VISUS

Bachelorarbeit Nr. 0142

Molekulare Geräusche: Audio-visuelle Darstellung molekularer Simulationen

Benjamin Rau

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Dipl.-Inf. Michael Krone Dipl.-Inf. Christoph Müller
Beginn am:	9. Mai 2014
Beendet am:	8. August 2014
CR-Nummer:	H.5.2, I.3.8

Kurzfassung

Die Simulation von molekularen Vorgängen nimmt als wissenschaftliches Hilfsmittel immer mehr an Bedeutung zu. Infolgedessen nehmen auch die Datenmengen zu, die analysiert werden müssen. Neben der visuellen Analyse, bietet die Sonifikation die Möglichkeit, die Daten mit Hilfe von Klangereignissen zu vertonen.
Da die Sonifikation noch nicht für die Analyse von molekularen Simulationen verwendet wurde, wird in dieser Arbeit untersucht, inwieweit sich diese für das Gebiet der molekularen Simulationen eignet. Dazu wird eine Unterstützung, zur Vertonung verschiedener Ereignisse, für das MegaMol-Visualisierungstool implementiert. Mit dieser Implementierung wird eine einfache Möglichkeit geschaffen, verschiedenste Ereignisse in den Simulationsdaten automatisch zu erkennen, und diese Ereignisse, durch unterschiedliche Töne, anzuzeigen. Zur besseren räumlichen Einordnung dieser Ereignisse wird zudem eine Unterstützung von 3D-Sound implementiert.
Weiterhin werden verschiedene Einsatzszenarien beschrieben, sowie eine konkrete Eventerkennung implementiert und abschließend die dabei gewonnen Erkenntnisse vorgestellt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Listings

1. Einleitung

Die Simulation von molekularen Daten nimmt, genau so wie Simulationen allgemein, eine immer größere Rolle in der Forschung ein. Neben dem klassischen Methoden des Experiments und der Theorie ist sie eine weitere Möglichkeit, wissenschaftliche Erkenntnis zu erlangen. Da immer mehr molekulardynamische Simulationen durchgeführt werden und auch deren Komplexität steigt, werden auch immer mehr Daten produziert, die analysiert werden müssen. Normalerweise geschieht dies unter Zuhilfenahme von Visualisierungstechniken. Dazu wird eine geeignete graphische Repräsentation der zeitlichen Entwicklung der simulierten Moleküle gezeigt. Fortgeschrittene Visualisierungen, wie das am Visualisierungsinstitut der Universität Stuttgart entwickelte *MegaMol* [GKM+14] heben zusätzlich spezielle Eigenschaften oder Ereignisse grafisch hervor, welche aus den Simulationsdaten extrahiert wurden. Da die Informationsmenge enorm ist, mit der der Benutzer bei einer solchen Visualisierung konfrontiert wird, stellt sich die Frage, wie der Benutzer auf besondere Ereignisse aufmerksam gemacht werden kann. In dieser Arbeit soll deshalb untersucht werden, inwieweit sich die grafische Darstellung durch Geräusche unterstützen lässt, welche spezielle Ereignisse in der Simulation signalisieren sollen. Dazu soll *MegaMol* um eine entsprechende Möglichkeit erweitert werden, bestimmte Ereignisse zu erkennen und die Aufmerksamkeit des Benutzers durch Wiedergabe von Audiodateien auf dieses Ereignis zu lenken. Die daraus resultierende audio-visuelle Darstellung soll es dem Benutzer ermöglichen, auch das Gehör für die Analyse zu verwenden. Eine Erkennung dieser Ereignisse soll sowohl live während der Visualisierung stattfinden können oder bei komplexeren Ereignissen durch eine Vorberechnung geschehen, welche dann während der Visualisierung zu den richtigen Zeitpunkten die entsprechenden Klänge erzeugt. Da die Daten räumlich vorliegen, soll dabei auch geprüft werden ob, der Einsatz von räumlichen Tönen einen Vorteil bringt und in wie weit sich einzelne Ereignisse noch unterscheiden lassen können.

Gliederung

Die Arbeit ist in folgender Weise gegliedert:

Kapitel 2 – Auditive Wahrnehmung: Hier werden die Grundlagen des menschlichen Gehörs beschrieben und erläutert, welche Toneigenschaften der Mensch wie unterscheiden kann.

Kapitel 3 – Sonifikation: Gibt einen Einblick in die Vertonung von Daten.

Kapitel 4 – 3D-Sound: Da bei dieser Arbeit die Sonifikation von räumlichen Daten getestet wird, gibt dieses Kapitel einen Überblick über die technische Umsetzung und Vorraussetzungen.

Kapitel 5 – Verwandte Arbeiten: Beinhaltet einen Überblick über ähnliche Forschungsansätze.

Kapitel 6 – Implementierung: Erklärt die Vorgehensweise bei der Implementierung

Kapitel 7 – Eventerkennung Beschreibung der Wasserstoffbrücken-Eventerkennung und mögliche weitere Events.

Kapitel 8 – Ergebnisse Diskutiert die Ergebnisse der Implementierung.

Kapitel 9 – Zusammenfassung und Ausblick Dieses Kapitel fasst die Ergebnisse der Arbeit zusammen und enthält zudem einige Verbesserungsvorschläge.

2. Auditive Wahrnehmung

Neben dem Seh- und Tastsinn ist der Hörsinn einer der wichtigsten Sinne eines Menschen. Das hochkomplexe Sinnesorgan Ohr hilft uns dabei, uns in unserer Umwelt zurechtzufinden und ermöglicht die Kommunikation untereinander. In diesem Abschnitt soll die Fähigkeit des menschlichen Gehörs erläutert und die Rolle des Hörens in der menschlichen Wahrnehmung beschrieben werden. Als auditive Wahrnehmung bezeichnet man dabei die Fähigkeit, Schallwellen wahrzunehmen. Sie besitzt sie einige Besonderheiten, die sie von anderen Wahrnehmungsarten abhebt. So ist es im Gegensatz zum Sehen, bei dem einfach die Augen geschlossen werden können, nicht möglich, die Ohren zu verschließen. In Folge dessen nehmen wir ständig alle Geräusche in unserer Umgebung wahr. Das Filtern dieser Datenflut findet erst in unserem Gehirn statt und ermöglicht es, uns auf wichtige Geräusche zu konzentrieren, während nebensächliche Geräusche im Hintergrund ausgeblendet werden können.

2.1. Welche Merkmale kann das Gehör unterscheiden

Das menschliche Gehör ist in der Lage eine Vielzahl unterschiedlicher Eigenschaften des wahrgenommenen Schalls zu unterscheiden. Eine visuelle Darstellung dieser Eigenschaften findet sich auch in Abbildung 2.4. Über diese Eigenschaften sowie über die Grenzen, die dem Gehör dabei gesetzt sind, soll im folgenden ein Überblick gegeben werden.

2.1.1. Grundfrequenz

Die Tonhöhe wird durch die Frequenz des Schalls bestimmt. Der Bereich, den der Mensch wahrnehmen kann, liegt zwischen 20Hz und 20 000Hz [RH11], wobei im Frequenzraum von 2000Hz bis 5000Hz am sensitivsten auf Veränderungen reagiert wird. Auch die individuellen Fähigkeiten schwanken von Person zu Person und die Frequenzspanne, die ein Mensch wahrnehmen kann, sinkt mit zunehmendem Alter. Änderungen in der Tonfrequenz können dabei in einer sehr hohen Auflösung wahrgenommen werden. So ist der Mensch in der Lage bis zu 1500 Änderungen im hörbaren Spektrum der Frequenzen wahrnehmen. Zu der Fähigkeit des Gehörs unterschiedliche Frequenzen zu unterscheiden, gibt es im Grunde zwei Theorien. Zum einen die *Resonanz* oder *Place Theory* [Gul71], die davon ausgeht, dass unterschiedliche Frequenzen verschiedene Teile der Hörschnecke stärker anregen (siehe Abb. 2.1) als andere. Nave [Nav99] gibt jedoch an, dass ein Erkennen der Frequenz allein durch diesen Aufbau physikalisch unlogisch ist. Die *Frequenz Therorie* geht hingegen davon aus, dass die unterschiedlichen Frequenzen durch zeitliche Auflösung wahrgenommen werden. Kunchur [Kun08] hat durch Experimente herausgefunden, dass die zeitliche Auflösung mit 5 Mikrosekunden um einiges niedriger ist als bisher angenommen und damit um einiges niedriger als für die Wahrnehmung von Frequenzen bis 20kHz benötigt wird. Manell [Man08] fasst die Schwächen der beiden Theorien zusammen und

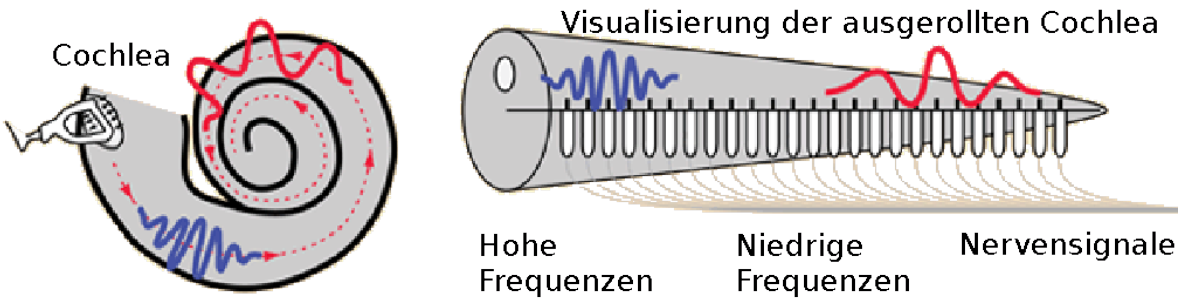
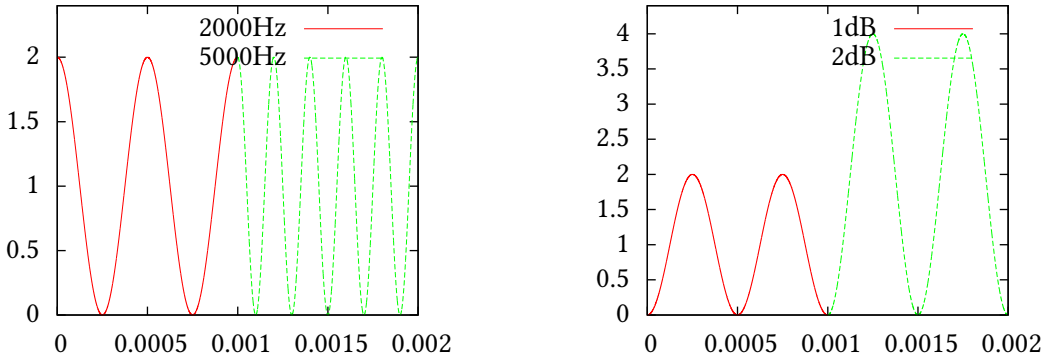


Abbildung 2.1.: Verschiedene Frequenzen stimulieren die Hörmuscheln an unterschiedlichen Stellen. Durch diese Eigenschaft kann die Frequenz eines gehörten Tones bestimmt werden kann. [Nav99]



(a) **(b)**

Abbildung 2.2.: (a) Frequenzunterschied zwischen 2000Hz und 5000Hz, also jenen Frequenzen zwischen denen der Mensch am besten hört. (b) Unterschiedliche Schalldruckpegel. Dezibel gibt die mittlere Auslenkung an.

geht davon aus, dass beide Mechanismen zusammenarbeiten um die hohe Auflösung der Frequenzen zu ermöglichen.

2.1.2. Lautstärke

Auch der Schalldruck kann über einen breiten Bereich wahrgenommen werden. Der wahrgenommene Schalldruck wird oft in einem logarithmischen Maß als Schalldruckpegel in Dezibel (dB) angegeben. Als Bezugswert wird dabei $p_0 = 20\mu Pa$ festgelegt. Der Schalldruckpegel L_p lässt sich somit mit Hilfe der Formel $L_p = 20log_{10}\left(\frac{\tilde{p}}{p_0}\right)$ aus dem Effektivwert des Schalldrucks $\tilde{p}$ berechnen. Die Wahrnehmung der Lautstärke ist nicht absolut, sondern hängt von der Frequenz des Tones ab (siehe Abb. 2.3). Dabei hat jedoch nicht nur die Frequenz, sondern auch das Alter einen Einfluss auf die wahrnehmbaren Lautstärken. Vor allem bei höheren Frequenzen steigt die Wahrnehmungsgrenze im Durchschnitt

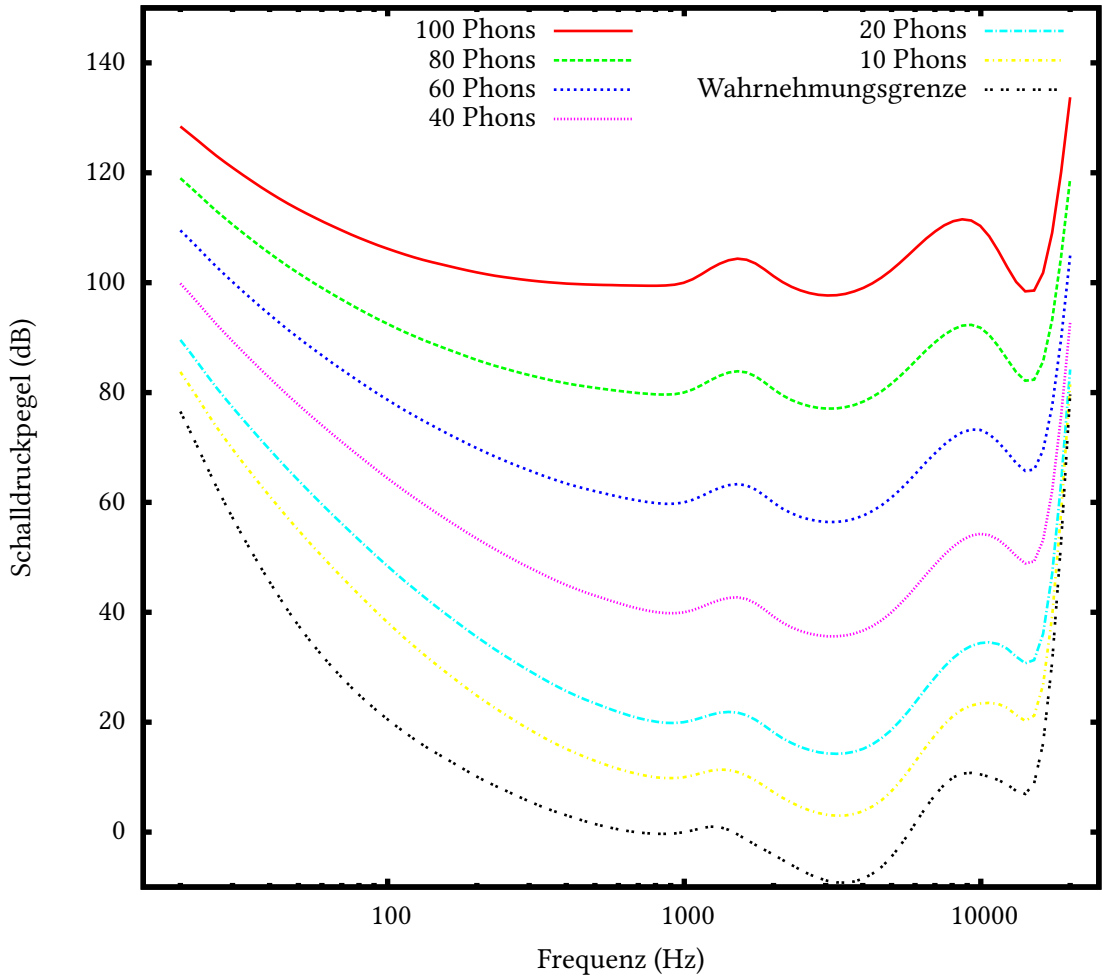


Abbildung 2.3.: ISO 226: Lautstärken die für verschiedene Frequenzen als gleich laut wahrgenommen werden befinden sich auf einer Linie.

um 15dB an. Umgekehrt hat auch die Lautstärke einen Einfluss auf die wahrgenommene Frequenz. So werden tiefe Töne mit einer Frequenz < 4000Hz mit zunehmender Lautstärke als immer tiefer werdend wahrgenommen, während bei hohen Tönen mit Frequenzen > 4000 Hz die wahrgenommene Frequenz mit der Lautstärke ansteigt. Eine angepasste Maßeinheit, die beschreibt wie laut ein Ton wahrgenommen wird, ist das sogenannte Phon (siehe Abb. 2.3).

2.1.3. Klangfarbe

Die Klangfarbe oder auch das Timbre beschreibt im Grunde das Zusammenspiel verschiedener physikalischer Eigenschaften eines Tones (siehe Abb. 2.4). Es gibt verschiedene Definitionen, was genau unter Timbre zu verstehen ist, wobei sich die früheste Definition laut Muzzulini [Muz06] bei Rousseaus finden lässt. Dieser beschreibt Timbre in seinem Eintrag zur *Encylopédia* 1749 als eine Eigenschaft

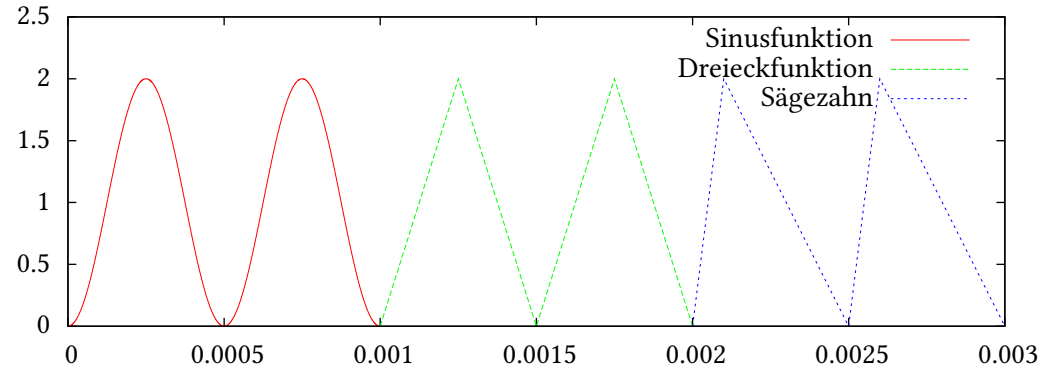
des Tones, die diesen sauer oder süß, dumpf oder hell, trocken oder weich erscheinen lässt. Das American National Standards Institute definierte Timbre im Jahr 1960 als all jene Klangeigenschaften, die nicht direkt die Tonhöhe oder die Lautstärke beeinflussen. Zu diesen Eigenschaften werden die spektrale Leistungsverteilung des Tons, dessen Hüllkurve, die Häufigkeit und Stärke der Amplituden- und Frequenzmodulationen sowie der Grad der inharmonischen Teiltöne gezählt. Die spektrale Leistungsverteilung beschreibt wie sich das Spektrum aus Grundschwingung, harmonischen Anteilen und Rauschanteilen zusammensetzt. Zusammen mit der Hüllkurve macht diese Eigenschaft den Großteil des Klangbildes aus. Die Hüllkurve beschreibt die zeitliche Entwicklung des Tones und kann idealisiert durch ADSR-Zeiten angegeben werden [PTP09]. ADSR steht hierbei für **A**ttack (Anstiegszeit), **D**ecay (Abfall), **S**ustain (Halten des Tones), **R**elease (Freigeben) (siehe Abb. 2.4b).

Zusammenfassend kann gesagt werden, dass die Klangfarbe viele unterscheidbare Merkmale liefert, die es beispielsweise erlauben, zwischen verschiedenen Instrumenten, die die gleichen Noten spielen, oder auch verschiedenen Umgebungen, in denen der Ton abgespielt wird (z.B. geschlossener Raum oder Wald), zu unterscheiden. Eine genaue Quantifizierung der Klangfarbe ist laut Giovanni [DP02] jedoch schwierig. Verschiedene subjektive Eindrücke wie stumpf/brillant, kalt/warm, klar/voll, gedämpft/scharf und kompakt/verstreut lassen sich zwar einfach durch Skalare beschreiben aber eine Übertragung auf die physikalischen Eigenschaften des Tons stellt sich oftmals als schwierig heraus.

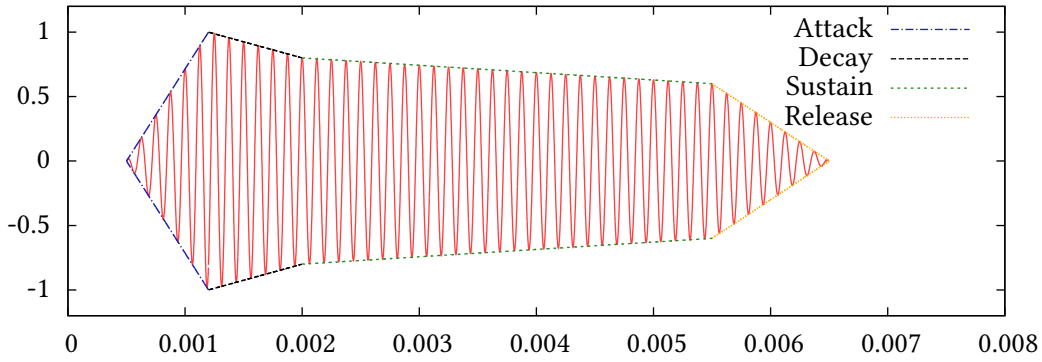
2.1.4. Position im Raum

Sowohl die Richtung, aus der ein Geräusch wahrgenommen wird als auch die Entfernung oder Richtung, in der sich die Quelle des Geräusches bewegt, können vom Menschen wahrgenommen werden. Diese Fähigkeit erlaubt unter anderem das Fokussieren auf einen Sprecher in einem lauten Raum, allgemein hin auch als „Cocktailparty-Phänomen "[Che53] bekannt. Die Lokalisation wird durch verschiedene physikalische Eigenschaften ermöglicht und durch physiologische Eigenschaften der Ohren erleichtert, über die im folgenden ein kurzer Überblick gegeben werden soll.

