TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

MASTERARBEIT

Ein virtuelles Turntable

Ausgeführt am Institut für
Department of Distributed and Multimedia Systems
der Universität Wien
unter der Anleitung von
Univ.Prof. Dr.techn. Wolfgang Klas
und
Mag. Stefan Leitich
als verantwortlich mitwirkendem Universitätsassistenten

durch

Nikolaus Weinmann
Hickelgasse 18/28
A-1140 Wien

Wien, am 30.11.2007

Inhaltsverzeichnis

Zusammenfassung

Im Bereich des DJ–Interfaces gibt es immer mehr Errungenschaften, die den Anspruch hegen die traditionelle Umgebung des DJ's zu verbessern. Als solche gilt die Idee einer kreisförmigen Darstellung der Metadaten, welche eine Projektion auf das Turntable ermöglicht. Diese Umsetzung basiert auf dem von Tue Haste Andersen proklamierten *Augmented Turntable*. Bei den projezierten Metadaten handelt es sich um eine Wellenformdarstellung der Amplituden sowie einer Markierung von Onsets, die den Beginn eines neuen Tons definieren. Ziel dieser Arbeit ist die Erstellung eines Prototyps, der sich, nebst der auditiven Umsetzung einer DJ–Umgebung, auf den eingangs beschriebenen visuellen Aspekt konzentriert. Es sollen die hierfür benötigten Teilbereiche erläutert und die Eigenschaften einer solchen Visualisierung diskutiert werden. Aufgrund der praktischen Umsetzung sind vor allem neue Erkenntnisse und Verbesserungsansätze dieser Visualisierungsform zu erwarten. Des Weiteren wird das als Programmiersprache gewählte Java hinsichtlich der Echtzeittauglichkeit im Audiobereich untersucht, welche als Grundvoraussetzung für eine erfolgreiche Umsetzung in diesem Bereich hervorzuheben ist.

1 Einleitung

> *Deshalb ist die einzige lebendige Sache, die mit einer Schallplatte geschehen kann, daß man sie auf eine Weise gebraucht, die etwas Neues entstehen läßt. Wenn man zum Beispiel mit Hilfe einer Schallplatte ein anderes Musikstück machen könnte, indem man eine Schallplatte oder andere Geräusche der Umwelt oder andere Musikstücke einbezieht, dann würde ich das interessant finden.* [John Cage]

Der Plattenspieler, im Englischen und umgangssprachlich *Turntable* genannt, hätte wohl aus wirtschaftlicher Sicht nach Einführung der Compact Disk (CD) 1980 bis heute schon in Vergessenheit geraten müssen. Im Vergleich zu seinem Vorgänger kann durch die digitale Technik vor allem durch bessere Fehlerkorrekturmöglichkeiten adäquate Qualität zu geringeren Kosten erzeugt werden. Außerdem ist gezieltes Manipulieren von Audio möglich. Ein größerer Dynamikumfang, kompaktere Abspielgeräte und -medien sind weiterere Vorteile. Doch zwei Gruppen, einerseits die Audiophilen, welche auf den Klang der Analogtechnik schwören und Diskjockeys, kurz DJs, sorgten weiterhin für Nachfrage.

Vor allem Letztere sind für die heutige Popularität und den Fortbestand dieses rund ein Jahrhundert alten Mediums verantwortlich. Grundsätzlich unterscheidet man zwei Arten des DJ's. Ein *Mix*–DJ arrangiert und manipuliert während des Auftritt in Echtzeit mehrere bereits bestehende Musikstücke jeglicher Stilrichtung, bzw. Teile daraus, zu einer Collage, einem neuen musikalischen Ausdruck, und/oder er synchronisiert dies mit Live-Instrumenten. Der *Scratch*[1]–DJ, auch genannt *Turntablist*, fokussiert sich auf die Manipulation des Tons durch Interaktion mit Turntable und Mischpult. Durch die schnellen, geübten und geschickten Bewegungen, die hierbei nötig sind, entsteht neben dem auditiven auch ein visueller Eindruck. Obwohl noch genauer differenziert werden kann, wird in dieser Arbeit davon Abstand genommen, da die Grenzen und Definitionen dieser meist ungenau sind.

In dem von mir behandelten DJ–Umfeld wird das Turntable weniger als reines Abspielgerät gesehen, als ein unersetzliches künstlerisches Mittel, ein Instrument, als Symbol einer Generation. Hierfür sind laut Beamish [Bea03] vor allem zwei Gründe zu nennen:

1. Das physische Medium selbst, welches Interaktion und Manipulation ermöglicht und durch haptische Rückkoppelung über Abspielgeschwindigkeit, –beschleunigung und –richtung Aufschluss gibt.

2. Die Umdrehung des Mediums, welches den Fortschritt des Musikstücks visuell darstellt.

Somit ist eine Unterteilung in einen haptischen und einen visuellen Aspekt sinnvoll. Vor allem die Interaktionsmöglichkeiten sind den Mögichkeiten der meisten digitalen

[1] Unter Scratchen oder Scratching (engl. „to scratch“ = „zerkratzen“) versteht man im DJ–Kontext das rhythmische Hin- und Herbewegen einer abspielenden Schallplatte, um neue Töne zu generieren. Ein hierbei wichtiges Instrument ist der Crossfader des Mischpults, welcher es ermöglicht Töne rhythmisch ein– und auszublenden.

Abspielgeräte überlegen. Der direkte und somit verzögerungsfreie Zugriff auf die Abspielgeschwindigkeit und die dadurch bedingte Manipulationsfunktionalität, in Kombination mit den Abmischmöglichkeiten eines Mixers und der Schallplatte, verhalfen dem DJ zu neuen Möglichkeiten im musikalischen Ausdruck. Die bekannteste Ausdrucksform das *Scratchen* kann nur durch optischen und physischen Kontakt zum Medium und der Musik optimal erbracht werden. Der visuelle Aspekt ist nicht einzig die sich drehende Schallplatte, sondern diese birgt auch gewisse Metadaten: die Dauer bzw. Dynamik eines Titels ist aufgrund von Unterteilungen bzw. Tiefe der Rillen sogar für Laien ablesbar. Ein geschultes Auge erkennt mitunter sogar den formalen Aufbau eines Stückes. Weiters ist hierfür auch der Winkel der Etikette (das sogenannte *Label*), welches in der Mitte der Schallplatte angebracht ist, ein optisches Hilfsmittel, wodurch leicht eine ganze bzw. eine teilweise Umdrehung der Schallplatte festgestellt werden kann. Beim *Cueing*[2]–Prozess werden auf der Schallplatte Aufkleber oder andere Markierung angebracht, welche ebenfalls der Orientierung dienen.

Diese Arbeit befasst sich mit den neuen Technologien im DJ–Bereich, wobei der Schwerpunkt auf der Emulation des Schallplattenspielers liegt. Die Forschung lässt sich hierbei in visuelle sowie haptische Aspekte unterteilen. Der vorliegende theoretische Teil befasst sich mit beiden Themengebieten. In der Einleitung wird zuerst eine geschichtliche Entwicklung des Turntables behandelt, die gleichzeitig einen gewissen Einblick in die Praktik und Terminologie ermöglicht. Ein erster Schwerpunkt wird durch die Vorstellung der neuen DJ Interfaces gesetzt, die hierbei verwendete Unterteilung in haptische und visuelle Rückkoppelung wurde bereits Eingangs erwähnt. Das Kapitel „Digitale Tontechnik“ soll das Grundverständnis für die Abspielgeschwindigkeitsmanipulation, welche von fundamentaler Bedeutung im DJ–Bereich ist, und die Onsetdetection liefern. Abschliessen wird die Arbeit mit der Beschreibung der Umsetzung des Prototypen, welcher im folgenden Absatz kurz dargestellt wird.

Der in dieser Arbeit behandelte Prototyp orientiert sich an den visuellen Aspekten des von Tue Haste Andersen [And03] proklamierten *Augmented Turntable*. Durch eine kreisförmige Darstellung wird eine Projektion auf einem DJ–Interface ermöglicht, wobei sich diese an die traditionelle Bedienbarkeit des Turntable anlehnt. Der Prototyp wird anhand einer reinen Softwareimplementierung die benötigten Bestandteile einer solchen Umsetzung vorstellen. Neben den bereits erwähnten Geschwindigkeitsmanipulation, kreisförmiger Visualisierung und Onsetdetection ist hierbei wird noch ein physikalisches Modell benötigt. Des Weiteren wird die Methodik der Umsetzung erläutert, da jede Form unterschiedliche Begebenheiten birgt. Dies gilt besonders in Hinblick auf die Latenzzeit, welche einen wesentlichen Einfluss auf das Resultat hat.

[2]Unter Cueing versteht man das Auffinden und die Identifizierung bestimmter Stellen eines Musikstücks.

2 Die geschichtliche Entwicklung

2.1 Die Schallplatte

1589 verfasste der italienische Physiker Giovanni Battista della Porta den Gedanken gesprochene Worte in einem Behältnis aufzubewahren. Jedoch schlugen jegliche Versuche der „Konservierung des gesprochenen Worte“ fehl. Im Jahre 1807 erfand der englische Gelehrte Thomas Young ein Gerät welches die Anzahl und Stärke von Schwingungen mittels einer Stimmgabel maß. Diese, mit einer Nadel versehen, schrieb eine gewellte Kurve auf einen Zylinder. 50 Jahre später 1857 gelang es wiederum einem Engländer mit seinem Phon–Autographen die Schallschwingungen sichtbar zu machen. Die Wiedergabe dieser Daten war jedoch nicht möglich.

1877 gelang es dem berühmten Erfinder Thomas Alva Edison die menschliche Stimme einzufangen und wiederzugeben: Ein mit Paraffin überzogenes Papier wurde an einen mit einer Nadel versehenen Membran vorbeigezogen. Als er nun das Wort „Hello“ gegen die Membran sprach und anschließend die Nadel wieder über das Papier zog vernahm er leise seine eigene Stimme. Dies war das erste historisch verbürgte konservierte Wort. Einige Monate darauf entstand ein überarbeitetes Model, er nannte es Phonograph.

1887 meldete Emil Berliner sein Grammophon zum Patent an, welche dem Phonographen von Edison sehr ähnlich war, doch er umging rechtliche Probleme, indem er in der technischen Umsetzung leicht variierte. Er versprach sich von der dazugehörigen Schallplatte eine leichtere Duplikationsmöglichkeit als bei den von Edison verwendeten Walzen. Bei der Produktion von 500 Walzen mit dem gleichen Lied wurden 5 Phonographen aufgestellt und der Sänger musste diese 100-mal bespielen. Ausdauer und vor allem eine kräftige Stimme waren die Anforderungen an die ersten Tonträgerkünstler. Erst 1903 entwickelte Edison ein Verfahren, um seine Walzen zu kopieren, die Goldgusswalzen. Die Schallplatte erwies sich als besseres Medium und setzte sich gegenüber den Walzen durch. Aus der ursprünglich aus Wachs überzogenem Zink und später aus Celluloid bestehende Schallplatte entstand in Folge die Schellackplatte, welche die Plattenindustrie die nächsten 60 Jahre dominierte.

Um das Konzept einer Stereoplatte zu verstehen, muss man zuerst die Plattenschriftarten erläutern. Die Tiefenschrift (auch Vertikalschrift genannt) wird die Eintauchtiefe des Tonabnehmers für die Gewinnung des Tonsignals herangezogen. Bei der Seitenschrift (Horizontalschrift) wird die hozizontale Auslenkung der Rille ausgewertet. Die Auslenkung ist direkt proportional zur Amplitude des Signals. Für eine Stereowiedergabe können nun die beiden Modulationen genutzt werden, dies entspricht dem sogenannten „+“–System. Der Name resultiert aus der Anordnung der Tonabnehmer (eine horizontale und eine vertikale Linie ergeben ein Plus). Jedoch ist die Tiefenschrift der Seitenschrift, welche einen höheren Dynamikbereich und geringere Verzerrungen vorweist, unterlegen. Das „+“–System hat also keine Möglichkeit beide Kanäle in selber Qualität abzuspeichern. Den Kompromiß stellte das sogenannte „x“–System dar, welches die Tonsignale beider Kanäle in um 45° gegen die Senkrechte geneigte Schwingungen umsetzte (hieraus lässt sich der Name ableiten: Die Drehung der Plus–Anordnung um 45° ergibt ein „x“). Einfacher gesagt wird hierbei jeder Kanal halb vertikal und halb horizontal abgetastet. Das System bietet mit der vollkommene Kompatibilität zur Monoaufzeichnung einen weiteren Vorteil und setzte sich schlussendlich gegenüber dem

„+“–Systems durch.

Das teure Naturprodukt Schellack wich dem preiswerteren synthetischen Vinyl, wobei durch schmalere Rillen kleinere Abtastnadeln ermöglicht wurden, dadurch sich die Tonqualität verbesserte und die Abspieldauer erhöhte. Der Durchbruch der Vinylplatte wurde durch einen weiteren Formatkrieg, hauptsächlich eine Frage der Größe und Abspielgeschwindigkeit, noch ein wenig gebremst. Wobei Columbia Records 1948 mit der 12–Inch(entspricht 30,48 cm)–Langspielplatte (LP) mit $33\frac{1}{3}$ RPM[3] und kleinem Mittelloch und RCA Victor 1949 mit der 7–Inch(17,78 cm)–Schallplatte (EP) mit 45 RPM und größeren Mittelloch die Hauptvertreter waren. Nach einer Formatbereinigung und damit verbundenen Plattenspielern, die beide Formate abspielen konnten, gelang in den 60er endgültig der Durchbruch.

Das Prinzip ist all die Zeit immer gleich geblieben: Die Schallplatte ist eine runde Scheibe auf der beidseitig analogen Signale aufgezeichnet werden können. Der Schall wird mechanisch in einer spiralförmigen Rille von Außen zur Mitte aufgezeichnet, das sogenannte Nadeltonverfahren, und kann dann mit einer bestimmten Geschwindigkeit oder elektromechanisch abgetastet werden. Die Spieldauer der Langspielplatte beträgt 20 bis 25 Minuten. Die Spieldauer ist von mehreren Faktoren abhängig: von Größe und Art der Schallplatte, der Dynamik und Qualität der Aufzeichnung und je nachdem wie tief zur Mitte geschnitten wird. Hohe Dynamik und hoher Bassanteil, bedingt durch höhere Auslenkungen, wirken sich negativ auf die Spieldauer aus. Aus diesem Grunde werden bei der Aufzeichnung die Höhen angehoben (Preemphasis) und die Tiefen abgeschwächt, was bei der Wiedergabe von einem Vorverstärker wieder entzerrt wird. Der durch technische Vorgaben gegebene geringe dynamische Umfang ermöglicht keine realistische Wiedergabe beispielweise eines Orchesters wo eine Dynamik von *ppp*[4] bis *fff* üblich ist. Aber auch in Klaviermusik sind sehr große Abstufung möglich, welche das Medium nur eingeschränkt wiedergeben kann. Ein weiteres Manko ist die unterschiedliche Spurgeschwindigkeit, die vom Rand gegen die Mitte der Schallplatte abnimmt. [Wik07h] [Wei04]

2.2 Der Plattenspieler

Bei DJs sind die Modelle *SL1200* und *SL1210 (MK–II)* von Technics sehr beliebt, durch den starken Direktantriebmotors kann die eingestellte Drehgeschwindigkeit nach dem Abstoppen schneller wieder aufgenommen werden. Neue Modelle aus dem DJ Bereich übertreffen diese beim Drehmoment sogar. Im *HiFi*[5]–Bereich sind Riemenantriebsmotoren beliebter, da diese gleichmäßiger laufen und unerwünschte Vibrationen vermindern. Während bei dem Direktantrieb der Plattenteller den Motor eines elektrischen Wechselstrommotors stellt, wird bei dem Riemenantrieb die Last mittels Riemen übertragen. Bei letzteren entsteht somit eine zusätzliche Reibung, wodurch die Platte schneller zum

[3] RPM steht für Rounds per minute, Umdrehungen pro Minute.

[4] Dynamikangabe, in diesem Fall das dreifache piano (ital. „still, leise, zart“), abgekürzt *p*. Das Gegenteil bezeichnet man forte (ital. „stark, laut, kräftig“), abgekürzt *f*. Die Anweisungen können durch den Buchstaben *m* wie mezzo (ital. „mittel, halb“) abgeschwächt werden. Im Gegensatz dazu können die Buchstaben zur Steigerung verdoppelt werden: *ff* steht für *fortissimo* („sehr laut“). In Laufe der Zeit entstanden dann weitere Abstufungen wie *ppp* („pianissimo piano“) und *fff* („fortissimo forte“). György Ligeti schreibt sogar achtfaches piano bzw. forte vor.

[5] HiFi ist die Kurzform von High Fidelity, steht für hohe (Klang-)Treue.

Stillstand kommt. Die Reibung des Tonabnehmers ist hierbei negierbar, sie resultiert aus dem Gegengewicht des Tonabnehmers. Ein weitere und heutzutage unübliche Antriebsform ist die der Reibradübertragung, dessen größten Manko die Übertragung der Vibration durch starre Koppelung zwischen Motorachse und Plattenteller ist.

Der Motor treibt also den Plattenteller an, worauf sich die Schallplatte befindet und wird von zwei binären Tasten gesteuert: Die *Power*–Taste steuert die Stromzufuhr und *Play/Pause*–Taste eine Bremsvorrichtung (dies wird bei dem *SL1200* durch eine magnetische Steuerung realisiert). Folglich gibt es zwei Arten den Plattenteller zum Stehen zu bringen, einerseits diesen per *Play/Pause*–Taste mit dem Motor abzubremsen, oder indem man per *Power*–Taste die Stromzufuhr abstellt somit den Teller nur durch Reibung auslaufen zu lässt.

Rechts vom Plattenteller ist der Tonarm, an dessen einem Ende sich der Tonabnehmer mit der Nadel und am anderen Ende ein verstellbares Gegengewicht, um das Auflagegewicht der Nadel (in der Regel im Bereich von 2-6g[6] einzustellen, befindet. Weiters ist am Tonarm eine Anti–Skating–Einrichtung vorhanden, die zur Kompensation der Skatingkraft des Tonarmes dient d.h. der Tonarm wird beim Abspielen stets durch die Rille in der Schallplatte etwas nach innen gezogen und die Abtastnadel durch die Massenträgheit des Tonarmes an die äußere Rillenflanke gedrückt, das wird durch diese Einstellung (meist in der Größe des Auflagegewichts) kompensiert. Legt man nun den Tonabnehmer auf die gewünschte Stelle der Schallplatte wird diese beim laufenden Motor über die spiralförmig angelegte Tonspur der Schallplatte fahren.

Ein Plattenspieler verfügt zu meist über zwei Grundgeschwindigkeiten $33\frac{1}{3}$ und 45 RPM, einige Geräte können auch mit 78 RPM laufen, um auch Schelllacks abspielen zu können. Weiters gibt es für die Abspielgeschwindigkeit einen stufenlosen Regler, dessen Bereich üblicherweise -8% bis +8% umfasst. Es exitieren jedoch Umbauten die diesen Bereich auf +/-30% ausweiten. Man spricht hierbei von einem *pitchbaren*[7] Schallplattenspieler, da durch die Veränderung in der Abspielgeschwindigkeit systembedingt auch die Tonhöhe variiert wird. Daher wird dieser Bestandteil Pitchregler genannt. [Wik07h]

2.3 Der Diskjockey

Die Ursprung des DJ's ist laut Kjetil Falkenberg Hansen in seiner Arbeit *Turntable Music* [Han00] schwer zu bestimmen. Er nennt einerseits Edison 1877 oder Berliner 1887, die den Weg durch ihre technischen Errungenschaften geebnet hatten. Anderseits einige Vertreter der *E-Musik*[8] darunter sehr bekannte Persönlichkeiten wie Paul Hindemith, Ernest Toch, Percy Grainger, Edgar Varèse und Darius Milhaud in den Zwanzigern, welche allesamt mit Turntables oder Abspielgeschwindigkeitsmodifikationen experimentierten. Besonders hervorzuheben sind John Cage mit seinem Werk *Imaginary Landscapes #1* 1939, er verändert darin die Abspielgeschwindigkeit von Testtönen, und Pierre Schaeffer Begründer der *musique concrète*. Die Idee dieser Strömung wird von seinem Schüler und Vertrautem Pierre Henry wie folgt beschrieben:

[6] Wobei im HiFi–Bereich 2-3g üblich sind, jedoch im DJ–Bereich ein Gegengewicht von >3g zum Einsatz kommen, dadurch wird eine höhere Spurtreue erzielt, aber gleichzeitig auch mehr Abtrieb.

[7] Pitch (engl. für Tonhöhe).

[8] E-Musik ist eine Abkürzung für die sogenannte „ernste“ Kunstmusik, die „ernst zu nehmende“ oder „kulturell wertvolle“ Musik [Wik07d].

> *Musique concrète entsteht aus zwei Phasen: Erstens die Isolation des Tons, einen neuen Anfang und ein neues Ende für etwas bereits Bestehendes schaffen. Zweitens dieses im Aufnahmestudio zu expandieren, transformieren und transponieren. [Kha97]*

Obwohl diese Tätigkeit die dem des DJ's sehr ähnelt, hatten Schaeffer und Henry den Schwerpunkt auf das Studio verlegt und Turntables größtenteils außer Acht gelassen. So tritt es erst wieder durch Christian Marclay in Erscheinung. 1979 setzte er Turntables erstmalig in Interaktion mit anderen Instrumenten ein und unterstrich somit den instrumentalen Charakter des Turntables. Sein Schaffen, welches von Cage und Schaeffer geprägt ist, sich aber stärker am Geräuschhaftigen orientiert. Die Abnutzung des Vinyls bildete das Grundkonzept. Er manipuliert, verformt und zerteilt sogar Platten, um sie anschließend wieder zusammenzukleben oder lässt das Publikum über die Schallplatten gehen, welche er in Folge für seine Auftritte verwendete. Bei einer nicht im direkten Zusammenhang stehende Kunstaktion *Footsteps* ließ er den Boden einer Ausstellung mit 3500 Schallplatten auslegen, welche dann verpackt und verkauft wurden. Marclay versuchte damit auf die Minderwertigkeit einer musikalischen Reproduktion aufmerksam zu machen.

Viel mehr jedoch prägend für die heutige DJ-Kultur war eine andere Strömung: die des *Hip-Hops* der Mitt–70er in Bronx, New York. Heute ist der allgemeine Konsens, dass drei Personen für die Entwicklung der DJ–Technik ausschlaggebend waren. DJ Kool Herc, Grandmaster Flash und Afrika Bambaataa entwickelten eine Technik, welche zwei Turntables, ein Mischpult und ein paar identischer Schallplatten verwendete und die selben Segmente immer wiederholte. Dies wurde in Folge *Mixing* genannt. [Han00]

2.4 Das Mischpult

Revolutioniert wurde diese neue Kunstform durch Grandmaster Flash, welcher das Mischpult, dank seiner technischen Versiertheit, den neuen Anforderungen anpassen konnte. Er ersetzte die zwei Lautstärkeregler durch einen *Crossfader*, welcher beide Eingänge steuerte. Wenn dessen Regler sich auf der ganz linken Seite befindet, isoliert das Mischpult den Ton des linken Turntables, und vice versa auf der rechten Seite das andere Turntable. Befindet sich der Knopf in der Mitte, so werden beide Quellen gleich laut wiedergegeben. Dadurch wurde die Steuerung mit nur einer Hand möglich. Hier wird auch deutlich, dass das Mischpult eine enorme Bedeutung in diesem Bereich hat und Analysen von Scratching Aufführungen bestätigen diese Aussage [Han03a].

Es bildet das Bindeglied zwischen Lautsprecher und Turntable, möglicherweise noch über einen Verstärker, jedes Signal geht somit über das Mischpult. Es verfügt über eine begrenzte Anzahl von Eingangskanälen, welche die Anzahl der daran angeschlossenen Geräte limitiert. Standardmäßig sind es zwei Eingänge. Es befinden sich des Weiteren der Line/Phono Schalter, einige Regler für die Tonkontrolle und meistens auch für Effekte. Der Crossfader wird im DJ Bereich am häufigsten verwendet. Während der letzten Jahre hat er sich vom linearen zu einem stark absteigenden logarithmischen Abklingen entwickelt, welche in der Praxis nun mehr einem schnellen Ein- und Ausschalten des Tones gleichkommt, wodurch er die Funktionalität des Line/Phono Schalters übernommen hat [Han06].

2.5 Scratchen

Das bereits erwähnte Scratchen entstand durch einen Zufall. Während Grandwizard Theodore (Alias von Theodore Levingston) 1977 mit seiner Mutter über die Lautstärke der Musik stritt und er die aktuelle Position der Schallplatte zu halten versuchte, bewegte er die Schallplatte unabsichtlich vor und zurück und entdeckte somit eine neue musikalische Ausdrucksform. Nachdem er diese Technik bei seinen Auftritten offenbarte, wurde dies von anderen DJs übernommen und in Folge immer weiterentwickelt [Han00].

3 Das Zusammenspiel Interface und Scratching

Beim Scratching verwendete *Samples*[9] sind meist alltägliche, vertraute Töne, welche durch das Scratchen bis zur Unkenntlichkeit verfremdet werden. Eine Veränderung findet nicht nur im Zeitbereich statt, sondern der komplette Frequenzbereich wird beeinflusst. Das Sample selber ist von großer Bedeutung, kurze Töne niedriger Frequenz sind schwerer zu dehnen als lange Töne, um sie dann per Crossfader manipulieren zu können. Nicht nur der Ton selbst, auch das dem Ton Vorhergehende und Nachfolgende muss beachtet werden. Wenn der DJ die Platte vor Tonbeginn in den Ton bewegt, beginnt der Ton mit einem *high pitch*, einer hohen Tonlage. Bewegt man sie während des Tons ist ein plötzlicher *Glissando*[10]–Effekt zu vernehmen. Bewegt man die Platte immer über den ganzen Ton, entsteht eine Abfolge scharfer *Onsets*[11]. Bewegt man sich andererseits innerhalb eines Tons, so entsteht ein sirenenähnliches Geräusch.

Mit dem *Onset* eines Tons meint man den Beginn eines Tones und dessen Charakteristik. Ein langsamer Anschub der Schallplatte resultiert in einen langsamen Anstieg im Pitch und einer „weichen“ Tonbeginn, anders ein schneller Anschub der eine „härteren“ Tonbeginn verursacht. Töne an sich haben auch unterschiedliche Formen des Tonbeginns zur Folge, ein Blasinstrument beispielsweise hat eine „weichere“ als ein Schlaginstrument. Die schnelle Abfolge von Onsets ist spezifisch für das Scratching. Bei typischen Techniken können in ein paar wenigen Zehntel Sekunden 6-9 Onsets vorkommen.

Kjetil Falkenberg Hansen unterteilt in [Han06] weiters in folgende Kontroll- und Tonausgabeparameter. Unter Kontrolle zählt er die Plattengeschwindigkeit, die Schallplatte als Tonquelle, die Position des Tonabnehmers und auf dem Mischpult der Crossfader, den Lautstärkenregler und die Tonregler, welche in Bezug auf Tonhöhe, Ton Onsets und Dauer, und Toncharakteristika, wie Klangfarbe und Dynamik Einfluss nehmen. Des Weiteren hängt Klangfarbe von Ausgangsmaterial und Abspielgeschwindigkeit ab. Die Dynamik wird durch den Lautstärkenregler auf dem Mischpult, die Quelle und Abspielgeschwindigkeit bestimmt. Der Pitch wird direkt durch die Abspielgeschwindkeit eines Tones gesteuert. Viele DJs verwenden den Pitchregler auch um die Geschwindigkeit auf einander folgender Stücke anzupassen. Beim Scratchen wird diese Funktion angewandt, um das Material auf das Stück in Bezug auf Tonfarbe abzustimmen. Die meisten der gescratchten Töne erreichen ungefähr die Geschwindigkeit der rotierenden Platte [Han03a].

Die Abbildung 1 aus [Han06] soll das Zusammenspiel im traditionellem DJ Setup verdeutlichen, wobei die Stärke der Linien den Ausmaß des Einflusses zwischen Kontroll– (Mitte) und Tonausgabeparameter (links und rechts) darstellen. Am wichtigsten in diesem Diagramm sind bei den Kontrollentitäten die fett umrahmten Felder und die rechts angeordneten Ausgabeparameter.

[9] Ein Sample stellt hierbei ein Ausschnitt einer (Musik-)Aufnahme dar, welcher in Folge in einem anderen musikalischen Kontext verwendet wird.

[10] Der Begriff Glissando (auch *glissato, glisscato, glissicando*; vom französischen *glisser* „gleiten“ abgeleitet) bezeichnet in der Musik die kontinuierliche (gleitende) Veränderung der Tonhöhe innerhalb eines größeren Intervalls [Wik07e].

[11] Ein Onset steht für den Beginn einer neuen Note oder eines neuen musikalischen Teils. Die Technik aus dem *Music-Informationretrival*, die der Berechnung dieser dient, bezeichnet man als *Onsetdetection*.

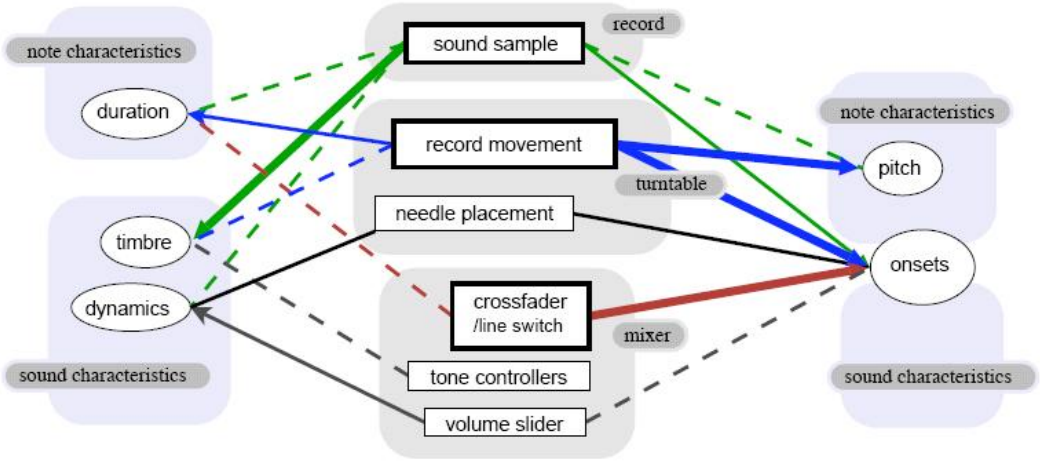


Abbildung 1: Die Parameter bei Scratching [Han06]

4 Die Möglichkeiten des digitalen Audios

Wenn bei einer analogen Aufnahme die Geschwindigkeit der Tonaufzeichnung nicht exakt konstant bleibt, wird eine gültige Wellenform in eine andere Wellenform transformiert, des Weiteren können Spannungsfehler auftreten. Es ist also bei der analogen Technik so, dass Fehler, die sich durch Qualitätsverlust bemerkbar machen, nicht entdeckt werden können. Tritt bei einer digitalen Aufnahme ein Wert zur falschen Zeit auf, ändert sich die Frequenz und der Fehler kann erkannt werden. Um solche Abweichungen zu vermeiden, werden die Samples in den Speicher geladen und dort mittels einem stabilen, lokalen Taktgeber ausgelesen. Sie ermöglicht nun im Gegensatz zu seinem Vorgänger einerseits, dass Fehler vermieden werden können, und anderseits, dass man nun diese Samples gezielter manipulieren kann. Die Produktion bei analogen Medien ist also deutlich aufwändiger, wobei sich dies in höheren Produktionskosten auswirkt, als in der mit einer überlegenen Fehlerkontrolle ausgestatteten Digitaltechnik. Aber auch gleichwertige Abspielgeräte sind im Vergleich günstiger als die des analogen Pendants, was sich also in besserer Tonqualität bei Produkten derselben Preiskategorie wiederspiegelt. Der, dem der Schallplatte weit überlegene, Dynamikumfang ermöglichte eine realistischere Reproduktion von Orchester- und Klaviermusik, dies ist im DJ Bereich jedoch von geringer Bedeutung.

