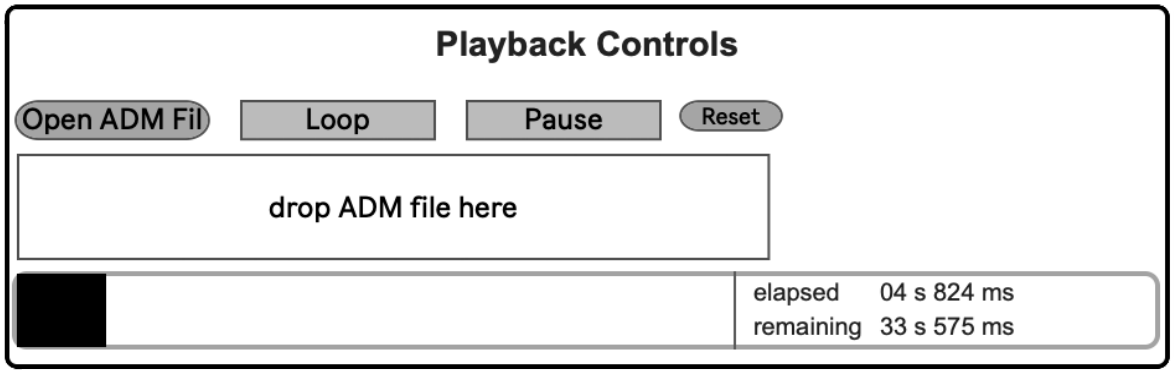

Abbildung 6.2: Benutzeroberfläche: Wiedergabesteuerung. Übernommen und erweitert aus dem Objekt `spat5.adm.play128~` von Ircam (2022)

Diese Benutzeroberfläche ist aus dem `spat5.adm.play128~` Objekt entnommen und mit dem Reset-Button zum Zurücksetzen aller Parameter innerhalb des Patches erweitert worden.

Mit den Schaltflächen in der Sektion *Output Format* (siehe Abbildung 6.3) kann die Konfiguration des Wiedergabesystems festgelegt werden. Dabei können sowohl Kopfhörer (*Headphones*) als auch verschiedene Lautsprecher-Anordnung (*Loudspeakers Layout*) ausgewählt werden.

Beim Rendering für Kopfhörer werden die Audio-Objekte binauralisiert. Dafür stehen verschiedene HRTFs (*hrtf*) zur Verfügung. *HRTFs*, (engl. *Head Releated Transfer Function*) sind kopfbezogene Übertragungsfunktionen. Sie beschreiben die Veränderungen, die das Signal einer Schallquelle erfährt, bis es auf das Trommelfell der Hörer/-innen trifft. Diese *HRTFs* sind für jeden Menschen individuell, da die Verzerrungen des Signals von Charakteristika des Oberkörpers, des Kopfes und der Ohren abhängig ist.

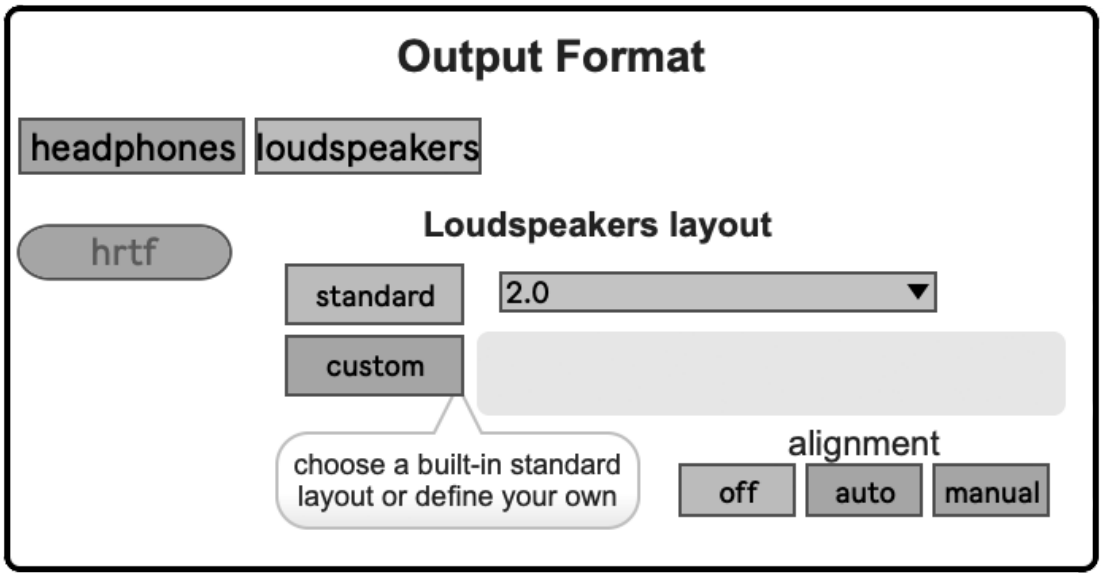

Abbildung 6.3: Benutzeroberfläche: Ausgabe-Format. Übernommen aus dem Objekt `spat5.adm.speaker.or.hrtf` von Ircam (2022)

Beim Rendering für Lautsprecher können standardisierte Anordnungen wie z. B. 2.0, 5.1, 22.2, 13.1 Auro-3D oder Dolby 7.1.4 über das Dropdown-Menü ausgewählt werden. Alternativ kann eine individuelle Lautsprecher-Anordnung erstellt werden, bei der die Anzahl und die Positionen eingestellt werden können. Sind Lautsprecher für die verwendete Anordnung nicht optimal platziert, kann eine automatische oder manuelle Anpassung (*alignment*) vorgenommen werden. Die Signale der Lautsprecher werden dazu entsprechend ihrer Position verzögert und im Pegel angepasst. Diese Benutzeroberfläche wurde von dem Objekt `spat5.adm.speaker.or.hrtf` übernommen.

Für die Bereichsdefinition des Kompressor-Bereichs wird ein zweidimensionales Koordinatensystem verwendet (siehe Abbildung 6.4). Mit den blauen Punkten wird das Polygon für den Bereich definiert, in dem die Audio-Objekte vom Kompressor bearbeitet werden. Unter Berücksichtigung der Reihenfolge (im Uhrzeigersinn Rf, Rb, Lb, Lf), können diese Punkte beliebig positioniert werden. Wird die Reihenfolge vertauscht, entsteht ein Polygon, bei dem sich zwei Seiten kreuzen, sodass zwei Dreiecke entstehen. Dies kann zu Verwirrung führen, da die Linien der Punkte nicht dargestellt werden.

Um die Positionierung des Polygons zu erleichtern, werden die aktiven Audio-Objekte als grüne Punkte ebenfalls im Koordinatensystem visualisiert. Die Nummern auf den Punkten entsprechen den Kanalnummern. Dadurch können die Produzenten/-innen sehen, wann und ob sich ein Audio-Objekt innerhalb des definierten Kompressor-Bereichs befindet. Die Verbindung zwischen ADM-Player und dem Koordinatensystem ist unidirektional. Eine Positionsveränderung der Audio-Objekte durch das Bewegen eines grünen Punktes ist nicht möglich.

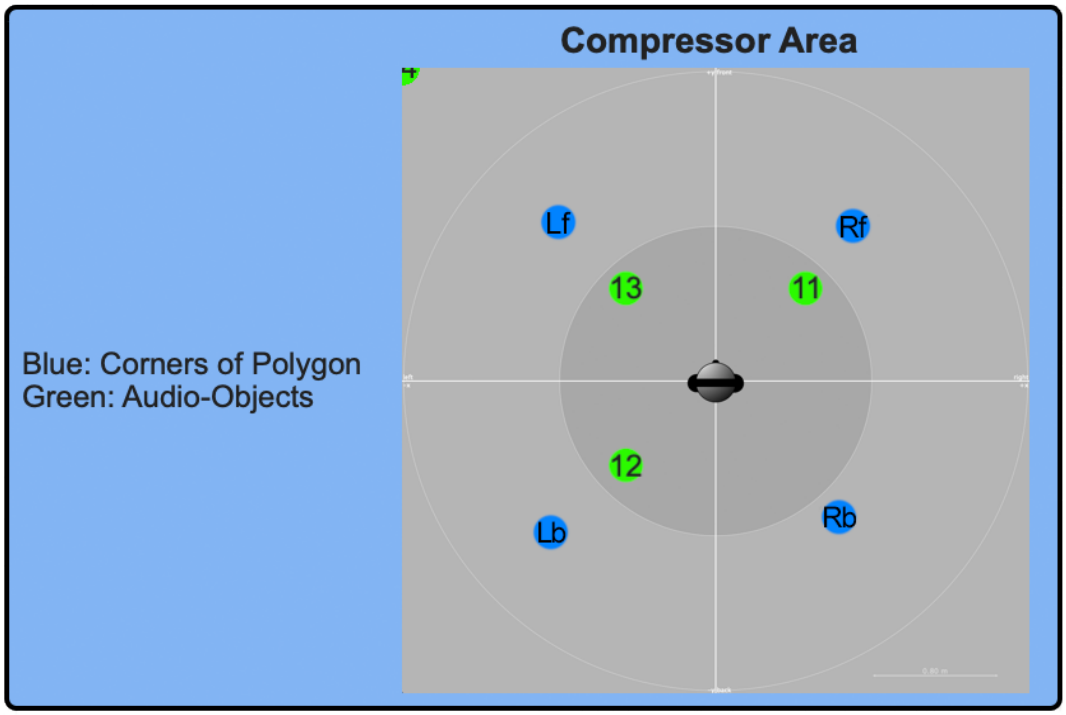

Abbildung 6.4: Bereichsdefinition: Kompressor-Bereich

Obwohl die Darstellung ausschließlich zweidimensional ist, befinden sich die Audio-Objekte im dreidimensionalen Raum. Das Polygon ist ebenfalls als dreidimensional zu verstehen. Alle Audio-Objekte, die auf der Fläche des Polygons dargestellt werden, liegen innerhalb des definierten Bereichs unabhängig von ihrer z-Koordinate.

Die Bereichsdefinition für das *Key-Input*-Signal in Abbildung 6.5 verhält sich äquivalent zu der Bereichsdefinition für den Kompressor. Die roten Punkte stellen die frei bewegbaren Polygonecken dar. Die grünen Punkte visualisieren die aktiven Audio-Objekte.