Schon 1907 veröffentlicht Lord Rayleigh eine Theorie, wie die Lokalisation von Audioquellen auf lateralen Ebene, also die Unterscheidung von links, vorne und rechts, möglich ist [Ray07]. In seiner sogenannten „Duplex Therorie“ gibt Rayleigh zwei Faktoren an, die diese Richtungsbestimmung ermöglichen. Diese sind zum einen die Zeitdifferenzen „Interaural Time Difference (ITD)“ (siehe Abb. 2.5a), die wahrgenommen werden, wenn der Schall seitlich auf den Kopf trifft (siehe Abb. 2.6a), und zum anderen die Lautstärkeunterschiede „Interaural Intensity Difference (IID)“ (siehe Abb. 2.5c), die durch Verschattung durch den Kopf hervorgerufen werden. Die Laufzeitdifferenzen treten immer dann auf, wenn sich die Audioquelle nicht direkt vor oder hinter dem Kopf befindet sondern versetzt dazu. Bei einer maximalen Laufwegdifferenz von $22,5cm$ und einer Schallgeschwindigkeit in der Luft von $343\frac{m}{s}$ erhält man eine maximale Laufzeitdifferenz von $659\mu s$, welche deutlich über der zeitlichen Auflösung von $5\mu s$ [Kun08] liegt. Da für periodische Töne gilt, dass diese nur bei Frequenzen, deren Periodendauer mehr als die Hälfte der maximalen Laufzeitdifferenz beträgt, eindeutig unterschieden werden können, folgt daraus, dass die Interaural Time Difference vor allem bei Frequenzen kleiner als 1,5kHz eine Rolle spielt. Im Gegensatz dazu greift die Iteraural Intensity Difference hauptsächlich bei Frequenzen größer als 1,5kHz, da bei niedrigeren Frequenzen eine stärkere Beugung der Schallwellen stattfindet und die Verschattung durch den Kopf deshalb keine große Rolle spielt. Bei Tönen, die



(a)

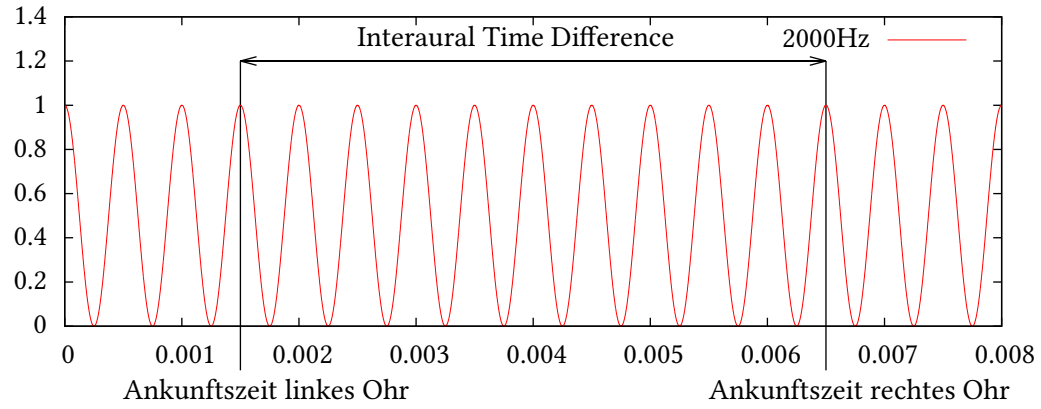


(b)

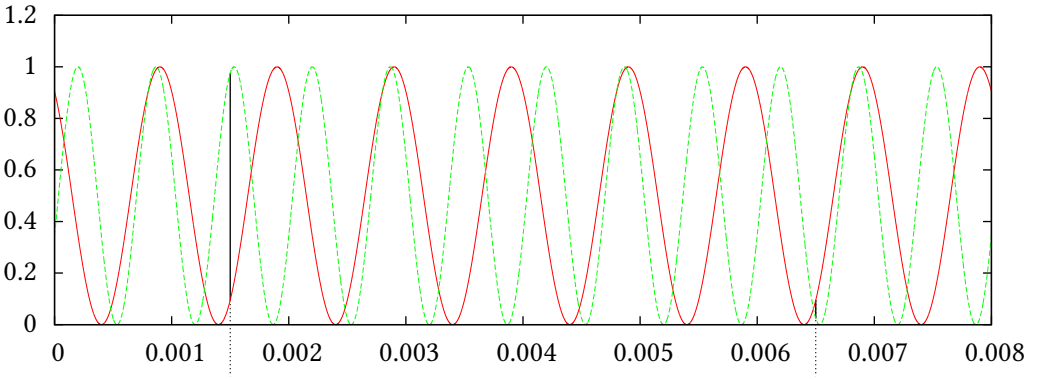
Abbildung 2.4.: Verschiedene grundlegende Eigenschaften von Sound die der Mensch unterscheiden kann. Unter [Wil12] können die Auswirkungen einiger Parameter auf den Sound ausprobiert werden.

aus vielen Frequenzen zusammengesetzt sind, spielt auch die Phasenverschiebung „Interaural Phase Difference (IPD) “ (siehe Abb. 2.5b) der Grundfrequenzen zueinander eine Rolle und ermöglicht eine noch genauere Auflösung als das bei einfachen Sinustönen möglich ist [Hir48].

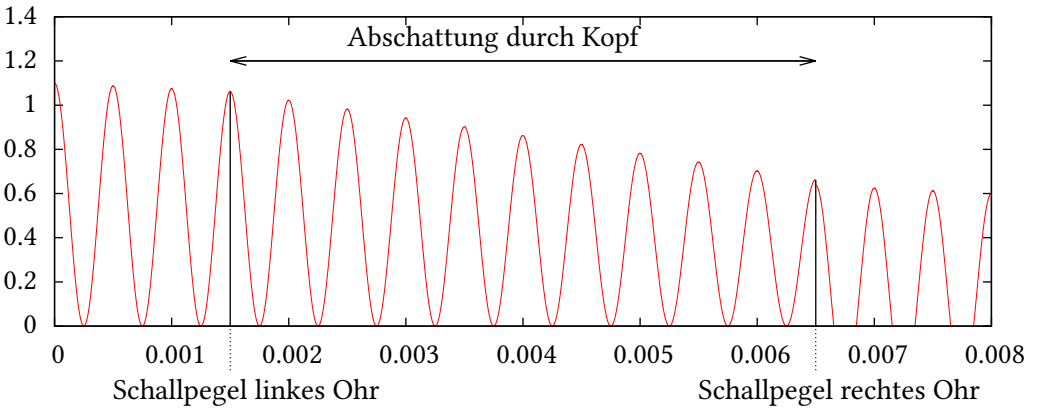
Diese Phänomene reichen jedoch nicht aus um zwischen vorne/hinten und oben/unten zu unterscheiden. Audioquellen, die auf einem Kreis, der senkrecht zur durch die Ohren verlaufenden Achse liegen, haben alle die selben Laufzeitunterschiede und von der Dämpfung durch den Körper abgesehen auch auch die gleichen Intensitätsunterschiede. Bei der Richtungserkennung in der vertikalen Ebene spielen zwei Mechanismen eine wichtige Rolle: die sogenannten „monaural cues “sowie die „dynamic binaural cues “. Monaural cues beschreiben die richtungsabhängige Veränderung des Frequenzsprektrums durch das Außenohr, also der Ohrmuschel und dem äußeren Gehörgang, sowie durch den Kopf und Rumpf der Person [MB85]. Diese Veränderungen im Frequenzspektrum, die auch durch eine *Head-Related Transfer Funcion* oder kurz *HRTF* beschrieben werden können, können erkannt werden und somit eine Lokalisation auf der vertikalen Ebene erfolgen. Diese Lokalisationsart ist am genausten



(a)



(b)

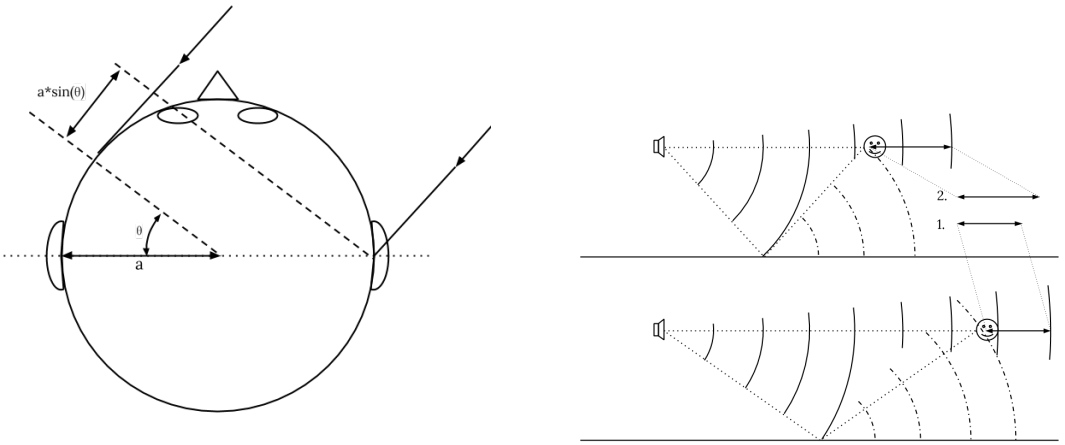


(c)

Abbildung 2.5.: Verschiedene Eigenschaften, die die Lokalisation auf der horizontalen Ebene erlauben.

für breitbandige, hochfrequente und komplexe Geräusche. Die dynamic binaural cues beschreiben die Rolle von Kopfbewegungen in der Soundlokalisation [Wal40]. So kann schon durch leichte Kopfbewegungen eine Richtungsbestimmung auf der vertikalen Ebene ermöglicht werden. Wenn beispielsweise ein Ton von oben kommt verändert sich durch Drehung des Kopfes um die vertikale Achse weder die ITD noch die IID. Grantham et al geben die Auflösung dieser Richtungsbestimmungen mit bis zu 1° auf der horizontalen Ebene und $4^\circ - 5^\circ$ auf der vertikalen Ebene an [GHE03]. Bemerkenswert dabei ist, dass die Bestimmung der horizontalen Richtung in einer diagonalen Ebene auf annähernd konstant hoher Genauigkeit bleibt, auch wenn Ebene um bis zu 85° geneigt wurde.

Für eine vollständige räumliche Wahrnehmung muss auch die Entfernung der Audioquelle wahrgenommen werden. Auch dies geschieht über eine ganze Reihe verschiedener Eigenschaften, die ausgewertet und kombiniert werden, um die Entfernung möglichst genau abzuschätzen [LKPG98]. Den ersten Anhaltspunkt zur Entfernung erhält man durch das Betrachten der Lautstärke. Wenn man nur die Ausbreitung im offenen Raum betrachtet, nimmt die Intensität der Schallwellen bei steigendem Radius r nach dem Abstandsgesetz proportional zu $\frac{1}{r^2}$ ab [WK59]. Es gibt also sehr starke Intensitätsunterschiede die relativ leicht wahrgenommen werden können. Um jedoch aus der Intensität zurück auf die Entfernung schließen zu können, benötigt man eine Referenzintensität, zu der die Entfernung bekannt ist. Aus diesem Grund funktioniert dieser Ansatz nur, wenn es sich um eine gut bekannte Audioquelle handelt, die eine möglichst konstante Lautstärke hat. Für sehr nahe Audioquellen (<1m) können die Pegelunterschiede an den Ohren außerdem zur Entfernungsermittlung beitragen. Beispielsweise können ein Flüstern und ein Gespräch in einem Meter Entfernung auf einem Ohr mit der gleichen Intensität wahrgenommen werden, wobei das Flüstern am anderen Ohr wahrscheinlich nicht mehr wahrgenommen wird, während das Gespräch in einem Meter Entfernung auch auf der anderen Kopfseite noch gut wahrnehmbar ist. Wenn das Frequenzspektrum der Audioquelle bekannt ist, kann auch die Analyse des am Ohr angekommenen Frequenzspektrums Aufschluss über die Entfernung geben [Zah96]. Ab Entfernungen von ungefähr 15 Metern wird das Audiospektrum durch die Luft merklich beeinflusst, da höhere Frequenzen stärker durch die Luft gedämpft werden als tiefere. Bei der Entfernung von Sound innerhalb von geschlossenen oder halboffenen Räumen spielt auch das Verhältnis von Primärschall zu Sekundärschall eine große Rolle [BH99]. Mit Hilfe dieses Verhältnisses, oft auch als „*direct-to-reverberant*“ beschrieben, kann die Entfernung in bestimmten Umgebungen, zumindest für nahe Audioquellen, annähernd linear bestimmt werden. Wie beschrieben nimmt die Intensität des direkten Schalls umgekehrt proportional zur Entfernung ab. Die Intensität des Sekundärschalls, der alle Reflexionen umfasst, bleibt jedoch weitestgehend unabhängig von der Entfernung der Audioquelle [Zah02] konstant. Durch diese Eigenschaft lässt sich die Lautstärke der Audioquelle bestimmen und somit die absolute Entfernung dieser ableiten. In reflektierenden Umgebungen kann auch die Anfangszeitlücke zur Ableitung der Entfernung herangezogen werden. Die Anfangszeitlücke beschreibt die Zeit, die zwischen dem Erreichen der direkten Schallwelle und der ersten Reflexion liegt. Diese ist umso größer, je näher sich die Audioquelle befindet (siehe Abb. 2.6b). In der visuellen Wahrnehmung kann die Bewegungsparallaxe dabei helfen, die Enfernung von Objekten, zu denen man sich parallel bewegt, gegeneinander abzuschätzen. Obwohl dieser Effekt auch bei Audioquellen auftritt, zu denen man sich parallel bewegt, konnten Simpson und Stanton [SS73] zeigen, dass dieser Effekt vom Menschen nicht für die Entfernungsbestimmung von Audioquellen verwendet wird. Alle diese Eigenschaften erlauben es nicht nur die Position einer Audioquelle zu bestimmen, sondern auch deren Bewegung zu verfolgen. Das Erkennen der Richtung, in der sich eine Audioquelle dabei bewegt, wird auch durch den Doppler-Effekt vereinfacht (siehe Abb. 2.7). Jeder



(a) Je größer der Winkel ϕ der eintreffenden Schallwellen, desto größer die Zeitverzögerung zwischen den Ohren

(b) 1. Größere Anfangszeitlücke, wenn die Audioquelle näher ist, 2. Kleinere Anfangszeitlücke, wenn die Audioquelle weiter entfernt ist

Abbildung 2.6.: Zeitdifferenzen können sowohl Aufschluss über die Richtung (a) sowie über die Entfernung (b) geben.

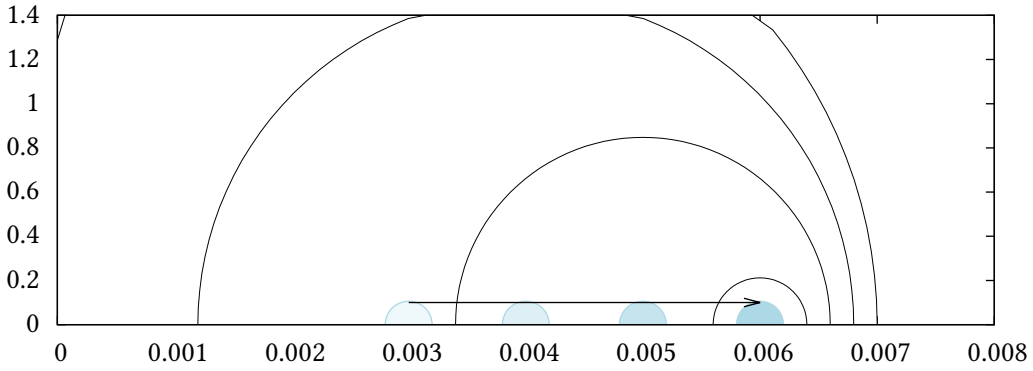


Abbildung 2.7.: Dopplereffekt bei sich bewegenden Audioquellen.

kennt wohl das Phänomen, dass die Sirenen von Einsatzwagen höher klingen, solange der Wagen auf einen zu fährt und tiefer, wenn er sich wieder von einem entfernt.

3. Sonifikation

3.1. Definition und Taxonomie

All die Eigenschaften von Sound, die der Mensch wahrnehmen kann, erlauben eine Abbildung von Daten mit vielen Eigenschaften in Töne, die der Mensch unterscheiden kann. Dieser Vorgang wird im allgemeinen als Sonifikation bezeichnet und Systeme, die diese nutzen, werden als *Auditory Displays* beschrieben. Die Sonifikation von Daten ist ein relativ junges Forschungsgebiet und wird von Kramer et al. [KWB+10] wie folgt definiert:

> Sonification is defined as the use of nonspeech audio to convey information.

Es geht also darum, das Potential, das in der auditiven Wahrnehmung liegt, zu nutzen um Informationen zu übermitteln. Eine ausführliche Definition liefert Hermann [Her08], der die Sonifikation als Methode sieht um einen Datenstrom in eine Folge von Geräuschen umzuwandeln, die dabei die folgenden Eigenschaften erfüllen muss:

- Ein Geräusch spiegelt objektive Eigenschaften oder Beziehungen in den Eingabedaten wieder.
- Die Umwandlung findet systematisch statt. Das bedeutet, es ist eindeutig festgelegt, wie die Daten und optionale Eingaben die Sequenz der Töne beeinflusst.
- Der Umwandlungsvorgang läuft deterministisch ab, weshalb die gleichen Eingabedaten auch das gleiche Ergebnis liefern.
- Das System kann auf verschiedene Daten angewendet werden und nicht nur für eine spezielles Set an Daten.

Unter diese allgemeine Definition von Sonifikation fallen einige Klassen, die sich klar voneinander abgrenzen:

- *Audifikation*
  Die Audifikation ist die einfachste Form der Sonifikation und beschreibt die direkte Abbildung von Daten in Klänge und besitzt deshalb nur wenige Freiheitsgrade. Dabei werden die zu Grunde liegenden Daten einfach als digitale Repräsentation des Audiosignals gesehen. Die Audifikation ist also nur möglich, wenn viele Datenwerte vorliegen, die entweder zeitlich oder räumlich voneinander abhängen und genügend Schwankungen enthalten, um eine Abbildung auf ein Audiosignal zu erlauben. Spher [Spe09] begründet, dass sich der Einsatz von Audifikation nicht für eine allgemeine Analyse von höherdimensionalen Daten verwenden lässt, da das Verfahren nicht allgemein anwendbar ist. Beispiele für sinnvolle Anwendungen sind die Audifikation des menschlichen EEG [HB08] oder der Einsatz zur Vertonung von seismologischen Daten [Dom01].

- *Earcons* und *Auditory Icons*
  Bei dieser Art der Sonifikation geht es darum, auditive Icons zu erzeugen, die die Interaktion der Benutzer mit dem Computersystem erleichtern sollen. Als Icons werden im Allgemeinen symbolhafte Bilder beschrieben (*Piktogramme*), die bestimmte Objekte oder Aktionen abstrahiert darstellen. Durch die vielen unterscheidbaren Eigenschaften der Töne ist es möglich, viele verschiedene Töne zu unterscheiden, wodurch zum einen Aufteilungen durch unterschiedliche Sounds ermöglicht wird. Zum anderen ensteht die Möglichkeit, Gruppierungen durch gemeinsame klangliche Merkmale zu bilden. So könnten alle Töne, die zur Identifikation von Aktionen in einem Untermenü dienen, alle die gleiche Klangfarbe besitzen und nur in der Höhe variieren. In dieser Sonifikationsklasse unterscheiden Dingler et al. [DLW+08] weiterhin zwischen *Earcons*, *Auditory Icons*, *Speech* sowie *Spearcons*.

  Der Begriff *Earcons* wurde 1985 von Sumikawa [Sum85] erfunden als Wortspiel, das analog zu Icons (*Eye-cons*) andeuten sollte, dass dieser Ausdruck hörbare Icons beschreibt. Bei *Earcons* handelt es sich dabei um abstrakte, synthetische Töne, während der Begriff *Auditory Icons* für akustische Icons verwendet wird, die als natürliche Repräsentation von Objekten, Funktionen oder Aktionen betrachtet werden können. Der Startsound oder die akustischen Fehlermeldungen, die in einigen Betriebssystemen verfügbar sind, sind also Beispiele für solche *Earcons*. Beispielhaft für *Auditory Icons* wäre das Abspielen eines Tones, der sich wie das Zusammenknüllen von Papier anhört, wenn der Nutzer eine Datei in den Papierkorb schiebt. Die wohl einfachste Art der Audio Icons ist mit *Speech* das einfache Vorlesen von Informationen wie beispielsweise Menütexten. Während diese Art mit vorhandenen *Text-to-speech-Frameworks (TTS)* einfach zu realisieren und ohne Training zu benutzen ist, hat sie den Nachteil, dass sie im Vergleich zu den anderen Möglichkeiten eine relativ hohe Konzentration benötigt und es beispielsweise sehr schwer ist, ein solches System zu bedienen, während man sich mit jemanden unterhält.

  Das letzte Untersystem bilden die 2006 von Walker et al. vorgestellten *Spearcons* [WNL06]. Dabei wird zunächst ähnlich wie bei normalen *Speech Icons* eine textuelle Beschreibung des Objekts oder der Aktion mit Hilfe von *TTS* in eine Aufnahme verwandelt, die dann jedoch so schnell abgespielt wird, dass das gesprochene selbst nicht mehr erkennbar ist, sondern ähnlich wie bei *Earcons* ein synthetischer Sound entsteht. Diese *Spearcons* haben den Vorteil, dass sie sich einfach erzeugen lassen, aber keine hohe mentale Herausforderung an die Benutzer stellen. Weiterhin resultieren ähnliche Einträge wie zum Beispiel *Speichern* und *Speichern unter* auch in ähnlichen Sounds, wodurch die automatische Klassifizierung erleichtert wird. Insgesamt lässt sich über die auditiven Icons sagen, dass diese die Benutzungsgeschwindigkeit und Effizienz erhöhen, sowie die Reaktionszeiten und die benötigte Zeit zum Erlernen eines Systems verkürzen können.

- *Datenstrombasierte Sonifikation* und *Parameter-Mapping*
  Die datenstrombasierte Sonifikation und das *Parameter-Mapping* beschreiben zwei Sonifikationsarten, die sich im Grunde nur durch den Interaktionsgrad der Eingabe unterscheiden. Bei der datenstrombasierten Sonfikation werden die zu vertonenden Daten zuerst aufgenommen und eventuell Vorverarbeite um anschließend die Daten auf bestimmte Töne abzubilden. Ein Beispiel dieser Sonifikationsart findet sich bei Barass et al. [BB08], die die Sonifikation von verschiedenen Diagrammen untersuchen. Im Gegensatz dazu wird beim *Parameter-Mapping*

der Ton quasi in Echtzeit aus einem Eingabestrom berechnet. Diese beiden Sonifikationsarten sind die am häufigsten anzutreffenden und haben gemeinsam, dass hierbei bestimmte Eigenschaften in den Daten entweder auf Eigenschaften der Töne wie zum Beispiel Tonhöhe oder Länge abgebildet werden. Dies geschieht im Allgemeinen durch eine Datenvorverarbeitung, die gewünschte Eigenschaften extrahiert und eventuell gefiltert. Auch eine Normierung der Daten ist wichtig, um diese auf vom Menschen wahrnehmbare Sounds zu projizieren. Im zweiten Schritt werden dann die Eigenschaften auf Töne projiziert. Dabei können laut Grond und Berger [HHN11] verschiedene Abbildungsarten zum Einsatz kommen: Die Eins-zu-Eins-Abbildung, die einzelne Dimensionen in den Daten unabhängig auf Eigenschaften der Töne abbildet, die Eins-zu-Viele-Abbildung variiert mehrere Toneigenschaften in Abhängigkeit einer Datendimension und der Fall der Viele-zu-Eins-Abbildung, bei der mehrere Eigenschaften in den Daten auf einzelne Eigenschaften in den Tönen abgebildet werden. Das *Parameter-Mapping* unterscheidet sich von der Audifikation dadurch, dass keine großen Datenmengen gebraucht werden und auch mehrdimensionale Datensätze sonifiziert werden können. Einige Hörbeispiele für diese Art der Sonifikation finden sich unter [Her11].

- *Modell-basiert*
  Die *Model-basierte* Sonifikation wurde 1999 von Hermann [HR99] beschrieben und ist die umfassendste Sonifikationsart. Hierbei findet nicht nur eine einfache Abbildung von Datenwerten auf Toneigenschaften statt, sondern es wird ein Modell entworfen, dass komplexe Zusammenhänge in den Daten auf Töne abbilden kann. Analog zur realen Welt, in der das Stimulieren von Materialien diese zum Schwingen bringen kann, was letztendlich von Menschen als Ton wahrgenommen wird, gibt das Modell virtuelle physikalische Gesetze vor, was die Daten laut Hermann letztendlich in gewissem Sinne selbst zum Instrument macht, mit welchem der Benutzer durch Explorieren des Datenraums Töne erzeugt.

Dieses große Spektrum an unterschiedlichen Sonifikationsarten erlaubt, deren Einsatz zu vielen unterschiedlichen Anwedungsgebieten, in folgende Klassen zu unterteilen.

- Alarm, Benachrichtigungen und Warnmeldungen:
  Nees et al. [NW09] beschreiben diese kurzen, unregelmäßigen Töne, die durch ihre Charakteristiken hervorstechend sind. Also solche vermitteln sie dem Benutzer rein binäre Informationen. So können Benachrichtigungen am Handy durch einen Signalton zwar angeben, dass eine neue Nachricht eingetroffen ist, geben jedoch keinen Aufschluss über deren Inhalt.

- Statusanzeige:
  Auch Statusanzeigen sind wie Benachrichtigungen nur kurz, übermitteln jedoch mehr Informationen. Hierzu gehört beispielsweise die Gruppe der auditiven Icons, die nicht nur über einen Mausklick informieren können, sondern auch darüber, welche Schaltfläche angeklickt wurde.