Bei der Lebensdauer ist die CD zu favorisieren, vor allem wenn man die *Backup*–Möglichkeiten[12] in Betracht zieht. Wobei erwähnenswert ist, dass die propergierte Lebensdauer der CD von 100 Jahren äußerst umstritten ist, da bereits Datenverluste bei frühen Produktionen auftraten [Hei07]. Der Leiter des Lehrgebietes Multimedia und Internetanwendungen der Fernuniversität Hagen, Matthias Hemmje beklagt im selben Artikel: „Unter idealen Lagerungsbedingungen gebe ich einer CD 50 bis 80 Jahre Lebensdauer.“. Eine „Lagerung unter idealen Bedingungen“ hieße jedoch, dass die Cds nicht angefasst und staub– sowie lichtgeschützt in klimatisierten Räumen aufbewahrt werden müssten, der Temperatur 18 Grad Celsius nicht übersteigt. Noch kurzlebiger seien selbst gebrannte CDs. Empfehlenswert sind hierbei Backups auf alternativen Speichermedien. Bei Schallplatten kann falsche Lagerung zu Verformungen führen, und dies in einer unregelmäßigen Abspielgeschwindigkeit und dadurch in einer verfälschten Wellenform, resultiert. Vor allem durch das Scratchen wird eine Schallplatte in Mitleidenschaft gezogen, es wird die Klangqualität durch die Abnutzung von Platte und Nadel beeinträchtigt. Wobei die Nadel des Tonabnehmers auch bei normaler Verwendung unter Abnutzung leidet, diese muss gegebenenfalls ausgewechselt werden. Sein digitales Pendant, die Laserlinse, kann wiederum durch Verstaubung oder durch gewöhnliche Abnutzung an Signalstärke verlieren.

Der Tonarm kann bei versehentlichem Berühren oder bei zu schnellen Bewegungen aus der Rille springen und so zu unerwünschten Ergebnissen führen. Außerdem erschwert der Tonarm die Such- und Interaktionsmöglichkeiten, da man bei einer vollen Umdrehung an diesem anstoßen würde. Des Weiteren können Vibration das gewünschte Ergebnis manipulieren.

Durch die digitale Technik wurde auch die Portabilität der Musikdaten gegenüber

[12] Unter Backup versteht man die teilweise bzw. komplette Kopie von Daten auf ein alternatives Speichermedium abzulegen, um Datenverluste zu vermeiden.

Schallplatten um ein Vielfaches gesteigert. Außerdem hat man Zugriff auf mehrere Medien, wenn der DJ nicht auf die Schallplatte eingeschränkt wird, da ein Datentransfer wohl von Analog zu Digital, nicht jedoch umgekehrt, möglich ist und die Auswahl im digitalen Bereich überwiegt.

Dadurch, dass der Zugriff auf das Signal möglich ist, wird somit dessen direkte Manipulation möglich. Dies nennt man dann im Allgemeinen *Digital Signal Processing* (DSP). Für die digitale Signalverarbeitung ist der direkte Zugriff zwar von fundamentaler Bedeutung, aber diese Darstellungsform ist stark limitiert. Um die Möglichkeiten zu erweitern, wird die auf der Zeitachsen basierte Darstellung in die Frequenzdomäne umgewandelt (siehe Abschnitt 6.5). Hierfür wird die *Fast Fourier-Transformation* (FFT) verwendet, welche die vorhandenen Frequenzen im abgetasteten Signal untersucht. Nun können die Frequenzen beliebig bearbeitet werden und in Folge kann durch die inverse Funktion wieder die ursprungliche Form, ein abspielbare Signal, erzeugt werden.

Unerwünschte Frequenzen, wie Störgeräusche, können eliminiert, sprich gefiltert, werden. Bestimmte Frequenzen können entweder betont oder weggeschalten werden, dies wird durch einen *Equalizer* realisiert. Aber auch die Dauer eines Tones kann manipuliert werden, dies ist vor allem von größter Bedeutung im DJ Bereich und die jeweiligen Techniken werden in Abschnitt 7 noch detailliert betrachtet. Den Entwicklungen von Effekten, Synthesizern, Klangbibliotheken und -erzeugung sind keine Grenzen gesetzt.

Doch nicht nur Manipulation eines Tones, sondern auch die Generierung von *Metadaten*[13] ist von großer Bedeutung. Diese Disziplin wird *Music Information Retrival* (MIR) genannt und eröffnet immer mehr Möglichkeiten für den visuellen Aspekt. Die einfachste Möglichkeit einen Ton visuell darzustellen ist die Wellenform, dies wird in Abschnitt 6.1 vertieft. Bereits in den Kontext von Scratchen wurden Onsets gesetzt, welche einen Schwerpunkt dieser Arbeit bilden. Weiters ist es erdenklich, einer *Waveform* weitere Daten hinzuzufügen, welche über die Art des Tones, beispielsweise durch Farbgebung, Auskunft geben können.

Bei einer Gegenüberstellung der Vor- und Nachteile analoger und digitaler Medien fallen die erweiterten Möglichkeiten im Digitalbereich auf. Jedoch ist der haptische, direkte und stufenlose Zugriff auf die Abspielgeschwindigkeit ohne jegliche Latenz und die daraus entstandenen DJ-Techniken, auch aus Gründen der Bedienbarkeit und Tradition, nicht mehr wegzudenken. Die immerwährend steigende Leistungsfähigkeit der Computer würde jedoch eine Synergie beider Techniken mit neuen Interfaces ermöglichen. Da in der Entstehung neuer musikalischen Ausdrucksformen Experimentieren und Modifizieren eine wichtige Rolle spielen, ist bei der Entwicklung neuer Systeme Offenheit ein wichtiger Aspekt.

[13] Daten über Daten oder Information über Information. Eine Musikdatei kann zusätzlich zu den einzelnen Samples noch u. a. Stückdauer, Stilrichtung und Entstehungsjahr beinhalten. Bei der Schallplatte kommt dies dem Label und den auf der Schallplatte ersichtlichen Rillen gleich.

5 Die neuen DJ Interfaces

Andersen [And03] unterteilt DJ Systeme in vier Klassen:

1. Das rein analoge System,
2. das rein digitale WIMP (Windows, Icons, Mouse, and Point-and-Click) Interface,
3. das digitale Interface, welches die Interaktion des analogen Systems emuliert, und
4. Lösungen, welche Synergieeffekte beider Systeme vereinen.

Die grundlegendste Technik ist die des (2) WIMP. Diese erweitert die bisher verfügbaren Funktionalitäten, ermöglicht die Einbindung digitaler Daten, bietet aber einen völligen differenten Ansatz des Interfaces, auf Kosten einiger Vorteile der Schallplatte. Um diesem entgegen zu wirken, entstanden Hardwarelösungen, deren Userinterface der traditionellen Umgebung des DJs gleicht. In den letzten Jahren hat es einen Boom in kommerziellen Markt für DJ bezogene Produkte gegeben, auch die akademische Forschung befasst sich mit diesem Thema. Das Ziel ist allgemein den DJs neue Möglichkeiten zu eröffnen, welche bereits oberflächlich erörtert wurden, ohne hierbei die Vorteile der Schallplatte, namentlich die des visuellen und haptischen Feedbacks, zu verlieren. Bei Scratching war der erste Schritt das Turntable mit alternativen Kontrollern, z.B. anhand eines drehenden Rads, nachzubilden, dies steht jedoch laut Hansen [Han06] im Konflikt mit den von Cook definierten Design Prinzipien [Coo00]. Diese besagen, dass eine Replikation des Instruments nicht erstrebenswert ist, sondern die Verbesserung dieses oberste Zielsetzung ist. Außerdem stellt Hansen [Han03a] fest, dass bei der Entwicklung neuer Alternativen dem Mischpult eine genauso große Beachtung gegeben werden sollte wie dem Turntable. Diese zwei Geräte bilden die Kontroll- und Interfaceeinheit, welche von der Audioquelle der Schallplatte abhängen. Jedoch wird bei den neuen Entwicklung immer eine Trennung dieser beiden angedacht, da wohl so Hansen [Han06] weiter der Schwerpunkt auf die akzeptable Abbildung des Turntables gelegt wird. Somit entstanden in letzter Zeit zwar viele gute virtuelle Turntables jedoch blieben Lösungen die über zusätzliche Parameter Kontrolle ermöglichen ein unterentwickeltes Gebiet. Dieser Abschnitt soll einen Überblick über den aktuellen Stand dieses Gebiets verschaffen. Wie einleitend bereits erwähnt, geht es vor allem um zwei Faktoren, welche hierbei beachtet werden müssen: Das physische und das visuelle Feedback.

5.1 Die haptische Rückkoppelung

Um den Turntable Metapher zu generieren, welcher das physische Feedback ermöglicht, benötigt es ein sogenanntes *tangible*[14] Interface.

5.1.1 Die originalgetreue Emulation

Der *CDJ-1000* [Pio07] von *Pioneer*, ein CD-Player, versucht, den Plattenspieler über eine 7-Inch Drehscheibe auf der Einheit zu emulieren. Durch Wegfall des Tonarms können Umdrehungen ohne Störfaktor stattfinden, jedoch wurde die Einheit nicht mit einem

[14] Wörtlich übersetzt: greifbar, fühlbar oder konkret.

Motor ausgestattet, wodurch ein wichtiger Aspekt, die des physischen Feedbacks, verloren geht. Der Numark [Num07] folgte mit einer motorisierten 12-Inch Drehscheibe, wodurch das Turntable sehr original getreu nachgebildet wurde. Einen anderen Ansatz bietet *Final Scratch* [Sta07] von Stanton, welches das wohl am meisten verbreitete und akzeptierte Produkt ist, mit dem digitales Material manipuliert werden kann. Es benutzt ein eigenes Interface und zeitkodierte Schallplatten, dessen Signale die Audiodaten am Computer über ein unmodifiziertes Turntable steuern. Diese Gruppe von Produkten verfolgen das gleiche Ziel: eine original getreue Abbildung des Turntables. Dies bietet jedoch keinerlei neue Einsichten in Bereich der Steuerung oder der Abbildungsstrategien, es bietet keinerlei Vorteile außer dem Zugriff auf digitale Medien. Das DJ Set bleibt unverändert, nur ein Laptop ersetzt die Schallplatten und bietet zusätzlich Visualisierungen, die Aufführungstechniken bleiben dadurch ebenso unberührt. Desweiteren haben alte Probleme weiter Bestand, das Vinyl wird abgenutzt, zerkratzt und die Nadel springt oder wird beschädigt. Anderseits ist *Final Scratch* gerade deswegen auch sehr beliebt, da laut Beamish [Bea04] vor allem Turntablist neuen Interfaces gegenüber sehr skeptisch sind. Verallgemeinert kann man sagen, dass während Mix–DJs neue Möglichkeiten erkennen, jedoch Turntablists dies als Akt des Schummelns verstehen.

5.1.2 Ms Pinky

Eine sehr ähnliche Umsetzung ist *Ms Pinky*, dessen große Beliebtheit resultiert aus der Flexibilität, da man per Max/MSP[15] [Puc07] bestehende Module erweitern und jegliche Daten steuern kann. Durch die zeitkodierte Schallplatte werden Informationen über absolute Position und Geschwindigkeit an Max/MSP geliefert, womit man sich die Vorteile dieser Entwicklungsumgebung zu Nutze macht. Durch die gewährleistete Offenheit ermöglichte *Ms Pinky* Visualisierungsformen wie das Video–*Scratching* und *Visual Scratch* [Kri07].

5.1.3 D'Groove

D'Groove von Beamish et al. [Bea03], [Bea04] zeigt einige Vorteile der Migration zu digitalen Medien, wobei darauf geachtet wird, dabei nicht den Benutzkomfort des traditionellen DJ Setup zu beeinträchtigen, sondern neue Aufführungsmöglichkeiten zu eröffnen, welche mit analogen Medien nicht möglich sind. Er konzentriert sich verstärkt auf die Tatsache, dass ein Turntable durch den Motor und Nadelposition eine optische und fühlbare Auskunft, über die Abspielgeschwindigkeit, -richtung und -fortschritt liefert und erweitert dieses Konzept um *haptic force feedback* und musikalisch intelligenten Kontrolleinheiten. Sein Setup besteht aus vier Einheiten: einem Turntable, einem Pitchregler, einem Queue–Regler und einer Softwareumgebung.

Das D'Groove Turntable besteht aus einer Schallplatte, welche über einen Motor gesteuert und von einem Sensor überwacht wird. Bewegt man die Platte, steuert sie das Signal auf dem Computer in derselben Weise, wie es das konventionelle Turntable machen

[15] Max/MSP ist eine grafische Entwicklungsumgebung für Musik, Audio und Mulitmedia. Sie ermöglicht interaktive Umsetzungen, welche zumeist von Musikern und Künstlern verwirklicht werden. Das Programm ist modular aufgebaut und objektorientiert. Eigene, erweiternde Module können in C geschrieben werden.

würde. Erweitert wird das Konzept durch das sogenannte *Beatmatching Aid*, welches ermöglicht, die Umdrehungsgeschwindigkeit der Platte an die des Musikstücks anzupassen. Die Einschränkung ist, dass von einem 4/4 Takt konstanter Geschwindigkeit ausgegangen wird, aber Forschungen in Bereich der Onsetdetection arbeiten bereits an Takterkennung, welche in Zukunft dynamischere Lösungen ermöglichen sollten. Die vier Linien auf der Schallplatte stellen die 4 Grundschläge eines Taktes dar, wobei die rote für den Ersten steht. Mit zwei D'Groove Turntables könnte man die Markierungen nutzen, um einen visuellen Vergleich zu bekommen. Außerdem stellt D'Groove vier haptische Feedback Modi, welche zur Zeit per Tastatur und *Graphical User Interface* (GUI)[16] ausgewählt werden, zur Verfügung: Beim *spring*–Modus osziliert das Turntable um einen gesetzten Punkt, wie ein *plucked string*. In *bumps–for–beats* wird ein virtueller Hügel bei jedem Grundschlag erzeugt. Dies soll einerseits die Navigation durch das Stück erleichtern und andererseits beim Scratchen eine periodische Gegenkraft bei Grundschlag erzeugen. Der ähnliche Modus *resistance* erschwert die Bewegung der Schallplatte bei *High Energy*–Teilen (große Variationen in der Amplitude) eines Stücks und erleichtert sie bei niedriger Amplitude. Der letzte Modus, der *textured–record* werden mehrere kleine Sperren eingebaut, vergleichbar mit einer holprigen Straße. Verleiht man der Platte einen Schwung so schwankt diese bis zu ihrem Stillstand, dadurch entsteht ein interessantes Flimmern in der Musik. Beim Drehmoment musste ein Kompromiss zwischen Motorstärke und Reaktionszeit gemacht werden.

Der Pitchregler steuert die miteinander gekoppelten Pitch und Tempo. Der pitchbare Bereich ist nicht begrenzt, erreicht der DJ das Ende des Reglers, muss dieser per Schalter deaktiviert, zurückgesetzt und nach Aktivierung kann die Pitchmodulation in gewünschte Richtung fortgesetzt werden. Wodurch die Erweiterung des Bereichs nicht auf Kosten von Genauigkeit erbracht werden kann. Problematisch ist weniger der zusätzliche Schritt, der hierfür notwendig ist, sondern das Zurücksetzen auf die Ausgangsgeschwindigkeit, welche sich bei traditionellen Systemen in einer eingerasteten Position befindet und somit leicht auffindbar ist. Ein System mit zwei Pitchregler: einen für kleine, den anderen für große Sprünge, wurde bereits angedacht.

Der Queue–Regler soll die Direktzugriff-Funktion des Tonabnehmers übernehmen. Beide sind ein eindimensionaler linearer Vorgang mit einem visuellen Feedback. Hier wird das digitale Stück abgebildet, ganz links befindet sich der Anfang gegenüberliegend das Ende des augenblicklich gespielten Stücks, jede Bewegung der Platte wird auf der motorisierten Kontrolleinheit dargestellt. Der Regler bewegt sich also während das Stück abgespielt wird weiter und stellt somit die aktuelle Position dar. Wie bei seinem analogen Gegenstück ihn bewegen, um somit die Abspielposition zu beeinflussen. Außerdem wird bei der Interaktion mit dem Regler der *resistance*-Modus angewandt, wodurch die Suche von bestimmten Stellen besonders bei schlechten Lichtverhältnissen verbessert werden soll.

Bei der Software wurde vor allem auf eine geringe Latenz Wert gelegt, da diese für die realistische Abbildung unabdingbar ist. Bei einer Latenzzeit von unter 10 ms ist wahrnehmungsmäßig keine Verzögerungen mehr festzustellen. Im Hintergrund werden die Sensordaten und die auditive Ausgabe verwaltet. Die Audiosoftware wurde in *JASS* [Doe01b] realisiert, welche auch bei dem Prototypen zum Einsatz kommt und

[16] Engl. für Grafische Benutzeroberfläche.

in Folge noch genauer betrachtet werden soll (siehe Abschnitt 10.1.2). Des Weiteren können über eine grafische Benutzeroberfläche die verschiedenen Modi gewählt, sowie verschiedene Parameter eingestellt werden.

5.1.4 Feel the Beat

Auch Andersen in [And06] spricht sich für eine Interaktion mit haptischer Rückkoppelung aus. Der von Beamish proklamierte Queue–Regler wird verwendet, um durch ein Musikstück zu navigieren, doch Anderson geht einen Schritt weiter und experimentiert mit anderen Abbildungs- und Manipulationsmöglichkeiten. Ein zeitvariierender Parameter gibt zu jeder Zeit Auskunft über ein zu wählendes Maß, jedoch limitiert durch die Reaktionszeit und den Ausgabebereich. Es ist also wieder ein Kompromiss zwischen Latenzzeit und Stabilität zu finden.

Angefangen wurde mit Experimenten über den Schalldruckpegel (die Amplitude), aber obwohl die Abbildung auf den Regler möglich war, resultierte jede Manipulation in Stille, da wohl hierdurch sämtlich Schwingungsinformationen verloren gehen. Die Amplitudenhülle jedoch erwies sich als geeignete Stellgröße, siehe Abbildung 2 [And06]. Hier sind Änderungen hörbar. Diese Systematik soll auch auf andere Bereiche übertragen werden: In Zukunft sollen Regler über Geschwindigkeit oder Beschleunigung weitere Möglichkeiten eröffnen.

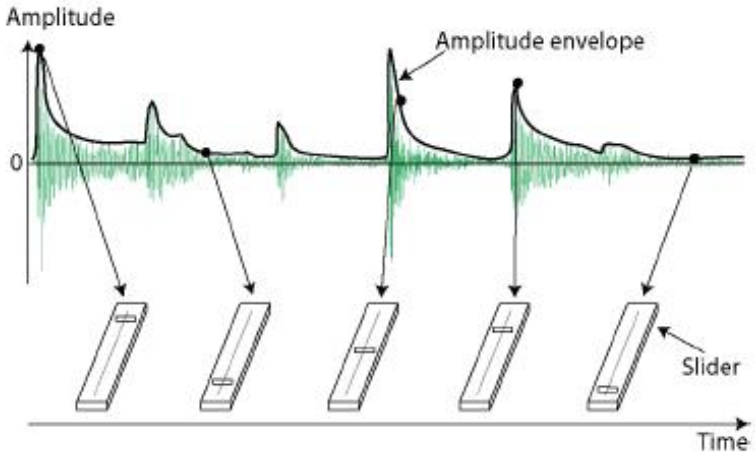
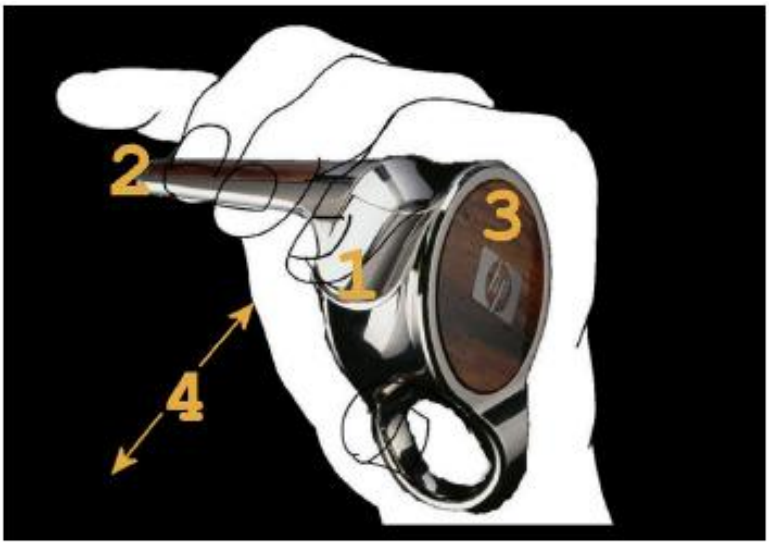


Abbildung 2: Die Amplitudenhülle und die korrespondierende Position auf dem Queue–Regler [And06]

5.1.5 DJammer

Ein ganz anderer Ansatz ist es, dem DJ mehr Bewegungsfreiheit zu geben, ohne dabei seine Möglichkeiten einzugrenzen. Der *DJammer* [Han05], ein portables, voll funktionales Gerät, welches das komplette DJ Setup vereint, wurde von *Hewlett-Packard* entwickelt. Die Basis hierfür bildet ein MP3-Player, welcher mit Sensoren, Knöpfen und drahtlosem Netzwerk ausgestattet wurde. Das Design entstand durch einen *rapid*

prototyping[17] Prozess, wobei DJs bei der Beurteilung des jeweiligen Prototypen mit eingebunden waren. Bewegungssensoren sollen Parameter wie Beschleunigung und Geschwindigkeit aufnehmen. Die Funktionsweise des DJammer ist sehr intuitiv und lässt sich einfach an Hand einer Abbildung (siehe Abbildung 3 aus [Han05]) klären. Leicht zu erkennen ist jedoch auch, dass durch dieses Konzept die Möglichkeiten der physischen Rückkoppelung entfallen.



1. Mute: Crossfading mit einem starkem Anstieg/Abfall
2. Scratch Enable: Man legt die Hand virtuell auf die Platte
3. Breakpoint: Die virtuelle Nadel auf die Platte setzen
4. Pitch Speed/Direction: schnelle/langsame Abspielmanipulation in beiden Abspielrichtungen

Abbildung 3: Die Funktionsweise des DJammer's [Han05]

5.1.6 16padjoystickcontroller

Das *Kaos Pad* und *Dr. Sample* sind die unter den Samplern[18] führenden Instrumente. Durch die weiter erhöhte Anzahl seiner Geräte ist es für den DJ nun unmöglich, diese

[17] Ein Fertigungsverfahren, welches versucht Pläne direkt und so schnell wie möglich in einen benutzbaren Prototypen umzusetzen. Es handelt sich um einen iterativen Prozess, das heißt die Ergebnisse des umgesetzten Prototypen werden analysiert und fliessen in die Entwicklung der nächsten Version des Prototypen mit ein.

[18] Sampler ist ein elektronisches Instrument, Klangerzeuger. Töne jeglicher Art können aufgenommen und jederzeit wieder abgespielt werden. Aufgenomme Samples können auch, beispielsweise in der Tonhöhe, manipuliert werden. Sie unterstützen zumeist MIDI, sind somit leicht ansteuerbar.

gleichzeitig zu bedienen. Deswegen haben manche Geräte, namentlich die Line 6 Pod Serie und RC-20, fußgesteuerte Kontrolleinheiten, wie bei dem Klavier oder der Gitarre. In diesem Fall sind jedoch alle Interaktionsmöglichkeiten über den Fuß zu regeln, wodurch ein Koordinationsproblem entsteht. Weiters ist es mit manchen Geräten nicht möglich, die Sampling–[19] und Abspielfunktionalität gleichzeitig durchzuführen. Lippit's Projekt Lupa [Lip06], welchem zwei Prototypen [Lip04] vorhergingen, entstand aus Frustation über diese Tatsache. Des Weiterem wurde die Interaktion mit dem Laptop Computer auf eine rein visuelle Basis reduziert.

Sein System, welches zusätzlich zu dem traditionellen Setup verwendet wird (siehe Abbildung 4 aus [Lip04]), besteht aus einer Kontrolleinheit, einem Pedal, sowie der dazugehörigen Software. Töne werden per Pedal gesampled und in Folge mit den verschiedene Modi des Joysticks transformiert.

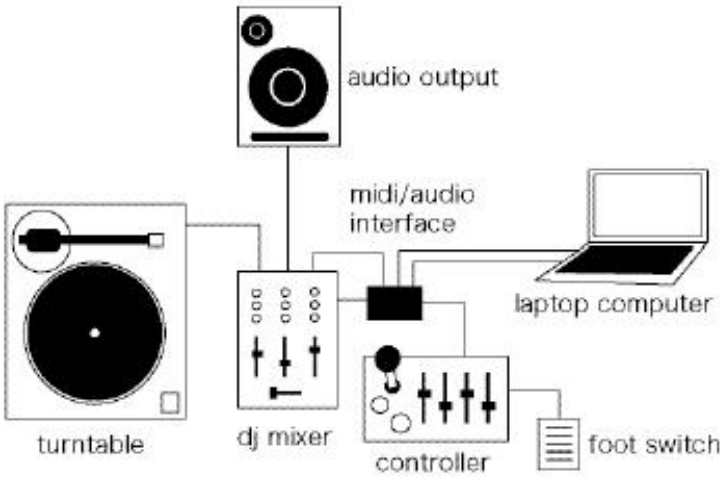


Abbildung 4: Systemübersicht 16padjoystickcontroller [Lip04]

Die Software (siehe Abbildung 5 aus [Lip06]) wurde in Max/MSP [Puc07] geschrieben verfügt über vier *Sample Banks*, welche jeweils über eine Darstellung der Wellenform, Loop-Punkt[20], Zeitfortschritt und Lautstärke verfügen.

Sie ermöglicht es nicht nur, jegliches Sample aufzuzeichnen, es erlaubt auch den physischen Input der Kontrolleinheit aufzunehmen.

Die grafische Benutzeroberfläche sollte alles das darstellen, wozu die Kontrolleinheit nicht in der Lage ist. Es wurde alles auf einer Oberfläche platziert, um einen Überblick zu verschaffen, den Kontext darzustellen und die Orientierungszeit zu verringern. Weiters wurde, obwohl technisch keine Herausforderung, auf Voreinstellungen jeglicher Art verzichtet, um den Aufführungseffekt des traditionellen Setups treu zu bleiben. Jede

[19] Musikalisch gesehen bezeichnet Sampling den Vorgang, einen Teil einer Musikaufnahme (ein Sample) in einem neuen musikalischen Kontext zu verwenden.

[20] Unter Loop, (engl. für Schleife), versteht man in der Musik ein abgegrenztes Klangelement, welches durch technische Mittel endlos wiederholt werden kann. Ein Sample muss also so geschnitten werden, dass man es ohne Brüche mehrfach hintereinander abspielen kann.

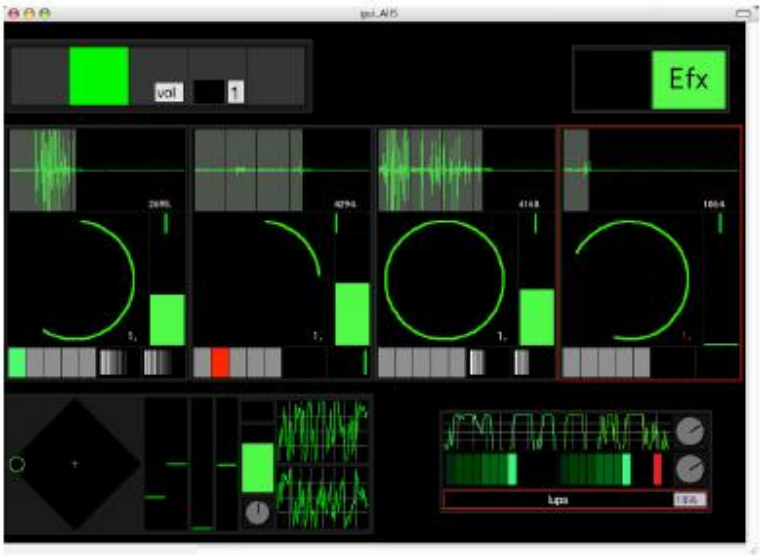


Abbildung 5: Grafische Benutzeroberfläche Lupa [Lip06]

Tonerzeugung und -manipulation muss während des Auftrittes auf der Bühne passieren. Dies lässt den Druck auf den Künstler wachsen, solange nicht konstant Änderungen vollzogen werden, kann die Struktur schnell repetitiv und langweilig werden. Abschließend kann man Lupa als eine intuitive Samplingumgebung für den Musiker bezeichnen, welche dem Publikum klare Zusammenhänge zwischen Interaktion und Tonerzeugung erkennen lässt.

5.2 Die visuelle Rückkoppelung

Durch die vermehrten Möglichkeiten des digitalen Audios entstanden immer mehr Softwarelösungen, welche zeitabhängige Daten und Parameter visualisierten. Die gängigste Funktion ist die Darstellung der Wellenform, als Metapher für die Rillen der Schallplatte. Gängige Software wie *Traktor DJ* [Nat07], siehe Abbildung 6 [Nat07], ermöglichen dem Benutzer per WIMP Interface zwar sämtliche Möglichkeiten der Tonmanipulation, jedoch reizen sie den visuellen Aspekt nicht komplett aus.

Bedingt durch mehrere Einheiten, die gesondert voneinander entwickelt wurden und viele Anzeigen und Parameter mitbringen, erhöht sich die kognitiven Last des DJs, welcher sich bei seiner Tätigkeit auf auditive Reize konzentrieren sollte. Designziele bei der Umsetzung sind laut Andersen [And02]:

1. Hinweise auf die Struktur zu geben, ohne der Notwendigkeit das Stück anzuhören. Dies kann anhand Wellenformdiagrammen, Klangfarbe oder Tonalität erbracht werden.

2. Die Möglichkeit Parameter, anhand von vergleichenden Displays, abzugleichen.

3. Die Vereinigung von real-world Parameter mit Visualisierung, um die DJ-Tätigkeit z.B. durch die Reduzierung der kognitive Last zu erleichtern.