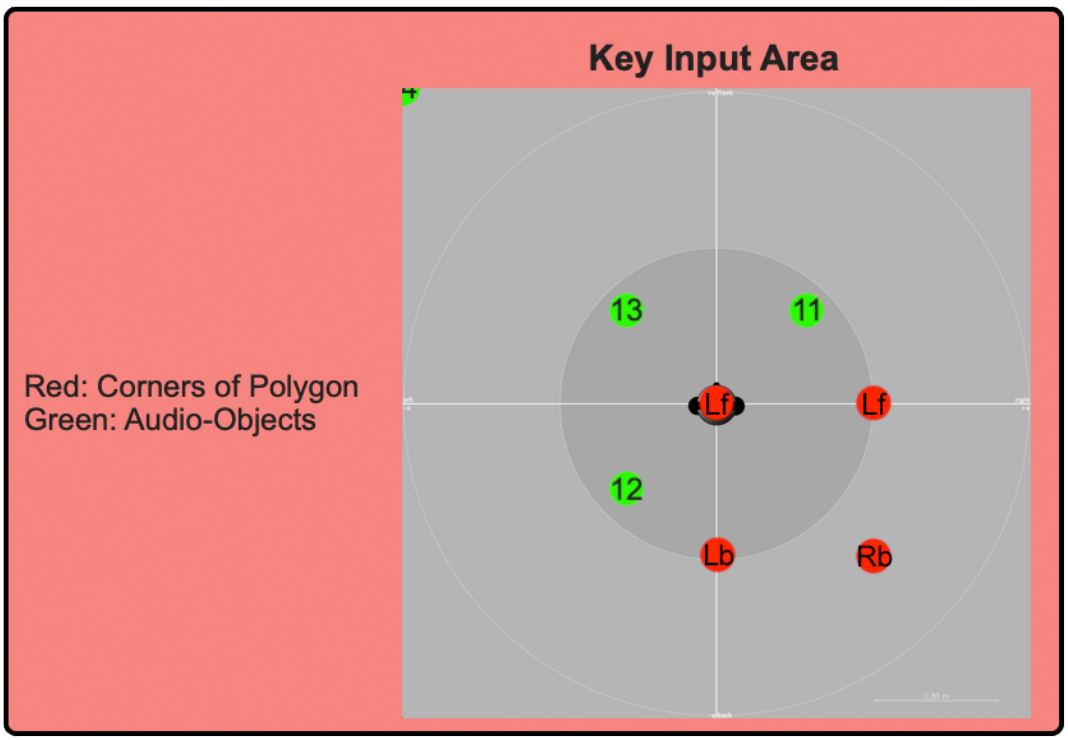

Abbildung 6.5: Bereichsdefinition: Key-Input-Bereich

In der Sektion *Compressor*, dargestellt in Abbildung 6.6, können die Parameter des Kompressors eingestellt werden. Dazu wird eine Kombination aus Buttons, Toggle-Buttons, Drehreglern und veränderbaren Zahlenfeldern verwendet.

Die Einstellungen der Parameter beziehen sich auf alle 32 Kompressor-Instanzen innerhalb des Patches und können über die Benutzeroberfläche nur global verändert werden.

Mit dem Toggle-Button *On* bzw. *Off* wird der Kompressor ein- oder ausgeschaltet. Bei ausgeschaltetem Kompressor wird das Signal nicht durch den Kompressor geroutet, sondern umgeht diesen vollständig.

Das Zahlenfeld *Envelope Time* stellt die Dauer der Durchschnittsbildung der Hüllkurve ein. Diese ist in Millisekunden angeben. Das danebenliegende Zahlenfeld *Delay Compensation* verzögert das Hauptsignal um einen einstellbaren Prozentwert. Dieser Wert wird mit der *Envelope Time* multipliziert. Mit dem in der Abbildung dunkelgrün dargestellten Toggle-

Button *RMS* kann die Art der Hüllkurven-Erzeugung zwischen *Peak* und *RMS* umgeschaltet werden (siehe Kapitel 2.5).

Die Drehregler *Input*, *Thr*, *Ratio*, *Att*, *Rel* und *Make-Up* verhalten sich wie die in Kapitel 2.3.2.1 beschriebenen Parameter. Mit *Input* lässt sich der *Gain* des Eingangssignals verändern. *Thr* ist die Abkürzung für *Threshold* und stellt den Schwellenwert ein, den ein Signal überschreiten muss, damit es komprimiert wird. Att und Rel stehen für *Attack* und *Release* und verändern das Verhalten bei oder nach einer Pegelveränderung des *Overshoots* (siehe Kapitel 2.5).

Die drei oberen Buttons auf der rechten Seite beziehen sich alle auf das Input-Signal des *Side-Chains*. Ist der Toggle-Button *SC On* deaktiviert (*SC Off*), wird das Eingangssignal jedes Kompressors gesplittet und über den Hauptweg und den *Side-Chain* geroutet. Dementsprechend wird jedes Audio-Objekt mit dem eigenen Audiosignal komprimiert. Ist der Toggle-Button wie in Abbildung 6.6 eingeschaltet, werden die Audio-Signale mit einem *Key-Input*-Signal komprimiert, das sich in Abhängigkeit der Einstellungen anders zusammensetzt. Der dafür relevante Einstellungsparameter befindet sich direkt unterhalb des *SC On* Toggle-Buttons. Die Buttons *Internal* und *External* bilden zusammen einen Schalter, sodass immer nur ein Button ausgewählt sein kann. Ist *Internal* ausgewählt, besteht das *Key-Input*-Signal aus der Summe aller Audiosignale der Audio-Objekte, die sich innerhalb des Kompressor-Bereichs befinden. In diesem Fall werden die Audio-Objekte mit der Summe aus ihrem eigenen Audiosignal und dem der umliegenden Audio-Objekte komprimiert.

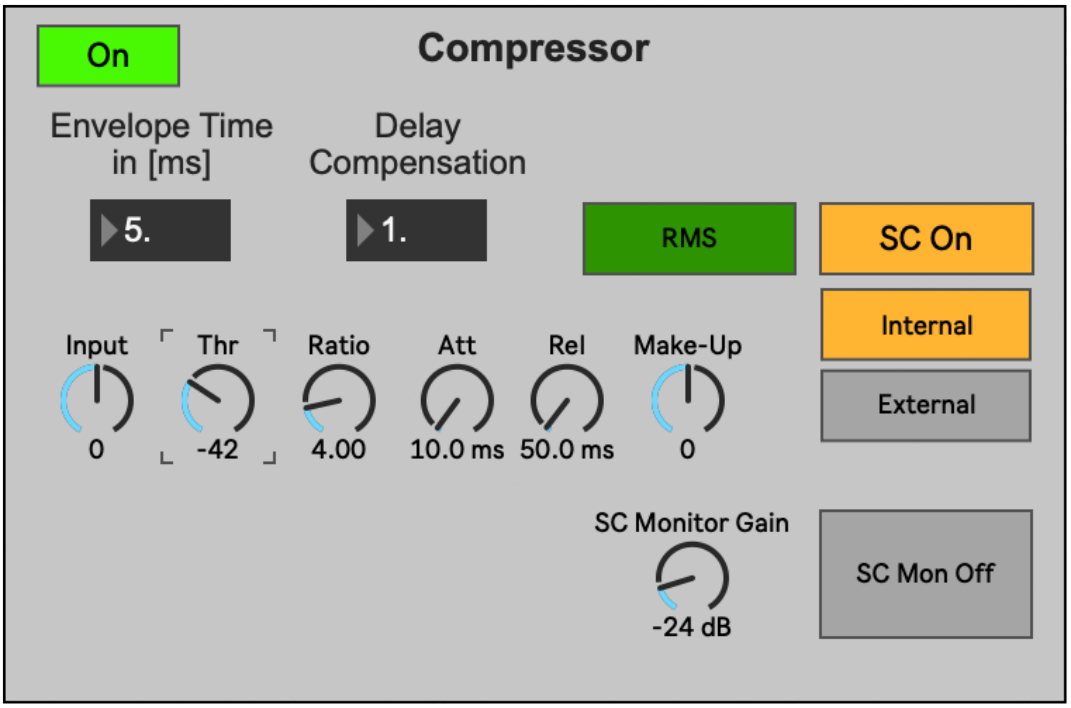

Abbildung 6.6: Benutzeroberfläche: Kompressor-Parameter

Die Einstellung *External* wird verwendet, wenn ein *Key-Input*-Bereich definiert wird. In diesem Fall besteht das *Key-Input*-Signal aus der Summe der Audiosignale aller Audio-Objekte, die sich innerhalb des zweiten *Key-Input*-Bereichs befinden. Damit können Audio-Objekte innerhalb des Kompressor-Bereichs von Audio-Objekten außerhalb des Kompressor-Bereich komprimiert werden.

Der Toggle-Button *SC Mon Off* bzw. *SC Mon On* (Abkürzung für *Side-Chain Monitoring*) und der links davon positionierte Drehregler *SC Monitor Gain* ermöglicht das Abhören des *Side-Chain*-Signal. Dies ist eine hilfreiche Funktion, um zu prüfen, ob der ausgewählte Bereich die korrekten Audio-Objekte für das *Key-Input*-Signal umfasst. Ist das *Side-Chain-Monitoring* eingeschaltet, wird statt des komprimierten Audiosignals das *Side-Chain*-Signal ausgegeben. Da dieses Signal über alle 32 Audiokanäle wiedergegeben und dadurch deutlich verstärkt wird, kann der Pegel über den *SC Monitor Gain* reduziert werden. Diese Funktion ist ausschließlich zur Überprüfung gedacht.

Abbildung 6.7 stellt die *Metering*-Sektion dar. Diese ist in 32 Untersektionen (in Abbildung 6.7 sind nur 16 dargestellt) unterteilt, die mehrere *Meters* für jedes der 32 Audio-Objekte beinhalten. Das linke *Meter* (*Input*) zeigt stets den Pegel des Eingangssignals. Der rechts danebenliegende *Meter* (*Output*) zeigt das Ausgangssignals. Erfährt das Signal keine Bearbeitung, sind *Input* und *Output* identisch. Je größer der Pegel, desto mehr horizontale Balken werden grün, gelb oder orange. Hat das Signal $0\ dB$ oder mehr auf der verwendeten Skala, zeigt das *Meter* einen roten horizontalen Balken ganz oben an.

Zwei weitere *Meter* bilden die *Gain Reduction* ab. Auf der rechten Seite stellt ein *Meter* ähnlich dem von *Input* und *Output* die *Gain Reduction* mit roten horizontalen Balken dar. Je größer die *Gain Reduction*, desto mehr horizontale Balken werden rot. Die *Gain Reduction* ist gegenläufig zum Pegel des Audiosignals. Deshalb bewegen sich die *Meter* in entgegengesetzter Richtung (von oben nach unten) zu den von *Input* und *Output*.

Links neben dem *Gain-Reduction-Meter* wird die *Gain Reduction* als Kurve im zeitlichen Verlauf dargestellt. Diese Kurve entspricht der invertierten Hüllkurve, die als Steuersignal genutzt wird, um das Hauptsignal zu komprimieren. Mit dieser Kurve kann die *Gain Reduction* sehr genau dargestellt werden.