- Sonifikation von wissenschaftlichen Daten:
  Wie anhand der vielen Beispiele in dieser Arbeit und an dieser Arbeit selbst zu sehen ist, eignet sich die Sonifikation auch zur alternativen oder ergänzenden Präsentation und Exploration von Datenmengen. Insbesondere das Erkennen von Mustern innerhalb dieser Daten kann durch die Sonifikation stark verbessert werden.

- Musikalische Unterlegung als Kunst:
  Als letzter Punkt ist noch zu erwähnen, dass die Sonifikation auch in der Kunst Verwendung findet. Ein Beispiel findet sich in Kapitel Sonifikation in Mathematik und Physik.

3.2. Sonifikation in 3D

Da in dieser Arbeit dreidimensionale Simulationsdaten sonifiziert wurden, soll im folgenden insbesondere auf die Besonderheiten bei räumlichen *Auditory Displays* eingegangen werden.

Die Anwendung von räumlichen *Auditory Displays* insbesondere dann sinnvoll, wenn die Daten selbst schon räumlich vorliegen. Aber auch wenn dies nicht der Fall ist, ermöglichen *Auditory Displays* mit räumlichem Sound mehr Informationen an die Benutzer zu übermitteln. Den Fall, dass bereits räumliche Informationen vorliegen, haben Grohn et al. [GLT03] untersucht. In einem Versuch wurde dabei die Navigation von Benutzern durch einen dreidimensionalen Raum untersucht und in wie weit *Auditory Displays* hierbei sinnvoll sind. Dazu mussten die Probanden, ähnlich wie bei einem Rennspiel, möglichst viele Tore passieren. Die Position dieser Tore wurde dabei sowohl rein durch Töne übermittelt, als auch rein visuell sowie in einer gemischten Form. Der Versuch hat dabei gezeigt, dass eine Mischung aus visuellen und auditiven Hinweisen die besten Ergebnisse liefert und reine Audiohinweise am schlechtesten funktionieren. Die Autoren weisen darauf hin, dass meist die grobe Richtung mit Hilfe des Audioinformationen ermittelt wurde und dann die genaue Navigation unter Verwendung der visuellen Hinweise stattfand. Dies zeigt, dass sich *Auditory Displays* dazu eignen die Aufmerksamkeit auf bestimmte Ereignisse zu lenken. Pennock [Pen] beschreibt verschiedene Anwendungsgebiete von räumlichen *Auditory Displays* im Auto. Dazu gehören zum einen die Reduktion von Fehlern bei der Bestimmung, woher eine bestimmte Meldung kommt und ob diese für ihn gerade wichtig ist und verbesserte Navigation und Reaktionszeiten durch frühere akustische Warnzeichen. Scarpaci et al. [SC05] schlagen ein System vor, bei dem das Tracking, die Soundgenerierung bzw. Laden, Positionierung der Töne sowie die Umsetzung des räumlichen Eindrucks durch verschiedene Module übernommen wird und das System deshalb leicht angepasst werden kann. Die Ergebnisse von Carlile et al. zeigen die Limits, die mit räumlichen *Auditory Displays* erreicht werden können. Insbesondere konnten sie zeigen, dass die Verwendung von mehr als 150 aufgenommenen Punkten für die HRTF (siehe Kapitel 4.3.3) keine signifikanten Verbesserungen für die Lokalisation bringen. Ein weiteres Ergebnis ist, das komplexere Sounds einfacher zu lokalisieren und deshalb für *Auditory Displays* besser geeignet sind als pure Sinustöne.

3.3. Sonifikation und Visualisierung

Auch wenn in vielen Anwendungen nur reine Visualisierung eingesetzt wird, hat die Sonifikation einige Vorteile gegenüber der Visualisierung, die zumindest eine Erweiterung der reinen Visualisierung mit Hilfe der Audifikation begründen können. Zum einen gilt, dass beim gleichzeitigen Abspielen von zwei Noten, diese immer noch einzeln erkannt werden können, während sich in der Visualisierung verschiedene Farben vermischen und eine neue Farbe erzeugen. Zwar könnten Daten auch auf verschiedenen Monitoren parallel visualisiert werden, jedoch hat der Mensch nur ein eingeschränktes

Sichtfeld, weshalb dieser Ansatz schwierig sinnvoll umzusetzen ist. Als weitere Vorteile listet McGee [McG09] die Fähigkeit des Menschen auf, Geräusche aus allen Richtungen wahrzunehmen. Diese Fähigkeit ermöglicht einerseits ein Eintauchen in die Daten und zum anderen erlaubt es dem Benutzer, sich auf einen Bildschirm zu konzentrieren, während man gleichzeitig noch andere Prozesse verfolgen kann. Ein weiterer Vorteil der Audifikation besteht laut McGee darin, dass der Mensch seine Ohren nicht wie seine Augen schließen oder wegsehen kann. Somit ist gewährleistet, dass nicht aus Versehen ein wichtiges Ereignis nicht erkannt wird. Audifikation. Durch die Rundum-Wahrnehmung von Sound kommt jedoch auch die Schwierigkeit hinzu, dass sich *Auditory Displays* leicht von außen stören lassen. Dies kann zwar durch Verwenden von abschirmenden Kopfhörern verhindert werden, allerdings erschwert sich dadurch wiederum Richtungsbestimmung von Tönen, da Kopfposition und Ausrichtung eigentlich Einfluss auf den Ton nehmen müssten.

Als zusätzlichen Vorteil der Sonifikation kommt hinzu, dass das Gehör langsamer ermüdet als die Augen. Dies gilt natürlich nur, solange keine unangenehmen oder zu lauten Töne abgespielt werden. Diese Eigenschaft stellt die Sonifikation jedoch auch vor ein Problem, welches in der Visualisierung nicht in gleichem Maße auftritt: die Ästhetik des produzierten Bildes hat in der Visualisierung in der Regel keinen großen Einfluss auf das Wohlbefinden des Benutzers. Dagegen ist ein angenehmer Sound in der Sonifikation meist wichtig um kein Unbefinden bei den Benutzern zu erzeugen. Als letztes ist noch der Vergleich der Menge an produzierten Daten zu vergleichen. Hier gibt McGee als Beispiel einen 16-Bit-Videostream an, welcher mit einer Auflösung von 640×480und 30 Frames/s unkomprimiert eine Größe von 18,4MB/s besitzt, während ein Audiostream mit 2 Kanälen und einer Abtastung mit 16-Bit Genauigkeit und 44100 Samples/s gerade einmal auf 0,17MB/s kommt. Ein großer Nachteil gegenüber der Visualisierung besteht darin, dass zum Abspielen von Tönen immer ein Gerät verwendet werden muss, während Visualisierungen, die nicht interaktiv sind auch in Form von Bildern übermittelt werden können, die auf Papier gedruckt wurden. Als letzten Vorteil der Sonifikation kann man die Möglichkeiten aufführen, die die Sonifikation Menschen mit Sehbehinderungen eröffnet [RFCZ12]. Insbesondere kann die Sonifikation es sehbehinderten Forschern erlauben, bestimmte Experimente selbst durchzuführen.

3.3.1. Erfolgreiche Sonifikationen

Dass die Sonifikation nicht nur in der Forschung eingesetzt werden kann, zeigen einige prominente Beispiele für erfolgreiche Sonifikationen, die zum Teil auch im *Sonification Report 2010* [KWB$^+$10] Erwähnung finden. Dazu gehört der Geigerzähler, mit dessen Hilfe die Intensität einer radioaktiven Strahlung als Knackgeräusche wahrnehmbar wird, sowie das Sonar, bei dem die Schallwellen, die an Objekten reflektieren, über einen Lautsprecher wiedergegeben werden und somit die Erkennung von Hindernissen ermöglichen. Für Fallschirmspringer existieren Höhenmesser, die Sonifikation benutzen, um dem Springer über die Höhe zu informieren. Auch für medizinische Anwendungen wurden einige Sonifikationen entwickelt. Dazu gehört beispielsweise das hörbare Thermometer sowie das sogenannte Pulsoximeter. Bei diesem Gerät wird die Tonhöhe des abgespielten Tones variiert und somit der Sauerstoffgehalt des Blutes hörbar gemacht. Da es die konstante Beobachtung erlaubt, auch wenn der Chirurg sich gerade auf die Operation konzentriert, gehört es heute zur Standardaustattung eines Krankenhauses. Gaver et al. [GSO91] konnten zeigen, dass die Sonfikation bei komplexen Aufgaben nicht nur ein angenehmeres, sondern auch ein effektiveres Zusammenarbeiten erlaubt.

Dazu simulierten sie eine Fabrik, in der Cola hergestellt und in Flaschen abgefüllt werden musste. Die komplette Produktionslinie umfasst acht Automaten, von denen jeweils vier von einer Person gesteuert wurde. Die zwei Personen, die zusammen den ganzen Ablauf steuern werden dabei in getrennten Räumen untergebracht und können über ein Mikrophon und Lautsprecher miteinander kommunizieren. Zusätzlich konnte über die Lautsprecher auditives Feedback der einzelnen Maschinen abgespielt werden. Gaver gibt dabei an, dass die Benutzer die verschiedenen Geräusche schnell zu unterscheiden lernten und nur einige wenige Teilnehmer der Studie vergaßen, wie die Sounds mit den Vorgängen zusammen hingen. Insgesamt konnte mit dem eingeschalteten auditiven Feedback die Produktivität erhöht werden, und die Forscher sehen hier großen Nutzen für komplexe Systeme. Auch wenn die Sonifikation also noch immer hauptsächlich in der Forschung eingesetzt wird, zeigen diese Beispiele doch den praktischen Nutzen der Sonifikation und begründen somit die Entwicklung von Systemen, die *Auditory Displays* einsetzen.

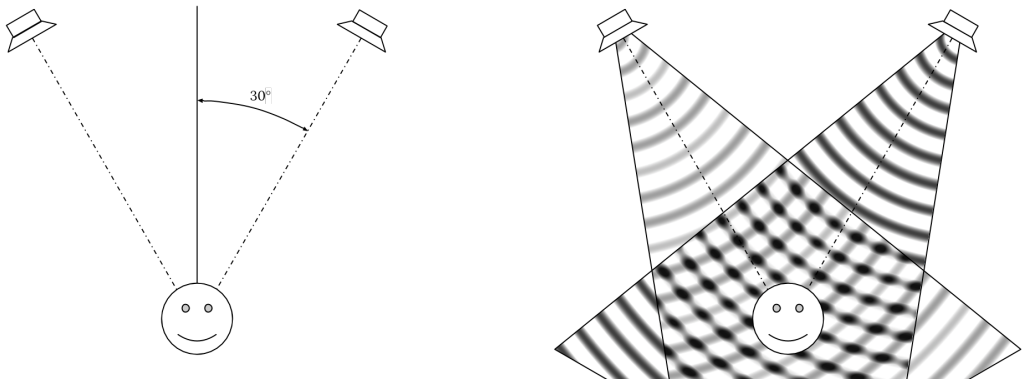
4. 3D-Sound

Es gibt verschiedene Möglichkeiten um räumliche Audio wiederzugeben. Die verschiedenen Möglichkeiten, einen räumlichen Eindruck der Audioquelle zu erhalten, sollen in diesem Abschnitt beschrieben werden.

4.1. Wiedergabe von räumlichen Audiodaten

Die verschiedenen Arten, räumliche Töne wiederzugeben, kann durch Variation der Lautsprecheranzahl sowie der Techniken, um diese Lautsprecher möglichst effektiv benutzen zu können, klassifiziert werden. In der folgenden Auflistung finden sich die möglichen Lautsprecherinstallationen nach ihren Fähigkeiten räumliche Töne zu erzeugen sortiert vor.

- Mono-Lautsprecher: Die einfachste Methode benutzt nur einen Audiokanal und kann deshalb nur zum Abspielen von Mono-Sounds verwendet werden. Dabei werden alle Audiosignale zusammengemischt und auf einem Kanal an alle Lautsprecher übertragen. In der Regel reicht ein Lautsprecher für diese Variante, jedoch können auch mehrere Lautsprecher eingesetzt werden um beispielsweise einen großen Raum zu beschallen. Die einzige Variation, die einen räumlichen Eindruck erzeugt, ist die Lautstärkeänderung, durch welche die Entfernung der Audioquelle simuliert werden kann.
- Stereo-Lautsprecher: Durch Hinzunahme eines weiteren Audiokanals kann schon zwischen zwei Richtungen unterschieden werden. Wenn die Lautprecher nicht ausgerichtet sind oder das Audiosignal nicht entsprechend gemischt wird, kann man dabei jedoch nur zwischen zwei Richtungen entscheiden. Erst durch das Ausrichten der Lautsprecher und Anpassung des Audiosignals kann eine Audioquelle simuliert werden, die zwischen den beiden Lautsprechern liegt. Als optimale Ausrichtung hat sich laut Rumsey [Rum01][S.53] eine Installation herausgestellt, bei der die Lautsprecher im Winkel von $\pm 30^\circ$ vor dem Benutzer aufgestellt sind (siehe Abb. 4.1a). Das Problem, das bei allen Installationen entsteht, die mit zwei oder mehreren Lautsprechern versuchen, virtuelle Audioquellen durch Simulation von ITD und IID zu simulieren, besteht darin, dass die Töne vom linken Lautsprecher auch das rechte Ohr erreichen und umgekehrt. Dadurch, dass sich die beiden Töne überlagern, kann es zu Übersprechungen kommen die den 3D-Effekt auslöschen. Schon 1966 hatten Atal und Bishnu [Ata66] deshalb an einer Technik geforscht, die dieses Übersprechen verhindern kann. Das Grundprinzip dieser als *Crosstalk-Cancellation* oder kurz *XTC* bekannten Technik wird in Abbildung 4.2 erläutert. Theoretisch reichen also zwei Audioquellen aus um an den beiden Ohren verschiedene Signale zu generieren. Sieht man von Reflexionen im Raum und am Körper, sowie der Wahrnehmung von Vibrationen außerhalb der Ohren ab, wäre es also möglich, an den Ohren jeweils genau jene Signale zu



(a) Optimales Setup für Stereolautsprecher

(b) Überlagerungen der Schallwellen die von unterschiedlichen Lautsprechern kommen.

Abbildung 4.1.: Optimale Stereoinstallation und Übersprechen von Schallwellen durch Überlagerung.

erzeugen, die von einer frei im Raum platzierten Audioquelle ankommen würden. Dazu müssten nur all jene Eigenschaften simuliert werden, die der Mensch unterscheiden kann (siehe dazu Kapitel 2). Dennoch gibt es einige Probleme bei dieser Art, dreidimensionalen Sound zu erzeugen: Zum einen funktioniert *XTC*, wie auch in Abb. 4.2 zu sehen ist, nur lokal beschränkt. Es entstehen also *Sweet spots*, an denen die *XTC* funktioniert. Ist der Nutzer nun nicht perfekt ausgerichtet oder bewegt seinen Kopf etwas, reduziert dies schon den räumlichen Eindruck drastisch. Eine mögliche Verbesserung kann hier durch das Tracking des Benutzers [Gar97] und entsprechende Adaption des XTC-Filter erreicht werden. Durch Sensoren wie die Kinect wird es also möglich, den Sweetspot perfekt an den Nutzer anzupassen. Da unterschiedliche Frequenzen unterschiedlich stark durch die Luft gedämpft werden, kommt es außerdem zu einer spektralen Verzerrung des Audiobildes, wenn dieser Effekt nicht aktiv ausgeglichen wird [Cho08]. Ein weiterer Nachteil dieser Technik besteht darin, dass nur ein *sweet spot* erzeugt werden kann, was nur durch die Verwendungung von mehr Lautsprechern, wie beispielsweise in einem *Phased-Array System*, [Mil06] verbessert werden kann. Ein weiterer Nachteil ist die genaue Synchronisation und Kalibrierung, die nötig ist und die die bisherigen Audiosysteme, die auf dieser Technik beruhen, relativ teuer machen.

- Kopfhörer: Der Einsatz von Kopfhörern entspricht im Allgemeinen jenem von zwei Lautsprechern mit perfektem *XTC*, da die Ohren jeweils mit verschiedenen Audiokanälen beschallt werden können. Der Nachteil von Kopfhörern gegenüber Lautsprechern besteht darin, dass der Schall nur über die Kopfhörer übertragen wird und deshalb keine Vibrationen an anderen Körperteilen spürbar sind. Jeder, der schon etwas lautere Musik gehört hat, weiß, dass vor allem laute Bässe im ganzen Körper zu spüren sind. Dieses Gefühl und der damit eventuell erzeugte Effekt in Filmen, sich wirklich im Geschehen zu befinden geht somit mit Kopfhörern verloren. Auch eine möglicherweise eingeschränkte Bewegungsfreiheit durch kabelgebundene Kopfhörer ist unvorteilhaft gegenüber Lautsprechern. Außerdem ist das Tragen der Kopfhörer manchen zu ungemütlich. Vorteilhaft sind Kopfhörer in Situationen, in denen man sowohl von Umgebungsgeräuschen gestört werden kann oder ein Einsatz von Lautsprechern nicht möglich

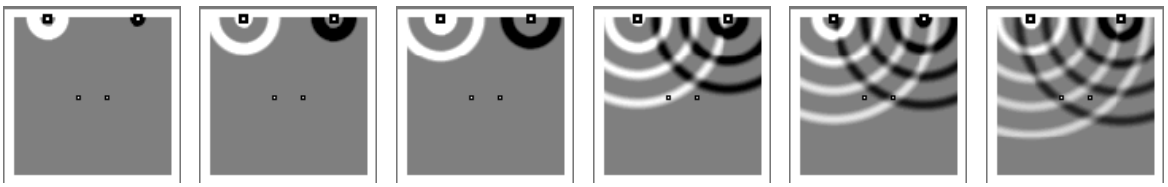


Abbildung 4.2.: Grundprinzip der *Crosstalk-Cancellation* [DAG]: a) Ein positiver Impuls, der für das linke Ohr gedacht ist, wird über den linken Lautsprecher ausgegeben. b) Mit kurzer Verzögerung wird ein etwas schwächerer negativer Impuls über den rechten Lautsprecher abgegeben. c) Kurz darauf wird wieder ein noch schwächerer Impuls über den linken Lautsprecher wiedergegeben. Dieser Vorgang wird nun wiederholt bis die Impulse schwach genug sind, um nicht mehr wahrgenommen zu werden. d) Das Signal, das für das linke Ohr gedacht war, erreicht dieses. e) In dem Moment, in dem das Signal, welches für das linke Ohr gedacht war, das rechte Ohr erreicht, erreicht auch der negative Impuls das rechte Ohr und das Signal wird ausgelöscht. d) Dies wiederholt sich bei allen weiteren Signalen, bis das Signal zu schwach wird, um wahrgenommen zu werden

ist, ohne andere Personen zu stören. Generell kann man also sagen, dass die Erzeugung eines räumlichen Klangs mit Kopfhörern einfacher und billiger zu realisieren ist, man dadurch jedoch kleinere Opfer in der Bequemlichkeit eingeht.

- Surround-System: Obwohl Surround-Sound für eine einzelne Person theoretisch mit zwei Lautsprechern realisierbar ist, ist dazu eine sehr genaue Kalibrierung auf diese Person notwendig. In den meisten Audiosystemen, die zwei Lautsprecher einsetzen, wird deshalb nur eine Simulation von Audioquellen ermöglicht, die zwischen den beiden Lautsprechern auf einer Linie liegen. Durch den Einsatz von Surround-Systemen, in denen man beispielsweise zwei weitere Lautsprecher hinter dem Kopf des Benutzers anbringt, kann relativ einfach die kompletten 360° des Horizontes abgedeckt werden. Wie oben beschrieben erlaubt der Einsatz von mehreren Lautsprechern auch das Erzeugen mehrerer *sweet spots* unter Einsatz fortgeschrittener *XTC*-Techniken, was jedoch auch größeren Aufwand und teurere Systeme bedeutet.
- Lautsprecherarrays: Die wohl einfachste, aber effektivste Methode Raumklang zu erzeugen, ist das Verwenden von möglichst vielen Lautsprechern, die jede Richtung abdecken. Dieser Brute-Force-Ansatz hat den Vorteil, dass er Raumklang für beliebig viele Menschen ermöglicht, da er nicht auf *sweet spots* angewiesen ist. Die meisten Soundeigenschaften, die eine Lokalisation der Audioquelle erlauben, müssen auch nicht simuliert werden, sondern sind durch den Aufbau schon vorgegeben. Lediglich die Entfernung sowie mögliche Reflexionen müssen simuliert werden, um eine genaue Positionserkennung zu ermöglichen. Nachteil dieser Systeme ist zum einen ihr hoher Preis und ihre Komplexität, die den Einsatz im Alltag stark beeinträchtigen. Ein weiterer Nachteil gegenüber dem Verwenden von Kopfhörern oder entsprechenden Surround-Systemen besteht darin, dass es nicht möglich ist, jedem Benutzer das gleiche Audiobild zu präsentieren. So ist beispielsweise ein Flüstern entweder nur für die hörbar, die nahe am

Lautsprecher stehen, oder es muss so laut abgespielt werden, dass es sich für Personen, die sich in der Nähe des Lautsprechers befinden, nicht mehr wie ein Flüstern anhört.

4.2. Speicherung von 3D Sounddaten

Auch die Art und Weise, wie die Audiodaten gespeichert werden, kann unterschieden werden. Hier gibt es im Grunde genommen die vier folgenden Möglichkeiten:

- *Channel-basiert*: Die meisten Audioformate, die eingesetzt werden, beruhen auf einer Audiokanal basierten Speicherung von Audiodateien (siehe zum Beispiel Dolby [for14a] oder DTS [for14b]). Im Grunde genommen entspricht jeder Kanal den Audiosignalen, die zu einem bestimmten Lautsprecher geschickt werden. Falls die Anzahl der Channel nicht mit denen der Lautsprecher übereinstimmt, kann auch eine Interpolation durchgeführt werden um neue Kanäle zu erzeugen oder aber eine Reduktion auf weniger Kanäle, in dem die Audiosignale der Kanäle, die wegfallen, auf die Übrigbleibenden aufgeteilt wird. Vorteil dieser Methode ist das relativ unkomplizierte Abspielen, insbesondere wenn die Anzahl der Kanäle mit der der Lautsprecher übereinstimmt.
- Binaural: Als binaurale Audiodateien können alle Audiodateien betrachtet werden, die abspeichern, welche Audioinformationen genau an den Ohren ankommen. Es handelt sich also hierbei um eine spezielle Form von Audiodateien mit zwei Kanälen. Alle Richtungsinformationen sind dabei explizit dadurch gegeben, dass der Sound genau die Signale repräsentiert, die an den beiden Ohren ankommen. Dieses Audioformat eignet sich zur Wiedergabe mit Kopfhörern oder über Lautsprecher unter Verwendung von *XTC*.
- Klangfeld: Anstatt einzelne Kanäle für die verschiedenen Lautsprecher abzuspeichern, kann auch einfach das Klangfeld aufgenommen werden. Diese Art ist auch unter dem Namen Ambisonics bezeichnet [MM95]. Zum Beschreiben des Klangfeldes wird ein vierdimensionaler Vektor verwendet, der zum einen die Schalldruckkomponente enthält und zusätzlich die drei Schallschnellekomponenten der drei Raumachsen. Dadurch wird also beschrieben, von welcher Richtung und mit welcher Intensität zu einem bestimmten Zeitpunkt eine Schallwelle auf einen bestimmten Punkt im Raum trifft. Der Vorteil dieser Methode gegenüber den Mehrkanalverfahren besteht darin, dass es sich um ein isotropes Format handelt, also jede Richtung gleichwertig kodiert wird und dass nur vier Kanäle ausreichen um eine vollständige Abdeckung des Raumes zu erreichen.
- Audioquellen: Die einfachste Möglichkeit dreidimensionle Klänge abzuspeichern, besteht darin für jede Audioquelle abzuspeichern, wo sich diese befindet, während sie einen bestimmten Ton abspielt. Der Vorteil dieser Art besteht darin, dass relativ einfach eine virtuelle Soundumgebung geschaffen werden kann. Der Benutzer kann sich dann frei durch diese Umgebung bewegen, während aus dieser virtuellen Umgebung von einem System die passenden Audioausgaben erzeugt werden.