Abbildung 6: GUI Traktor DJ Studio 3.6 [Nat07]

5.2.1 Mixxx

Mixxx kann einerseits als ein Aufführungsset anderseits als ein Mittel um Aufführungen zu analysieren gesehen werden. Diese Studien können quantitative Evaluierungen, die der Visualisierungstechniken, Kontrolleinheiten, oder Abbildung dieser, einschließen sowie solche qualitativer Natur, wie die Beurteilung der DJ Situation im Allgemeinen. Ergebnisse der Studien sollen zu neuen Einsichten im Bereich des Interfacedesigns und der Medieninteraktion führen.

Das *Open Source*[21] Projekt ist modular aufgebaut, gewährleistet Offenheit, und ermöglicht mehrere Ebenen von graphischer und physischer Interfaces. Um Anforderungen einer Aufführungspraxis gerecht zu werden emuliert es das traditionelle DJ-Setup mittels Mixer, zweier Abspielkanäle, Einbindung von MIDI Kontrolleinheiten und Parametervisualisierung. Eine Übersicht ist der Abbildung 7 [And03] zu entnehmen. Das Mischpult wurde so modifiziert, dass es anstelle von analogen Signalen MIDI Nachrichten ausgibt. Ähnlich verhält es sich bei dem Turntable Metapher, einer rotierenden Einheit, wobei die ausgegeben Signale die Geschwindigkeit repräsentieren. Dadurch wird

[21] Open Source bezeichnet quelloffene Software, welche unter einer von der Open Source Initative (OSI) anerkannten Lizenz steht. Eckpfeiler dieser sind folgende drei Merkmale: 1. Der Quelltext liegt in einer für den Menschen verständlichen Form vor. 2. Die Software darf beliebig kopiert, verbreitet und genutzt werden. 3. Die Software darf verändert und in der veränderten Form weitergegeben werden [Wik07g].

die Navigation durch das Musikstück ermöglicht, weil keine zeitkodierte Schallplatte wie bei Final Scratch [Sta07] nötig ist, entfällt die Notwendigkeit des Tonabnehmers. Jedoch entsteht auch ein anderes Feeling gegenüber dem traditionellen Turntable, wobei der Motor aktive physische Rückkoppelung liefert und der angetriebene Schallplattenteller selbst nicht unbedingt der Bewegung der Schallplatte folgen muss. Der Anspruch der Entwickler ist jedoch nicht das Turntable zu ersetzen, sondern neue Einsichten zu erlangen.

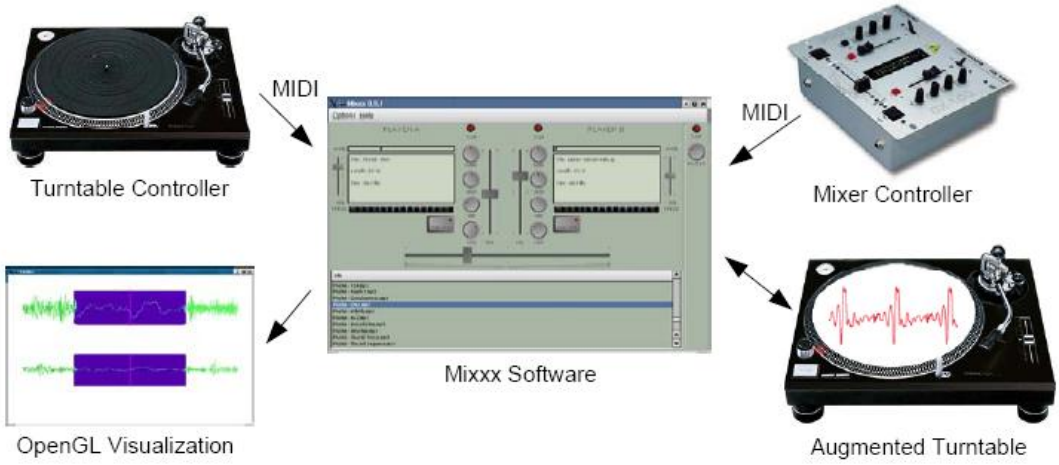


Abbildung 7: Systemübersicht Mixxx [And03]

In der *Mixxx* Software sind mehrere Interfaces wählbar, welche anhand von Vorschlägen und Beobachtungen aus Interviews mit zwei DJs, einem traditionellen und einem WIMP Interface Benutzer, erstellt wurden. Die grafische Benutzeroberfläche stellt Konfiguration und die Auswahl der Musiktitel zur Verfügung. Es vereint die Funktionen der kommerziell erhältlichen Software. Jedoch wird der visuelle Aspekt verstärkt eingesetzt, ein Ziel ist es die kognitive Last bei der DJ Tätigkeit zu minimieren. Die zwei Abspieleinheiten werden als Wellenform dargestellt, wobei diese untereinander angeordnet ist. Dies ist vor allem im Bereich des Beatmatchings von großer Hilfe, durch diese Anordnung kann Geschwindigkeit und Phase jener zwei Stücke leichter angepasst werden. Der *AudioFish* geht in den Bereich der Informationsvisualisierung, wobei die *Fisheye*-Technik genutzt wird, welche es ermöglicht Regionen nahe der Abspielposition (Fokus) vergrößert darzustellen, ohne jedoch dabei die Übersicht (Kontext) zu verlieren. Das links unten befindliche Fenster in Abbildung 7 stellt diese Technik dar. Die Darstellung kann noch durch weitere Layer, wie z.B. Onsets, Pitch und Tonfarbe, erweitert werden. Die Größe des Zooms kann angepasst werden, aber erste Tests legten eine automatische Einstellung, z.B. anhand von der Abspielgeschwindigkeit, nahe.

Das von Andersen [And03] vorgestellte *Augmented Turntable* vereint viele der dargestellten Ideen in einem Gerät. Es soll *AudioFish* und die Turntable Kontrolleinheit vereinen, und somit wieder kognitive Last vermindern. Es sollen auf dem visuell noch ungenutzten Schallplattenteller die in *FishEye* verwendeten Parameter projeziert werden. Um die kreisförmige Fläche optimal ausnutzen zu können werden die Kartesischen

Koordinaten in ein Polarkoordinatensystem übertragen. Wobei die dadurch entstandene Kreisform eine Metapher zu den Schallplattenrillen bilden. Die Reduktion der visuellen Rückkoppelung auf die Abbildung auf der Platte ist erstrebenswert, da diese Form dem Original am nächsten kommt. Die optische Sinnesüberflutung wird verhindert, der DJ kann sich somit auf seine Tätigkeiten konzentrieren. Die kreisförmige Darstellung ermöglicht eine freie Einsicht aus jedem Winkel. Weitere Verbesserungen verspricht man sich durch Sensoren an den Plattentellern, welche ein einfaches *point-and-click* System ermöglichen und somit die gesamte Kontrolle auf einem Display zusammenfassen würde. Dies würde der Vorstellung von Hansen [Han06] entsprechen, welcher sich für die Entwicklung eines einheitlichen Geräts ausspricht. Derzeit wird die Abbildung über Projektoren, die über dem Turntable angebracht sind, realisiert, diese Lösung ist nur vorübergehend. Eine Integration der Visualisierungstechnik in das Interface selbst ist hierbei erstrebenswert. Eine sich drehende, durchsichtige Schallplattenmetapher, unter welcher sich das Display befindet, ist eine mögliche Umsetzungsform.

Ausgehend von der Idee des *Augmented Turntable* wird, nach einer Vertiefung der digitalen Tontechnik, ein WIMP-Prototyp vorgestellt, welcher die Möglichkeiten und Einschränkungen einer solchen Lösung aufzeigen soll.

6 Die Digitale Tontechnik

Die Anforderungen an die Tonaufnahme sind die Transparenz und die Reproduktion der originalen Wellenform ohne Fehler. Natürlich kann weder die analoge noch die digitale Tontechnik diesen Anforderungen komplett entsprechen, jedoch heutige Aufnahmen beider Verfahren liefern durchaus brauchbare Ergebnisse.

Die analoge Aufnahmetechnik verwendet mechanische, elektronische oder magnetische Parameter, welche in selber Form variieren wie der zu erzeugende Ton den Luftdruck verändert. Die vom Mikrophon erzeugte Spannung ist die analoge Form der ständig variierenden Schallwellen. Weiters stellt die Distanz über dem analogen Medium die fortlaufende Zeit dar. Abbildung 8 aus [Sch00] stellt die Visualisierung eines einfachen, konstanten Sinustons dar.

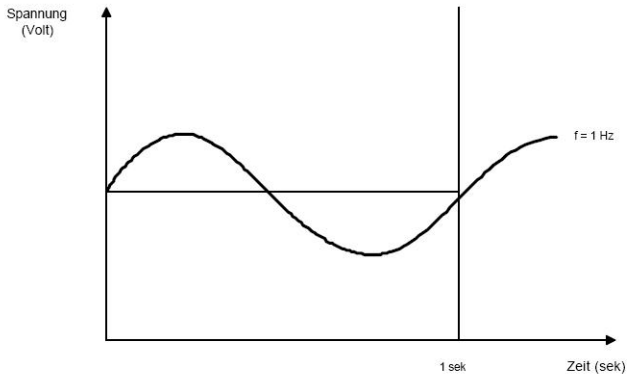


Abbildung 8: Konstanter Ton als Sinuskurve abgebildet [Sch00]

Das populärste System der digitalen Tontechnik heißt *Pulse Code Modulation* (PCM), welche ein analoges Signal binär kodiert. Diesen Prozess nennt man (im Bereich der Technik) *Sampling*[22] oder Digitalisierung (siehe 9 [Sch00]). Die Umwandlung wird mittels eines Analog/Digital–Wandlers ermöglicht, welcher sich unter anderem auf jeder gewöhnlichen Soundkarte befindet. Er ist also für die Güte der digitalen Form des Signals verantwortlich.

In regelmäßigen Zeitabständen, genannt die Samplingrate, –frequenz oder Abtastrate, wird die Auslenkung (Elongation) einer Schwingung als ein binärer Wert (mit endlicher Auflösung) repräsentiert. Es handelt sich um eine Umwandlung eines stetigen Signal in eine diskrete Darstellung. Es sind also zwei Parameter, welche die Qualität des Digitalisierungsprozess bestimmen:

1. Die Abtastrate (angegeben in Hertz oder Sekunden^{-1}) Abbildung 10 [Mar07]
2. Die Samplinggröße (in Bit) Abbildung 11 [Mar07]

Bei dem von Philips und Sony 1980 festgelegten Red-Book Standard der Compact Disk, kurz CD genannt, werden jede Sekunde 44100 Samples zu je 16 Bit ($2^{16} = 65536$

[22] Nicht zu verwechseln mit den gleichnamigen Begriff im musikalischen Kontext.

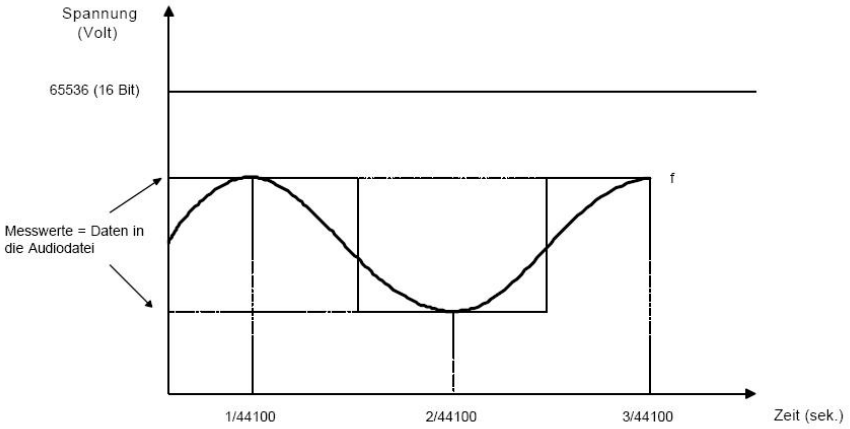


Abbildung 9: Digitalisierung eines einfachen Sinustons [Sch00]

verschiedene Werte, daraus resultieren 65536 Dynamikabstufungen) pro Kanal gespeichert. Diese Standardisierung obliegt folgender Begründung: Um die Wandlung in ein zeitdiskretes (digitales) Signal wieder rekonstruieren zu können muss das Nyquist-Shannon-Theorem beachtet werden:

> If a function $f(x)$ contains no frequencies higher than w_{max} (in radians per second), it is completly determined by giving its ordinates at a series of points spaced $T = \pi / w_{max}$ seconds apart. [Uns00]

Dies bedeutet, dass ein kontinuierliches, bandbegrenztes Signal, mit einer Minimalfrequenz von 0 Hz und einer Maximalfrequenz f_{max}, mit einer Frequenz größer als $2 * f_{max}$ abgetastet werden muss, damit man aus dem so erhaltenen zeitdiskreten Signal das Ursprungssignal ohne Informationsverlust (aber mit unendlich großem Aufwand) rekonstruieren bzw. (mit endlichem Aufwand) beliebig genau approximieren kann [Wik07f]. Dies wird im Abschnitt 7.3 noch genauer erläutert. Abbildung 12 [Dra07] soll diese Aussage visualisieren.

Durch den menschlichen Hörbereich ist eine obere Grenzfrequenz von 20 kHz gegeben, da jedoch viele Instrumente Töne oberhalb dieser Frequenz erzeugen, würden hier hörbare Verzerrungen entstehen. Diese Töne werden nun gefiltert, wobei die notwendige Dämpfung bei 22 kHz erreicht wird. Daraus ergab sich die CD übliche Samplingfrequenz von 44.1 kHz, womit das Nyquist–Theorem erfüllt ist [Sch00]. Die Auflösung lässt sich folgend erklären: die in Abbildung 9 dargestellte treppenförmige Nachbildung des Signals wird Quantisierung genannt. Die daraus entstehende Ungenauigkeit verursacht Rauschen. Dieses sogenannte Quantisierungsrauschen ist jedoch bei einer Auflösung von 16 Bit nicht mehr wahrnehmbar.

6.1 Die Wellenformdarstellung

Bei der Wellenform handelt es sich um eine grafische Repräsentation der PCM-Daten. Einzelne Werte dieser stellen die Amplitude dar zumeist normalisiert auf [1. -1], dar-

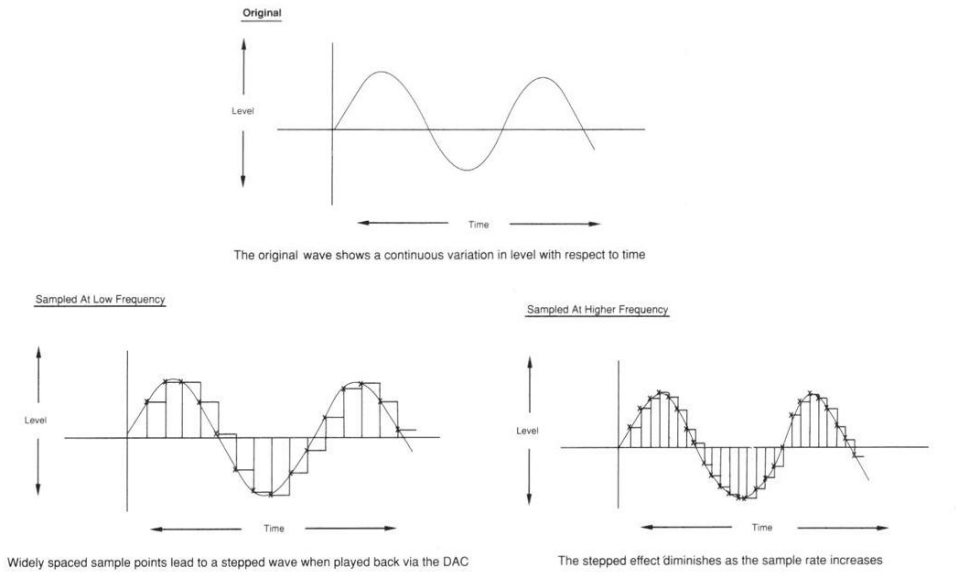


Abbildung 10: Zwei verschiedene Abtastraten [Mar07]
Die obere Grafik stellt das Ausgangssignal dar, unten befinden sich die mit zwei unterschiedlichen Raten gesampleten Daten. Die linke Grafik stellt die niedrigere Abtastrate dar, welche das ursprüngliche Signale verfälscht. Die rechte Grafik hingegen bildet das Signal durch die höhere Abtastung viel getreuer nach.

gestellt auf der Y–Achse. Als fortlaufende Daten, mit der Zeiteinheit auf der X–Achse welche sich leicht aus der Samplingrate ableiten lässt, machen sie die Schwingung ersichtlich. Es handelt sich um eine Visualisierung in der Zeitdomäne. Eine Reduktion der Daten ist erstrebenswert, da eine Unmenge an Daten (44100 Werte/Sekunde bei dem Redbook Standard) eine Übersicht erschweren bzw. die Last des Computers erhöhen würde. Dies wird anhand von lokalen Minima und Maxima innerhalb eines Ausschnitts gewährleistet, dieses Konzept wird auch bei dem Prototyp umgesetzt, um bestimmte Zeitausschnitte darzustellen (siehe Abschnitt 10.2). Somit kann eine Gesamtübersicht von einer Audiodatei dargestellt werden. Der Benutzer kann in Folge manuell Segmente vergrößert darstellen. Anhand eines Sinustones ist dieses Konzept leicht erklärbar: Der obere Teil der Abbildung 13 (erstellt mit Audacity [Aud07]) zeigt die Gesamtübersicht eines Sinustons mit 440 Hz über die komplette Dauer, wobei die Komprimierung eine Aussage über die Schwingung verhindern. Wählt man hingegen einen Teilausschnitt, wie im unteren Teil der Abbildung 13, so ist diese ersichtlich. Da es bei Musikstücken nicht um reine Schwingungen handelt, sondern um additive Schwingungsanteile, ist in diesem Falle in erster Linie die Übersicht interessant, da die repräsentierten Auslenkungen der Lautstärke entsprechen und somit eine Aussagekraft bezüglich des Aufbaus ermöglichen. Im Abschnitt 6.5.4 werden weitere Visualisierungsmöglichkeiten in der Frequenzdomäne diskutiert.

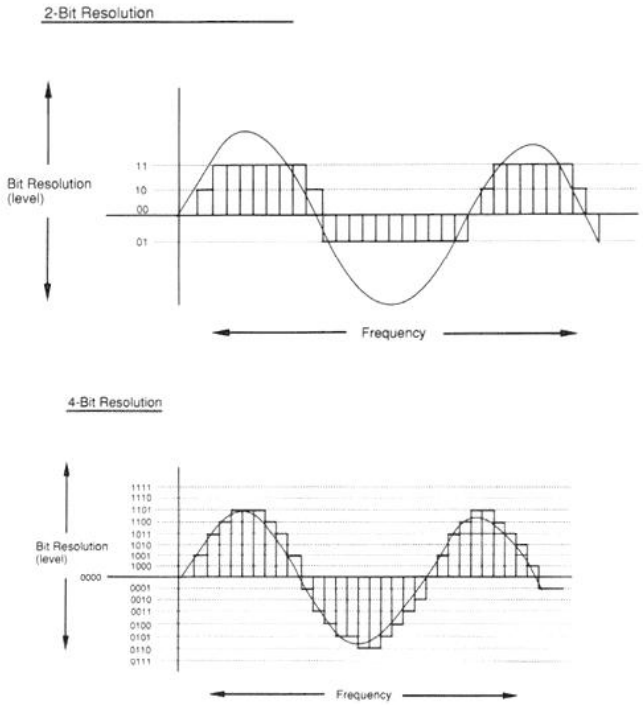


Abbildung 11: Die Samplinggröße [Mar07]

Der zweite qualititätsmaßgebende Wert ist die vordefinierte Größe wie ein Wert zur Abtastrate abgespeichert wird. Dieser ist systembedingt in Bit angegeben. Oben ist eine Abtastung in 2 Bit dargestellt. 2 Bit ermöglichen 4 unterschiedliche Werte ($2^2 = 4$). Unter wird hingegen eine Auflösung von 4 Bit ($2^4 = 16$ unterschiedliche Werte) dargestellt. Je höher die Samplingrate ist, desto höher ist die Qualität und desto mehr Dynamikabstufungen sind möglich.

6.2 Die Speicherungsmöglichkeiten eines digitalen Signals

6.2.1 Die Red Book Compact Disk

Die CD dient als Speichermedium, die Daten werden wie bei der Schallplatte spiralförmig angeordnet, jedoch von innen nach außen. Die Spiralspur hat etwa eine Länge von 6 Kilometern. Pits (Vertiefungen) und Lands (Erhöhungen) repräsentieren die Information, jedoch stellen diese die binären Werte der Daten nicht direkt dar. Will man eine Eins erzeugen wird ein Übergang von Pit zu Land oder umgekehrt gemacht (NRZ-M-Codierung). Damit die Daten zuverlässig ausgelesen werden können müssen sich zwischen Einsen, also den Übergängen, zumindest zwei und höchstens zehn Nullen befinden (d/k-Bedingung). Um dies zu gewährleisten wird die sogenannte *Eight to Fourteen Modulation* eingesetzt. Wie der Name schon erahnen lässt wird aus 8-Bit Daten (entsprechen einem Sample eines Kanales) nach einer festen Regel ein 14-Bit Wert, welcher den

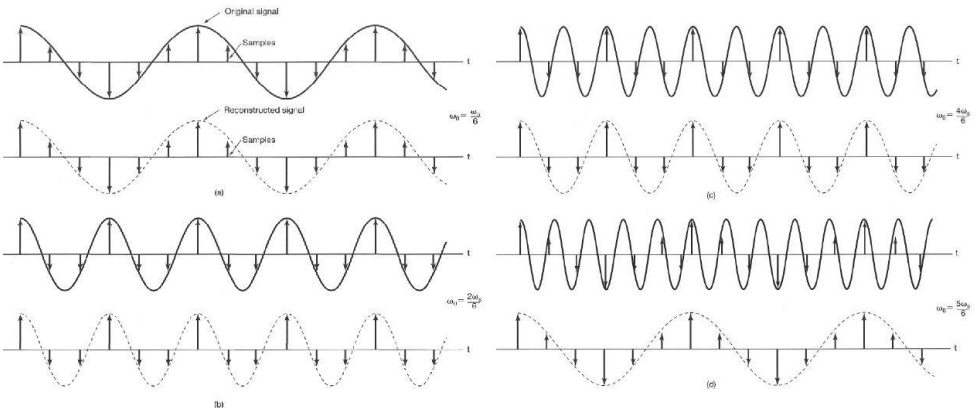


Abbildung 12: Das Nyquist Thereom visuell dargelegt [Dra07]
Hier zu sehen sind vier Signale mit jeweils unterschiedlicher Frequenz, dargestellt durch volle Linien. Diese werden mit der gleichen Abtastrate gesamplet, das rekonstruierte Signal wird durch die strichlierte Linien direkt unter den jeweiligen Ursprungssignals dargestellt. Pfeile stellen die abgetasteten Werte dar.

Anforderungen entspricht, errechnet. Damit die Bedingung auch bei aufeinanderfolgenden Codewörtern gewährleistet bleibt werden zwischen diesen drei Trennbits (000, 001, 010 oder 100) eingefügt. Die Trennbits werden nach einem Verfahren gewählt, welches eine Unterdrückung der niederfrequenten Signalanteile gewährleistet. Dies lässt Verschwendung anmuten ist aber ein ausgeklügestes System: Da die d/k-Bedingung erst bei dreimal größeren Bitzellen außer Kraft treten würde, wird selbst mit dem Overhead fast 50% an Speicherkapazität gewonnen. Auf der CD können heute bis zu 99 Minuten Tonmaterial gespeichert werden, die gängige Kapazität ist jedoch 80 Minuten. Die CD hat außerdem keine feste Winkelgeschwindigkeit; diese wird der momentanen Position des Lesekopfs angepasst, so dass die Bahngeschwindigkeit (CLV) und nicht, wie bei der Schallplatte, die Winkelgeschwindigkeit (CAV) konstant ist. Wenn der Lesekopf weiter außen auf der CD liest, wird die CD also langsamer gedreht. Auf diese Weise kann überall auf der CD mit voller Aufzeichnungsdichte gearbeitet werden, und es ist ein konstanter Datenstrom gewährleistet, wie er bei Audio-CDs benötigt wird [Wik07c].

Des Weiteren verfügt das Red–Book Format über ein Fehlerkorrektur System (CIRC).

6.2.2 Die Kontainerformate

Die CD ist natürlich nicht die einzige Möglichkeit Musik in digitaler Form zu speichern. Es existieren hierfür Containerformate[23], welche auf jedwegen Speichermedium ablegt werden können. Die wohl bekanntesten Vertreter sind *Wave* ((eigentlich RIFF WAVE, abgekürzt WAV) von Microsoft und *Audio Interchange File Format* (AIFF) von *App-*

[23] In der Computertechnik bezeichnet man als *Container* (englisch für „Behälter“) ein Dateiformat, dessen Inhalt mehrere andere Datenformate erlaubt. Typischerweise definiert ein Containerformat nur die Art und Struktur, wie der Inhalt aufzubewahren ist. Container ermöglichen so das synchrone Wiedergeben von Audio- und Videospuren [Wik07a].

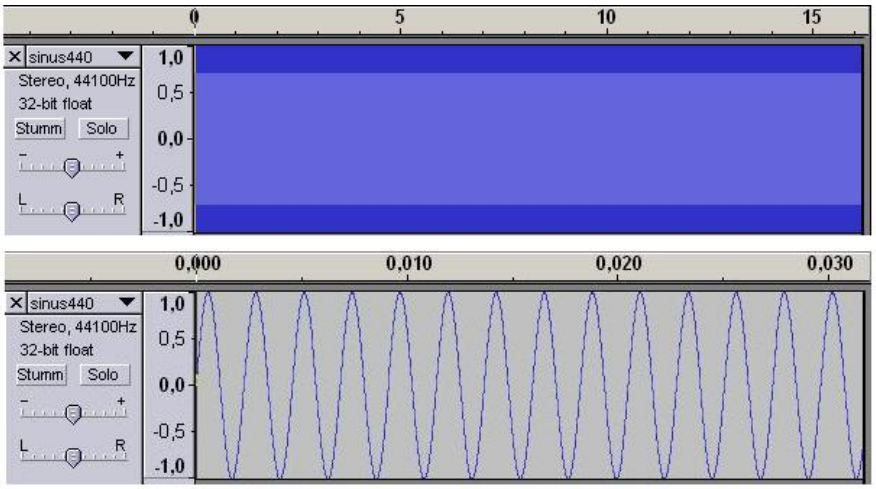


Abbildung 13: Wellenformdarstellung eines Sinustons 440 kHz [Aud07]

le, welche den Standard für deren Betriebssysteme bilden. Die Dateiendungen lauten „.wav“ respektive „.aif”.

Im folgenden Abschnitt wird anhand des Wave-Formats die Funktionsweise beschrieben. Die Datei besteht aus mehreren vordefinierten Datensequenzen, genannt *Chunks*, die größtenteils optional sind. Die WAV-Spezifikation definiert drei Chunks als verpflichtend:

1. Der *RIFF-Chunk* identifiziert die Datei als WAV-Datei.
2. Der *FORMAT-Chunk* hat einige Parameter wie z. B. die Sample-Rate gespeichert.
3. Der *DATA-Chunk* enthält die tatsächlichen Audiodaten, beispielsweise PCM-Rohdaten.

Die Datei muss mit dem *RIFF WAVE-Chunk* beginnen, bestehend aus der *groupID RIFF* und der *riffType WAVE*. Unmittelbar darauf folgt der *Format-Chunk*, der genau einmal in der Datei vorkommen muss. Dieser Chunk charakterisiert das *WAV*-Signal mit den wichtigsten Parametern. Der *Format-Chunk* setzt sich aus folgenden Elementen zusammen:

- *chunkID* (int)
- *chunkSize* (long)
- *wFormatTag* (short)
- *wChannels* (unsigned short)
- *dwSamplesPerSec* (unsigned long)
- *dwAvgBytesPerSec* (unsigned long)
- *wBlockAlign* (unsigned short)
- *wBitsPerSample* (unsigned short)

Die *chunkID* lautet bei dem *Format-Chunk* immer „fmt“ und wird von der Größe des Chunks *chunkSize* ergänzt. Ist die Variable *wFormatTag* vorhanden, beschreibt sie die Kompressionsart der Daten mit einem Wert ungleich 1. Ist dieser vorhanden werden weitere Felder im *Format-Chunk* eingefügt, nach dem *wFormatTag* mit einer Größenangabe der zusätzlichen Bytes als *unsigned short*. Danach folgt ein *Fact-Chunk*, der in einem *unsigned long* Auskunft über die Größe der unkomprimierten Daten gibt. Die verschiedenen Kompressionarten der Audiodaten machen eine vollständige Unterstützung des WAV-Formats für Entwickler noch schwieriger. Wird keine Kompression verwendet, hat *wFormatTag* den Wert 1. Die Anzahl der Kanäle werden durch *wChannels* beschrieben. Bei Monosignalen ist dieser Wert gleich 1, Stereosignalen gleich 2, usw., dabei kann eine beliebige Anzahl von Kanälen verwendet werden. Für Multikanaldaten wird jedes Sample pro Kanal zusammen in einem *Sample-Frame* zusammengefasst. Für die Anzahl der Samples pro Sekunde, steht *dwSamplesPerSec*, also die Abtastrate in *Hertz* (Hz). *dwAvgBytesPerSec* bezeichnet die durchschnittliche Anzahl von Datenbytes pro Sekunde. Sie entspricht dem Produkt aus Abtastrate und Framegröße:

$$dwAvgBytesPerSec = dwSamplesPerSec * wBlockAlign$$

Die Framegröße *wBlockAlign* setzt sich aus

$$wBlockAlign = wChannels * (wBitsPerSample/8)$$

zusammen, wobei *wBitsPerSample* die Quantisierungsrate in Bit und die Aufrundungsfunktion ist.

Die eigentlichen Audiodaten beinhaltet der *Data-Chunk*:

- *chunkID* (int)
- *chunkSize* (long)
- *waveformData* (unsigned char*)

Die *chunkID* lautet hier immer „data”, gefolgt von der Größe des Chunks *chunkSize*. Im Array *waveformData* sind anschließend die Bytes jedes Sampleframes als 8-Bit-Blöcke gespeichert. Die Amplitude wird als Serie von aufeinanderfolgenden Blöcken zu je 8 Bit im Zweierkomplement abgebildet, was den Vorteil hat, dass sie so von der CPU schneller verarbeitet und auch schneller aus dem bzw. in den Speicher gelesen bzw. geschrieben werden können. Sind Quantisierungsraten nicht durch 8 teilbar, werden die restlichen Bits bis zum nächsten durch 8 ganzzahlig teilbaren Wert mit Nullen aufgefüllt (*Zero-Padding*).