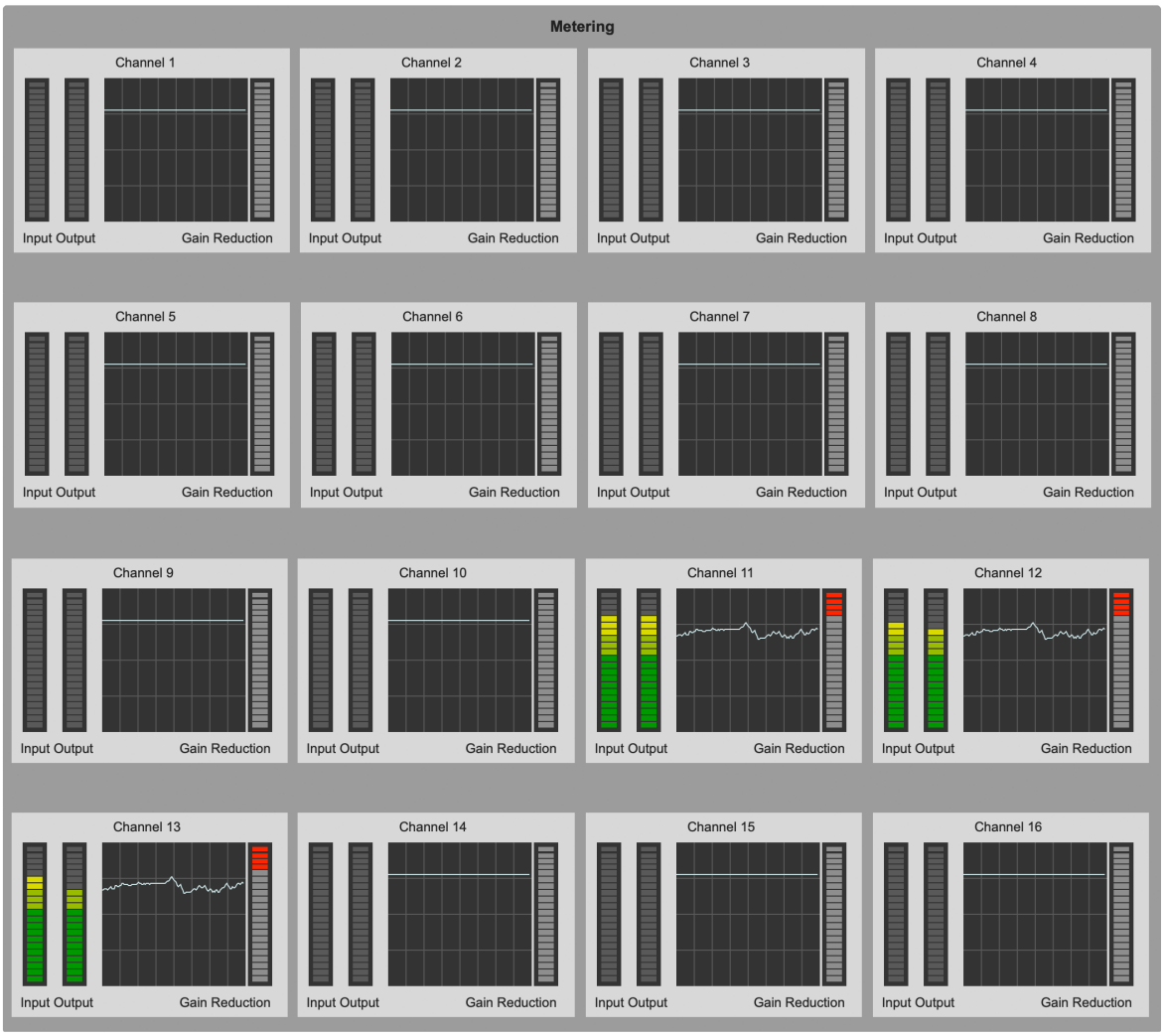

Abbildung 6.7: Benutzeroberfläche: Metering

6.3 Signalverarbeitung

Um den Signalfluss des Prototyps zu erläutern, wird nur auf die wesentlichen Bearbeitungsprozesse eingegangen und nicht auf die einzelnen Fallentscheidungen, die aufgrund von Parametereinstellungen wie z. B. das Umschalten zwischen *RMS* und *Peak* durchlaufen werden. Da der Signalfluss des Prototyps den in Kapitel 5.2 beschriebenen Varianten aus Abbildung 5.3, Abbildung 5.5 und Abbildung 5.6 (mit Ausnahme der Gewichtung) entspricht, wird der Fokus in diesem Kapitel auf die Bearbeitungsprozesse, deren Aufbau und Funktion gelegt. Für ein besseres Verständnis werden – nach einem Überblick über den gesamten Signalfluss und die darin eingebundenen Bearbeitungsprozesse – die beiden wichtigsten Objekte separat erläutert.

6.3.1 Überblick

In Abbildung 6.8 ist der Überblick des Signalverlaufs für die Einzelbearbeitung grafisch dargestellt. Für die Wiedergabe von objekt-basierten Audio-Dateien wird der Patch *ADMix Renderer* aus dem *Spat5*-Erweiterungspaket von Ircam adaptiert. Zum einen ist es damit möglich, ADM-Dateien wiederzugeben und Audio-Objekte auf Kopfhörer und verschiedene Lautsprecher-Anordnungen zu rendern. Zum anderen werden die Metadaten zur Position als *OSC*[7]-Stream ausgegeben.

Die Audio-Objekte werden nach dem Decodieren durch das Ircam-Objekt `spat5.adm.play~` in den Bearbeitungsweg geroutet. Erst nach der Bearbeitung werden die Audio-Objekte von dem Ircam-Objekt `spat5.adm.renderer~` für das korrekte Ausgabeformat gerendert. Nach der Decodierung liegen die Audiosignale als einzelne Kanäle und die Metadaten als kontinuierlicher Datenstrom vor, dessen Inhalt den separaten Kanälen zugeordnet werden kann.

Die Metadaten für jedes Audio-Objekt werden innerhalb des Objekts `isInsidePolygon2D_Patch` mit den Koordinaten des über die Benutzeroberfläche aus Abbildung 6.4 definierten Bereichs verglichen. Dabei gibt es für jedes Audio-Objekt eine separate Instanz des `isInsidePolygon2D_Patch` Objekts. Der Vergleich zwischen Metadaten und Polygon geschieht innerhalb des Objekts `js polygonAlgorithm.js`. Dieses ermittelt, ob ein Audio-Objekt innerhalb des Polygons liegt oder nicht.

Der Ausgabewert des Objekts wird an das Objekt `signalProcessing` weitergeleitet. Dieses liegt ebenfalls für jedes Audio-Objekt separat vor. Innerhalb dieses Objekts wird die Ausgabe des Algorithmus genutzt, um zu entscheiden, ob das Audiosignal des Audio-Objekts bearbeitet wird oder nicht. Soll das Audiosignal bearbeitet werden, wird es zu der Kompressor-Instanz geroutet. Hier wird das Audiosignal komprimiert.

Nach der Bearbeitung wird das Audiosignal an den `spat5.adm.renderer~` gesendet. Der Renderer interpretiert die Metadaten und berechnet Lautsprechersignale. Diese werden über entsprechenden Audioausgänge ausgegeben.

Für ein externes *Key-Input*-Signal wird mit der Benutzeroberfläche aus Abbildung 6.5 der *Key-Input*-Bereich definiert. Diese Koordinaten werden für jedes Audio-Objekt innerhalb des Objekts `SCisInsidePolygon2D_Patch` verglichen. Abgesehen von den Werten ist dieser Prozess identisch mit dem innerhalb des Objekts `isInsidePolyogn2D_Patch`. Der Ausgabewert wird an das Objekt `SCsignalProcessing` geroutet. Ist der Ausgabewert größer als 1, wird das Signal an die Summierung für den externen *Side-Chain* gesendet. Andernfalls wird das Audiosignal nicht weitergeleitet.

[7] OSC: „Open Sound Control“. Protokoll für netzwerkbasierte Kommunikation zwischen Geräten

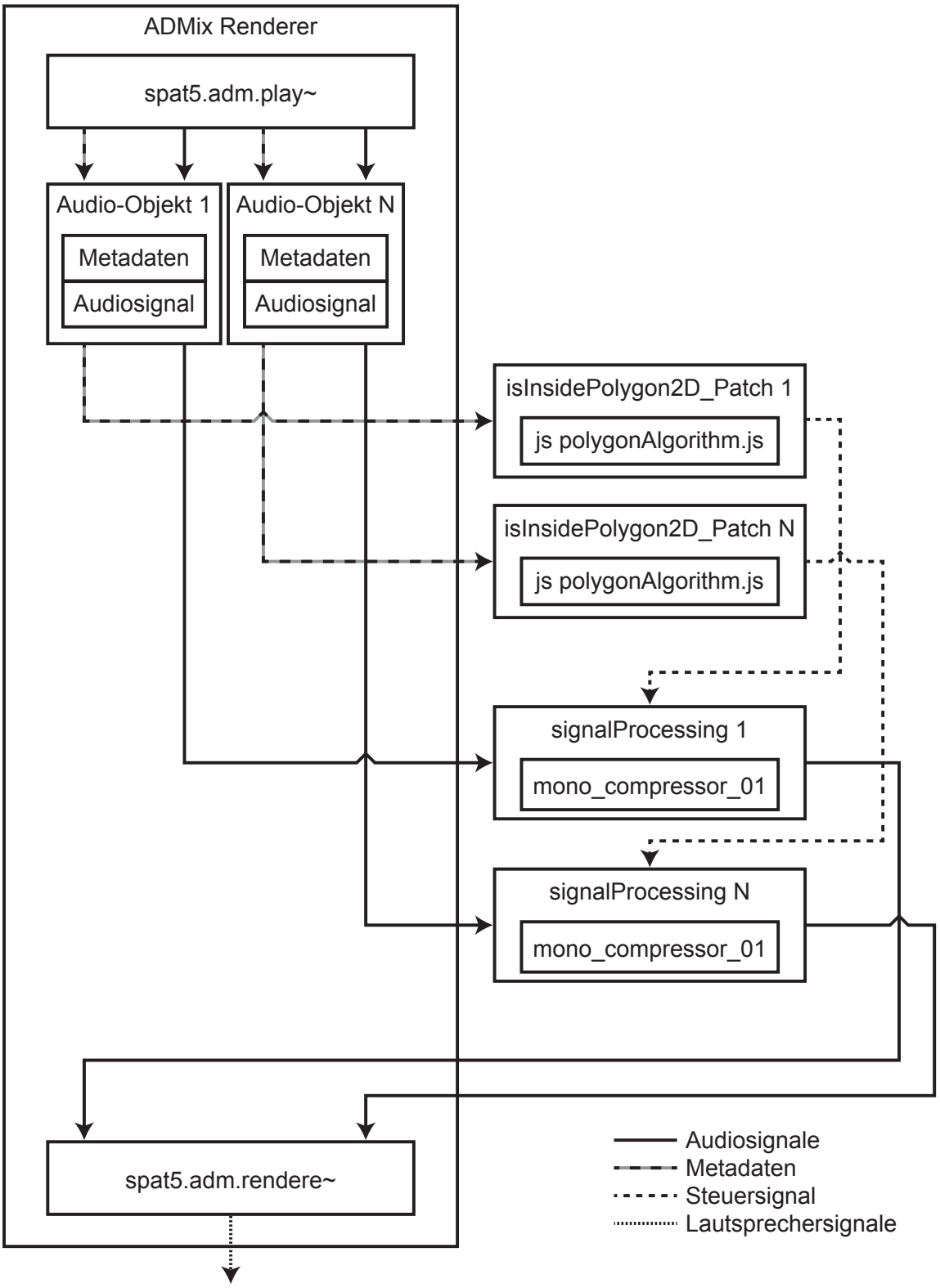

Abbildung 6.8: Überblick über den Signalfluss ohne Key-Input des Prototyps. Quelle: In Anlehnung an Melchior et al., 2011, S. 7

6.3.2 Metadaten Verarbeitung

Bei der Wiedergabe einer ADM-Datei mit dem `spat5.adm.player~` werden die Audiosignale und die Metadaten getrennt voneinander ausgegeben. In einem Paket sind alle Metadaten für einen Zeitabschnitt (Frame) enthalten. Das Ircam-Subpatch `p convert` zerlegt dieses Paket in seine einzelnen Komponenten und filtert dabei die Positionsmetadaten heraus. Die Ausgabe dieses Subpatches gibt die Audio-Objekt-Nummer und die Polarkoordinaten der Position an: z. B. `/source/13/aed/ 0. 3.0756 1.0014.` Dieser Stream beinhaltet die Positionsdaten von allen aktiven Audio-Objekte.