4.3. Erzeugung

Dieser Abschnitt wird sich mit der Frage befassen, wie man überhaupt räumliche Klänge erzeugen kann. Eine Möglichkeit besteht in der Aufnahme von räumlichen Tönen. Die Aufnahmetechnik, die am häufigsten Verwendung findet, sind einfache Stereomikrofone, bei denen zwei Mikrofone in gewissem Abstand zueinander platziert sind und somit den Aufbau des Gehörs nachahmen. Durch diese Technik kann man jedoch nur die ITD und IIL wahrnehmen und nicht die Änderungen des Frequenzspektrums, die durch Reflexionen am Oberkörper, Kopf und Ohrmuschel entstehen. Einen Überblick über die vielen verschiedenen Stereoaufnahmearten findet sich in der Arbeit von Patrick Blomqvist [Blo10]. Eine Verbesserung von räumlichen Aufnahmen ist mit Hilfe eines Kunstkopfes möglich, in dessen Ohren die Mikrofone platziert sind [CJM00]. Als letzte Möglichkeit sei hier noch die Aufnahme von Klangfeldern genannt, die durch Einsatz eines Kugelmikrofons für den Lautstärkepegel, sowie sechs Mikrofonen zur Aufnahme der Schallschnellen auf den drei Raumachsen, realisierbar ist.

Interessanter für die Sonifikation ist die Synthetisierung von räumlichen Klängen durch virtuelle Audioquellen. Ob die Töne, die von diesen Audioquellen abgespielt werden, Aufnahmen sind oder auch synthetisch hergestellt wurden spielt dabei keine Rolle. Obwohl eine genaue Berechnung der Ausbreitung von Schallwellen theoretisch durch Lösen der Wellengleichung möglich ist [DZL$^+$05], ist dieser berechnungsintensive Ansatz für den Gebrauch in Systemen, die in Echtzeit arbeiten sollen, nicht praktikabel. Aus diesem Grund wird meist eine vereinfachte Berechnung verwendet, durch welche die Eigenschaften, die eine Lokalisation der Audioquellen erlauben, angenähert werden.

4.3.1. Entfernung

Eine Vereinfachung ist die Approximation der Lautstärkeveränderung, die von der Entfernung der Audioquelle vom Hörer abhängt. Begault [B$^+$94] schlägt hier eine relative Veränderung der dB vor, welche durch die Funktion:

$$\Delta_{dB} = 20log_{10}\left(\frac{1}{distance\ increment}\right) \tag{4.1}$$

beschrieben wird. Hierbei beschreibt ein $distance\ increment$ von 1 den lautest möglichen Ton direkt neben dem Kopf (10cm von der Kopfmitte entfernt). Jede weitere Verdopplung würde die $distance\ increment$ um zwei erhöhen. Nach Auflösen der Formel $distance\ increment = 10 * 2^{(d-1)/2}$ erhält mal für die relative Veränderung in Abhängigkeit von der Distanz d in cm:

$$\Delta_{dB}(d) = 20log_{10}\left(\frac{1}{2 * log_2(\frac{\sqrt{(2)}}{10}d)}\right). \tag{4.2}$$

Für eine noch realistischere Dämpfung muss die Funktion jedoch auch abhängig von der Frequenz sein (siehe Tabelle 4.1).

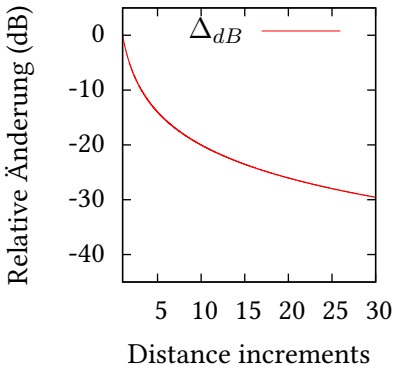


Abbildung 4.3.: Lautstärkeänderung in Abhängigkeit der Entfernung.

Frequenz (Hz)	Dämfpung (dB/Km)
50	0.057
100	0.22
250	1.13
500	2.80
1000	4.89
5000	30.3
10000	118

Tabelle 4.1.: Dämpfung bei 20° und 70% Luftfeuchtigkeit in Abhängigkeit der Frequenz

4.3.2. ITD

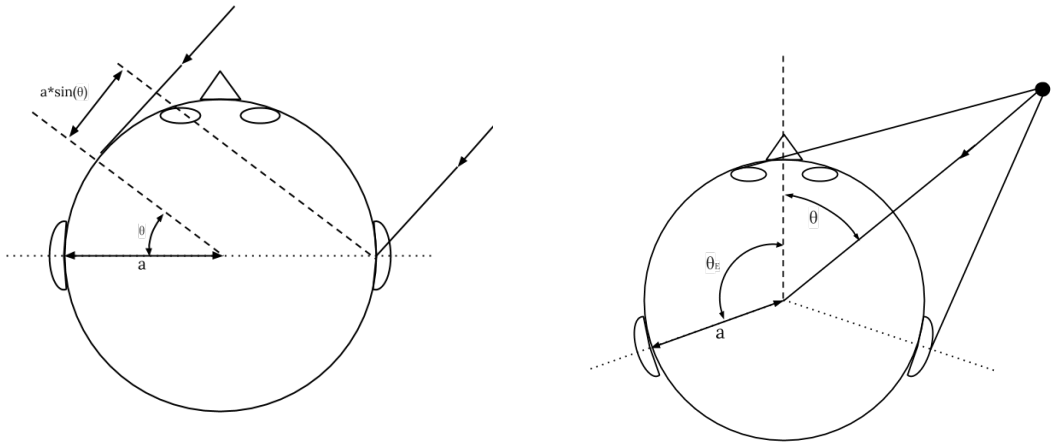
Die wahrscheinlich wichtigste Eigenschaft zur Lokalisation auf der horizontalen Ebene ist die ITD (siehe 2). Woodworth beschrieb hier das erste vereinfachte Modell [Woo38]. Bei diesem Modell wird davon ausgegangen, dass sich die beiden Ohren im Winkel von 90° zur Vorderseite des Gesichtes befinden und als weitere Vereinfachung wird angenommen, dass die Schallwellen sich in einer Ebene ausbreiten, die Audioquelle also unendlich weit entfernt ist (siehe Abb 4.4a). Die ITD kann somit in Abhängigkeit des Winkels ϕ, aus der die Schallwellen eintreffen, dem Radius des Kopfes a sowie der Schallgeschwindigkeit c mit folgender Formel bestimmt werden:

$$ITD(a,c,\phi) = \begin{cases} (a/c)[\phi + sin(\phi)] & , falls\ 0 \leq \phi \leq \pi/2 \\ (a/c)[\pi - \phi + sin(\phi)] & , falls\ \pi/2 \leq \phi \leq \pi \end{cases} . \tag{4.3}$$

Eine Verbesserung, die es unter anderem erlaubt, Punktschallquellen und verschiedene Winkel der Ohren zu definieren, wird von Aaronson und Hartman vorgestellt [AH14].

4.3.3. HRTF

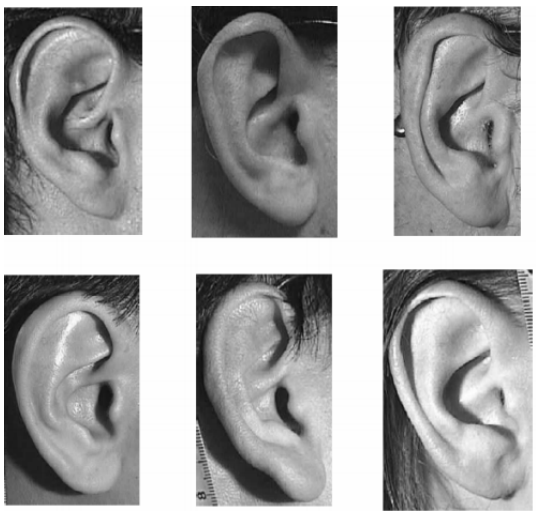
Wie in Kapitel 2 beschrieben, findet die Unterscheidung zwischen vorne/hinten und oben/unten dadurch statt, dass das Audiospektrum durch den Oberkörper, den Kopf sowie die Ohrmuscheln verändert wird. Diese Veränderung kann man durch eine *Head Related Transfer Functions* (HRTF) beschreiben. Die HRTF gibt das Verhältnis zwischen zwei Audiospektren an: jenem, welches an den Tromelfellen gemessen werden kann sowie jenes, welches man an dieser Stelle vorfinden könnte, wenn keine Veränderung des Spektrums durch den Körper stattfinden würde. In der Regel werden diese Funktionen durch Aufnehmen von Kontrollsounds erstellt, die von verschiedenen Punkten aufgenommen werden, die auf einer Kugel um einen Menschen oder eine menschliche Puppe mit in den Ohrkanälen platzierten Mikrofonen [PC94] liegen. Zwischen diesen Punkten kann dann die HRTF mit verschiedenen Interpolationsverfahren angenähert werden [NMK+96]. Ein Problem bei der Verwendung von HRTFs ist, dass sich die HRTFs von verschiedenen Personen stark unterscheiden



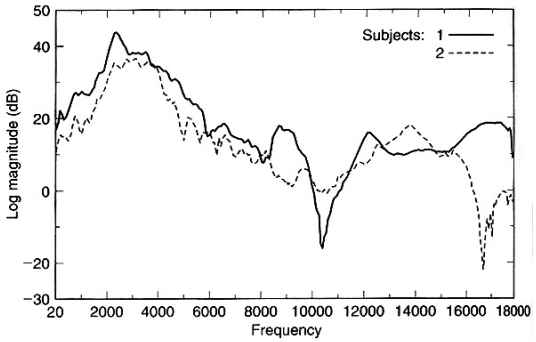
(a) Woodworth's Modell zur Berechnung der IDT **(b)** Verbessertes Modell von Aarson und Hartman

Abbildung 4.4.: Woodworth's Modell geht von parallelen Schallwellen (unendlich weit entfernte Schallquelle) aus und berücksichtigt nicht unterschiedliche Winkel der Platzierung der Ohren. Das verbesserte Modell von Aaronson und Hartman erlaubt hingegen die Verwendung von Punktschallquellen und beliebige Winkel an den Ohren.

können (siehe Beispiel in Abb 4.5). Da sich sowohl die Rumpfform als auch die Kopfform sowie der Aufbau der Ohrmuscheln (siehe Abb. 4.5a) von Person zu Person stark unterscheiden, können sich die *HRTFs* zweier Personen stark unterscheiden (siehe Abb. 4.5b). Zur Lösung dieses Problems wurden verschiedene Ansätze vorgeschlagen. Tan et al. [TG98] die Möglichkeit, die HRTFs vom Benutzer selbst anpassen zu lassen, während Duraiswani et al. [ZDD+02] eine Methode zur automatischen Kalibrierung mit Hilfe der Abmessungen und Aufbau der Ohrmuschel untersuchen.



(a) Verschiedene Ohrformen [Vasa]



(b) Unterschiedliche HRTFs zweier Personen [Vasb]

Abbildung 4.5.: Unterschiedliche Formen der Ohrmuscheln tragen zu unterschiedlichen *HRTFs* bei.

5. Verwandte Arbeiten

5.1. Sonifikation in Mathematik und Physik

Auch in der Molekularphysik gibt es einige Ansätze zur Sonifikation. Delatour [Del00] erklärt ein System, mit dem Molekülschwingungen hörbar gemacht werden können. Die Atome, aus denen ein Molekül besteht, schwingen in bestimmten Frequenzen, sodass eine Oszillation des gesamten Moleküls stattfindet. Diese Schwingungen sind zu schnell, um sie durch einfache Verstärkung hörbar zu machen, jedoch schlägt Delatour eine Abbildung in das hörbare Spektrum vor. Die Messung der Frequenz wird dabei nicht in Zeitdomäne vorgenommen, sondern mithilfe eines Fourier-Transform-Infrarotspektrometer vollzogen. Die Frequenzen werden dabei durch eine Fourier-Transformation von gemessenen Interferogrammen ausgerechnet und diese entweder direkt als Waveform in Audioinformationen übertragen oder fließen indirekt, durch Abbildung auf Tonhöhen und Laustärken einer vorgegebenen Waveform, in ein Tonstück ein. Delatour lässt die Anwendungsgebiete offen, gibt jedoch an, dass die produzierten Tonstücke sowohl in wissenschaftlichen als auch in künstlerischen Anwendungen Gebrauch finden könnten. Als Beispiel für eine wissenschaftliche Anwendung ist beispielsweise die Identifikation von chemischen Substanzen mithilfe von Tönen zu nennen. Den eher künstlerischen Anwendungsfall untersuchte auch Sturm [Stu01], der durch verschiedene Abbildungen versucht, Musik aus unterschiedlichen physikalischen Eigenschaften von Atomen zu komponieren.

5.2. Auditory Displays in der Chemie

Schon 1980 versuchte Yeung [Yeu80] die Vertonung von wissenschaftlichen Daten in der Chemie. In einem Versuch wurden verschiedene chemische Eigenschaften auf einen Ton gemappt. In ihrem Versuch konnten die Daten in neun Dimensionen unterschieden werden, indem diese auf verschiedene unterscheidbare Eigenschaften von Tönen abgebildet wurden. Die verwendeten Toneigenschaften waren die Tonfrequenz, Lautstärke, Dämpfung und Richtung. Eine Normierung dieser Eigenschaften muss vorgenommen werden, da die Wahrnehmung dieser Eigenschaften auch begrenzt ist. Ein zu leiser Ton oder eine zu hohe Frequenz kann nicht mehr wahrgenommen werden. Außerdem können manche Töne Unbehagen verursachen, wenn beispielsweise sehr hohe Frequenzen verwendet werden. In einem Versuch bildeten sie die Konzentration von verschiedenen Metallen in Obsidianproben auf die verschiedenen Eigenschaften der Töne ab. Nachdem die Studienteilnehmer ein kurzes Training erhielten, waren sie in der Lage, die zu unterscheidenden Vektoren in 90% der Fälle zu identifizieren. Nach einem weiteren Training konnte diese Rate auf 98% zu erhöhen.

Eine ähnliche Studie wurde 1982 von Bly [Bly82] durchgeführt, wobei noch weitere unterscheidbare Soundmerkmale eingeführt wurden und dadurch die möglichen Dimensionen der abzubildenden Daten

erhöht wurden. Die neu eingeführten Merkmale waren die grundsätzliche Form der Soundwaves, das Hinzufügen von Oberschwingungen sowie eine Variation der Anschwellzeit. Aus den Ergebnissen der Studie kann abgeleitet werden, dass auch diese neu eingeführten Merkmale unterscheidbar sind und sich grundsätzlich zur Sonifikation eignen.

Im Überblick über verschiedene auditive Displays führen Garzia-Ruiz et al. [GRGP06] auch einige Beispiele aus dem Bereich der Lehre aus. Zu den frühesten Ansätzen gehören dabei Audiohilfen für Menschen mit eingeschränkter visueller Wahrnehmung. Es gibt einige Geräte wie Multimeter, Thermometer oder Computerprogramme, die gemessene Werte in Sprache umwandeln. Interessanter für den Anwendungsfall zur Analyse der Molekulardaten sind jedoch jene Arbeiten, die auf die sprachliche Ebene verzichten. Lunney et al. [Lun94] entwickelten beispielsweise 1981 ein System, dass den pH-Wert einer Substanz auf die Frequenz eines Tones abbildet. Lunney et al. geben dabei an, dass durch Änderungen in der Tonfrequenz bestimmte Punkte in Datenreihen, wie beispielsweise Extrema im pH-Wert, mit einer Genauigkeit von 0.1% bestimmt werden konnten. Weiterhin schreiben die Forscher, dass sich Töne, die der menschlichen Sprache ähnlich sind, gut dazu eignen, bestimmte Muster in Daten aus chemischen Prozessen zu erkennen. Ob oder inwieweit diese Analyseart erst antrainiert werden muss, geben die Forscher jedoch nicht an. Sowohl Lunney [Lun94] als auch Berenato [BM97] verwenden die Anpassung der Tonhöhe zur Audifikation von verschiedenen wissenschaftlichen Instrumenten. Neben dem Gerät zur Audifikation des pH-Wertes entwickelte Lunney noch weitere Geräte die die Messungen von UV-Strahlen, Infrarotspektronomie sowie Chromazitätsmessungen durch den Einsatz von Tönen erleichtern bzw. für Menschen mit eingeschränkter Sicht erst ermöglichen. Berenato et al. testeten ein Gerät, dass die elektrochemische Leitfähigkeit einer Probe vertonen kann. Dies wurde durch Veränderung der Tonfrequenz in Abhängigkeit des elektrochemischen Widerstandes realisiert. Berenato beschreibt, dass dieses Gerät die Aufmerksamkeit der Studenten auf sich zog und es den Studenten mehr Spaß machte, mit diesem Gerät zu arbeiten, als mit der Variante, die nur ein visuelles Feedback gibt. Daraus lässt sich ein möglicher Anwendungsfall für die Vertonung der molekularen Simulationen für die Lehre begründen. Auf die Frage, ob dieser Effekt der verstärkten Aufmerksamkeit anhält oder nur kurzfristig wegen des Neuheitseffekts auftritt, geben die Forscher jedoch keine Antwort.

5.3. Sonifikation von DNA-Sequenzen

Erste Ansätze zur Sonifikation von DNA-Sequenzen wurden schon 1984 von Hayashi und Munakata [HK86] ausprobiert und fanden auch Verwendung in diversen Computerprogrammen zur Sequenzanalyse. Laut Munaka [Mun] waren deren Möglichkeiten jedoch durch die schlechte Audioqualität und Audioverarbeitung durch die beschränkte Hardware eingeschränkt, was weitere Entwicklungen in dieser Richtung verhinderte. Grund für die frühe Adaption von Audioausgaben bei der DNA-Sequenzierung waren vor allem die Möglichkeit, eine einfache Abbildung zwischen der Struktur einer DNA-Sequenz und Tonsequenz einer Audiospur zu erzeugen. Die einfachste Abbildung ist hierbei wohl das Zuweisen verschiedener Grundtöne zu den vier Basen und eine Variation der Tonhöhe, die von der Aminosäuresequenz abhängt (Siehe Abb. 5.1). Dadurch konnten einfach MIDI-Sequenzen erzeugt werden, die anschließend durch Synthesizerhardware in Töne umgewandelt wurden.

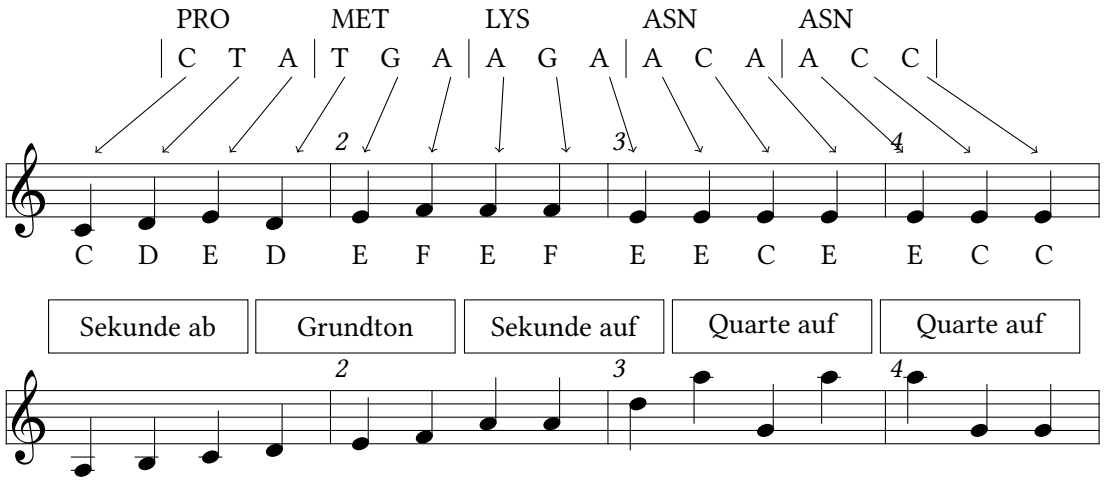


Abbildung 5.1.: Einfache Sonifikation einer DNA Sequenz. Die Nukleinbasen **A**denin, **G**uanin, **C**ytosin und **T**hymin werden den Grundtönen E, F, C und D zugeordnet. Anschließend wird der Ton abhängig von der Aminosäuresequenz variiert (PRO=Sekunde abwärts, MET=Grundton, LYS=Sekunde aufwärts, ASN=Quarte aufwärts.

Ein etwas fortgeschrittenes Mapping wenden King et al. [KA96] an, die noch andere Eigenschaften der Aminosäuren, wie beispielsweise Hydrophobisch, betrachten und aus diesen Eigenschaften eine Bassstimme ableiten. Zwar geben die Forscher an, ihren Algorithmus bei der Analyse verwendet zu haben, jedoch nicht ob die Tonauswertung erkennbare Verbesserungen dabei brachte.

Im Gegensatz zu diesen Sonifikation untersucht Won [Won05] einen konkreten Anwendungsfall. Das Finden von Cytosin-phosphatidyl-Guanin-Inseln (CpG-Inseln) wird dabei durch Variation der Tonhöhe eines Soundstücks erleichtert. Diese CpG-Inseln sind insofern für die Genomforschung von Interesse, da sie auf interessante Sequenzteile hinweisen. Inwieweit dieser Ansatz das Finden der CpG-Inseln vereinfacht, wird dabei jedoch nicht beschrieben. Weitere Ansätze, Anwendungsgebiete und eine weitere Implementierungen finden sich in [Sch11]. Schuchart listet hier mehrere Anwendungsfälle auf, bei denen die Sonifikation hilfreich sein könnte. Dazu gehören zum einen das Finden von Tandem Repeats, also Mustern, die wiederholt auftreten sowie das Erzeugen eines Audiofingerprints zum einfachen finden ähnlicher Genomsequenzen in einem Genombrowser.

Neben all diesen wissenschaftlichen Ansätzen finden sich außerdem noch etliche künstlerische Projekte. So kann über die Firma **Your DNA Song Ltd** [Ltd14] beispielsweise ein Musikstück bestellt werden, in dem die individuellsten Aminosäuresequenzen der eingeschickten DNA-Probe in ein angepasstes Musikstück einfließen.

Die Sonifikation von DNA-Sequenzen unterscheidet sich in einigen Punkten von der Audio-visuellen Darstellung, die in dieser Arbeit untersucht wird. Da die DNA-Sequenzen als String repräsentiert werden können, ist ein einfaches Mapping zwischen DNA-Sequenz und Audio-Sequenz möglich. Im Gegensatz dazu liegen die Daten der molekularen Simulationen nicht nur im dreidimensionallen Raum vor, sondern verändern sich über die Zeit. Bei der DNA-Analyse kann also grundsätzlich einfach eine visuelle Repräsentation zugezogen werden. Dies ist bei den molekularen Simulationsdaten nicht

ohne weiteres möglich. Dazu müsste man jene Aspekte, die man untersuchen möchte, entweder beim Rendern des Bildes mitzeichnen - was natürlich die ursprünglichen Daten überlagert - oder jedoch parallel dazu anzeigen, was den Nachteil hätte, dass zusätzliche Bildschirmfläche verloren geht und der Benutzer die getrennten Ausgaben mental zusammenführen muss. Die Erkenntnis, die aus den Experimenten mit der auditiven Analyse von DNA gewonnen werden kann, ist, dass Töne nicht nur als einzelne Indikatoren für Ereignisse dienen können, sondern auch als Teil einer größeren Melodie wahrgenommen werden und es somit erlauben sowohl Einzelheiten hervorzuheben als auch den Gesamtstatus eines Systems zu vermitteln.

6. Implementierung

Dieses Kapitel befasst sich mit der Implementierung des Sonifikationstools. Dabei wird die verwendete Audio API, sowie die Architektur des Audio-Frameworks beschrieben, welches für diese Arbeit entworfen wurde. Im zweiten Teil dieses Kapitel, geht es darum, wie dieses Frameworks in das *MegaMol*-Visualisierungstool eingebunden wurde und es wird an einem Beispiel gezeigt, wie die Vertonung von bestimmten Ereignissen mit diesem Framework möglich ist.