Das WAV-Format verwendet die bei Intel Prozessoren übliche *Little-Endian*–Anordnung wobei das *Least Significant Bit* (LSB) an der ersten Stelle steht, das bedeutet, dass das Byte mit den niederwertigsten Bits an der kleinsten Speicheradresse gespeichert wird[24]. Seit Mac OS X und der Einführung von Intelprozessoren in Macintosh

[24] Die gewöhnliche Darstellung von (Dezimal-)Zahlen ist – im Sinne der Leserichtung der meisten europäischen Sprachen von links nach rechts – ist also *Big Endian*. ($123 = 1 * 10^2 + 2 * 10^1 + 3 * 10^0$) Dies kommt dadurch zustande, dass die Ziffernreihenfolge der arabischen Zahlen bei den Sprachen Mitteleuropas beibehalten wurde. Im Arabischen, das sich von rechts nach links liest, werden die Zahlen gleich wie im Deutschen geschrieben, wir lesen Zahlen unter 100 – abgesehen von Ausnahmen wie 11 und 12 – auch als „*Little Endian*”: $23 = 3 + 20$ (”Dreiundzwanzig“). Bei Zahlen über Hundert beginnt man von links nach rechts zu lesen: 1200 = 1000 (+) 200 (”Eintausentzweihundert“).

Computern, sind bei „AIFF“ beide Anordnungen möglich. Desweiteren muss der Inhalt noch nach *signed*(mit Vorzeichen) und *unsigned*(ohne Vorzeichen) unterschieden werden.[Kel07]

6.3 Die Soundkarte

Die Hauptkomponenten einer Soundkarte sind der D/A-Wandler, der A/D-Wandler, beide verbunden mit ein oder mehreren Anschlüssen, der Puffer, und ein Chip. Dieser Chip verarbeitet die Befehle. In den meisten Fällen sind die Wandler nicht separat auf der Soundkarte, sondern werden deren Funktionalitäten von diesem Chip verarbeitet. Bei den Wandlern, beziehungsweise bei den Chips, ist erwähnenswert, dass diese meist multikanalfähig sind, und somit die Ausgabe beziehungsweise Aufnahme von mehereren Datenströmen zu unterschiedlichen Raten ermöglicht. Die Soundkarte verfügt über zumindest je einen analogen Ein- und Ausgang. Sie bestimmen außerdem bei welcher Auflösung und Abtastraten diese Datenströme möglich sind.

Kategorisierung kann nach Qualitätsmerkmalen und Art der Soundkarte erbracht werden. Qualitätsmerkmale werden hauptsächlich nach zwei Anforderungsprofilen festgelegt: der Heimbereich und der Studiobereich. Wobei die Qualität der Wandler, die Fähigkeit mehrere Kanäle gleichzeitig abzuspielen und aufzunehmen, die Anzahl und Möglichkeiten der Schnittstellen, und die Treiber ausschlaggebende Qualitätsmerkmale sind. In beiden Bereichen sind interne und externe Lösungen möglich. Im Heimbereich kommen immer öfters Onboardlösungen zum Einsatz, wobei auf eine gesonderte Soundkarte verzichtet und diese auf dem Mainboard integriert wird.

Auf die Bedeutung der einzelnen Komponenten und deren Qualitätsmerkmalen für den Prototypen wird im folgenden Abschnitt, Die Latenzzeit, noch detailiert eingegangen.

6.4 Exkurs: Die Latenzzeit

Der zuständige Wandler arbeitet die Daten mit einem vorgegeben Takt ab, da dies unabhängig davon geschieht ob die Samples rechtzeitig bereitgestellt werden, kann dies zu Knacksen, Knistern, Aussetzer oder einer verlangsamten Darstellung führen. Um dies zu vermeiden verfügen Soundkarten über einen Ein- und Ausgangsbuffer. Dieser wird mit Blöcken von Samples angefüllt, um in Folge abgearbeitet zu werden.

Wird ein Musikstück abgespielt geschiet dies durch einen konstanten Datenstrom von einer Datei in den internen Puffer der Soundkarte, von wo aus der D/A-Wandler diesen verarbeitet. Die Latenzzeit, welche hier die Verzögerung zwischen Abspielwunsch und Tonausgabe bezeichnet, resultiert aus der benötigten Übertragung von Medium zu der Soundkarte und der Verarbeitung des Signal mittels D/A Wandler. Es ist ferner nur eine Verzögerung bevor das erste Sample gespielt wird, ab dann sind ausreichend Daten (bei nicht zu klein gewählter Puffergröße) vorhanden um eine verzögerungsfreie Ausgabe zu gewährleisten. Dieser Verzug ist negierbar da es sich bei modernen Systemen um ungefähr 30 ms handelt.

Bei Echzeitapplikationen mit Manipulationsmöglichkeiten spielt diese Verzögerung jedoch eine große Rolle. Die Latenzzeit bezeichnet hier die Verzögerung von Auswahl und Betätigung des gewünschten Effekts (Eingabe) bis zu dessen tatsächlichen Ein-

tretens (Event). Ein Echtzeitsystem muss hierbei definierte Zeitschranken einhalten, man kann hierbei zwischen weichen und harten Anforderungen differenzieren. Bei den weichen Echtzeitanforderungen wird die Aufgabe typischerweise schnell genug abgewickelt, Verzögerungen sind jedoch tolerierbar. Bei den harten Bestimmungen wird das Überschreiten der Antwortzeit als Fehler gewertet. Bei der Entwicklung eines virtuellen Turntables müssen letztere Anforderungen erfüllt werden, da ein Verzug Fehler in der Ausgabe bewirkt.

Es ist systembedingt nicht möglich ist eine wahre Echtzeit, also eine Latenzzeit von Null, zu erzielen. Jedoch kann dieser Effekt der Wahrnehmung nach erbracht werden. Levitin [Lev99] legt hierfür eine obere Schranke von 10 Millisekunden zwischen manueller Eingabe und auditiver Ausgabe fest. Lago [Lag04] definiert diese Anforderung jedoch genauer: Seine Forschungen ergaben, dass Verzögerungen bei den meisten melodiösen Instrumenten bei 20 Millisekunden weitgehend unbemerkt bleiben, wobei aber perkussive Instrumente sensibler reagieren und erst unter 4 Millisekunden eine Verzögerungsfreiheit der Wahrnehmung nach ermöglichen.

Allein durch die Einführung einer Pufferung entsteht eine Latenzzeit. Abbildung 14 [Jui07] zeigt die Leitung einer Echtzeitapplikation, unter der Annahme, dass die Applikation einen direkten Zugriff auf die Soundkarte hat. Durch die Nutzung von Puffer werden zwei Verzögerungen eingeführt [Jui07]:

1. Record delay: Samples können nicht verarbeitet werden solange nicht ein Pufferblock befüllt ist.

2. Synchronization delay: Die Umwandlung des digitalen Signal im D/A Wandler benötigt auch Zeit (cirka 1ms). Daher startet die Wiedergabe nicht bevor der der nächste Pufferblock bereit steht.

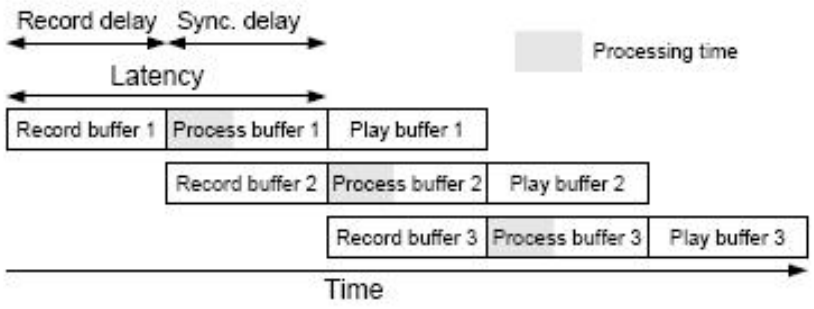


Abbildung 14: Die Verarbeitung von Audio [Jui07]

Die Latenzzeit wird also als absolutes Minimum die zweifache Puffergröße betragen. Je kleiner der Puffer also gewählt wird, desto weniger Zeit braucht die Verarbeitung dessen. Jedoch bleibt in diesem Fall weniger Freiraum für Verzögerungen und so müssen etwaige Manipulation schneller abgearbeitet werden damit die Daten rechtzeitig ankommen. Es muss hierbei beachtet werden, dass es weitere Faktoren in diesem System (siehe Abbildung 15) gibt, welche zu Verzögerungen führen.

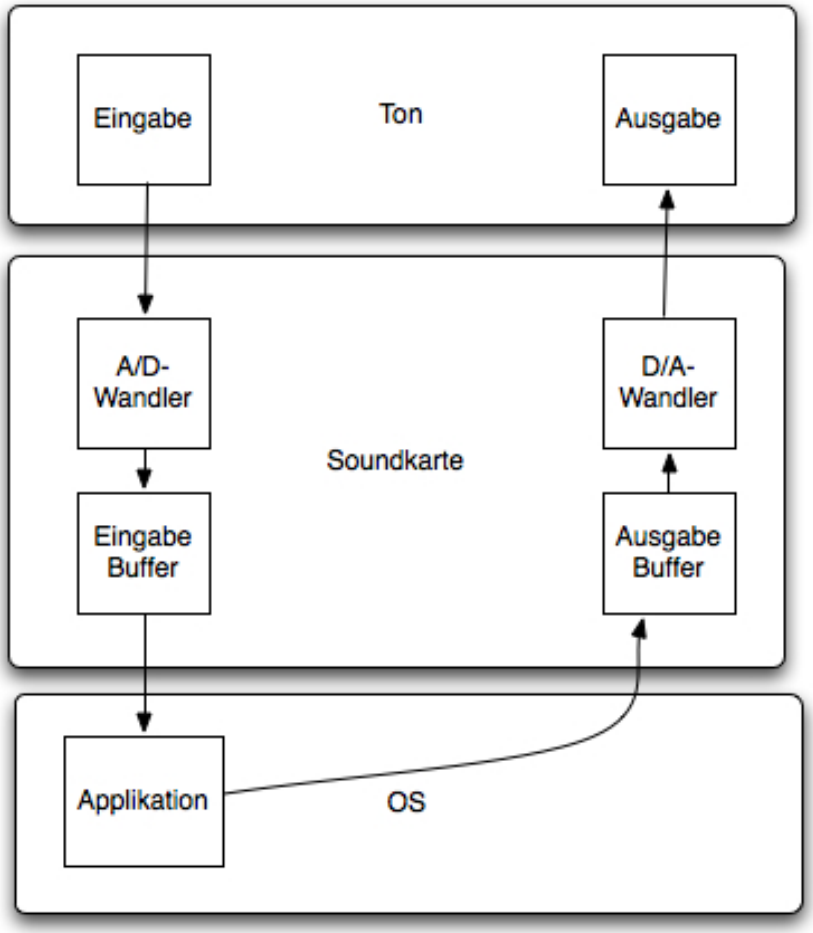


Abbildung 15: Das Zusammenspiel der Soundkarte mit Applikationen

Neben der Übertragung der Daten über den Bus und der Verarbeitungszeit von Algorithmen für die Manipulation, sind das Betriebssystem und dessen Treibermodell zu nennen. Außer dem Treiber des Betriebssystems, normalerweise erstellt von den Produzenten des Produkts, werden von unterschiedlichen Entitäten Programmierschnittstellen (kurz API für Application Programming Interfaces) zur Verfügung gestellt. Dieses definiert die Methodik wie eine Applikation auf eine Funktion zugreifen kann. Im Audiobereich ist also die Interaktion zwischen Applikation und Soundkarte gemeint. Unter *Windows* sind bekannte Vertreter: *DirectSound*, *ASIO* und *MME*. Vielerseits wird hierbei der Begriff Treiber verwendet, jedoch erfüllen sie nur einen kleinen Umfang dessen, was ein Treiber erfüllen muss. Die APIs greifen nicht direkt auf die Soundkarte zu, sondern steuern nur das Kernel Modul der Gerätetreiber an, welcher bei dem Systemstart geladen wird. Das Diagramm (Abbildung 16) soll die Interaktion verdeutlichen. Man kann also allein auf der Treiberebene von drei Faktoren sprechen die die Latenzzeit beeinträchtigen:

1. Der Gerätetreiber
2. Die Programmierschnittstellen
3. Das Framework

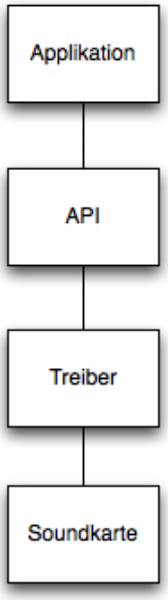


Abbildung 16: Die Interaktion mit der Soundkarte über ein API

6.4.1 Die Gerätetreiber

Gerätetreiber, oder kurz Treiber (engl. *Driver*), sind kleine Programme oder Module, die für die Übersetzung der Funktionsaufrufe des jeweiligen Betriebssystems in gerätespezifische Steuersignale zuständig sind. Dazu nutzt es Schnittstellen zum Kommunikationsbus oder anderen Kommunikationssystemen, an denen das Gerät angeschlossen ist, um Steuersignale und/oder Daten zum Gerät zu senden bzw. von ihm zu empfangen. Diese Treiber werden meist von den Hardwarefirmen selbst zu den einzelnen Geräten entwickelt und sind meist nur für dieses eine Gerät und nur für eine Art von Betriebssystem verwendbar. Allerdings bieten diese Treiber selten eine fehlerlose Unterstützung des jeweiligen Gerätes in seiner Systemumgebung, sodass oft auch verschiedene (Entwicklungs-)Versionen existieren, die entweder Bugs beseitigen oder Erweiterungen der Funktionsweise für die jeweilige Plattform bieten. Der Idealfall wären Universaltreiber eines Betriebssystems für die gesamte Hardware, wie es sie ja jetzt schon für Bildschirme, Mäuse, Drucker u.a. gibt. Diese sind allerdings nur für die Grundfunktionen verwendbar. Es liegt ja auch im Interesse der Hardwarefirmen, besondere Funktionen ihrer Geräte durch ihre eigenen Treiber zu aktivieren.

Ein Umstand, welcher aus dem Treibergewirr entstand, ist, dass nicht alle Soundkarten die in weiterer Folge erklärten APIs und Treibermodelle nativ beherrschen, sondern dies erst durch Emulation ermöglichen, wodurch eine erhöhte Latenzzeit entsteht.

6.4.2 Die Programmierschnittstellen

MME API

MME ist die Kurzform von MultiMedia Extensions und wurde von Microsoft für eine Vorgängerversion von Windows 3.1 entwickelt. Es ermöglichte den Zugriff auf jede Soundkarte, sofern ein Treiber erhältlich war. Dieselbe API mit kleinen Modifikationen wird bis heute verwendet, um Audio abzuspielen und aufzunehmen.

DirectSound API

DirectSound ist Teil der DirectX-API von Microsoft, welches nach dem Erscheinen von Windows 95 entwickelt wurde. Das Ziel war Spiele– und Multimediaentwicklern einen flexiblen und standardisierten Zugriff auf die Soundkarte zu ermöglichen. Dies sollte vor allem Entwickler dazu motivieren für das neue Betriebssystem Spiele zu erzeugen und somit ein weiteres Kaufargument für das neue Produkt zu bieten. Doch hatte DirectSound einen weiteren Vorteil: die Latenzzeiten wurden im Vergleich zu MME drastisch verringert. Jedoch verhinderte die fehlende Funktion der Aufnahme DirectSound–basierte Audiosoftware.

ASIO API

ASIO die Kurzform von Audio Streaming Input/Output wurde entwickelt von Steinberg [Ste07]. Vorgestellt wurde es zusammen mit dem Programm Cubase VST 3.5. Aufnahmefunktionalität und gleichzeitig ansprechbare Kanäle hatten Priorität bei der Entwicklung. Mit ASIO 1.0 war es zum ersten Mal möglich alle physischen Ein- und Ausgaben über eine Schnittstelle zu steuern. Ein angenehmer Nebeneffekt war das sich die CPU Last verringerte und niedrigere Latenzzeiten, vergleichbar mit denen von DirectSound, möglich waren. Mit ASIO 2.0 wurden noch Monitoring Funktionen implementiert und die Verzögerung wurde weiter verringert. Die ASIO API etablierte sich in Folge bei der Audiosoftwareentwicklung. Ein weiterer Vorteil ist, dass es nebst Windows auch Apple Macintosh Betriebssysteme unterstützt. In diesem Zusammenhang kann auch noch das Enhanced Audio Streaming Interface (EASI) erwähnt werden, welche dieselben Designziele verfolgte und dies, im Gegensatz zu ASIO, ohne Lizenzierung offen für alle Entwickler und gut dokumentiert zur Verfügung stellte. EASI konnte sich jedoch nicht durchsetzen und spätestens seit dem Verkauf an Apple und der damit verbundenen Negierung des Windowssystems wird diesem Standard keine große Zukunft mehr vorausgesagt.

6.4.3 Das Framework

WDM Framework

Windows Driver Model (WDM) ist keine API, wie fälschlicherweise vielerorts angenommen wird, es ist ein Framework für Gerätetreiber. Es soll die Kompatibilität zwischen den verschiedenen Treiber gewährleisten. Dieses Framework (und auch andere wie VXD, NT4 Kernel Mode und das für Vista entwickelte WDF) kann mit den APIs interagieren. Doch hat WDM eine zusätzliche Funktion, den sogenannten Ker-

nelMixer (KMixer). Integriert im Betriebssystem ermöglicht dieser Effektbearbeitung, Enkodierung/Dekodierung auf Kernelbasis. Jedoch ist diese Funktionalität mit einer zusätzlichen Latenzzeit von circa 20 bis 30ms verbunden. Das bedeutet bei der Verwendung von DirectSound oder MME ist die Verwendung von WDM nicht so leistungsfähig wie andere Frameworks. Die ASIO Leistung bleibt uneingeschränkt, da sie den KMixer umgeht. Bei Windows 2000 war die Anzahl von MME Geräten mit WDM Drivern auf zehn begrenzt. Da dies in der Praxis überschritten wurde, wurde dies in Windows XP korrigiert.

In Bedacht auf diese Nachteile wurde mit WDM Kernel Streaming (WDM KS) eine neue Methode vorgestellt, um auf die Soundkarte zuzugreifen. Sie obliegt der gleichen Struktur wie jeder andere WDM Treiber, nur wird bei dieser Variante der KMixer umgangen. Der Zugriff erfolgt jedoch ohne Beihilfe einer API also direkt von der Audioapplikation auf das Kernel Modul des Gerätetreibers. Die Folge waren sehr niedrige Latenzzeiten. Viele Softwareentwicklungen im Audiobereich machen sich deshalb das WDM KS Framework zu Nutze.

6.5 Die Frequenzdomäne

Allgemein kann Digital Signal Processing (DSP) in folgende Domänen unterteilt werden: die Zeitdomäne, die räumliche Domäne, die Frequenzdomäne, die Zeit–Frequenzdomäne, die Autokorrelationsdomäne und die Waveletdomäne (eine besondere Form der Frequenzdomäne). Die Entscheidung für eine Darstellungsform wird getroffen indem die Domäne gewählt wird, welche die gesuchten Charakteristiken des Tons am besten darstellen kann. Eine Sequenz von Samples produzieren eine zeitliche oder räumliche Darstellung, während die diskrete Fourier Transformation eine Frequenzdomäne, das Frequenzspektrum, erzeugt. Eine Sequenz von Frequenzspektren generiert wiederum eine Zeit–Frequenzdomäne. Diese soll im folgenden Abschnitt vertieft werden, da dies eine Grundlage für das Verständnis der weiteren Kapiteln liefert.

6.5.1 Die diskrete Fourier Transformation

Ein maßgebendes Verfahren in der Signalverarbeitung ist die diskrete Fourier Transformation (DFT), mittels welcher ein periodisches Signal von der Zeitdomäne (Zeitpunkt, Abtastwert) in die Frequenzdomäne (Frequenzanteil, Amplitude, Phase) überführt werden kann. Diskret deshalb, da es sich bei dem Samplingverfahren entstanden Werte um eine diskrete Abbildung des stetigen Signals handelt. Mit der inversen DFT (iDFT) kann aus den Frequenzanteilen wiederum das Signal in seine Ursprungsform geführt werden. Bei Musikstücken handelt es sich aber um ein nichtperiodisches Signal, auf diese Problematik wird in Abschnitt 6.5.3 eingegangen.

6.5.2 Fast Fourier Transformation

Um ein Signal von der Zeitdomäne in die Frequenzdomäne zu überführen, wird aus Performancegründen die *Fast Fourier Transformation* (FFT) angewandt.

Die FFT wurde erstmals 1965 von den Amerikanern James W. Cooley und John W. Tukey vorgestellt. Die FFT nutzt Symmetrieeigenschaften aus und reduziert die Anzahl der erforderlichen Multiplikationen durch die Auswertung zunehmend größerer

zusammengefasster Abschnitte der Eingangsfolge. Sie liefert die gleichen Ergebnisse wie die DFT bei deutlich weniger Rechenaufwand, nebenbei werden Rundungsfehler vermieden, die bei der Zwischenrechnung entstehen. Voraussetzung für die Anwendung des Algorithmus ist die Länge $N = 2^M$ und M Element aus N.

Ist N die Größe der zu berechnenden DFT, so ergibt sich für die FFT ein Rechenaufwand der proportional zur Hochzahl von N in Bezug auf die Zweierpotenz ist. Bei der normalen DFT ergibt sich im Gegensatz dazu ein Rechenaufwand proportional zu N^2. Ist z.B. $N = 2^{10} = 1024$ folgt für die FFT ein Aufwand von 10240 Operationen und für die DFT ein Aufwand von 1048567 Operationen, also ca. 100-mal soviel wie bei der FFT. [TU 07].

6.5.3 Short Time Fourier Transformation

Eine, so zu sagen, gleitende Fouriertransformation wird angewandt, um die sinusförmige Frequenz und den Phasengehalt von lokalen Bereichen über einen Zeitraum zu bestimmen. Dafür muss das Signal zunächst aufgeteilt werden. Um ein nichtperiodisches Signal, wie es bei einem Musikstück vorliegt, analysieren zu können, wird immer nur ein kleiner Bereich (ein sog. Fenster oder Window) betrachtet, in welchem das Signal als periodisch angenommen wird. In Folge bewegt sich das Fenster über das Signal, die Größe dieser Sprünge nennt man *Hop-Size*. Fenster können, und sollten, sich auch überlappen, da dies eine bessere Rücktransformation ermöglicht. Man bezeichnet diese Vorgehensweise Short Time Fourier Transformation (STFT) und ist jene welche Anwendung im Audiobereich findet.

Ein andauerndes Signal wird wie gesagt in Blöcken verarbeitet. Durch die Nichtperiodizität entsteht ein Problem: Wenn die Blocklänge nicht gerade ein Vielfaches der Periode des Signals ist, kommt es zum sogenannten Leck-Effekt (engl. Leakage Effect). Durch die entstandene abrupte Änderung der Amplitude, wird das errechnete Spektrum zu breit, es ist bildlich gesprochen „verschmiert“. Dies kann mit der Abbildung 17 [Wik07b] verdeutlicht werden:

Aus einem harmonischen[25] Signal (Grundfrequenz f_0) wird mittels Rechteck-Fenster ein Messsignal mit 5 Perioden (rot), 5.25 Perioden (grün) und 5.5 Perioden (blau) ausgeschnitten.

Wie bereits erwähnt wird für die FFT das Messsignal als periodisch angenommen, daher wird dieses periodisch wiederholt — Deshalb kann, je nach Fensterlänge, eine Unstetigkeit oder ein Sprung zwischen Ende und Anfang des Signals entstehen. Das Signal ist somit nicht mehr harmonisch. In der rechten Spalte wird das Signal samt einer Wiederholung darstellt (wobei es eigentlich unendlich viele Wiederholungen in positiver wie negativer Richtung gibt).

Das Spektrum, unterhalb abgebildet, für das harmonische Signal ist eindeutig. Bei den unharmonischen Werte ist der Leck-Effeckt deutlich ersichtlich: Das Spektrum des

[25] Harmonie bezog sich im antiken Ursprung auf Erscheinung einer Symmetrie. Neben bestimmter Regelmäßigkeit in Anordnung der einzelnen Objekte, der Sinn, also eine Wertbezogenheit, ein wichtiger Parameter der Harmonie. Der artverwandte Begriff aus der Musiktheorie, die Harmonik (von lat.–griech. *harmonia*, „Zusammenfügung, Einklang“), hingegen bezieht sich im Speziellen auf den Zusammenklang verschiedener Töne.

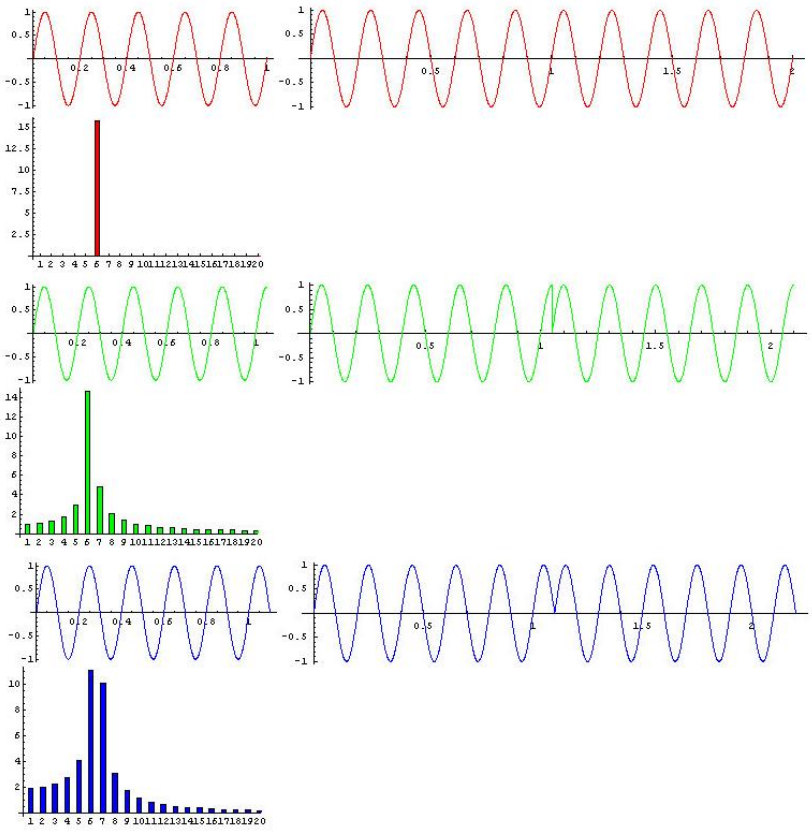


Abbildung 17: Der Leckeffekt [Wik07b]

Messsignals „verschmiert", es enthält neben der Hauptfrequenz f0 noch andere Frequenzen.

Dieser Effekt resultiert aus den Eigenschaften der Fourier-Transformation, da es eigentlich eine Multiplikation des Signales mit 0 bis zu Beginn des Zeitfensters, 1 während der Zeitfensterdauer und wieder 0 ab dem Ende des Zeitfensters darstellt (Multiplikation von Signalen führt zu Faltung im Frequenzraum). Durch die Verwendung einer geeigneten Fensterfunktion lässt sich der Effekt vermindern, aber nicht ganz vermeiden. Das Signal wird hierbei am Fensterbeginn „eingeblendet" und am Fensterende „ausgeblendet", was zu einer künstlichen Periodisierung des Signals innerhalb der Zeitfensterlänge führt. Die Fensterfunktion legt die Gewichtung fest, mit der die, bei der Abtastung eines Signalausschnitts gewonnenen Abtastwerte in nachfolgende Berechnungen eingehen. Neben der spektralen Verbreiterung werden auch die Frequenzselektivität und die maximal möglichen spektralen Fehler beeinflusst.

Es gibt verschiedene Fensterfunktionen unterschiedlicher Komplexität. Die Auswahl einer passenden Fensterfunktion, die den speziellen Anforderungen des jeweiligen Anwendungsfalls Rechnung trägt, ist daher stets ein Kompromiss.

Das Hamming Fenster ist hierbei eine weit verbreitete Methode. Es rundet das Signal

folgendermaßen:

$$w(m) = 0.54 - 0.46cos(\frac{2\pi m}{N})$$

wobei N die Fensterbreite und m der aktuelle Wert des Eingangssignals ist. Woraus eine wie in Abbildung 18 [Dir07] dargestellt, resultierende Gewichtung entsteht.

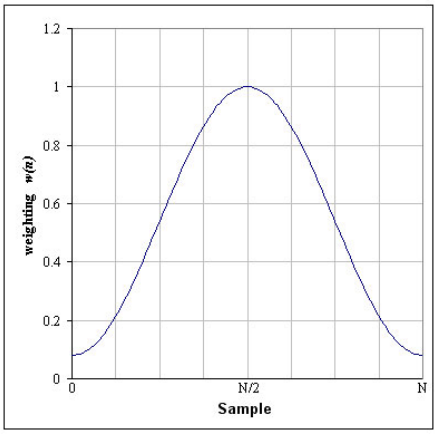


Abbildung 18: Die Gewichtungsfunktion des Hamming Fensters [Dir07]

Die Zeit- und Frequenz-Auflösung einer gleitenden DFT können aber nicht unabhängig voneinander gewählt werden, da die Wahl der Fenstergröße und -funktion bei der STFT statisch ist. Daher liefert ein breites Fenster eine hohe Frequenzauflösung bei schlechter Zeitauflösung, dies verhält sich bei einem schmalen Fenster genau umgekehrt. Dieses Phänomen zwischen Zeit- und Frequenzauflösung nennt man Unschärferelation (nach Werner Heisenberg). Vereinfacht wird dies anhand einer Formeln dargestellt:

$$Zeitfensterbreite * Frequenzauflösung = 1$$

Die relativ neue Wavelet-Transformation vermeidet dies durch eine signalangepasste Auflösung. Die Fensterbreite wird adaptiv ermittelt, sie orientiert sich an gleich bleibender Schwingungszahl pro Fenster. Auf diese Technik die zweifelsohne eine bedeutsame Rolle im Audiobereich spielt, wird nichtsdestotrotz nicht weiter eingegangen. Die anschließend vorgestellten Techniken basieren auf dem Prinzip der DFT.

Abschließend werden die Begriffe nochmals abgegrenzt, um Verwechselungen auszuschließen. Von der Methodik der Transformation der Fourier Analyse, lässt sich die DFT ableiten, welche die Transformation von diskreten Daten gewährleistet. Die FFT ist eine DFT, wobei diese bei bestimmten Voraussetzungen den Rechenaufwand minimiert. Eine STFT bezeichnet nur das Verfahren einer gleitenden DFT, meist einer FFT.

6.5.4 Die visuelle Rückkoppelung

Als ein Ergebnis einer gleitenden DFT, beispielsweise die STFT, kann ein Zeit-Frequenz-Spektrogramm gewonnen werden. Sie ist eine Visualisierung der durch die Transformation gewonnenen Daten. Es handelt sich um eine Zeit-Frequenz-Raumrepräsentation,

wobei die horizontale Achse die Zeit, die vertikale Achse die Frequenzbänder und die Intensität der resultierenden Punkte die Amplitude darstellt. Zumeist kommt diese 2–dimensionale Darstellung zum Einsatz, wobei die Farbintensität als Maß für die Amplitude verwendet wird (siehe Abbildung 19 [Wik07i]).