Das Ircam-Objekt `spat5.viewer` kann mit diesem Stream die Audioszene visualisieren. Außerdem konvertiert es die Polarkoordinaten (AED) in kartesische Koordinaten (xyz). Dies erleichtert den Vergleich zwischen den Koordinaten des Polygons und der Audio-Objekte. Die Ausgabe des Ircam-Objekts `spat5.viewer` wird gesplittet. Während diese einerseits einen weiteren Konvertierungsprozess durchläuft und anschließend an den `spat5.adm.renderer~` gesendet wird, wird sie andererseits an das Subpatch `p objectDisplay` und an die Subpatches `p MetadataController1` und `p SCMetadataController2` gesendet.

`p objectDisplay` bereitet die Metadaten so auf, dass sie innerhalb der Benutzeroberflächen *Compressor Area* (siehe Abbildung 6.4) und *Key Input Area* (siehe Abbildung 6.5) dargestellt werden können.

Die Subpatches `p MetadataController1` und `p SCMetadataController2` verwenden den Metadatenstream, um die Audiobearbeitung zu steuern. Beide sind mit Ausnahme der verwendeten Werte identisch. `p MetadataController1` vergleicht das Polygon der *Compressor Area* mit dem Metadatenstream und steuert das Routing für das Eingangssignal der Kompressor-Instanzen. `p SCMetadataController2` vergleicht das Polygon der *Key Input Area* mit dem Metadatenstream und steuert das Routing für das externe *Key-Input*-Signal. Im Folgenden wird die Beschreibung auf den Subpatch `p MetadataController1` beschränkt.

Für die Definition der Polygone wird das Max-Objekt `nodes` verwendet. Dieses kann Punkte (sogenannte *Nodes*) innerhalb eines zweidimensionalen Koordinatensystems darstellen und die Position dieser *Nodes* ausgeben. Die Ausgabe erfolgt in dem Format:

`[query] [Node-Number] [x] [y] [Size] [get-message]`

Innerhalb des Subpatches `p VertexesPrep1` bzw. `p VertexesPrep2` wird die Ausgabe auf die nötigsten Informationen reduziert und für eine Darstellung im `spat5.viewer` adaptiert. Die Ausgabe wird innerhalb des Subpatches `p sort` wie folgt verändert:

```
node 4 0. 0. 0. 1
```

Ausgabe

```
4 0. 0. 0. 1
```

[1]

```
0. 0. 0. 1
```

[2]

```
0. 0.
```

[3]

```
-3.5 -3.5
```

[4]

Da ausschließlich die *Nodes* des Polygons in dem Max-Objekt enthalten sind, kann im ersten Schritt [1] die Ausgabe um die *query*-Information reduziert werden. In Schritt [2] werden die Ausgaben nach ihrer *Node*-Nummer sortiert und auf die Koordinaten und die Größe und die *get-Message* reduziert. Diese beiden letzten Informationen sind für den Prototypen irrelevant und werden im dritten Schritt [3] entfernt. Abschließend werden die Koordinaten so skaliert, dass sie dem Koordinatensystem des `spat5.viewer` entsprechen [4]. `p sort` gibt die reduzierten Polygonskoordinaten gesammelt als einzelnen Stream aus:

```
[x1] [y1] [x2] [y2] [x3] [y3] [x4] [y4]
```

Der Stream für die Polygonpunkte wird im Subpatch `p PackForMetadataController1` bzw. `p PackForMetadataController2` wieder in die vier ursprüngliche xy-Koordinaten zerlegt.

Das Subpatch `p MetadataController1` empfängt sowohl die Positionskoordinaten der Audio-Objekte als auch die Koordinaten des definierten Polygons. Innerhalb dieses Subpatches sind 32 Instanzen der Objekte `isInsidePolygon2D_Patch` und `signalProcessing` enthalten, sodass die Positionsdaten und die Audiosignale jedes Audio-Objekts separat verarbeitet werden.

Das Patch `isInsidePolygon2D_Patch` bereitet den Metadatenstream und die Polygonskoordinaten für den *JavaScript*-Algorithmus innerhalb des Objekts `js polygonAlgorithm.js` vor. Dazu werden aus dem Metadatenstream die Metadaten für ein spezifisches Audio-Objekt gefiltert. In jeder der 32 Instanzen des Objekts `isInsidePolygon2D_Patch` wird ein anderes Audio-Objekt gefiltert. Die vier vorliegenden Koordinaten des Polygons werden ebenfalls an jede der 32 Instanzen geroutet. Für das Aufrufen der *JavaScript*-Funktion muss den Koordinaten noch der Funktionsname *coordinates* vorangestellt werden.

Der Wert dieser Ausgabe wird im mehrfach instanziierten Objekt `signalProcessing` verwendet, um das Max-Objekt `matrix~` zu kontrollieren. Das Max-Objekt steuert das Routing zum Kompressor.

Um eine abrupte Änderung des Audiosignals beim Betreten und Verlassen des definierten Bereichs zu verhindern, werden die Werte mit dem Max-Objekt `slide` über einen definierten Zeitraum interpoliert. Dies geschieht in dem Subpatch `p scale & invert`. Die interpolierten Werten können danach verwendet werden, um das Routing zum Kompressor zu kontrollieren. Bei einem Wert von 0 wird das Audiosignal direkt zum Ausgang des Objekts `signalProcessing` geroutet, ohne bearbeitet zu werden. Ist der Wert 1, wird das Audiosignal durch den Kompressor geroutet, bevor es zum Ausgang des Objekt `signalProcessing` geroutet wird.

JavaScript-Algorithmus

Innerhalb des Objekts `js polygonAlgorithm.js` bestimmen mehrere Methoden, ob das Audio-Objekt innerhalb des Polygons liegt. Dazu müssen die Koordinaten aus *Max 8* als Arrays in dem *JavaScript*-Code deklariert werden. Dies geschieht in der Funktion `coordinates`. Diese wird aus *Max 8* durch das Voranstellen ihres Namens vor den Koordinaten aufgerufen. Die Koordinaten werden in Abhängigkeit ihres Eingangs, dem sogenannten `inlet`, entweder der Variablen `object` oder den Variablen `p1, p2, p3` oder `p4` zugeordnet. Diese letzten Variablen werden im Array `polygon` abgelegt.

Wenn in den Variablen Koordinaten für das Audio-Objekt und das Polygon vorhanden sind, wird die Funktion `test (p1, p2, p3, p4, object)` aufgerufen (siehe Anhang A1). Diese Funktion ist die Hauptfunktion, die alle weiteren Funktionen in Abhängigkeit des Verlaufs aufruft.

Zunächst wird mit der Funktion `objectMatchPoint(p1, p2, p3, p4, object)` getestet, ob sich das Audio-Objekt auf einem der Polygonpunkte befinden (siehe Anhang A2). Dazu wird die x-Koordinate des Audio-Objekt mit jeder x-Koordinate der Polygonpunkte verglichen. Sind zwei x-Koordinaten identisch, wird die y-Koordinate des Audio-Objekts mit der y-Koordinate des Polygonpunktes verglichen. Sind sowohl x-Koordinate als auch y-Koordinate eines Polygonpunktes identisch mit den Koordinaten des Audio-Objekts endet die Funktion, indem sie den booleschen Wert `true` zurückgibt. Andernfalls ist der Rückgabewert `false`.

Ist der Rückgabewert `true` wird der Wert der Variablen `inOut` auf `true` gesetzt. Diese Variable gibt an, ob sich das Audio-Objekt innerhalb des Polygons befindet. Bei dem Rückgabewert `false` wird die nächste Test-Funktion aufgerufen.

Bei dieser nächsten Test-Funktion, `isOnBorder(object, p1, p2, p3, p4)`, wird getestet, ob sich das Audio-Objekt genau auf einer Strecke zwischen zwei

nebeneinander liegenden Polygonpunkten befindet (siehe Anhang A3). Dazu wird die x-Koordinate des Audio-Objekts mit den x-Koordinaten zweier benachbarter Punkte z. B. `p1`, und `p2`, oder `p4`, und `p1`, verglichen. Gleiches geschieht mit den y-Koordinaten. Liegen beide Koordinaten des Audio-Objekts zwischen den Koordinaten der zwei benachbarten Polygonpunkte, befindet sich das Audio-Objekt auf einer Strecke zwischen diesen beiden Punkten. Damit wird es als *innerhalb* des Polygons bewertet und der Rückgabewert der Funktion ist `true`.

Auch in diesem Fall wird die Variable `inOut` auf `true` gesetzt, wenn der Rückgabewert der Funktion `true` entspricht. Wenn der Rückgabewert `false` ist, wird eine letzte Funktion aufgerufen.

Diese letzte Funktion, `pointInPolygon(object, polygon)` (vlasky, n.d.), testet, ob sich das Audio-Objekt innerhalb des Polygons befindet, indem die Umlaufzahl ermittelt wird (siehe Anhang A4). Zunächst wird getestet, ob die y-Koordinate eines Polygonpunktes y_j kleiner oder gleichgroß wie die y-Koordinate des Audio-Objekts ist. Ist diese kleiner oder gleichgroß, wird als nächstes festgestellt, ob die y-Koordinate y_i des Polygonpunktes, der gegen den Uhrzeigersinn vor dem zuerst getesteten Polygonpunkt liegt, größer als die y-Koordinate des Audio-Objekts ist. Ist dies der Fall, wird die Umdrehungszahl wn mit der Formel 6.1 berechnet:

$$wn = (x_i - x_j) \cdot (y - y_j) - (x - x_j) \cdot (y_i - y_j)$$

Formel 6.1: Berechnung der Umlaufzahl (nach vlasky (n.d.))

Ist das Ergebnis größer als 0, wird die Variable `wn` mit 1 addiert.

Ist y_j größer als die y-Koordinate, wird getestet, ob y_i kleiner-gleich y ist. Ist dies der Fall, wird die Umlaufzahl mit der Formel 6.1 berechnet. Bei einem Ergebnis kleiner als 0 wird die Variable `wn` um 1 subtrahiert.