6.1. OpenAL

Wie bei der Erzeugung von graphischen Inhalten, für die meist eine Grafikkarte zuständig ist, wird auch die Erzeugung von Audioausgaben meist durch spezialisierte Hardware ermöglicht. Diese Soundkarten können sich je nach Hersteller und Preis in den angebotenen Funktionen stark unterscheiden. Dazu gehören verschiedene Auflösungen (z.B. 8bit oder 24bit), verschiedene Abtastfrequenzen (22 bis 192 kHz), unterschiedliche Anzahl an Kanälen für die Audioausgabe oder auch Onboard Speicher für bessere Performanz. Damit der Entwickler sich nicht mit diesen Unterschieden befassen muss, existieren einige APIs, die einen abstrahierten Zugriff auf die Hardware erlauben. Verfügbare APIs sind beispielsweise *XAudio2* für Windows, XBox und Windows Phone, *ALSA* für Linux sowie *FMOD* und *OpenAL* mit Unterstützung für Windows, Linux, Unix, OS X, Android, XBox, Playstation und iOS. Im Gegensatz zu OpenGL wird OpenAL nicht von einem Gremium (Khronos Group bei OpenGL) entwickelt und gemanaged, sondern wurde zuerst von Loki Software entwickelt und später von Creative Technology übernommen. Im Gegensatz zu FMOD unterliegen jedoch große Teile des API unter der freien und quelloffenen GNU Lesser General Public License vor und es gibt für die meisten Plattformen kostenlose Implementierungen, weshalb die Wahl auf dieses Audioframework fiel.

Wie auch OpenGL, in dessen Anlehnung der Name gewählt wurde, ist OpenAL als Zustandsautomat modelliert, bei dem die Aufrufe in einem Stack landen und nacheinander abgearbeitet werden. Konzeptionell gibt es in OpenAL ein *Device*, *Context* und *Listener* sowie mehrere *Sources* und *Buffers* für die Sounds. Das *Device* ist die Schnittstelle zur Audiohardware, während der *Context* den aktuellen Zustand von OpenAL beschreibt. Der *Listener* ist das Gegenstück der Kamera in OpenGL. Er beschreibt die Position, an der das virtuelle Mikrofon platziert wird, die maximale Lautstärke und eine Richtung. Um Töne abspielen zu können benötigt man die sogenannten *Sources*. Jedes *Source* Objekt beschreibt eine Audioausgabe und deren Position, Richtung, Geschwindigkeit, Tonhöhe, Lautstärke und die Tatsache ob der Ton einmal abgespielt wird oder solange wiederholt wird, bis er aktiv gestoppt wird. Die eigentlichen Audiodaten werden als *Pulse-Code-Modulation* in den *Buffern* gespeichert. Die Generierung eines räumlichen Eindrucks wird bei OpenAL durch verschiedene Distanzmodelle, durch Unterstützen des Dopplereffekts sowie einer HRTF und ITD (siehe Kapitel 4.3) ermöglicht.

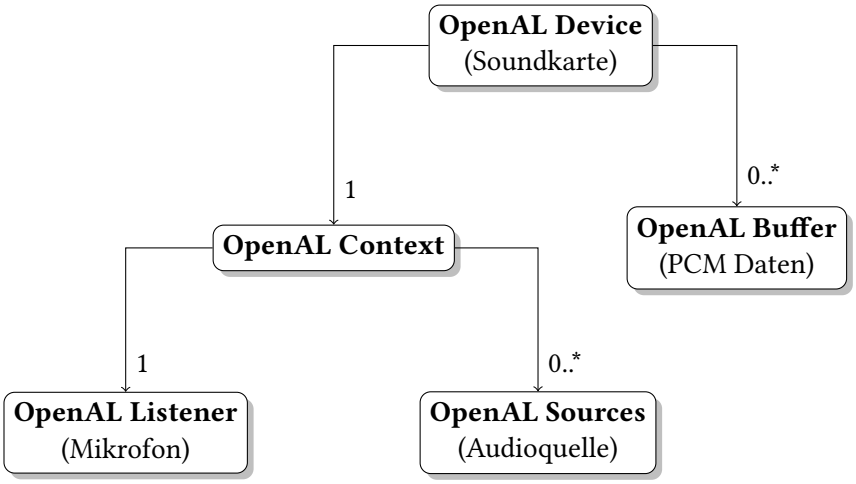


Abbildung 6.1.: Komponenten von OpenAL und deren Zusammenspiel [ope].

6.2. Audiowrapper

Wie auch bei OpenGL macht es auch bei OpenAL Sinn, die zur Verfügung gestellten Schnittstellen in entsprechende Klassen zu verpacken und gleiche Abläufe zu automatisieren. So ist beispielsweise das Laden von Audiodateien (siehe Listing A.1) ein relativ aufwändiger Vorgang und sollte deshalb nicht für jeden Sound neu geschrieben werden. Die Struktur die dieses Wrapping besitzt und die Funktionalitäten der einzelnen Klassen sollen in diesem Kapitel vorgestellt werden.

Der Aufbau besteht hauptsächlich aus vier Klassen (siehe Abbildung 6.2): Dem AudioManager, SoundRessourceManager und den zwei Klassen MonoSound und StereoSound. Der AudioManager stellt das Herzstück des Wrappers dar und implementiert das *Device*, den *Context* und den *Listener*. Da von diesen jeweils nur eine einzige Instanz existiert, wurde der AudioManager als Singleton implementiert um somit einen einfachen globalen Zugriff auf diese Instanzen zu erhalten. Die erste Aufgabe des AudioManagers besteht darin, das *Device* zu initialisieren. Ein System kann durchaus mehrere Soundkarten installiert haben, weshalb eine Auswahl der von OpenAL verwendeten Soundkarten möglich sein soll. Dies geschieht mit Hilfe einer Konfigurationsdatei, welche die verfügbaren Soundkarten auflistet und eine Auswahl der zu verwendeten Karte erlaubt (siehe Listing A.2). Falls beim Start des Programms noch keine Konfigurationsdatei existiert, wird diese erstellt und eine Standardkarte ausgewählt. Nachdem das *Device* und der *Context* erstellt wurden, erlaubt es das Interface, die Position und den Lautstärkepegel des *Listeners* zu konfigurieren. Die zweite Aufgabe des AudioManagers besteht darin, die *Sources* zu verwalten. In der Regel können sehr viele Audiodateien in *Buffer* geladen, jedoch nur eine bestimmte Anzahl dieser Audiodateien über die zahlenmäßig beschränkten *Sources* wiedergegeben werden. Die Anzahl der verfügbaren *Sources* ist von der Soundkarte abhängig und kann deshalb von System zu System variieren. In der Regel sind dabei mindest eine Audioquelle für Stereotöne und 16 Audioquellen für Monotöne verfügbar. Zum Abspielen eines Tones muss dabei immer der entsprechende *Buffer* an eine passende *Source* gebunden werden. Um die *Sources* also dynamisch an jene Sounds zu übergeben, die abgespielt werden sollen, implementiert der AudioManager eine Verwaltung der *Sources*. Dazu wird überprüft, wenn ein Soundobjekt abgespielt werden soll, ob es zur Zeit schon ein *Source* Objekt besitzt (siehe Abb. A.1). Ist dies nicht der Fall, wird vom

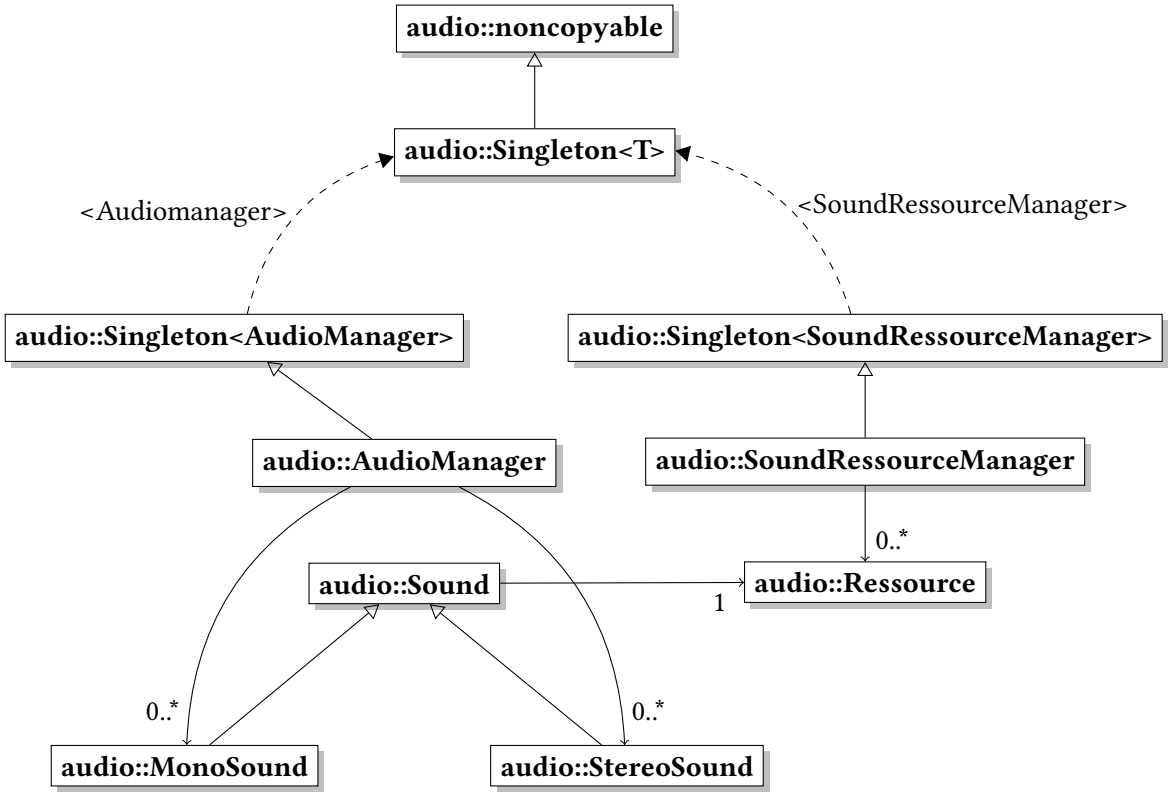


Abbildung 6.2.: Der Aufbau und Inhalt des Audiowrappers.

Audiomanager ein solches *Source* Objekt angefordert. Dieser versucht erst ein neues *Source* Objekt zu erstellen und gibt dieses gegebenenfalls zurück. Ist es nicht möglich, ein neues Objekt zu erstellen, werden alle Soundobjekte, die zur Zeit ein *Source* Objekt besitzen, angefragt, ob dieses gerade benötigt wird. Falls dabei auch kein freies *Source* Objekt verfügbar wird, muss nach einer bestimmten Metrik vorgegangen werden, um zu entscheiden, welches Audioquelle gestoppt werden kann. Mögliche Metriken sind beispielsweise die Entfernung der Audioquelle vom Mikrofon oder die Zeit, die das Soundobjekt schon im Besitz der Audioquelle ist. In diesem Fall wurde der zweite Ansatz gewählt, da davon ausgegangen werden kann, dass die meisten Töne gegen Ende leiser werden und deren Stopp dadurch nicht hörbar wird. Außerdem hat diese Methode den Vorteil, dass sie bei jedem Durchlauf das gleiche Ergebnis produziert, während die erste Methode davon abhängt, wo sich die Kamera zur Zeit befindet.

Die zweite Klasse die wie der Audiomanager von der Klasse Singleton erbt und damit global zugänglich ist, ist der SoundRessourceManager. Dieser dient dazu, die Audiodateien zu laden, ein Vorladen der Audiodateien zu ermöglichen und zu verhindern, dass eine Datei mehrfach geladen wird. Dazu implementiert er eine Hashmap, welche die geladenen Sounds und Referenzzähler für diese beinhalten. Soll eine Audiodatei geladen werden, fordert das entsprechende Soundobjekt vom SoundRessourceManager einen *Buffer* an, der die PCM Daten dieser Audiodatei beinhaltet (siehe Abb. A.2). Falls der SoundRessourceManager die Datei schon geladen hat, gibt er einfach einen Zeiger auf die Daten zurück und inkrementiert den entsprechenden Zähler, ansonsten wird die Datei geladen

und in die Hashmap eingefügt. Wenn ein Sound-Objekt zerstört wird, schickt es eine Nachricht an den Ressourcen Manager, welcher diesen Zähler dekrementiert und für den Fall, dass kein Soundobjekt mehr die Datei braucht, den entsprechenden *Buffer* freigibt.

Zusammen ermöglichen diese beiden Klassen also das einfache Laden und Abspielen von Sounds. Es reicht aus, ein entsprechendes MonoSound oder StereoSound Objekt mit dem Pfad der Audiodatei zu erzeugen und auf diesem die *play()* Methode aufzurufen. Die Soundklasse fasst dabei die Eigenschaften zusammen, die bei allen Sounds auftreten. Dazu gehören das Laden und Entladen der Audiodatein sowie das Setzen der Lautstärke und das Variieren der Tonhöhe. Die beiden Klassen MonoSound und StereoSound erben von dieser Sound Klasse und implementieren die Methoden *play()* und *loopPlay()*, welche den Sound solange abspielt, bis er mit Hilfe der *stop()* Methode wieder angehalten wird. Zusätzlich implementiert die MonoSound Klasse die Methoden *setPosition()*, *setOrientation()* und *setVelocity()*, welche es erlauben, der Audioquelle einen räumlichen Eindruck zu verschaffen. Dabei wird die Position und Ausrichtung der Audioquelle verwendet, um mittels Lautstärkeunterschieden und *HRTF* die Richtung der Audioquelle zu simulieren, während die Bewegungsrichtung mittels Dopplereffekt simuliert wird.

6.3. Megamol

Der vorgestellte Audiowrapper ist unabhängig von *MegaMol* selbst und kann auch in anderen Projekten eingesetzt werden, erlaubt jedoch eine einfache und modulare Lösung, um Sounds in *MegaMol* zu implementieren. *MegaMol* selbst ist extrem modular aufgebaut. Es besteht aus einem Frontend, welches die Interaktion mit dem Nutzer erlaubt, dem Kernframework, welches viele Funktionen und Werkzeuge zum Rendering, anzeigen von Menüs, dem Laden und Schreiben von Daten und vieles mehr bietet. Außerdem verwaltet dieser Kern die Plugins und die Aufrufe zwischen verschiedenen Modulen. Die Plugins dienen dazu, spezialisierte Funktionen zu ermöglichen wie zum Beispiel das Rendern von Proteinen oder, wie in diesem Fall, die Verwendung von Sounds. Jedes Plugin besteht aus einer Reihe von Modulen und *Callern*. Diese Module können dann vom Kernframework initialisiert und mit Hilfe der *Caller* aufgerufen werden.

Jedes Modul und jeder *Caller* besitzt dabei einen eindeutigen Klassennamen, über die diese identifiziert werden können und eine textuelle Beschreibung, was das jeweilige Modul bzw. der *Caller* genau macht.

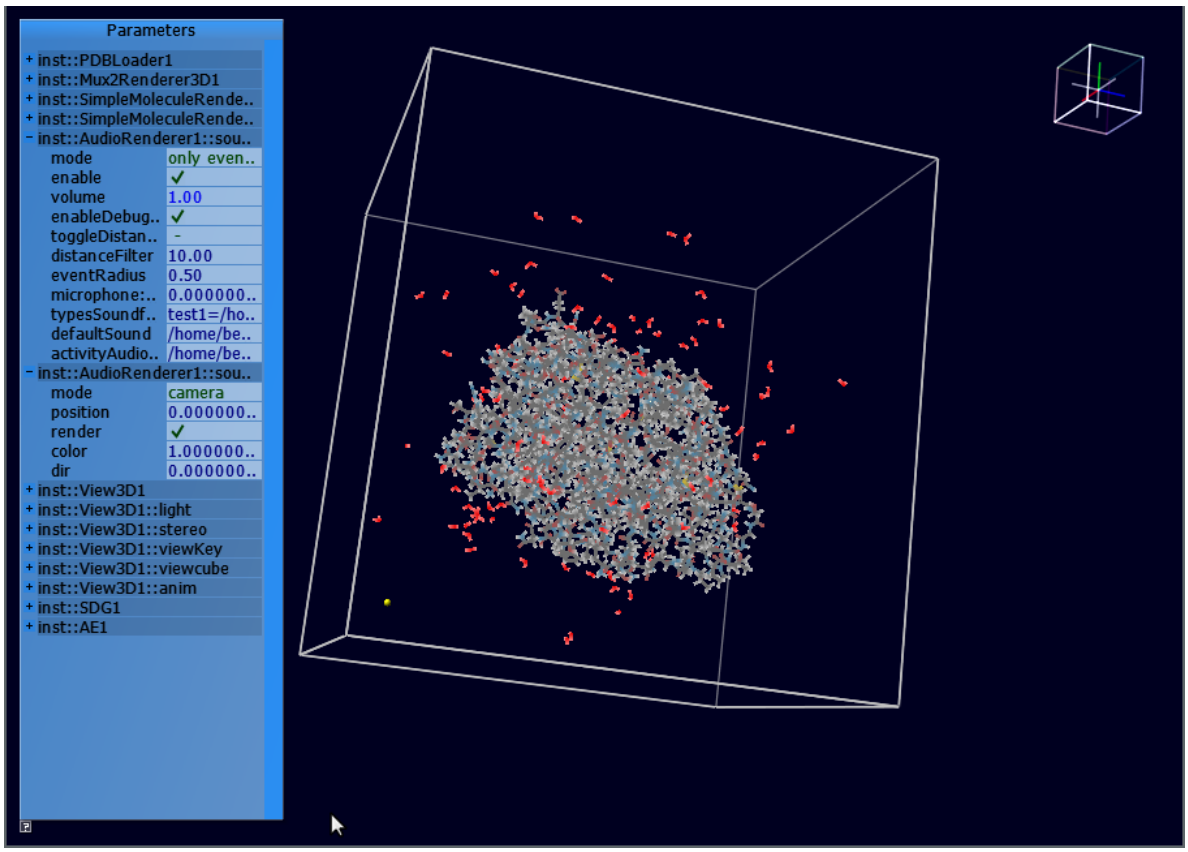


Abbildung 6.3.: Einstellungsmöglichkeiten des Audiorenderers.

Module können zudem noch eine Reihe verschiedener Slots besitzen:

- Parameter Slots: Diese Slots veröffentlichen die verschiedenen Einstellungen, die an einem Modul vorgenommen werden können. Beim Instanziieren eines Objekts werden die Einstellungsmöglichkeiten beim Kernmodul registriert. Ein Verändern der Einstellung ist entweder über die Projektdatei (siehe unten) oder über ein grafisches Menü möglich (siehe Abb. 6.3). Die grafische Oberfläche unterstützt dabei eine Reihe verschiedener Parameter wie beispielsweise Dropdown-Listen, Farben, Positionen uvm.
- Caller Slots: Durch diese Slots wird angegeben, welche *Caller* aufgerufen werden können. Zusammen mit den Callee Slots geben diese Slots also an, mit welchen *Callern* die verschiedenen Module verbunden werden können. Über den Caller Slot kann ein Modul die Instanz des *Callers*, die an diesem Modul hängt, vom Kernframework anfordern um auf diesem dann die Methoden aufzurufen, welche der *Caller* unterstützt.
- Callee Slots: Die Callee Slots beschreiben, durch welche *Caller* das Modul aufgerufen werden kann und registriert die entsprechenden *Callbacks*.

Die Beschreibung davon, welche Module beim Start von *MegaMol* geladen werden sollen und wie diese zusammenhängen, wird in einer Projektdatei realisiert, welche eine XML-Beschreibung der

verwendeten Module und *Caller* beinhaltet. Diese kann sowohl manuell als auch über ein grafisches Konfigurationstool erzeugt werden. Dabei wird zwischen zwei grundsätzlich verschiedenen Projektarten unterschieden: den *Jobs* und den *Views*. *Jobs* werden immer dann benutzt, wenn eine Aufgabe durchgeführt wird, die keine Interaktion mit dem Benutzer erfordert, sondern einfach automatisch abläuft. Eine beispielhafte Projektdatei, welche einen Job beschreibt der Audioevents, welche zuerst aus einer anderen Datei geladen werden, in eine Datei schreibt, findet sich in Listing A.2. Dies wird beispielsweise benötigt, um kompliziertere Berechnungen nicht zur Laufzeit durchführen zu müssen, sondern um diese vorberechnen zu können. In dem angegebenen Beispiel werden die Daten einfach aus einer schon vorhandenen Audiodatei geladen. Enthält die Projektbeschreibung einen Job, wird dieser, nachdem alle Module instanziiert wurden, einmalig gestartet und das Programm läuft so lange, bis dieser Job beendet wurde oder einen Fehler wirft. *Views* werden von Anwendungen benutzt, bei denen eine Interaktion mit dem Nutzer stattfindet. Das Listing A.1 enthält ein Beispiel, bei dem der *View* aus einem MultiplexRenderer besteht. Dieser wird für jedes zu renderende Frame aufgerufen und ruft seinerseits nacheinander zwei Renderer auf.

Für die Daten, die *MegaMol* zur Visualisierung benutzt, können durch verschiedene Module unterschiedliche Formate unterstützt werden. Intern wird in vielen Modulen jedoch ein framebasiertes Partikelformat verwendet. Das bedeutet, dass für äquidistante Zeitabschnitte die aktuelle Position und weitere Eigenschaften der einzelnen Teilchen gespeichert wird. Bei der Visualisierung können diese Eigenschaften dann interpoliert werden, um ein flüssiges Bild zu generieren.

6.4. Architektur

In diesem Abschnitt werden die implementierten Module und *Caller* vorgestellt.

6.4.1. AudioDataCall

Die AudioDataCall Klasse repräsentiert alle Ereignisse, die während einem Frame auftreten. Sie besteht zum einen aus den AudioEvents welche für jedes Event dessen Position, Soundtyp, Lautstärke, Tonhöhe, Bewegungsrichtung sowie den genauen Anfangszeitpunkt beinhalten und zum anderen aus dem AudioDataHeader. Dieser speichert eine Abbildung des Soundtyps auf dessen Audiodatei und meldet ob diese Datei nur einmal abgespielt werden soll oder wiederholt werden soll und beinhaltet noch weitere allgemeine Informationen wie beispielsweise die aktuelle Framenummer und Zeit, die dieser AudioDataCall repräsentiert. Der AudioDataCall bietet außerdem die Möglichkeit, für jedes Frame ein *Activity Level* anzugeben. Dieses kann dazu verwendet werden eine bestimmte Eigenschaft der Events auf einem Dauerton durch Verändern der Lautstärke oder der Tonhöhe zu vermitteln.