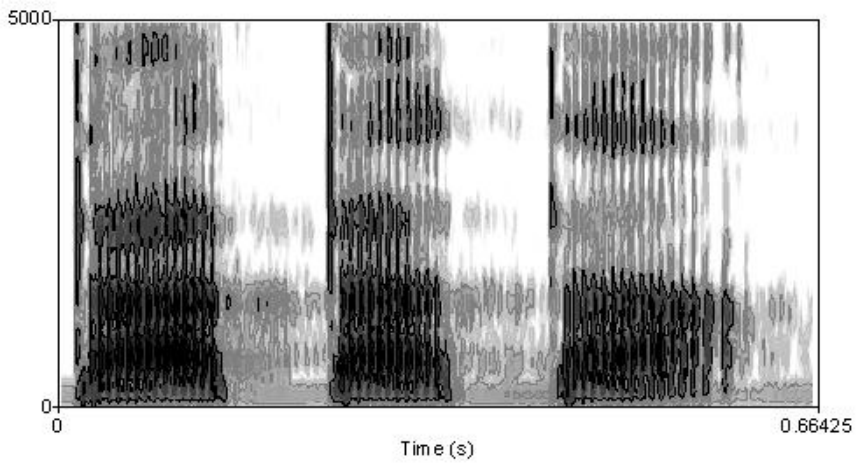


Abbildung 19: Ein Spektrogramm von einer männlichen Stimme „tatata“ sagend [Wik07i]

Im Audiobereich ist auch die Wasserfalldarstellung verbreitet, wobei ein 3–dimensionaler Raum verwendet wird und die Amplitude auf der z-Achse abgebildet wird (siehe Abbildung 20 [Ana07]).

Folgendes Beispiel mit Abbildung 21 [FH-07] soll die Unterschiede der Frequenzdomäne und der Zeit-Frequenzdomäne verdeutlichen:

Ausgegangen wird von zwei Signalen, einem periodischen und einem nichtperiodischen. In Folge werden eine reine Fourier-Analyse und eine gleitende Fourier-Analyse erstellt. Bei der in der Mitte befindlichen Fourier-Analyse ändert sich wenig, das verdeutlicht dass beide Signale die gleichen Frequenzen enthalten. Die ähnlichen Spektrogramme rühren daher, dass nicht unterschieden werden kann wann die Signale eintreten. Bei den Spektrogrammen ganz unten ist ein zeitlicher Bezug gegeben. Bei den periodischen Signal bleibt die Visualisierung konstant, bei dem rechten jedoch ist der Eintritt der Änderung genau erkennbar.

Die Frequenz- und Amplitudenachse kann entweder linear oder logarithmisch gewählt werden. Audio wird zumeist in der Amplitudenachse logarithmisch (z.B. in dB) dargestellt und Frequenzen können linear, für eine harmonische Analyse, oder logarithmisch, für die musikalische, tonale Analyse, sein. Im Audiobereich ist eine Darstellung der Frequenzbänder in Bereich von 20 Hz - 20 kHz (Menschliche Hörschwellen, siehe Kapitel 8) sinnvoll.

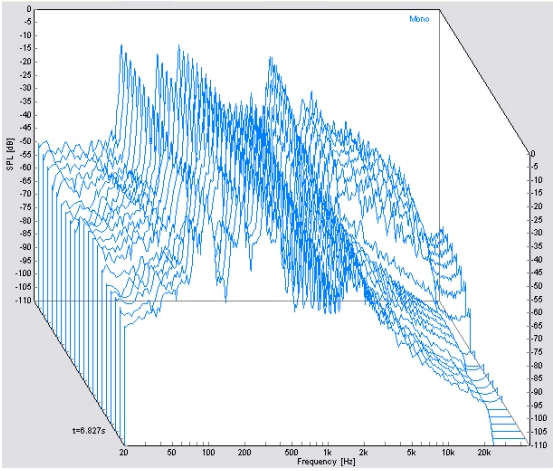


Abbildung 20: Ein Wasserfall–Spektrogramm [Ana07]

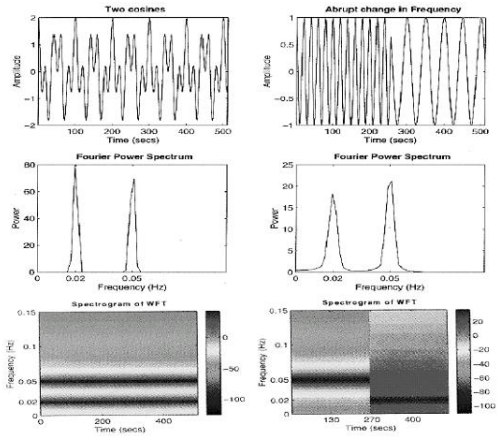


Abbildung 21: Unterschiede der Frequenzdomäne und der Zeit-Frequenzdomäne [FH-07]

7 Die Abspielgeschwindigkeitsmanipulation

7.1 Die unterschiedlichen Ansätze der Manipulation

Die Manipulation der Abspielgeschwindigkeit, also die Dehnung oder Stauchung des Signals, spielt im DJ–Bereich eine wichtige Rolle. Bei dem Turntable kann dies durch einfache mechanische Manipulation der Umdrehungsgeschwindigkeit erreicht werden. Im digitalen Bereich würde dies der Veränderung der Abtastrate entsprechen. Da jedoch die Soundkarte nur bestimmte Abtastraten beherrscht, beziehungsweise die Umstellung zwischen diesen nicht zur Echtzeit möglich ist, ist eine feingranulare Manipulation zur Laufzeit nicht möglich.

Der gesuchte Ansatz ist der des *Resamplings*. Bei einer fixen Abtastrate muss für eine Manipulation flexibelen Umfangs ein fortlaufendes Signal aus den diskreten Daten approximiert werden. Diesen Schritt nennt man Interpolation. Einfach gesagt müssen N Zwischenwerte errechnet werden, um das Signal bei gleicher Abtastrate in die Länge ziehen zu könnnen. Zu gegebenen diskreten Daten (das PCM Signal) soll eine kontinuierliche Funktion (die sogenannte Interpolante oder Interpolierende) gefunden werden, welche die gesuchten Daten abbilden. Für die Evaluierung der Ergebnisse muss zunächst zwischen zwei Arten des Resamplings unterscheiden: das *Upsampling* (die Berechnung des Signals von der ursprünglichen in eine höhere Frequenzrate) und das *Downsampling* (vice versa).

Im Idealfall bleibt bei dem Upsampling–Prozess das Spektrum des Ausgangsignals unverändert, ohne Amplitudenanteil in den Frequenzbereich zwischen der ursprünglich höchsten Frequenz ($Samplingrate_{orig}/2$) und der neuen höchsten Frequenz ($Samplingrate_{neu}/2$). Dies ist gemäß dem Nyquist–Theorem (siehe Abschnitt 6). Bei dem normalen Downsampling–Verfahren hingegen wird der Amplitudenanteil im Bereich von ($Samplingrate_{neu}/2$) bis ($Samplingrate_{orig}/2$) entfernt, somit entsteht ein neues Signal. Im Falle des des Upsamplings ergeben sich also folgende Qualitätsmerkmale [Wil07]:

1. Niedriger Amplitudenanteil in den Frequenzen über ($Samplingrate_{orig}/2$)
2. Hoher beibehaltener Anteil des Ausgangsignals

7.2 Die lineare Interpolation und das Drop–Sample–Verfahren

Die wohl bekannteste Methode ist die lineare Interpolation, wobei die fortlaufende Funktion als lineare Punkt zu Punkt Verbindung zwischen den fortlaufenden Daten angenommen wird. Ein ähnlicher Ansatz ist das *Drop–Sample–Verfahren*, hier wird eine horizontale Verbindung der einzelnen Daten angenommen. Keines der beiden Verfahren ist für Resampling hoher Qualität geeignet: Durch die schlechte Annäherung wird das Ausgangssignal verfremdet und ein hoher Amplitudenanteil in den Frequenzen über ($Samplingrate_{orig}/2$) entsteht.

7.3 Die bandbreitenbeschränkte Interpolaton

Um optimale Ergebnisse zu erzielen wird die bandbreitenbeschränkte Interpolation angewandt. Sie basiert auf der Annahme, dass unter Einhaltung des Nyquist Theorems,

die eindeutige und exakte Reproduktion der Zwischenwerte mit unendlichem Aufwand möglich ist. Bei diesem Verfahren wird das Signal mit einem ganzzahligen Faktor L interpoliert und dann mit einem Faktor M dezimiert. Dadurch kann jede Samplingratenkonvertierung mit einer Rate von L/M vollbracht werden. Das Interpolationsverfahren spielt hierbei eine untergeordnete Rolle, da die entstandenen Interpolationsfehler mittels *Lowpass*–Filterung, welche von $max\{L, M\}$ abhängt, eliminiert werden. Daher wird ein möglichst unaufwändiges Verfahren gewählt, wie das des *Drop–Samples*. Ein idealer Lowpass–Filter ist jedoch mit unendlich großem Aufwand verbunden. Hierfür muss das Signal mit einer Rechteckfunktion in der Frequenzdomäne multipliziert, oder eine Konvolution mit einer Sinc–Funktion in der Zeitdomäne vorgenommen werden. Da jedoch die durch FFT die Frequenzdomäne nur in einer endlicher Auflösung vorliegt und die Konvolution mit einer Sinc–Funktion ebenfalls eine unendliche Auflösung in der Zeitdomäne benötigt, ist der ideale Lowpass–Filter also nur von rein theoretischer Natur. Um eine endliche Lösung zu ermöglichen muss das Signal gefenstert werden.

7.4 Die polynominale Interpolation

Eine anderer Ansatz ist das Interpolationsverfahren zu optimieren, wobei dies eine anschliessende Filterung nicht ausschließt. Die oben erwähnten Verfahren, die lineare Interpolation und das *Drop–Sample*–Verfahren fallen unter den Begriff stückweise Interpolation durch Polynome. Die gewünschte Qualität kann mittels Interpolation durch ein Polynom höheren Grades erreicht werden. Neben dem Grad ist die Anzahl der benachbarten Samples, welche benötigt werden um einen Wert zu interpolieren, ein weiterer Faktor. Die Abhängigkeit Interpolationsaufwand und Interpolationsfehler ist eine lineare Funktion mit einer Versetzung [Nie01]. Leider ist diese leicht abfallend, so dass auch der Grad des Polynoms bis ins unendliche gesteigert werden müsste, ähnlich wie bei der bandbreitenbeschränkten Interpolation, um eine transparente Qualität zu erreichen. Jedoch ist dies bei der begrenzten Sinneswahrnehmung des Menschen gar nicht notwendig. Verfahren die speziell für den Audiobereich entwickelt wurden ermöglichen einen guten Kompromis zwischen Komplexität, Zeitaufwand und Qualität. Die 4-Punkt Interpolation vierten Grades aus [Nie01], welche im Prototypen verwendet wird, benötigt 28 Operationen (14 Multiplikationen und 14 Additionen, Subtraktionen) für die Approximation eines Wertes. Die Funktion ist in Abbildung 22 [Nie01] ersichtlich. Ein Nachteil dieses Verfahrens ist jedoch, dass diese auf einen bestimmten Faktor optimiert sind. Eine Möglichkeit um dies zu vermeiden, wäre eine Tabelle mit der jeweils optimierten Interpolationsfunktion zu verwenden.

7.5 Die richtige Interpolationsmehtode

Während die lineare Interpolation nur mit einem sehr geringen Rechenaufwand verbunden und die Implementation leicht ausfällt, ist die Qualität höchstens akzeptabel. Deshalb eignet sich diese für Echtzeitanwendungen auf langsamen Rechnern. Polynomiale Interpolation bietet hingegen sehr gute Ergebnisse bei nicht all zu großem Aufwand. Wenn die Abtastrate reduziert werden soll (Downsampling) ist diese auf jeden Fall empfehlenswert. Bei Upsampling ist diese in den meisten Fällen ebenfalls aureichend. Bei höchsten Qualitätsanspruchen, wobei der Berechnungsaufwand eine untergeordne-

$$f(x) = \begin{cases} 0.04252164479749607x^4 + 0.29412762852131868x^3 - 0.75864870418278074x^2 + 0.04442540676862300x + 0.58448510036125145 & 0 \le x < 1 \\ -0.04289144034653719x^4 + 0.13781898240764315x^3 + 0.21256821036268256x^2 - 1.16581445347275190x + 1.06598379704160570 & 1 \le x < 2 \\ 0 & 2 \le x \\ f(-x) & \text{otherwise.} \end{cases}$$

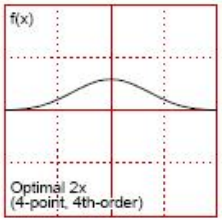


Abbildung 22: Formel, Figur: 4-Punkt Interpolation 4. Grades [Nie01]

te Rolle spielt, sollte der bandbreitenbeschränkten Interpolation den Vorzug erhalten [Pit07].

7.6 Die erweiterten Möglichkeiten durch die Digitaltechnik

Bei Resampling und gleichbleibender Abtastfrequenz werden die Frequenzen abhängig von der Abspielgeschwindigkeit transponiert. Verlangsamen der Aufnahme wirkt sich somit in einer Dehnung der Schwingungen ergo in einer tieferen Tonhöhe aus und vice versa. Dies entspricht auch der Geschwindigkeitsmanipulation bei der Schallplatte, jedoch ermöglicht es die Digitaltechnik die Zeitmanipulation von der Tonhöhe zu trennen. Durch Kopplung von DFT und iDFT kann ein Signal im Frequenzbereich manipuliert werden. Die Manipulation findet in drei Phasen statt:

1. Analyse aufgrund der von der DFT gelieferten spektralen Struktur
2. Datenverarbeitung in der Frequenzdomäne
3. Resynthese, die Rückführung in die Zeitdomäne

Dies kann für den DJ von Bedeutung sein, da er die Geschwindigkeitsmanipulation bei Beatmatching bzw. die Tonhöhenmanipulation beim Stimmen für das Scratching getrennt von einander benötigt, jedoch den jeweils anderen Effekt in Kauf nehmen muss.

Die Technik welche reine Zeitmanipulation ermöglicht heißt Time Stretching und die Reziproke, der Prozess die Tonhöhe zu verändern ohne die Abspielgeschwindigkeit zu beeinflussen, Pitch Shifting. Für diese Methoden gibt es unterschiedliche Techniken. Typischerweise, kann bei guter Umsetzung, die Tonhöhe um fünf Halbtöne gehoben und die Länge um 130% in adäquater Qualität variiert werden. Dies hängt jedoch auch vom Tonmaterial ab, beispielsweise kann eine Aufnahme von einem Soloinstrument um 200% und um eine Oktave manipuliert werden ohne dabei hörbare Verluste in der Qualität zu verursachen [Ber07].

8 Exkurs: Grundlagen der Psychoakustik

Die Psychoakustik ist ein Teilgebiet der Psychophysik und stellt eine wissenschaftliche Disziplin dar, die den Zusammenhang der physikalischen Eigenschaften eines Schallsignales mit den Hörempfindungen des menschlichen Sinnesorganes untersucht. Es wird verlangt, dass die Auswirkungen der physikalischen Reizen auf die sinnespsychologischen Reaktionen durch Messungen eindeutig erfasst werden können und die Zusammenhänge durch allgemeine Funktionen (Kennlinien) eindeutig beschrieben werden können.

Die Anfänge der Psychoakustik gehen auf Untersuchungen des deutschen Physikers Heinrich Georg Barkhausen zurück, der in den 20er Jahren des letzten Jahrhunderts die Lautstärkeeinheit Phon und das Bark einführte. 1935 entdeckte Stanley Smith Stevens, dass die wahrgenommene Tonhöhe auch von der Schallintensität abhängt. Eberhard Zwicker postulierte zur physikalischen Größe Intensität die Lautheit als Empfindungsgröße.

8.1 Die psychoakustischen Parameter

Die wichtigsten psychoakustischen Parameter sind die Lautheit (Einheit sone), Schärfe (Einheit acum), Tonhöhe (Einheit mel), Rauigkeit (Einheit asper) und Schwankungsstärke (Einheit vacil). Daneben sind Tonhaltigkeit und Impulshaltigkeit bedeutsame Größen; sie werden auch bei der Bildung von Beurteilungspegeln herangezogen [Zwi82].

Da kein Gerät die Wahrnehmung von Signalen messen kann, basiert diese Wissenschaft auf professionellen Hörtests. Insbesondere baut sie auf zwei Schwächen unserer Wahrnehmung auf: die Hörschwelle des Menschen und die Maskierung von Signalen.

8.2 Die Hörschwelle des menschlichen Gehörs

Das menschliche Gehör kann je nach Alter und Person Töne im Bereich von ungefähr 20 Hz bis 20 KHz (je nach Quelle bis 16 KHz) wahrnehmen, wobei für jede Frequenz eine bestimmte Mindestlautstärke nötig ist (siehe Abbildung 23 [Blu]). Die Frequenzen zwischen 2 und 4 KHz, in deren Bereich die menschliche Sprache liegt, spielen dabei eine besondere Rolle.

8.3 Die Maskierung von Signalen

Die Beeinflussung der Hörbarkeit eines Schalles durch die Überlagerung eines oder mehrerer Störschalle bezeichnet man als Maskierungs- oder Verdeckungseffekt.

Ein auf das Gehör wirkender Reiz setzt gleichzeitig die Empfindlichkeit für andere Reize herab. Einfaches Beispiel: Ein Gespräch ist in ruhiger Umgebung leicht zu führen, ohne dass die akustische Verständlichkeit darunter leidet. Wirkt nun während des Gesprächs ein Störschall als Maskierer ein, kann dieser den Sprachschall überdecken (maskieren) so dass das Gespräch gar nicht mehr oder nur mit lauterer Stimme weitergeführt werden kann. Folglich werden zwei Arten der Maskierung vorgestellt: die simultane und die zeitliche Maskierung. Für eine detaillierte und weiter gegliederte Definition ist das Werk von Zwicker [Zwi82] heranzuziehen.

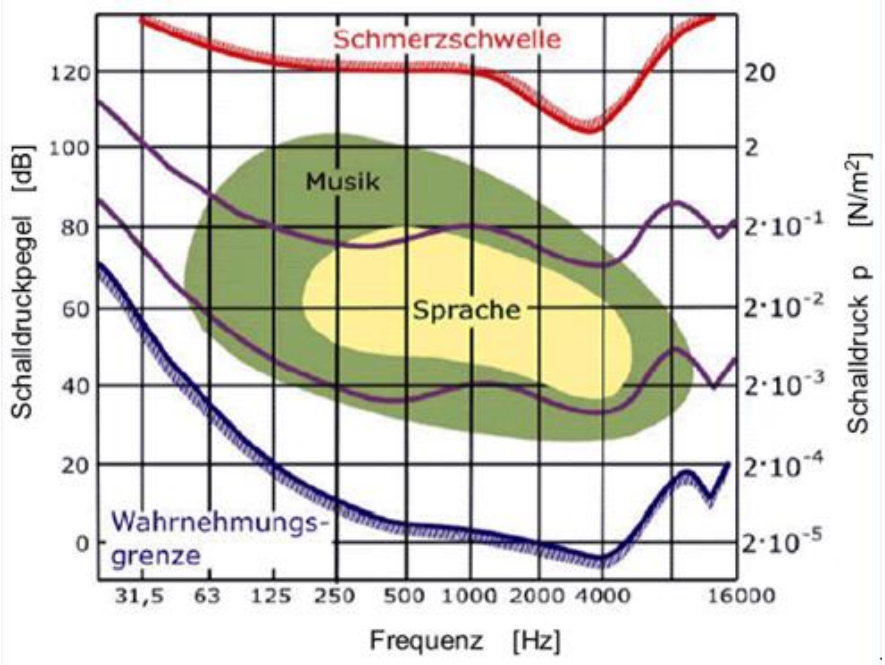


Abbildung 23: Darstellung des Frequenz– und Lautstärkenumfangs (die Hörflächen) [Blu]

8.3.1 Die simultane Maskierung

Ein Maskierungston von 1 kHz und einer Lautstärke von 60 dB verändert die Hörschwelle drastisch, Töne ähnlicher Frequenz müssen nun wesentlich lauter werden, um ebenfalls wahrgenommen werden zu können. Das Gehör kann zwar die Frequenzen unterscheiden, aber die Rezeptoren im Innenohr beeinflussen sich gegenseitig.

Wenn neben einem Signal von 1.000 Hz und 0 db ein Signal von 1.100 Hz und -10 db besteht, wird das zweite Signal vom durchschnittlichem Gehör nicht erkannt. Dies liegt einerseits daran, dass das zweite Signal leiser ist, aber ausschlaggebend ist, dass sie sich ähnlich sind. Wenn man nun die Frequenz des leiseren Signals langsam ändert, wird dieses bei gleich bleibender Lautstärke ab einem gewissen Zeitpunkt hörbar. Die Abbildung 24 [Hac00] visualisiert dieses Phänomen.

8.3.2 Die zeitliche Maskierung

Das menschliche Gehör benötigt, besonders nach lauten Signalen, eine gewisse Regenerationszeit bis es wieder aufnahmefähig für einen folgenden leiseren Ton ist, da die Basilarmembran im Ohr Zeit braucht, um in ihre Ruheposition zu kommen. Die Dauer dieser Aussetzer bestimmt das Lautstärke-/Frequenzverhältnis beider Töne, es kommt zu Verdeckungen bis zu 200 ms. Dieser Effekt wird Nachverdeckung genannt. Interessanterweise kommt es aber auch vor einem sehr lauten Signal zur Maskierung, hier wirkt diese schon 20-50 ms bevor der Schall überhaupt auftritt. Dies wird als Vorverdeckung bezeichnet (Abbildung25 [Zwi82]).

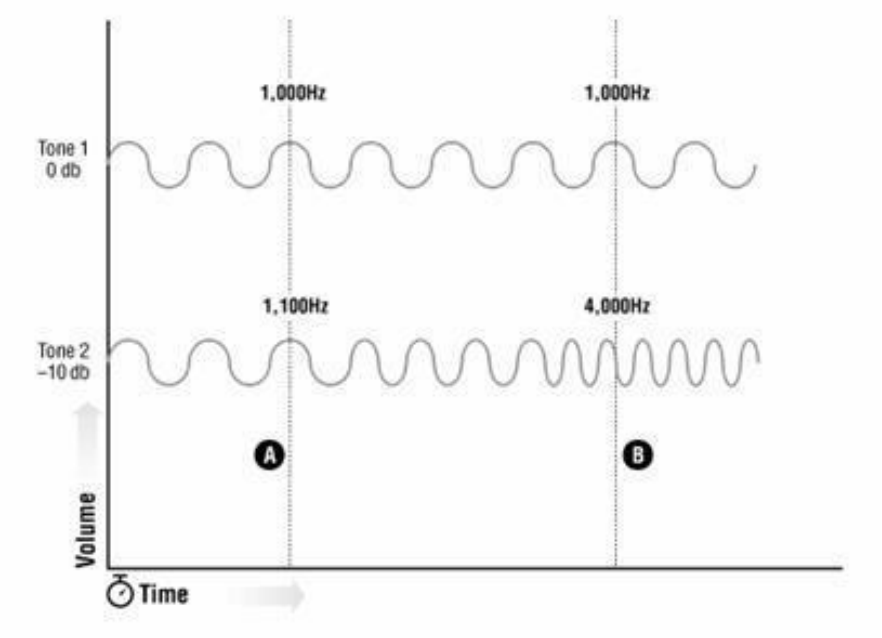


Abbildung 24: Ein Beispiel der simultanen Maskierung [Hac00]
Bei diesem Beispiel wäre zum Zeitpunkt A nur Ton 1 wahrnehmbar, während zum Zeitpunkt B der 2. Ton, bei gleich bleibender Lautstärke, nicht mehr überhörbar ist.

Die genaue Dauer der Vor- und Nachverdeckung hängt von der Stärke und Dauer des Maskierers ab.

8.4 Die Kritischen Bänder

Wahrnehmungsexperimente zeigen, dass das Gehör offensichtlich in eng begrenzten Frequenzbereichen Intensitäten (und damit *Lautstärken*) von verschiedenen Schallreizen zusammenfasst. Diese Frequenzbereiche werden als Frequenzbänder (engl. critical bands) bezeichnet. Aufgrund von empirischen Forschungen wurden (je nach Autor) 24-27 kritische Frequenzbänder definiert, innerhalb derer sich der Maskierungseffekt besonders auswirkt.

Mit zunehmender Frequenz des Maskierungstons wird der maskierte Frequenzbereich breiter. Die Breite dieser sogenannten *kritischen Frequenzbänder* liegt zwischen unter 100 Hz für die tiefsten wahrnehmbaren Töne und über 4 kHz für die höchsten. Der Grad der Maskierung einer bestimmten Frequenz ist lediglich abhängig von der Signalintensität im kritischen Band dieser Frequenz.

Die Unterteilung des hörbaren Frequenzspektrums in kritische Frequenzbänder, die von Heinrich Barkhausen vorgenommen wurde, spiegelt das Auflösungsverhalten des menschlichen Ohres wider (Abbildung 26 [Zwi82]).

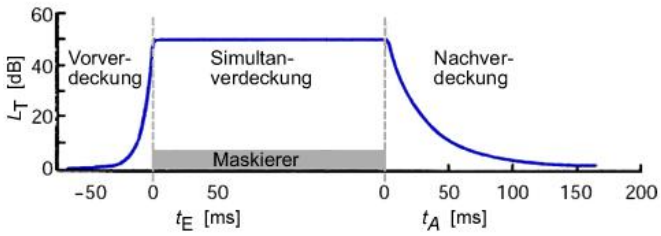


Abbildung 25: Die zeitliche Maskierung [Zwi82]

Band Nr	Frequenz (Hz)	Band Nr	Frequenz (Hz)
0	50	14	1970
1	95	15	2340
2	140	16	2720
3	235	17	3280
4	330	18	3840
5	420	19	4690
6	560	20	5440
7	660	21	6375
8	800	22	7690
9	940	23	9375
10	1125	24	11625
11	1265	25	15375
12	1500	26	20250
13	1735		

Abbildung 26: Obere Frequenzen der kritischen Frequenzbänder [Zwi82]

8.5 Das Weber-Fechner'sche Gesetz

Die subjektive Empfindung der Stärke eines physikalischen Reizes ist proportional zum Logarithmus seiner Maßzahl.

$$dB = Phon : x = 10log(I/I_0); I_0 = Hörschwelle$$

8.6 Die Lautstärke, Lautheit nach Barkhausen

Anhand seiner Forschung hat Barkhausen um 1920 herausgefunden, dass die Schallstärke über die Empfindung der Lautstärke oder Lautheit wahrgenommen wird. Die hierbei wahrgenommene Intensität des Schalles hat einen physikalischen und einen subjektiven Aspekt. Physikalisch ist die Schallstärke eindeutig über den Schalldruck definiert. Für die Beschreibung der aus der Schallstärke resultierenden subjektiven Hörempfindung Lautheit, reicht die physikalische Schallstärke allein nicht aus. Für die vom Gehör gebildete Lautheit haben daher Schalldruckpegel, Frequenzbereich und Dauer des Schallsignals Einfluss.

Für die Lautheitsbildung gelten erst Schalle oberhalb der Dauer von 200 ms als Dauerschall. Das bedeutet, dass die Lautheitsempfindung nicht weiter zunimmt, wenn man die Darbietungsdauer über diesen Zeitraum ausdehnt. Sinustöne mit einer Dauer

unter 200 ms erscheinen indes leiser als Sinustöne gleicher Frequenz und Amplitude mit längerer Schalldauer.

Basierend auf seinen umfangreichen Hörtests hat Zwicker ein Lautheitsmaß (engl. loudness) entwickelt, das für stationäre Signale ein deutlich verbessertes Maß als dB(A) darstellt. Die Einheit der Lautheit ist Sone. Im Gegensatz zur dB(A) ist dies eine lineare Größe. Daher auch der große Vorteil der Sone-Skala: sie gibt das menschliche Lautstärkeempfinden linear wieder - 7 Sone sind auch tatsächlich doppelt so laut wie 3,5 Sone.

Der Bezugspunkt ist 1 Sone, dies entspricht einem Sinuston mit 1000Hz bei einem Pegel von 40dB.

8.7 Die Tönhohe, Tonheit

Die Tonhöhe[26] wird durch die Frequenz des Schallereignisses bestimmt. Für den Zusammenhang zwischen physikalischer Größe und Gehörempfindung muss die Frequenz durch die empfundene (subjektive) Tonhöhe, die sogenannte *Verhältnistonhöhe* oder *Tonheit*, ersetzt werden. Die Einheit für die Tonheit ist [Mel] (nach [Ste37]) und [Bark] (1Bark = 100 Mel).

Die Mel–scale [Ste37] ist eine Skala für die Wahrnehmung der Tonhöhe. Sie bildet die Frequenzen so ab, wie die Abstände der Tonhöhen der Wahrnehmung nach empfunden werden. Dies wird erreicht indem ein Ton von 1000Hz bei 40dB mit einer Tonheit von 1000 Mel festlegt werden. Ab etwa 500 Hz müssen immer größere Sprünge in der Frequenz vollzogen werden um einen wahrnehmbare Steigerung in der Tonhöhe zu erzielen.

Die Bark–scale entwickelt 1961 von Eberhard Zwicker ist eine weitere psychoakustische Skala, er benannt diese nach Heinrich Barkhausen. Diese unterteilt die Frequenzen in Frequenzbänder. Die Skala reicht von 1 bis 24, diese Einheiten repräsentieren die kritischen Frequenzbänder des Hörens.

[26] Definition nach DIN 1320 in „Akustik Grundbegriffe“ 1969: „Zuordnung eines Tones oder Tongemisches zu einer bestimmten Skala zwischen tiefen und hohen Tönen.“.

9 Onset detection

9.1 Der Versuch einer Definition und die Zielsetzung

Bello et al. [Bel07] definieren folgendermaßen: Ein musikalischer Ton ist die Summe der augenblicklich abgespielten Noten. *Onsets* bezeichnen den Anfang einer neuen Note oder Tons, worauf der *Attack* des Tons folgt, wobei der starke Anstieg in der Amplitude gemeint ist, welcher in den *Decay* mündet, den Abfall eben dieser. Im direkten Zusammenhang steht der *Transient*, wobei dessen Definition nicht genau ist. Wie man an dem Namen erahnen kann handelt es sich um ein kurzes, vorübergehendes Segment, in welchem sich der Ton in einer nichttrivialen oder relativ unbestimmbaren Art ändert. Sie sind von impulsartiger Natur und enthalten keine vorherrschenden tonalen bzw. periodischen Signalanteile. Der Onset ist also ein einziger Augenblick, welcher den erweiterten Transienten markiert. In den meisten Fällen wird der Onset gleichgesetzt mit dem Beginn des Transienten, oder mit dem frühesten Zeitpunkt, am welchen der Transient entdeckt werden kann. Wobei zu beachten ist, dass ein Offset, also der endgültige Abklang, auch eine transiente Periode verursachen kann. Abbildung 27 [Bel07] stellt eine visualisierte Form einer Definition dar.

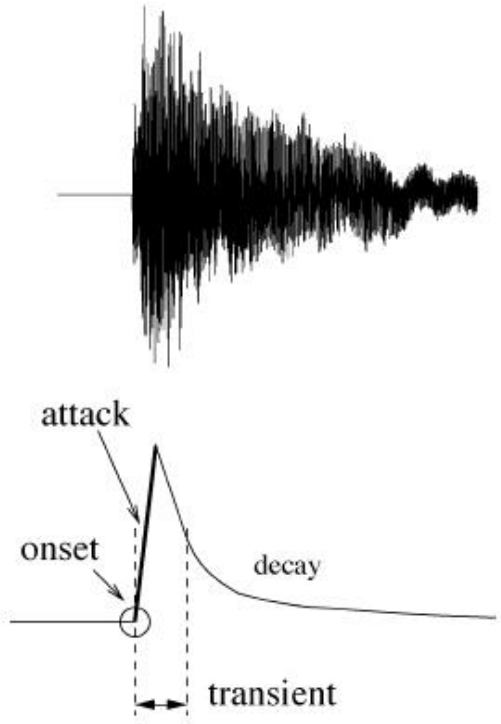


Abbildung 27: Der Ton einer einzelnen Note und dessen Bestandteile: Onset, Attack, Transient und Decay [Bel07]

Grundsätzlich kann nach Vos und Rasch [Vos81] in physikalische Onsets, also der Zeitpunkt ab welchem der Ton messbar ist, und wahrnehmungsmäßigen Onsets, ab

wann der Mensch den Ton hört, unterschieden werden. Die Wahrnehmung eines Onsets wird durch eine bemerkbare Änderung in Intensität, Tonhöhe oder Klangfarbe verursacht [Kla99].

Forschungen von Vos und Rausch [Vos81] haben erwiesen, dass die Wahrnehmung bei 6-15 dB unter dem maximalen Wert des Tons einsetzt. Es ist es daher schwer die genaue Onsetzeit nach der Wahrnehmung zu bestimmen, da Streich- und Blasinstrumente einen langen Anstieg vorweisen, und dadurch die Wahrnehmung dementsprechend später einsetzt, als bei Perkussionsinstrumenten, wo sich die Differenz im Bereich von ein paar wenigen Millisekunden bewegt. Dieses Phänomen wurde bereits bei der wahrnehmungsmäßigen Echtzeit erwähnt.

Diese Definition beruht jedoch auf der Annahme, dass es sich um einen einzelnen, abgeschotteten Ton handelt. Ein realistischeres Szenario sind jedoch mehrere simultane Akkorde verteilt über ein Zehntel einer Sekunde, wobei Maskierung und Hörschwellen eine genaue Definition noch weiter erschweren. Hinzu kommt die Unterscheidung zwischen wahrnehmbaren und messbaren Onsets.

9.2 Die Funktionsweise der Onsetdetection

Audiosignale sind additiver (Töne überlagern sich, sie verbergen einander nur in bestimmten Situationen. Siehe hierfür Maskierung, Abschnitt 8.3) und oszillierender Natur. Deswegen sind Änderungen in der Zeitdomäne kein Nachweis für ein Onset. Es muss daher mittels einer Zwischengröße die Struktur des Signals in vereinfachter Weise dargestellt werden. *Onset Detection* Funktionen arbeiten meist, im Vergleich zu Audiosignalen, mit einer niedrigen Abtastrate (z.B. 100Hz). Dadurch wird eine hohe Datenreduktion erzielt, ohne dabei die benötigten Informationen zur Auffindung von Onsets zu verlieren. Dieser Prozess wird *Detection Function* genannt, welcher eine optionale Vorverarbeitung vorhergeht. Der zweite wichtige Vorgang ist der Peak Picking Algorithmus, welcher die Onsets schlussendlich festlegt. Abbildung 28 [Bel07] verdeutlicht dieses Konzept.

9.2.1 Detection Function

In einem 2005 erschienen Artikel klassifiziert Bello et al. [Bel07] die Detection function in zwei Hauptkategorien: Methoden basierend auf explizit vordefinierten Signaleigenschaften und Methoden basierend auf statistischen Signal Modellen. Es wird nun eine selektive Übersicht der ersten Kategorie vorgestellt, des Weiteren werden Vorschläge und Implementierungen von Dixon [Dix06] und Klapuri [Kla99] mit eingearbeitet.

Diese Methoden basieren auf der Annahme, dass ein Signal selbst alle benötigten Informationen enthält, welche für die Auffindung der Onsets benötigt werden.

Zeitliche Eigenschaften, Zeitdomäne

Bei Beobachtung der Zeitdomäne ist bei Onsets meistens eine Steigung in der Amplitude zu bemerken. Erste Methoden der Onsetdetection basierten auf dieser Idee, es wurden Algorithmen entwickelt welche den Verlauf der Hüllenkurve der Amplitude verfolgten [Sch85]. Dies wurde mittels Gleichrichtung[27] und Glättung (z.B. durch einen

[27] Gleichrichtung (nicht zu verwechseln mit Gleichschaltung) bezeichnet die Umwandlung von Wechselstrom in Gleichstrom.

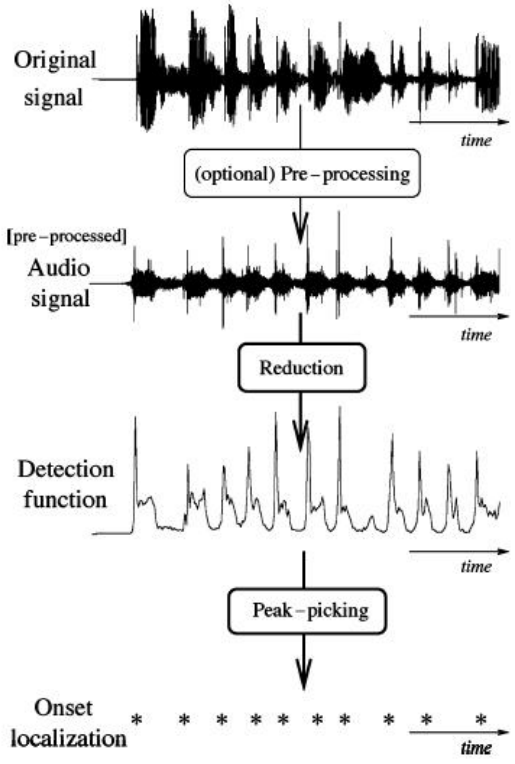


Abbildung 28: Die Vorgehensweise einer typischen Onset–Erkennung [Bel07]

Low-Pass Filter) erreicht. Dieser Ansatz liefert bei reinen Perkussionsaufnahmen mit stillem Hintergrund adäquate Ergebnisse. Jedoch ist dieser Ansatz bei polyphonen Ereignissen unzulänglich.

Frequenz Domäne, Spektrale Domäne

Die Grundidee der meisten Algorithmen basiert auf beobachteten Änderungen in ein oder mehreren Eigenschaften eines Signals. Schreier war der Erste welcher auf die Notwendigkeit hinwies, dass die Aufdeckung dem menschlichen Gehör angepasst und deshalb in mehrere Frequenzbänder unterteilt werden sollte [Sch96]. Die frühesten Systeme verwendeten zwei Bänder, unterteilt in hohe und tiefe Töne [Bil93], dies erwies sich jedoch als nicht ausreichend.

Die weitere Abfolge lässt sich anhand der wichtigsten Techniken bestimmen: *Spectral flux* (Spektraler Fluss), *Phase Deviation* (Phasen Abweichung), *Complex Domain* (Komplexe Domäne, gemeint ist die Kombination der ersten zwei Techniken) und deren Erweiterungen von Dixon. Alle diese Methoden nutzen die STFT mit einer Fensterfunktion $w(m)$, welche mit einer 100 Hz Rate berechnet wird. Wenn $X(n,k)$ das $k$te Frequenzband des $n$ten Fenster der STFT repräsentiert:

$$X(n,k) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(hn+m)w(m)e^{-\frac{2j\pi mk}{N}}$$

Wobei Dixon die Fenstergröße mit $N = 2048$ (46 ms bei einer Abtastrate von 44100 Hz) und die Sprunggröße (die sogenannte *hopsize*) mit $h = 441$ (10 ms, oder 78.5% Überlappung) definiert.

Auf Spektrum basierend

Wie schon erwähnt ist die Auffindung von Änderungen die Grundlage, genau das ist auch der Fall bei der bereits angesprochenen Methode den Anstieg in der Energie als Onset anzunehmen. Masri [Mas96] entwickelte eine *High Frequency Content* Erkennungsfunktion, welche auf der Annahme basiert, dass die Amplitude sich in den niedrigen Frequenzen konzentriert und somit transiente Perioden in höheren Frequenzen leichter zu erkennen sind. Deshalb gewichtet er die Bänder linear, mit der Betonung auf hohen Frequenzen. Die Summe der Energie kann nun als Vergleich mit anderen Fenstern dienen. Dies ist jedoch wie sein Pendant in der Zeitdomäne nur bei perkussiven Stücken erfolgreich.

In Folge entstand die *Spectral Flux*–Funktion $SF(n)$ (auch *Spectral Difference* genannt), welche in mehrere Frequenzbänder unterteilte und diese $((n, k))$ mit deren direkten Vorgängern $((n-1, k))$ ungewichtet miteinander vergleicht. Es wird also nicht nach einer Änderung der gesamten Energie gesucht, sondern nach der Änderung in der Verteilung der Energie über die Frequenzbänder. Hierbei wird eine Distanzfunktion formuliert, welche das Spektrum als Punkte in einem x-dimensionalen Raum annimmt. Man errechnet dies mittels summierten Betrags der ersten Differenz über alle Frequenzbänder:

$$SF(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} H(|X(n,k)| - |X(n-1,k)|)$$

Wobei H(x) = x die Halbwellengleichrichtungsfunktion ist (also 0 für negative Argumente). Die Gleichrichtung hat zur Folge dass nur ein Anstieg der Energie in die Berechnung einfließt, um somit nicht fälschlicherweise Offsets zu erkennen. Laut Dixon haben empirische Tests ergeben, dass die L1-Norm der L2-Norm vorzuziehen ist und die lineare Größenordnung der logarithmischen von Klapuri [Kla99].

Phasen basierend

Vorherige Methode verwendet nur die Größenordnung der Frequenzbänder als Information. Man kann sich aber auch das Phasenspektrum zu Nutze machen. Die Phase $\psi(n, k)$ von $X(n, k)$ erhalten wir aus den Polarkoordinaten, wobei $X(n, k)$ wie folgt umgewandelt werden kann:

$$X(n,k) = |X(n,k)|e^{j\psi(n,k)}$$

wobei $-\pi < \psi(n, k) <= \pi$.

Während einem gleich bleibenden Zustand sind die momentanen Sinuskurven ungefähr konstant über die benachbarten Fenster. Die Grafik (Abbildung 29 [Bel07]) soll dies visualisieren. Betrachtet man also das $k$te Frequenzband muss für einen konstanten Ton gelten:

$$\psi(n,k) - \psi(n-1,k) \approx \psi(n-1,k) - \psi(n-2,k)$$

Analog dazu kann man kann die erste Ableitung verwenden. Besser geeignet ist hierbei die 2te Ableitung, welche wieder über die gesamten Frequenzbänder gemittelt wird. Die

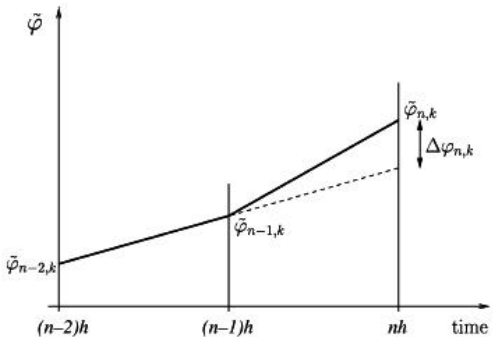


Abbildung 29: Das Phasendiagramm aus [Bel07], welches die augenblicklichen Frequenzen als Phasenableitungen über angrenzende Fenster darstellt. Stationäre Sinusoide sollten konstant sein (strichlierte Linie).

resultierende Funktion wird in der Literatur *Phase Deviation* $PD(n)$ genannt

$$PD(n) = \frac{1}{N} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |\psi''(n,k)|$$

Die hybriden Lösungen, die komplexe Domäne

Die spektrumbasierte Methode ist unkompliziert, daher weit verbreitet, sie basiert aber einzig auf dem betonten Energieanstieg bei einem Akkord. Dies ist jedoch nicht immer der Fall, im Speziellen bei komplexen Musikstücken wenn Maskierungseffekte eintreten. Als Alternative wurde die phasenbasierte Methodik vorgestellt, welche die Auffindung von weniger prominenten Akkorden verbesserte. Jedoch ist dies anfällig durch Phasenverzerrungen und Schwankungen verursacht durch die Phase geräuschartiger Komponenten. [Dux02]

Amplitude und Phase kann aber auch gemeinsam für die Auffindung von Abweichungen verwendet werden indem ein Frequenzband $X(n,k)$ aufgrund der zwei vorherigen Bänder approximiert wird. Für den Zielwert $X_T(n,k)$ wird angenommen, dass wenn kein Onset vorhanden ist, die Amplitude und Rate der Änderung der Phase konstant bleibt:

$$X_T(n,k) = |X(n-1,k)|e^{\psi(n-1,k)+\psi'(n-1,k)}$$

Hieraus kann die so genannte *Complex Domain*–Funktion $CD(n)$, als Summe der absoluten Abweichungen von den Zielwerten definiert werden:

$$CD(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k) - X_T(n,k)|$$

Die Erweiterungen von Dixon

Eine bereits erwähnte Problematik der phasenbasierten Methode ist die Anfälligkeit bei Geräuschen, verursacht von Komponenten mit geringer Energie. Dixon sieht dieses Problem dadurch verursacht, dass alle Frequenzbänder gleich bewertet werden, obwohl sich die Energie des Signals auf die Frequenzbänder, welche Teile des aktuellen Tons beinhalten, konzentrieren. Also schlägt er eine Gewichtung der Frequenzbänder nach deren Größenordnung vor. Ähnlich wie bei der Complex Domain Funktion werden hier Amplitude und Phase kombiniert, jedoch wird hierbei an Hand der Phase eine Gewichtung durchgeführt. Die resultierende Funktion nennt Dixon *Weighted Phase Deviation* $WPD(n)$:

$$WPD(n) = \frac{1}{N} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k)\psi''(n,k)|$$

Desweiteren schlägt Dixon hierbei eine Normalisierung vor, welche in der *Normalised Weighted Phase Deviation*–Funktion NWPD(n) umgesetzt wurde:

$$NWPD(n) = \frac{\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k)\psi''(n,k)|}{\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k)|}$$

Ein weiteres Problem der Complex Domain Funktion ist, dass zwischen Anstiegen und Abstiegen nicht unterschieden wird. Da dies zu der unerwünschten Auffindung von Offsets führen kann, wird ein ähnlicher Ansatz wie bei der Spectral Flux Funktion gewählt. Dies kann beispielsweise mit der Gleichrichtungsfunktion erbracht werden. Die *Rectified Complex Domain*–Funktion $RCD(n)$:

$$RCD(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} RCD(n,k)$$

wobei

$$RCD(n) = \begin{cases} |X(n,k) - X_T(n,k)|, & \text{falls } |X(n,k)| \geq |X(n-1,k)| \\ 0, & \text{sonst} \end{cases}$$

Die psychoakustisch motivierten Methoden

Ansii Klapuri [Kla99] nimmt an, dass eine logarithmierte Energie in den Bänder der menschlichen Wahrnehmung mehr entsprechen würde. Er entwickelte eine *Detection* Methode, wobei vorhergehend das Signal auf 70dB normalisiert wird. Eine Filterbank unterteilt dieses in 21 sich nicht überlappende Bänder. Die Aufteilung dieser Bänder basiert auf Psychoakustik, so dass die untersten drei Bänder jeweils Sprünge um eine Oktave darstellen, während die restlichen eine Auflösung von einer Drittel Oktave vorweisen. Analog zu den anderen Methoden, wird nun jedes Band analysiert, jedoch bezieht er auch die Lautstärke mit ein.

Klapuri verwendet in der Frequenzbandebene den Amplitudenumfang, befindet aber die erste Ableitung als Indikator für zu ungenau. Er verweist darauf, dass tiefe Töne

längere *Attack*–Phasen haben, und sich dadurch Onsetzeiten der Töne je nach Art verschieben. Außerdem verhalten sich transiente Perioden unvorhersehbar und weisen daher keine monotone Steigung auf, wodurch sich mehrere lokale Maxima ergeben. Er verwendet daher eine relative erste Ableitung, wobei die Amplitude logarithmiert wird:

$$W(n) = logA'(n)$$

wobei $A(n)$ den Amplitudenumfang bezeichnet.

Die Begründung hierfür liegt in der Psychoakustik, dieselbe Steigerung an Intensität ist bei einem leisen Signal prägnanter als bei einem lauten Signal. Laut Moore [Moo95] ist die kleinste wahrnehmbare Änderung proportional zu der Intensität des Signals. Also ist $\Delta I/I$, der Webersche Bruch konstant. Onset-Komponenten werden also aus lokalen Maxima in $W(n, k)$, welche über einen bestimmen Grenzwert liegen, gewonnen.

Folglich werden die Intensitäten der einzelnen Komponenten festgelegt. Diese werden aus der ersten Ableitung multipliziert mit der mittleren Frequenz des Bandes gewonnen. Dies wird für alle lokalen Maxima der *Attacks* berechnet. Sobald alle Intensitäten bestimmt wurden, werden alle Komponenten, welche sich im Umfeld von 50 ms zu einer lauteren Komponente befinden, entfernt.

Die finale Phase umfasst die Kombination der gewonnenen Daten und die Anwendung des Lautstärkemodells. Hierfür können verschiedenste Modelle herangezogen werden, wobei Klapuri eine vereinfachte Fassung des von Moore, Glasberg und Baer vorgeschlagenen Modells [Moo97] verwendet. Durch die Intensitäten der einzelnen Frequenzbänder kann die Lautstärke des Signals berechnet werden.

Abschliessend werden alle Onset-Komponenten verschiedener Bänder nach Zeit sortiert, wodurch Onset-Kandidaten entstehen. Danach werden diesen eine korrespondierende Lautstärke zugewiesen, dies geschieht indem all jene Komponenten, welche sich innerhalb eines Zeitfensters von 50 ms um den Kandidaten befinden, durch das Lautstärkemodell aufsummiert werden. Dadurch entsteht ein Vektor mit Kandidaten und deren Lautstärke, wobei jene gewählt werden, welche den Grenzwert von 25 dB (bei normalisierter Lautstärke von 70 dB) überschreiten.

Soweit ein kurzer Überblick der verschiedenen Detection Functions, für eine detaillierte Untersuchung wird auf Collins [Col05] für die psychoakustischen Methoden, und auf Bello et al. [Bel07] für einen Querschnitt der gängigen Methoden, verwiesen. Wobei letzterer auch die in dieser Arbeit nicht beschriebenen statistischen Methoden umfasst. Beide erklären detailliert die Funktionsweisen und Zusammenhänge und schließen sie mit umfassendem Vergleich und Bewertung ab.

9.2.2 Peak Picking

Bei einer guten Detection Function, werden Onsets als gut lokalisierbare Auslenkungen sichtbar. Als identifizierbare Merkmale gelten hierbei lokale Maxima in der Funktion, ein steiler Anstieg in selbiger oder aber andere charakteristische Formen. Da diese Formen stets variabel sind und andere Umstände, wie Maskierung oder andere Einflüsse, welche nicht von einem Onset rühren, wie beispielsweise Vibrato, das Ergebnis bestimmen, ist eine robuste Peak Picking Funktion von fundamentaler Bedeutung.

Bello et al. [Bel07] unterteilen diesen letzten Abschnitt der Onsetdetection in eine optionale Postbearbeitung, Festlegung der Grenzwerte und das eigentliche Peak Picking.

Die Postverarbeitung ist ein Prozess welcher die folgenden Verarbeitungsschritte vereinfachen soll. Bello et al. definiert genauer in einen Prozess, welcher durch Steigerung der Uniformität und Konsistenz der auf Onset hinweisenden Daten, diese idealerweise in isolierte, leicht erkennbare lokale Maxima transformiert. In diesen Bereich fällt beispielsweise die Reduktion von Störsignalen durch Glättung oder Normalisierung.

Aber selbst nach diesem Prozess müssen lokale Maxima nicht unbedingt einen Onset darstellen. Deshalb ist es nötig Grenzwerte festzulegen. Dieser Vorgang nimmt direkten Einfluss auf die Ergebnisgüte, spezifisch auf die der „False Positive"[28] und der „False Negatives". Ein zu hoher Grenzwert reduziert die Anzahl der „False Positives" steigert jedoch die der „False Negatives" [Dix06]. Zwei Vorgehensweisen sind hierbei möglich: der adaptive oder der fixe Grenzwert.

Während der fixe Grenzwert bei Musikstücken mit geringer Dynamik durchaus gute Ergebnisse liefern kann, liegt die Problematik bei Stücken mit großem Dynamikumfang. Bei diesen neigt diese Methodik dazu Onsets in leisen Passagen zu übersehen, dadurch die Anzahl der „False Negatives" ansteigen lässt, während bei lauten Passagen überbewertet wird, wodurch vermehrt „False Positives" markiert werden.

Der adaptive Ansatz wird diesen Anforderungen gerecht. Der Grenzwert wird hierbei im Allgemeinen durch eine geglättet Version der Detection Function berechnet. Lineare Methoden, wie beispielsweise ein Low-Pass-Filter oder nichtlineare Methoden, wie der lokale Median können hierbei verwendet werden. Bello et al. spricht sich für den Medianfilter aus, da andere nichtlineare Verfahren wie die Quadratur der Detection Function zu Ausreißern bei großen Maxima führen kann, welche kleinere angrenzende Maxima maskieren würden.

Jedoch gibt es selbst bei der adaptiven Methode Parameter wie Konstanten und Cutoff Frequenzen die manuell durch Versuche eingestellt werden, somit ergibt sich für jede Onsetdetection eine unterschiedliche Peak Peaking Funktion. Das bedeutet, dass die Detection Function und die Peak Peaking Funktion nicht getrennt voneinander evaluiert werden können. Anhand der Funktion von Dixon soll nun die Funktionsweise detailliert dargestellt werden.

Er normalisiert vorerst die Dection Function, so dass 0 der Mittelwert und 1 die Standardabweichung ist. Ein Maxima in dieser Funktion bei der Zeit $t = nh/r$ (n Anzahl der Fenster, h Hopsize, r Samplingrate) wird dann als Onset erkannt, falls es folgende Bedingungen erfüllt:

$$f(n) \geq f(k) \text{für alle k so dass } n - w \leq k \leq n + w$$

$$f(n) \geq \frac{\sum_{k=-n-mw}^{n+w} f(k)}{mw + w + 1} + \delta$$

$$f(n) \geq g_\alpha(n - 1)$$

Wobei w = 3 die Fenstergröße für die Auffindung des lokalen Maxima und m = 3 ein Multiplikator ist, damit der Mittelwert bei Spitzen über ein größeres Feld berechnet wird. δ ist eine Konstante, welche zum lokalen Mittelwert hinzu addiert einen Grenzwert bildet, welcher überschritten werden muss. Dixon verwendet noch eine zweite Grenz-

[28] „False Positves" bezeichnen in diesem Fall als Onset markierte Stellen, wobei dies jedoch nicht der Fall ist. Analog dazu bezeichnen „False Negatives" nicht markierte Onsets.

wertfunktion mit dem Parameter α, diese berechnet sich rekursiv:

$$g_\alpha(n) = max(f(n), \alpha g_\alpha(n-1) + (1-\alpha)f(n))$$

Wobei laut Dixon letztere Funktion weniger ins Gewicht fällt, sobald ein passender Wert für δ gewählt wurde.

9.2.3 Die Bewertung der Algorithmen

Die Hauptschwierigkeit bei der Bewertung ist die Beschaffung von Testdaten, dabei wird ein ausreichend großer und ausgeglichener Satz an Musikstücken samt deren Onset Zeiten benötigt. Dies nennt man *„Ground Truth Data“*. Da aber nur bei einer kleinen Gruppe von Musikdaten exakte Onset Zeiten vordefiniert sind, wie bei computergestützten Klavieren, muss bei allen übrigen manuell markiert werden. Wobei die computergestützte Markierung den physikalischen Onsets gleich kommt, wohingegen die manuelle Onsetmarkierung der Wahrnehmung nach erfolgt.

Bei ausgeglichen ist gemeint, dass die Daten repräsentativ für alle Musikkategorien sind, da die Methoden unterschiedlich gut auf unterschiedliche Problemstellungen reagieren. Grundsätzlich kann man ein Musikstück in Stil kategorisieren. Des Weiteren unterscheidet man bei Instrumenten in gestimmte Perkussion (z.B. Klavier), nicht gestimmte Perkussion (z.B. Schlagzeug) und gestimmte nicht perkussive Instrumente (z.B. Streichinstrumente). Des Weiteren kann Musik monophonischer oder polyphonischer Natur sein, was den Grad der Komplexität bestimmt. Ein typisches Testset ist das von Bello, welches auch von Dixon für einen besseren Vergleich übernommen wurde, es umfasst je eines aus der Instrumentengruppe und ein polyphones, komplexes Werk aus der populären Musik und bei Dixon noch zusätzlich Jazz.

Ein weiteres Problem ist wie Ergebnisse bewertet werden, wobei das Konzept der „False Positives“ und „False Negatives“ bereits vorher genommen wurde. Die Bewertung dieser kann aufgrund von Precision P and Recall R erfolgen, welche in Information Retrival häufig Anwendung finden.

$$P = \frac{t^+}{t^+ + f^+}$$
$$R = \frac{t^+}{t^+ + f^-}$$

Wobei f^+ „False Positives“, f^- „False Negatives“ und t^+ „True Positives“, also richtig markierte Onsets, repräsentieren.

Da aber Stellgrößen diese Werte beeinflussen, ist mit einem einzelnen Parameter noch kein aussagekräftiger Vergleich möglich. Deshalb wird eine bestimmte Anzahl von Parameter verwendet. Daraus ergibt sich ein 2 dimensionaler Raum, welcher die Beziehung zwischen Precision und Recall mit den unterschiedlichen Stellgrößen darstellt. Diese wird Receiver Operating Characteristic (ROC) genannt. Ist ein einzelner Wert erwünscht kann die *F–measure*–Funktion den optimalen Punkt in der ROC berechnen.

$$F = \frac{2PR}{P+R} = \frac{2t^+}{2t^+ + f^+ + f^-}$$

Dieser Wert wird auch bei der MIREX[29] [MIR07] verwendet, wobei dieser für die Endbeurteilung über das Feld der verschiedenen Musikstücke gemittelt wird. Außerdem werden allgemeine Leistungskriterien wie Geschwindigkeit, Skalierbarkeit und Robustheit gegenüber Lautstärke und Geräusche bewertet. Wobei auch Zeitpräzision (Toleranz +/- 50 ms) beurteilt und eine unterschiedliche Gewichtung für unterschiedliche Instrumententypen mit einbezogen wird. Ein weiterer Aspekt wird beachtet, liegen Onsets sehr nahe zusammen, hängt die richtige Erkennung von Zeitabstand und Maskierungen ab. MIREX führt deshalb zwei weitere Größen ein, die Merged (Engl. für verschmelzen (mit)) Onsets (Zwei Onsets als ein Onset erkannt) und Double (Engl. für doppelt) Onsets (Ein einzelner Onset als zwei Onsets erkannt).

Für eine genaue Analyse samt Testergebnissen wird auf die Arbeit von Dixon [Dix06], Bello et al [Bel07], Collins [Col05] und die Homepage des MIREX Contest [MIR07] verwiesen. Mehr zu der Methologie bei der Evaluierung ist in [Lev04] zu finden.

Dixon macht jedoch darauf aufmerksam, dass es sich bei den Tests um optimistische Ergebnisse handelt. Da meistens manuell markiert werden muss, handele es sich folglichermaßen um simple Daten. Das in Vergleich dazu von Dixon erstellte Datenset besteht aus Mozart Sonaten aufgenommen mit einem computerüberwachten Bösendorfer Klavier. Das Datenset umfasst 106054 Noten und ist von komplexer Natur als die von anderen Evaluierungen. Das Ergebnis fiel daher weniger positiv aus. Wobei Dixon anmerkt, dass es sich noch immer um ein zu optimistisches Ergebnis handelt, da die Parameter anhand von Feedback der Ground Truth Daten erstellt wurden und alle Aufnahmen von dem gleichen Instrument entstammen bei gleich bleibenden Aufnahmebedingungen.

Bello et al definieren die Anforderung an eine Onsetdetection, als eine Methodik die mit minimaler Komplexität die erforderlichen Aufgaben erfüllt. Genauer gesagt eine Balance zwischen Komplexität der drei Hauptaufgaben. Weiter muss festgestellt werden, dass die verschiedenen Methoden verschiedenen Anforderungen gerecht werden müssen. Des Weiteren sehen sie Potential in der Kombination von Techniken wie es bei der Complex Domain vorgezeigt wurde.

Nach Dixon liegt der Bedarf in der Automatisierung der Parametereinstellung, mit dem Endziel einen vollkommen selbstständigen Onsetdetection Algorithmus zu ermöglichen. Er verweist darauf, dass die MIREX 2005 ein neuronales Netzwerk gewonnen hat. Des Weiteren schlägt er eine detaillierte Fehleranalyse vor, um besser bestimmen zu können welche Methoden in welchen Bereichen am Besten geeignet sind, und spricht sich gleichzeitig auch für die Kombination von Methoden aus.

9.2.4 Die Einsatzgebiete der Onsetdetection

Durch Resultate der Onsetdetection können Bereiche wie beispielsweise die Datenkomprimierung oder die Audio/Video Synchronisierung verbessert, erleichtert oder gar automatisiert werden. Für den DJ Bereich ergeben sich jedoch auch Möglichkeiten: Der haptische Feedbackmodus bumps-for-beats von D'Groove basiert auch auf der Onsetdetection. Die Möglichkeiten bei dem visuellen Feedback sind jedoch viel größer.

[29] MIREX ist die Abkürzung von „Music Information Retrieval Evaluation eXchange". Für verschiedene Disziplinen des MIR's werden jährlich Wettbewerbe organisiert.

Wie schon einleitend erklärt spielen Onset insbesondere bei Scratching eine wichtige Rolle. Zur Erinnerung: Wenn die Platte vor Beginn des Onset in den Beginn des Tones hineinbewegt wird, entsteht ein scharfer Onset, wird jedoch während (oder innerhalb) des Tones gescratcht entsteht ein Glissando-Effekt. Es ist also wünschenswert diese Onsets sichtbar zu machen. Dies wird in dem praktischen Teil der Arbeit umgesetzt indem bei der Wellenformabstraktion die Linien die einen Onset markieren Rot markiert werden.

Tempo Tracking, welches sich direkt aus der Onsetdetection ableiten lässt, ermöglicht einerseits ein Stück nach Geschwindigkeiten zu analysieren ohne es dabei hören zu müssen und außerdem liefert es bei Beatmatching ein visuelles Feedback. Die Aufgabe eines Tempo Trackers ist zu jeder Zeit in einer verständlichen Form Auskunft über die augenblicklich wahrgenommene Geschwindigkeit zu geben. Ein möglicher Indikator hierfür ist Beats per Minute (BPM).

Dixon [Dix] sieht Notwendigkeit solcher oder ähnlicher Trackingsysteme für Analysezwecke bei expressiven Musikaufführungen. Hierbei ist gemeint, dass ein Künstler während einer Aufführung über die vorliegenden Noten hinaus dem Musikstück einen persönlichen Ausdruck verleiht, indem er allmählich oder abrupt das Tempo oder die Lautstärke variiert. Dixon sieht dies als ein Hauptelement der E-Musik an, im Speziellen der klassischen Musik. Jedoch ist dieses Paradigma leicht auf DJ Aufführungen anzuwenden. Jedoch ist die von Dixon gewählte Darstellungsform *„The Performance Worm“* für klassische Musik, im Speziellen Klaviermusik, zwar sehr wohl, jedoch für die stark Tempo– und Dynamikschwankungen in Turntablismbereich nicht geeignet.

Ein weiterer Bereich der sich durch die Onsetdetection eröffnet hat ist die Musiksegmentation. Hier ergibt sich einerseits die Unterteilung des Musikstücks in seine formale Struktur, wobei dies als visuelles Feedback dienen kann. Andererseits kann eine Unterteilung in weiter verwendbare Samples erfolgen. Dieses Konzept kann durch ein automatisierte Bestimmung von Looping–Zeitpunkten erweitert werden.

10 Das virtuelle Turntable — Die Umsetzung

Der Prototyp implementiert eine Benutzeroberfläche, welche sich nach den Vorschlägen von Andersen in [And02] orientiert. Eine kreisförmige Fläche, eine Anlehnung an die Schallplatte, und ein darunter befindlicher Scrollbalken bilden die Hauptkomponenten des visuellen Feedbacks.

10.1 Die Programmierumgebung

10.1.1 Java

Erste Versuche für den Prototyp fanden mittels der API Java Sound statt. Die hierbei erzielten Latenzzeiten lagen jedoch über den anvisierten Wert. Dieser sollte sich an 10 ms orientieren, da dies ungefähr der Wahrnehmung nach einer Echtzeit entsprechen würde. Vor allem da bei der letzten Version von Java Sound [SUN] die meisten interimistischen Buffer zwischen Applikation und API entfernt wurden [Jui07], sind weitere Besserungen zu erwarten. Nichtsdestotrotz mussten die Alternativen evaluiert werden.

Es existieren viele Frameworks, welche eine niedrige Latenzzeit gewährleisten, jedoch sind diese in C oder C++ geschrieben. Diese Präferenz gegenüber Java resultiert aus einigen Vorteilen welche C bietet: die Echtzeitgarantie und die bessere Leistungsfähigkeit. Jedoch bietet Java auch Vorteile durch Programmier Komfort, weitere Verbreitung und Portabilität. Aus diesem Grund entstand Decklight 4 [Jui07], ein Java Framework welches die Latenzzeiten minimiert. Dieses wurde leider erst in der Endphase des Projekts entdeckt, wodurch keine praktische Evaluierung möglich war.

10.1.2 Java Audio Synthesis System

Für die Entwicklung des Prototyps wurde letztendlich Java Audio Synthesis System (JASS) [Doe] gewählt, eine Programmierumgebung geschrieben in Java. Es besteht aus Java Interfaces und abstrakten Klassen, welche die volle Funktionalität für ein Audioprogramm implementieren, ohne jedoch einzuschränken. Ein weiterer Vorteil ist, dass der gesamte Source Code für nicht kommerzielle Nutzung frei verfügbar ist. Latenzzeiten werden durch *native*–Methoden [30] minimiert. Augenblicklich werden LINUX (ALSA and OSS), Macintosh (OS/X), and Windows (DirectX, ASIO blocking API, ASIO callback API) unterstützt. Alle *native*–Methoden bis auf ASIO callback benutzen RtAudio [Sca], eine Echtzeit Audio i/o Klasse in C++. Als Schnittstelle für die Aufrufe dient Java Native Interface[31] (JNI).

10.1.3 Processing

Processing [Fry] ist eine Open Source Programmiersprache, welche Interaktion, Animation und visuelle Elemente und auch Audio auf Java Basis ermöglicht. Zielgruppe sind Studenten, Künstler, Designer, Wissenschaftler und ist vor allem für die Entwicklung von Prototypen sehr beliebt. Der Vorteil liegt darin, dass bestehende Java Programme

[30] Engl. für „einheimisch“, bezogen auf das Betriebssystem

[31] Eine API welche Aufrufe von betriebssystemspezifischen Funktionen bzw. Methoden aus der betriebssystemunabhängigen Programmiersprache Java heraus ermöglicht.

leicht in die Processing Umgebung integrierbar sind. Des Weiteren gibt es viele Frameworks, welche die Möglichkeiten von Processing noch erweitern. Es existieren auch einige für den Audiobereich, jedoch erwiesen sich diese für die gewünschten Anforderungen als unzureichend, wieder aufgrund von zu hoher Latenzzeit.

Ausschlaggebend für die Entscheidung bezüglich GUI Umgebung war jedoch eine OPEN-GL Unterstützung, welche von Processing gewährleistet wird.

10.2 Die Umsetzung der Benutzeroberfläche - Das visuelle Feedback

Der erste Schritt für die Benutzeroberfläche war die Abstraktion der Wellenform. Da von einer kreisförmigen Darstellung ausgegangen wird, ergab sich ein Richtwert von 360 gleichzeitig dargestellten Werten. Die Berechnung dieser Daten sollte noch vor der Laufzeit geschehen, um die Latenzzeiten nicht zu beeinträchtigen. Entnommen aus [Esp04] wurde eine Methode, wobei eine gewünschte Anzahl von Samples, ein Ausschnitt, gewählt und dessen Minima und Maxima bestimmt werden. Die Fenster über das gesamte Musikstück ergeben somit ein Array welches eine Repräsentation der Amplitude ermöglicht, also die maximale Auslenkung, wobei diese Lautstärke darstellt. Dabei muss jedoch beachtet werden, dass bei Stereoaufnahmen, jeweils zwei Minima und Maxima bestimmt werden.

Bei diesen Daten handelt es sich um eine orthogonale Darstellung(siehe Abbildung 13), entsprechend dem kartesischen Koordinatensystems. Um diese auf der Kreisform darstellen zu können müssen sie in das Polarkoordinatensystems überführt werden. Für die Umrechnung zwischen Polarkoordinaten und kartesische Koordinaten gilt:

$$\vec{r} = rcos\varphi\vec{e_x} + rsin\varphi\vec{e_y}$$

Wobei φ den Winkel bezeichnet. Demnach gilt für die Umrechnung von Polarkoordinaten in das kartesische System:

$$x = r\cos\varphi$$

$$y = r\sin\varphi$$

Der Winkel φ stellt den Zeitausschnitt dar. Ermittelt werden nun zwei Punkte $(minX, minY)$ und $(maxX, maxY)$, nämlich die minimale und die maximale Auslenkung innerhalb des darzustellenden Ausschnitts. Die hierfür benötigte Variable r, welche den Radius repräsentiert, wird folgendermaßen bestimmt: Ein imaginärer Kreis, 128 Punkte von dem Mittelpunkt entfernt, stellt die Auslenkung der Höhe Null dar. Aus einer Addition der Auslenkungen mit den erwähnten 128 Punkten lässt sich Radius r bestimmen.

Bei reinen Abspieleinheiten werden bei der Wellenrepräsentation einzig die noch folgenden Daten dargestellt. Bei Bearbeitungs- und DJ-Programmen sind jedoch auch die vergangenen Daten von Bedeutung. Insbesondere ist das bei Scratching der Fall, da hier eine Manipulation der Abspielgeschwindigkeit auch gegen die Abspielrichtung möglich ist.

Die aktuelle Position, welche die derzeitig abspielenden Daten markiert, wurde analog zu dem Tonarm bestimmt, sie liegt also bei 315°. Dies ist zwar realitätsnahe, bedarf jedoch einer gewissen Gewöhnung, da sich bei der klassischen orthogonalen Darstellung die Zeit von links nach rechts gelesen wird. Die auditive Umsetzung liefert die benötigten

Daten, welche für die Steuerung der Wellenform notwendig sind. Die Position kann hierbei auf das Sample genau angegeben und mittels einfacher Umrechnung in eine Zeitangabe ($Sample/Frequenzrate$) oder in das korrespondierende Amplitudenfenster ($Sample/Fenstergröße$) umgewandelt werden. Wobei die Darstellung nur aktualisiert wird, wenn sich die Position soweit verändert, so dass eine Verschiebung der Wellenform notwendig wird.

Folgender Code umfasst die beschriebene Funktionalität: Das Array *waveForm* beinhaltet die Amplitudenwerte, *sampleAnzahl* die Gesamtzahl an Samples und *pos* die Position in den Samples.

Listing 1: Polarkoordinaten

```
for (int i=0;i < 310 && pos + i < sampleAnzahl / fensterGroesse; i++)
{
  minX = (x + 128 + (waveForm[pos + i][0]
         + waveForm[pos + i][2])/2.5) * cos(radians(45 - i));
  minY = (x + 128 + (waveForm[pos + i][0]
         + waveForm[pos + i][2])/2.5) * sin(radians(45 - i));

  maxX = (x + 128 + (waveForm[pos + i][1]
         + waveForm[pos + i][3])/2.5) * cos(radians(45 - i));
  maxY = (x + 128 + (waveForm[pos + i][1]
         + waveForm[pos + i][3])/2.5) * sin(radians(45 - i));

  //Onsets rot, sonst weiß
  if(waveFormOnsets[pos + i])
       stroke(255,0,0);
  else
    stroke(255);

  line(pX + minX, pY + minY, pX + maxX, pY + maxY);
}
```

Ein Amplitudenfenster, dargestellt als einfacher Strich, ist also dann sichtbar wenn sie sich innerhalb einer bestimmten Distanz in beiden Richtungen zu der jetzigen Position befindet und verschiebt sich in Abhängigkeit von Abspielgeschwindigkeit und -richtung. Um Vergangenheit und Zukunft voneinander abzugrenzen, wird die Farbintensität in der Vergangenheit vermindert. Eine weitere Abgrenzung ist bei der zweiten Grenze zwischen Vergangenheit und Zukunft notwendig, hierbei wird die Vergangenheit fortlaufend langsam ausgeblendet. Die Abspielrichtung ist wie bei dem analogen Vorbild im Uhrzeigersinn.

Rot gefärbte Amplituden stellen Onsets dar. Diese wurden mit der Onsetdetection aus Dixons Beatroot [Dix] bezogen. Die Berechnung, da zeitaufwändig, wird ebenfalls vor der Laufzeit berechnet. Die Angabe von Beatroot erfolgt in Sekunden, welche in Samples ungewandelt und dem zugehörigen Amplitudenfenster zugeordnet werden.

Zusätzlich kann zwischen drei Größenabstufungen der Darstellung gewählt werden. Per Mausklick auf das *Zoom*–Feld können diese angewählt werden. Folgende Abbildung 30 stellt die Benutzeroberfläche des Prototypen dar.

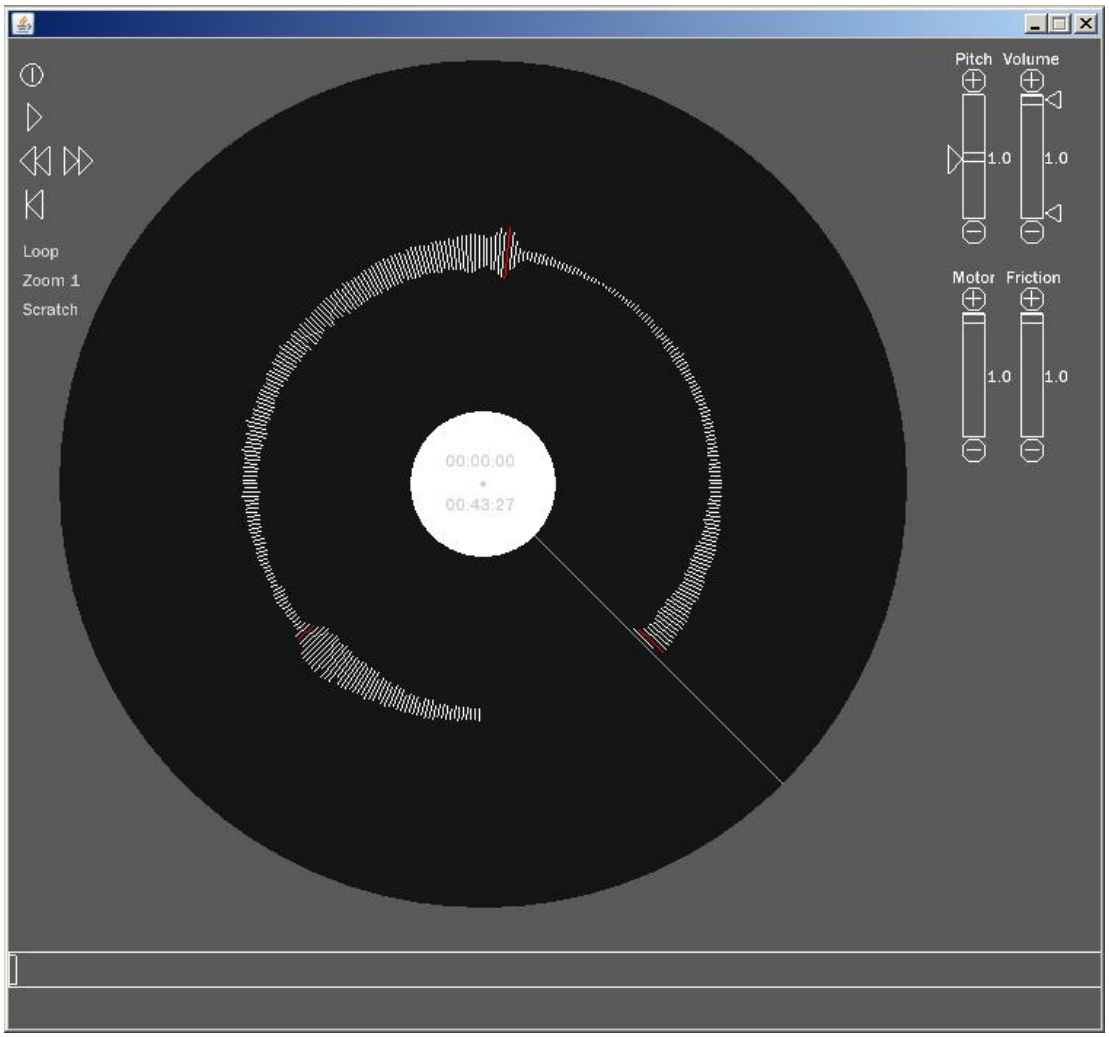


Abbildung 30: Die Benutzeroberfläche des Prototypen

10.3 Die Steuerung

Seitlich von dem virtuellen Turntable befinden sich Kontrolleinheiten. Der Motor wird mit den Ein/Aus Schalter ganz oben aktiviert/deaktiviert. Durch Betätigung des Play Knopfs fängt die virtuelle Platte an zu drehen. Nun gibt es im Bedienfeld zwei Optionen: Betätigt man den Play Knopf noch einmal, wird abgebremst, dies geschieht bei einem Plattenspieler beispielsweise durch eine magnetische Vorrichtung. Drückt man jedoch den Motor Knopf, wird der Antrieb zwar gestoppt, jedoch besteht weiterhin ein Drehmoment, weshalb sich die Abspielgeschwindigkeit nur durch Reibung verringert. Es ist hierbei zu bemerken, dass durch die verschiedenen Arten der Antriebsmechanismen, unterschiedliche Faktoren der Reibung eintreten. Deshalb befindet sich ein Friction[32]–Schubregler auf der rechten Seite, welcher die Reibung bei Inaktivität des Motors regelt.

[32]Engl. für Reibung.

Ein Pitch–Regler befindet sich ebendort und hat wie sein Vorbild einen anwählbaren Bereich und verändert wie dieser die Tonhöhe durch Modifikation der Abspielgeschwindigkeit. Da jedoch die Abspielgeschwindigkeit nicht durch physikalische Eigenschaften begrenzt ist, kann ein beliebiger Wert als Grenze gesetzt werden. Hier muss man jedoch beachten dass die Qualität im direkten Verhältnis mit dem Modifikationfaktor steht. Je höher die Abspielgeschwindigkeit desto mehr Werte müssen interpoliert werden, desto ungenauer das Ergebnis oder desto höher der Aufwand für eine Rekonstruktion.

Der Lautstärkenregler befindet sich daneben. Es multipliziert das Ausgabesignal, also den Buffer, mit einem Faktor im Bereich von [0,1], wobei 1 das Originalsignal darstellt und 0 absolute Stille. Ein Buffer gefüllt mit 0en bedeutet perfekte Stille ohne Störgeräusche. Dies wird auch beim Pausieren der Platte zum Nutzen gemacht, wobei die Position angehalten wird und die Ausgabe mit einem solchen Array versorgt wird. Man muss jedoch anmerken, dass dieser lineare Regler nicht einer linearen Wahrnehmung der Lautstärke gleichkommt, also der Lautheit. Während die Änderung von 1 auf 0.9 keinen hörbaren Unterschied verursacht, sind in den unteren Regionen Änderungen von 0.05 hörbar. Um diesem entgegen zu wirken muss die Veränderung logarithmiert werden, wodurch eigentlich von einem Lautheitregler gesprochen werden sollte. Ein Crossfader wurde nicht implementiert, da bei einem Abspielgerät kein Bedarf besteht, dies würde aber auf demselben Prinzip basieren.

Der Motor-Schieberegler gibt einfach die gewünschte Beschleunigung an, das heißt es bestimmt wie lange der Motor braucht um die erstrebte Abspielgeschwindigkeit zu erreichen. Erwähnenswert ist hier, dass bei einem Plattenspieler die Drehgeschwindigkeit niemals ganz konstant sein kann, und der Riemenantrieb hier bessere Ergebnisse erzielt als der Direktantrieb. Letzterer ermöglicht jedoch eine schnellere Beschleunigung.

Die Scrollbar unterhalb der virtuellen Platte stellt das Pendant des Queue Sliders aus [Bea03] dar. Jedoch ohne deren haptischen Feedbacks. Er dient zur Navigation durch das Musikstück ohne Tonausgabe.

Des Weiteren gibt es zwei Abspielmodi, wobei Looping das Musikstück endlos wiederholt, und bei Play Once die Wiedergabe nach Ablauf des Stückes angehalten wird.

Weitere Funktionen, eher von geringer Bedeutung, sind *Fastforward* (FFW), *Fastrewind* (FRW), und *Rewind* (RW). FFW und FRW ermöglichen einfach ein schnelleres Abspielen (2fache Geschwindigkeit) in Abspiel- und gegensätzlicher Richtung samt benötigter Beschleunigung. Dies wurde für erste Versuche implementiert und wurde beibehalten. Der RW-Knopf implementiert eine Reset-Funktionalität, wobei die Abspielposition augenblicklich an den Anfang gesetzt wird und das Musikstück angehalten wird.

Eine weitere erwähnenswerte Funktionalität ist, dass zu Musikdaten, welche das erste mal abgespielt werden, die Onsetdaten gespeichert werden und diese somit nur einmal berechnet werden müssen. Dies geschieht indem eine Datei, lautent auf einen berechneten *Hashcode* der Musikdatei, angelegt wird, welche folglich alle Onsetzeiten beinhaltet.

10.4 Die Umsetzung der auditiven Anforderungen

Bevor ein Sample abgespielt, bzw. manipuliert werden kann muss dieses erst richtig konvertiert und als Buffer bereitgestellt werden. Jedoch sind sämtliche dafür benötigte Informationen in dem Format-Chunk einer Wave Datei enthalten. Dies könnte also

automatisiert erfolgen. Da die Bufferbeschaffung bei JASS durch AudioFileBuffer für kleine Dateien konzipiert ist, muss man die Daten selbst bereitstellen. Hierfür kann Java Sound von Nutzen sein, welches es ermöglicht die Daten mittels der Klasse AudioInputStream stückweise auszulesen. Wir erhalten ein Array von Bytes (bytes[]) nach Instanzierung der Klasse AudioInputStream durch:

Listing 2: Instanzierung von AudioInputStream

```
AudioInputStream audioInputStream
= AudioSystem.getAudioInputStream(filename);
nBytesRead = audioInputStream.read(abData, 0, abData.length);
```

wobei abData ein byte[] Array ist worin die Samples gespeichert werden. 0 steht für den Offset und abData.length ist die Größe des Array, legt also fest wie viele Samples gelesen werden sollen. Das Array abData legt also die erste Buffergröße fest. Der Rückgabewert ist die Anzahl der gelesenen Bytes, oder -1 falls keine Daten mehr vorhanden sind. Unabhängig von der Buffergröße wird immer nur ein integraler Wert von Frames gelesen. Ein Potenzwert der Samplegröße, ausgedrückt in Bytes, multipliziert mit der Kanalanzahl ist daher empfehlenswert.

Würden wir die Daten direkt mit Java Sound abspielen ohne sie zu manipulieren würde man, nach Festlegung des Formats, diesen Buffer an die Soundkarte schicken und währenddessen den nächsten Buffer besorgen. Wir wollen jedoch durch JASS abspielen und die Geschwindigkeit manipulieren. Hierfür müssen alle Daten bezogen und dann in das richtige Format umgewandelt werden.

10.4.1 Die Umwandlung in den richtigen Datentyp

Eine Umwandlung nach *double* oder *float* ist möglich. Bei der Frage des optimalen Datentyps kann keine definitive Antwort gegeben werden [Pfi07]. Beide haben Vor- und Nachteile, die Auswahl muss je nach Bedarf getroffen werden. *Float* benötigt nur die Hälfte des Speicherplatzes: 4 Bytes pro Sample statt der 8 Bytes bei *Double*. Berechnungen können hierdurch bei *Float* schneller sein. Bei Pentium Prozessoren kommt es zu keinem Leistungsschub, da sowohl *float* als auch *double* intern per 80 Bit Repräsentation verarbeitet werden. Jedoch sind hier manche *multithreading*[33] Befehle nur bei *float*–Typen möglich. Die benötigte Speicherbandbreite wird halbiert, dies ist also bei Echtzeitsystemen, welche nur eine begrenzte Bandbreite zur Verfügung haben von Bedeutung. Dafür führt der erhöhte Speicherbedarf von *Double* zu weniger Rundungsfehlern, wodurch die Wahrscheinlichkeit einer Instabilität reduziert wird. Einige Algorithmen, wie beispielsweise *Reverb*[34], setzen daher diesen Datentyp voraus. Genau dies ist auch bei einigen mathematischen Funktionen (z.B. sin(), log(), pow()) der Fall, welche *Double* als Parameter und/oder Returnwert verwenden. Ein Mehraufwand entsteht dann bei der Konvertierung der Datentypen.

In vorliegendem Fall ist eine Konvertierung nach *float* von Nöten, da dies wie erwähnt für Echtzeitsysteme geeignet ist und aus diesem Grunde JASS die Daten auch so verarbeitet. Bei der Umwandlung ist darauf zu achten, dass das Format richtig bestimmt und somit das Signal fehlerfrei umgewandelt werden kann. Nimmt man beispielsweise

[33] Die zeitgleiche Abarbeitung mehrerer Threads.
[34] Engl. für Hall, gemeint ist ein künstlich erzeugter Nachhall.

an ein Sample sei 1 Byte groß, während die wahre Größe bei 2 Byte liegt, ist beim Abspielen nur mehr Geräusch zu vernehmen. Nimmt man bei einer Stereoaufnahme an es handele sich um Monodaten, würde sich die Dauer des Stücks verdoppeln und es würde jedes Sample eines Zeitpunkts hintereinander, einmal den linken und einmal den rechten Kanal, auf beiden Kanälen abspielen. Genauso wichtig ist es auf den richtigen Endian zu achten.

Folgender Code stellt die Umwandlung von 16 Bit repräsentiert als Little Endian in das Floatformat, normalisiert auf [-1.0 - +1.0]:

Listing 3: Umwandlung Byte in Float

```
float sample = ((buffer[offset + 0] & 0xFF)
              | (buffer[offset + 1] << 8)) / 32768.0F;
```

Java Sound bietet die Möglichkeit für die Umwandlung benötigten Informationen aus dem Format Chunk auszulesen:

Listing 4: Format

```
fmt = audioInputStream.getFormat();
fmt.isBigEndian());
fmt.getChannels();
fmt.getFrameSize();
fmt.getFrameRate();
```

Weitere Methoden sind der Java Sound API [SUN] zu entnehmen. In der Umsetzung wurde die Formatkonvertierung von Tritonus [Bom07] verwendet. Für eine bessere Übersicht wird abschließend der Buffer noch auf einen linken und einen rechten Kanal verteilt und jeder einzelne Buffer des Musikstücks in einer *LinkedList* abgelegt. Hierbei handelt es sich um einen Zwischenbuffer. Dieser für die Bereitstellung der Daten dienende Buffer, wird je nach Größe des abzuspielenden Files gewählt. Dies ist von Nöten, da beispielsweise große Dateien einen für die verzögerungsfreie Wiedergabe einen großeren Buffer benötigen, jedoch Dateien die kleiner als der Buffer sind Verzögerungen im Looping Modus erzeugen. Dies hat jedoch keinen Einfluss auf die Latenzzeit, da dies nur als Zwischenbuffer dient und niemals direkt für die Wiedergabe verwendet wird.

10.4.2 Die Wiedergabe der Daten

Jetzt kann man mittels JASS Daten abspielen, man benötigt hierfür die Klasse RTPlay:

Listing 5: Initialisierung für die Wiedergabe

```
rtPlay = new RTPlay(srate,2,bufferSize/2);
pointer = rtPlay.initNativeSound(2, (int)srate, bufferSize/2, 5);
```

Wobei hier ein kleinerer Buffer gewählt werden kann, als bei dem Zwischenbuffer. Man nennt diesen: internen Buffer. Der returnierte Wert ist der C++ Objekt Pointer, dieser wird für die native Methode des Abspielen eines Buffers benötigt:

Listing 6: Buffer abspielen

```
rtPlay.writeNativeSoundFloat(pointer, buf[], buf.length);
```

Bei dem verwendeten Computer (P4, 2.4 GHz, 1 GB RAM, Windows Xp, Soundblaster Live, KxDriver) sind interne Buffer von 512 Samples möglich. Daraus ergibt sich folgende Buffergröße in Sekunden:

$$512 \text{ Samples}/2 \text{ Kanäle}/44100 \text{ Samples pro Sekunde} = 0,0058 \text{ Sekunden}$$

10.4.3 Die Manipulation der Daten

Somit fehlt nur mehr die Resampling Methode, welche die Geschwindigkeitsmanipulation der einzelnen (JASS-)Buffer ermöglicht. Timothy Beamish verwendet für das D'Groove [Bea03] eine in JASS enthaltene Methodik (*LoopBuffer*):

Eine bestimmte Anzahl an Samples, abhängig von der Abspielgeschwindigkeit, wird in einem internen Buffer, fixer Größe N (derzeit 256 Samples jeweils für 2 Kanäle, bei D'Groove 128 Samples in Mono), eingelesen. Soll nun mit 0.7facher Geschwindigkeit abgespielt werden, so werden 89.6 (128 * 0.7) Samples eingelesen. Ein Pointer, welcher auf Anfang und Ende zeigt, ist das Pendant zur Nadel, wenn die vorherige Position 12.3 war, dann entsteht dadurch ein Segment [12.3-101.9]. Dies muss jetzt auf die Buffergröße N approximiert werden, hierfür wird bei Beamish eine lineare Interpolation angewandt. Es ist anzumerken, dass diese Methode der Tempomanipulation zu starken Alias-Effekten führt, deshalb wurde für den Prototypen eine kubische Interpolation gewählt. Abschließend wird der Buffer abgespielt und die neue Anfangsposition der Nadel beträgt nun 101.9. Diese Technik wurde auch in [Doe01a] angewandt. Genauer kann diese Methodik anhand des Sourcecodes von LoopBuffer beschrieben werden. Zunächst muss die Geschwindigkeit festgelegt werden:

Listing 7: Abspielgeschwindigkeit bestimmen

```
public void setSpeed(float speed)
{
  this.speed = speed;
  float tmp = speed * srateRatio;
  dix = (int)tmp;
  dx = tmp - dix;
}
```

Wobei speed die normalisierte Form (unabhängig von der Frequenzrate) der Geschwindigkeit darstellt, wobei 1 der originalen Geschwindigkeit entspricht. Die Variable *dix* bestimmt den Ganzzahlwert der augenblicklichen Geschwindigkeit, ausgedrückt in der Samplerate. Und *dx* stellt den dazugehörigen Fließkommawert dar, der Wertebereich liegt zwischen 0 und 1. Diese Unterteilung ist nötig, da es nicht möglich ist, wie oben beschrieben, 89.6 Samples einzulesen. Ein Sample ist nicht teilbar, es ist eine atomare Einheit.

Folgende Methode wird verwendet, um die manipulierte Form eines Samples zu approximieren. Wobei hier zwecks besserer Verständlichkeit eine lineare Interpolation verwendet wird, welche bei dem Prototypen auf Grund der schlechten Qualität nicht zum Einsatz kommt:

Listing 8: Interpolation

```
protected float getNextSample()
{
  ix += dix;
  x += dx;
  if(x > 1.f)
  {
    x -= 1.f;
    ix++;
  }

  return float val = (1.f - dx) * loopBuffer[ix]
                   + dx * loopBuffer[next_index];
}
```

Die Variablen *ix* und *x* sind der ganzzahlige Wert und der nichtganzzahlige Anteil [0,1] der Position im Buffer, sie werden bei jedem Durchlauf um *dix* respektive *x* erhöht. Falls Variable *ix* den Wert 1 übersteigt muss diese solange um den Faktor 1 dezimiert werden, während in gleichem Maße *x* um selbiges erhöht wird, bis der erlaubte Wertebereich erreicht wird. Mit diesen Werten kann dann ein Wert approximiert werden.

Die Methode computeBuffer() ruft eben erklärte Methode solange auf bis der Buffer angefüllt ist:

Listing 9: Abspielbuffer

```
public void computeBuffer()
{
  int bufsz = getBufferSize();
  for(int k=0;k<bufsz;k++)
  {
     buf[k] = getNextSample();
  }
}
```

Diese Methode geht jedoch davon aus, dass der Buffer gestreamt werden kann. Jedoch bietet weder Java Sound noch JASS eine solche Möglichkeit für große Datenmengen, wie sie bei einem Musikstück vorkommen. Deshalb musste der Zwischenbuffer manuell angelegt werden und genauso muss auch auf ihn zugegriffen werden. Es muss also ein zweiter Pointer eingeführt werden, welcher die Position im Zwischenbuffer bestimmt:

Listing 10: Pointer für die Position im Zwischenbuffer

```
while (ix >= linkedListElementSize)
{
  ix -= linkedListElementSize;
  linkedListZeiger++;
}

bufferLinkedListLeft =
(float[]) myBufferLeft.get(linkedListZaehler);

bufferLinkedListRight =
(float[]) myBufferRight.get(linkedListZaehler);
```

Weiters ist zu beachten, dass bei der im Prototypen angewandten 4-Punkt Interpolation, 4 Punkte für die Approximation nötig sind im Gegensatz zu der linearen Methode, wobei 2 Punkte verwendetet werden.

Hierfür müssen vorherige bzw. folgende Zwischenbuffer geladen werden und ein dritter Zeiger verweist auf das richtige Element (Anfang oder Ende) im benachbarten Buffer.

Listing 11: Pointer für die verschiedenen Szenarien

```
int next_index;
next_index = linkedListElementSize - 1;
```

Außerdem handelt es sich um Monodaten, deshalb wurden die Kanäle im Zwischenbuffer separat abgelegt um die Interpolation nach dieser Methodik zu gewährleisten. Das einzige was sich hierbei ändert ist, dass in getNextSample() beide Kanäle interpoliert werden müssen und das computeBuffer() den internen Buffer in zweier Schritten befüllt (links dann rechts). Mit wenigen Änderungen können auch negative Geschwindigkeiten ermöglicht werden, wobei hier $speed = -1$ in ursprünglicher Geschwindigkeit rückwärts abspielt.

Die verwendete 4-Punkt Interpolation 4. Grades sieht dann folgendermaßen aus:

Listing 12: 4-Punkt Interpolation 4. Grades

```
float z = -x - 1/2.0f;
float even1 =
bufferLinkedListLeft[next_index] + bufferLinkedListLeft[ix];
float odd1 =
bufferLinkedListLeft[next_index] - bufferLinkedListLeft[ix];
float even2 =
bufferLinkedListLeft[next_index - 1] + bufferLinkedListLeft[ix + 1];
float odd2 =
bufferLinkedListLeft[next_index - 1] - bufferLinkedListLeft[ix + 1];
float c0 =
even1 * 0.45645918406487612f + even2 * 0.04354173901996461f;
float c1 =
odd1 * 0.47236675362442071f + odd2 * 0.17686613581136501f;
float c2 =
even1 * -0.25367479420455821f + even2 * 0.25371918651882464f;
float c3 =
odd1 * -0.37917091811631082f + odd2 * 0.11952965967158000f;
float c4 =
even1 * 0.04252164479749607f + even2 * -0.04289144034653719f;

val[0] = (((c4 * z + c3) * z + c2) * z + c1) * z + c0;
```

10.5 Der Scratch Button

Die Bedienung mit der Maus birgt den gewaltigen Nachteil, dass keine genauen Scratchabläufe möglich sind. Deshalb wurde ein Button erstellt, welcher einen Scratch automatisiert durchführt. Vorbild hierfür war der Babyscratch, eine Technik die den Crossfader nicht involviert. Folgende Abbildung 31 visualisiert links die Funktion der Nadelposition und rechts die Funktion der Geschwindigkeit.

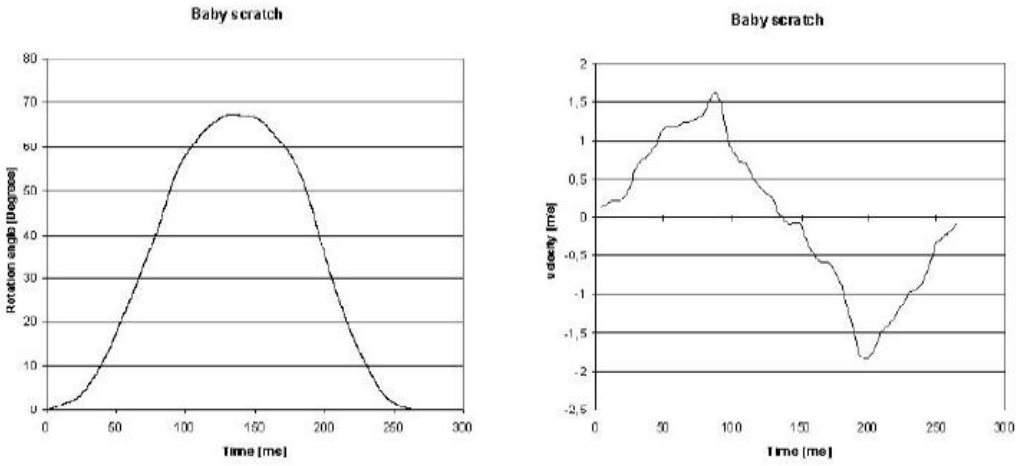


Abbildung 31: Ein Babyscratch [Han03b]

Der Babyscratch stellt eine einfache Vor– und Zurückbewegung der Platte dar. Hierbei ist auffällig, dass die Endposition bei der Vorwärtsbewegung, das lokale Maxima, schneller erreicht wird, als das darauf folgende Minimum bei der Rückwärtsbewegung. Dies liegt an der Tatsache, dass bei der Rückwärtsbewegung der Motor des Plattenspielers dagegen wirkt. Das Geschwindigkeitsmodell weist ein dem *Target Approach Task* [Fla85] ähnlichen Verlauf vor [Han03b].

Als Annäherung wurde eine Sinuskurve gewählt, welche die erwähnten Charakteristika nicht beinhaltet. Da die Buffergröße nicht bekannt ist, handelt es sich um hierbei relative Daten. Daher ist es notwendig drei Stellgrößen in der Konfigurationsdatei festzulegen. Diese dienen aber auch dem Zweck die Eigenschaften des Scratchs frei wählbar zu machen. Es handelt sich um:

1. Die Anzahl der Wiederholungen des Scratchs
2. Die Geschwindigkeit des Scratchs
3. Die Dauer, bzw. die Distanz des Scratchs, welche wiederum abhängig von 2. ist

10.6 Das physikalische Modell

Als ein wichtiger Bestandteil des Prototypen ist das Verhalten bei Änderung der Abspielgeschwindigkeit hervozuheben. Dies kann sowohl durch interne sowie externe Einflüsse geschehen. Bei den internen Ereignissen sind der Anlauf, die Phase in welcher der Motor beschleunigt um die gewünschte Abspielgeschwindigkeit zu erreichen, und der Auslauf, wobei jene Zeit gemeint ist die der Plattenteller benötigt um zum Stillstand zu kommen. Externe Einflüsse können etwa ein widerfahrener Drall sein, welcher die Schallplatte antreibt bzw abstoppt. Bei der traditionellen Form auf der Schallplatte gibt es folgende Faktoren:

- Der Drehimpuls...L
- Die Masse...m
- Das Drehmoment...$\vec{M}$
- Das Trägheitsmoment...I

All dies resultiert in einer Beschleunigung und Geschwindigkeit der Schallplatte. Bei einer Kreisbewegung nennt man diese Winkelbeschleunigung α und Winkelgeschwindigkeit w, diese beträgt zur Abspielzeit $33\frac{1}{3}$ RPM gerundete $3,49 rad/s$:

$$w = 33\frac{1}{3} * 360°/60s = 200°/s \approx 3,49 rad/s$$

Die Winkelbeschleunigung lässt sich wie folgt berechnen:

$$\alpha = \frac{\vec{M}}{I}$$

Und die Winkelgeschwindigkeit mit:

$$w = \frac{L}{I}$$

Wobei sich L aus

$$\Delta L = \alpha * \Delta t * I$$

Berechnet und das Drehmoment $\vec{M}$ aus

$$\vec{M} = F * r_2 * sin(\varphi)$$

Trägheitsmoment I für einen Vollzylinder bzw. Scheibe mit Radius r:

$$I = \frac{1}{2} m * r^2$$

Ein starrer Körper (mit konstantem Trägheitsmoment J) kann, bei Vernachlässigung der Reibung, nur dann eine Winkelbeschleunigung erfahren, wenn ein Drehmoment $\vec{M}$ wirkt [Wik07j]. F ist hierbei die Kraft, r_2 die Distanz des Kontaktes zu dem Mittelpunkt und φ die Richtung in welche die Kraft wirkt. Ein maximales Drehmoment lässt sich durch einen Winkel von 90° erzielen (sin(90) = 1), beziehungsweise für die entgegen gesetzte Richtung 270° (sin(270) = -1). Des Weiteren lässt sich leicht aus der Gleichung ableiten, dass die Entfernung von dem Mittelpunkt möglichst groß zu wählen ist, diese ist jedoch durch die Schallplatte begrenzt (30 cm).

Die Problematik der Steuerung per Maus liegt darin, dass die Druckstärke nicht messbar ist. Ein Mausdruck repräsentiert nur einen binären Wert, somit ist F nicht gegeben. Deshalb konnte nur ein Pseudophysikmodell implementiert werden. Wobei nur zwei Größen die lineare Beschleunigung beeinflussen, einerseits die Motorstärke andererseits die Reibung.

Ein weiterer Aspekt ist, dass bei einem Turntable die Interaktion mit der Abspielgeschwindigkeit intuitiv passiert. Es entwickelt sich schneller ein Gespür für die Manipulation, da bei nicht idealem Winkel ein größerer Widerstand widerfährt. Durch die

feste Verankerung der Platte wird das rhythmische Hin- und Herbewegen vereinfacht, wogegen bei der Maus dies durch ihre relative Position noch weiter erschwert wird. Außerdem ist kein feingranulares Erfassen der Mauseingabe bei einer Abtastrate von 60 s^{-1} (Framerate von Processing) und der schnellen Bewegungen der Maus möglich.

Es wurden schlussendlich zwei Modi implementiert, welche das Verhalten bestmöglich emulieren sollen. Bei beiden wird je nach Mausposition auf der X-Achse ein konstanter Wert zu der aktuellen Geschwindigkeit addiert bzw. subtrahiert. Der Unterschied der zwei Modi liegt jedoch darin, dass bei konstanter Mausposition bei einem die augenblickliche Geschwindigkeit weiterläuft und bei dem anderen die virtuelle Platte angehalten wird.

10.7 Die Installation und die Config Datei

Dieser Prototyp ist unter Microsoft Windows lauffähig, wobei Java Runtime Environment und DirectX benötigt wird. Auf der beiliegenden CD befindet sich ein Ordner „\turntable“, welcher auf der Hauptpartition „C:\“ abgelegt werden muss. In dessen Unterordner „\application.windows“ befindet sich die ausführbare Datei. Vorher muss jedoch die Config Datei, welche sich im Pfad „C:\turntable“ befinden muss, konfiguriert werden. Sie ermöglicht die Auswahl des Musikstücks, sowie andere Parameter die nicht zur Laufzeit angegeben werden können. Es handelt sich hierbei um eine einfache Textdatei, welche aus acht Zeilen besteht. Es ist unbedingt folgende Zeilenreihenfolge einzuhalten:

1. Pfad des Musikstücks. Wichtig ist hierbei, dass keine zurückgelehnten Schrägstriche verwendet werden, wie allgemein üblich. In diesem Pfad wird auch die korrespondierende Onsetsdatei abgelegt. (z.B.: C:/turntable/)
2. Dateiname des Musikstücks. Vorraussetzung: Stereo im Wav–Format (z.B.: musikstueck.wav)
3. Zoomstufe 1, gemessen in der Anzahl der Samples, die eingelesen und zusammengefasst werden sollen. Je größer die Anzahl, desto größer der sichtbare Ausschnitt. (z.B.: 1000)
4. Zoomstufe 2 (siehe Zoomstufe 1)
5. Zoomstufe 3 (siehe Zoomstufe 1)
6. Die Samplingrate (z.B.: 44100)
7. Buffergröße, angegeben in Samples. Ein niedriger Wert ist zu präferieren, kommt es jedoch zu Aussetzern oder sonstigen Störgeräuschen muss der Wert erhöht werden. (z.B.: 512)
8. Scratchmodus (0 für Modus 1, jeder andere Wert für den Modus 2)
9. Die Anzahl der Wiederholungen des Scratchbuttons
10. Die Geschwindigkeit des Scratchbuttons
11. Die Dauer, bzw. die Distanz des Scratchbuttons

Der Quellcode des Prototypen befindet sich im Ordner „\sources“. Für Modifikation und Ausführung der Sourcen mittels Processing muss man zusätzlich zu den oben erwähnten Schritten wie folgt vorzugehen: Zuerst muss Processing Expert installiert werden, für die Entwicklung wurde Version 0135 Beta verwendet. Die benötigten Libraries (auf der CD in „\libraries“) müssen in das Verzeichnis „\libraries“ im Installationspfad von Processing kopiert werden. Des Weiteren muss das Java Development Kit (JDK) installiert werden, verifizierte Versionnummer ist hierbei 1.6.0.03. Jetzt kann Processing gestartet und die Datei „turntable.pde“ im Pfad „C:\turntable“ geöffnet werden. Bevor der Code mittels „Run“ ausgeführt werden kann, muss jedoch unter „Preferences“ der Wert von „Set Maximum available memory to“ auf beispielsweise 1500 erhöht werden.

10.8 Die kreisförmige Darstellung — Vor– und Nachteile

Neben dem offensichtlichen Vorteil, dass die Visualisierung der der Schallplatte ähnelt, und somit eine vertraute Umgebung darstellt, gibt es doch so manche Nachteile und Limitierungen die beachtet werden müssen. Die angestrebte Unterteilung in Vergangenheit und Zukunft, birgt das Problem in sich, dass bei voller Nutzung des Kreises, die Vergangenheit in die Zukunft münden würde und somit in diesem Bereich keine Differenzierung möglich wäre. Diesem kann einerseits durch einen Abstand zwischen diesen Daten oder andererseits durch unterschiedlicher Farbgebung selbiger entgegen gewirkt werden. Eine weitere Variante ist, dass die Vergangenheit fortlaufend ausblendet und somit eine sichtbare Trennung ermöglicht.

Die mehrdimensionalen Visualisierungsformen der Frequenzdomäne sind auf diese Weise nicht abbildbar. Zweidimensionale Darstellungen eines Spektrogramms mit Farbgebung als Maß der Amplitude sind jedoch denkbar. Eine ähnlicher Ansatz ist es die Wellenform je nach Frequenzanteilen eine Farbkomponente zuzuordnen (siehe hierfür [Ric05]). Hierbei sind die Errungenschaften in der Psychoakustik von fundamentaler Bedeutung und birgen das Potential die Informationen nach der Wahrnehmung abzubilden und somit aussdrucksstärker zu gestalten. Endzweck einer solchen Visualisierung sollte die oft erwähnte Möglichkeit sein, ein Maximum der Stuktur eines Musikstücks erahnen zu lassen ohne der Notwendigkeit es vorher gehört zu haben.

Ein weiteres Problem ist, dass je mehr die Visualisierung sich dem Mittelpunkt nähert, desto weniger Platz dort vorhanden ist. Diese Verzerrung ist eine Eigenschaft dieser Darstellungsform und kann nur umgangen werden, indem man nicht zu weit in die Mitte zeichnet.

Da die gesamte Wellenform eines durchschnittlichen Musikstücks in unveränderter Form sich nicht komplett auf den Kreis transponieren lässt, leidet die Übersicht. Eine Möglichkeit um diesem entgegen zu wirken ist die Zoomfunktion. Eine weitere umgesetzte Variante ist der Scrollbalken, welcher die Position innerhalb des Musikstücks wiederspiegelt und durch Fortschritt ein Zeitgefühl vermittelt. Dieser befindet sich jedoch ausserhalb des Kreises, wodurch eine kreisformige Projektion nicht möglich wäre. Dieser Scrollbalken liesse sich jedoch auch kreisförmig an dem äußeren Rand der virtuellen Schallplatte anbringen. Eine horizontale Darstellung hätte jedoch wiederum den Vorteil, dass man als deren Hintergrund die Gesamtübersicht der Wellenform darstellen könnte. Eine weitere Variante wäre die von Tue Haste Andersen prolongierte *AudioFish*–Visualisierung auf den Kreis zu übertragen, wobei ein Fokus auf aktuelle Daten

gelegt wird, während die restlichen kleiner dargestellt werden, aber dennoch eine Interpretation des Kontexts zulassen. Des Weiteren ist eine spiralförmige Anordnung der Wellenform denkbar, wobei hier die erwähnte *Fisheye*–Technik noch mehr Vorteile mit sich bringen könnte.

Die Abspielrichtung und die Position des virtuellen Tonabnehmers spielen auch eine wichtige Rolle für die Bedienbarkeit. Ist der Tonabnehmer, welcher die aktuelle Abspielposition markiert, wie bei vorliegendem Prototypen bei 315° angebracht und ist die Abspielrichtung dem Uhrzeigersinn entsprechend, so kommt dies dem analogen Pendant gleich. Dennoch ist diese Darstellungsform gewöhnungsbedürftig, da sich die zukünftigen Daten links von der aktuellen Position befinden, im Gegensatz zu der orthogonalen, also gewohnten, Darstellung. Eine Alternative würde eine dem Uhrzeigersinn entgegengesetzte Abspielrichtung mit einer Tonabnehmerposition von 225° bieten. Jedoch fiel die Entscheidung bei dem vorliegendem Prototypen zu Gunsten der dem Plattenspieler entsprechenden Darstellung.

11 Conclusio

Die Einbindung von digitalen Daten bietet zweifelsohne einen großen Vorteil. Man muss jedoch bei der Entwicklung von neuen Interfaces die erweiterten Möglichkeiten mit der resultierenden Akzeptanz der Musiker abwiegen. Man sollte also den Gegebenheiten der Tradition und Aufführungspraxis die größte Bedeutung beimessen.

Besonders im Scratch-DJ Umfeld werden Erweiterungen, als eine Form des Schummelns betrachtet. Deshalb konzentriert sich Lippit nicht auf die Vereinfachung des Prozesses sondern eher auf die Erweiterung, indem er bestimmte Funktionalitäten, beispielsweise Speicherungsmöglichkeiten der Einstellungen, welche im Bereich des Möglichen wären außen vor lässt. Es benötigt vor allem solcher Umsetzungen um das negative Image neuer DJ Interfaces aufzuwerten.

Eine Synergie der zwei Einheiten, Turntable und Mixer, wie von Hansson vorgeschlagen, stellt keinen gravierenden Eingriff in die Aufführungspraxis dar. In Folge kann die Steuerung der Einheiten eine Offenheit gewährleisten, welche neue Techniken in diesen Bereich ermöglichen würde. Dies ist besonders erstrebenswert, da viele Errungenschaften in diesem Bereich und selbst das heute übliche DJ-Setup auf Pionierleistungen der DJs basieren. Erstrebenswert ist hier ein Modell a la Ms Pinky, welches mittels Pure-Data einen modulhaften und somit erweiterbaren Aufbau gewährleistet.

Visuelle Erweiterungen sollten sich zusätzlich zu den bereits erwähnten Aspekten gemäß den Richtlinien von Andersen verhalten. Das Augmented Turntable bietet hier einen vielversprechenden Ansatz, welcher der Anstoss zu dieser Arbeit war. Die Vor- und Nachteile dieser Visualisierungsform wurden bereits in Abschnitt 10.8 beschrieben.

Listings

Abbildungsverzeichnis

Literatur

[Ana07] Anafonesis. Die Stimmanalyse, August 2007. http://www.anafonesis.com/stimmanalyse/index.html.

[And02] Andersen, T.H. and Erleben, K. Sound Interaction by use of Comparative Visual Displays. In *Proceedings of the Second Danish HCI Symposium*, 2002.

[And03] Andersen, T. H. Mixxx: Towards Novel DJ Interfaces. In *Proceedings of the Conference on New Interfaces for Musical Expression*, pages 30–35, 2003.

[And06] Andersen, T. H. and Huber, R. and Kretz, A. and Fjeld, M. Feel the Beat: Direct Manipulation of Sound during Playback. In *Proceedings of the TableTop*, 2006.

[Aud07] Audacity, Oktober 2007. http://audacity.sourceforge.net.

[Bea03] Beamish, T. and Doel, K. v. d. and Maclean, K. and Fels, S. D'Groove: A Haptic Turntable for Digital Audio Control. In *Proceedings of the International Conference on Auditory Display*, 2003.

[Bea04] Beamish, T. and Maclean, K. and Fels, S. Manipulating Music: Multimodal Interaction for DJs. In *Proceedings of the International Conference on Human Factors in Computing Systems*, pages 327–334, 2004.

[Bel07] Bello, J. P. and Daudet, L. and Abdallah, S and Duxbury, C and Davies, M. and Sandler, M. P. A Tutorial on Onset Detection in Music Signals, August 2007. http://www.lam.jussieu.fr/src/Membres/Daudet/Publications_files/Onset_Tutorial.pdf.

[Ber07] Bernsee, S. M. Time Stretching And Pitch Shifting of Audio Signals - An Overview, August 2007. http://www.dspdimension.com/index.html?timepitch.html.

[Bil93] Bilmes, J. Timing is of the Essence: Perceptual and Computational Techniques for Representing, Learning and Reproducing Expressive Timing in Percussive Rhythim. Master's thesis, Massachusetts Institute of Technology, 1993.

[Blu] Blumenberg, J. and Spinnler, M. Energieoptimierung für Gebäude. TU München, Fakultät für Architektur, Fakultät für Maschinenwesen.

[Bom07] Bomers, F. and Pfisterer, M. Tritonus: Open Source Java Sound, August 2007. http://tritonus.org.

[Col05] Collins, N. A Comparison of Sound Onset Detection Algorithms with Emphasis on Psychoacoustically Motivated Detection Functions. In *Audio Engineering Society Convention*, 2005.

[Coo00] Cook, P. Principles for Designing Computer Music Controllers. In *Proceedings of the Conference Conference on Human Factors in Computing Systems*, 2000.

[Dir07] Diracdelta.co.uk. Hamming Window, November 2007. http://www.diracdelta.co.uk/science/source/h/a/hamming%20window/source.html.

[Dix] Dixon, S. and Goebl, W. and Widmer, G. Real Time Tracking and Visualisation of Musical Expression. Austrian Research Institute for Artifical Intelligence.

[Dix06] Dixon, S. Sound onset detection by applying psychoacoustic knowledge. In *Proceedings of the 9th Int. Conference on Digital Audio Effects (DAFx'06)*, 2006.

[Doe] Doel, K. v. d. JASS (Java Audio Synthesis System). http://www.cs.ubc.ca/~kvdoel/jass/.

[Doe01a] Doel, K. v. d. and Kry, P. G. and Pai, D. K. FoleyAutomatic: Physically-based Sound Effects for Interactive Simulation and Animation. In *Computer Graphics (ACM SIGGRAPH 01 Conference Proceedings)*, 2001.

[Doe01b] Doel, K. v. d. and Pai, D. K. JASS: A Java Audio Synthesis System for Programmers. In *Proceedings of the ICAD*, 2001. The JASS toolkit is available at http://www.cs.ubc.ca/ kvdoel/jass.

[Dra07] Drakos, N. The Sampling Theory – E186 Handout, Oktober 2007. http://fourier.eng.hmc.edu/e161/lectures/sampling/sampling.html.

[Dux02] Duxbury, C. and Sandler, M. and Davies, M. A hybrid approach to musical note onset detection. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 33–38, 2002.

[Esp04] Esparza, M. R. R. Java Sound als Plattform für die Entwicklung Studiotauglicher Audioapplikationen. Master's thesis, FH Stuttgart – Hochschule der Medien, 2004.

[FH-07] FH–Wedel. Fourier-Transformation, August 2007. http://www.fh-wedel.de/cis/archiv/seminare/ws0304/sz/grundlagen/wavelet4.htm.

[Fla85] Flash, T. and Hogan, N. The coordination of arm movements: an experimentally confirmed mathematical model. In *The Journal of Neuroscience*, pages 1688–1703, 1985.

[Fry] Fry, B. and Reas, C. Processing. http://www.processing.org/.

[Hac00] Hacker, S. *MP3: The Definitive Guide*. O'Reilly–Verlag, 2000.

[Han00] Hansen, K. F. Turntable Music, 2000. Department of Speech, Music and Hearing, KTH, Schweden.

[Han03a] Hansen, K. F. and Bresin, R. Analysis of a Genuine Scratch Performance. In *Proceedings of the Gesture Workshop*, pages 519–528, 2003.

[Han03b] Hansen, K. F. and Bresin, R. and Högskolan, K. T. Sounding Object, Chapter 11: Complex gestural audio control: the case of scratching, 2003.

[Han05] Hans, M. and Slayden, A. and Smith, M. and Banerjee, B. and Gupta, A. DJammer: a New Digital, Mobile, Virtual, Personal Musical Instrument, 2005. Hewlett–Packard.

[Han06] Hansen, K. F. and Bresin, R. Mapping strategies in DJ scratching. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 188–191, 2006.

[Hei07] Heise.de. Deutsches Musikarchiv: CD-Zerfall bedroht Kulturerbe, Oktober 2007. http://heise.de/newsticker/meldung/85686.

[Jui07] Juillerat, N. and Müller Arisona, S. and Schubiger-Banz, S. Real-Time, Low Latency Audio Processing in Java, 2007.

[Kel07] Kellermeyer, A. and Wimmer, A. Digital Audio: Abtasttheorem, A/D– und D/A–Wandlung, Audioformate (WAV, CD, DAT, Minidisk), Audioplayer, Oktober 2007. http://atknoll1.informatik.tu-muenchen.de:8080/tum6/lectures/seminars/ss03/audio/v1-digitalaudio-2.pdf.

[Kha97] Khazam. Khazam. *The Wire Magazine 160*, page 38, 1997.

[Kla99] Klapuri, A. Sound onset detection by applying psychoacoustic knowledge. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (IVASSP-99)*, pages 115–118, 1999.

[Kri07] Kriss, J. Visual Scratch, August 2007. http://jklabs.net/projects/visualscratch.

[Lag04] Lago, N. The Quest for Low Latency. In *Proceedings of the International Computer Music Conference*, 2004.

[Lev99] Levitin, D. J. and Mathews, M. V. and Maclean, K. The perception of cross-modal simultaneity. In *Proceedings of the International Journal of Computing Anticipatory Systems*, 1999.

[Lev04] Leveau, P. and Daudet, L. and Richard, G. Methodology and tools for the evaluation of automatic onset detection algorithms in music. In *5th International Conference on Music Information Retrieval*, pages 72–75, 2004.

[Lip04] Lippit, T. M. Realtime Sampling System for the Turntablist Version 2: 16padjoystickcontroller. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 211–212, 2004.

[Lip06] Lippit, T. M. Turntable Music in the Digital Era: Designing Alternative Tools for New Turntable Expression. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 71–74, 2006.

[Mar07] Marshall, D. Introduction to Sampling, Oktober 2007. http://www.fortunecity.com/emachines/e11/86/synth5.html.

[Mas96] Masri, P. *Computer Modeling of Sound for Transformation and Synthesis of Musical Signal*. PhD thesis, Univ. of Bristol, Bristol, U.K, 1996.

[MIR07] MIREX. Onset Detection Contest, August 2007. http://www.music-ir.org/mirex2006/index.php/Audio_Onset_Detection.

[Moo95] Moore, B. Hearing, 1995. Academic Press.

[Moo97] Moore, B. and Glasberg, B. and Baer, T. A Model for the Prediction of Thresholds, Loudness, and Partial Loudness. In *Journal of the Audio Engineering Society*, pages 224–240, 1997.

[Nat07] Native Instruments. Traktor DJ, Oktober 2007.

[Nie01] Niemitalo, O. Polynomial Interpolators for High–Quality Resampling of Oversampled Audio, August 2001.

[Num07] Numark. CDX1 CD Player, August 2007. http://numark.de/index.php?option=com_content&task=view&id=149 &Itemid=176.

[Pfi07] Pfisterer, M. and Bomers, F. JavaSound FAQ, Oktober 2007. http://www.jsresources.org/faq_audio.html.

[Pio07] Pioneer. CDJ-1000, August 2007. http://www.pioneerprodj.com.

[Pit07] Pitt, M. Interpolation diskreter Signale, November 2007. http://www.piware.de/docs/siginterp.pdf.

[Puc07] Puckette, M. Max/MSP, Oktober 2007. http://www.cycling74.com/products/maxmsp.

[Ric05] Rice, S. V. Frequency-Based Coloring of the Waveform Display to Facilitate Audio Editing and Retrieval. In *Audio Engineering Society Convention*, 2005.

[Sca] Scavone, G. P. RtAudio. http://ccrma-www.stanford.edu/g̃ary/rtaudio/.

[Sch85] Schloss, A. W. *On the Automatic Transcription of Percussive Music — From Acoustic Signal to High-Level Analysis*. PhD thesis, Dept. Hearing and Speech, Stanford Univ., Stanford, CA, USA, 1985.

[Sch96] Scheirer, E. Tempo and Beat Analysis of Acoustic Musical Signals, 1996. Machine Listening Group, MIT Media Labaratory.

[Sch00] Schriber, P. Das neue Audiospeicherformat MP3. Master's thesis, Institut für Informatik der Universität Zürich, 2000.

[Sta07] Stanton. Final Scratch, August 2007. http://www.finalscratch.com.

[Ste37] Stevens, S. S. and Volkmann, J. and Newman, E. B. A Scale for the measurement of the psychological magnitude of pitch. In *Journal of the Acoustic Society of America*, volume 8, pages 185–190, 1937.

[Ste07] Steinberg. Steinberg Media Technologies, Oktober 2007. http://www.steinberg.net/.

[SUN] SUN. Java Sound API. http://java.sun.com/products/java-media/sound/.

[TU 07] TU Berlin. Digitale Signalverarbeitung, Kapitel „Eigenschaften der zeitdiskreten Fourier–Transformation", Oktober 2007. http://www.kgw.tu-berlin.de/statisch/lehre/skript/ds/node36.html.

[Uns00] Unser, M. Sampling—50 Years after Shannon. In *Proceedings of the (IEEE)*, volume 88, pages 569–587, 2000.

[Vos81] Vos, J. and Rasch, R. The percuptual onset of musical onsets. In *Perception and Psychophysics*, volume 29, pages 323–335, 1981.

[Wei04] Weinmann, N. Audiokodierung, 2004.

[Wik07a] Wikipedia. Containerformat, August 2007. http://de.wikipedia.org/wiki/Containerformat.

[Wik07b] Wikipedia. Datenkompression, August 2007. http://de.wikipedia.org/wiki/Datenkompression.

[Wik07c] Wikipedia. Die Compact Disk, August 2007. http://de.wikipedia.org/wiki/Compact_Disk.

[Wik07d] Wikipedia. E-Musik, August 2007. http://de.wikipedia.org/wiki/E-Musik.

[Wik07e] Wikipedia. Glissando, August 2007. http://de.wikipedia.org/wiki/Glissando.

[Wik07f] Wikipedia. Nyquist-Shannon-Abtasttheorem, August 2007. http://de.wikipedia.org/wiki/Nyquist-Shannon-Abtasttheorem.

[Wik07g] Wikipedia. Open Source, August 2007. http://de.wikipedia.org/wiki/Open_source.

[Wik07h] Wikipedia. Schallplatte, August 2007. http://de.wikipedia.org/wiki/Schallplatte.

[Wik07i] Wikipedia. Spektrogramm, August 2007. http://en.wikipedia.org/wiki/Spectrogram.

[Wik07j] Wikipedia. Winkelgeschwindigkeit, August 2007. http://de.wikipedia.org/wiki/Winkelgeschwindigkeit.

[Wil07] Wilde, A. and Bradley, K. An Analysis of Sample Rate Conversion in Sox, November 2007. http://www.leute.server.de/wilde/resample.html.

[Zwi82] Zwicker, E. *Psychoakustik*. Springer-Verlag, 1982.