Sind diese Berechnungen für jeden Polygonpunkt durchgeführt worden, ist der Rückgabewert ein boolescher Wert, der `true` ist, wenn `wn` ungleich 0 ist.

Dieser Rückgabewert wird in der Variablen `inOut` abgelegt. Die Variable wird als Stream von dem Objekt `js polygonAlgorithm.js` ausgegeben. Der Datenstream wird verwendet, um das Routing des Audiosignals zu steuern.

6.3.3 Signalverarbeitung

Das Ircam-Objekt `spat5.adm.play~` separiert die Audiokanäle der ADM-Datei. Dadurch besteht die Ausgabe des Ircam-Objekts aus dem Metadatenstream und aus den einzelnen Audiosignalen der Audio-Objekte. Der Prototyp unterstützt bis zu 32 Audio-Objekte und hat dementsprechend 32 Audiokanäle. Diese separaten Audiosignale laufen

zunächst durch das Max-Objekt `live.gain` das ein *Metering* und ein Anpassen der globalen Lautstärke ermögliche. Dieses *Metering* dient ausschließlich der Kontrolle und ist nicht in der Bedienoberfläche enthalten.

Im Anschluss an das *Metering* werden die Audiosignale als Mehrkanalsignal verpackt und an die Subpatches `p MetadataController1` und `p SCMetadataController2` gesendet. Dort wird das Mehrkanalsignal wieder entpackt. Da der Signalfluss aller Audiosignale identisch ist, ermöglicht das Versenden mehrere Audiosignale als Mehrkanalsignal eine übersichtlichere Darstellung.

Innerhalb des Subpatches werden die wieder getrennten Audiosignale an die ihnen zugeordnete Instanz des Objekts `signalProcessing` geroutet. Hier findet das Routing für die Bearbeitung statt. Im ersten Schritt routet das Max-Objekt `matrix~` das Audiosignal entweder direkt zum Ausgang oder zur Kompressor-Instanz. Dabei wird das Max-Objekt von den verarbeiteten Metadaten gesteuert (siehe Kapitel 6.3.2). Ist das Ergebnis des angewandten Algorithmus `false`, liegt das Audio-Objekt außerhalb des Polygons und wird direkt an den Ausgang des `signalProcessing`-Objekts geroutet. Ist das Ergebnis `true`, liegt das Audio-Objekt innerhalb des Polygons und das Audiosignal wird Richtung Kompressor-Instanz geroutet.

Im letzten Fall muss das Audiosignal, bevor es die Kompressor-Instanz erreicht, noch ein weiteres `matrix~`-Objekt passieren. Dieses wird von dem On/Off-Toggle-Button der Benutzeroberfläche aus Abbildung 6.6 gesteuert. Wenn der Kompressor nicht von der/dem Nutzer/-in aktiviert wird, leitet das `matrix~`-Objekt das Audiosignal direkt an Ausgang des `signalProcessing`-Objekts. Bei einem aktivierten Kompressor wird das Audiosignal an das Objekt `mono_compressor_01` geleitet, in dem es komprimiert wird.

Das Audiosignal wird auch an ein `matrix~`-Objekt geleitet, das von dem Toggle-Button *SC On* bzw. *SC Off* gesteuert wird. Ist der Toggle-Button aktiviert, wird das Audiosignal zur Summierung des internen *Key-Input*-Signals an ein Max-Objekt `live.gain~` geschickt. Dieses summiert alle eingehenden Audiosignale und gibt sie als Summe wieder aus. Die Summe wird an die einzelnen Kompressor-Instanzen geroutet, wenn der entsprechende *Side-Chain* Einstellung ausgewählt ist.

Das komprimierte Audiosignal gelangt über den Ausgang des Objekts `mono_compressor_01` und `signalProcessing` zurück in das Subpatch `p MetadatenController1`. Dort wird es mit den anderen Audiosignalen zu einem Mehrkanalsignal zusammengefasst und an das Ircam-Objekt `spat5.adm.renderer~` gesendet.

Kompressor

In dem Objekt `mono_compressor_01` findet die vollständige Bearbeitung der Audiosignale statt. Dazu empfängt das Objekt die Einstellungen der Parameter von der Benutzeroberfläche aus Abbildung 6.6 und hat zwei Eingänge für Audiosignale. Die zwei Ausgänge werden für das bearbeitete Audiosignal und das *Metering* der *Gain Reduction* verwendet.

Der Signalfluss und die Bearbeitung durch die verschiedenen Max-Objekte ist von Cipriani und Giri (2019) adaptiert. Signalfluss und Bearbeitung entsprechen im Grundsatz dem in Kapitel 2.5 beschriebenen Aufbau, weshalb im Folgenden vor allem auf die Umsetzung dieses Aufbaus mit den Max-Objekten eingegangen wird.

Das Eingangssignal wird auf Hauptweg und *Side-Chain* gesplittet. Über das Max-Objekt `selektor~` wird eingestellt, welches Signal (Eingangssignal oder *Key-Input*-Signal) zur Bearbeitung verwendet wird. Innerhalb des Subpatches `p calcEnvelope` wird die Hüllkurve generiert. Dazu wird das Max-Objekt `average~` verwendet, das entweder im *RMS*- oder im *Peak*-Modus über eine in Samples definierte Zeit den Durchschnittswert berechnet. Für den darauffolgenden Schritt wird die Hüllkurve von einem Amplitudenwert in einen Dezibel-Wert umgerechnet. Diese Aufgabe übernimmt das Max-Objekt `atodb~`.

Die in dB konvertierte Hüllkurve wird mit dem Max-Objekt `maximum~` mit dem eingestellten *Threshold* verglichen. Dieses Max-Objekt hat zwei Eingänge und einen Ausgang. Es vergleicht die beiden Eingangswerte und gibt den höheren Wert aus. Das bedeutet, dass die Hüllkurve nur dann von diesem Max-Objekt weitergeleitet wird, wenn sie größer als der eingestellte *Threshold* ist.

In dem folgenden Subpatch `p calcAttenuation` wird die Kompression des Signals berechnet. Die Ausgabe des `maximum~`-Objekts wird mit dem Max-Objekt `scale~` skaliert. Hierbei kann ein unterer und oberer Eingangswert und ein unterer und oberer Ausgangswert festgelegt werden. Wenn der untere Eingangswert mit dem Wert des *Thresholds* festgelegt wird und der oberen Ausgabewerts mit der Formel 6.2 erzeugt das `scale~`-Objekt einen invertierten *Overshoot* aus der Hüllkurve. Dieser invertierte *Overshoot* entspricht der *Gain Reduction*.

$$at = thr - \frac{thr}{rt}$$

Formel 6.2: Berechnung der Reduktion (nach Cipriani und Giri, 2019, S. 424)

Das Signal der *Gain Reduction* wird vom `slide~`-Objekt und den eingestellten *Attack*- und *Release*-Werten geglättet und mit dem *Make-up-Gain* summiert.

Bevor das *Gain-Reduction*-Signal den Subpatch verlässt, wird es mit dem Max-Objekt `dbtoa~` von einem Dezibel- in einen Amplituden-Wert konvertiert.

Das erzeugt *Gain-Reduction*-Signal wird abschließend mit dem Eingangssignal, das innerhalb des Subpatches `p DelayCompensation` um das Produkt der definierten Dauer zur Erzeugung des Durchschnittswerts (*Envelope Time*) und dem *Delay Compensation*-Wert verzögert wurde, multipliziert.

Für die Funktion des *Side-Chain-Monitorings* wird das Ausgangssignals des Kompressors von einem `selektor~`-Objekt gesteuert. Dieses Objekt schaltet zwischen dem komprimierten und dem verzögerten *Side-Chain*-Signal um. Der Selektor wird über den Toggle-Button *SC Mon Off* bzw. *SC Mon On* in der Benutzeroberfläche aus Abbildung 6.6 gesteuert. Ist der Toggle-Button deaktiviert, ist das Ausgabesignal des Objekts `mono_compressor_01` das komprimierte Audiosignal.

7 Diskussion

Mit wachsender Relevanz von objekt-basierten Audioformaten wird die Notwendigkeit von Audioprozessoren größer, die für diese Formate konzipiert sind und die vorhandenen Metadaten mit in die Bearbeitung einbeziehen. Insbesondere im Mastering-Prozess, bei dem ein fertiger Mix hauptsächlich als Einheit bearbeitet werden soll, bieten die bisherigen Audioprozessoren kaum Möglichkeiten für eine effiziente Bearbeitung von objekt-basiertem Audio. Durch die fehlenden diskreten Lautsprechersignale während des Produktionsprozesses ist es schwierig, eine einheitliche Bearbeitung von objekt-basierten Audioformaten umzusetzen. Dies trifft besonders auf die Kategorie der dynamischen Audioprozessoren zu, deren Bearbeitung in Abhängigkeit vom Eingangssignal oder vom *Key-Input*-Signal ist. Das Ziel dieser Studie war die prototypische Implementierung eines entwickelten Konzepts für einen objekt-basierten Mastering-Kompressor.

Im Verlauf der Studie wurde zunächst auf den Prozess des Masterings von Musik und die dabei hauptsächlich verwendeten Arten von Audioprozessoren eingegangen. Bei den beschriebenen Audioprozessoren wurde ein besonderer Fokus auf die Regelverstärker und insbesondere auf den Abwärts-Kompressor gelegt. Dieser spielt beim Mastering eine essenzielle Rolle und dessen Signalverarbeitung stellt einen Teil der Grundlage für das entstandene Konzept dar.

Die Vorstellung von 3D-Audio und den damit verbunden Formaten zeigt, dass objekt-basierte Formate zukünftig die Musikproduktion nachhaltig verändern und die Art des Konsums beeinflussen können. Die Funktionsbeschreibung von objekt-basierten Formaten, deren Audio-Objekte, Metadaten und Rendering liefert den zweiten Teil der Grundlage, auf der das in dieser Arbeit vorgestellte Konzept basiert.

Aufbauend auf diese Grundlagen wurde ein Lösungsansatz entwickelt, der die seit Jahrzehnten etablierten Werkzeuge, die maßgeblich für die Hörgewohnheiten der Konsumenten/-innen verantwortlich sind, für die neuen objekt-basierten Audioformate adaptiert.

Die in Kapitel 5.3 beschriebenen Anforderungen an objekt-basierte dynamische Audioprozessoren werden von den präsentierten Lösungsansätzen erfüllt:

Mit den verschiedenen Möglichkeiten der Bereichsdefinierung (Polygon, Winkel, Objekt) können unabhängig von den Audio-Objekten Bereiche für die Kompression und das *Key-Input*-Signal definiert werden. Die Nutzer/-innen müssen dabei, außer bei der Bereichsdefinierung um ein Objekt, weder den Spurnamen noch die Audio-Objekt-Nummer kennen. Stattdessen nutzt die Metadatenverarbeitung die Positionsdaten und vergleicht diese mit dem definierten Bereich, um zu bestimmen, ob ein Audio-Objekt bearbeitet wird oder nicht (Anf. 1, Anf. 6). Durch diesen automatisierten Vorgang, erfüllt das Konzept auch die

Anforderung Anf. 2, dass keine Automationsdaten von den Nutzern/-innen geschrieben werden müssen, wenn sich Audio-Objekte bewegen.