6.4.2. AudioRenderer

Der AudioRenderer ist das Modul, welches die Benutzung des Audio Wrappers in *MegaMol* ermöglicht. Es erbt die Eigenschaften eines 3D-Renderer Moduls und besitzt als solches zwei wichtige Funktionen: die *GetExtend()* Methode gibt zurück, wie viele Frames verfügbar sind und in welchem Bereich sich die

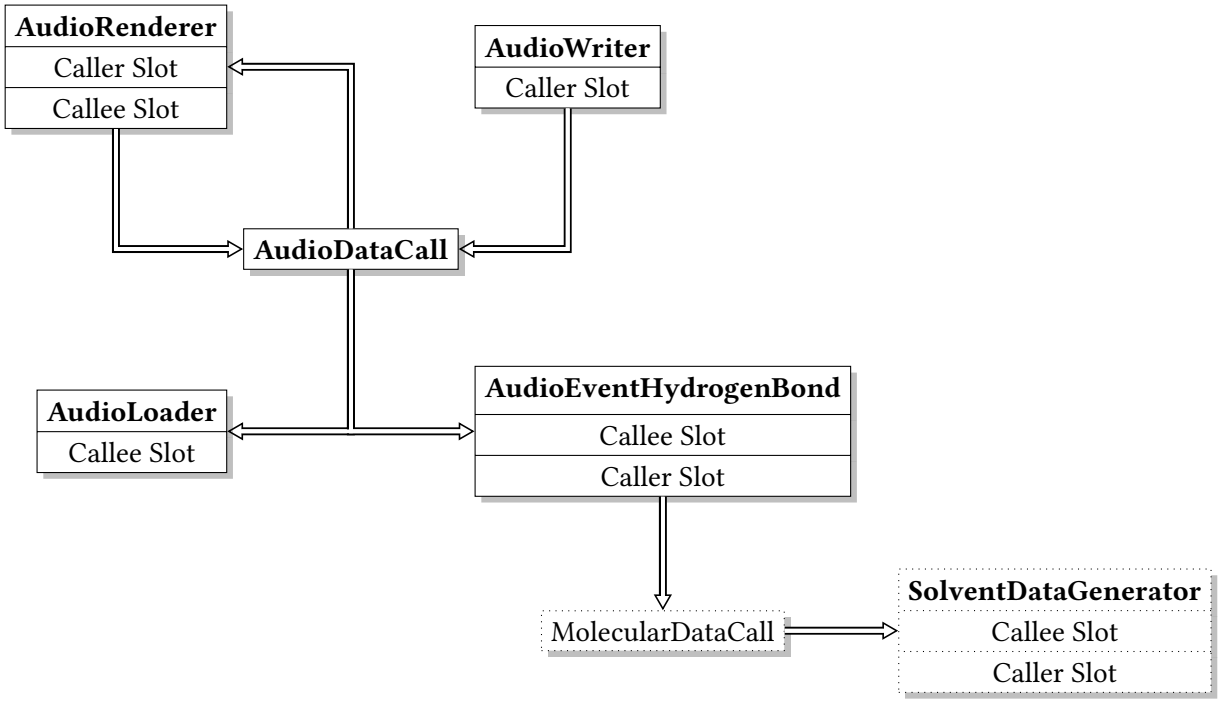


Abbildung 6.4.: Implementierte Module und *Calls* und deren Zusammenspiel.

Teilchen bewegen. Die zweite Methode ist die *Render()* Methode. Nachdem vom Kernmodul die Anzahl der Frames ermittelt wurde, die abgespielt werden sollen, wird diese Methode einmal pro Frame aufgerufen. Dabei werden das aktuell zu renderende Frame sowie die genaue Zeit übergeben, welche dazu benutzt werden kann, zwischen den einzelnen Frames zu interpolieren. Der AudioRenderer verfügt über einen Caller und einen Callee Slot, an denen jeweils ein AudioDataCall angeschlossen werden kann. Mit Hilfe des Caller Slots kann der AudioRenderer das aktuelle Frame von einem AudioDataCall anfordern, um die die darin enthaltenen Events abzuspielen. Der Callee Slot hingegen dient dazu, diese Daten für ein weiteres Modul zugänglich zu machen. So kann beispielsweise ein AudioWriter diese Daten abspeichern. Beim Aufruf der Rendermethode wird überprüft, ob für das aktuelle Frame schon die entsprechenden Events durch den AudioDataCall angefordert und zwischengespeichert wurden und falls dies nicht der Fall war, wird dies durchgeführt. Aus der Liste der Events werden dann jene ausgewählt, deren Anfangszeitpunkt vor der Aufrufzeit liegen. Für jedes dieser Events wird überprüft, ob zur Zeit schon ein Soundobjekt existiert, welches dieses Event repräsentiert. Ist dies der Fall, werden die Eigenschaften wie Position oder Lautstärke des Soundobjekts an die des neuen Events angepasst. Für den Fall, dass kein solches Soundobjekt existiert, wird ein neues erzeugt, welche diese Events repräsentieren und auf dem Soundobjekt anschließend die Wiedergabe des Tons gestartet. Falls der Benutzer eine entsprechende Option ausgewählt hat, wird dann außerdem mit Hilfe des *Activity Levels* die Tonhöhe oder Lautstärke eines Tones angepasst, welcher in einer Dauerschleife abgespielt wird.

Um dem Benutzer ein Anpassen des AudioRenderers an seine Bedürfnisse zu erlauben, verfügt der AudioRenderer außerdem über eine Reihe an Einstellungsmöglichkeiten (siehe Abb. 6.3), welche in Tablle 6.1 aufgeführt werden.

Parameter	Option	Effekt
mode	only events	Nur Töne bei Events abgespielt.
	only activity	Es wird nur das Aktivitätslevel vertont.
	activity & events	Events und Aktivitätslevel werden vertont.
enable	bool	Audioausgabe ein-/ausschalten.
volume	0.0 - 1.0	Lautstärke aller Audioausgaben anpassen.
enableDebugRendering	bool	Ermöglicht eine Visualisierung der Audioevents.
toggleDistanceFilter abgespielt werden,	bool	Aktiviert einen Filter, bei dem nur Sounds die nahe genug am Mikrofon sind.
distanceFilter	float	Setzt die Distanz des Filters.
eventRadius	float	Setzt den Radius der Kugeln, die für die Debugausgabe verwendet werden.
typesSoundfiles	string	Mapping von Audiotypen auf die Audiodateien (z.B hBondCreation=/path/to/audio.wav)
defaultSoundfile	string	Audiodatei, die verwendet werden soll, falls kein Audiofile für ein Event angegeben wurde.
activityAudioFile	string	Pfad zu der Audiodatei, die zum Vertonen des Aktivitätslevels dienen soll.
microphone::mode	camera	Die Kamera ist das virtuelle Mikrofon.
	marker	Ein frei platzierbares Mikrofon
microphone::position	vec3	Die Position des Mikrofons
microphone::dir	vec3	Die Richtung des Mikrofons
microphone::up	vec3	Der Up-Vektor des Mikrofons
microphone::render	true/false	Anzeigen des Markers

Tabelle 6.1.: Einstellungsmöglichkeiten des AudioRenderers

6.4.3. AudioEventHydrogenBond

Das Modul AudioEventHydrogenBond dient zur Generation von Audio Events bei Wasserstoffbrückenbildung bzw. der Auflösung dieser und wird in Kapitel 8 genauer beschrieben.

6.5. Vorausberechnung

Um auch Events zu unterstützen, deren Berechnung nicht in Echtzeit während der Visualisierung möglich ist, soll versucht werden, diese Events im Voraus zu berechnen und in einer Datei zu speichern. Während der Visualisierung soll diese Datei dann geladen und die entsprechenden Events im richtigen Moment abgespielt werden. Um dies zu realisieren, wurden der AudioWriter und der AudioLoader implementiert.

Listing 6.1 AudioFormat.ax

```
1 Frames 500
2 ContainsMovement 0
3 MinFloats 0 0 0
4 MaxFloats 1000 1000 1000
5 SoundType 0 hbondCreate /home/benjamin/studium/bachelor/bubble.wav
6 SoundType 1 hbondRelease /home/benjamin/studium/bachelor/clong.wav
7 HEADER_END
8 F 0 51 0.8
9 1703011 0 15.59 42.97 25.28 1 0.0774233
10 22601823 0 25.85 15.89 55 1 0.0835414
11 ...
12 F 1 48 0.7
13 ...
```

6.5.1. AudioWriter

Beim AudioWriter handelt es sich um einen Job. Dies bedeutet, dass er ohne Interaktionsmöglichkeiten gestartet werden kann und dann von alleine durchläuft. Der Benutzer hat zwar noch immer einige Einstellungsmöglichkeiten, wie z.B den Dateipfad und -name der zu speichernden Datei oder die maximale Anzahl an Frames, die in diese Datei gespeichert werden soll, jedoch müssen diese Einstellungen schon im voraus festgelet werden und können, während das Programm läuft, nicht mehr geändert werden. Beim Start des Jobs wird dazu über den Caller Slot, an welchen ein AudioDataCall angeschlossen ist, ermittelt, wie viele Frames gespeichert werden sollen. Anschließend wird die Datei angelegt und der Header des AudioDataCalls in diese geschrieben (Zeile 1-7 in Listing 6.1). Dieser beschreibt, wie viele Frames in der Datei gespeichert sind, ob für die AudioEvents nur dessen Position oder auch eine Bewegungsrichtung gespeichert wurde, die Bounding-Box, in der sich alle Events befinden, sowie die verschiedenen Soundtypen mit deren Identifikatoren, Namen und AudioDatei. Anschließend werden die einzelnen Frames abgespeichert, wobei einzelne Frames durch einen Marker separiert werden (siehe Zeile 8 und 12). Jeder Marker speichert dabei ab, um welches Frame es sich handelt, gibt die Anzahl der AudioEvents in diesem Frame und das Activity Level dieses Frames an. Die einzelnen Einträge der AudioEvents sind dann wie folgt aufgebaut:
EventID SoundTypeID Pos.X Pos.Y Pos.Z [Mov.X Mov.Y Mov.Z] Volume Time
Dieses auch von Menschen einfach lesbare Format wurde gewählt, um ein Testen des AudioRenderers zu ermöglichen, ohne eine AudioEvent-Generierung implementieren zu müssen und weil es eine einfache Überprüfung der Plausibilität der generierten Daten ermöglicht. Ein Umstieg auf ein binäres Format, um Platz zu sparen und die Schreib- und Lesegeschwindigkeiten zu erhöhen, ist jedoch ohne großen Aufwand realisierbar.

6.5.2. AudioLoader

Der AudioLoader ist das Gegenstück des AudioWriters und ermöglicht es, die von diesem erzeugten Dateien wieder einzulesen. An seinem Callee Slot kann dementsprechend ein AudioDataCall angeschlossen werden, um diese Daten anzufordern.

6.6. Aufgetretene Probleme

Ein Problem betrifft die Verwendung von Audio APIs im Allgemeinen. Wie oben beschrieben, gibt es nur wenige freie 3D-Sound APIs wobei, von allen verfügbaren APIs, OpenAL noch über die meisten Features. HRTFs finden sich beispielsweise nicht einmal bei vielen kommerziellen Produkten. Das Problem mit OpenAL besteht darin, dass die Spezifikationen nicht weiterentwickelt werden. Die Spezifikation zu OpenAL 1.0 wurde im Jahr 2000 veröffentlicht und das letzte Update (Version 1.1) stammt aus dem Jahr 2005 [Lab]. Die Implementierung dieses Standards wird zwar durch OpenAL Soft noch immer aktiv weiterentwickelt, jedoch ist die Entwicklung nicht vergleichbar mit OpenGL. So gibt es außer der Spezifikation kaum offizielle Anleitungen zur Benutzung und auch nur einige inoffizielle Tutorials, die sich meist auch noch stark unterscheiden, da unterschliedliche Bibliotheken für das Laden und Streamen von Sounds verwendet werden.

Da sich *MegaMol* noch in der Entwicklung befindet, existiert auch hier nur wenig Dokumentation. So findet sich auf der *MegaMol* Website [meg] zwar eine ausführliche Anleitung zum Builden des Projekts, jedoch keine Anleitung, wie neue Module und Jobs implementiert werden können. Außerdem findet man ab und zu noch kleinere Fehler, die erst auffallen, wenn vorhandene Module mit neuen Modulen verbunden werden sollen.

7. Eventerkennung

7.1. Wasserstoffbrückenbildung

Um das Audioframework zu testen, wurde die Erkennung von Wasserstoffbrücken verwendet und für das Entstehen und Auflösen von Wasserstoffbrücken entsprechende Events generiert. Wasserstoffbrücken sind für molekulardynamische Simulationen interessant, da sie zum einen einem Protein Stabilität verleihen können und somit dabei helfen, dessen Struktur aufrecht zu erhalten. Auch das Blockieren von Bindungsstellen an der Oberfläche eines Proteins durch an diesen Stellen angelagertes Wasser lässt sich durch das Betrachten von Wasserstoffbrücken erkennen. Wasserstoffbrücken können immer dann auftreten, wenn bei einem positiv geladenen Wasserstoffatom, welches bereits eine Bindung mit einem Atom (*Donor*) eingegangen ist, eine anziehende Wirkung zwischen dem Wasserstoffatom und einem weiteren negativ geladenen Atom (*Acceptor*) stattfindet [DS01] (siehe Abb. 7.1). Für das MegaMol Projekt existierte schon das Modul *SolventDataGenerator*, welches eine Liste mit wahrscheinlichen Wasserstoffbrücken generiert. Um herauszufinden, ob zwischen einem Acceptor und Donor eine Wasserstoffbrücke existiert, können verschiedene Kriterien angewandt werden. Eine unkomplizierte und schnelle Überprüfung kann man durch das von Berndt et al. [Ber96] beschriebene Abstand-Winkel Kriterium vornehmen. Eine Wasserstoffbrücke wird dann angenommen, wenn der Acceptor und der Donor nicht weiter als 2.4 Ångström voneinander entfernt sind und der *Hydrogen-Donor-Acceptor* Winkel kleiner als 35° ist. Für einen ersten Test wurden für die

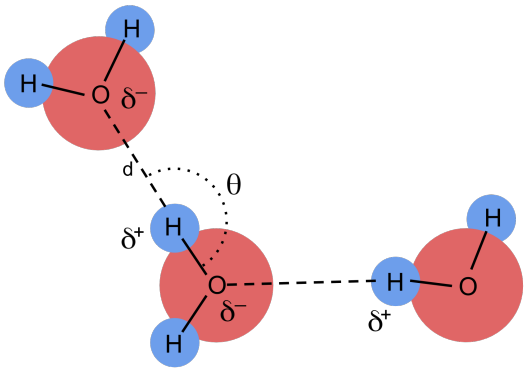


Abbildung 7.1.: Wasserstoffbrücken zwischen mehreren Wassermolekülen. Das rechte Sauerstoffatom ist ein Donor, das links oben ein Acceptor. Das mittlere Sauerstoffatom nimmt sowohl die Rolle des Donors sowie die des Acceptors ein. θ beschreibt den *Hydrogen-Donor-Acceptor* Winkel, d die *Hydrogen-Acceptor* Distanz.

Bestimmung der AudioEvents eines Frames einfach die Wasserstoffbrücken dieses Frames sowie jene des darauf folgenden Frames verwendet. Falls eine neue Wasserstoffbrücke auftritt oder sich eine löst, wird ein entsprechendes AudioEvent generiert. In einer zweiten Version wurden dann die genauen Zeitpunkte, wann sich die Wasserstoffbrücken bilden oder auflösen, berechnet, damit nicht alle Events zum Anfang des Frames erzeugt werden. Dazu wird die Entfernung der Winkel zwischen den Atomen, an denen sich Wasserstoffbrücken bilden, linear zwischen den Frames interpoliert und an diesen interpolierten Eigenschaften der genaue Zeitpunkt für ein entsprechendes Event berechnet. Außerdem wurde ein *Activity Level* und ein Filter eingeführt. Das *Activity Level* soll anzeigen, wie viele Wasserstoffbrücken zur einem bestimmten Frame existieren. Dazu werden beim ersten Aufruf des Moduls die Wasserstoffbrücken aller Frames vorberechnet und zwischengespeichert. Das *Acitivity Level* eines Frames i kann dann mit Hilfe der minimalen und maximalen Anzahl an Wasserstoffbrücken in allen Frames mit folgender Formel auf das Einheitsintervall $[0, 1]$ abgebildet werden:

$$Activity\ Level_i = \frac{\#WB_i - \min\limits_{\forall f \in Frames} \#WB_f}{\max\limits_{\forall f \in Frames} \#WB_f - \min\limits_{\forall f \in Frames} \#WB_f}, \tag{7.1}$$

wobei $\#WB_i$ die Anzahl der Wasserstoffbrücken des Frames i beschreibt. Der Buffer, in dem die vorberechneten Wasserstoffbrücken gespeichert werden, wird außerdem dazu verwendet um für jede Wasserstoffbrücke zu überprüfen, wie lange diese existieren wird. Diese Länge erlaubt es die Lautstärke eines Events mit Hilfe zweier Grenzwerte anzupassen. Die Lautstärke L_i eines Events, welches das Erzeugen oder Auflösen einer Wasserstoffverbindung W_i anzeigt, wird also über die Dauer δ_i die diese Bindung existiert, sowie der unteren Grenze G_u und der oberen Grenze G_o mit folgender Formel berechnet:

$$L_i = \begin{cases} 0 & , \delta_i < G_u \\ \dfrac{\delta_i - G_u}{G_o - G_u} & , G_u \leq \delta_i < G_o \\ 1 & , G_o \leq \delta_i \end{cases} \tag{7.2}$$

Der Einsatz dieses Filters macht beispielsweise Sinn, falls Wasserstoffbrücken, die sich durch oszillieren der Atome immer wieder bilden und auflösen, nicht berücksichtigt werden sollen. Zwei weitere Einstellmöglichkeiten dieses Moduls ermöglichen dem Benutzer außerdem eine bequemere Auswahl der zu verwendeten Audiodateien für die beiden Events.

7.2. Weitere Möglichkeiten

Die Sonifikation ist vor allem dann sinnvoll, wenn etwas nicht leicht auf einem Bild sichtbar ist oder um die Aufmerksamkeit des Benutzers auf ein bestimmtes Ereignis zu lenken. Da es bei der Visualisierung der Molekulardaten meist nur möglich ist, einen Teil der Daten zu betrachten, gibt es noch etliche weitere Einsatzmöglichkeiten für die Verwendung des AudioRenderers. In diesem Abschnitt sollen deshalb einige weitere Events beschrieben werden, bei denen die Sonifikation sinnvoll sein könnte und beschrieben werden, wie diese umgesetzt werden können.

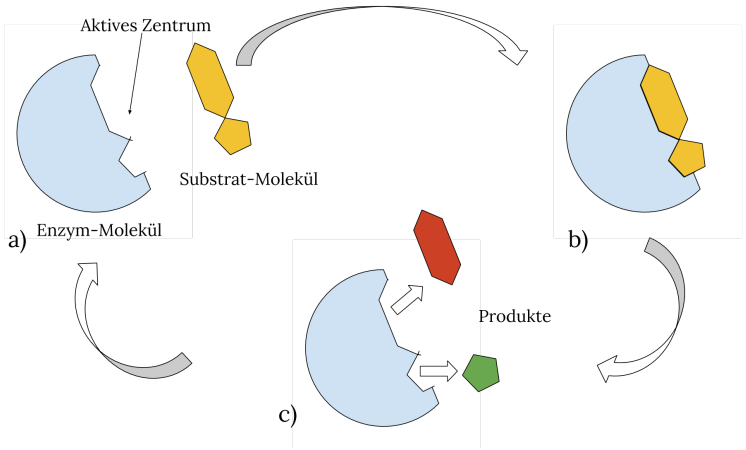


Abbildung 7.2.: Schematische Darstellung einer Substrat-Zerlegung mit Hilfe der Bindungstasche eines Enzyms. a) Freies Enzym und Substrat. b) Substratbindung an das Enzym und infolgedessen stattfindende Reaktion (Spaltung). c) Ablösen der Produkte

7.2.1. Anlagerung eines Lösungsmittels

Die Anlagerung eines Lösungsmittel an einem Protein ist interessant, da die physikalischen und funktionalen Eigenschaften der Proteine abhängt von deren Lösungsmittelumgebung und der Art und Weise, wie das Lösungsmittel mit dem Protein reagiert [Gre95]. Auch für die Fällung, also dem Ausscheiden eines Proteins aus seiner Lösung, ist es wichtig, die Anlagerung des Lösungsmittels an das Protein zu untersuchen. Analog zu den Wasserstoffbrücken können Events für das Anlagern oder Lösen des Lösungsmittels an der Proteinoberfläche definiert werden. Dazu kann man einfach die Umgebung des Proteins mit einem bestimmten Abstand auf Lösungsmittelatome hin überprüfen. Auf das *Activity Level* kann entweder die Anlagerungsmenge selbst oder aber die Anlagerungsrate abgebildet werden.

7.2.2. Bindungsstellen und Bindungstaschen

Bindungsstellen beschreiben bestimmte Stellen in einer molekularen Struktur, z.B der eines Proteins, RNA- oder DNA-Kette, an der sich andere Strukturen, deren molekularer Aufbau zu diesen Stellen passt, verbinden können. Bindungstaschen beschreiben eine spezielle Klasse dieser Bindungsstellen. Diese Art von Bindungsstellen bilden beispielsweise das aktive Zentrum eines Enzyms. Diese aktiven Zentren wirken dabei als Katalysator, um ein Substrat in einzelne Produkte zu zerlegen [Ver99]. Abbildung 7.2 zeigt die schematische Darstellung dieses Vorgangs. Die Erforschung und Simulation dieses Vorgangs spielt beispielsweise bei der Herstellung von Arzneimitteln eine wichtige Rolle [ZD08].

Im Gegensatz zur Wasserstoffbrückenbildung tritt eine solche Bindung seltener auf, weshalb ein auditives Feedback Sinn machen könnte, um die Aufmerksamkeit des Benutzers darauf zu lenken. Ein Modul, dass diese Events erkennt, könnte beispielsweise dadurch realisiert werden, dass der Benutzer eine Atomgruppe vorgibt, die entweder eine Bindungsstelle oder eine Bindungstasche bilden. Wenn ein

anderes Molekül sich dann eine Zeit lang in der Nähe dieser Bindungsstelle aufhält und entsprechend ausgerichtet ist, ist davon auszugehen, dass eine Bindung stattgefunden hat. Eine mögliche Benutzung des *Activity Levels* würde darin bestehen, die Aktivität innerhalb einer Bindungstasche anzuzeigen. Darüber ließe sich beispielsweise erkennen, ob die Bindungstasche von anderen Molekülen blockiert wird.

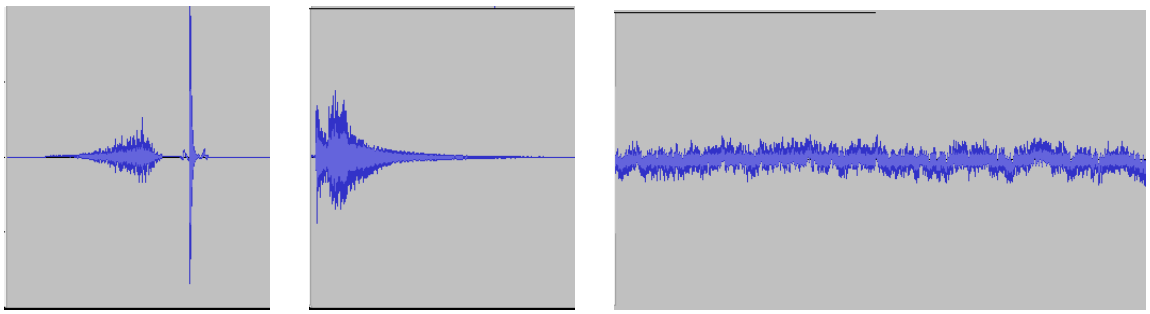
7.2.3. Strukturelle Veränderungen

Auch um strukturelle Veränderungen innerhalb eines Proteins zu analysieren, könnte die Vertonung Vorteile bringen. Wenn nur die Atome ausgegeben werden, aus denen das Molekül besteht, ist die Struktur eventuell nicht erkennbar, da das Protein wie ein Ball aus einer Molekülkette zusammengeknüllt sein kann. Die Struktur könnte zwar halb transparent über das Bild gezeichnet werden, nimmt dadurch aber zum einen die Sicht auf Vorgänge an der Oberfläche und zum anderen ist es schwierig, die Faltung aus verschiedenen Winkeln richtig zu interpretieren. Um die Analyse dieser Daten zu erleichtern, können AudioEvents eingesetzt werden, die ein Ändern der Sekundärstrukturen hörbar machen. So könnte man zum einen große Veränderungen mit AudioEvents versehen oder alternativ eine Struktur vorgeben und Töne erzeugen, falls sich die Struktur zu stark von der vorgegebenen unterscheidet. Das *Activity Level* könnte in diesem Fall eingesetzt werden, um die Änderungsrate zu vertonen.

8. Ergebnisse

8.1. Geeignete Töne

Zum Testen des AudioRenderers mit Hilfe der Wasserstoffbrücken mussten drei passende Klänge ausgewählt werden. Zwei Klänge für die Events beim Entstehen und Aufbrechen der Brücken sowie einer für das *Activity Level*. Hierfür wurden jeweils verschiedene Töne ausprobiert, die lizenzfrei von verschiedenen Quellen zur Verfügung gestellt wurden, und die dabei gewonnenen Erkenntnisse sollen im Folgenden erläutert werden.
Getestet wurden unter anderem verschieden lange Geräusche. Dabei fällt auf, dass die Geräusche, je nach Abspielgeschwindigkeit der Simulation, nicht mehr richtig unterscheidbar sind, wenn sie zu lange ausfallen. Da natürlich klingende Töne immer eine gewisse Anstiegs- und Abfallszeit besitzen und nicht abrupt lauter oder leiser werden, kommt bei langen Tönen dazu, dass die Anstiegs- und Abfallszeit meist noch durch andere Audioquellen überdeckt werden. Wählt man jedoch zu kurze Töne, die nur 100ms lang andauern, können zwar einzelne Events gut unterschieden werden, jedoch wird die Lokalisation nahezu unmöglich. Als geeignet haben sich Töne herausgestellt, die etwas länger andauern, jedoch ein kurzes prägnantes Merkmal besitzen. Eine sinnvolle Verbesserung des AudioRenderers könnte darin bestehen, je nachdem, wie viele Events pro Frame eintreffen und mit welcher Geschwindigkeit die einzelnen Frames abgespielt werden, entweder die Töne entsprechend zu modifizieren, um ihre Länge optimal anzupassen.
Auch die Art der Töne spielt eine Rolle. Zum einen lassen sich einfache Töne, die kein großes Spektrum besitzen, schwerer lokalisieren. Dies liegt daran, dass die Filterung durch die HRTF die Töne kaum verändert und somit die Unterscheidung von hinten/vorne und oben/unten erheblich erschwert. Bei sehr tiefen Tönen scheint selbst die Unterscheidung von rechts/links schwieriger zu werden. Dies legt den Schluss nahe, dass die Implementierung der Interaural Time Difference, die gerade für tiefe Töne für die Lokalisation auf der horizontalen Ebene eine wichtige Rolle spielt (siehe Kapitel 2.1.4), nicht gut funktioniert. Dies kann unter anderem daran liegen, dass Abweichungen zwischen realer Kopfform und den Modellannahmen existieren, die für das Erzeugen dieses Effekts getroffen werden. Für hochfrequente Töne funktioniert diese Lokalisation besser, da diese größtenteils durch die Interaural Intensity Difference realisiert werden und bei dieser die Kopfform keinen so entscheidenden Einfluss besitzt. Jedoch werden die hochfrequenten Töne deutlich lauter wahrgenommen und treten somit stärker in den Vordergrund als tiefere Töne. Wenn also nicht gewollt ist, dass die Aufmerksamkeit des Benutzers stärker auf bestimmte Events gelenkt wird, ist es sinnvoll, die Töne so anzupassen, dass diese gleich laut wahrgenommen werden. Zu laute Töne oder auch sehr hohe Töne eignen sich sowieso nicht, da diese eher als unangenehm wahrgenommen werden und und deshalb nicht für einen längeren Zeitraum eingesetzt werden können. Fürs erste wurden deshalb zwei Töne ausgewählt, der sich am besten mit dem Platzen einer Kaugummiblase (siehe Abb. reffig:wavBubble) und einem Schlag auf einen Topf (siehe Abb. reffig:wavClong) beschreiben lassen. Beide Töne dauern ungefähr



(a) (b) (c)

Abbildung 8.1.: Die drei verwendeten Töne: a) Ausdehnen einer Blase und anschließendes Platzgeräusch. b) Schlag auf einen Topf bei dem der Ton abklingt. c) 1/f Rauschen (alle Frequenzen werden gleich laut wahrgenommen).

eine halbe Sekunde und besitzen ein ausgeprägtes Merkmal, das nur kurz anhält und das somit die Vermischung mehrerer Events verhindert.
Auch für das *Activity Level* wurden verschiedene Töne ausprobiert. Getestet wurden unter anderem durchgängige Töne sowie kurze Töne, die in regelmäßigen Abständen abgespielt wurden und sich somit wie eine Art Sonar verhielten. Beide Arten eignen sich gut für diesen Zweck, wenn jedoch sowohl das *Activity Level* als auch AudioEvents abgespielt werden, ist ein durchgehender Ton besser geeignet, da sich die einzelnen Töne überlagern können. Aus diesem Grund wurde für den Ton des *Activity Level* ein Hintergrundrauschen (siehe Abb. 8.1c) verwendet, dass außerdem den Vorteil hat, das man es leicht ausblenden kann, wenn man sich auf die anderen Töne konzentriert. Wenn das *Activity Level* jedoch nicht über die Lautstärke sondern über die Tonhöhe angezeigt werden soll, funktioniert dies mit dem Rauschen nur sehr schlecht. Hierfür ist ein schmalbandiger kurzer Ton besser geeignet, da bei dem breitbandigen Rauschen die Veränderung der durchschnittlichen Frequenz keine gut wahrnehmbaren Unterschiede erzeugt.

8.2. Test des AudioRenderers

Ausprobiert wurde das System vor allem an zwei Systemen: einem Laptop mit angeschlossenen Kopfhörern und einem PC mit 5.1 Surroundsystem. Außerdem wurde die Unterstützung des 7.1 Systems des Visualisierungslabors der Universität Stuttgart erfolgreich getestet.

Bei allen Systemen funktioniert das System aus technischer Sicht problemlos. Bei den ersten Tests der AudioEvents für Wasserstoffbrücken, bei dem die Länge der Wasserstoffbrückenbindung noch keinen Einfluss genommen hat, ist die Zuordnung der hörbaren Events zu den sichtbaren Hinweisen nicht möglich, da schlichtweg zu viele Events auftreten, die dann nicht mehr unterscheidbar sind. Die Events werden somit vielmehr selbst zu einer Art *Activity Level*-Indikator (siehe Abb. 8.3a). So lässt sich ausmachen, ob gerade viele Wasserstoffbrücken entstehen oder aufbrechen und auch eine Abschätzung davon, wie viele Events gerade gleichzeitig passieren, ist möglich. Mit der Einführung des Filters wurde dann die Möglichkeit geschaffen, diese Anzahl zu reduzieren, sodass die einzelnen Events

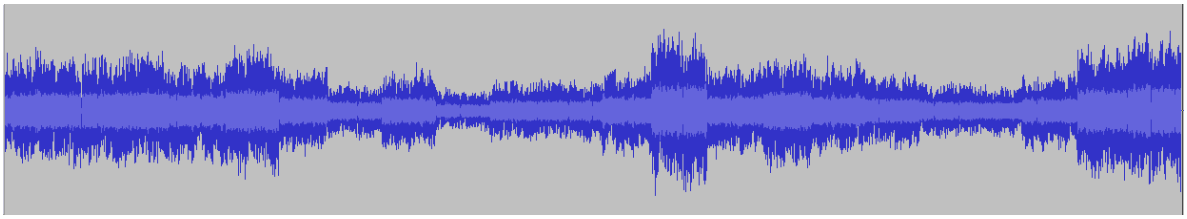
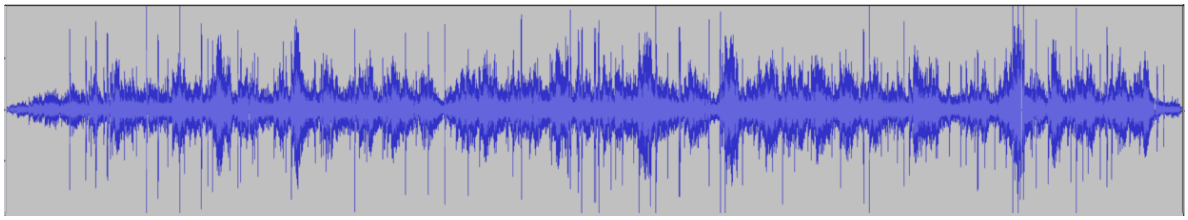
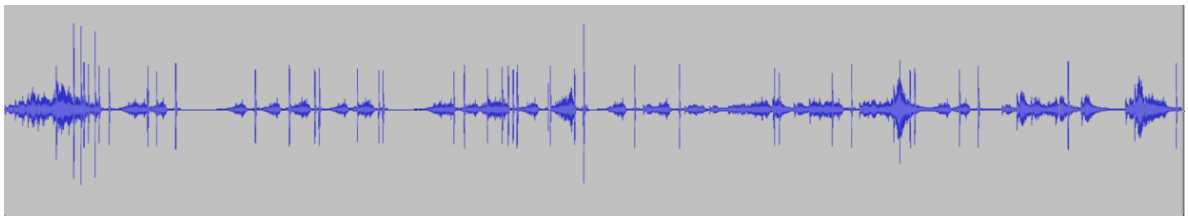


Abbildung 8.2.: Vertonung des *Activity Levels* mit Hilfe des Rauschens. Die einzelnen Frames sind teilweise deutliche Abstufungen zu erkennen.

unterscheidbar werden und eine Zuordnung des Gehörten zu dem, was angezeigt wird, machbar wird. Bei einer Filtereinstellung, die nur Wasserstoffbrücken berücksichtigt die länger als zehn Frames existieren, bleiben ungefähr drei AudioEvents pro Sekunde übrig (siehe Abb. 8.3b). Diese Rate gibt dem Benutzer genügend Zeit, auf ein Event zu reagieren und den Auslöser des Events auf dem Bild zu lokalisieren.
Um Feedback von Benutzern zu bekommen, die tatsächlich *MegaMol* für ihre Arbeit verwenden, wurde das System einigen Biochemikern vorgeführt. Die Rückmeldungen dazu, fielen durchgehend positiv aus. Auch wenn die Erkennung der Wasserstoffbrücken zur Zeit keine praktische Verwendung findet, wurden einige mögliche Events diskutiert, für welche eine Sonifikation auch sinnvoll sein köennte (siehe Kapitel 7). Auch mögliche Einsatzarten des AudioRenderers wurden erörtert. Eine Einsatzmöglichkeit besteht bei der Analyse von Daten. Viele Anwendungen der Sonifikation haben gezeigt, dass die Sonifikation bei der Datenanalyse durchaus Sinn machen kann (siehe Kapitel 5) und die Analyse dadurch nicht nur interessanter, sondern auch effektiver gestaltet werden kann. In wie weit dies auch für die Analyse von molekulardynamischen Simulationen zutrifft, wurde bisher noch nicht untersucht und soll nun, unter Verwendung dieser Arbeit, in einer Studie untersucht werden. Neben der Analyse der Daten, sahen die Biochemiker vor allem Potential für die Präsentation von Daten. Sowohl in der Lehre, als auch beim Vorstellen von Forschungsergebnissen auf Konferenzen, kann die Sonifikation dazu beitragen, die Präsentation interessanter zu gestalten und die Aufmerksamkeit der Zuhörer auf bestimmte Ereignisse zu lenken.
Wie und wo sich die Vertonung der molekularen Daten am besten einsetzen lässt, muss also erst noch untersucht werden. Jedoch ist schon jetzt klar, dass es verschiedenste Möglichkeiten gibt, die Sonifikation sinnvoll in diesem Gebiet einzusetzen.



(a)



(b)

Abbildung 8.3.: Die ersten 15 Frames bei normaler Geschwindigkeit. a) Ohne Filter, viele Events Überlagern sich, jedoch sind noch einige Merkmale vorhanden, die einen groben Überblick erlauben. b) Herausfiltern der Events die kürzer als 10 Frames andauern. Das Erkennen einzelner Events ist möglich.

9. Zusammenfassung und Ausblick

9.1. Zusammenfassung

In dieser Arbeit wurde zuerst auf das menschliche Gehör eingegangen. Dabei wurde beschrieben, welche Merkmale der Mensch überhaupt unterscheiden kann und wo die Grenzen des Gehörs liegen. Im folgenden wurde gezeigt, wie diese Fähigkeiten in der Sonifikation benutzt werden, um wissenschaftliche Daten zu vertonen und an einigen Beispielen ausgeführt, dass diese Art der Datenrepräsentation durchaus vielversprechende Ergebnisse liefern kann. Insbesondere wurde gezeigt, dass sich die Sonifikation schon in einigen verwandten Gebieten wie der DNA-Analyse und bei einigen chemischen Untersuchungen bewiesen hat.
Da es sich bei den zu vertonenden Daten außerdem um räumliche Daten handelte, wurde dabei auch speziell auf Fähigkeit des Menschen eingegangen, eine räumliche Lokalisation der Geräusche durchzuführen. Außerdem wurde ein kurzer Überblick darüber gegeben, wie räumliche Geräusche technisch überhaupt umgesetzt werden können. Unter Berücksichtigung dieser Grundlagen wurde das *MegaMol* Projekt um die Möglichkeit ergänzt, bestimmte Ereignisse zu sonifizieren. Dazu wurde zunächst ein Audioframework erstellt, welches erlaubt, möglichst einfach Objekte zu erstellen, deren Eigenschaften mittels verschiedener Parameter angepasst werden können. Das Framework erlaubt zum einen ein einfaches Laden und Buffern von Audiodateien und außerdem ein Anpassen deren Lautstärken, Tonhöhen sowie Positionen und Bewegungsrichtungen im Raum. Mit Hilfe dieser Implementierung wurden verschiedene Module für das *MegaMol* Visualisierungstool erstellt, welche es ermöglichen, verschiedene Events zu vertonen. Die dabei entstandene Implementierung ist flexibel und kann sinnvoll und einfach erweitert werden. Unter anderem erlaubt sie das Speichern und Laden der Events und damit das Vorberechnen komplizierterer Events, deren Erkennung nicht in Echtzeit umsetztbar ist. Um zu testen, ob überhaupt einzelne Events hörbar sind und man diese dem Gesehenen zuordnen kann, wurde dann eine Erkennung von Wasserstoffbrücken vertont. Mit diesem System wurden verschiedene Töne getestet und überprüft, ob die Sonifikation von molekulardynamischen Simulationen überhaupt sinnvolle Ergebnisse liefern kann, oder ob am Ende nur Geräusche erzeugt werden, aus denen nichts mehr herauszuhören ist. Neben der Erkennung der Wasserstoffbrücken wurden in Zusammenarbeit mit einigen Biochemikern, welche *MegaMol* tatsächlich für ihre Arbeit einsetzen, einige weitere Events überlegt, deren Erkennung sinnvoll sein kann.

9.2. Weitere Arbeiten

Um ein immersiveres Erlebnis zu produzieren und die 3D Lokalisation zu verbessern, bietet es sich an einige Änderungen vorzunehmen, die die Benutzung mit Kopfhörern stark verbessern kann. Eine Verbesserung könnte darin bestehen, die HRTF an den Benutzer anzupassen. Dies erlaubt eine genauere

Lokalisation, da sich die HRTF von zwei verschiedenen Menschen stark unterscheiden kann. Leider ist diese Personalisierung noch immer mit hohem zeitlichen und technischem Aufwand verbunden, da zum einen sehr viele Positionen um den Kopf herum getestet werden müssen und zum anderen ein schalltoter Raum benötigt wird, um die Reflexionen der Schallwellen und eine damit einhergehende Verzerrung der HRTF zu vermeiden. Hier könnten die in Kapitel 4.3.3 erwähnten Verfahren Abhilfe schaffen. Eine weitere Verbesserungsmöglichkeit bietet das Headtracking des Benutzers. Durch Anpassen des virtuellen Mikrofons an die Kopfausrichtung ist es möglich, ein eindringenderes Erlebnis zu schaffen. Auch die Lokalisation kann durch minimale Bewegungen des Kopfes verbessert werden (siehe Kapitel 2.1.4). Moderne Systeme wie die *Kinect* und höhere Rechenleistung erlauben inzwischen auch für Endanwender eine ziemlich genaue Positionsbestimmung des Benutzers, welche für ein besseres Nutzererlebnis sorgen kann. Auch das Visualisierungslabors verfügt über ein solches Tracking-System, welches benutzt werden könnte. Als letzte Verbesserung des räumlichen Klangerlebnisses ist eine Simulation des Nachhalls denkbar. Dazu wird eine virtuelle Umgebung geschaffen, in der sich der Benutzer und die Audioquelle befinden. Wenn an der Audioquelle ein Ton abgespielt wird, wird berechnet, wo und wann der Schall an der Umgebung reflektiert wird. An dieser Stelle wird dann ein entsprechend der Reflexionseigenschaften der virtuellen Umgebung angepasstes, abgeschwächtes Echo des Tons erzeugt. Diese drei Verbesserungen wurden auch von Begault et al. [BWA01] untersucht. Die Ergebnisse zeigen zwar, dass für die Lokalisation im Raum das Headtracking nur eine geringe Verbesserung bringt, der Einsatz von personalisierten HRTF aber immerhin eine Verbesserung der Genauigkeit von ungefähr 25% bei der Lokalisation brachte und die Verwendung von simuliertem Nachhall im Vergleich die Anzahl der Internalisierungsfehler stark reduzieren konnte. Als Internalisierungsfehler bezeichnet man dabei den Effekt, dass sich eine Audioquelle bei der Lokalisierung scheinbar im Inneren des Kopfes befindet. Eine Implementierung dieser drei Verbesserungen kann sich also durchaus lohnen.

In der bisherigen Implementierung konnten nur vorgegebene Töne, die im WAV Format vorliegen, abgespielt werden und deren Tonhöhe, Lautstärke und Position im Raum verändert werden. Hier wäre zum einen die Unterstützung weiterer Formate denkbar. Die Architektur würde eine solche Erweiterung leicht erlauben, da der SoundRessourceManager je nach Dateiendung verschiedene Methoden aufrufen kann. Als Decoder für weitere Formate lässt sich beispielsweise die Bibliothek *FFmpeg* [dev14] verwenden, welche eine Unterstützung für alle gängigen Audioformate liefert. Anstatt nur Audiodateien abzuspielen, wäre auch eine dynamische Generierung der Töne aus der Simulation denkbar. Dies würde auch ein einfaches Anpassen der Länge der Geräusche an die Anzahl der abzuspielenden Events erlauben. Außerdem bietet eine solche Synthese weitaus höhere Freiheitsgrade und ermöglicht damit, mehr Eigenschaften zu unterscheiden. Zur Synthese von Sounds gibt es einige Frameworks, die verwendet werden könnten. Hierzu gehören zum Beispiel der *Csound Synthesizer* [Kon14] oder das *Synthesis ToolKit* [CS13]. Während es sich beim *Csound Synthesizer* um ein Tool handelt, bei dem die zu erzeugenden Töne durch eine eigene Sprache beschrieben werden und dann durch ein Kommandozeilentool in entsprechende Audiodateien umgewandelt werden können, handelt es sich beim *Synthesis ToolKit* um eine C++ Bibliothek, die direkt in das Programm eingebunden werden kann. Beide erlauben jedoch ein einfaches Erzeugen und Modifizieren von Tönen. So können beispielsweise verschiedene Sinustöne erzeugt und gemischt, oder auf vorgegebenere komplexere Audioklassen zurückgegriffen werden, die sich wie echte Instrumente anhören.

9.3. Ausblick

Das Ziel dieser Arbeit, nämlich die Implementierung eines Audioframeworks für das *MegaMol* Projekt, mit dem bestimmte Events erkannt und diese dem Benutzer mit Hilfe von Tönen angezeigt werden, wurde erreicht. Es wurde ein System geschaffen, das leicht modifizierbar und somit in neuen Einsatzgebieten verwendet werden kann und mit dem die Implementierung neuer Events ohne großen Aufwand betrieben werden kann. Als erste solche Eventerkennung wurde zudem die Erkennung von Wasserstoffbrücken implementiert und mit dieser ausprobiert, ob der Benutzer überhaupt in der Lage ist, etwas aus den Tönen heraus zu erkennen. Inwieweit diese Sonifikation ermöglicht, die Analyse von molekulardynamischen Simulationen zu verbessern oder ob die Vertonung nur dazu verwendet werden kann, die Präsentation der Daten durch das Hinterlegen mit Tönen interessanter zu gestalten, muss durch eine Studie getestet werden. Durch die Nachforschung im Bereich der Sonifikation und die vielen Anwendungsbeispiele, bei denen die Sonifikation mit großem Erfolg eingesetzt wird, kann man jedoch davon ausgehen, dass die Vertonung auch für molekulardynamischen Simulationen großes Potential besitzt.

A. Ein Anhang

Listing A.1 Laden einer Audiodatei

```
ALuint *buffer; // hold the pcm data.

// Load the data into the buffer
buffer = alutCreateBufferFromFile("SomeAudio.wav");
// Test if everything went right.
if(*buffer == AL_NONE) {
            std::cerr << "Error generating sound buffers: " << alutGetError() <<std::endl;
            return false;
}

// Create a source.
ALuint source;
alGenSources(1, &source);
// Test for errors;
ALuint error = alGetError();
if(error != AL_NO_ERROR || src == 0) {
            src = 0;
            std::cerr << "Could not allocate source. " << error << std::endl;
}

// Link the buffer to the source.
alSourcei(source, AL_BUFFER, buffer);
```

Listing A.2 Soundkarte.cfg

```
// Verfuegbare Soundkarten (Auswahl einer Soundkarte durch Markieren mit *)
*ALSA Default
HDA Intel PCH, ALC663 Analog (CARD=PCH,DEV=0)
USB Sound Device, USB Audio (CARD=Device,DEV=0)
```