Die Anforderung Anf. 4 zur Bearbeitung in Abhängigkeit externer *Key-Input*-Signale wird mit einem internen Routing der Audiosignale erfüllt. Dabei können mit den verschiedenen Gewichtungsmodi die Audiosignale für das *Key-Input*-Signal gewichtet werden (Anf. 5). Die Gewichtung kann ebenfalls auf Parameter der Audioprozessoren erfolgen (Anf. 3).

Indem der vorgestellte Lösungsvorschlag für einen objekt-basierten Mastering Kompressor alle Anforderungen erfüllt, kann davon ausgegangen werden, dass er den Nutzern/-innen eine effektive und vereinfachte Bearbeitungsmöglichkeit von objekt-basiertem Audio unter Berücksichtigung der audio-objekt-spezifischen Metadaten bietet.

Mit der Umsetzung des Konzepts im Rahmen einer prototypischen Implementierung konnte der Lösungsansatz technisch validiert werden. Dieser Prototyp des objekt-basierten Mastering-Kompressors ermöglicht eine separate und einheitliche Bearbeitung von mehreren Audio-Objekten in Abhängigkeit ihrer Position (siehe Anhang B, Audiobeispiele 01 bis 06b). Zum einen wurde das Bereichsauswahlverfahren über ein definierbares Polygon umgesetzt. Zum anderen wurde das Routing und die dynamische Summierung für das *Key-Input*-Signal realisiert, das für eine einheitliche Kompression notwendig ist (siehe Anhang B, Audiobeispiele 03 bis 06b).

In weiteren Studien muss die Umsetzung einer positionsabhängigen Gewichtung und Steuerung beliebiger Kompressor-Parameter erforscht werden. Diese sind in diesem Prototyp nicht implementiert. Stattdessen wird im Prototyp, abhängig von den Positionsdaten der Audio-Objekte, nur entschieden, ob ein Audiosignal durch den Kompressor geroutet wird. Ebenfalls sind die vorgeschlagenen Bereichsauswahlverfahren für Polarwinkel und den Radius um ein Audio-Objekt nicht in diesem Prototyp implementiert und müssen in weitergehenden Forschungen validiert werden.

Dennoch zeigt der Prototyp, dass die grundsätzlichen Erwartungen für einen objekt-basierten dynamischen Audioprozessor erfüllt werden können und dass die Anwendung vielversprechend sein kann. Dadurch sind objekt-basierte Audioproduktionen nicht mehr auf statische Audioprozessoren beschränkt. Die Metadaten der Audio-Objekte können verwendetet werden, um die dynamische Audioprozessoren zu steuern und deren *Key-Input*-Signale zu manipulieren.

Die prototypische Implementierung innerhalb von *Max 8* unterliegt einigen Beschränkungen, die zum einen auf die Kombination von Ircam *Spat5* und Dolby Atmos, und zum anderen auf die Prozessorleistung zurückzuführen sind.

Obwohl die *Spat5* Software für *Max 8* Dolby Atmos Dateien wiedergeben und rendern kann, entspricht das Rendering nicht dem Rendering eines Dolby Atmos Renderers. Es

wurde festgestellt, dass bei identisch eingestellten Lautsprecher-Anordnungen in *Spat5*-Renderer und im Dolby Atmos Renderer bei der Reproduktion die Lautsprechersignale nicht identisch sind. Das liegt an den unterschiedlichen *Panning*-Algorithmen, die die jeweiligen Renderer verwenden.

Tests mit unterschiedlichen Metadaten haben ergeben, dass der *Spat5*-Renderer eine z-Koordinate größer als 0 benötigt, um ein Audio-Objekt korrekt zu rendern. Andernfalls bleibt dieses visuell an der letzten Position stehen und wird über den *Center* bzw. an dessen Position im Koordinatensystem 0, 1, 0 (Angabe der kartesischen Koordinaten) wiedergegeben. Deshalb musste bei der Produktion der Audiobeispiele immer die Höhe der Audio-Objekte verändert werden, was in einer reale Mix-Situation eine Einschränkung ist.

Das Erzeugen dreidimensionaler Polygone über eine geeignete Benutzeroberfläche stellte sich als Schwierigkeit heraus, die im Rahmen der Arbeit nicht gelöst werden konnte. Im Prototyp können die Polygone nur als zweidimensionale Form festgelegt werden, die auf der z-Achse als unendlich betrachtet werden. Diese Einschränkung kommt dem zuvor beschriebenen Problem der Höhe größer 0 zugute, da die z-Koordinate ohne Auswirkungen auf die Bereichsauswahl verändert werden kann.

Die hohe Spurenanzahl, die Dolby Atmos Dateien beinhalten können (bis zu 128), schränkten die Prozessorleistung von *Max 8* deutlich ein. Aufgrund der separaten Instanzen und *Meter*-Visualisierung für jedes Audio-Objekt, musste der Prototyp auf 32 Kanäle beschränkt werden. Dieses Problem zeigt die Schwäche von objekt-basierten Audioprozessoren auf: die Prozessorlast.

Obwohl das Konzept die Verarbeitung der Metadaten für eine Gewichtung der Audio-Objekte und der Parameter der Audioprozessoren enthält, ist die Frage, auf welche Art die Parameter gewichtet werden, unbeantwortet geblieben. Eine einfache Veränderung eines Parameters würde nicht immer den gewünschten Zweck erzielen. Um eine höhere Kompression zu erzeugen, gibt es verschiedene Möglichkeiten: Einerseits kann der *Threshold* gesenkt werden, so dass der *Overshoot* größer wird und dadurch auch die *Gain Reduction*. Andererseits kann die *Gain Reduction* auch durch eine Erhöhung der *Ratio* oder des Eingangspegels vergrößert werden. Zwar ändert sich die Kompression in Abhängigkeit der einzelnen Parameter, häufig besteht das erfolgreiche Einstellen eines Kompressors aber aus der Kombination von mehreren Parametern. Um das zu ermöglichen, muss mit der Gewichtung nicht nur ein Parameter, sondern eine definierbare Gruppe von Parametern gesteuert werden. Dabei muss die Einstellungen der Parameter mit relativen Werten verändert werden, damit das eingestellte Verhältnis erhalten bleibt. Die Entwicklung eines solchen Konzeptes und die damit verbundenen Tests müssen in weiteren Studien ergänzt werden

Durch die fehlende Gewichtung in der prototypischen Implementierung bleibt auch die Frage offen, wie sich eine solche Gewichtung klanglich auf das Ergebnis der Bearbeitung auswirkt. In dem vorliegenden Konzept wird davon ausgegangen, dass die Bearbeitungen weniger hörbar werden, da keine sprunghaften Veränderungen vorkommen. Das konnte anhand des Prototyps nicht getestet werden.

Neben der Gewichtung von Parametern bleibt auch offen, inwieweit eine separate Einstellung der einzelnen Kompressor-Instanzen von Nutzen ist. Ein Lösungsansatz für eine Benutzeroberfläche wurde vorgeschlagen, dieser wurde jedoch nicht im Prototyp implementiert und konnte nicht getestet werden. Hinsichtlich einer einheitlichen Bearbeitung ist das separate Einstellen der einzelnen Kompressor-Instanzen vor allem dann sinnvoll, wenn die Gewichtung der Parameter mit relativen Werten erfolgt, sodass das eingestellte Verhältnis zwischen den Kompressor-Instanzen erhalten bleibt.

Während sich die vorliegende Arbeit ausschließlich auf die Bearbeitung objekt-basierter Audiokanäle beschränkt, bleibt offen, wie der Lösungsansatz auf hybride Systeme erweitert werden kann. Ein möglicher Ansatz wäre, dass die diskreten Kanäle innerhalb der Metadatenverarbeitung standardisierte Metadaten zugewiesen bekommen, sodass sie von dem Audioprozessor ebenfalls als Audio-Objekt behandelt werden können.

Die implementiere Bereichsauswahl über Polygone bietet eine maximal flexible und freie Definition der zu bearbeitenden Bereiche, doch stellt sich die Frage, ob diese Flexibilität in der Praxis überhaupt notwendig ist. Beim Mixing für 3D wird in der Regel das *interior Panning* vermieden. *Interior Panning* bedeutet, dass das Audiosignale innerhalb des Abhörbereichs gepannt werden (Thomas & Robinson, 2017).

Ein Grund der gegen ein *interior Panning* spricht, ist die Art der Umsetzung durch die Wiedergabesysteme. Für die Reproduktion von Schallquellen innerhalb des Abhörbereichs werden mehrere Lautsprecher verwendet, die teilweise nicht nebeneinander liegen, sondern sich sogar gegenüber befinden. Durch die daraus resultierenden Phasenprobleme entsteht ein inkonsistentes Klangbild im Abhörbereich. Um dieses zu vermeiden, mischen Toningenieure/-innen vorwiegend so, dass die Audiosignale sich an der Wand bzw. in der Entfernung der Lautsprecher befinden. Dadurch werden für das Erzeugen von Phantomschallquellen benachbarte Lautsprecher verwendet, wodurch ein transparentes und stabileres Ergebnis entsteht.

In Bezug auf die Bereichsauswahl würde das bedeuten, dass eine Auswahl durch Polarwinkel vollkommen ausreichend wäre, da diese alle Audio-Objekte innerhalb eines Winkels unabhängig ihrer Entfernung umfasst. Die fehlende Differenzierung zwischen der

Entfernung von Audio-Objekten ist obsolet, wenn sich in der Praxis keine Audio-Objekte innerhalb des Abhörbereichs befinden.

Abschließend bleibt die Frage der Einbindung in die Produktionsumgebung ungeklärt. Die Möglichkeiten, objekt-basierte Audioprozessoren als Plug-in oder als Stand-alone-Software zu integrieren, sind nur angerissen worden und erfordern weitere Forschung.

Während gegen die Verwendung von Plug-ins die große Prozessorlast spricht, spricht gegen eine Stand-alone-Software inklusive Player und Renderer das Problem der korrekten Reproduktion, wie die Kombination von Dolby Atmos und Spat5 innerhalb dieser Studie zeigt. Die Nutzung einer Stand-alone-Software, die zwischen DAW und Renderer geschaltet ist und die Audiosignale und Metadaten von der DAW abfängt und bearbeitet an den Renderer weiterleitet, ist am naheliegendsten, erhöht durch eine zusätzliche Software aber auch die Prozessorlast.

In der vorliegenden Arbeit konnte die Umsetzung eines objekt-basierten Mastering-Kompressors realisiert werden. Dessen Anwendung in der Praxis wurde im Rahmen dieser Arbeit weder erforscht noch evaluiert. Bei einer solchen Evaluation wäre nicht nur die Nutzerfreundlichkeit und die Anwendungsmöglichkeiten interessant, sondern auch das Verhalten in unterschiedlichen objekt-basierten und hybriden Formaten.

8 Fazit und Ausblick

Die vorliegende Arbeit ging der Frage nach wie ein objekt-basierter Mastering-Kompressor für immersive 3D-Audioformate realisiert werden kann. Aufgrund der Funktionsweise von Kompressoren, die beim Mastering verwendet werden und von objekt-basierten Audioformaten wurde ein Konzept entwickelt, dass Lösungsansätze zu einer positionsabhängigen einheitlichen Bearbeitung mehrerer Audio-Objekte vorschlägt. Diese Lösungsansätze beziehen sich vorwiegend auf dynamische Audioprozessoren mit externen Steuersignalen unter Berücksichtigung der Metadaten zur Angabe der Position.

Um das Konzept inhaltlich zu bestätigen, wurde es in Kapitel 7 mit den in Kapitel 5.3 aufgestellten Anforderungen geprüft. Da das Konzept alle Anforderungen erfüllt, ist anzunehmen, dass durch die Verwendung eines objekt-basierten dynamischen Audioprozessors nach den präsentierten Lösungsansätzen das Mastering von objekt-basierten Audioformaten effektiver und intuitiver gestaltet werden kann.

Zur technischen Validierung des entwickelten Konzepts wurde dieses in einer prototypischen Implementierung umgesetzt. Dieser Prototyp setzt dabei die Funktionen der Bereichsauswahl über Polygone für einen Kompressor-Bereich und einen *Key-Input*-Bereich, die Einzelbearbeitung sowie die einheitliche Bearbeitung mehrerer Audio-Objekte mit verschiedenen *Key-Input*-Signale um.

Der Prototyp belegt, dass das grundlegende Konzept der positionsabhängigen Audiobearbeitung und die für dynamische Audioprozessoren notwendige *Key-Input*-Signal-Summierung technisch umsetzbar und für die Bearbeitung von dreidimensionalen immersiven Musikproduktion einsetzbar ist.

Weitergehenden Forschungsbedarf gibt es vor allem bei der metadatenabhängigen Gewichtung. Dabei gilt es zu erforschen, wie die Parameter von Kompressoren gewichtet werden können, um bei einer Veränderung die ursprünglichen Einstellungen nur zu verstärken ohne dabei den Klang der Kompression zu verändern.

Auch die Verwendung für kommerzielle objekt-basierte Audioformate gilt es zu erforschen. Da diese teilweise eine Kombination aus objekt-basierten, szenen-basierten und kanal-basierten Kanälen verwenden, wird ein Lösungsansatz benötigt, der diese nicht-objekt-basierten Kanäle in der Bearbeitung berücksichtigt.

Generell muss für weitere dynamische Audioprozessoren, wie *Limiter*, dynamische Equalizer und Prozessoren für Saturation das Konzept validiert und ggf. adaptiert werden.

Am wichtigsten ist eine Evaluation zur praktischen Verwendung des Konzepts und des tatsächlichen Nutzens in einer realen Musikproduktion.

Literatur

Apple Inc. (2021a, 17. Mai 2021). *Apple Music kündigt 3D-Audio mit Dolby Atmos an; gesamter Katalog in Lossless Audio [Pressemitteilung]* https://www.apple.com/de/newsroom/2021/05/apple-music-announces-spatial-audio-and-lossless-audio/

Apple Inc. (2021b, 18. Oktober). *Final Cut Pro und Logic Pro erhalten Updates mit leistungsstarken neuen Funktionen und beispielloser Performance auf dem neuen MacBook Pro mit M1 Pro und M1 Max [Pressemitteilung]* https://www.apple.com/de/newsroom/2021/10/final-cut-pro-and-logic-pro-updated-on-the-new-macbook-pro-with-m1-pro-m1-max/

Auro Technologies. (2015). *AURO-3D® HOME THEATER SETUP Installation Guide*. Auro Technologies. Abgerufen am 21. August 2022 von https://www.auro-3d.com/wp-content/uploads/documents/Auro-3D-Home-Theater-Setup-Guidelines_lores.pdf

Boren, B. (2017). History of 3D sound : the art and science of binaural and multi-channel audio. In Immersive sound (S. 40-62). Routledge.

Braddock, J.-P., Hepworth-Sawyer, R., Hodgson, J., Shelvock, M., & Toulson, R. (2021). *Mastering in Music* (1. ed.). Routledge, Taylor & Francis Group.

Bresler, Z. (2021). Immersed in Pop: 3D Music, Subject Positioning, and Compositional Design in The Weeknd's "Blinding Lights" in Dolby Atmos. *Journal of Popular Music Studies*, *33*(3), 84-103. https://doi.org/10.1525/jpms.2021.33.3.84

Buff, H.-M. (2020). *Überall – Musikproduktion in 3D-Audio für Kopfhörer*. Ebner Media Group GmbH & Co. KG.

Cipriani, A., & Giri, M. (2019). *Electronic Music and Sound Design - Max 8* (Vol. 2). Contemponet s.a.s.

Cousins, M., & Hepworth-Sawyer, R. (2013). *Practical mastering: a guide to mastering in the modern studio*. Focal Press.

Cycling '74. (n.d.-a). *Basic Javascript programming for the js and jsui objects*. Cycling '74. Abgerufen am 29. August 2022 von https://docs.cycling74.com/max8/vignettes/jsintro

Cycling ‘74. (n.d.-b). *What is Max?* Cycling ‘74. Abgerufen am 29. August 2022 von https://cycling74.com/products/max

Dickreiter, M., Dittel, V., Hoeg, W., & Wöhr, M. (2014). *Handbuch der Tonstudiotechnik* (8. ed.). De Gruyter Saur. https://doi.org/doi:10.1515/9783110316506

Dolby Laboratories Inc. (2021a). *What is Dolby Atmos?* Dolby Laboratories Inc., Dolby. Abgerufen am 29. August 2022 von https://professionalsupport.dolby.com/s/article/What-is-Dolby-Atmos?language=en_US

Dolby Laboratories Inc. (2021b). *What is the Dolby Atmos Renderer?* Dolby Laboratories Inc., Dolby. Abgerufen am 29.08.2022 von https://professionalsupport.dolby.com/s/article/What-is-the-Dolby-Atmos-Renderer?language=en_US

EBU. (2014). Audio Definition Model – Metadata Specification. In *TECH 3364*. Genf: European Broadcasting Union.

Friesecke, A. (2014). *Die Audio-Enzyklopädie: Ein Nachschlagewerk für Tontechniker* (2 ed.). De Gruyter Saur. https://doi.org/doi:10.1515/9783110340181

Hamasaki, K., Hiyama, K., & Okumura, R. (2005, 28. – 31. Mai). *The 22.2 Multichannel Sound System and Its Application* AES 118th Convention, Barcelona. http://www.aes.org/e-lib/browse.cfm?elib=13122

Hestermann, S., Seideneck, M., & Sladeczek, C. (2018, 06. – 09. August). An Approach for Mastering Audio Objects. Conference on Spatial Reproduction, Tokyo, Japan.

Hiller, R. S., & Walter, J. M. (2017). The Rise of Streaming Music and Implications for Music Production. *Review of Network Economics*, *16*(4), 351-385. https://doi.org/doi:10.1515/rne-2017-0064

Inc., D. L. (2020, 28. Mai 2020). *TIDAL and Dolby Bring Dolby Atmos Music to the Home [Pressemitteilung]* https://news.dolby.com/en-WW/189109-tidal-and-dolby-bring-dolby-atmos-music-to-the-home

International Telecommunication Union. (2012). ITU-R BS.775-3. In *Multichannel stereophonic sound system with and without accompanying picture*.

Ircam. (2022). *Spatialisateur – spat5*. In (Version 5.2) STMS Lab (UMR 9912), Ircam – CNRS – Sorbonne Université. https://forum.ircam.fr/projects/detail/spat/

Ircam Forum. (n.d.). *Spat*. Ircam Forum. Abgerufen am 29.08.2022 von https://forum.ircam.fr/projects/detail/spat/

Izhaki, R. (2008). *Mixing audio : concepts, practices and tools*. Focal Press.

Katz, R. A. (2010). *Mastering Audio : Über die Kunst und die Technik* (3. ed.). GC Carstensen Verlag.

Kim, S. (2017). Height channels. In *Immersive Sound : the art and science of binaural and multi-channel audio* (S. 221-243). Routledge.

Lawrence, R. (2019). Producing Music for Immersive Audio Experiences. In *Producing Music* (S. 134-155). Routledge.

Melchior, F., Michaelis, U., & Steffens, R. (2011, 31. July - 05. August). Spatial Mastering-a new concept for spatial sound design in object-based audio scenes. International Computer Music Conference, University of Huddersfield, UK.

Merging Technologies SA. (2022, Januar). *Merging Launches Pyramix 14 [Pressemitteilung]* https://merging.com/news/press_releases/merging-launches-pyramix-14

Noltemeyer, S. (2012). *Mastering*. PPV Medien. http://d-nb.info/1022876937/04

Oramus, T., & Neubauer, P. (2019, 16. – 19. Oktober). Comparison Study of Listeners' Perception of 5.1 and Dolby Atmos. AES 147th Convention, New York.

Oramus, T., & Neubauer, P. (2020, 02. – 05. Juni). Comparison of Perception of Spatial Localization Between Channel and Object Based Audio. AES 148th Convention, Wien.

Owsinski, B. (2009). *Mastern wie die Profis : Das Handbuch für Toningenieure* (T. L. Agency, Trans.). Carstensen.

Pulkki, V. (2001). *Spatial Sound Generation and Perception by Amplitude Panning Techniques* Helsinki University of Technology. Helsinki. http://lib.tkk.fi/Diss/2001/isbn9512255324/

Roginska, A., & Geluso, P. (2017). *Immersive Sound : the art and science of binaural and multi-channel audio*. Routledge.

Rumsey, F. (1999). Controlled Subjective Assessments of Two-to-Five-Channel Surround Sound Processing Algorithms. *Journal of the Audio Engineering Society*, *47*(7/8), 563-582. http://www.aes.org/e-lib/browse.cfm?elib=12099

Rumsey, F. (2013). *Spatial audio*. Focal Press.

Self, D. (2015). *Small Signal Audio Desing* (2 ed.). Focal Press.

Sony Electronics Inc. (2019, 15 Oktober). *Sony Corporation, Music Industry Partners and Leading Artists Gather to Reveal a New Music Ecosystem with 360 Reality Audio [Pressemitteilung]* https://www.sony.com/content/sony/en/en_us/SCA/company-news/press-releases/sony-electronics/2019/sony-corporation-music-industry-partners-and-leading-artists-gather-to-reveal-a-new-music-ecosystem-with-360-reality-audio.html

Sony Electronics Inc. (2021, 19. Oktober). *360 Reality Audio Now Available on Amazon Music Unlimited with Any Headphones [Pressemitteilung]* https://www.sony.com/content/sony/en/en_us/SCA/company-news/press-releases/sony-electronics/2021/360-reality-audio-now-available-on-amazon-music-unlimited-with-any-headphones.html

Steinberg Media Technologies GmbH. (2022, 02. März). *Cubase 12: Nahtlose Integration und reibungslose Performance [Pressemitteilung]* https://ocl-steinberg-live.steinberg.net/_storage/asset/164428/storage/master/Pressemitteilung%20-%202022-03-02%20-%20Cubase%2012%20-%20DE.pdf

Stone, M., & Moore, B. (2003). Effect of the speed of a single-channel dynamic range compressor on intelligibility in a competing speech task. *The Journal of the Acoustical Society of America*, *114*(2), 1023-1034. https://doi.org/10.1121/1.1592160

Taylor, R. W. (2017). Hyper-compression in Music Production; Agency, Structure and the Myth that ‘Louder is Better’. *Journal on the Art of Record Production*(11).

Thomas, M. R., & Robinson, C. Q. (2017, 8. Oktober). *Amplitude Panning and the Interior Pan* AES 143rd Convention, New York, NY.

Torick, E. (1998). Highlights in the History of Multichannel Sound. *Journal of the Audio Engineering Society*, *46*, 27-31. http://www.aes.org/e-lib/browse.cfm?elib=12177

Tsingos, N. (2017). Object-Based Audio. In *Immersive Sound : the art and science of binaural and multi-channel audio* (S. 244-275). Routledge.

vlasky. (n.d.). point_in_polygon_using_winding_number.js. *GitHub*. https://gist.github.com/vlasky/d0d1d97af30af3191fc214beaf379acc

Weinzierl, S. (2008). *Handbuch der Audiotechnik* (1 ed.). Springer Berlin, Heidelberg. https://doi.org/https://doi.org/10.1007/978-3-540-34301-1

Zielinski, S. K., Rumsey, F., & Bech, S. (2003). Effects of Down-Mix Algorithms on Quality of Surround Sound. *Journal of the Audio Engineering Society*, *51*(9), 780-798. http://www.aes.org/e-lib/browse.cfm?elib=12208

Anhang A

A1 JavaScript Function: test

```
function test(p1, p2, p3, p4, object) {
    var inOut;

    if (objectMatchPoint(p1, p2, p3, p4, object) == true){
        inOut = true;
    } else if (isOnBorder(object, p1, p2, p3, p4) == true){
        inOut = true;
    } else {
        inOut = pointInPolygon(object, polygon);
    }

    outlet(0, inOut);
}
```

Anmerkung: Die Funktion ruft verschiedene Funktionen auf, um zu testen, ob das Audio-Objekt innerhalb des Polygons liegt.

A2 JavaScript Function: objectMatchPoint

```
function objectMatchPoint(p1, p2, p3, p4, object) {

    if (object[0] == p1[0]) {
        if (object[1] == p1[1]) {
            return true;
        }
    } else if (object[0] == p2[0]) {
        if (object[1] == p2[1]) {
            return true;
        }
    } else if (object[0] == p3[0]) {
        if (object[1] == p3[1]) {
            return true;
        }
    } else if (object[0] == p4[0]) {
        if (object[1] == p4[1]) {
            return true;
        }
    }
}
```

Anmerkung: Die Funktion testet, ob ein Audio-Objekt auf einem Polygonpunkt liegt.

A3 JavaScript: Function isOnBorder

```
function isOnBorder(object, p1, p2, p3, p4) {

    if (object[0] <= Math.max(p1[0], p2[0]) &&
        object[0] >= Math.min(p1[0], p2[0]) &&
        object[1] <= Math.max(p1[1], p2[1]) &&
        object[1] >= Math.min(p1[1], p2[1])) {
        return true;
    } else
    if (object[0] <= Math.max(p2[0], p3[0]) &&
        object[0] >= Math.min(p2[0], p3[0]) &&
        object[1] <= Math.max(p2[1], p3[1]) &&
        object[1] >= Math.min(p2[1], p3[1])) {
        return true;
    } else
    if (object[0] <= Math.max(p3[0], p4[0]) &&
        object[0] >= Math.min(p3[0], p4[0]) &&
        object[1] <= Math.max(p3[1], p4[1]) &&
        object[1] >= Math.min(p3[1], p4[1])) {
        return true;
    } else
    if (object[0] <= Math.max(p4[0], p1[0]) &&
        object[0] >= Math.min(p4[0], p1[0]) &&
        object[1] <= Math.max(p4[1], p1[1]) &&
        object[1] >= Math.min(p4[1], p1[1])) {
        return true;
    } else {
        return false;
    }
}
```

Anmerkung: Die Funktion testet, ob das Audio-Objekt auf einer Strecke zwischen zwei Polygonpunkten liegt.

A4 JavaScript: Function pointInPolygon

```
//JavaScript implementation of winding number algorithm to
determine whether a point is inside a polygon by vlasky
(https://gist.github.com/vlasky/d0d1d97af30af3191fc214beaf379
acc)
//Based on C++ implementation of wn_PnPoly() published on
http://geomalgorithms.com/a03-_inclusion.html

function pointInPolygon(point, vs) {
    var x = point[0], y = point[1];
    var wn = 0;

    for (var i = 0, j = vs.length - 1; i < vs.length; j =
i++) {
        var xi = vs[i][0], yi = vs[i][1];
        var xj = vs[j][0], yj = vs[j][1];

        if (yj <= y) {
            if (yi > y) {
                if (isLeft([xj, yj], [xi, yi], [x, y]) > 0) {
                    wn++;
                }
            }
        } else {
            if (yi <= y) {
                if (isLeft([xj, yj], [xi, yi], [x, y]) < 0) {
                    wn--;
                }
            }
        }
    }
    return wn != 0;
};

function isLeft(P0, P1, P2) {
    var res = ((P1[0] - P0[0]) * (P2[1] - P0[1])
        - (P2[0] - P0[0]) * (P1[1] - P0[1]));
    return res;
}
```

Anmerkung: Die Funktion testet, ob ein Audio-Objekt innerhalb des Polygons liegt. Code übernommen von vlasky (n.d.)

Anhang B

Anmerkung:

Anhang B ist der digitale Anhang. Jeder gedruckten Arbeit liegt ein USB-Stick mit dem Folgenden Inhalt bei:

B1 Digitaler Anhang

/00_Bachelorarbeit Ernst Benedikt

Prototypische Implementierung eines objekt-basierten Mastering-Kompressors für 3D-Musikproduktion.pdf

/01_Prototyp: objekt-basierter Mastering-Kompressor

Object-based_Compressor.mxf

/02_Beispieldateien für die Verwendung im Prototyp

Unbearbeitete ADM-Dateien der Audiobeispiele (siehe /03_Audiobeispiele

/03_Audiobeispiele

Anmerkung: Die Audiobeispiele liegen in den Formaten Dolby Atmos ADM, Binaural, 5.1 Disket und 7.1.4 Diskret vor. Die Beschreibungen zu den Dolby Atmos ADM Beispielen gelten für die Beispiele aller Formate.

/00_Dolby Atmos ADM

Audiobeispiel_01_Einzelbearbeitung.wav

Beschreibung: Die beiden Audio-Objekte bewegen sich auf der rechten (Schlagzeug) bzw. auf der linken (Synthesizer) vor und zurück. In der vorderen Hälfte der Audioszene werden sie separat komprimiert.

Audiobeispiel_02a_Einzelbearbeitung.wav

Beschreibung: Zum Vergleich mit Audiobeispiel_02b ist Audiobeispiel_02a vollständig unbearbeitet.

Audiobeispiel_02b_Einzelbearbeitung.wav

Beschreibung: Die Audio-Objekte werden in der vorderen Hälfte der Audioszene komprimiert.

Audiobeispiel_03_interne_SC_Summe.wav

Beschreibung: Die Audio-Objekte bewegen sich gegenläufig vor und zurück. In der Mitte der Audioszene werden sie gemeinsam komprimiert.

Audiobeispiel_04a_interne_SC_Summe.wav

Beschreibung: Zum Vergleich mit Audiobeispiel_04b ist Audiobeispiel_04a vollständig unbearbeitet.

Audiobeispiel_04b_interne_SC_Summe.wav

Beschreibung: Die vordere Hälfte der Audioszene wird von der Summe aller Audio-Objekte komprimiert, die sich in der vorderen Hälfte befinden. Dies ist deutlich zu hören, wenn das Klavier von der hinteren Hälfte in die vordere Hälfte wechselt.

Audiobeispiel_05_externe_SC_Summe.wav

Beschreibung: Der Synthesizer in der linken vorderen Hälfte der Audioszene wird von einem externen Signal komprimiert. Das Schlagzeug bewegt sich vor und zurück. In der hinteren linken Hälfte der Audioszene liegt es im *Key-Input*-Bereich und ist das *Key-Input*-Signal für den Kompressor auf dem Synthesizer.

Audiobeispiel_06a_externe_SC_Summe.wav

Beschreibung: Zum Vergleich mit Audiobeispiel_06b ist Audiobeispiel_06a vollständig unbearbeitet.

Audiobeispiel_06b_externe_SC_Summe.wav

Beschreibung: Die vordere Hälfte der Audioszene wird von der hinteren Hälfte komprimiert. Dies ist deutlich zu hören, wenn die kreisende Bassdrum die hintere Hälfte durchquert.

/01_Binaural

Audiobeispiel_01_Einzelbearbeitung_BIN.wav

Audiobeispiel_02a_Einzelbearbeitung_BIN.wav

Audiobeispiel_02b_Einzelbearbeitung_BIN.wav

Audiobeispiel_03_interne_SC_Summe_BIN.wav

Audiobeispiel_04a_interne_SC_Summe_BIN.wav

Audiobeispiel_04b_interne_SC_Summe_BIN.wav

Audiobeispiel_05_externe_SC_Summe_BIN.wav

Audiobeispiel_06a_externe_SC_Summe_BIN.wav

Audiobeispiel_06b_externe_SC_Summe_BIN.wav

/02_5.1 Diskret

Audiobeispiel_01_Einzelbearbeitung_5.1.wav

Audiobeispiel_02a_Einzelbearbeitung_5.1.wav

Audiobeispiel_02b_Einzelbearbeitung_5.1.wav

Audiobeispiel_03_interne_SC_Summe_5.1.wav

Audiobeispiel_04a_interne_SC_Summe_5.1.wav

Audiobeispiel_04b_interne_SC_Summe_5.1.wav

Audiobeispiel_05_externe_SC_Summe_5.1.wav

Audiobeispiel_06a_externe_SC_Summe_5.1.wav

Audiobeispiel_06b_externe_SC_Summe_5.1.wav

/03_7.1.4 Diskret

Audiobeispiel_01_Einzelbearbeitung_7.1.4.wav

Audiobeispiel_02a_Einzelbearbeitung_7.1.4.wav

Audiobeispiel_02b_Einzelbearbeitung_7.1.4.wav

Audiobeispiel_03_interne_SC_Summe_7.1.4.wav

Audiobeispiel_04a_interne_SC_Summe_7.1.4.wav

Audiobeispiel_04b_interne_SC_Summe_7.1.4.wav

Audiobeispiel_05_externe_SC_Summe_7.1.4.wav

Audiobeispiel_06a_externe_SC_Summe_7.1.4.wav

Audiobeispiel_06b_externe_SC_Summe_7.1.4.wav