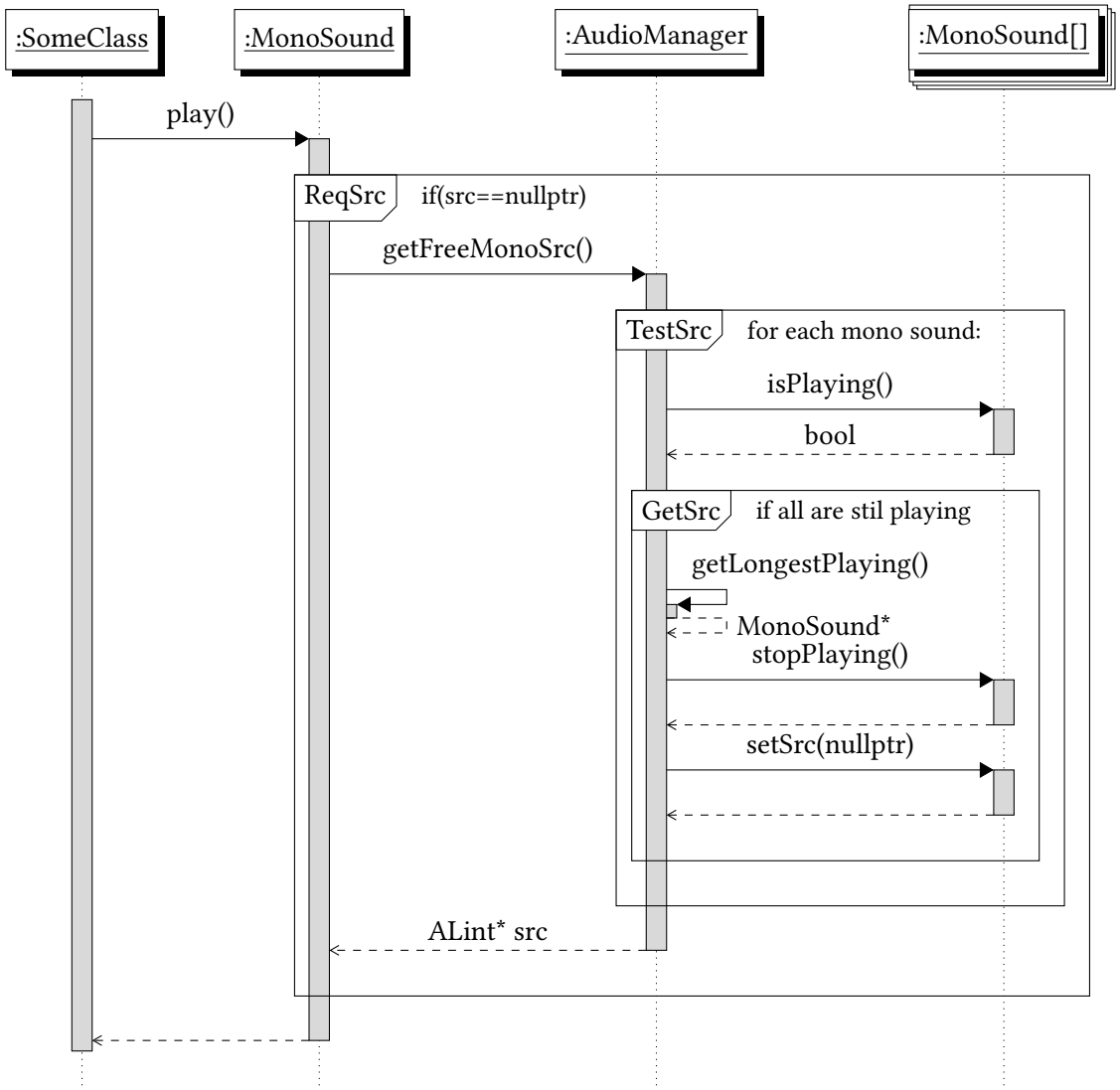


Abbildung A.1.: UML Sequenzdiagram für das Abspielen eines MonoSounds

Listing A.1 Beispiel einer Projektdatei, die den AudioRenderer benutzt um die Bildung von Wasserstoffbrücken hörbar zu machen.

```
<?xml version="1.0" encoding="utf-8" ?>
<MegaMol type=""project"" version=""1.0"">

<!--
Use this command line arguments to start MegaMol
in Cmd:
  -p path\to\this\file\audioPlugin.mmprj -i MUX inst
-->
      <view name="MUX" viewmod="View3D1">
      <!-- Loader for the molecular data -->
      <module class="PDBLoader" name="PDBLoader1" confpos="{X=1046,Y=283}">
         <param name="pdbFilename"
              value="/home/benjamin/studium/bachelor/mount/protein-water-f100.pdb" />
         <param name="xtcFilename"
              value="/home/benjamin/studium/bachelor/mount/protein-water-900f.xtc" />
         <param name="solventResidues" value="SOL" />
      </module>

      <!-- Multiplexer to have multiple renderers -->
            <module class="Mux2Renderer3D" name="Mux2Renderer3D1" confpos="{X=300,Y=276}" />
      <!-- One for rendering for the molecular data -->
      <module class="SimpleMoleculeRenderer" name="SimpleMoleculeRenderer1"
          confpos="{X=631,Y=220}" />
      <!-- The other for rendering the audio -->
      <module class="AudioRenderer" name="AudioRenderer1">
         <param name="sound::enableDebugRendering" value="1"/>
         <param name="sound::activityAudioFile"
              value="/home/benjamin/studium/bachelor/static.wav"/>
      </module>
      <!-- for navigation in the view -->
      <module class="View3DSpaceMouse" name="View3D1" confpos="{X=36,Y=281}" >
         <param name="anim::speed" value="0.15" />
      </module>

      <!-- This calculates the Hydrogenbonds -->
      <module class="SolventDataGenerator" name="SDG1" />
      <!-- Which are used by this module to create the audio events -->
      <module class="AudioEventHydrogenBond" name="AE1">
         <param name="hBondCreationSoundFile" value="/home/benjamin/studium/bachelor/bubble.wav"/>
         <param name="hBondReleaseSoundFile" value="/home/benjamin/studium/bachelor/clong.wav"/>
      </module>

      <!-- Forwarding the molecular data from the Loader to the renderer -->
      <call class="MolecularDataCall" from="SimpleMoleculeRenderer1::getdata"
          to="PDBLoader1::dataout" />
      <!-- And pass the data through the SDG and the Audio event creator to the AudioRenderer -->
      <call class="AudioDataCall" from="AudioRenderer1::getaudio" to="AE1::audioout" />
      <call class="MolecularDataCall" from="AE1::getdata" to="SDG1::dataout" />
      <call class="MolecularDataCall" from="SDG1::getInputData" to="PDBLoader1::dataout" />

      <!-- The main render method is the Mux2Renderer which calls the other two renderer -->
            <call class="CallRender3D" from="View3D1::rendering" to="Mux2Renderer3D1::rendering"
                />
            <call class="CallRender3D" from="Mux2Renderer3D1::renderer1"
                to="SimpleMoleculeRenderer1::rendering" />
      <call class="CallRender3D" from="Mux2Renderer3D1::renderer2" to="AudioRenderer1::rendering"
          />
      </view>
</MegaMol>
```

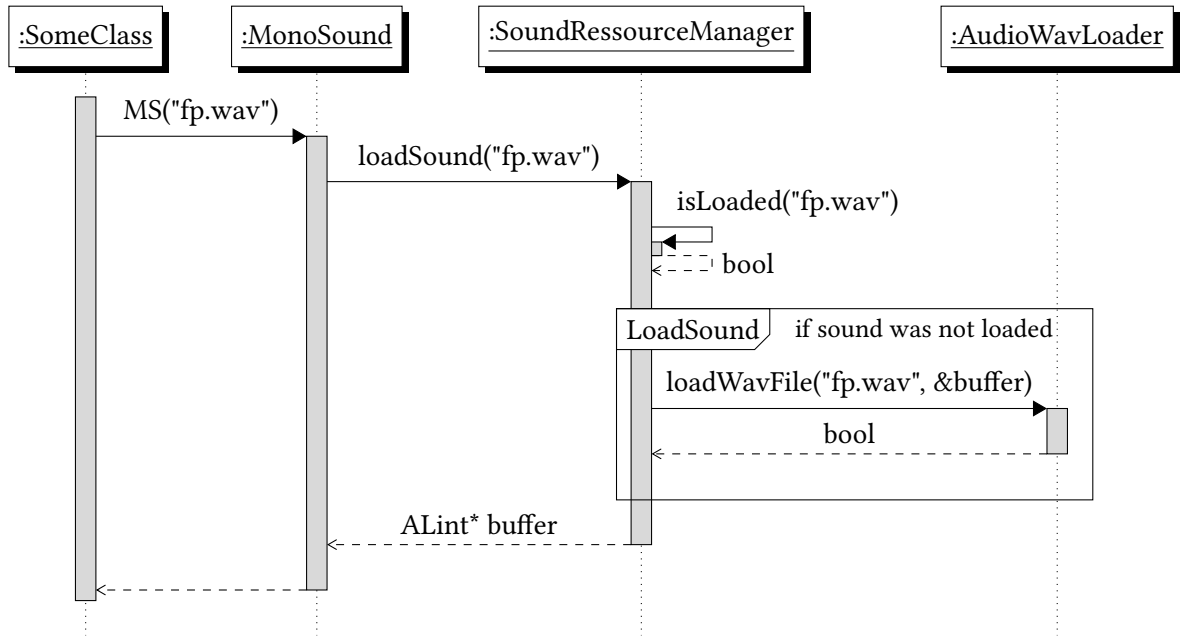


Abbildung A.2.: UML Sequenzdiagram für das Laden einer Audiodatei

Listing A.2 Beispiel einer Projektdatei für einen Job, bei dem Daten aus einer Datei geladen werden und in eine andere Datei geschrieben werden.

```
<?xml version="1.0" encoding="utf-8"?>
<MegaMol type="project" version="1.0">

<!--
Use this command line arguments to start this MegaMol job
  -p path\to\this\file\saveLoop.mmprj -i AUDIO2AX inst
-->
      <job name="AUDIO2AX">
      <!-- Module to load precalculated audio from a file -->
      <module class="AudioLoader" name="AudioLoader1" confpos="{X=1046,Y=283}">
         <param name="axFilename" value="/path/to/load/the/audiofile.ax" />
      </module>
      <!-- Module to save data coming from an AudioDataCall -->
            <module class="AudioWriter" name="AudioWriter1" confpos="{X=397,Y=617}">
         <param name="filenamePrefix" value="savefile.ax" />
         <param name="outputFolder" value="/path/to/folder" />
            </module>

      <!-- Simple call where the writer requests the data from the loader -->
      <call class="AudioDataCall" from="AudioWriter1::getaudio" to="AudioLoader1::dataout" />
      </job>
</MegaMol>
```

Literaturverzeichnis

[AH14] N. L. Aaronson, W. M. Hartmann. Testing, correcting, and extending the Woodworth model for interaural time difference. *The Journal of the Acoustical Society of America*, 135(2):817–823, 2014. (Zitiert auf Seite 32)

[Ata66] B. S. Atal. Apparent sound source translator, 1966. US Patent 3,236,949. (Zitiert auf Seite 27)

[B+94] D. R. Begault, et al. *3-D sound for virtual reality and multimedia*, Band 955. AP professional Boston etc, 1994. (Zitiert auf Seite 31)

[BB08] S. Barrass, V. Best. Stream-based sonification diagrams. *Proc 15th ICAD, IRCAM, Paris, July*, 2008. (Zitiert auf Seite 22)

[Ber96] K. D. Berndt. Identification of 3D Structures, 1996. URL http://www.cryst.bbk.ac.uk/PPS2/course/section8/ss-960531_19.html. (Zitiert auf Seite 49)

[BH99] A. W. Bronkhorst, T. Houtgast. Auditory distance perception in rooms. *Nature*, 397(6719):517–520, 1999. (Zitiert auf Seite 19)

[Blo10] P. Blomqvist. Stereo Microphone Techniques in Drum Recording, 2010. (Zitiert auf Seite 31)

[Bly82] S. Bly. Presenting information in sound. In *Proceedings of the 1982 conference on Human factors in computing systems*, S. 371–375. ACM, 1982. (Zitiert auf Seite 35)

[BM97] G. Berenato, D. F. Maynard. A Simple Audio Conductivity Device. *Journal of chemical education*, 74(4):415, 1997. (Zitiert auf Seite 36)

[BWA01] D. R. Begault, E. M. Wenzel, M. R. Anderson. Direct Comparison of the Impact of Head Tracking, Reverberation, and Individualized Head-Related Transfer Functions on the Spatial Perception of a Virtual Speech Source. *J. Audio Eng. Soc*, 49(10):904–916, 2001. URL http://www.aes.org/e-lib/browse.cfm?elib=10175. (Zitiert auf Seite 58)

[Che53] E. C. Cherry. Some experiments on the recognition of speech, with one and with two ears. *The Journal of the acoustical society of America*, 25(5):975–979, 1953. (Zitiert auf Seite 16)

[Cho08] E. Choueiri. Optimal crosstalk cancellation for binaural audio with two loudspeakers. *Princeton University*, 2008. (Zitiert auf Seite 28)

[CJM00] F. Christensen, C. B. Jensen, H. Møller. The design of VALDEMAR-an artificial head for binaural recording purposes. In *Audio Engineering Society Convention 109*. Audio Engineering Society, 2000. (Zitiert auf Seite 31)

[CS13] P. R. Cook, G. P. Scavone. The Synthesis ToolKit in C++ (STK), 2013. URL https://ccrma.stanford.edu/software/stk/. (Zitiert auf Seite 58)

[DAG] F. Dynamics, U. o. S. Acoustics Group. Handbook of Sonification Chapter 15. URL http://resource.isvr.soton.ac.uk/FDAG/VAP/html/xtalk.html. (Zitiert auf den Seiten 7 und 29)

[Del00] T. Delatour. Molecular music: the acoustic conversion of molecular vibrational spectra. *Computer Music Journal*, 24(3):48–68, 2000. (Zitiert auf Seite 35)

[dev14] T. F. developers. FFmpeg Documentation, 2014. URL http://ffmpeg.org/ffmpeg.html. (Zitiert auf Seite 58)

[DLW+08] T. Dingler, J. Lindsay, B. N. Walker, et al. Learnability of sound cues for environmental features: Auditory icons, earcons, spearcons, and speech. In *Proceedings of the 14th International Conference on Auditory Display, Paris, France*, S. 1–6. 2008. (Zitiert auf Seite 22)

[Dom01] F. Dombois. Using audification in planetary seismology. 2001. (Zitiert auf Seite 21)

[DP02] H. J. G. De Poli. 9.1 Definition of timbre modelling. 2002. (Zitiert auf Seite 16)

[DS01] G. R. Desiraju, T. Steiner. *Weak hydrogen Bond*. Oxford University Press New York, 2001. (Zitiert auf Seite 49)

[DZL+05] R. Duraiswami, D. N. Zotkin, Z. Li, E. Grassi, N. A. Gumerov, L. S. Davis. High Order Spatial Audio Capture and Binaural Head-Tracked Playback over Headphones with HRTF Cues. 2005. (Zitiert auf Seite 31)

[for14a] Dolby, 2014. URL http://de.wikipedia.org/wiki/Dolby. (Zitiert auf Seite 30)

[for14b] DTS, 2014. URL http://de.wikipedia.org/wiki/DTS_(sound_system). (Zitiert auf Seite 30)

[Gar97] W. G. Gardner. Head tracked 3-d audio using loudspeakers. In *Applications of Signal Processing to Audio and Acoustics, 1997. 1997 IEEE ASSP Workshop on*, S. 4–pp. IEEE, 1997. (Zitiert auf Seite 28)

[GHE03] D. W. Grantham, B. W. Hornsby, E. A. Erpenbeck. Auditory spatial resolution in horizontal, vertical, and diagonal planes. *The Journal of the Acoustical Society of America*, 114(2):1009–1022, 2003. (Zitiert auf Seite 19)

[GKM+14] S. Grottel, M. Krone, C. Müller, G. Reina, T. Ertl. MegaMol - A Prototyping Framework for Particle-based Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2014. (Zitiert auf Seite 11)

[GLT03] M. Grohn, T. Lokki, T. Takala. Comparison of auditory, visual, and audio-visual navigation in a 3D space. *ICAD2003*, 2003. (Zitiert auf Seite 24)

[Gre95] R. Gregory. *Protein-solvent interactions*. CRC Press, 1995. (Zitiert auf Seite 51)

[GRGP06] M. A. Garcia-Ruiz, J. R. Gutierrez-Pulido. An overview of auditory display to assist comprehension of molecular information. *Interacting with Computers*, 18(4):853–868, 2006. (Zitiert auf Seite 36)

[GSO91] W. W. Gaver, R. B. Smith, T. O'Shea. Effective sounds in complex systems: The ARKola simulation. In *Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, S. 85–90. ACM, 1991. (Zitiert auf Seite 25)

[Gul71] W. Gulick. *Hearing: physiology and psychophysics*. Oxford University Press, 1971. URL http://books.google.de/books?id=69lqAAAAMAAJ. (Zitiert auf Seite 13)

[HB08] T. Hermann, G. Baier. Die Sonifikation des menschlichen EEG. *Katalog: Wien Modern*, S. 25–27, 2008. (Zitiert auf Seite 21)

[Her08] T. Hermann. Taxonomy and definitions for sonification and auditory display. In *Proceedings of the 14th International Conference on Auditory Display (ICAD 2008)*. 2008. (Zitiert auf Seite 21)

[Her11] N. Hermann, Hunt. Handbook of Sonification Chapter 15, 2011. URL http://sonification.de/handbook/index.php/chapters/chapter15/. (Zitiert auf Seite 23)

[HHN11] T. Hermann, A. Hunt, J. G. Neuhoff. *The sonification handbook*. Logos Verlag Berlin, 2011. (Zitiert auf Seite 23)

[Hir48] I. J. Hirsh. The influence of interaural phase on interaural summation and inhibition. *The Journal of the Acoustical Society of America*, 20(4):536–544, 1948. (Zitiert auf Seite 17)

[HK86] M. N. Hayashi K. Basically Musical. *Nature*, 310, 1986. (Zitiert auf Seite 36)

[HR99] T. Hermann, H. Ritter. Listen to your data: Model-based sonification for data analysis. *Advances in intelligent computing and multimedia systems*, 8:189–194, 1999. (Zitiert auf Seite 23)

[KA96] R. D. King, C. G. Angus. PM—Protein music. *Computer applications in the biosciences: CABIOS*, 12(3):251–252, 1996. (Zitiert auf Seite 37)

[Kon14] C. Konopka. CSound, 2014. URL http://www.csounds.com/. (Zitiert auf Seite 58)

[Kun08] M. N. Kunchur. Temporal resolution of hearing probed by bandwidth restriction. *Acta Acustica united with Acustica*, 94(4):594–603, 2008. (Zitiert auf den Seiten 13 und 16)

[KWB+10] G. Kramer, B. Walker, T. Bonebright, P. Cook, J. H. Flowers, N. Miner, J. Neuhoff. Sonification report: status of the field and research agenda. 2010. (Zitiert auf den Seiten 21 und 25)

[Lab] C. Labs. OpenAL Spezifikation 1.1. URL http://www.openal.org/documentation/openal-1.1-specification.pdf. (Zitiert auf Seite 48)

[LKPG98] J. M. Loomis, R. L. Klatzky, J. W. Philbeck, R. G. Golledge. Assessing auditory distance perception using perceptually directed action. *Perception & Psychophysics*, 60(6):966–980, 1998. (Zitiert auf Seite 19)

[Ltd14] Y. D. S. Ltd. Your DNA Song, 2014. URL http://www.yourdnasong.com/. (Zitiert auf Seite 37)

[Lun94] D. Lunney. Development of a data acquisition and data analysis system for visually impaired chemistry students. *Journal of Chemical Education*, 71(4):308, 1994. (Zitiert auf Seite 36)

[Man08] R. Manell. Theories of Hearing, 2008. URL http://clas.mq.edu.au/speech/perception/psychoacoustics/hearing_theory.html. (Zitiert auf Seite 13)

[MB85] A. D. Musicant, R. A. Butler. Influence of monaural spectral cues on binaural localization. *The Journal of the Acoustical Society of America*, 77(1):202–208, 1985. (Zitiert auf Seite 17)

[McG09] R. McGee. Auditory Displays and Sonification: Introduction and Overview, 2009. (Zitiert auf Seite 25)

[meg] URL http://go.visus.uni-stuttgart.de/megamol. MegaMol project website http://go.visus.uni-stuttgart.de/megamol/. (Zitiert auf Seite 48)

[Mil06] J. P. Milsap. Phased array sound system, 2006. US Patent 7,130,430. (Zitiert auf Seite 28)

[MM95] D. G. Malham, A. Myatt. 3-D sound spatialization using ambisonic techniques. *Computer Music Journal*, S. 58–70, 1995. (Zitiert auf Seite 30)

[Mun] N. Munakata. Gene Sequence Analysis with Auditory Display. (Zitiert auf Seite 36)

[Muz06] D. Muzzulini. *Genealogie der Klangfarbe*, Band 5. Peter Lang, 2006. (Zitiert auf Seite 15)

[Nav99] R. Nave. Hearing, 1999. URL http://hyperphysics.phy-astr.gsu.edu/hbase/sound/pitch.html#c1. (Zitiert auf den Seiten 7, 13 und 14)

[NMK+96] T. Nishino, S. Mase, S. Kajita, K. Takeda, F. Itakura. Interpolating HRTF for auditory virtual reality. *Journal of the AcousticalSociety of America*, 100(4):2602–2602, 1996. (Zitiert auf Seite 32)

[NW09] M. A. Nees, B. N. Walker. Auditory interfaces and sonification. *The universal access handbook*, S. 507–521, 2009. (Zitiert auf Seite 23)

[ope] (Zitiert auf den Seiten 8 und 40)

[PC94] D. Pralong, S. Carlile. Measuring the human head-related transfer functions: A novel method for the construction and calibration of a miniature “in-ear”recording system. *The Journal of the Acoustical Society of America*, 95(6):3435–3444, 1994. (Zitiert auf Seite 32)

[Pen] S. Pennock. (Zitiert auf Seite 24)

[PTP09] T. J. Pinch, F. Trocco, T. Pinch. *Analog days: The invention and impact of the Moog synthesizer*. Harvard University Press, 2009. (Zitiert auf Seite 16)

[Ray07] L. Rayleigh. XII. On our perception of sound direction. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 13(74):214–232, 1907. (Zitiert auf Seite 16)

[RFCZ12] F. Ribeiro, D. Florencio, P. Chou, Z. Zhang. In *MMSP*. IEEE, 2012. (Zitiert auf Seite 25)

[RH11] S. Rosen, P. Howell. *Signals and systems for speech and hearing*, Band 29. BRILL, 2011. (Zitiert auf Seite 13)

[Rum01] F. Rumsey. *Spatial audio*. Taylor & Francis, 2001. (Zitiert auf Seite 27)

[SC05] J. W. Scarpaci, H. S. Colburn. A system for real-time virtual auditory space. In *In Proceedings of the International Conference on Auditory Display (ICAD 2005)*. 2005. (Zitiert auf Seite 24)

[Sch11] E. Schuchart. Zur Sonifikation genetischer Daten, 2011. (Zitiert auf Seite 37)

[Spe09] G. Spehr. *Funktionale Klänge: hörbare Daten, klingende Geräte und gestaltete Hörerfahrungen*, Band 2. transcript Verlag, 2009. (Zitiert auf Seite 21)

[SS73] W. Simpson, L. D. Stanton. Head movement does not facilitate perception of the distance of a source of sound. *The American journal of psychology*, S. 151–159, 1973. (Zitiert auf Seite 19)

[Stu01] B. L. Sturm. Composing for an ensemble of atoms: the metamorphosis of scientific experiment into music. *Organised Sound*, 6(02):131–145, 2001. (Zitiert auf Seite 35)

[Sum85] D. A. Sumikawa. Guidelines for the integration of audio cues into computer user interfaces. Technischer Bericht, 1985. (Zitiert auf Seite 22)

[TG98] C.-J. Tan, W.-S. Gan. User-defined spectral manipulation of HRTF for improved localisation in 3D sound systems. *Electronics Letters*, 34(25):2387–2389, 1998. doi: 10.1049/el:19981629. (Zitiert auf Seite 33)

[Vasa] P. N. Vassilakis. (Zitiert auf Seite 34)

[Vasb] P. N. Vassilakis. (Zitiert auf Seite 34)

[Ver99] S. A. Verlag. Lexikon der Biologie: aktives Zentrum, 1999. URL http://www.spektrum.de/lexikon/biologie/aktives-zentrum/1791. (Zitiert auf Seite 51)

[Wal40] H. Wallach. The role of head movements and vestibular and visual cues in sound localization. *Journal of Experimental Psychology*, 27(4):339, 1940. (Zitiert auf Seite 19)

[Wil12] C. Wilson. Webaudio Playground, 2012. URL http://webaudioplayground.appspot.com/#. (Zitiert auf den Seiten 7 und 17)

[WK59] F. M. Wiener, D. N. Keast. Experimental study of the propagation of sound over ground. *The Journal of the Acoustical Society of America*, 31(6):724–733, 1959. (Zitiert auf Seite 19)

[WNL06] B. N. Walker, A. Nance, J. Lindsay. Spearcons: Speech-based earcons improve navigation performance in auditory menus. In *Proceedings of the International Conference on Auditory Display, London, UK*, S. 63–68. 2006. (Zitiert auf Seite 22)

[Won05] S. Y. Won. Auditory display of genome data: Human chromosome 21. 2005. (Zitiert auf Seite 37)

[Woo38] R. S. Woodworth. Experimental Psychology. S. 520–525, 1938. (Zitiert auf Seite 32)

[Yeu80] E. S. Yeung. Pattern recognition by audio representation of multivariate analytical data. *Analytical Chemistry*, 52(7):1120–1123, 1980. (Zitiert auf Seite 35)

[Zah96] P. Zahorik. Auditory distance perception: A literature review. *Phd Preliminary Examination, University of West Maddison, Deptartment of Psychology*, 1996. (Zitiert auf Seite 19)

[Zah02] P. Zahorik. Direct-to-reverberant energy ratio sensitivity. *The Journal of the Acoustical Society of America*, 112(5):2110–2117, 2002. (Zitiert auf Seite 19)

[ZD08] M. ZÃ¼rcher, F. Diederich. Structure-Based Drug Design: Exploring the Proper Filling of Apolar Pockets at Enzyme Active Sites. *The Journal of Organic Chemistry*, 73(12):4345–4361, 2008. doi:10.1021/jo800527n. URL http://pubs.acs.org/doi/abs/10.1021/jo800527n. (Zitiert auf Seite 51)

[ZDD+02] D. Zotkin, R. Duraiswami, L. Davis, A. Mohan, V. Raykar. Virtual audio system customization using visual matching of ear parameters. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, Band 3, S. 1003–1006 vol.3. 2002. doi:10.1109/ICPR.2002.1048207. (Zitiert auf Seite 33)

Alle URLs wurden zuletzt am 05. 08. 2014 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift