Assampler

Henning Thielemann

COLLABORATORS			
	TITLE : Assampler		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Henning Thielemann	January 19, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Assampler

1.1 Assampler

Assampler 00.99

Der etwas eigenartige Soundprozessor

Arbeits(=Energie)erhaltungssatz:

Was man an Denkarbeit einspart,

muß man an Tipparbeit zusetzen.

1.2 Wie man sich dieses Guide erschließt

Wie man diese Beschreibung lesen sollte

Ich habe schon viele Guides gelesen, und versuche nun aus den Fehlern anderer schlau zu werden. Wichtig ist natürlich, von den Benutzern, sprich: Ihnen, Rückmeldung zu erhalten, was Sie unter anderem von dieser Dokumentation halten. Sobald Sie mich irgendwas fragen, was auch in dieser Beschreibung steht, dann nehme ich das als Zeichen, daß der Text entweder unübersichtlich gegliedert oder schwer verständlich geschrieben ist.

Meine Erfahrung ist die, daß oft Hilfetext und Dokumentation identisch sind. Das ist zwar in Ordnung und ich mache es auch nicht anders, nur geht damit oft einher, daß das Guide genauso gegliedert ist, wie die Bedienoberfläche. Das ist aber nur zum Nachschlagen geeignet und hilft wenig beim Neueinstieg. Dummerweise fängt man meistens beim Anfang an.

Deshalb biete ich drei verschiedene Einstiege an, je nachdem zu welcher Gruppe Benutzer Sie sich hingezogen fühlen. Wenn Sie einigermaßen sattelfest sind, mit den Funktionen vertraut sind und sich über die Umständlichkeit mancher Vorgänge geärgert haben, dann sind Sie reif für Optimierungen. Dort werden die Dinge beschrieben, die man anders auch schneller erreicht.

Wollen Sie kompiziertere selbsterstellte Klangalgorithmen abspeichern und später wiederverwenden oder auch weitergeben wollen, lesen Sie unbedingt meine Richtlinien, die helfen sollen, Konflikte mit späteren Programmversionen oder anderen Konfigurationen zu verhindern.

Workshops direkt habe ich nicht, ich hoffe, daß das die Erklärungen zu den mitgelieferten Algorithmen ausgleichen.

Ich versuche außerdem darauf einzugehen, was mein Programm (noch) nicht kann, damit Sie sich nicht tot suchen. Denn oft bin ich bei anderen Programmen an Stellen gelangt, wo es auf der Hand lag, noch einen Schritt weiterzugehen, und man sich nicht ganz sicher sein konnte, ob es da wirklich nicht mehr weitergeht, oder ob man nur noch nicht richtig gesucht hat.

Sollte einmal etwas nicht funktionieren (und das wird wahrscheinlich schneller passieren, als es Ihnen wie auch mir lieb ist), befragen Sie den Abschnitt Fehlersuche. Sollte das keine Klärung bringen und ihr Problem hat etwas mit einer bestimmten Soundklasse zu tun (z.B. Sample-Sound oder Oszillator), dann schauen Sie in die Beschreibung der entsprechenden Soundklasse. Finden Sie dort auch nichts, konsultieren Sie die entsprechende Überklasse, zu welcher sich am Anfang jeder Sound-Beschreibung ein Link befindet.

1.3 Einleitung

Einleitung

Zuerst ein paar Worte zu dem eigentümlichen Namen des Programms, der sicher noch einige Kritiker auf den Plan rufen wird, weil er ziemlich verwirrend ist. Es ist beabsichtigt, daß "Assampler" wie "Assembler" klingt, sind doch alle Berechnungsroutinen in MC68000-Assembler (praktisch in Maschinensprache) verfaßt. Das erklärt die unerhörte Geschwindigkeit, die sich vor allem auf minderbemittelten Amigas auszahlt.

In den Namen habe ich geschickt das englische Wort sample (zu deutsch Probe) eingebettet, was sich im Computersprachgebrauch schon zu Audio-Sample, also digitalisierten Schallsignalen verselbständigt hat.

Eine weitere Besonderheit des Assamplers läßt sich in seinen Namen hineindeuten: Man kann aus Prozessen Algorithmen zusammensetzen (engl. assemble), während sich Algorithmen selbst wieder als Bausteine noch größerer Einheiten verwenden lassen.

Und nicht zuletzt soll man mit Assampler in Zukunft auch Digitalisieren können, d.h. man kann es "as sampler" verwenden.

Ganz schön konstruiert, wie? Wer noch mehr Interpretationen parat hat, die ich übersehen habe, soll nicht zögern, sie mir zukommen zu lassen.

Gedacht ist das Programm zur Synthese von Klängen. Wir alle haben uns an Synthesizer-Musik gewöhnt, und ich frage mich, wieso sich so wenig Leute fragen, wie sie entsteht. Die Verfahren mit denen Synthesizer Klänge erzeugen, kann man im Computer simulieren. Gut, zum Simulieren könnte man auch Computerprogramme zur Simulation elektrischer Schaltungen einsetzen, was aber kriechend langsam wäre.

Bei einem Sound-Synthese-Programm geht man vielmehr von den gleichen mathematischen Überlegungen aus, die zu den Analogschaltungen im Synthesizer geführt haben, um sie dann auf Digitalrechnern umzusetzen. Das erlaubt übrigens mehr Möglichkeiten als sie konventionelle Synthesizer bieten, hat aber den Nachteil, das es auf normalen Rechnern nicht immer in Echtzeit geht. Allerdings muß ich einräumen, daß die Ur-Synthesizer (analog arbeitend) kaum noch zu haben sind, die meisten heute erwerbbaren Synthesizer arbeiten auf Digitalbasis. Den Geschwindigkeitsvorteil haben diese Geräte gegenüber dem Heimcomputer, weil sie auf Klangerzeugung spezialisiert sind.

Woran viele zuerst denken, wenn sie "Sound-Programm" hören, nämlich komplette Musikstücke zu erstellen, muß ihnen Assampler leider (noch) schuldig bleiben. Vielmehr liefert dieses Synthesizerprogramm Sample-Sounds ab, die als Instrumente in Musikprogrammen (den zahlreichen Sound-, Noise-, Star-, Pro- usw. trackern) weiterverwendet werden können.

Nun fragt sich der eine oder andere, ob es neben dem Programmnamen noch Neuerungen gibt. Irgendwo muß das Programm schließlich seine Existenzberechtigung haben. Manchmal zweifle auch ich daran, ob es bei so vielen Konkurrenten noch Sinn hat ein weiteres Gefährt ins Rennen zu schicken. Immerhin gibt es schon einige schnellste, stabilste, mächtigste, kurz beste Soundprogramme. Da ist es schwer einen eigenen Platz zu finden und auch noch den Verdacht des Abkupferns abschütteln zu können.

Selbst für die spezielle Zielstellung Sounds zu synthetisieren, gibt es eine Reihe anderer Programme, weshalb mich der Verdacht beschleicht, daß die Leute, die solche komplexen Programme noch ausreizen können, gerade die sind, die bereits eigene Systeme dafür entwickelt haben.

Wer Programme wie SoundFX, AudioLab oder Wavetracer gesehen hat, für den ist klar, daß die Entwicklung eines Synthesizerprogramms hauptsächlich in der Erweiterung der Operatorenpalette liegt. Aber der Schein, Sie werden es ahnen, trügt.

Angefangen hat die Geschichte von Assampler 1992 als GFA-Basic-Programm ebenfalls mit der synthesizer.library in Assembler. Auch da konzentrierte sich meine Arbeit auf das Kreieren von immer mehr Effekt-Funktionen. Seit dem Neubeginn mit Cluster als Programmiersprache und MUI als Bedienoberfläche änderte sich das. Ich habe diesmal sehr viel länger damit zugebracht, das richtige Umfeld für die Funktionen zu entwerfen.

Und meine Philosophie heißt jetzt - täterätä - Weniger ist oft mehr. Man muß eben nicht jede erdenkbare Funktion "fest verdrahten". Es ist viel lehrreicher und flexibler, wenn der Benutzer selber in die Vorgänge eingreifen kann. Also kommt ein Grundstock von Funktionen zum Einsatz, aus dem komplexere Strukturen aufgebaut werden können. Nennen Sie das Baukastenprinzip oder Modularität, meinetwegen auch objektorientiert, aber bitte auf keinen Fall, verstehen Sie, auf gar keinen Fall Multimedia. Brrr ich hasse dieses Wort, seit es von den PC-Werbefuzzis gepachtet wurde.

Um Zeit und Nerven zu sparen, habe ich alle Möglichkeiten ergriffen, die Programmierung eigener Treiber für Soundkarten oder exotische Dateiformate zu umgehen, da das in meinen Augen uneffiziente und damit Amiga-untypische Programmierung wäre. Ich bin zum Glück nicht monopolistisch veranlagt, von wegen "alles aus einer Hand" und so ein Plunder, deswegen kommen für diese Zwecke AHI und datatypes zum Einsatz.

Vergessen Sie alle Neuerungen a la "verwaltet doppelt so viele Sounds", "erlaubt dreimal so viele Kanäle", oder "viermal mehr Treiber als die Konkurrenz". Hier geht es an die Substanz, alles klar?

Ich möchte im folgenden an einem Beispiel verdeutlichen, wie man an meinem Programm verzweifeln kann, wenn man sich das Konzept desselben noch nicht zu eigen gemacht hat.

Wir wollen einen digitalisierten Klang nachträglich maximal aussteuern, anders gesprochen: die lauteste Stelle soll gerade den zur Verfügung stehenden Lautstärke-Bereich ausreizen. Die Qualität wird dadurch zwar nicht besser, aber nachfolgende Berechnungen können dann mit kleinerem relativen Rundungsfehler durchgeführt werden.

Das Einladen des Sample-Sounds wird Ihnen keine Schwierigkeiten bereiten. Dann kennen Sie aus anderen Programmen vielleicht eine Funktion "Maximieren", die dort die gestellte Aufgabe erfüllt. Sie werden sich durch einige Fenster, Register und Menüs hangeln, um vielleicht dahinter zu kommen, daß in dem Listtree (Kasten der eine Baumstruktur anzeigt) eines Fensters unter "Processes" verschiedene Funktionen aufgelistet sind und das auch noch übel in Kategorien unterteilt. Na gut, eine Funktion "Maximieren" gibt's jedenfalls nicht.

Beim Durchstöbern der Funktionsnamen wird Ihnen "Verstärken" auffallen. Na gut, vielleicht erledigt der Verstärker auch solche Spezialaufgaben. Fehlanzeige, nur ein paar Stringgadgets, ein paar Knöpfe, das war's. Bei den anderen Funktionen der gleichen Kategorie das gleiche Bild. Kann man dieses Programm, daß sich so kompliziert bedienen läßt, nicht mal für solch simple Aufgaben einsetzen?

Bitte noch nicht aufgeben! Klicken Sie im Listtree doppelt auf "Prozeß/Bearbeiter/Y Werte/linear/Verstärken". Im erscheinenden Fenster bitte über dem Volume-Stringgadget die rechte Maustaste drücken - es erscheint ein Kontext-Menü. Unter dem Punkt "Vorgabe" wählen Sie ".Input.Volume/.Input.MaxVol" aus. Geht Ihnen ein Licht auf?

Im Volume-Stringgadget wird immer der Verstärkungsfaktor angegeben (1 = keine Änderung). Die größtmögliche Lautstärke eines Sample-Sound überhaupt ist der Wert der in seiner Variablen Volume enthalten ist. Der absolut größte Wert (der an der lautesten Stelle) eines Klanges kann über MaxVol ermittelt werden. Verstärkt man diesen Sample-Sound um Volume/MaxVol, ist der größte Wert danach MaxVol * (Volume/MaxVol) = Volume. Damit hätten wir den Sound auf maximale Aussteuerung (Lautstärke, Volume) gebracht.

Das ist zunächst einfach nur komplizierter als in anderen Programmen gelöst. Aber es ist flexibler, denn: Sie müssen nicht immer auf volle Lautstärke verstärken, nehmen Sie 0.5/MaxVol um auf halbe Aussteuerung zu verstärken, nehmen Sie 0.2/AvrgVol um die Verstärkung von der wirklich empfundenen Lautstärke abhängig zu machen, oder nehmen Sie 2/pi/AvrgVol, um dabei Sinuswellen auf maximale Aussteuerung zu bringen. Na, ist das ein Fortschritt oder doch? Gewöhnungsbedürftig - akzeptiert. Aber flexibler, ätsch!

Ein Beispiel noch. Wir möchten die leisen Stellen eines Sample-Sounds noch mehr abschwächen, weil dort vermutlich sowieso nichts wichtiges zu hören ist, und das Rauschen, das dort besonders zum Vorschein tritt, damit unterdrückt werden kann. Gehen wir mal analytisch vor: Zuerst müssen wir wohl die leisen Stellen finden und dann müssen wir sie abschwächen. Um leise Stellen ausfindig zu machen bräuchten wir den Lautstärkeverlauf des Sample-Sounds. Dazu bietet es sich an, den Sample-Sound

gleichzurichten (d.h. die absoluten Beträge aller Sample-Werte zu berechnen) und das Ergebnis mit einem Tiefpass zu glätten. Wenn Sie sich vorstellen, daß ein Tiefpass fortlaufend den Mittelwert bildet, erkennen Sie, daß sich die resultierende Kurve etwa durch die Mitte des Verlaufs der Maxima des gleichgerichteten Signals ziehen wird. Je nach Tiefpass-Frequenz reagiert diese Kurve langsamer oder schneller auf plötzliche Änderungen.

Jetzt wollen wir im Lautstärke-Verlauf die leisen Stellen suchen. Für so etwas bietet sich der Trigger an. Damit können wir den Unterschied zwischen lauten und leisen Stellen verschärfen. Als Ausgangswerte nehmen wir mal FillL=0 und FillH=1 um den Wertebereich maximal auszunutzen. Bei den Kippwerten orientieren Sie sich am besten an der Durchschnittslautstärke (AvrgVol), setzen Sie ThresholdHL leicht darunter und ThresholdLH leicht darüber, Herumprobieren wird Ihnen wohl nicht erspart bleiben.

Das getriggerte Signal ist zwar vom Prinzip als Hüllkurve für unser Originalsignal geeignet, aber noch etwas scharfkantig. Wir hätten zwar jetzt absolute Ruhe an den zuvor leisen Stellen, aber an den Übergängen von laut zu leise könnte es knacken. Vorsichtshalber werden wir also das getriggerte Signal noch einmal durch einen Tiefpass jagen. Aber dann können wir endlich die gewonnene Kurve als Hüllkurve auf unser Originalsignal prägen.

Wieder einmal ist es von großem Vorteil, daß wir für unsere Aufgabe keine BlackBox haben, auf deren Funktionieren wir blind vertrauen müssen. Wir können jederzeit Tiefpass-Grenzfrequenz und Triggerverhalten ändern, wenn es die Umstände erfordern. Das besondere an der modularen Lösung ist aber, daß man auch ganze Teile des Algorithmus austauschen kann. Wäre es Ihnen lieber, auf eine andere Weise den Lautstärkeverlauf des Originalsignals zu ermitteln, ist das kein Problem. Wollen Sie alles von der Lautstärke eines Frequenzanteils abhängig machen, schalten Sie einen Bandpass vor den Gleichrichter. Wollen Sie die leisen Stellen nur dämpfen aber nicht komplett eliminieren, ersetzen Sie den Verstärker durch ein Tiefpass, usw. usf.

So, genug eingeführt. Ich hoffe, ich konnte Sie ein bißchen neugierig machen. Wo Sie sich schon die Einleitung bis zuletzt durchgelesen haben, habe ich da keine Bedenken.

1.4 Features

Die Features von Assampler im Überblick

MUI

So wie's aussieht, ist Assampler entweder das erste nennenswerte Soundtool mit MUI-Oberfläche, oder wenigstens eines von ganz wenigen. Assampler nutzt so ziemlich jedes Feature von MUI aus:

Sprechblasen-Hilfe, AmigaGuide-Online-Hilfe, Drag&Drop, virtuelle Gruppen, Custom-Layout-Gruppen, GUI-Veränderung zur Laufzeit, Tastaturbedienung, Gadgetinhalte (Konfiguration) abspeichern, und bestimmt habe ich noch etwas vergessen.

Nur eine Sache bleibt mir ein Rätsel: Wie sich gegen MUI eine derart hartnäckige Abneigung bei manchen Amiga-Benutzern festgesetzt hat. Ist es nicht MUI, das dem Begriff objektorientiertes GUI ganz neue Bedeutung eingehaucht hat? MUI ist viel mehr als das fontsensitive GUI, daß auch andere derartige Projekte bieten. Wer die obige Feature-Liste durchsieht und das Programm bedient hat, wird merken, daß für Assampler ein anderes GUI-System gar nicht in Frage kommt.

AHI

Inzwischen der Standard (wird übrigens auch nach neuer Rechtschreibung hinten mit d geschrieben) bei Soundausgabe, darf es natürlich auch bei mir nicht fehlen. Für AHI habe ich dann auch meine bereits funktionierende 14Bit-Stereo-Play-Routine über Bord geworfen, um ohne Handstände Mono/Stereo, 8Bit/14Bit abspielen zu können. Ich gebe mich sogar der Hoffnung hin, daß Assampler auf Soundkarten korrekt Laut gibt - ausprobieren konnte ich es nie.

Modell kontinuierlicher Signale

Alle Berechnungen versuchen sich dem Ideal anzunähern, daß die Klänge in Wirklichkeit kontinuierliche (und nicht quantisierte) Signale wären. Das bedeutet, daß man einen Effekt auch unter verschiedenen Sample-Raten (= Quantisierung) reproduzieren kann.

Um noch näher an die physikalische Realität heranzurücken, werden alle Zahlenwerte (siehe nächster Punkt) mit physikalischen Einheiten erfaßt. Mit dem entsprechenden Wissen (das zum Teil in dieser Dokumentation vermittelt wird) lassen sich damit Maße direkt aus natürlichen Vorgängen (Echos, bewegte Schallquellen, elektronische Schaltungen) übernehmen und diese Vorgänge simulieren.

Formeln

(Fast) Alle Zahleneingaben können über Formeln umschrieben werden, um größtmögliche Flexibilität zu erreichen. Algorithmen können sich damit automatisch neuen Gegebenheiten anpassen. Was eine gelunge Überleitung zum nächsten Punkt ist.

Algorithmen

Prozesse können miteinander verschaltet werden, um größere Vorhaben zu realisieren. Es entfällt das wiederholte stupide Abarbeiten von mehreren Schritten, wenn Änderungen im ersten Schritt ausgetestet werden sollen. Selbstverständlich kann man die Algorithmen auch abspeichern, um nicht nur den ausgerechneten Sample-Sound sondern auch den Weg dorthin vor dem Vergessen zu bewahren. Warum das Zusammenschalten verschiedenartiger Prozesse gelingt, erklärt der folgende Punkt:

objektorientierte Sounds

Klingt toll und zeitgemäß, hat aber trotzdem einen Sinn. Ich möchte den Begriff "Klang" etwas weiter fassen als nur "Audio-Sample". Jede Klangart hat seine Eigenheiten und ist für etwas besonderes geeignet. Soweit sinnvoll lassen sich verschiedenartige Klänge ineinander konvertieren. Berechnungsbasis bleiben dennoch Samples.

Die kühnste Erweiterung ist die Auffassung von Prozessen (Operatoren) als Klänge. Zum besseren Verständnis kann man sich vorstellen, daß ein Prozess so klingt, wie das, was er aus einem anderen Klang fabriziert.

hierarchische Verwaltung der Sounds

Assampler verwaltet nicht einen und nicht zwei Klänge im Speicher, aber auch nicht eine Liste von Klängen, nein! Assampler verwaltet alle Klänge in einer Verzeichnisstruktur, ähnlich dem Amiga-FileSystem. Anlaß ist die Ordnung, die sich damit ins Gewühl bringen läßt. Das eröffnet neue Möglichkeiten: Voreingestellte Wellenformen und Filterkurven haben eigene Verzeichnisse. Die Zwischenablage fristet kein Schattendasein mehr als schwer zugängliches Schattenprojekt, sondern hat ihr spezielles Verzeichnis.

beliebig Stereo, Quadro ...

Bei dieser Verzeichnisstruktur bietet es sich an, auch Stereo-Kanäle als Untersounds einer speziellen Gruppe zu verwalten. Um keine künstliche Beschränkung aufzuerlegen, erübrigt sich das Feilschen um Stereophonie, Quadrophonie oder Surroundklang, denn jetzt kommt Polyphonie. Ok, das ist Blödsinn, Polyphonie ist Mehrklang im Sinne von mehreren Tönen nicht Kanälen. Macht aber nichts. Leider, leider habe ich mir noch nicht die Mühe gemacht, einen Surround-Encoder/Decoder oder einen AHI-Treiber, der das auch wirklich auf getrennten Boxen zum Klingen bringt, zu entwickeln. Aber das Klangmaterial kann man schon erzeugen.

Sounds direkt auf Diskette bearbeiten

Digitalisierte Klänge im Rohformat kann man auch direkt auf einem Datenträger bearbeiten, wenn sie nicht in den Speicher passen.

Sample-Sounds nicht auf zusammenhängende Speicherblöcke angewiesen

Wird Sie wenig jucken, aber ich erwähne es trotzdem. Assampler legt die Sample-Daten nicht am Stück im Speicher ab, sondern verteilt sie gegebenenfalls bunt im Speicher. D.h. Speicherfragmentierung stört auch bei großen Sample-Sounds nicht, Assampler treibt die Fragmentierung aber noch zusätzlich voran.

Berechnung komplett in Maschinensprache

Die eigentlichen Berechnungen in 16 Bit werden durch reine Assembler-Routinen (daher der Name) abgewickelt, weitgehend mit Festkomma-Zahlen. Das wird besonders Besitzer langsamerer Amigas freuen (wie mich, als mein Amiga 500 zu Beginn der Entwicklung noch mit MC68000, 14MHz, 1MB Chip, 5MB Fast, 210MB HD ausgerüstet war). Für Fließkommaberechnungen werden selbstgeschusterte Routinen (68000er Version) oder FPU-Befehle (68882er Version) verwendet. Die FPU führt aber nur in seltenen Fällen zur Geschwindigkeitssteigerung!

Multithreading

Dank dem Amiga-Multitasking und meiner Umsetzung, kann man asynchron zur Berechnung arbeiten. Das heißt, Sie müssen nicht auf das Ende einer Berechnung warten, sondern können schon die nächste Aufgabe ersinnen und in Auftrag geben, Ihr Amiga sieht dann zu, wie und wann er damit fertig wird. Abspielen kann man prinzipiell auch parallel, was aber etwas mehr Rechenleistung voraussetzt. Ein (noch nicht ganz ausgereiftes, arrgh) Lock-System bewahrt Sie währenddessen davor, Dummheiten zu machen, z.B. noch gebrauchte Sounds zu löschen.

Soweit zu dieser Liste. Ziemlich viel für den Anfang, trotzdem bin ich zuversichtlich, daß Ihnen das Programm unfertiger erscheint als alle bisherigen. Dieser Eindruck drängt sich mir jedenfalls von Zeit zu Zeit auf und es wäre auch schlimm, wenn ich nicht mehr wüßte, was ich als nächstes verzapfen könnte.

Vermutlich vermißt der eine oder andere einen ARexx-Port. Ich muß gestehen, mich selbst noch nicht intensiv mit ARexx beschäftigt zu haben und hoffe, daß die meisten Wünsche durch die Klangalgorithmen abgedeckt werden.

1.5 Voraussetzungen

Voraussetzungen zum Betrieb von Assampler

Zunächst benötigen Sie einen IBM-kompatiblen PC mit Pentium 400MHz, 64MB Ram, MS Windows 98, 1GB freie Festplattenkapazität, Maus und Soundkarte im Desktopgehäuse(!). Diese Fußheizung werden Sie brauchen, da Sie jetzt nicht mehr so schnell freiwillig Ihren Amiga verlassen werden. Wenn Ihnen eine billigere Heizung (eine preiswertere kriegen Sie leicht, aber eine billigere?) zur Verfügung steht, können Sie auch diese nutzen.

Assampler ist in den Hardwareanforderungen recht genügsam, dafür schlägt er bei der Software (alles frei zu haben) rabiat zu:

Sie brauchen

- einen genial konzipierten Computer, also einen Amiga

- OS3.0

- allermindestens 2 MB; wenn davon 0.5 MB ChipRam sind, reicht das

- MC68000 reicht, FPU wird nur bei besonders schweren Operationen eingebunden (aber nie benötigt)

- eine Festplatte ist für die erdrückende Menge der benötigten Software unabdingbar

und dann geht die Kramerei auf AmiNet-CDs und im Netz selber los:

- MUI 3.8 (util/libs/mui38usr.lha)

- AHI 4.7 (dev/misc/ahiusr.lha)

- reqtools

empfohlen außerdem:

- datatypes.libraryV45

- SearchGuide (text/hyper/SearchGuide.lha)

1.6 Installation

Wie man Assampler betriebsklar macht

Ich möchte fast sagen: Die Installation geschieht einfach und bequem per Commodore-Installer. Aber ich kann es nicht lassen, zu bemerken, daß der Installer ziemlicher Ausschuß ist. Eine Verbesserung gegenüber Batchdateien ja, aber den selbstgestellten Ansprüchen genügt er nicht. Die Trennung zwischen Einsteiger-, Fortgeschrittenen- und Experten-Installation muß vom Skript-Schreiber übernommen werden, so daß die Auswahl oft unterdrückt wird. Deinstallieren ist gar nicht vorgesehen, der FileRequester viel zu niftlig, das Fenster nicht umgrößerbar, bei Fehlern wird sofort abgebrochen - dem Benutzer wird keine Chance gegeben, die Ursache zu beheben. Und zum Schluß haben wir Programmierer neben ARexx und DOS-Skripts noch eine weitere Skript-Sprache am Hals.

Verbesserungen wenigstens an der Oberfläche erhält man übrigens mit dem Savage-Installer, einem Installer-Ersatz mit MUI-Oberfläche.

Die wahren Probleme beim Installieren von Assampler ergeben sich frühestens im Praxisbetrieb, und ich fürchte, ich kann hier erst weitere Tips geben, nachdem es ordentlich Beschwerdebriefe geregnet hat.

1.7 Nehmen wir mal an, die Dampfmaschine wäre ein Loch ...

Die verschiedenen Einstiegsmöglichkeiten

Ich habe verschiedene Einstiege vorbereitet, die Sie je nach Ihren Erfahrungen im Umgang mit Sound-Programmen in die Arbeitsweise von Assampler anhand von konkreten Beispielen einführt. Man könnte das auch Workshops nennen, aber besondere Tricks und Kniffe werden hier noch nicht vermittelt, es soll ja erstmal eine Einführung sein.

Das erste Vertrautmachen mit Assampler kann entweder

Ausführlich,

Schnell, oder auch

Blitzschnell

erfolgen.

1.8 Ausführlicher Einstieg

Ausführlicher Einstieg

Sie haben noch nie mit einem Soundprogramm gearbeitet und sind nur durch einen dummen Zufall an den Assampler geraten? Mein Beileid, denn für reine Samplebearbeitung ist mein Programm wohl etwas zu kompliziert. Aber es geht natürlich trotzdem.

Schon die Erstellung von Sample-Sounds ist nicht ohne Umschweife zu erklären, deswegen wollen wir uns zunächst mit der Verarbeitung vorliegenden Materials begnügen. Haben Sie etwas mit einem anderen Soundprogramm digitalisiert oder sind über Sample-Sammlungen oder Musikprogramme an Sound-Samples gekommen, dann laden Sie so eine Datei gleich mal ein. Dazu räumen Sie sich das Hauptfenster ins Blickfeld, erkennbar an den senkrechten Knopfleisten. Dort sollten Sie in der Spalte Datei (ganz rechts) einen "Laden"-Knopf finden, den Sie bei dieser Gelegenheit anklicken. Daraufhin erscheint ein Dateirequester, über den Sie Ihre Sample-Datei auswählen können.

Nach der erfolgreichen Auswahl (irgendwann muß Ihnen dabei ein Ok-Knopf untergekommen sein) beginnt im Idealfall irgendein Laufwerk zu schnarren und es baut sich kurz darauf in einem Fenster, im folgenden Anzeigefenster genannt, eine Kurve auf. Diese Kurve stellt den zeitlichen Verlauf der Schwingung der Membran in Ihrem Lautsprecher dar. Auf der Abszisse (x-Achse, waagerecht) werden die Zeit, auf der Ordinate (y-Achse, senkrecht) die Sample-Werte abgetragen.

Als erstes werden wir das tun, wofür die Sound-Samples überhaupt kreiert wurden: Wir hören uns den Klang an. Dazu richten wir unser Augenmerk wieder auf das Hauptfenster und steuern in der linken Spalte den Wiedergabe/Klang-Knopf an. Auf Klick sollte früher oder später Ihr Klang zu hören sein. Dabei können Sie einen Strich verfolgen, der anzeigt, welche Stelle gerade abgespielt wird. Das ist besonders bei Monumentalklängen z.B. von ganzen Musikstücken unheimlich hilfreich. Da man dem Kurvenverlauf nur mit Übung und auch dann noch nicht immer die Abfolge des Musikstückes entnehmen kann, lassen sich anhand des Wiedergabe-Zeigers visuell bestimmte Stellen merken.

Sollten Sie auf die Idee kommen, nur einen Teil des Klanges abspielen zu wollen, müssen sie im Anzeigefenster einen Bereich auswählen. Die Technik ist Ihnen sicher vertraut: Maus zur gewünschten Anfangsposition bewegen, linke Maustaste drücken und halten, währenddessen die Maus zum Ende des beabsichtigten Intervalls verschieben und dann die linke Maustaste wieder loslassen. Wenn jetzt im Anzeigefenster ein Teil der Kurve (besonders aber der Hintergrund) farbig hervorgehoben ist, haben Sie's richtig gemacht. Dann können Sie mit Wiedergabe/Bereich den markierten Bereich abspielen.

Nun ist man ja irgendwie neugierig, was sich in so einem Bereich genau versteckt. Eine Art Mikroskop bietet uns nun die Funktion Anzeige/Bereich. Was bisher als Bereich Teil des Bildes war, wird nun so gestreckt, daß es die ganze Fensterbreite einnimmt. Dadurch werden mehr Details sichtbar und natürlich ist nun der sichtbare Bereich zugleich der markierte. Wollen Sie weiter vergrößern, drücken Sie den Pfeil links neben dem Cyclegadget mit der 2 in der Anzeige-Spalte. Verkleinerung bewerkstelligt man, Sie können es fast erraten, mit dem Knopf rechts daneben. Der Schieber in der Mitte läßt eine Änderung des Faktors zu, um den vergrößert oder verkleinert werden soll.

Sie können übrigens über die Auflösung (äquivalent zur Sample-Frequenz) der Sample-Sounds hinaus vergrößern. Sie erkennen das daran, daß Treppen in der Kurve sichtbar werden. Weiter zu vergrößern bringt dann keine neuen Details, weil einfach nicht mehr gespeichert sind. Um eine einzelne Schwingung fensterfüllend darzustellen ist diese Möglichkeit ganz nützlich. Die größte sinnvolle Vergrößerung, also wenn jedem Sample-Wert genau eine Pixel-Breite zugeordnet ist, erhalten Sie mit Anzeige/1:1.

Als scharfem Beobachter ist Ihnen sicher aufgefallen, daß sich der Balken analog zur Vergrößerung verändert hat. Das sagt nichts über Ihre Lügen-Aktivität aus, gebogen hat sich der Balken ja nicht. Vielmehr kann man an ihm die Proportionen ablesen, wie kurz der besichtigte Teil bezogen auf den ganzen Klang ist und an welcher Stelle er sich befindet. Der Balken kann auch benutzt werden, um den Anzeigeausschnitt zu verschieben. Ziehen Sie ruhig einmal den Balken hin und her, klicken Sie auch mal links oder rechts daneben oder benutzen Sie die Pfeilknöpfe am rechten Ende des Balkens, das sollte alle Möglichkeiten erklären.

Statt mit der Maus den Bereich abzufahren und dabei zu markieren, kann man auch die Marken von anderen Werten ableiten. In der Spalte Bereich finden Sie Funktionen, um z.B. den ganzen Klang zu markieren oder den gerade sichtbaren Bereich.

Sie haben hoffentlich bemerkt welche Logik den drei Knopfleisten Wiedergabe, Anzeige und Bereich innewohnt: "Wiedergabe" spielt ein bestimmtes Stück des Klanges ab, "Anzeige" rückt ein bestimmtes Stück des Klanges ins rechte Licht, sprich: in die Anzeige, "Bereich" markiert ein bestimmtes Stück des Klanges. Ein bestimmtes Stück können dann der ganze Klang, der sichtbare Bereich, der markierte Bereich und in bestimmten Fällen auch andere Anhaltspunkte sein.

Was wir mit den markierten Bereichen bisher angestellt haben, war doch recht mager. Richtig wichtig werden sie erst für die Schnittfunktionen. Wenn Sie einen Sample-Sound mit gesprochenem Text zur Hand haben, wäre das ideal, daran kann man die Schnittfunktionen gut erklären.

Angenommen, Sie haben selber ein Stück Text gesprochen und dann digitalisiert und wollen dem Sprachfetzen jetzt ein bißchen Kosmetik angedeihen lassen. Wissen Sie, daß Sie sich irgendwo versprochen oder zu lange Pausen gemacht haben, spielen Sie zunächst den kompletten Sprach-Sound ab. Merken Sie sich die Stellen, wo Ihr unerwünschtes Geräusch auftritt, wählen Sie gegebenenfalls an der entsprechenden Stelle einen engeren Anzeigeausschnitt. Dann markieren Sie den Bereich mit dem Störgeräusch und spielen diesen noch einmal mit Wiedergabe/Bereich ab, um ganz sicher zu sein, daß Sie nur das Störgeräusch eingefangen haben. Dann entledigen Sie sich dieses Geräusches durch Klick auf Bearbeiten/Löschen. Ähm, das Geräusch ist dann aber wirklich unwiederbringlich weg.

Wenn Sie es noch brauchen oder sich nicht ganz sicher sind, dann empfiehlt sich die Funktion Bearbeiten/Schnitt. Das Geräusch wird dann zwischengelagert und kann mit Bearbeiten/Einschieben wieder an die vorherige Stelle zurückgebracht werden. Man kann die Kombination Schnitt-Einschieben auch gezielt anwenden, um bestimmte Stellen woanders hin zu verschieben. Sie kennen das sicher von Textverarbeitungen her, Sie schneiden ein Stück Text aus und fügen es an einer anderen Stelle wieder ein. Im Assampler gibt es allerdings keinen Cursor, der dem Computer die Einfügeposition mitteilen könnte. Stattdessen wird auch dafür die Bereichsmarkierung verwand. Beim Befehl Einschieben wird das Stück Sample am Anfang der Bereichsmarkierung eingefügt.

Ebenfalls möglich ist das Kopieren von Teilen des Klanges. Dazu wenden Sie die Kombination von Kopie-Einschieben an. Kopie erstellt eine Kopie des markierten Teils des Klanges und legt diese getrennt ab, ohne etwas am Original zu verändern. Auch das ist Ihnen von Textverarbeitungen bestimmt geläufig.

Dann gibt es noch eine Art umgekehrter Schnitt. Mit der Schnitt-Funktion hatten wir einen Bereich ausgeschnitten und zur Seite gelegt, während der Klang vor und nach dem Bereich erhalten blieb. Mit Bearbeiten/Freistellen behält man den markierten Bereich zurück, und entfernt die Daten drumherum. Sinn und Zweck ist das Befreien des Klangs von Pausen zu Beginn und am Ende, wie sie häufig beim Digitalisieren auftreten.

Die Suche nach den Stellen, wo der Klang tatsächlich beginnt und wo er aufhört, kann Ihnen das Programm abnehmen. In der Spalte Bereich findet sich am unteren Ende ein Stringgadget, daß von zwei Pfeilen umgeben ist. Das Stringgadget enthält einen Wert, der angibt, ab welcher Lautstärke ein Klang als vorhanden betrachtet werden soll. Beachten Sie, daß die Lautstärke eines datenbasierten Klanges niemals dessen Volume-Wert überschreiten kann. Mit dem linken Pfeil (der nach rechts zeigt) setzen Sie den Markierungsanfang auf den vermutlichen Geräuschbeginn, mit dem rechten Pfeil entsprechend das Markierungsende auf das Ende des hörbaren Signals.

Am Ende können Sie, und für diesen Fall sollten Sie auch zwischendurch öfter, abspeichern. Erstens kann ich meinem Programm keine 100%ige Absturzsicherheit bescheinigen, zweitens können Sie auch selber ungewollt etwas versaubeuteln und wünschen sich dann, daß es rückgängig zu machen ginge. Entschuldigung, aber für eine Funktion "Rückgängig machen", vielleicht noch mehrstufig, habe ich noch keine Zeit gefunden. Datei/Sichern ermöglicht das Abspeichern, wie schon beim Laden erscheint der Dateirequester, auch hier können Sie wieder einen Dateinamen eingeben und mit Ok bestätigen.

Die meisten eben beschriebenen Funktionen sind übrigens auch im Programm-Menü zu finden.

So, das sollte für den Anfang reichen. Zugegeben, für einen abgehobenen Programmierer ist es immer etwas schwer, sich in die Lage und den Wissensstand eines Neulings zu versetzen. Ich hoffe, ich habe die Gratwanderung zwischen "viel zu schnell und unverständlich knapp" und "gähnend lang und langweilig" halbwegs hinbekommen. Andernfalls sollten Sie mir schreiben, damit der Mißstand behoben werden kann.

Wenn Sie mit Ihrem bis jetzt gewonnenen Wissen ein bißchen an Sample-Sounds herumspielen, können Sie sich bald als Fortgeschrittenen betrachten. In diesem Fall, oder wenn Sie während einer Sitzung total den Durchblick verloren haben, sind Sie reif für die nächste Stufe.

1.9 Schneller Einstieg

Einstieg für Fortgeschrittene

Sie haben den Einstieg für Einsteiger eingängig studiert oder sind Einleitungen zu Sample-Bearbeitungsprogrammen leid? Dann sind Sie hier genau richtig.

Assampler bietet mehr Sound-Typen als nur Sample-Sounds. Mit einem Typ, den Sie besonders im Zusammenhang mit den Prozessen zu schätzen lernen werden, wollen wir beginnen: Den Spline-Sounds. Stören Sie sich nicht daran, daß ich es Sound (=Klang) nenne. Mein Begriff von Klang ist etwas weiter gefaßt, sollte vielleicht besser Signal heißen, denn da erwartet keiner, daß man was hören kann.

Spline-Klänge sind in Assampler Kurven die aus mehreren Intervallen kubischer Funktionen zusammengesetzt sind. An den Stützstellen lassen sich Höhe und Anstieg einstellen. Die Anstiege können auf beiden Seiten eines Stützpunktes unterschiedlich sein, lassen sich aber für glatte (differenzierbare) Übergänge miteinander koppeln. In meiner alten GFA-Basic-Version hatte ich dieses Aufgabengebiet durch einen Freihand-Editiermodus für Sample-Sounds abgedeckt. Hört sich gut an, erforderte aber eine ruhige Hand und eine gut geölte Maus, und es ließen sich darüber hinaus nur schwer Änderungen vornehmen. In der vorliegenden Assampler-Version wurde diese Option komplett den Spline-Sounds geopfert.

Was macht man nun mit diesen Sounds, die so recht keine sind? Ihre Hauptaufgabe erfüllen Sie beim Steuern von Prozessen. Ob Sie den Lautstärkeverlauf ändern (Hüllkurve aufprägen), die Frequenz modulieren oder einen Filter steuern wollen, für all dies ist ein Spline-Sound erste Wahl und überdies speichersparend.

Laden Sie von den mitgelieferten Beispiel-Sounds "ADSR.spline" ein. Auch der klassische ADSR-Generator findet sich nicht festverdrahtet im Assampler, weil ein Spline mit entsprechenden Parametern viel eleganter und flexibler ist. ADSR.spline ist genau so ein Spline.

Entlang der Kurve sind senkrechte Strichlein angebracht. Das sind die Stützstellen, ein Klick auf eine solche, macht ein Kästchen sichtbar, das zur Linken und zur Rechten noch mal je ein Kästchen als Nachbarn hat. Verschieben Sie die Kästchen mit der Maus, werden Sie feststellen, daß das mittlere die Stützstelle festlegt, während die Kästchen an den Seiten die Anstiege repräsentieren. Anstieg links und rechts lassen sich miteinander koppeln oder unabhängig voneinander betreiben. Die Spline-Anzeige besitzt ein kontextabhängiges Menü, d.h man gelangt nur dahin, wenn das Anzeigefenster aktiv ist und man über der Anzeige die rechte Maustaste drückt. Dort findet man unter Stützstelle/gemeinsame Tangente die Einstellung für die gerade gewählte Stützstelle.

Dabei werden Sie im Kontextmenü noch die Punkte "Neue Stützstelle" und "Lösche Stützstelle" gefunden haben, ich glaube dazu muß ich nichts weiter sagen.

Auch die Ansammlung von mehreren Sounds in einer Gruppe wird von Assampler wiederum als Sound gehandhabt. Als brauchbares Analogon kennen Sie das Amiga-FileSystem, auch hier existieren Verzeichnisse und Dateien in der gleichen Umgebung. Trotz gravierender Unterschiede (aus Verzeichnissen kann man keine Daten lesen, Dateien dagegen können keine weiteren Dateien verwalten) kann man auf sie mit dem gleichen Satz an Befehlen zugreifen. Typisch objektorientiert - und entstanden um eine Zeit, als der Begriff noch nicht existierte! Die Analogie Assampler<->DOS wird noch dadurch unterstrichen, daß Gruppen als Verzeichnisse lad- und speicherbar sind. In Assampler erfüllen die Gruppen die gleiche Aufgabe wie die Verzeichnisse unter AmigaDOS: Ordnung halten. Das erfordert aber auch verantwortungsvolles Umgehen damit, schnell kann man mit diesem Mittel genauso die Unübersichtlichkeit steigern.

Folgende Verzeichnisstruktur wird beim Start des Assamplers bereits initialisiert:

Root

User hier sollten alle Sachen zum eben mal schnell

Bearbeiten abgelegt werden, diese Gruppe ist

auch bei Programmstart zu sehen

Projects hier sollten alle größer angelegten Projekte

erstellt werden

Processes hier sollten alle Prozesse plaziert werden

Waveforms hier liegen bereits die wichtigsten Wellenformen,

beliebig erweiterbar,

zur Verwendung im Oszillator

TransferFunctions hier liegen bereits die wichtigsten

Übertragungsfunktionen, können im Spectrum-Filter

verwendet werden

FilterWindows hier sollten Filter-Fenster abgelegt werden,

wie sie vom nichtrekursiven Filter

genutzt werden

Clipboards die Schnittoperationen legen hier die

ausgeschnittenen Stücke ab

Sie sehen: Alles hat seine Ordnung. Jetzt wissen Sie auch, wo Ihre ausgeschnittenen Klangteile landen. Dieses Wissen können Sie durchaus einsetzen, um z.B. ausgeschnittene Dinge wiederzuholen und weiterzuverwenden. So gesehen ist die Schnitt-Funktion Löschen und Freistellen zugleich, je nachdem welchen der resultierenden Klänge man betrachtet.

Jetzt habe ich soviel über Gruppen erzählt, und Sie haben noch keine einzige zu Gesicht bekommen. Im Anzeigefenster befinden sich unter der eigentlichen Anzeige halbwegs im Quadrat angeordnete Knöpfe mit Pfeilen in alle vier Himmelsrichtungen. Mit den oberen Pfeiltasten nach links und nach rechts können Sie zum vorhergehenden oder nachfolgenden Klang einer Gruppe springen. Mit den unteren Pfeiltasten nach oben und unten können Sie die Ebene wechseln. Hoch geht's zur Gruppe, in der sich der aktuelle Klang befindet, runter geht's nur bei Gruppen und da zum ersten Angehörigen derselben. Direkt anwählen kann man einen Klang über das PopUp-Gadget links unter der Anzeige, dort steht auch immer der Name des angezeigten Klanges.

Nun ist es endlich an der Zeit, Ihnen mitzuteilen, wie Sie eigene Klänge erstellen können. Irgendwo auf dem Screen, wo sich das Assampler-GUI ausbreitet, treibt sich das Klassen-Fenster herum, erkenntlich an dem Listtree-Gadget auf der linken Seite. Dort sind alle Soundklassen, die der Assampler anbietet, in einer Baumstruktur untergebracht. In dieser Einleitung sollen nur die grundlegenden Klassen behandelt werden, die weiteren Soundklassen insbesondere die zahlreichen Prozesse erschließen Sie sich am besten anhand des gutsortierten Nachschlageteils.

Legen wir also spaßeshalber eine Untergruppe in der Gruppe User an. Diese Art Gruppe findet man unter Gruppe/Liste. Auch anklicken, bitte! Uns interessiert jetzt nur das Name-Stringgadget. Dort können Sie den Namen für Ihre zukünftige Gruppe (oder auch andere Sounds) eintragen. Bleibt das Gadget leer, bekommt der Sound den Namen seiner Klasse. Dann müssen Sie sich noch entscheiden, wo Ihr neuer Sound entstehen soll. Soll er auf gleicher Ebene, wie der gerade sichtbare Klang angelegt werden, muß das mit (Erzeugen) Geschwister geschehen, soll der Klang in einer Gruppe erscheinen, dann ist (Erzeugen) Kinder anzuwenden. Da wir in der User-Gruppe eine neue Gruppe brauchen, ist für uns das letztere wichtig.

Das Anlegen von Sample- und Spline-Sounds geht ganz ähnlich. "Datenbasierter Klang/digitalisiert/im Speicher" bzw. "Datenbasierter Klang/frei geformt" heraussuchen und die Länge einstellen. Eine Sekunde ist für das folgende Beispiel in Ordnung. Beim Spline erhöhen Sie die Anzahl der Stützstellen auf 3. Dann erzeugen Sie einen Sample- und einen Spline-Sound. Im Spline-Sound verschieben Sie dann den mittleren Stützpunkt ganz nach oben.

Was wir mit dem Krempel anfangen, kommt jetzt: Wir wollen uns an die Prozesse herantasten. Zuerst generieren wir in dem vorhin angelegten Sample-Sound einen Ton. Im Klassenbaum bitte auf "Prozeß/Erzeuger/Tongenerator/einfach" doppelt klicken, also zweimal kurz hintereinander anklicken! Daraufhin erscheint ein Informationsfenster, in dem man die Parameter, die einen Ton ausmachen, festlegen kann. Die wichtigsten davon sind die Frequenz und die Wellenform.

Für die Frequenz lassen Sie am besten den Ausdruck "Tone(c1)" stehen, denn der berechnet die Frequenz des eingestrichenen c. Die Wellenform wird durch einen anderen Klang repräsentiert. Wie oben erwähnt, befinden sich standardmäßig die wichtigsten Wellenformen in der Gruppe Waveforms. Nun müssen Sie noch als Ziel des Oszillators den vorhin angelegten Sample-Sound festlegen. Wählen Sie dazu im Anzeigefenster mit den Pfeiltasten oder direkt den Sample-Sound an. Mit einem Klick auf den Oscillator-Knopf rechts unten im Informationsfenster wird die Berechnung gestartet. Während der Berechnung können Sie weiterarbeiten, erwarten Sie also nicht, daß Sie ein Warte-Mauszeiger davon abhält. Um sich über den Fortgang der Berechnung informieren zu können, habe ich ein Fenster mit einer Fortschritts-Anzeige vorgesehen.

Dann möchten wir dem Ton noch eine Hüllkurve aufprägen. Dafür habe ich Sie vorhin das Spline erstellen lassen. Der benötigte Prozeß ist "Prozeß/Bearbeiter/Y Werte/linear/Verstärken", also ein steuerbarer Verstärker. Bei Envelope (Hüllkurve) stellen Sie das Spline ein. Ihr Anzeigefenster sollte immer noch den Sample-Sound anzeigen. Ist dies der Fall genügt ein Klick auf "Amplifier", wieder rechts unten und die Berechnung beginnt. Ist Sie beendet, können Sie Ihr Werk sowohl visuell als auch akustisch begutachten. Der Sound-Sample sollte einen Ton enthalten, der zu Beginn leise ist, dann lauter wird und am Ende wieder abklingt.

Nun sollten Sie mit dem nötigen Handwerkszeug zum Bedienen des Assamplers ausgerüstet sein. Nach längerer Einarbeitungszeit sollten Sie auch daran denken, Ihre Arbeit zu optimieren.

1.10 Blitzeinstieg

Blitzeinstieg

Interessant, Sie überhaupt in diesem Guide anzutreffen. Für Sie habe ich nur einen wichtigen Tip:

Assampler wurde auf größtmögliche Flexibilität ausgelegt, so daß es möglich ist (sein sollte), alles mögliche mit allem möglichen zu kombinieren. Machen Sie davon Gebrauch und scheuen Sie sich nicht vor ungewöhnlichen Kombinationen. Wenn irgendwas nicht wie erwartet klappt, ist das ein echtes Manko und sollte mir berichtet werden. Allerdings gibt es bestimmte Dinge, die man nicht tun sollte.

1.11 Warum einfach, wenn's auch umständlicher geht

Wie man schneller oder fehlerfreier zum Ziel kommt

Drag&Drop

Drag&Drop beschleunigt die Arbeit erheblich. Fast alles, was im GUI Zeichenketten enthält oder Klänge repräsentiert kann per Maus woanders hin verschoben werden. Die vielfältigen Möglichkeiten lernen Sie am besten durch Ausprobieren.

Alternativ zum Kopieren von Zeichenketten über Drag&Drop kann man auch Patches wie MCP verwenden, welche es erlauben, den Inhalt von Stringgadgets zu kopieren und in andere Stringgadgets zu einzufügen.

Auch Piktogramme der Workbench lassen sich in Anzeigefenster verschieben, die entsprechende Datei wird dann eingeladen.

Klänge erzeugen

Klänge können ganz unkompliziert erstellt werden, wenn man im Klassenbaum auf die gewünschte Klasse doppelt klickt. Dann wird der Klang in einer vorgegebenen Gruppe abgelegt.

Anders beim Erzeugen per Drag&Drop: Sie ziehen einen Eintrag in das Anzeigefenster einer Gruppe und ein neuer Klang wird erzeugt und dort abgelegt.

Klangauswahl

Das Einstellen eines Klanges an einem Klangwahlgadget geht natürlich auch mit Drag&Drop. So lange das Auswahlfenster ausgeklappt ist, ist es auch möglich, die Anfangsbuchstaben des Klanges einzugeben, der Name wird nach Drücken von <ENTER> vervollständigt.

Zoom

Den sichtbaren Ausschnitt eines Sample-Sound wählt man gewöhnlich über die Funktionen im Hauptfenster. Intuitiver ist aber, mit einer gehaltenen Umschalttaste (voreingestellt: Control) auf das Proportionalgadget zu klicken und dann zu ziehen.

Laden

Im Dateiauswahlfenster dürfen beim Laden auch mehre Einträge und Verzeichnisse angewählt werden, welche alle geladen werden.

Sitzung

Sie können eine Sitzung speichern oder auch die voreingestellte Baumstruktur mit vorgegebenen Wellenformen und Übertragungsfunktionen. Dazu speichern Sie einfach die Wurzelgruppe ab und vermerken die Datei in den Einstellungen als Sitzungsstruktur.

Berechnungen auf Gruppen

Sie müssen nicht jede Berechnung einzeln durchführen. Berechnungen an Gruppen werden automatisch für alle enthaltenen Klänge durchgeführt.

So ist es möglich in einem Rutsch alle Klänge einer Gruppe zu verstärken, oder mit Hilfe der Variablen Index ein und dieselbe Berechnung mit verschiedenen Parametern durchzuführen.

Öffentliche Bildschirme

Wenn Sie andere Klangbearbeitungsprogramme in Kombination mit dem Assampler einsetzen wollen, kann es hilfreich sein, mit Assampler und den anderen Programmen (so weit der Platz das zuläßt) auf einem Bildschirm zu arbeiten. Das geht problemlos über die MUI-Einstellungen, nun müssen nur noch die anderen Programme mit öffentlichen Bildschirmen klar kommen. Das tun auch alle modernen Programme - oder andersherum: Die Programme, deren Oberfläche nicht auf öffentlichen Bildschirmen geöffnet werden kann, sind nicht modern :-)

Algorithmen

In aufeinanderfolgenden Berechnungen darf jeder Klang wiederverwendet werden. Der Knackpunkt ist, daß es auch erlaubt ist, wenn vorangehende Berechnungen noch nicht vollendet sind. Beispiel: Sie öffnen die Anzeige eines Sample-Sounds, erzeugen dann einen Verstärker-Prozessen und starten diesen. Noch während der Verstärker seine Arbeit erledigt, können Sie einen anderen Verstärkungsfaktor eintragen und den Verstärker auf einen anderen Sample-Sound anwenden.

Auf genau die gleiche Weise können Sie verschiedene Berechnungen auf einem Sample-Sound hintereinander ausführen. Wenn Sie die Berechnungen schneller starten, als sie ausgeführt werden können, merkt sich Assampler die Reihenfolge.

Soll die gleiche Berechnungsreihenfolge oft wiederholt werden, stellen Sie in jedem beteiligten Schritt als Eingangsklang den jeweils vorangehenden Prozess ein. Dann brauchen Sie für jeden weiteren Versuch nur noch den letzten Prozess zu starten.

Ist es Ihnen dann noch zu umständlich, sich zwischen den vielen Prozessfenstern zu den auszutestenden Werten vorzuarbeiten, verlagern Sie alle Prozesse in einen Algorithmus.

Steuerkurven

Steuerkurven sind meistens tieffrequent, ändern sich also nicht allzu schnell. Deswegen reicht es oft, sie mit geringeren Abtastfrequenzen zu speichern. Das spart Resourcen. Um Treppen in der Kurve zu vermeiden, sollte man hinter so einen Steuer-Sample-Sound immer einen Resample-Prozeß mit linearer Interpolation schalten.

Für die Eingabe von Hand werden Sie aber fast immer Freiform-Kurven verwenden.

Gute Klangqualität

Wie Sie vielleicht bemerkt haben, paßt Assampler die Abtastfrequenzen aller in einen Prozeß hineingehenden Signale an. Die dabei stattfindenden Konvertierungen sind immer mit Verlusten behaftet, auch beim Konvertieren in höhere Abtastfrequenzen!

Wenn Ihr Sample-Material verschiedene Abtastfrequenzen besitzt, sollten Sie es so belassen, und die Sounds den Prozessen so zuführen, wie sie sind. Nur bei den Sample-Sounds für die Zwischen- und Endergebnisse sollten Sie immer gleiche Abtastfrequenzen verwenden. Beachten Sie hierbei, daß aufgrund von Rundungsfehlern beispielsweise 22050 Hz nicht äquivalent zu 22.05 kHz ist!

Ebenso mit Verlusten sind Konvertierungen zwischen Sample-Sounds, Freiform-Kurven und Spectrum-Sounds behaftet.

1.12 Hints and Kunz

Was man noch alles machen kann

Ausschnitt Klang als Wellenform oder Kennlinie

Frequenzmodulation Klang zeitweise anhalten

Impulse Klang von Impulsen variieren

Laden Sample-Sounds aus dem Speicher lesen

Mischen Klang anhand eines Kontrolltones stimmen

Nulldurchgang Frequenzverlauf bestimmen

Oszillator Wellenformen von natürlichen Instrumenten übernehmen

Rauschen Rauschen als Steuerkurve

Sample-Sounds Schwingungsanimation

1.13 Lassen Sie sich nicht aus dem Konzept bringen

Das Konzept

1 Objektorientierte Techniken

2 Signale

2.1 kontinuierliche Funktionen

2.2 Einheiten

2.3 Attribute

2.4 Geschichte

3 Koppelung

3.1 Anpassung von Signal-Attributen

3.2 Entkoppelung von Signal-Attributen

3.3 Erzwingen von Signal-Attributen

4 Variablen

4.1 Typen

4.2 Anwendung

5 Darstellung

5.1 Anzeige

5.2 Information

6 Fäden

1.14 Bevor Sie bei objektorientierten Programmen die Orientierung verlieren

Objektorientierte Techniken

Wenn es bisher eine Programmsparte gab, die von objektorientierter Programmierung (OOP) verschont blieb, dann war es doch wohl der musikalische Bereich. Warum muß man mit Assampler diese letzte Nische beseitigen? Und überhaupt, wo kann man denn bei einer Klangverarbeitung vorteilhaft objektorientierte Techniken einbringen? (Man kann den einleitenden Teil auch überspringen, und kommt dann direkt zur Antwort, was konkret Assampler mit OOP zu tun hat.)

Gehen wir noch einmal zum Urschleim zurück. Systematisierung, d.h. Einteilung in Klassen (was dann zur objektorientierten Programmierung führt) lohnt sich überall dort, wo es gilt, Objekte mit ähnlichen Eigenschaften zu erfassen. Dann sucht man sich die Eigenschaften heraus, die einem am wichtigsten erscheinen und fängt an, nach diesen Gesichtspunkten zu sortieren. Ziel der Aktion ist eine Baumstruktur, in der jede Gabelung einer Klasse entspricht und die Äste von dort zu den Unterklassen führen. Sie kennen solche Schemata vielleicht von der Einteilung von Tieren und Pflanzen. Sie werden immer gerne verwandt, um einen Einstieg in objektorientierte Denkweisen zu vermitteln, ich habe aber die Erfahrung gemacht, daß sie zu weit hergeholt sind, um für die Erklärung der Programmiertechnik geeignet zu sein.

Begeben wir uns mal in unserem Amiga auf die Suche nach Anwendungen von OOP. Da finden wir zum Beispiel AmigaDOS mit seinen Dateien und Verzeichnissen. Allerdings lohnt sich für zwei Klassen der Aufwand für OOP nicht. Betrachtet man noch die unter OS2.0 hinzugekommenen Links (Verweise), steht man vor dem Problem, daß die Entscheidung, wie man den Baum aufbaut, sehr schwer ist. Jede Kombination aus {Datei, Verzeichnis} $\times$ {Objekt selbst, Verweis auf Objekt} (das soll ein Mengenprodukt sein) ist möglich. Wie nimmt man die Einteilung vor,

Objekt

Datei <

/ Verweis

so: DOS-Objekt <

\ Objekt

Verzeichnis <

Verweis

Datei

Objekt <

/ Verzeichnis

oder so: DOS-Objekt < ?

\ Datei

Verweis <

Verzeichnis

Kurz gesprochen, bei AmigaDOS hat man OOP weggelassen, wahrscheinlich auch, weil zu seiner Entstehungszeit OOP noch nicht existierte. (Oder irre ich mich? Jedenfalls kannte es kaum einer, ich zum Beispiel.)

Aber bei graphischen Bedienoberflächen kommt OOP voll zum Zuge. Bestes Beispiel ist MUI (welches auch für die Bedienung von Assampler eingesetzt wird). Sie werden sicher verstehen, daß sich z.B. Knöpfe, die beim Loslassen wieder in den Ausgangszustand fallen (Taster), und welche die in der gewechselten Stellung verharren (Schalter), recht ähnlich sind, jedenfalls ähnlicher als es Knöpfe und Schieberegler sind. Deswegen würden Sie sicher auch Knöpfe in Schalter und Taster einteilen, und Schieberegler als getrennte Klasse betrachten. So in etwa unterteilt auch MUI die Bedienelemente, kennt aber noch viel mehr davon. Bedienelemente bilden schon wieder eine Klasse, und zwar eine Klasse, deren Objekte man sehen kann und die auf Benutzereingaben reagieren. MUI kennt aber auch Klassen, deren Objekte man nicht sehen kann.

Auf der Suche nach OOP wird man ebenfalls bei den Datatypes fündig. Hier erfolgt die Einteilung in erster Ebene in Bilder, Animationen, Klänge, Musikstücke, Texte etc. In der zweiten Ebene erfolgt dann die Einteilung nach Datei-Formaten. Man hätte die erste Ebene auch nach den Datenquellen, also z.B. DOS oder RAM einteilen können, hat man aber nicht, da für die Anwendung der Datatypes das Einsatzgebiet die wichtigere Eigenschaft ist. Oft interessiert sich nämlich ein Programm nur für eine bestimmte Art Daten, z.B. ein Textverarbeitungsprogramm für Texte. Für Assampler sind dementsprechend die Sound-Datatypes am wichtigsten.

Eigentlich könnte man den Datatypes-Baum an der Stelle Sound direkt in Assampler fortsetzen. Eine Klangart ist zum Beispiel der digitalisierte Klang, wie wir ihn aus allen Amiga-Musikprogrammen kennen. Hier fällt es leicht, sich einen Klang vorzustellen. Auch auf den Klang eines Spektrums kann man sich einen Reim machen. Bei den anderen Klassen wie Gruppen und Prozessen wird das schon schwieriger.

OOP setzt im Assampler bei der Aufgabe an, verschiedene Klangverarbeitungen (Prozesse) in einem Algorithmus unterzubringen. Dazu muß ein gemeinsamer Nenner gefunden werden. Dieser Hauptnenner aller Klänge ist ein Datenstrom, der von jedem Klangobjekt bereitgestellt werden kann. Um diesen Datenstrom in Bewegung zu setzen, muß man einen Klang konvertieren. Dieses Konvertieren tritt oft in versteckter Form auf, was beabsichtigt ist, um den intuitiven Umgang zu wahren, den Sie womöglich von anderen Soundprogrammen gewohnt sind.

Über die Konvertierung erreicht man einen großen Teil der Funktionalität von Assampler. Konvertieren Sie zwischen Spektrum und Sample-Sound, führt Assampler Fouriertransformationen aus - eine Aufgabe die in manch anderem Fall schon das ganze Programm ausmacht. Weiter veranlaßt das Konvertieren Sample-Sound -> Spline eine Regression.

Nicht so offensichtlich sind die Konvertierungen bei Prozessen. Wenn Sie im aktuellen Anzeigefenster einen Sample-Sound eingestellt haben und im Informationsfensters eines Oszillators den Startknopf betätigen, starten Sie eine Konvertierung! Sie konvertieren nämlich die Oszillator-Ausgabe in Sample-Daten. Noch verworrener ist es bei modifizierenden Prozessen. Wenn Sie einen Sample-Sound mit einem Verstärker bearbeiten, dann wird wie gehabt die Ausgabe des Verstärkers in den Sound-Sample geschrieben. Der Verstärker bezieht aber schon selber Daten aus dem Sound-Sample, nicht eingestellte Sounds werden bei modifizierenden Prozessen nämlich als der Klang interpretiert, der Ziel der Konvertierung ist. Wollen Sie das Original nicht zerstören, können Sie den Quellklang für den Verstärker auch anders wählen.

1.15 Geben Sie Zeichen

Signale

Signale wollen wir hier im Sinne von Funktionen verstehen. (Oh Schreck, das klingt ja nach Mathematik.) Im Falle von Audiosignalen sind das Funktionen die Zeitwerte auf Auslenkungswerte abbilden.

kontinuierliche Funktionen

Obwohl der Computer immer nur mit diskreten Werten (d.h. jeder Abtastwert hat zeitlich gesehen genau einen Nachfolger und dazwischen ist nichts) rechnen kann, sind alle Berechnungen daran orientiert, was herauskommen würde, wenn man alle Berechnungen an stetigen Funktionen vornehmen und erst am Schluß in diskrete Funktionen umwandeln würde.

Deshalb ist alles darauf ausgelegt, bei unterschiedlichen Abtastraten des Eingangsmaterials zu den gleichen Ergebnissen zu kommen. Das heißt, ein Effekt auf ein 8 kHz-Sample angewandt klingt (im Rahmen der Möglichkeiten) wie wenn der Sample in 22kHz vorläge und dort mit dem Effekt versehen würde. Genausogut kann man mehrere Sample-Sounds unterschiedlicher Abtastfrequenz mischen. Das ist bei Sample-orientierten Programmen nicht selbstverständlich!

Es ist allerdings auch ein schmerzlicher Einschnitt gegenüber diesen Programmen, denn mithin können einzelne Sample-Werte nur noch schwer angesprochen werden, worunter die Genauigkeit leidet.

Einheiten

Signale besitzen entsprechend ihrer Anwendung je eine Einheit für die abzubildenden Werte (Funktionsargumente, x-Werte) und für die abgebildeten Werte (Funktionswerte, y-Werte). Audiosignale bilden Zeit (Sekunde) z.B. auf Spannung (Volt) ab. Ein Volt ist beim Assampler standardmäßig die maximale Auslenkung von Sample-Sounds. Das ist eine willkürliche Festlegung, andere Einheiten bei der Auslenkung z.B. Längen (Meter) oder Winkel (Grad) haben auch ihre Berechtigung.

Die Signaleinheiten können zuweilen auch recht merkwürdig sein, z.B. s->Hz (Zeit wird auf Frequenz abgebildet) für die Steuerkurve, die die Filterfrequenz eines Bandpasses moduliert. Oder auch V->V bei Diodenkennlinien, oder gar Hz->V bei einem Frequenzspektrum.

Sie werden sich während der Einarbeitungszeit wahrscheinlich öfter daran stoßen, daß man einen Sound z.B. nicht sofort als Hüllkurve für einen anderen verwenden kann, weil die daraus resultierende Einheit V^2 keinen Sinn ergibt. Betrachten Sie es weniger als Gängelei, sondern mehr als Hilfestellung. Je weniger Einheiten Sie durch Skalierungsfaktoren der Art "Zieleinheit/Quelleneinheit" künstlich korrigieren müssen, desto näher ist Ihr Klangalgorithmus an der physikalischen Realität.

Wenn Sie versuchen herauszufinden, warum hier oder da Einheiten nicht zusammenpassen, werden Sie einige Zusammenhänge besser verstehen können. Oft hilft es, wenn man gedanklich neue Einheiten (z.B. 1 Sample) einführt, um sich von der Korrektheit einer Formel zu überzeugen. Die Programmierer unter Ihnen werden sicher zustimmen, wenn ich sage, daß das Rechnen mit Einheiten so was wie ein physikalischer TypeCheck ist. (... und die C-Programmierer werden es genau aus diesem Grunde lästig finden.)

Im GUI kommen Sie nur über die Zahlen-Gadgets mit den Einheiten in Berührung.

Attribute

Da intern nur mit Festkomma-Zahlen gearbeitet wird (nämlich mit 16Bit breiten) und davon auch nur endlich viele verwendet werden können, bekommt jedes Signal drei Attribute, welche festlegen, wie der reine Zahlensalat zu interpretieren ist.

Zuerst ist da die maximale Lautstärke, die gleichzeitig auch die absolute Genauigkeit festlegt (nämlich maximale Lautstärke geteilt durch 2^15 = 32768).

Als zweites haben wir die Abtastfrequenz. Sie bestimmt, wie viele Werte pro irgendwas (meistens pro Zeiteinheit) ausgeben werden. Also praktisch, wie nah beieinander die einzelnen Werte aus dem kontinuierlichen Datenstrom herausgepickt wurden.

Und nicht zuletzt ist da der (zeitliche) Beginn des Signals, der auch nicht immer null sein muß.

Geschichte

Die Erfahrungen mit dem Ur-Assampler, wo man die Zahleneingaben sehr maschinennah vornehmen mußte (-32768 bis 32767 für die Auslenkung, Sample-Anzahl für Zeitangaben), waren die, daß einmal für einen Effekt ausgetüftelte Werte bei anderen Samplefrequenzen auf einmal nicht mehr verwendbar waren. Besonders ärgerlich ist das bei einem 10-Kanal-Vocoder dessen Bandpässe erstmal mit brauchbaren Werten dimensioniert werden müssen. Außerdem hat nicht jeder die Zweierpotenzen im Kopf um korrekte Lautstärkeangaben zu machen, und erst recht hat man selten ein Gefühl dafür, wie lange das Abspielen von 10000 Samples bei verschiedenen Abtastraten dauert.

Der Schritt zum Modell von kontinuierlichen Signalen war deshalb unvermeidbar. Lange Zeit hatte ich den Assampler darauf fixiert, alle Klänge als Zeit-Auslenkungs-Funktionen zu betrachten. Das ist auch so ziemlich das äußerste, was man dem Begriff Klang noch zumuten kann. Wegen dieser fixen Einheiten war es am Anfang auch völlig ausreichend komplett einheitlos zu rechnen.

Einheiten für Zahlenwerte hatte ich schon länger im Blick, aber eigentlich wäre das nicht so wichtig gewesen. Nur dachte ich mir, daß wenn einmal Klänge mit Assampler erstellt und abgespeichert sind, ein späteres Ändern aller verwendeten Formeln keinem zuzumuten ist. Deshalb entschloß ich mich, Einheiten schon in die erste Assampler-Version aufzunehmen.

Zunächst existierten Einheiten nur in den Formeln, die Klänge waren weiterhin auf Zeit->Auslenkung fixiert. So wurde beim Auslesen der Werte nur die Einheit überprüft und intern konnte weiter einheitenlos gerechnet werden.

Die Macken dieser Vereinfachung stellten sich bald heraus. War es bei einheitenlosen Signalen noch vertretbar, Signale unterschiedlichster Bedeutung (also egal ob Frequenzspektrum, Übertragungskurve, Steuer-/Hüllkurve) gleichzuschalten, war das mit den Einheiten sehr unschön. Oder würden Sie gerne Werte von einem als Frequenzspektrum verwendeten Sample-Sound ablesen, bei dem die Auslenkung an der Stelle 0.37s z.B. die Amplitude der Frequenz 1274Hz darstellen soll?

Es führte also kein Weg daran vorbei, das Rechnen mit Einheiten weiter ins Innere des Programms zu verlegen. Aber ich denke der Aufwand hat sich gelohnt.

1.16 Koppelung

Koppelung

Anpassung von Signal-Attributen

Wann immer Signale aufeinandertreffen, müssen sie aneinander angepaßt werden. Signale treffen aufeinander sobald sie gemischt werden sollen, ein Signal als Hüllkurve auf ein zweites geprägt wird und so weiter. Dabei werden die Quellen immer an das Ziel angepaßt, und das so früh wie möglich. Kopieren Sie einen 16kHz-Sample-Sound A in einen 22kHz-Sample-Sound B wird A zunächst ausgelesen, mit Zwischenwerten aufgefüllt und damit gestreckt (resample) und das auf die Weise angepaßte Signal landet in B.

Verstärken Sie A und schreiben das Ergebnis nach B, wird erst die Abtastfrequenz von A angepaßt, dann verstärkt und dann nach B geschrieben.

Eigentlich soll dieser ganze Krempel völlig automatisch im Hintergrund ablaufen und Sie nicht belästigen, damit Sie wie mit analogen Signalen arbeiten können. Leider haut das nicht immer so hin und dann müssen Sie helfend eingreifen.

Die Anpassung von Abtastfrequenz und Signalbeginn bereitet recht wenig Schwierigkeiten, aber die Anpassung der Lautstärke hat es in sich. Hier muß man mit den 16 Bit wirtschaften - übersteuert werden soll nichts, die Genauigkeit soll aber auch nicht durch Untersteuerung verloren gehen.

Entkoppelung von Signal-Attributen

Angenommen Sie haben den Sample-Sound A mit einer maximalen Lautstärke von 1V der ein Signal enthält, daß diese Lautstärke voll ausschöpft. Sie wollen nun Sound A mit Prozess B bearbeiten und in den Sound C schreiben, der die maximale Lautstärke 2V besitzt.

Kein Problem, Assampler wird das Signal aus A in das 2V-Gitter umbetten, dann Prozess B darauf anwenden und das Ergebnis nach C schreiben. Fertig.

Vertauschen wir Lautstärke von A und C, A hat jetzt 2V, C nur 1V. B sei ein Prozess (z.B. Filter) der die Lautstärke ohnehin auf die Hälfte drückt. Es würde also alles gerade so passen.

Was passiert aber intern? Das Signal aus A wird zuerst in die 1V-Form gepreßt und muß dazu beschnitten werden. Nun setzt Prozess B an und bearbeitet das verstümmelte Signal, so daß es nochmals leiser wird.

Was ist zu tun? Man muß Quelle und Ziel entkoppeln!

Wie macht man das? Man schaltet einen Verstärker nach den Prozess B. Nun arbeitet B mit der Lautstärke von A, bekommt also das Signal von A unmodifiziert auf den Tisch. Erst das Ergebnis von B wird auf die Ziellautstärke gebracht.

Ein Verstärker mit dem Faktor 1 ist also nicht von vornherein Unsinn!

Auch die anderen beiden Attribute lassen sich entkoppeln. Für die Abtastfrequenz übernimmt das die Frequenzmodulation, für den Signalbeginn erledigt das die Phasenmodulation. Auch andere Prozesse entkoppeln mitunter, was dann in den Beschreibungen zu den Prozessen erwähnt ist.

Wann immer Berechnungen in Einzelschritten funktionieren, als kompakter Algorithmus allerdings nicht mehr, ist das ein deutliches Zeichen für das Versagen der Signalanpassung (meistens der Lautstärke). Greifen Sie dann an verdächtigen Stellen das Signal ab, um herauszufinden, wo das Signal das erste Mal vestümmelt wird und fügen Sie dort entkoppelnde Prozesse ein.

Erzwingung von Signal-Attributen

Leider ist die Sache damit immer noch nicht gegessen. Manche Prozesse können ihre Ausgabe-Lautstärke einfach nicht vorhersagen (das trifft für den Integrator genauso zu, wie für den Universalfilter, wenn er in Resonanz gerät), ihnen muß man mit Verstärkungsfaktoren, die man direkt im Prozeß einstellen kann, nachhelfen.

Selbst wenn die Prozesse glauben, eine Vorhersage über die Lautstärke machen zu können, kann diese immer noch falsch sein. Dann muß man helfend eingreifen und den Prozeß vor seinem Irrtum bewahren. Das bedeutet den Weg gerade anders herum zu gehen, also dem betreffenden Prozess nicht durch Entkoppelung größtmöglichen Handlungsspielraum zu geben, sondern ihn in vorgegebene Bahnen zu zwängen.

Auch hier lautet das Prinzip Nachschalten von geeigneten Prozessen. Mit dem Ausschnitt-Prozess kann man einem Signal nachträglich einen Anfang verpassen. Außerdem enthält dieser Prozess eine Length-Variable, was ihn z.B. zur Anwendung als Wellenform eines Oszillators geradezu prädestiniert. (Boh, Fremdwörter machen doch echt was her, sofern man die richtigen verwendet.) Zur Festlegung der Lautstärke benutzt man den Begrenzer, für die Abtastfrequenz hingegen ist ganz klar die X-Quantisierung Spitzenkandidat.

Zukunft

Wenigstens das Problem der Lautstärke werde ich in ferner Zukunft entschärfen können, in dem ich alle Berechnungen auf Fließkomma-Zahlen umschreibe. Damit die Geschwindigkeit nicht auf der Strecke bleibt, kann das nur eine Lösung für Next-Generation-Amigas sein, deren Prozessoren Fließkommazahlen direkt verarbeiten können (bei PowerPCs ist das zum Beispiel der Fall).

1.17 Variablen

Variablen

Im Assampler finden wir an vielen Stellen (z.B. in Informationsfenstern) Variablen, welche dazu gut sind, Platzhalter für konkrete Zahlen zu sein. Na Sie kennen das schon aus früheren Mathematikstunden. Variablen werden über ihre Namen identifiziert (Groß- und Kleinschreibung werden unterschieden!), wobei leider kaum ein Weg an den englischen Variablenbezeichnungen vorbeiführt, weil die Klangdateien (in welchen auch Variablenbezeichner abgelegt werden) international austauschbar sein sollen.

Typen

Assampler kennt folgende Typen von Variablen, eingeteilt danach, welche Inhalte sie haben können:

Klang

Ein Klang verkörpert durch ein Klangwahl-Gadget.

Formel

Ein mathematischer Ausdruck verkörpert durch ein Zahlen-Gadget.

Schalter

Eine ja/nein Aussage dargestellt durch ein Kästchen, das im ausgewählten Zustand abgehakt oder angekreuzt ist.

Auswahl

Ein Eintrag aus einer vorgegebenen Liste, verkörpert durch ein Cycle-Gadget.

Schieber

Eine ganze Zahl in einem vorgegebenen Intervall, verkörpert durch ein Slider-Gadget.

Anwendung

Die Variablen werden in verschiedener Weise eingesetzt:

Werte

Variablen, die fest zu einem Klang gehören, nenne ich im folgenden Werte oder Klangwerte. Angesehen und verändert werden können diese in den Informationsfenstern der Klänge. Prozesse leben ausschließlich von ihren Werten, während bei den Daten-Klängen die (Sample-)Daten das wesentliche sind.

Der Grundstock an Werten, den jeder Klang mitbringt, kann von Ihnen erweitert werden. Natürlich weiß Assampler mit den von Ihnen angelegten Werten zunächst nichts anzufangen und beachtet sie deswegen nicht. Sie können aber diese zusätzlichen Variablen in die Berechnung der anderen Variablen einbeziehen und damit zum Beispiel einen Prozeß auf Ihre Anwendung anpassen. Weiterhin ist es möglich, einen Algorithmus um benutzerdefinierte Werte zu erweitern, und mit Hilfe von Variablen-Referenzen (siehe unten) von einem Klang aus viele andere zu steuern.

Nicht zuletzt wird es damit möglich, bei bestimmten Klängen fehlende Eigenschaften nachzubilden. Zum Beispiel besitzt eine exponentielle Kurve prinzipiell keine Länge. Während bei einem Sound-Sample nach dem Ende der Daten einfach Schluß ist, könnte man eine exponentielle Kurve theoretisch unendlich lange fortsetzen. Möchte man mit einem Oszillator einen Ton mit exponentieller Wellenform erzeugen, ist man dazu gezwungen, die exponentielle Kurve in der Länge zu beschränken. Fügt man eine Formel-Variable mit dem Namen Length zu einem Exponential-Prozeß hinzu, so kann man diesen Prozeß auch als Wellenform verwenden.

Referenzen

Statt einem Klang-Wert direkt einen Inhalt zuzuweisen, kann man den Inhalt auch von einer anderen Variablen ableiten. Sinn ist, daß Prozesse, die sich in einem Algorithmus befinden, Informationen des Algorithmus in ihre Berechnungen einbeziehen können. Somit ist es möglich zentral von einem Algorithmus aus mehrere Prozesse zu koordinieren. Für gewöhnlich baut man dafür im Algorithmus die wichtigsten Variablen aus den Unterprozessen als benutzerdefinierte Werte ein und wandelt die entsprechenden Werte der Unterprozesse in Referenzen um.

Parameter

Während ein Klang seine Werte freiwillig von woanders her beziehen kann, gibt es auch die Möglichkeit einem Klang von außen bestimmte Parameter zu verordnen. Jeder Parameter ersetzt den gleichnamigen Wert des Klanges. Vorteil ist, daß diese Parameter bei jedem Aufruf eines Klanges verschieden sein können. Aufruf soll bedeuten, daß irgendein Prozeß diesen Klang für seine Berechnung verwendet, genau dort, nämlich über die entsprechenden Klangwahl-Gadgets kann man auch die Parameter, mit denen die eingestellten Klänge eingebunden werden sollen, festgelegen.

Wertereferenzen und Parameter müssen im Typ nicht unbedingt mit den vorgegebenen Klangwerten übereinstimmen. In diesen Fällen werden die Inhalte konvertiert, was allerdings zwischen Klangvariablen und den anderen Typen nicht sinnvoll und daher nicht erlaubt ist. Alle Nicht-Klang-Variablen lassen sich auf Zahlenwerte reduzieren, welche untereinander umwandelbar sind. Die Zahlenwerte ergeben sich als

das Ergebnis der Berechnung bei Formelvariablen,

als 0 für abgeschaltet und 1 für eingeschaltet bei Schaltervariablen,

als Eintragsnummer bei Auswahlvariablen,

als Position des Schiebers bei Schiebervariablen.

Referenzen und Parameter können dazu beitragen, einen Klang-Algorithmus übersichtlich und effizient zu erstellen, man kann mit ihnen aber auch das komplette Gegenteil erreichen. Deshalb überlegen Sie gut, wann sich der Einsatz lohnt.

Sie sollten mit den Referenzen immer in einem überschaubaren Gebiet bleiben, z.B. innerhalb eines Algorithmus. Denn Sie müssen davon ausgehen, daß auf anderen Assampler-Konfigurationen die von weit her geholten Variablen nicht existieren. Am liebsten würde ich schon die Eingabe der Referenzen so einschränken, daß man nichts kurzsichtiges machen kann, nur ist mir das bisher noch nicht gelungen.

Sie brauchen keine Parameter, wenn Sie einen Klang ohnehin nur einmal verwenden oder wenn Sie bei jedem Aufruf sämtliche Original-Werte durch die Parameter unterdrücken. Aber Sie können einen aufwendigen Algorithmus mehrmals mit kleinen Änderungen anwenden und ersparen sich damit das mehrfache Erstellen, des gleichen Algorithmus.

1.18 Nur nicht den Faden verlieren

Paralleles Arbeiten mit Threads

Funktion

Dank des Amiga-Multitaskings und meiner Anwendung, kann man asynchron zu Berechnungen arbeiten. Das heißt, Sie müssen nicht auf das Ende einer Berechnung warten, sondern können schon die nächste Aufgabe ersinnen und in Auftrag geben, Ihr Amiga sieht dann zu, wie und wann er damit fertig wird.

Allerdings benötigt man ein bißchen Training, um dieses Feature ordentlich auszunutzen. So sehr hat man sich daran gewöhnt, daß der Computer nicht gestört werden will, wenn er rechnet und man wartet höflicherweise. Das muß aber nicht sein, denn während das Programm noch mit Hilfe eines Oszillators einen Ton generiert, können Sie schon einen weiteren Prozess (z.B. Filter) kreieren, Werte eingeben und starten.

Neben dem Hauptprogramm arbeiten drei Programm-Fäden, einer für die Soundwiedergabe, einer für das Editieren (sorry, der ist noch in Entwicklung, das folgende ist also teilweise Zukunftsmusik) und einer für Berechnungen. Für die Task-Prioritäten gilt (und so sollte es sich in den Einstellungen wiederfinden) Wiedergabe > Hauptprogramm > Editieren > Berechnung, und zwar aus diesen überlegungen heraus: Die Wiedergabe soll vom Fensterhandling usw. nicht gestört werden, was nämlich zu unschönen Aussetzern führen würde.

Andererseits soll die Arbeit mit der Benutzeroberfläche (welche im Hauptprogramm agiert) nicht vom Herumschnippeln in Klängen oder von Berechnungen beeinträchtigt werden. Wobei man beim Editieren meistens auf das Ergebnis wartet, und Editieren deswegen die anderen Berechnungen aufschiebt. Eigentlich könnte man das Editieren auch im Hauptprogramm erledigen (war auch lange Zeit so), dann wäre aber ein Abbrechen der Aktionen nicht möglich, da die Bedienoberfläche solange brach läge. Besonders beim Schneiden von Sample-Sounds direkt auf Disk werden Sie den Vorteil schnell schätzen lernen.

Im Programm-Fäden-Fenster läßt sich die Arbeit der einzelnen Fäden überwachen und gegebenenfalls abbrechen.

Technik

Beim Multi-Threading ergibt sich das Problem, daß mehrere parallel ablaufende Prozesse auf die gleichen Objekte (nämlich Klänge) zugreifen. Damit sich hier kein Chaos entwickelt, muß man ein paar Sicherheitsmaßnahmen durchführen. Im wesentlichen gibt es zwei Möglichkeiten:

1. Man kopiert alle Daten für die Berechnung.

2. Man läßt die Berechnung auf den Originaldaten laufen, und sorgt dafür, daß jeder weitere Zugriff oder sogar das Löschen der Daten solange unterbunden wird.

Kopieren erlaubt ein vollständiges Abkoppeln der Berechnung vom restlichen Geschehen, braucht aber mehr Zeit zum Starten der Berechnung und mehr Speicher und hat darüber hinaus einen entscheidenden Nachteil: Die Berechnung soll ja Ergebnisse liefern, und es ist problematisch diese wieder ins Hauptprogramm zurückzubekommen. Und wenn ich es schaffen würde, müßte man nach wie vor auf das Ende einer Berechnung warten, um die neue Operation auf deren Ergebnis auszuführen. Warte ich nicht, erhalte ich zwei Ergebnisse, kann aber mit beiden nichts anfangen.

Schützen der beteiligten Objekte geht schneller, kostet weniger Speicher, birgt aber die Gefahr in sich, daß sich das Programm selbst ausschließt. Ein typisches Problem des Multitaskings übrigens, daß sich an folgendem Parade-Beispiel demonstrieren läßt: Zwei Fäden A und B wollen beide sowohl an Objekt C wie auch an Objekt D arbeiten, nur daß sich Faden A zuerst Objekt C und Faden B zuerst Objekt D sichert. Als dann wartet Faden A auf Objekt D, was er zunächst nicht bekommen kann, weil Faden B dieses Objekt schon besitzt, seinerseits aber darauf wartet, daß Faden A Objekt C freigibt, und schon haben wir das Programm mit einer einfachen Aufgabe zum Straucheln gebracht.

Ich denke, dieses Problem habe ich inzwischen ganz gut im Griff. Immer noch störend ist aber, daß nun jeder beteiligte Klang während der Rechenzeit nicht angerührt werden kann. Das ließe sich verschmerzen, sagen Sie vielleicht. Vergessen Sie aber nicht, daß die Prozesse auch Klangobjekte sind, welche vom Zugriffsverbot betroffen sind. Das bedeutet, Sie könnten z.B. nicht einen Filter zweimal direkt hintereinander mit etwas unterschiedlichen Werten starten.

Um dieses Manko zu umgehen, habe ich eine Hybridlösung ersonnen, welche die Daten eines Klangobjektes schützt, während die Werte kopiert werden. Das geht zur Zeit reibungslos, weil die Variablen von den Klangobjekten nicht verändert werden. Allerdings wäre diese Option für Analyseprozesse sehr wichtig, und wird deswegen Gegenstand der Entwicklung folgender Assampler-Versionen sein.

1.19 Kann ich Ihnen helfen, oder ist Ihnen nicht mehr zu helfen?

Probleme mit Assampler

Beim Betrieb von Assampler können einige Probleme auftreten, von denen ich jetzt schon weiß, das es welche sein werden. Deswegen möchte ich hier zeigen, wie Sie künftigen Schwierigkeiten aus dem Weg gehen können, und wie Sie Fehler, wenn Sie denn einmal aufgetreten sind, aufspüren und beseitigen.

1.20 Wanzen entlarven

Assampler tut merkwürdige Dinge, die Sie sich nicht erklären können

Merkwürdigkeiten

Symptom: Assampler ignoriert vorgenommene Einstellungen

Erklärung: In Wirklichkeit ignoriert Assampler Ihre Einstellungen nicht, sondern befolgt Sie oft erst beim Neuaufbau von Fenstern.

Abhilfe: Sind es Anzeige- oder Informationsfenster betreffende Einstellungen, müssen Sie neue entsprechende Fenster öffnen und die alten schließen. Bei manchen Einstellungen kann es sogar nötig sein, Assampler zu beenden und nochmal neu zu starten.

Symptom: Obwohl Assampler nichts zu machen scheint, ist der Rechner deutlich ausgebremst

Erklärung: Wahrscheinlich läuft noch die Wiedergabe eines stillen, unendlich langen Klanges (z.B. Prozess ohne Length).

Abhilfe: Im Hauptfenster oder in den Programmfäden bei Play Stop drücken.

Symptom: Beim Abspielen einer Soundgruppe bleibt Wiedergabe bei einem Prozeß hängen

Erklärung: Diesem Prozeß ist sicher keine Länge zugeordnet worden und gilt daher als endlos.

Abhilfe: Length-Variable erzeugen und mit Formel wie ".Input.Length" oder ".Input.Length+5s" initialisieren.

Symptom: Bei Wiedergabe eines Sounds hört man nichts, obwohl der Wiedergabe-Zeiger läuft

Ursache 1: Die Lautstärke des Sounds ist zu leise. Kommt vor, falls man Sounds von anderen Programmen lädt, die den Lautstärkewert im 8SVX-Format falsch oder gar nicht setzen. Leider hat sich so ein Fehler bei vielen Sound-Datatypes eingebürgert, welche zum Einladen von Fremdformaten benötigt werden. (Dort wird oft $40 statt $10000 als volle Lautstärke gesetzt.)

Abhilfe: Fehler dem Autor des entsprechenden Datatypes mitteilen

Ursache 2: Die maximale Lautstärke in den Einstellungen ist zu groß oder die Einheit falsch.

Ursache 3: Es wurde versucht ein Sample-Sound mit entarteten Schleifen-Grenzen abzuspielen.

Abhilfe: Setze Werte LoopBegin = ".Begin + .Length" sowie LoopLength = "0"

Symptom: Ergebnis einer Berechnung ist ein leerer Sample-Sound

Details: Obwohl als Eingang verwendete Sample-Sounds sichtbar Daten enthalten, ist der Ziel-Sound im Prinzip leer

Erklärung und Abhilfe wie Ursachen 1 und 3 beim vorigen Stichpunkt

Symptom: Bei Wiedergabe eines Sounds ist das Geräusch verzerrt

Ursache 1: Die Lautstärke des Sounds ist zu laut. Kommt vor, falls man Sounds von anderen Programmen lädt, die den Lautstärkewert im 8SVX-Format falsch setzen oder ganz und gar uninitialisiert lassen. Auch das ist mir tatsächlich schon untergekommen.

Ursache 2: Die maximale Lautstärke in den Einstellungen ist zu klein.

Ursache 3: Die Wiedergabe-Abtastfrequenz in den Einstellungen wurde zu klein angesetzt. Unter 8kHz sind die Verzerrungen schon ziemlich kraß.

Symptom: Obwohl alle Einheiten stimmen müßten, erhält man Einheitenfehler

Erklärung: Manche Prozesse (insbesondere die Bearbeiter) betrachten nicht eingestellte Sounds als Platzhalter für den gerade aktuellen Ziel-Sound.

Abhilfe: Diese Prozesse abklemmen oder mit Prozessen verbinden, die einfach ein 0-Signal ausgeben (z.B. Lineare Kurve).

Symptom: Gestartete Berechnung läuft nicht an

Erklärung: Der zu bearbeitende Sound wird wahrscheinlich anderweitig genutzt (z.B. abgespielt), der Berechnungs-Thread wartet nun auf die Freigabe des Klanges.

Abhilfe: Wiedergabe stoppen.

Erklärung: Sie haben vielleicht (aus Versehen) als Zielsound einen Algorithmus ausgewählt, das wird aber nicht unterstützt.

Abhilfe: Aktivieren Sie das Anzeigefenster z.B. eines Sample-Sounds bevor Sie die Berechnung starten.

Symptom: Berechnung liefert unerwartetes Ergebnis

Details: Obwohl alles funktioniert, wenn man die an einem Algorithmus beteiligten Prozesse Schritt für Schritt auf einen Sample-Sound anwendet, funktioniert der Algorithmus als ganzes nicht

Erklärung: Zwischendurch ist ein Klang in der Lautstärke beschnitten worden, der Fehler hat sich fortgepflanzt und tritt zum Schluß in ganz ungewöhnlicher Weise in Erscheinung.

Abhilfe: Versuchen Sie durch Umlagern der Lautstärke-Informationen (in einem Prozeß heruntersetzen, in anderem im gleichen Maße herauf) und durch Entkoppelung den Fehler zu beheben.

Details: Ein Sample-Sound, der nur einen Impuls am Anfang enthält verhält sich nicht so, wie man das von einem Impuls erwarten würde

Erklärung: Die Daten von Datensounds werden vor dem ersten Sample mit dem ersten Sample aufgefüllt, und nach dem letzten Sample mit dem letzten Sample, oder auch einer Schleife, falls vorhanden. Der Impuls am Anfang wird also zu einer konstanten Funktion.

Abhilfe: Plazieren Sie den Impuls einen Deut später (mit Hilfe der Bereichsmarkierung).

Symptom: Bereits nach dem Einladen sind Klänge als modifiziert gekennzeichnet

Erklärung: Klänge, die als Stereo-Kanäle fungieren, können durch die übergeordnete Stereo-Gruppe Standardnamen zugewiesen bekommen

Abhilfe: Den entsprechenden Punkt in den Einstellungen abschalten, oder sich damit abfinden.

Symptom: Das Einladen von Klängen dauert unverhältnismäßig lange

Erklärung: Irgendein Datatype hält hier den Verkehr wahrscheinlich beim Überprüfen seines Formates auf

Abhilfe: Deaktivieren sie den Datatypes-Import in den Einstellungen

Symptom: Assampler (Debug-Version) beendet sich ohne Vorwarnung

Erklärung: Wahrscheinlich hat einer der internen Schutzmechanismen im Assampler angesprochen, der das komplette Programm beendet um schlimmeres zu verhindern. Zugegeben, keine schöne Methode, aber immer noch besser, als wenn er andere Programme in Mitleidenschaft zieht.

Das ganze weist Sie jedenfalls recht sicher auf einen Fehler im Assampler hin, der mir bislang unbekannt war (das dürfte der Großteil aller Fehler sein).

Dann erwartet Sie die unangenehme Aufgabe, mir den Fehler zu berichten.

So findet man Fehler in Berechnungen

Abgreifen von Zwischensignalen

Um Fehlerquellen aufzuspüren, kann man von beteiligten Prozessen Zwischenergebnisse abfragen. Starten Sie einfach nicht den Algorithmus sondern den Prozeß, dessen Ausgangssignal Sie untersuchen wollen. Beachten Sie aber, daß die Zwischensignale andere x- und y-Einheiten als das Endergebnis besitzen können.

Versteckte Übersteuerungen

Falls die automatische Anpassung der Lautstärke zwischen beteiligten Prozessen versagt, kann es passieren, daß ein Algorithmus funktioniert, wenn man seine Prozesse Schritt für Schritt von Hand startet, nicht aber, wenn man den Algorithmus als ganzes benutzt.

Länge des Zielklanges

Die Länge des Klanges, der das Berechnungsergebnis aufnehmen soll, muß von Ihnen vorgegeben werden und wird vom Assampler nicht angetastet. Sollte Ihr Zielklang nach Ende der Berechnung nur einen Teil des erwarteten Geräusches beinhalten oder eine lange Pause am Ende, müssen Sie den Zielklang entsprechend verlängern bzw. verkürzen.

eindeutige Klangnamen

Achten Sie darauf, daß alle Klänge deren Werte Ziel von Referenzen sind, eindeutige Namen besitzen. Haben in einer Gruppe mehrere Klänge den gleichen Namen, wird immer der erste von Ihnen genommen, wenn dieser Name in einem Klangpfad verwendet wird.

1.21 Dinge, die man niemals tut

Von Dingen, die Sie tun können, aber nicht sollten

Vermeiden Sie Kreationen deren Gelingen an seidenen Fäden hängt, sagen wir Rechenfehler ausnutzende Algorithmen. Zum Beispiel berechne man auf zwei rechnerisch unterschiedlichen aber prinzipiell gleichen Wegen einen Klang aus, subtrahiere beide Signale voneinander und verstärke das Ergebnis auf maximale Lautstärke. Für solche Algorithmen kann ich keine Gewährleistung geben, daß das Ergebnis in folgenden Assampler-Versionen reproduzierbar ist.

Betrachten Sie das Begrenzen bei Übersteuerungen als Notlösung, nicht als Effekt. Wenn Sie diesen Verzerrungseffekt bewußt haben wollen, fügen Sie einen Begrenzer ein. Auch wenn er zur Zeit nichts bewirkt, wird er wichtig, wenn Assampler eines fernen Tages mit Fließkommazahlen rechnet. Dann nämlich sind Übersteuerungen kaum noch möglich und der Begrenzereffekt wäre weg.

Man soll keine Annahmen über die eingestellte Interpolationsart zur automatischen Abtast-Konvertierung (Resample) machen. Erwarten Sie zum Beispiel nicht, daß Sample-Sounds in geringer Abtastfrequenz beim Konvertieren in höhere Abtastfrequenzen noch den gleichen Klirreffekt aufweisen. Wenn Sie wollen daß der Klirreffekt verschwindet, setzen Sie einen Frequenzwandler mit linearer Interpolation in die Prozeßkette, wollen Sie daß der Klirreffekt bleibt, setzen Sie einen derartigen Prozeß mit konstanter Interpolation ein.

Nutzen Sie nicht den Rückfaltungseffekt aus. Wenn Sie z.B. beim Konvertieren eines Pfeiftones in hoher Abtastfrequenz in einen Sample-Sound geringerer Abtastung einen sehr tiefen Ton erhalten, ist dieser Effekt eingetreten und er wird in Zukunft schwer zu reproduzieren sein.

Seien Sie skeptisch gegenüber allen Resultaten, die der Phasenmodulator abliefert. Alle Modulationstiefen über einer zehntel Sekunde bei 44 kHz können bereits die 16-Bit-(Un-)Genauigkeit der Steuerkurve hörbar werden lassen.

1.22 Fehlermeldungen

Fehlermeldungen

Kurzschluß in Berechnung

Erklärung: Formel in Variable A benutzt Variable B, die wiederum den Wert von A verwendet (Man kennt solche gegenseitigen Kompetenzverweise zur Genüge aus der modernen Bürokratie. :-)

Erklärung: Ein Prozeß A benutzt einen Prozeß B als Eingangssignal, dessen Eingänge wiederum von A abhängen. Beachten Sie: Nur innerhalb einer Anordnung von FM-Operatoren sind solche Zyklen erlaubt.

Ursache: Falsches Anzeigefenster war beim Start der Berechnung aktiv und Sie haben die Berechnung auf einem Klang gestartet, den Sie gar nicht meinten

Ursache: Beim Drag&Drop von Klängen auf Prozeß-Kästchen in Algorithmen-Anzeigen werden Variablenwerte des ersetzten Prozesses übernommen. Auf diese Weise kann es unbemerkt zu Ringschlüssen kommen.

Kein freier Speicher

Erklärung: Der freie Speicher im System reicht nicht aus, um die gerade ausgeführte Funktion erfolgreich abschließen zu können.

Ursache: Ist Zielklang zugleich Eingangssignal für die Berechnung, müssen Teile des Zielklanges zwischenzeitlich gerettet werden. Beispiel: Senkt man die Frequenz eines Sample-Sounds ab, wird der Klang dadurch länger. Der zwischenzeitlich benötigte Speicher wird immer größer und kann letztendlich an die Grenzen des Rechners stoßen.

Abhilfe: Wählen Sie einen anderen Zielsound, z.B. direkt auf einem Datenträger.

Ursache: Sie haben aus Versehen bei der Sound-Erstellung eine zu große Sound-Anzahl angegeben, oder bei datenbasierten Klängen eine zu große Länge oder zu große Abtastrate eingestellt, oder Sie haben ein ganzes DOS-Verzeichnis eingeladen, das zu viele oder zu große Dateien enthält

Einheit stimmt nicht

Erklärung: Assampler testet, ob die Einheiten aller Werte in den Klängen zusammenpassen. Das kann u.a. vor Flüchtigkeitsfehlern schützen.

Ursache: Die Einheiten der von Prozessen verwendeten Klänge oder deren Variablen passen irgendwo nicht zusammen. Falls das Programm eine Variable angibt, die es als schuldige vermutet, muß diese Vermutung nicht richtig sein. Wer kann schon sagen, welche unter mehreren Variablen die eine mit der falschen Einheit ist, es können ebenso mehrere falsch sein.

Abhilfe: In der Dokumentation der beteiligten Prozesse nachschauen, was für Einheiten erforderlich sind, bzw. wie die Werte miteinander zusammenhängen

Ursache: Sie haben einen Klang abgespielt, aber in der Konfiguration eine falsche maximale Lautstärke oder Abtastfrequenz eingestellt.

Ursache: Sie haben Einheiten nicht richtig geklammert und z.B. 1/2s fälschlicherweise anstelle von 1/(2s) geschrieben

Lesemarke eingeholt

Erklärung: Kann auftreten wenn Zielklang zugleich Eingangssignal für Berechnung liefert. In diesem Falle wird das Signal, das überschrieben zu werden droht, gepuffert. Der Fehler tritt auf, wenn einer der an der Berechnung beteiligten Prozesse beim Lesen zurücksetzt an eine Stelle, wo der Zielklang bereits nicht mehr gepuffert wird.

Ursache: Kritisch sind Schnittprozesse, Phasenmodulation und Frequenzmodulation.

Ursache: Auch Schleifen in Sample-Sounds erfordern das Zurücksetzen der Lesemarke (übrigens ein Grund warum die Übernahme der Schleifen als Spezialität der Sample-Sounds keine gute Idee war)

Abhilfe: Wählen Sie einen anderen Zielklang.

Abhilfe: Tritt der Fehler beim Abspielen auf, hilft manchmal die Vergrößerung des Wiedergabepuffers, tritt er beim Konvertieren in einen Sample-Sound auf Datenträger auf, probieren Sie es vielleicht mit einer Vergrößerung des Dateipuffers. Bei Sample-Sounds im Speicher geht das nicht so einfach, denn eine Vergrößerung der Blockgröße wirkt sich erst auf neu erzeugte Sample-Sounds aus.

Erklärung: Frequenzmodulationen mit negativem Frequenzverhältnis erfordern, daß das Signal rückwärts gelesen wird. Das ist aber so nicht vorgesehen.

Abhilfe: Auch hier müssen Sie probieren, die Puffer größer zu wählen oder es gelingt Ihnen, Ihren Algorithmus mit Hilfe des Umkehrprozesses zu realisieren.

Das läßt das Objekt nicht mit sich machen

Erklärung: Das objektorientierte Konzept des Assamplers erlaubt es, daß gleichlautende Operationen für verschiedene Arten von Klängen unterschiedlich definiert sind. Denken Sie nur an die Schnittfunktionen von Sample-Sounds und Klang-Gruppen. Genauso ist es auch möglich, daß eine Operation für eine Klangklasse gar nicht definiert ist.

Ursache: Sie wollten eine Klanggruppe in einen einzelnen Klang konvertieren

Ursache: Sie haben als Ziel einer Berechnung einen Algorithmus gewählt

So einen Sound kenne ich nicht

Erklärung: Es wurde versucht einen Klang anhand seines Namens zu finden, es konnte aber kein Klang mit diesem Namen gefunden werden

Ursache: Die Pfad-Angabe in einer Formel enthält Klangnamen, die es nicht gibt.

Ursache: Wird der Bezeichner einer Funktion, Einheit oder Konstanten falsch geschrieben, weiß Assampler nicht, daß es sich dabei um Funktionen etc. handelt und nimmt an, daß es Klangnamen wären und gibt dann den Fehler aus, wenn sich auch kein Klang mit diesem Namen findet.

Ursache: Sie haben als Referenz für einen Klang-Wert eine Formel angegeben statt eines einfachen Pfades.

Abhilfe: Falls die Referenz zu einer Formel-Variable gehört, schreiben Sie die Formel direkt als Variableninhalt. Falls es sich um einen anderen Variablentyp handelt, legen sie im gleichen Klang eine Formelvariable an, schreiben dort die Formel hinein und setzen die Referenz dann auf die neu angelegte Variable.

Als voreingestellten Klang darf ich wohl noch eine Gruppe erwarten

Erklärung: Beim Anlegen eines neuen Klanges, sei es durch Doppelklick im Klassenbaum oder durch Kopierfunktionen, wird eine Gruppe benötigt, in der der neue Sound abgelegt werden kann.

Ursache: Sie haben gar nichts richtiges oder etwas anderes als eine Gruppe eingestellt, oder sie haben eine eingestellte Gruppe inzwischen gelöscht.

Abhilfe: Stellen Sie als Zwischenablage bei Klangeinstellungen, bei "Anlegen in" in den Einstellungen für datenbasierte Klänge, Gruppen und Prozesse Klanggruppen ein.

Zukunft

Die Möglichkeiten einen Fehler zurückzuverfolgen sind zur Zeit unter aller Kanone. Eine umfassende Lösung dafür muß erst noch implementiert werden.

1.23 Bedienoberfläche

Die Bedienoberfläche

1 Fenster

1.1 globale Fenster

1.1.1 Edit

1.1.2 Klassen

1.1.3 Fäden

1.1.4 Rechner

1.1.5 Einsteller

1.2 lokale Fenster

1.2.1 Anzeige

1.2.2 Information

2 Menü

3 Bedienelemente für

3.1 Variablenamen

3.2 Zahleneingabe

3.3 Klangwahl

3.4 Proportionen

3.5 Listen

3.6 Zeicheneingabe

1.24 Die limitierte Edel-Edition

Das Hauptfenster

Hinweis: "Markierter Bereich" ist bei Gruppen als "markierte Klänge" zu verstehen.

Wiedergabe

Klang

Spielt den aktuellen Klang komplett ab.

Anzeige

Spielt den sichtbaren Bereich des aktuellen Klanges ab.

Bereich

Spielt den markierten Bereich des aktuellen Klanges ab.

Schleife

Spielt die Schleife des aktuellen Klanges ab.

Stop

Stoppt die Klangausgabe.

Beachten Sie auch die Einstellungen zur Klangwiedergabe!

Anzeige

Klang

Aktueller Klang wird in voller Länge angezeigt.

1 : 1

Jeder Sample-Wert bekommt eine Pixelbreite in der Darstellung.

Bereich

Es wird genau der markierte Bereich angezeigt.

Schleife

Anzeige zeigt genau die Klang-Schleife.

Schwingung

Anzeige zeigt genau eine Schwingungslänge, so diese beim aktuellen Klang richtig eingestellt wurde.

Zoom

Mit Cyclegadget wird Vergrößerungsfaktor eingestellt.

Pfeil nach links = Zoom hinein = Ausschnitt kleiner = mehr Details

Pfeil nach rechts = Zoom heraus = Ausschnitt größer = weniger Details

Bereiche

Klang

Markiert gesamten aktuellen Klang.

Anzeige

Markiert angezeigten Ausschnitt des aktuellen Klanges.

Schleife

Markiert Schleife des Klanges.

Vor Schleife

Markiert Bereich von Anfang des Klanges bis Schleifenbeginn.

Suche nach Anfang und Ende

Eingegeben wird Ausdruck für Trigger-Schwelle, also für den Lautstärkewert, ab dem ein Geräusch als vorhanden gelten soll. Maximal Lautstärke ist wie immer 1.

Pfeil links nach rechts: Bereichsbeginn wird auf Geräuschbeginn gesetzt.

Pfeil rechts nach links: Bereichsende wird auf Geräuschende gesetzt.

Sinnvoll als Vorbereitung zum Bearbeiten/Freistellen-Befehl.

Intellenztest! Wie kann man außer mit den oben genannten Funktionen (egal ob mit Gadget oder Tastenkombination ausgelöst) den ganzen Sound markieren oder anzeigen? Hinweis: Auf die gleiche Weise kann man eine Markierung der Größe des sichtbaren Bildausschnittes veranlassen und umgedreht den Bildausschnitt so groß wie die Markierung wählen.

Antwort

Bearbeiten

Alles was beim Ausschneiden anfällt landet in der Zwischenablage-Gruppe. Welche das sein soll, läßt sich beliebig konfigurieren. Vor jeder Aktion, die etwas in der Zwischenablage ablegt, wird diese komplett und stillschweigend freigeräumt!

Schneiden

Schneidet markierten Bereich aus und legt Ausgeschnittenes in Zwischenablage ab.

Kopieren

Kopiert markierten Bereich in Zwischenablage.

Einfügen

Fügt Klang aus Zwischenablage ab Anfang des markierten Bereichs ein.

Ersetzen

Ersetzt aktuellen Klang ab Anfang des markierten Bereichs durch Klang aus Zwischenablage. Länge des markierten Bereichs spielt keine Rolle.

Löschen

Entfernt Teil des Klanges, der markiert ist. Im Gegensatz zum Schneiden wird keine Kopie in der Zwischenablage abgelegt.

Leeren

Setzt markierten Teil des Klanges auf Nullpegel. Im Gegensatz zum Löschen entsteht eine Pause, der Klang bleibt genauso lang.

Freistellen

Befreit Klang von Teilen vor und nach dem markierten Bereich.

Vervielfachen

Führt Einschieben mehrfach hintereinander aus. Anzahl der Wiederholungen sind im Stringgadget einzustellen.

Leer einfügen

Fügt ab Bereichsbeginn eine Pause der nebenstehend angegebenen Dauer ein. (Einheit nicht vergessen! Bei Zeitangaben zum Beispiel Sekunden)

Datei

Laden

Lädt Klang zur Bearbeitung in den Speicher. Ist aktueller Klang Gruppe, wird eingeladener Klang in dieser untergebracht, andernfalls wird neuer Klang neben dem aktuellen auf gleicher Ebene abgelegt. Im Dateiauswahlfenster können mehrere Dateien, darunter auch Verzeichnisse ausgewählt und eingeladen werden. Möglich ist auch, Piktogramme von der Workbench in ein Anzeigefenster zu ziehen. Für die Ladefunktion gibt es zahlreiche Einstellungen allgemein und für Sample-Sounds.

Öffnen

Öffnet den Klang zur Bearbeitung auf dem Quell-Datenträger. Das Gerät sollte möglichst schnell sein, Festplatte oder RamDisk sind schwer zu empfehlen, mit Disketten werden Sie Ihres Lebens nicht froh. Funktioniert nur mit Roh-Sampleformaten und wird als Disk-Sample-Sound geführt.

Vorsicht!!! Jede Operation an einem Disk-Sample-Sound verändert die Datei direkt. Änderungen können nicht zurückgenommen werden.

Sichern

Speichert den aktuellen Klang auf einem Datenträger ab. Zuvor erwartet Sie noch ein ReqTools-Dateiauswahlfenster. Gesonderte Einstellungen gibt es für Gruppen und Sample-Sounds.

Klang

Löschen/Schließen

Löscht den Klang nach Sicherheitsrückfrage. Disk-Sample-Sounds werden auf dem Datenträger nicht angerührt, nur die Klangverwaltung im Assampler wird aufgelöst.

Schließ-Gadget

Das Schließ-Gadget des Fensters beendet das Programm.

Tips

Einlesen von Sample-Sounds aus dem Speicher

Mit dem MEM-Handler (AmiNet6:disk/misc/MEM_Handler371.lha) kann man Sample-Sounds direkt aus dem Speicher lesen, wenn man zum Beispiel nach einem Absturz seine Sampledaten noch retten möchte. - Allerdings nützt Ihnen das beim Assampler wenig, denn der zerhackt die Sample-Sounds im Speicher, so daß Sie kaum etwas wieder zusammengeflickt bekommen.

Nichtsdestotrotz ist es nützlich, aber nicht ganz einfach, das logische MEM:-Gerät zu benutzen. Da es kein Verzeichnis vorweisen kann, verweigert das ReqTools-Dateiauswahlfenster die Arbeit mit ihm. Um es auszutricksen, müssen Sie in dessen Pfad-Stringgadget irgendeinen gültigen Pfad (z.B. ram:) eingetragen haben, und das Verzeichnis sollte auch eingelesen sein. Nun tippen Sie beim Dateinamen den ganzen Pfad für den Mem-Handler ein, z.B. MEM:10000/100 und drücken nicht <ENTER>! Dann würde das Dateiauswahlfenster nämlich den Dateinamen auseinanderpflücken und versuchen, das Verzeichnis MEM:10000 einzulesen. Das gelingt ihm aber nicht, und es wird deswegen das Verlassen des Requesters egal ob über Ok oder Abbruch als Abbruch werten. Deshalb bei eingelesenem gültigen Verzeichnis (egal welches) und vollständigem MEM:-Pfad im Dateinamen-Gadget den Ok-Knopf drücken.

Dieselben Probleme dürfte es bei ähnlichen Geräten wie dem PIPE: geben, nur daß es in diesem speziellen Fall schon daran scheitert, daß die Dateien (=Warteschlangen-Kanäle) aus dem PIPE: keine Länge besitzen.

1.25 Erst versuchen selbst darauf zu kommen, bevor Sie illern!

Die Lösung heißt Drag&Drop: Man zieht einfach im Anzeigefenster das Klanglänge-Gadget auf das Anzeigenlänge-Gadget bzw. auf das Markierungslänge-Gadget. Da die Länge jeweils den ganzen Sound umfaßt, erübrigt sich die Frage, wo der Bildausschnitt bzw. die Markierung beginnt. Bei Anzeigen/Markierung oder Markieren/Anzeige ist das nicht der Fall, deshalb können diese Funktionen nicht durch Drag&Drop ersetzt werden.

1.26 Das Fenster zur Klangwelt

Das Klassenfenster

Aussehen

Das Fenster ist unterteilt in die Auflistung aller Klangklassen auf der linken Seite und der Wertetafeln für neu erzeugte Klänge auf der rechten Seite.

Erzeugung neuer Klänge

Zuerst wählen Sie eine Klangklasse aus dem Klassenbaum aus. Es erscheinen die dazu gehörigen Werte auf der rechten Seite. Ändern Sie die Werte nach Ihren Wünschen ab, und überlegen Sie sich, wo Sie den erzeugten Klang ablegen möchten.

Im Stringgadget rechts neben dem Wort "Erzeugen" tragen Sie die Anzahl der Klänge ein, die Sie mit einem Male erzeugen möchten. Klicken Sie auf den Knopf "Kinder", werden neue Klänge erzeugt und an die bestehenden untergeordneten Klänge der aktuellen Klanggruppe angehängt. Diese Funktion steht also nur zur Verfügung, wenn im aktiven Anzeigefenster eine Gruppe angezeigt wird. Der Knopf "Geschwister" veranlaßt, daß die neuen Klänge direkt hinter dem aktuell sichtbaren eingeordnet werden. Diese Klänge landen also in der gleichen Gruppen-Ebene. Ganz ähnlich sieht es mit "Klone" aus, zusätzlich wird hier noch für jeden neuen Klang eine Berechnung gestartet, welche den aktuellen Klang in den neuen überträgt.

"Konvertieren" ersetzt den aktuell sichtbaren Klang durch einen Klang anderer Klasse, konvertiert zuvor aber die Daten des alten Klanges in die des neuen Klanges.

Bestimmung der Klasse eines Klanges

Ziehen Sie einen Verweis zu dem betrachteten Klang über den Klassenbaum.

Schnelles Auffüllen der Wertetafeln

Wenn Sie einen Klang herumliegen haben und einen zweiten erzeugen möchten, der die gleichen Werte besitzt (aber nicht unbedingt der gleichen Klasse angehört), können Sie irgendein Element der Bedienoberfläche, das auf diesen Klang verweist, über die Wertetafel ziehen und fallen lassen. Die Werte des schon existierenden Klanges werden eingetragen und Sie können einen neuen Klang mit diesen Werten erzeugen.

Ebenso komfortabel wäre es, könnte man einfach einen Eintrag aus dem Klassenbaum in die Wertetafel ziehen, um Werte der einen Klasse in eine andere zu übernehmen. Das scheitert daran, daß dem Ziehen eines Eintrages dessen Anwahl vorausgeht, d.h. in dem Moment wo Sie den Eintrag wegziehen, wird schon die Wertetafel für diesen Eintrag aufgebaut. Folgender Umweg ist nötig: Sie ziehen den Eintrag der Klasse mit den zu übernehmenden Werten in die Ablage unterhalb des Klassenbaumes. Nun wählen Sie die Ziel-Klasse aus und ziehen nun die Ablage über die Wertetafel. Fertig.

1.27 Wieso sollten wir ausgerechnet Sie einstellen?

Das Einstellerfenster

Im Einstellerfenster finden Sie alle Assampler-spezifischen Einstellungen, welche zum einen das Erscheinungsbild (schön bunt!) und zum anderen die Funktionsweise des Programmes beeinflussen. Beachten Sie, daß die Einstellungen prinzipiell sofort wirksam sind. Sie sind nicht als Voreinstellungen anzusehen, die man einmal als seine Optimalkonfiguration austüftelt, um sie dann für immer unberührt zu lassen. Es ist vielmehr von Nutzen, das Einstellerfenster längere Zeit geöffnet zu halten, wenn man z.B. Lade- und Speicherformate ändern will. Änderungen am Outfit werden erst nach Umgrößern oder Öffnen der betroffenen Fenster sichtbar.

Puffergrößen werden immer in Samples angegeben, da wir es hier mit 16-Bit-Samples zu tun haben, ist der benötigte Speicher für einen Puffer mit n Samples 2*n Bytes.

Farbeinstellungen werden über das Farbstift-Gadget von MUI erfaßt. Bei Klick auf selbiges können Sie die Farben anhand der vorgegebenen Stifte oder über RGB-Werte festlegen. Die Farben werden dynamisch vergeben, das kann vor allem bei Screens mit wenig Farben dazu führen, das nicht alle Farben so dargestellt werden, wie man sich das wünscht.

Die Einstellungen der Soundklassen werden bei den jeweiligen Klassen beschrieben.

Einstellungen der Soundklassen

Sichern

Speichert Einstellungen nach ENVARC: und ENV:, also höchstwahrscheinlich auch auf Festplatte, so daß sie auch Systemstarts überstehen.

Benutzen

Speichert Einstellungen nach ENV:, also höchstwahrscheinlich in die RamDisk, so daß sie beim nächsten Programmstart vorhanden sind, solange kein Systemneustart dazwischen lag.

Zurück

Lädt die Einstellungen aus ENV: zurück, so daß die zuletzt gesicherten/benutzten Einstellungen wiederhergestellt werden.

1.28 Ihh, das Programm zieht Fäden!

Das Fenster für die einzelnen Programmfäden

Aussehen

Das Fenster zeigt für jeden Programmfaden eine Fortschrittsanzeige und einen Stop-Knopf.

Funktion

Die Fortschrittsanzeige stellt dar, wieviel von der gesamten Berechnung schon erledigt ist. Als Text enthält Sie Informationen darüber, wieviele Aufträge noch anstehen. Diese Zahl täuscht, denn eine Berechnung an einer Gruppe oder eine Berechnung die durch Ablage- oder Umkehrprozese in Etappen unterteilt ist, wird als ein Auftrag gewertet.

Der Stop-Knopf bricht die Berechnung so bald wie möglich ab. Beachten Sie, daß eine Unterbrechung nur zwischen zwei Blöcken erfolgen kann, je größer Sie bestimmte Puffer wählen, desto seltener wird ein Abbruch akzeptiert.

1.29 finstere Fenster

Soundfenster

Zu jedem Sound können verschiedene Fenster geöffnet werden. In den Fenstern selbst kann aber noch der betrachtete Sound gewechselt werden. Dies passiert mit dem Klangwahl-Gadget oder mit den Pfeiltasten:

1. Links: Vorhergehendes Geschwisterkind des aktuellen Sounds.

2. Rechts: Nachfolgendes Geschwisterkind des aktuellen Sounds.

3. Hinauf: Übergeordnete Gruppe

4. Hinunter: erstes Kind

Man kann diese Deutung auch unlogisch finden, weil in der Darstellung der Gruppen Waagerechte und Senkrechte vertauscht sind. D.h. nach oben und unten wechselt man zwischen Sounds der gleichen Ebene, links und rechts hingegen zwischen verschiedenen Ebenen.

Ein Klang-Fenster zeigt nur Daten über einen Sound an, es repräsentiert ihn nicht. Daher führt das Schließen eines Klang-Fensters nicht zum Löschen des betreffenden Klanges!

Anzeigefenster

Informationsfenster

1.30 Wir bringen jedes Sound-Experiment - auch den Versuch - zur Anzeige

Das Anzeigefenster

Dies ist ein Klangfenster.

Aussehen

Anzeigefenster enthalten mindestens ein Klangwahl-Gadget und entsprechende Pfeilknöpfe links unten, oft noch weiter Zahlengadgets für Bereichsmarkierungen und ähnliches. Auf jeden Fall enthält es die Anzeige eines Klanges, so es dort etwas anzuzeigen gibt (bei Prozessen ist das in der aktuellen Version nicht der Fall).

Funktion

Anzeigefenster sollen die Daten eines Klanges anzeigen (wahnsinnig informative Beschreibung, nicht wahr?), was eigentlich nur bei Datensounds und Gruppen Sinn hat (aber nicht Sinn macht, wie es sich mit der zunehmenden Einenglischung der deutschen Sprache eingeschliffen hat).

Es können mehrere Anzeigefenster gleichzeitig geöffnet werden, auch vom gleichen Klang. Das zuletzt aktivierte Anzeigenfenster und der darin angezeigte Klang gelten als Ziel bei den meisten Aktionen und heißen dann kurz aktuelle Anzeige bzw. aktueller Klang.

Jeder Klang besitzt außerdem ein Informationsfenster.

Drag&Drop

Wenn Assampler auf der Workbench betrieben wird, kann man Piktogramme von Sound-Dateien (auch mehrere zugleich) über ein Anzeigefenster ziehen und dort fallen lassen. Daraufhin werden die betreffenden Dateien eingeladen. Die Positionierung innerhalb der Gruppen-Hierarchie erfolgt genau wie beim Laden über das Hauptfenster bzw. Menü.

Tastatur

Alle Tastenkürzel sind konfigurierbar.

Zukunft

In Zukunft könnten auch für bestimmte Prozesse Anzeigefenster zur Verfügung stehen. Sinnvoll wäre zum Beispiel eine Anzeige der Übertragungsfunktion bei Filtern oder die von Erzeugern produzierten Kurven, mit entsprechenden Griffpunkten zum visuellen Verändern der Parameter. Es wäre allgemein überlegenswert, ob nicht grundsätzlich alle Prozesse im Anzeigefenster ihre Ergebnisse anzeigen würden. Das brächte aber große Schwierigkeiten mit sich, da für manche Herumscrollerei sehr viel berechnet werden müßte.

1.31 Nur zur Information

Das Informationsfenster

Dies ist ein Klangfenster.

Aussehen

Informationsfenster enthalten mindestens ein Klangwahl-Gadget und entsprechende Pfeilknöpfe links unten, rechts daneben ein paar Knöpfe. Im oberen Teil befinden sich einige Gadgets zur Werteerfassung, bei Prozessen können das recht viele sein.

Funktion

Informationsfenster zeigen die Werte eines Klanges an. Besonders bei Prozessen und Algorithmen ist es hilfreich, daß man über Parameter den Klang von außen verändern kann.

Im Gegensatz zu Anzeigefenstern, kann pro Klang nur ein Informationsfenster geöffnet werden. Außerdem hat das Aktivieren eines Informationsfensters keinen Einfluß darauf, welcher Klang als aktueller gilt.

Variablen

Öffnet das Fenster, in dem Anordnung und Art der benutzerdefinierten Variablen geändert werden können.

Wiedergabe

Spielt den Klang ab. Enthält der Klang eine Length-Variable, wird er nur für diese Zeit abgespielt.

Start

Konvertiert/kopiert den Klang dieses Informationsfensters in den Klang im aktuellen Anzeigefenster.

Wenn Sie Sounds kopieren wollen, erscheint die Aufschrift "Start" nicht besonders passend. Bedenken Sie aber, daß die Aktion hauptsächlich für die Prozesse eingerichtet wurde.

1.32 Liste der benutzerdefinierten Variablen

Das Fenster für die benutzerdefinierten Variablen

Diese Fenster gehören immer fest zu einem Informationsfenster.

Aussehen

Enthält eine große Liste und darunter eine zweizeilige Knopfleiste.

Funktion

Jeder Klang kann über die festverdrahteten hinaus auch benutzerdefinierte Werte enthalten. Um die Anordnung zu ändern oder Variablen hinzuzufügen oder zu löschen, rufen Sie mit dem "Variablen"-Knopf aus dem Informationsfenster dieses Fenster hier auf. Eigentlich ist es nicht als Requester gedacht, den man mit Ok oder Abbruch verläßt, sondern als Zustandsanzeige, so daß der Inhalt dieses Fenster auch geändert wird, wenn man im Informationsfenster den Klang wechselt. Diese Vorgehensweise ist etwas ungewöhnlich, vielleicht löse ich sie in Zukunft durch eine leichter erfaßbare ab.

Im Variablenfenster erscheint eine Liste aller aktuell eingetragenen benutzerdefinierten Variablen. Ein Umordnen innerhalb der Liste und Kopieren in andere Listen ist per Drag&Drop möglich.

Erzeugung von Variablen

Mit "Neu" wird der aktuelle Eintrag in der Liste deaktiviert. Stellen Sie nun links von dem String-Gadget den Variablentyp ein und rechts davon die Spalte in der die Variable im Fenster dargestellt werden soll. Dann tippen Sie den Namen der Variable in das Stringgadget und drücken <ENTER>. Weitere Variablen können einfach durch Namensänderung und erneutes <ENTER> erstellt werden. Noch bequemer gestaltet sich die Erzeugung, wenn man eine Variable aus einem bestehenden Klang an ihrem Bezeichner greift und sie in die Variablen-Liste zieht und dort fallen läßt.

Vervielfachen

Namens- und inhaltsgleiche Variablen von den gerade markierten Variablen werden mit dem "Duplizieren"-Knopf erstellt.

Löschen

Entsprechend werden die markierten Variablen mit einem Klick auf "Löschen" entfernt.

Ändern

Der Typ, der Name und die Spalte der Variable kann jederzeit geändert werden, indem man die betreffende Variable anklickt (= aktiviert, evtl. weitere markierte Variablen werden nicht beeinflußt) und die entsprechenden Änderungen am entsprechenden Knopf/Feld vornimmt. Bei Namensänderung <ENTER> nicht vergessen. Der einzige Unterschied zum Erzeugen von Variablen besteht darin, daß hier eine Variable in der Liste aktiviert ist, beim Erzeugen jedoch keine.

Wiederherstellen

Eine einzelne Variable kann anhand ihres Namens auf den Stand vor Öffnen des Variablenfensters zurückgesetzt werden mit "Restaurieren". Den kompletten Zustand vor Öffnen dieses Fensters erhält man mit "Alles Restaurieren" zurück und vernichtet damit alle bisher vorgenommenen Änderungen. Das ist gewissermaßen der Ersatz für einen Abbruch-Knopf.

Details festlegen

Je nach Variablentyp kann man noch weitere Einstellungen für die Variablen vornehmen (uff).

Mit Doppelklick auf einen Listeneintrag gelangen Sie in ein Fenster, das Sie die Eigenschaften einer Variable abhängig von ihrem Typ ändern läßt.

Klang

Symbol und Seite für den Anschluß an dem Kästchen, das in Algorithmus dargestellt wird; Die Behandlung für nicht eingestellte Klänge: überhaupt nicht zulassen, zulassen und als unbelegt weiterreichen oder aber zulassen und als Einstellung des jeweils aktuellen Sounds werten

Formel

Festlegung von oberen und unteren Grenzen und ob diese Grenzen einschließend oder ausschließend zu verstehen sind; Liste der vorgeschlagenen Formeln für diese Variable welche dann über das Kontextmenü abgefragt werden können

Schieber

Minimal- (ganz links) und Maximalwert (ganz rechts) des Schiebereglers

Auswahl

Liste aller Auswahlmöglichkeiten

Schalter

Hier braucht man ausnahmsweise mal nichts einzustellen

Ist man fertig mit der Eingabe, verläßt man das Variablen-Fenster über das Schließsymbol des Fensters. Die Änderungen werden nun in den Klang übernommen. Jeder Variablenname (die vorgegebenen Variablen mitgezählt) darf nur einmal vorkommen. Ist das nicht erfüllt, kann das Variablenfenster nicht geschlossen werden.

1.33 Taschenrechner für Nebenrechnungen

Das Taschenrechnerfenster

Aussehen

Anders als Sie wahrscheinlich erwartet haben, besteht das Fenster lediglich aus zwei Zahlen-Gadgets.

Funktion

Sie tippen in die Eingabe-Zeile eine Formel in Assampler-Notation ein und erhalten nach dem abschließenden <ENTER> Ihr Ergebnis in der Zeile darunter.

Zukunft

Für die Zukunft könnte man das Fenster noch richtig aufblähen, so daß alle Funktionen, Operatoren, Konstanten und Ziffern schnell per Maus erreichbar sind.

Da ergeben sich zwei Möglichkeiten:

1. Die konventionelle: Man empfindet das Fenster einem Taschenrechner nach und gibt jedem Formelelement einen eigenen Knopf. Das ist platzintensiv aber wahrscheinlich schneller zu bedienen.

2. Die dynamische: Man bringt alle Formelelemente in einer Liste oder einem Baum analog zur Struktur der Zahlen-Gadget-Menüs unter. Das ist platzsparend und leicht erweiterbar, man muß aber mehr kurbeln, um an alle Teile heranzukommen.

In beiden Möglichkeiten wäre es möglich, per Drag&Drop Formelelemente in andere Stringgadgets zu bugsieren.

Ungeklärt ist auch, ob man mit einem Rechner-Fenster auskommt, das mit einem Speicher für alle vorangegangen Rechnungen, ausgestattet ist, oder ob man weiterhin mehrere Rechnerfenster zuläßt (Platzbedarf!).

Was meinen Sie dazu?

1.34 aus unserem reichhaltigen Menü heute

Das globale Pull-Down-Menü

Menüpunkte mit nachgestelltem "..." lösen nicht direkt eine Aktion aus, sondern öffnen zuvor einen Requester, so daß ein Abbruch immer noch möglich ist.

Projekt

Laden, Öffnen, Sichern, Löschen

Erfüllen die gleiche Aufgabe, wie die entsprechenden Knöpfe im Hauptfenster.

Hilfe

Zeigt dieses AmigaGuide-Dokument an.

Über

Gibt eine kurze Information über Assampler und die vorliegende Version aus.

Über MUI

Gibt eine kurze Information über das Bedienoberflächensystem MUI aus.

Ikonifizieren

Die Bedienoberfläche wird geschlossen, was vor allem Platz spart. Das Programm ist nach wie vor aktiv und Berechnungen werden weitergeführt.

Beenden

Beendet das Programm.

Fenster

Mit diesem Untermenü können Fenster geöffnet und geschlossen werden. Eingerückte Menüpunkte weisen auf Fenster hin, die nur einmal vorhanden sind. Ein vorangestellter Haken zeigt geöffnete Fenster an. Alle anderen Menüpunkte öffnen Fenster, die mehrfach vorhanden sein können.

Bearbeiten

Alle Schnittfunktionen, die auch im Hauptfenster zu erreichen sind.

Einstellungen

Assampler

Öffnet das Assampler-Einstellerfenster. Weitere Einstellungen zum Outfit, aber auch zur Arbeitsweise vom Assampler können hier eingestellt werden.

MUI

Öffnet das MUI-Einstellerfenster. Hier können Einstellungen am Outfit von Assampler vorgenommen werden, sofern Sie keine Assampler-Besonderheiten (Farben bei Sample-Darstellung usw.) betreffen.

AHI

Startet das AHI-Einstellerprogramm. Hier können Einstellungen zur Audioausgabe getroffen werden. Die meisten sind allerdings belanglos, und werden durch Assampler-eigene Einstellungen ersetzt.

1.35 Variablennamen

Variablennamen

Aussehen

Völlig unscheinbare Texte, die links neben den klotzigen Aufbauten der Variablen in Informationsfenstern stehen.

Funktion

Mit Einfach-Klick auf den Namen wird das Einstellungsfenster der entsprechenden Variablen geöffnet. Handelt es sich um eine benutzerdefinierte Variable werden nach dem Schließen des Fensters alle Änderungen übernommen, andernfalls werden sie verworfen.

Tastatur

Die unterstrichenen Buchstaben zeigen das Tastenkürzel an, mit dem die Variable aktiviert werden kann. Ist kein Buchstabe unterstrichen, haben die Buchstaben nicht gereicht und es muß mit TAB durchgeschaltet werden.

Drag&Drop

Der Variablenname-Text repräsentiert die Variable, kann also in Variablenlisten oder String-Gadgets gezogen werden. Zieht man einen Variablennamen auf einen anderen werden sämtliche Einstellungen übernommen.

1.36 Pfadfinder

Pfadangaben für Klänge und Variablen

Pfade in Assampler sind ganz ähnlich zu denen im DOS. Ein Pfad gibt gewissermaßen die Zwischenstationen an, über die man zu einem Klang oder zu einer Variable gelangt.

Bei DOS-Pfaden wird der Schrägstrich zum Abtrennen der Pfadkomponenten benutzt. Im Assampler ist das nicht möglich, da der Schrägstrich bereits für die Division reserviert ist. Stattdessen wird der Rückwärts-Schrägstrich ("\", auch Bäcksläsch genannt :-) verwendet. Das erinnert stark an würg MS-DOS, aber ich dachte, besser erinnert es an überhaupt irgendwas, als daß Sie alles neu lernen müssen.

Die Möglichkeiten, wie ein Pfad beginnen kann:

\ Pfad startet bei der Wurzelgruppe

_ Pfad startet beim aktuellen Klang oder beim jeweiligen Zielklang einer Berechnung

sonst Pfad beginnt beim gerade betrachteten Klang

Beispiel:

\User bezeichnet die Gruppe User, welche sich in der ersten Ebene befindet

_ bezeichnet den aktuellen (Ziel-)Klang

Sine bezeichnet den Klang Sine, der sich innerhalb der aktuell betrachten Gruppe befindet

Mit zwei aufeinanderfolgenden Punkten .. wird die übergeordnete Gruppe bezeichnet.

Beispiel:

\Waveforms\Sine\.. ist ein besonders umständlicher Pfad für die Gruppe Waveforms

\Waveforms\Sine\..\.. ein noch umständlicherer Pfad für die Wurzelgruppe

Mit einem einzelnen Punkt . wird zu einer Variablen verzweigt oder eine Klang-Funktion eingeleitet. Zur Abgrenzung von zwei aufeinanderfolgenden Punkten kann ein \ eingeschoben werden.

.Input bezeichnet den Klang, der für die Variable Input des aktuellen Klanges eingestellt ist

.Input\.. bezeichnet die übergeordnete Gruppe zu diesem Klang

\User.Multipitch bezeichnet die Variable Multipitch der User-Gruppe, welche sich in der Wurzelgruppe befindet

_.Index bezeichnet die Variable Index des Zielklanges, Index gibt die Position des Klanges innerhalb der übergeordneten Gruppe an

1.37 Stringgadget für Formeln

Das Stringgadget für mathematische Ausdrücke

Aussehen

Sieht zunächst wie ein einfaches Stringgadget aus, hat darüber hinaus noch ein Kontextmenü.

Funktion

Format

Die Formeleingabe ist stark an die mathematische Schreibweise angelehnt. Groß- und Kleinschreibung ist einzuhalten. Komplexe Zahlen werden (fast) vollständig unterstützt. Pfadangaben die Zahlen enthalten, sind in Anführungszeichen zu setzen.

Beispiele: Exp 2, ".Array.0.Input", ".Input0"

Aber nicht: exp 2, .Array.0.Input, .Input0

Konstanten

Folgende Bezeichner sind mit vorgegebenen Werten belegt:

pi = 3.141...

e = 2.718...

i^2 = -1

% = 0.01

¼ = 0.25

½ = 0.50

¾ = 0.75

mach = 332 m/s Schallgeschwindigkeit

c = 300000 km/s Lichtgeschwindigkeit

G = 9.81 m/s^2 Erdbeschleunigung

Einheiten

Einheiten sind wesentlicher Bestandteil des Gesamt-Konzepts. Einheitenbezeichner sind Konstanten, ihr Wert wird mit der vorangestellten Zahl multipliziert. Klammern dürfen also nicht vergessen werden, wenn man eine Operation hoher Priorität auf einheitenbehaftete Werte anwendet.

Manche Einheiten muten etwas exotisch für ein Klangbearbeitungsprogramm an. Gerade bei der Simulation natürlicher Schallereignisse oder physikalischer Experimente können sie sehr nützlich sein.

elektrische Spannung: mV, V, kV

Winkel: rad, grad, °, ', " (Radianten, Neugrad, Grad, Bogenminute, Bogensekunde)

Zeit: us, ms, s, min, h, d, a (beachte: 1h+10min+27s statt 01:10:27)

Frequenz: Hz, kHz, MHz, bpm (beats per minute)

Länge: um, mm, cm, dm, m, km

Fläche: ha (Hektar)

Volumen: ml, cl, l

Masse: ug, mg, g, kg, dt, t, kt

Kraft: N, kN, kp (Kilopond)

Druck: Pa, kPa, bar

Energie: eV, J, kJ, cal, kcal

Leistung: mW, W, kW, PS

elektrische Ladung: C

elektrischer Strom: uA, mA, A

elektrischer Widerstand: Ohm, kOhm, MOhm

elektrische Kapazität: uF, nF, pF

Temperatur: K

Information: bit, B, kB, KB, MB, GB

Datenrate: baud, kbaud, Kbaud, Mbaud, Gbaud

Infix-Operatoren

+ Addition

- Subtraktion

* Multiplikation (kann weggelassen werden)

/ Division

^ Potenz, wird von rechts aufgelöst, d.h. a^b^c = a^(b^c)

Präfix-Funktionen

Wenn es die Prioritäten erlauben, dürfen bei Funktionen mit einem oder keinem Argument die Klammern um die Operanden entfallen.

Beispiele: Exp x, aber Exp (2x)

-z entgegengesetzte/negierte Zahl zu z

Re(z) Realteil der komplexen Zahl

Im(z) Imaginärteil der komplexen Zahl

Sin(z) Sinus, [z]=1° ([z] bedeute hier "Einheit von z")

Cos(z) Cosinus, [z]=1°

Tan(z) Tangens, [z]=1°

ArcSin(z) Arkussinus, [ArcSin(z)]=1°

ArcCos(z) Arkuscosinus, [ArcCos(z)]=1°

ArcTan(z) Arkustangens, [ArcTan(z)]=1°

Exp(z) eˆz

Ln(z) natürlicher Logarithmus

Log(z,base) Logarithmus von z zur Basis base

dekadischer Logarithmus falls Argument base ausgelassen

Sqrt(z) Quadratwurzel mit Re(Sqrt(z))>0 oder

(Re(Sqrt(z))=0 und Im(Sqrt(z))>=0)

Abs(z) Betrag der komplexen Zahl

Floor(z) rundet Real- und Imaginärteil auf ganze Zahlen ab

Beispiele:

Sin (20°)

Exp (2pi/5)

Klang-Funktionen

Diese Funktionen berechnen Eigenschaften eines Klanges. Der Pfad des Klanges wird dazu dem Funktionsnamen vorangestellt und mit einem Punkt vom Funktionsnamen abgetrennt. Fehlt der Pfad, bezieht sich die Funktion auf den Klang im aktuellen Anzeigefenster.

Beispiel: .Input.MaxVol, \User\Sample.ZeroLevel

Samples() Anzahl der Samples eines datenbasierten Klanges

(bitte nicht selbst verwenden!)

MaxVol() Betrag der maximalen Auslenkung

SqrVol() Durchschnittliche Auslenkung, quadratisches Mittel

AvrgVol() Durchschnittliche Auslenkung, arithmetisches Mittel

Maximum() Maximalwert

Minimum() Minimalwert

ZeroLevel() Gleichspannungsoffset

SoundBegin(y) Stelle, an der der Betrag der Auslenkung

erstmals y überschreitet, vorwärts gesucht

SoundEnd(y) Stelle, an der der Betrag der Auslenkung

erstmals y überschreitet, rückwärts gesucht

Y(x) Signalwert an Stelle x

FreqAmp(x,f) Amplitude der Frequenz f zum Zeitpunkt x im Spektrum-Klang

Tone(string)

Frequenz des Tones, dessen Name in String steht, große Buchstaben sind Subkontra-Töne

Beispiele: C2, ..., C1, ..., C0, ..., B0, H0, c0, cis0, d0, es0, e0, ..., c1, ..., c2

Beispiele: Tone(c1), Tone("ais3")

dB()

Berechnet Verhältnis, daß durch die dezi-Bel-Angabe bestimmt ist. Ein 1 B = 10 dB entspricht einem Verhältnis von 1 zu 10. Allerdings ist nicht immer klar welche Größen ins Verhältnis gesetzt werden. Diese Größen können sein Schalldruck, -leistung, -intensität, elektrische Leistung, Spannung. Meint man mit einer dB-Angabe ein Leistungsverhältnis, dann ist das entsprechende Spannungsverhältnis das Quadrat davon.

So richtig als Einheit eignet sich dB nicht, deswegen habe ich dB als Funktion angelegt (genaugenommen ist es schlicht eine Exponentialfunktion). Damit beuge ich auch solchem Unfug vor, daß 0dB + 0dB = 3dB seien. Das kommt nämlich ganz darauf an, wie man das Plus in diesem Falle definiert :-)

Beispiel: dB(20) = 100, dB(-10) = 0.1, dB(3.0103) = 2 (ungefähr)

RoundPrimeSum (n, maxSum, unit, deviation, dir)

RoundPrimeMax (n, maxPrime, unit, deviation, dir)

Beide Funktionen werden für die schnelle Fouriertransformation gebraucht, damit diese auch wirklich schnell ist.

n

Länge des zu transformierenden Signals

unit

Dauer eines Sample-Wertes (Sample-Periode), ist in der Regel 1/SampleFreq

maxSum

maximal zulässige Summe aller Primfaktoren als Verhältnis zu n = Rechenzeitbeschränkung

maxPrime

maximal zulässige Primzahl in der Primzahlzerlegung = Rechenzeitbeschränkung

deviation

relative Abweichung vom Idealwert

Innerhalb dieser werden ohne Rücksicht auf Verluste Primzahlenzerlegungen gesucht, die die Rechenzeit minimieren. Wenn sich innerhalb dieses Intervalls überhaupt keine Zerlegung finden läßt, die das maxSum- bzw. das maxPrime-Kriterium erfüllen, wird außerhalb der Abweichung nach dem nächsten n gesucht, daß wenigstens dem geforderten Kriterium genügt.

Faustregeln:

ist Rechenzeit am wichtigsten -> MaxPrime klein (3 oder 2), Deviation groß (30%-50%)

hat Genauigkeit Vorrang -> MaxPrime groß (20 - 30), Deviation klein (0-0.1%)

dir

Richtung in der gesucht werden soll (0 falls Argument ausgelassen)

-1 es werden nur Abweichungen von n nach unten betrachtet, also Werte kleiner als n

0 es werden Abweichungen ober- und unterhalb von n beachtet

1 es werden nur Abweichungen von n nach oben betrachtet, also Werte größer als n

Beispiele:

RoundPrimeSum (1s, 0.01, 1/(16726Hz), 2%)

Es soll ein Datensatz von 16726 Samples transformiert werden. Es wird versucht eine Zahl zwischen 16726*(1-2%)=16391 und 16726*(1+2%)=17061 zu finden, deren Primfaktorensumme besonders gering ist, auf jeden Fall aber kleiner als 16726*0.01 = 167. Falls das nicht gelingt, wird außerhalb des Bereichs nach so einer Zahl gesucht.

Postfix-Funktionen

^{2} Quadrat

^{3} Kubik

! Fakultät (nicht für komplexe Zahlen)

Klammern

() normale Klammerung

[] Floor()

|| Abs()

Prioritäten

()

Funktionen

^

* /

+ -

Intellenztest! Wie kann man mit Hilfe des Taschenrechners einen Wert von einer Einheit in eine andere Einheit umrechnen?

Antwort

Drag&Drop

Die Bedienung ist die gleiche wie beim normalen Stringgadget.

Zusätzlich dürfen auch Klangwahl-Gadgets über ein Formelgadget gezogen werden. Es wird dann der Pfad des Klanges an der Stelle des Cursors eingefügt.

Kontextmenü

Im Kontextmenü findet man einige oft gebrauchte Funktionen und Variablen. Gehört das Formelgadget zu einer Variablen eines Klanges, können im Kontextmenü auch häufig gebrauchte Formeln für diese spezielle Variable angegeben sein.

Tips

Wenn man es leid ist, die voreingestellten Ausdrücke immer aus dem Kontextmenü auszuwählen, das Formelgadget aber zu einer Sound-Variablen gehört, dann ist es auch möglich, das zur Variablen gehörige Eigenschaftenfenster zu öffnen und aus der dortigen Liste in das Stringgadget zu ziehen.

Da Assampler keine Funktion zur Verfügung stellt, um zu einer Frequenz den entsprechenden Ton zu bestimmen, muß man sich in diesem Falle anders helfen. Zuerst teilt man seine Frequenz durch eine Frequenz, deren Ton man kennt, z.B. Tone(c1). Das so gewonnene Frequenzverhältnis wandelt man in Halbtonschritte um, in dem man den binären Logarithmus zieht (ergibt Anzahl der Oktaven) und das mit zwölf multipliziert (da zwölf Töne pro Oktave). Macht zusammen einen Ausdruck wie "Log(Frequency/Tone(c1),2)*12". Die Halbtonschritte zählt man nun von c1 aus ab, wobei man bei größeren Anzahlen von Halbtönen immer zwölf auf einmal und damit ganze Oktaven überspringen kann.

Zukunft

Boolean-Typen, If und Vergleichsoperatoren

1.38 Intellenztest-Ablösung

Ach kommen Sie, so schwer ist das wirklich nicht! Aber wenn's gar nicht anders geht ...

1.39 Intellenztest-Auflösung

Sie dividieren einfach den Wert durch die Zieleinheit und erhalten einen einheitenlosen Wert. Wollen Sie Meter in Sekunden umrechnen, was natürlich nicht so ohne weiteres geht, zieht sich Ihr Amiga schlauerweise mit m/s aus der Affäre.

Um auch formal die Richtigkeit dieses Verfahrens überprüfen zu können, überlegen Sie sich folgendes: Der Taschenrechner zeigt nach Betätigen der <ENTER>-Taste im Eingabe-Gadget einen äquivalenten Wert im Ergebnis-Gadget an. Wollen Sie x Sekunden in y Minuten umrechnen, entspricht das der Gleichung

x s = y min

was gleichbedeutend ist mit

x * 1s = y * 1min

Sie können den Taschenrechner nicht dazu zwingen, einen Wert in einer bestimmten Einheit auszugeben, aber Sie können die Gleichung äquivalent umformen, indem Sie beide Seiten durch eine Minute dividieren:

x * 1s/(1min) = y

Beispiel: 120s/min = 2 bedeutet 120s = 2min

1.40 normales Stringgadget

Das Stringgadget für Eingabe von Worten/ganzen Zahlen

Aussehen

Sieht aus wie ein ganz normales StringGadget, ist auch nur ein ganz normales Stringgadget.

Funktion

Eigentlich gibt es hierzu nichts großartig neues zu berichten, in ein Stringgadget gibt man eben Zeichen ein. Mitunter wird die Eingabe von bestimmten Zeichen unterdrückt, z.B. für die Eingabe von ganzen Zahlen. Nun gelingt es aber trotzdem durch Drag&Drop unerlaubte Zeichen in das Stringgadget zu schleusen. Dann ist das weitere Editieren nahezu unmöglich, da sich nach dem Löschen eines einzelnen Zeichens immer noch verbotene Zeichen im String befinden können. Hier hilft nur das Löschen des komplette Gadgetinhalts mit Amiga-X weiter.

Tastatur

Amiga-X: Inhalt des Stringgadgets löschen.

Amiga-Q: Inhalt des Stringgadgets wiederherstellen.

Drag&Drop

Die Stringgadgets im Assampler unterstützen im allgemeinen Drag&Drop. Da die MUI-Stringgadgets auf den Original-Intuition-Gadgets aufbauen, welche kein Drag&Drop unterstützen, kann man das Stringgadget nur am Nicht-Intuition-Anteil des Gadgets aufgreifen. Das bedeutet, daß man Stringgadgets immer am Rand anklicken muß, was oft etwas Übung und Geduld (und einer niedrigen Bildschirm-Auflösung :-) bedarf.

1.41 PopUp-Gadget für Klangauswahl

Das PopUp-Gadget für die Klangauswahl

Aussehen

Es handelt sich hier um ein PopUp-Gadget und das enthält ein StringGadget und einen Knopf, mit dem der Baum aller Klänge zur Auswahl ausgeklappt wird.

Funktion

So wählen Sie einen Klang aus: Knopf neben dem Stringgadget betätigen und auf einen Eintrag der Liste doppelt klicken. Haben Sie die Liste aufgeklappt, einen Eintrag jedoch erst einmal angeklickt, können Sie Ihre Auswahl noch verwerfen, in dem Sie nochmals auf den PopUp-Knopf drücken.

Drag&Drop

Sie können das Gadget überall hin ziehen, wo Klangverweise oder Zeichenketten eingetragen werden können. Umgekehrt kann es als Ziel einer Drag&Drop-Aktion dienen und übernimmt dann den Verweis des gezogenen Bedienelementes. Damit ergibt sich eine besonders bequeme Möglichkeit, Klangverweise einzustellen.

Tastatur

Mit der TAB-Taste wählen Sie zwischen Stringgadget und PopUp-Knopf.

Mit <ENTER> auf den PopUp-Knopf öffnen Sie das Auswahlfenster.

Mit den Cursortasten suchen Sie einen Klang aus und bestätigen die Wahl mit <ENTER> oder verwerfen Sie mit <ESC>.

Bei aufgeklapptem Auswahlfenster können Sie die ersten Buchstaben des Namens des gesuchten Klanges in das Stringgadget eintragen und Assampler sucht Ihnen den ersten passenden Klang heraus.

Kontextmenü

Öffne Parameterfenster

Öffnet ein Fenster, mit dem die Werte des ausgewählten Klanges allein für diesen Aufruf abgewandelt werden können. Zwischen Variablenname und Variableninhalt befindet sich jeweils ein Schalter, der bestimmt, ob ein Wert verändert werden soll oder nicht. Geschlossen wird das Fenster mit dem Schließsymbol. Änderungen können nicht verworfen werden.

Parameter löschen

Löscht alle Parameter. Ist das Parameterfenster zu dieser Zeit geöffnet, bleiben die dort eingestellten Parameter erhalten.

Es können immer auch Parameter vorhanden sein, die man nicht sieht. Beispiel: Sie stellen einen Oszillator ein und öffnen das Parameterfenster, und setzen den Parameter Scale neu. Dann wählen Sie einen Rauschgenerator. Die Variable Scale kennt der Rauschgenerator nicht und er zeigt sie nicht an. Sollten Sie wieder einen Oszillator einstellen, kann man den Scale-Parameter wieder sehen.

Öffne Anzeigefenster

Öffnet für den eingestellten Klang ein Anzeigefenster.

Öffne Informationsfenster

Öffnet für den eingestellten Klang ein Informationsfenster.

1.42 größenverstellbares Proportionalgadget

Das größenverstellbare Proportionalgadget

Aussehen

Die Tarnung ist (fast) perfekt. Es sieht aus, wie ein normales Proportionalgadget, aber nicht jedes Proportionalgadget ist auch größenverstellbar!

Funktion

Neben den normalen Proportionalgadget-Funktionen, kann man durch Drücken einer bestimmten Umschalttaste (voreingestellt ist Ctrl) während eines Mausklicks folgende Funktionen erreichen:

Balken anklicken und ziehen

Bewegen nach links/oben verkleinert Balken, Bewegen nach rechts/unten vergrößert Balken.

Neben Balken klicken

Links/oben daneben klicken halbiert Balken, rechts/unten daneben klicken verdoppelt Balken.

Einstellungen

1.43 Variablenlisten

Variablenlisten

Aussehen

Variablenlisten werden durch virtuelle MUI-Gruppen dargestellt und darunter befinden sich zwei Knöpfe "Neu" und "Löschen".

Funktion

Variablenlisten dürfen beliebig viele Variablen einer vorgegebenen Anordnung erhalten.

Mit "Neu" wird ein neuer Eintrag am Ende eingefügt, mit "Löschen" wird der letzte gelöscht.

Drag&Drop

Zieht man die Nummer eines Eintrages auf "Neu", wird der entsprechende Eintrag mitsamt Inhalt verdoppelt. Zieht man eine Eintragsnummer über "Löschen" wird dieser Eintrag gelöscht. Zieht man eine Nummer über eine andere wird der Inhalt des gezogenen Eintrages kopiert. Eintragsnummern dürfen auch auf Eintragsnummern und "Neu"-Knöpfe anderer Variablenlisten gezogen werden.

Zur Vereinfachung der Pfadeingabe dürfen Eintragsnummern und Spaltenbezeichnungen in Zahlengadgets gezogen werden. Dabei empfiehlt sich die Reihenfolge: Erst Eintragsnummer, dann Spaltenbezeichner.

1.44 Nummern in Variablenlisten

Nummern in Variablenlisten

Aussehen

Sehen aus wie unscheinbare Texte.

Funktion

Die Nummern repräsentieren den Inhalt der Variablen einer Zeile.

Drag&Drop

Siehe Variablenlisten.

1.45 Tastenkürzel

Tastenkürzel

Die folgenden Informationen (welche sich auf das wichtigste beschränken) stehen garantiert irgendwo in Ihren Handbüchern, die Sie möglicherweise noch keines Blickes gewürdigt haben.

Aufbau der Tasten-Beschreibung:

[Umschalttasten] [Zeichen]

Umschalttasten können sein:

lshift linke Shift-Taste

rshift rechte Shift-Taste

shift irgendeine Shift-Taste

capslock Caps Lock

caps irgendeine Shift-Taste oder Caps Lock

control Ctrl-Taste

lalt linke Alt-Taste

ralt rechte Alt-Taste

alt irgendeine Alt-Taste

lcommand linke Amiga-Taste

rcommand rechte Amiga-Taste

rbutton rechte Maustaste (kann man also auch als Umschalttaste betrachten)

lbutton linke Maustaste

Zeichen können sein:

Einzelne Buchstaben oder Ziffern oder Schlüsselworte wie

space (Leerzeichen), backspace, del (Zeichen links bzw. rechts löschen), enter (Eingabetaste), f1 - f10 (Funktionstasten)

Vermeiden Sie Tasten, die schon durch MUI vorbelegt sind (in den MUI-Preferences konfigurierbar) wie Tabulator, Help, Cursortasten etc.

Beispiele:

"p" Taste p

"lshift control p" Taste p bei gehaltener linker Shift-Taste und Control-Taste

"alt f5" Funktionstaste 5 bei gedrückter Alt-Taste (egal welche)

1.46 Glossar hat nix mit Glosse zu tun

Glossar

Abtastfrequenz

ADSR-Generator

Amplitude

Auslenkung

Bitanzahl

Elongation

Flanging

FM-Synthese

Frequenz

Interpolation

Kohärenz

Lautstärke

LFO

Loop

Oberwellen

Periode

Phasenschieber

Puffer

Repeat

Ringmodulation

Rückfaltung

Sampleformate

Samplerate

Schwebung

Schleife

Spektrum

Tremolo

Vibrato

Wortbreite

1.47 ADSR-Generator

ADSR-Generator

Mit einem ADSR-Generator kann man Naturinstrumenten nachempfundene Hüllkurven erzeugen. ADSR steht für Attack-Decay-Sustain-Release (frei übersetzt: Anschlag-Nachlassen-Halten-Loslassen) und bezeichnet die typischen Phasen im Lautstärkeverlauf vieler Instrumente. Jede Phase wird mit einem Parameter charakterisiert, so jedenfalls ist es im (analogen) Ur-ADSR-Generator. Beachten Sie dabei, daß der ADSR auf Live-Synthesizern an einen Tastendruck gekoppelt ist.

Attack (Zeitspanne) Eine Taste wird gedrückt. Die Lautstärke steigt von 0 auf Maximum innerhalb der Attack-Zeitspanne an.

Decay (Zeitspanne) Danach nimmt die Lautstärke ab, und nähert sich innerhalb der Decay-Zeit dem ...

Sustain (Pegel). Dort wird Lautstärke gehalten, bis die Taste losgelassen wird.

Release (Zeitspanne) Nach Loslassen der Taste (in irgendeiner Phase) wird die Release-Phase eingeschlagen, und die Lautstärke fällt innerhalb der Zeitspanne auf null zurück.

Beispiele

Percussive Instrumente zeichnen sich dadurch aus, daß man ihren Ton nicht halten kann. Bei einer Trommel ist das leicht einzusehen, aber auch beim Klavier ist das so. Deshalb fehlt dort die Sustain-Phase, weiter sind Attack-Phase extrem kurz und Decay-Phase recht lang, länger als Release. Wird die Taste angeschlagen, ist der Ton praktisch sofort da und klingt auch wieder ab (Decay), wird die Taste losgelassen verstummt der Ton noch schneller (Release < Decay).

Bei Blasinstrumenten ist der Ton beim Anblasen typisch. Dort sind Attack- und Decay-Phase deutlich vernehmbar, danach kann der Ton gehalten werden (solange die Luft reicht). Die Ausklingphase ist wieder recht kurz.

Der Assampler besitzt wohlweißlich keinen ADSR-Prozeß, denn diese Hüllkurvenform läßt sich ohne weiteres mit den vorhanden Mitteln nachbilden. Das Beispiel ADSR.spline tut dies mit Hilfe eines Splines, ADSRExp.algo tut dies durch Zusammensetzen aus Exponentialkurven und ADSRLin.algo erledigt das mit linearen Übergängen.

1.48 LFO

Der Niederfrequenz-Oszillator (LowFrequencyOscillator)

Der LFO ist hauptsächlich ein Relikt aus Analog-Synthesizer-Zeiten, wo für unterschiedliche Frequenzbereiche unterschiedliche Schaltungen gebraucht wurden. Auf Digitalrechnern können die Aufgaben des LFO von Oszillatoren mit niedrigen Frequenzen erledigt werden.

LFO sollen Steuerkurven generieren, wie sie für periodische Vorgänge gebraucht werden, z.B. Vibrato- und Tremoloeffekt.

1.49 Wortbreite

Die Wortbreite

Wenn man mit Wort (= die vom Prozessor auf einen Rutsch verarbeitbare Informationsmenge) einen Sample meint, bedeutet die Anzahl der Bits (=Wortbreite) das Auflösungsvermögen in y-Richtung. Je größer die Anzahl der Bits, desto feiner die Auflösung, desto besser die Qualität.

Ein 8-Bit Sample repräsentiert eine 8-stellige Binärzahl, mit welcher man zwischen 256 verschiedenen Werten unterscheiden kann. Oder anders ausgedrückt, beim Digitalisieren mit 8 Bit wird der Bereich von größter bis kleinster Auslenkung in 256 (meist gleichgroße) Teile eingeteilt.

Übersicht über gebräuchliche Wortbreiten:

Bits Anzahl Unterteilungen

8 256

12 4096

16 65536

24 16777216

32 4294967296

Sie sehen, daß eine Verdoppelung bereits zur Quadrierung der Anzahl der Teilintervalle führt. Trotzdem ist das keine überdimensionale Erhöhung, denn unser Ohr empfindet die Lautstärken ebenfalls logarithmisch, d.h. jedes weitere Bit erweitert den Dynamikbereich (also der Unterschied zwischen lautest und leisest möglichem Geräusch) um einen konstanten Wert (etwa 3 dB).

Es bleibt zu bemerken, daß ein maximal ausgesteuerter Sample-Sound in 8 Bit vom Ohr bereits nicht mehr vom Original unterschieden werden kann, sofern die Abtastrate hinreichend groß ist (40 kHz reichen). Warum dennoch die Jagd auf immer höhere Bitbreiten anhält, hat verschiedene Gründe:

1. Irgendwie müssen die Festplattenhersteller ihre 30 GB-Exemplare loswerden.

2. Enthalten die Daten von vornherein viel Redundanz, können sich die Packalgorithmen mit atemberaubenden Packraten schmücken.

3. Ist der Sample-Sound schlecht ausgesteuert oder enthält große Lautstärkeschwankungen (klassisches Beispiel Ravels Bolero), bleibt immer noch genügend Genauigkeit für die leisen Stellen. Denn ist der lauteste Teil gut ausgesteuert, muß sich der leiseste Teil praktisch mit weniger Bits begnügen.

4. Wird der Sound viel bearbeitet, können die Rundungsfehler gering gehalten werden.

5. Werden die Sample-Sounds nicht nur zur Wiedergabe verwendet, sondern auch als Steuerkurven, reicht die 16 Bit-Genauigkeit des Assamplers manchmal nicht. Beispiel: Bei der Phasenmodulation kann man böse Einbrüche erleben, wenn die Modulationstiefe mehrere Sekunden umfaßt.

Aus dieser Erkenntnis heraus ist es naheliegend in Zukunft verstärkt Fließkommazahlen einzusetzen. Auf manchen Systemen (vom Archimedes weiß ich es mit Sicherheit) sind die möglichen Samplewerte exponentiell gestaffelt. Das erschwert allerdings schon einfache Operationen wie das Mischen von Klängen.

Der andere Eckwert, der über die Qualität eines Sample-Sounds Auskunft gibt, ist die Abtastfrequenz.

1.50 Abtastfrequenz

Die Abtastfrequenz

Formelzeichen f, Einheit [f] = 1 Hz (Hertz) = 1 / s (ein Sample pro Sekunde)

Die \{Sample, Abtast\} $\times$ \{rate, frequenz\} (soll heißen, daß alle vier Kombinationen gebräuchlich sind) ist wie die normale Frequenz ein Maß für die zeitliche Häufigkeit.

Hiermit wird die Anzahl der Probenahmen (engl. Samples) pro Zeiteinheit und damit das Auflösungsvermögen auf der Zeitachse festgelegt. Eine höhere Abtastrate garantiert hierbei höhere Qualität und schneller gefüllte Festplatten.

Die bei CDs verwendete Abtastrate von 44.1 kHz ist deshalb ausreichend, weil der damit höchste darstellbare Ton mit 22 kHz gerade nicht mehr hörbar ist.

Aufgrund des Rückfaltungseffekts ist bei einem Sample-Sound der höchste darstellbare Ton auf die Hälfte der Abtastfrequenz beschränkt. (Shannonsches Abtasttheorem)

Der andere Eckwert, der über die Qualität eines Sample-Sounds Auskunft gibt, ist die Bitanzahl. Interessant ist übrigens, daß man eine noch so geringe Wortbreite (0 ausgenommen), durch Erhöhung der Abtastfrequenz wettmachen kann. Man kann dann das träge Ohr durch Pulsweitenmodulation austricksen. Andersherum kann man verlorene Abtastauflösung nicht mit höherer Wortbreite zurückkaufen.

1.51 Sample-Formate

Sampleformate

Ja, ja, die ewige Plage mit diesen Formaten. Es geht jetzt also um die Frage, in welcher Form man die Daten aus dem Computer auf Disketten bringt. Dabei stellt man fest, daß die Formate auf Diskette sehr stark von der Datenspeicherung im Computer abhängen. Das ist wohl auch einer der vielen Gründe, warum die (meist sehr praktischen IFF-) Amiga-Formate auf anderen Plattformen kaum beachtet werden. Dabei bieten die vielen PC-Formate oft nichts wesentlich neues, sondern sind einfach nur anders, wie wir gleich sehen werden.

Grob können wir erst einmal unterscheiden zwischen Formaten mit Zusatzinformationen und Roh-Formaten.

Die Rohsampleformate zeichnen sich dadurch aus, daß sie keine weiteren Daten als die reinen Samples enthalten, also alles das, was man beim Assampler in den Anzeigefenstern zu sehen bekommt. Da kann man eigentlich nicht viel verkehrt machen, wie? Trotzdem gibt es schon hier einige Variationsmöglichkeiten:

Variieren kann die Wortbreite, beim Assampler werden 8 Bit und 16 Bit unterstützt. Die Wortbreite ist ein Maß für die Qualität des Sample-Sounds, also noch eine sinnvolle Option. Was wir dann noch haben, ist bei 16 Bit (oder höher) die Frage in welcher Reihenfolge die Bytes eines Wortes (= Samples) abgespeichert werden sollen. Beim Amiga kommt das Big-Endian zum Einsatz, weil das die MC680x0-CPUs verarbeiten. Stellen Sie es sich so vor, daß ein Sample mit dem Wert 101997 (hundert und eintausend neunhundert sieben und neunzig) in Bytes aufgespaltet als 10, 19, 97 abspeichert wird. Intel-Prozessoren (PCs) mögen's genau umgekehrt (little Endian) 97, 19, 10.

Ein weiterer Schlag gegen die Einfachheit ist die Frage des Vorzeichens: Amigas benutzen aufgrund der Soundhardware und weil sich's einfacher damit rechnen läßt, den Bereich von -128 - +127 (8 Bit) bzw. -32768 - +32767 (16 Bit) für die Samples. Liegt kein Signal an, sind alle Samples = 0. Genau entgegengesetzt PCs mit den Bereichen 0-255 und 0-65535, wobei 128 bzw. 32768 die Ruhepegel kennzeichnen.

Verrückt wird's, wenn in den Rohdaten mehrere Kanäle untergebracht sind. Entweder sind die Sounds der Kanäle einfach aneinandergereiht, oder aber die Samples der Kanäle sind verzahnt: Erst kommen die ersten Samples jeden Kanals, dann die zweiten usw. Die Mühe, mehrere Kanäle aus einer Rohformat-Datei zu lesen, macht sich Assampler gar nicht erst.

Nun lassen Rohformate aber noch einige Wünsche übrig. Um nach dem Digitalisieren von Klängen später alles reproduzieren zu können, müßte man wenigstens die Abspielfrequenz wissen. Und wenn man schon mal dabei ist, werden gleich Schleifen-Informationen, Copyright-Vermerke, Vereinbarungen über Packalgorithmen, mehrere Kanäle usw. abgelegt. Beim Amiga bieten sich dafür die IF-Formate an, speziell die Soundformate 8SVX (8 Bit) und 16SV (16 Bit). Alle anderen Formate werden vom Assampler über Datatypes unterstützt.

1.52 Amplitude

Die Amplitude oder auch Lautstärke

Formelzeichen ˆy (das Dach gehört über das y), Einheit [y] = 1 m (ein Meter - gilt zumindest in der Mechanik)

An sich kann man Amplitude und Lautstärke als das gleiche betrachten.

Bei einer reinen Sinus-Schwingung bezeichnet die Amplitude den Maximalwert der Schwingung, bildlich gesprochen die Höhe der Wellenberge. Sobald die Sinusschwingung Amplituden-moduliert wird oder auf andere Weise überlagerte Schwingungen hinzukommen, gibt es keine stichhaltige Definition für die Amplitude.

Definiert man die Amplitude weiterhin als Maximalwert, müßte man einen Zeitraum festlegen, über dem der Maximalwert bestimmt wird - das ist aber reine Ermessensfrage. Hat man eine nadelimpulsförmige Schwingung klingt diese leiser als eine Rechteckschwingung gleichen Maximalwertes. Sollte man vielleicht lieber die Fläche unter der Kurve als Maß nehmen? Bliebe trotzdem die Frage nach dem Zeitraum. Gar nicht beachtet wurde bisher, daß das Ohr besonders tiefe und besonders hohe Frequenzen als leiser empfindet, als die in einem mittleren Bereich um 1kHz, in der auch die menschliche Sprache angesiedelt ist.

Sie stellen fest, daß der Lautstärkeverlauf im Nachhinein nicht mehr eindeutig von einem Klang bestimmt werden kann. Deshalb stellt der Assampler auch keinen ultimativen Lautstärke-Detektor zur Verfügung, sondern läßt Ihnen die Wahl, wie Sie an eine Hüllkurve herankommen. Bewährt hat sich zum Beispiel eine Kombination aus Entfernen des konstanten Offsets, Gleichrichter und Tiefpass-Filter.

Gerne wird die Amplitude mit der Elongation verwechselt.

Siehe auch: Verstärker

1.53 Elongation

Die Elongation oder auch Auslenkung

Formelzeichen y, Einheit [y] = 1 m (ein Meter - gilt zumindest in der Mechanik)

Eine Sinus-Schwingung durchläuft in einer Periode mehrere Phasen. Von null auf Maximum dann wieder auf null, dann zum Minimum und zurück zur null. Der Zustand zu einem Zeitpunkt betrachtet nennt man Auslenkung (den Zeitpunkt relativ zum Beginn der Schwingung übrigens Phase). Das gilt auch für alle anderen Schwingungen, auch nicht-periodische.

Fälschlicherweise wird die Auslenkung oft mit der Amplitude gleichgesetzt, oder es wird gar der Begriff Momentanamplitude eingeführt. Stellen Sie sich mal vor, ich picke aus einer Schwingung den Nulldurchgang heraus und behaupte die Schwingung hätte an dieser Stelle die Lautstärke null! Oder hören Sie sich einen Sample-Sound an, der komplett mit dem Maximalwert gefüllt ist. Sie hören die ganze Zeit nichts (höchstens ein Knacken zu Beginn und Ende, wenn die Playerroutine zwischen Nullpegel und Samples umschaltet), weil keine Schwingung verhanden ist. Die vermeintliche Lautstärke ist hingegen maximal!

1.54 Frequenz

Die Frequenz

Formelzeichen f, Einheit [f] = 1 Hz (Hertz) = 1 / s (eins pro Sekunde)

Die Frequenz gibt an, wieviel Schwingungen pro Zeiteinheit von einem Schwinger ausgeführt werden. Bei Tönen aus dem Lautsprecher ist es eine Membran, die schwingt - je schneller, desto höher die Frequenz, desto höher der Ton.

Zur Orientierung: Menschen können Frequenzen etwa von 50 Hz bis 20 kHz wahrnehmen. Alles darunter ist Infraschall (z.B. Meereswellen) alles darüber ist Ultraschall (z.B. von Fledermäusen zur Ortung der Beute verwendet). Gemeint sind mit den Frequenzangaben reine Sinusschwingungen. Eine Rechteckschwingungen zu 50Hz würden Sie sehr wohl knattern hören, im Grunde hören Sie aber nur die Oberwellen. Aber ab 20 Hz abwärts empfinden Sie auch das Knattern nicht mehr als Ton - das Gehör findet einfach den Zusammenhang nicht mehr.

Man meint ebenfalls immer sinusförmige Schwingungen, wenn man davon spricht, daß ein Ton bestimmte Frequenzen enthält.

Siehe auch: Periode Spektrum

1.55 Periode

Die Periode

Formelzeichen T, Einheit [T] = 1 s (eine Sekunde)

Die Periode ist eine Zeitspanne. Der Name Periode deutet weiter darauf hin, daß sich ein Vorgang wiederholt. Bei einer Schwingung ist das der Fall, und wie mit der Frequenz auch, läßt sich mit der Periode die Tonhöhe beschreiben. Je größer die Periode, desto niedriger die Frequenz, um so tiefer der Ton. Es gilt die umgekehrte Proportionalität: f=1/T.

Gerade bei langsameren Schwingungen ist die Periode gebräuchlicher als eine Frequenzangabe, z.B. ist die Periode eines Erdumlaufs um die Sonne ein Jahr (1 a) und die Frequenz 1/(3600*24*365.2422) Hz ...

1.56 Frequenzspektrum

Das Frequenzspektrum

Das Frequenzspektrum gibt an, welche Frequenz mit welcher Amplitude in einem Klang vertreten ist. Das hilft, Eigenschaften eines Klanges sichtbar zu machen, die in der normalen Darstellung (Auslenkung eines Schwingers abgetragen über der Zeitachse) nicht sofort zu sehen sind. Gut, man kann in der normalen Signaldarstellung mit etwas Übung nicht nur laut und leise unterscheiden, sondern auch Klänge mit tiefen und hohen Frequenzen (scharfe Kanten und kurze Schwingungen oder langgezogene Kurvenzüge), aber im Frequenzspektrum sieht man es noch besser. Aber auch da darf man nicht erwarten, ablesen zu können, wie genau ein Sound klingt.

Neben der Vermittelung eines optischen Eindrucks von dem Klang, ist das Frequenzspektrum auch unschätzbare Hilfe für Manipulationen am Klang. Während man sich in der Analogtechnik schwer tut, Filter zu konstruieren, die vorgegebene Frequenzanhebungen/abschwächungen nebst Phasenverschiebungen möglichst genau bewältigen, kann man am Frequenzspektrum einfach die Bereiche herauslöschen, die stören oder welche verstärken, die herausgehoben werden sollen.

Ich habe übrigens bewußt nicht diese Einleitung gewählt: "Klänge bestehen bekanntlich aus Sinusschwingungen ..." In der Tat steht es völlig in den Sternen, woraus Klänge "aufgebaut" sind, es ist lediglich zweckmäßig, sie sich aus Sinusschwingungen aufgebaut vorzustellen, da das der Art und Weise wie wir hören am nächsten kommt. Ich hätte aber genauso recht, wenn ich behaupten würde, Klänge bestünden aus Potenzfunktionen (=Polynomen) oder aus Rechteckschwingungen. Allerdings muß ich einräumen, daß Sinusschwingungen bzw. Exponentialfunktionen im Komplexen eine mathematische Sonderrolle einnehmen, weil Sie die Eigenfunktionen von linearen zeitinvarianten Systemen sind.

Wie sehen nun gängige Klänge im Frequenzspektrum aus? Bei gleichbleibenden Tönen sind es Peaks, die sich in regelmäßigen (Frequenz-)Abständen wiederholen und dabei immer schwächer werden. Der erste Peak kennzeichnet die Grundfrequenz, die folgenden Peaks die Oberwellen, das heißt Sinusschwingungen mit einem ganzzahligen Vielfachen der Grundfrequenz. Diese Oberwellen geben dem Ton seine charakteristische Klangfarbe. Stark vertretene Oberwellen geben einen scharfen, quäkigen, spitzen (oder was einem noch so an Attributen einfällt) Klang, wenig Oberwellen bedeuten sanfte, dumpfe etc. Klänge. Interessanterweise braucht auch gar kein Peak bei der Grundfrequenz zu sein. Wenn nur genügend viele Oberwellen da sind, hören wir trotzdem einen Ton der Grundfrequenz. Deswegen auch meine vorige Bemerkung zu den aus Sinusschwingungen aufgebauten Klängen.

Bei Rauschen sieht das Frequenzspektrum genauso aus wie das zeitliche Signal, nämlich wieder Rauschen. Aber durch Filtern läßt sich auch Rauschen Charakter verleihen, was sich dann in herausgehobenen Frequenzbändern äußert.

1.57 Rückfaltung

Die Rückfaltung

Dieses Phänomen hat jeder schon mal optisch wahrgenommen, z.B. wenn sich die Rotorblätter eines Hubschraubers auf einmal rückwärts zu drehen scheinen, obwohl sie immer schneller werden. Wenn man einen Drehvorgang nur in bestimmten Abständen beobachtet, und bei aller Technik und auch dem menschlichen Gehirn ist das so, wenn auch die Abstände recht kurz sind, ist nicht klar, ob sich ein Gegenstand von einer Beobachtung zur nächsten um einen achtel Kreis nach links oder um einen sieben achtel Kreis nach rechts gedreht hat. Normalerweise vermutet man die kleinere Variante, aber Gegenstände können sich durchaus wirklich so schnell drehen.

Gleiches gilt bei Audiosignalen: Ihnen wird nicht das Kunststück gelingen, in einem 16kHz-Sample-Sound einen Ton von 20kHz unterzubringen, nicht einmal 10kHz! Überlegen Sie sich, daß ein Schwinger immer zwischen zwei Extremwerten pendelt. Die Abtastung muß wenigstens zu den Zeitpunkten der Extremwerte erfolgen, besser aber öfter. Die schnellstmögliche Schwingung, die man in einem Sample-Sound darstellen kann, sieht demnach so aus: +1, -1, +1, -1, +1. Auf die Weise kommt man darauf, daß zum Speichern eines Tones einer bestimmten Frequenz mindestens die doppelte Frequenz als Abtastrate verwendet werden muß. Beachtet man das nicht, erhält man wieder Töne tieferer Frequenz.

Siehe auch: Abtastrate

1.58 Vibrato + Tremolo

Der Vibrato- und der Tremoloeffekt

Ich möchte beide Effekte in einem Punkt abhandeln um die Unterschiede besser aufzeigen zu können. Oh, was für Wortgefechte habe ich mir mit ausgebildeten Musikern geliefert - nicht einmal vor Schlag- und Stichworten schreckten wir zurück.

Beim Vibratoeffekt sind wir uns alle einig, daß es sich hierbei um eine periodische Frequenzmodulation handelt. Extrem hören wir das bei einer Sirene, aber wenn die Modulationsperiode kürzer und die -tiefe geringer ist, kann der Effekt ganz angenehm klingen. Beim Violinenspiel ist er sehr beliebt, und auch bei Gesang häufig anzutreffen, wenn ein lang anhaltender Ton etwas belebt werden soll. (Streng genommen vibrieren Gegenstände zum Beispiel bei einem Erdbeben gar nicht, sondern oszillieren, aber vergessen Sie das lieber wieder.)

Beim Tremoloeffekt hingegen scheiden sich die Geister. Musiker verstehen darunter das Chaos, das sich eines Tones bemächtigt, wenn die Melodie zu einem Höhepunkt gelangt, abstrakt: eine Amplituden- und eine Frequenzmodulation. Wir wollen, konform zur Synthesizerliteratur, Tremolo als eine reine periodische Amplitudenmodulation ansehen. Es ist in etwa der Effekt, wenn Sie sich die Ohren im Wechsel zuhalten und wieder loslassen. Oder aber wenn jemand auf einem Drehstuhl rotiert und dabei einen langanhaltenden Ton von sich gibt.

1.59 FM Synthese

Die FM-Synthese

Die FM-Synthese wurde von der Firma Yamaha entwickelt und zeichnet sich dadurch aus, daß man schon mit einem kleinen Satz an Bausteinen eines einzigen Typs (FM Operatoren) durch Variation der Operatorenverknüpfung und deren Parametern bereits eine sehr große Klangvielfalt erreicht.

Beim heute legendären DX7 setzte Yamaha (soweit ich weiß) diese Synthesemethode das erste Mal ein und man findet sie auch heute noch auf (vorwiegend PC-)Soundkarten zur speichersparenden Klangerzeugung. Wenn Sie mal wieder einen dieser quäkigen metallischen Sounds aus einem PC hören, wissen Sie woher das rührt. - Ein interessantes Verfahren muß ja noch lange nicht richtig eingesetzt werden.

FM bedeutet (bevor ich's vergesse) Frequenzmodulation und ist damit auch schon wieder ziemlich irreführend, denn eigentlich handelt es sich um eine Phasenmodulation. Das funktioniert so: ein Operator kann einen anderen modulieren (im Prinzip zeitlich verzerren). Gibt ein modulierender Operator einen großen Signalwert aus, wird das Signal des modulierten Oszillators verzögert, oder im umgekehrten Fall vorgeschoben.

Da ein Modulator oft mit einer Frequenz in der gleichen Größenordnung wie der modulierte Oszillator schwingt, können die Schwingungen des Oszillators verzerrt und dadurch verschiedene Klangfarben erzeugt werden.

Variationsmöglichkeiten ergeben sich durch die Verknüpfung von Operatoren (Modulatoren können selbst wieder moduliert werden usw.), Modulationstiefe, Hüllkurven, Grundwellenformen, Rückkopplungen.

Achso: Wo war jetzt der Unterschied von Frequenz- zu Phasenmodulation? Bei der Frequenzmodulation gibt der Wert des Steuersignals die Änderung der Frequenz des Tongenerators vor. Ist das Steuersignal konstant aber ungleich null, erhält man durch die Modulation eine andere Frequenz als die Grundfrequenz. Da man mit der Modulation eigentlich nur die Klangfarbe ändern will, ist die Frequenzänderung störend. Sie tritt aber genauso auf, wenn das modulierende Signal einen Gleichspannungsanteil enthält, der nicht immer erkenntlich sein muß.

Deshalb kommt in der Tat eine Phasenmodulation zum Einsatz, hier kann das Steuersignal das Ausgangssignal des modulierten Oszillators nur verzögern - eine konstante Verzögerung ändert den Ton zum Beispiel nicht.

1.60 Ringmodulation

Ringmodulation

Was die Ringmodulation ist, kann ich Ihnen noch nicht genau sagen. Zwei verschiedene Erklärungen sind mir geläufig:

1. Es werden die Werte von zwei Eingangssignalen miteinander multipliziert.

2. Irgendeine Konstruktion aus Dioden, die Ringform besitzt aber auf jeden Fall nicht multipliziert.

Für einen Effekt braucht man also zwei Signale, die bezüglich des Ringmodulators gleichberechtigt verarbeitet werden. Beide Signale sollten reine Töne sein, möglichst keine Tongemische oder Rauschen. Dann kann man die Frequenz oder Wellenform der eingegeben Töne variieren, um hörenswerte Ergebnisse zu erhalten.

1.61 Was Du heute kannst verschieben, das verschiebe nicht erst morgen

Der Phasenschieber/Phasing/Flanging-Effekt, Schwebungen

Machen wir einmal folgenden Versuch: Sie nehmen eine Sinus-Schwingung zu 500 Hz und eine zu 501 Hz und mischen beide zusammen. Diese Frequenzen der Töne liegen so eng zusammen, daß man sie kaum unterscheiden kann. Beim Mischen tritt ein Effekt auf, den Sie vielleicht noch aus dem Physik-Unterricht als Schwebungen kennen: Der Ton schwillt an und ab.

Was ist passiert? Stellen Sie sich beide Schwingungen untereinander gezeichnet vor, oder wenn Sie sich es nicht vorstellen können, nehmen Sie irgendeinen Soundprozessor, sagen wir ... Assampler. Vergleichen Sie nun, wie am Anfang Wellenberge und Wellenberge gegenüberstehen und sich mit fortschreitender Zeit die Wellenberge verschieben, so daß sie irgendwann Wellentälern gegenüberstehen. Addiert man zwei positive Werte erhält man einen größeren postiven Wert, addiert man betragsgleiche entgegengesetzte Zahlen, erhält man Null. Analog verstärken sich Wellenberge mit Wellenbergen, Wellenberge und Wellentäler hingegen löschen sich aus.

Dieser Vorgang wiederholt sich und bringt regelmäßige Schwankungen der Lautstärke hervor. Je größer die Frequenzdifferenz, desto kürzer die Schwankungsperiode. Verwendet man mehrere Töne mit ähnlicher Frequenz, kann man auch komplexere Schwankungsmuster erzeugen. Umgekehrt kann man sagen, das jede ungleichmäßige Verstärkung Frequenzen verändert oder neue erzeugt. Wer's nicht glaubt, kann einen Sample-Sound einer Frequenzanalyse à la Fourier unterziehen und er wird feststellen, daß um jede Frequenz herum andere existieren, die für den Lautstärkeverlauf verantwortlich sind.

Interessant wird der Phasing-Effekt, wenn man nicht nur wie im Schulunterricht Sinusschwingungen überlagert sondern andere Wellenformen. Deren Oberwellen machen genau die gleichen Schwebungen durch, nur in einem anderen Takt. Die n. Oberwelle genau n-mal so schnell, rechnen Sie's nach (siehe unten). Das dabei entstehende komplexe Frequenzmuster macht einen Klang auf einfache Weise lebendiger. Gibt man beide Töne getrennt über zwei Lautsprecher aus, wird der Klang erst durch unser Gehör zusammengesetzt. Aus den ständig wechselnden Phasenbeziehungen versucht unser Gehirn Entfernungsinformationen zu gewinnen - der Klang wirkt raumfüllender.

Mischt man zwei Töne der Frequenzen f1 und f2 erhält man

Sin (f1*t) + Sin (f2*t) = 2 * Cos (½(f1-f2)*t) * Sin (½(f1+f2)*t)

Dieses Additionstheorem erhält durch Betrachtung des Imaginärteils der folgenden Beziehung:

Exp (i a) + Exp (i b) = Exp (i (a-b)/2) * Exp (i (a+b)/2) + Exp (i (b-a)/2) * Exp (i (a+b)/2)

= (Exp (i (a-b)/2) + Exp (-i (a-b)/2)) * Exp (i (a+b)/2)

= 2 * Cos (a-b)/2 * Exp (i (a+b)/2)

Das bedeutet für uns, daß das Mischen zweier gleich lauter Sinusschwingungen wieder zu einer Sinusschwingung führt, die maximal doppelt so laut ist, die Durchschnittsfrequenz beider Originalfrequenzen besitzt und durch eine Cosinus-Schwingung der halben Frequenzdifferenz Lautstärke-moduliert ist.

Flanging ist ähnlich zum Phasing. Was genau Flanging von Phasing unterscheidet, mag keiner so recht definieren. Die einen betrachten es als Flanging-Effekt, wenn die eine Schwingung das negierte Gegenstück zu der anderen ist, z.B. bei der Sägezahn-Wellenform ist da ein Unterschied. Andere sind aber der Meinung, daß ein Flanger ein Phaser mit Rückkopplung sei, daß würde dann zu einem rekursiven Filter führen.

Siehe auch: Verstärker, Filter

1.62 Interpolation

Interpolation

Manche Prozesse benötigen zuweilen Sample-Werte zwischen den wirklich vorhandenen. Diese muß sich das Programm dann aus den Fingern saugen. Im Assampler sind drei Abstufungen dieses Schätzvorganges implementiert:

1. Keine Interpolation (nullter Grad): Es wird einfach der vorhergehende Sample-Wert benutzt. Verursacht Klirrgeräusche.

2. Lineare Interpolation (erster Grad): Es wird ein gewichtetes Mittel vom vorhergehenden und nachfolgenden Sample-Wert angenommen. Verhindert Klirrgeräusche, schleift das Signal dafür mehr ab, d.h. Sound verliert an Brillianz.

3. Kubische Interpolation (dritter Grad): Es wird eine kubische Funktion durch zwei aufeinanderfolgende Samples gelegt, und zwar so, daß beim Zusammensetzen der einzelnen Kurvenstücke glatte (=differenzierbare) Übergänge entstehen. Das ist richtig gut, klirrt nicht, dämpft nicht, sozusagen nebenwirkungsfrei. Aber fragen Sie vorsichtshalber doch Ihren Arzt oder Apotheker.

Faustregel: Je höher der Grad um so besser das Ergebnis und umso aufwendiger und langsamer die Berechnung.

Vorsicht, Pferdefuß!

Oft lesen wir zum Beispiel in Scannerwerbungen, daß ein Scanner eine bestimmte optische Auflösung schafft, und durch Interpolation noch einmal ein Vielfaches an Auflösung herausholt. Das ist schlicht Humbug, denn Interpolation ist nur besseres Raten, es können dadurch keine neuen Informationen gewonnen werden. Ein Bild wird durch interpolieren nicht schärfer, genau wie ein Sound nicht brillianter wird.

Das kann man sich an einem Extremfall verdeutlichen: Sie haben einen 8 kHz Ton mit 32 kHz digitalisiert (folglich beträgt die Schwingungsperiode 4 Samples), diesen Sample-Sound rechnen Sie nachträglich herunter auf 4 kHz Abtastfrequenz. Sie erhalten dadurch das Ergebnis, das Sie auch erhalten hätten, hätten Sie den Ton gleich mit 4 kHz digitalisiert - nämlich gar nichts, oder wenigstens Müll. Einen 8 kHz Ton unter 4 kHz Samplingfrequenz darzustellen ist nicht möglich, wie Ihnen sofort auffällt (die Periode wäre rein rechnerisch ein halber Sample). Versuchen Sie dennoch, diesen Sound mit einer Wahnsinns-Interpolation wieder auf 32kHz hochzuschrauben, brauchen Sie natürlich nicht zu erwarten, daß sich der 8 kHz Ton wieder anfindet.

Aber selbst wenn man auf höhere Abtastfrequenzen (insbesondere wenn Original- und Zielsamplefrequenz nicht in einem ganzzahligen Verhältnis stehen) umrechnet, verliert ein Sound an Brillianz!

1.63 Kohärenz

Die Kohärenz (etwa: Gleichgang)

Als kohärent bezeichnet man Wellen mit gleicher Frequenz und gleicher Phase. Wichtig ist, daß man die Phase nicht hört, da es sich im Grunde um winzige Zeitverschiebungen handelt. Nehmen Sie z.B. eine Frequenz von 1kHz ist die Periode 1ms, folglich ist eine Sinusschwingung von 1kHz schon nach einer Verzögerung um eine 1ms wieder deckungsgleich mit der Originalschwingung. Was man dann eine Phasenverschiebung von 360° nennt, in Analogie zum Kreis, wo man nach 360° Drehung wieder in der Ausgangsposition ist.

Trotz daß phasenverschiedene Schwingungen gleich klingen, sind sie nicht in jeder Hinsicht gleich. Wissen Sie, wie ein negierter Klang (das heißt jeder Abtastwert des Sample-Sounds wird negiert) klingt? Er klingt haargenauso! Aber was passiert, wenn man einen Klang und sein negiertes Gegenstück mischt (überlagert, addiert ...), können Sie sich denken: Es ist null, absolute Stille, nichts. Soweit zum Extremfall.

Bei Sinusschwingungen kann man nachvollziehen, daß beim Überlagern von zwei solcher Schwingungen immer eine Sinusschwingung entsteht. Ist die Phasendifferenz null, ist die Amplitude des gemischten Signals maximal, ist sie um 180° verschoben ist sie null (sofern die Eingangs-Schwingungen gleiche Amplitude haben). Stellt man sich einen Klang aus mehreren Sinusschwingungen zusammengesetzt vor, erkennt man, daß beim Überlagern von zwei gleichklingenden phasenverschobenen Signalen, sich einzelne Sinusbestandteile sowohl verstärken als auch auslöschen können.

Beachten sollte man diesen Effekt z.B. beim Mischen von gefilterten Signalen. Normalerweise sind Sie's zufrieden, wenn Sie aus einem Klang bestimmte Frequenzen aussieben können. Das Mischen von Original- und gefilterten Signalen bringt nicht immer den gewünschten Erfolg. Z.B. ist Original minus Bandpass-gefiltert nicht unbedingt gleich Bandsperre-gefiltert, denn das setzt voraus, daß alle Frequenzen den Filter ohne Phasenverschiebung passieren, was gerade bei rekursiven Filtern nie Fall der ist.

Was aber gerne gemacht wird, um einen Ton lebhafter zu gestalten, ist die Überlagerung leicht dissonanter Töne, was sich dann Phasing nennt.

1.64 Schleifen

Schleifen (engl. loops)

Normalerweise besteht ein Sample-Sound aus einer begrenzten Anzahl Daten, d.h. der Klang verstummt sobald alle Daten bei der Wiedergabe ausgegeben wurden. Wie kann man aber beliebig lange Klänge auf Sample-Basis schaffen?

Das Problem läßt sich häufig lösen, in dem man dem Sound eine Schleife verpaßt. Beim Abspielen wird dann zuerst der Klangabschnitt ausgegeben der vor dem Schleifenanfang steht. Danach wird der Schleifenteil immer und immer wieder ausgegeben, prinzipiell ewig. In Musikstücken wird dann der Klang einfach ausgeblendet oder durch den nächsten Ton der Melodie abgelöst. Die Phase nach der Schleife könnte man eigentlich zum Ausklingen des Instrumentes verwenden, von dieser Möglichkeit wird allerdings kaum Gebrauch gemacht.

Bei Sample-Sounds von Instrumenten benutzt man Schleifen, um nach der Einschwingphase (welche meistens am charakteristischsten für ein Instrument ist) eine recht statische Phase zu erhalten, welche dann beliebig lange gehalten werden kann. Bei Bläser- oder Streicher-Klängen ist das Setzen von Schleifen (neudeutsch: Loopen) deshalb sehr passend und entsprechend beliebt.

Man kann auch ganze Sequenzen, wie ein paar Takte Rhythmus, in Schleifen (dann Drumloops genannt) verwandeln und damit eine durchgehende Untermalung erzielen.

Kurzum: Schleifen erlauben es, mit sowohl geringem Speicherbedarf als auch mit geringem Rechenaufwand Klänge zu verlängern.

Das Definieren von Schleifen wird vom Assampler nicht gesondert unterstützt. Die einzige halbwegs komfortable Methode ist, einen Bereich im Anzeigefenster des Sounds zu markieren und dann die Markierungsbeginn- und -länge-Gadgets in die entsprechenden Loop-Werte im Informationsfenster zu ziehen.

Liegen die Markierungsgrenzen weit auseinander, kann man die Mehrfachansicht des Anzeigefensters vorteilhaft einsetzen. Um möglichst saubere (=unauffällige) Schleifen hinzubekommen, sollte man beachten:

1. Die Wellenformen am Anfang und am Ende sollten möglichst gleich aussehen.

2. An den Grenzen sollten Nulldurchgänge sein.

3.1. Die gewählten Nulldurchgänge sollten in der gleichen Phase der Wellenform auftreten.

3.2. Ist das nicht möglich, sollte an beiden Nulldurchgängen (also am Anfang und am Ende) die Kurve die x-Achse in der gleichen Richtung schneiden (also an beiden Stellen von oben nach unten, oder an beiden Stellen von unten nach oben).

Bei der Frage, wie man denn nun einen Sample-Sound ohne Schleife zu kennzeichnen hat, wenn man trotzdem keinen eigenen Schalter "Schleife ja oder nein" verwenden will, gehen die Programmierermeinungen allerdings weit auseinander.

So sind für den Fall, daß ein Sound keine Schleife haben soll, in verschiedenen Programmen auch verschiedene und nicht immer logische Belegungen für die Werte Schleifenanfang und Schleifenlänge festgelegt.

Unlogisch ist zum Beispiel LoopBegin=0 und LoopLength=sehr klein, denn das bedeutet doch: "Beginne nach 0 Sekunden (also direkt am Anfang) mit einer sehr kurzen Schleife". Daraus folgt: Der eigentliche Klang kommt gar nicht zum Zuge.

Logisch ist dagegen LoopBegin = .Begin + .Length sowie LoopLength = 0 (mit den Variablennamen von Assampler gesprochen, Loop heißt auch manchmal Repeat = Wiederholen), denn das bedeutet, daß eine (unhörbare) Schleife nach Ende des Sample-Sounds einsetzt. Das ist praktisch wie ein Sample-Sound ohne Schleife.

1.65 Puffer

Puffer

Puffer kann man essen, wenn's Kartoffelpuffer sind, Puffer können Stöße abfangen, wenn sie an Eisenbahnwaggons hängen. Aber was suchen Puffer in einem Soundprogramm?

Puffer sind Speicherblöcke, in denen Daten zwischengespeichert werden, also irgendwie gepuffert. Beim Assampler stellen sie den Kompriß dar zwischen dem Berechnen jedes einzelnen Wertes und dem Berechnen eines ganzen Klanges in einem Zug.

Wählen Sie große Puffer (die Größe wird immer in Samples angegeben, bei 16Bit-Samples also Anzahl Bytes = 2 * Anzahl Samples) geht eine Berechnung besonders schnell, benötigt aber viel Speicher, wählen Sie sie klein, sparen Sie Speicher auf Kosten der Rechenzeit.

Welche Puffergröße letztendlich genommen wird, um eine Berechnung durchzuführen, hängt hauptsächlich vom letzten Objekt in der Kette ab. Kopieren Sie einen Klang in einen Sample-Sound oder üben einen Prozeß darauf aus (was hier bekanntlich das gleiche ist), wird die Blockgröße der Sample-Sounds eingehalten. Spielen Sie einen Algorithmus als Klang ab, wird die Wiedergabe-Puffergröße angewandt, schreiben Sie in Disk-Sample-Sounds, kommen entsprechend Puffer der Größe, wie sie für Disk-Sample-Sounds vorgesehen sind, zum Einsatz.

Daß es so viele verschiedene Größenangaben zu Puffern gibt, liegt daran, daß die Puffer immer mit bestimmten anderen Dingen zusammenhängen. Z.B. müssen Puffer bei Disk-Sample-Sounds den Datenaustausch mit meist langsamen Datenträgern puffern, so daß hier größere Puffer als normal zu empfehlen sind.

Beachten Sie auch, daß die Puffergröße nicht als Maximum der Speichernutzung anzusehen ist. Je nach Komplexität der Berechnung können mehrere Puffer der von Ihnen vorgegebenen Größe gebraucht werden. Wenn Sie z.B. 16 MB Speicher besitzen, ist eine Puffergröße von 5 MB auf jeden Fall zu groß.

1.66 Schreib mal wieder

Der Autor

Trotz meines skeptischen Blickes, daß mir keine allzu groben Schnitzer unterlaufen, kann ich keine Garantien abgeben, daß alles hier geschriebene unanfechtbar richtig ist. Machen Sie sich ruhig einmal klar, daß sich jeder, der einen Fehler in meiner Beschreibung entdeckt und mir nicht meldet, an (hoffentlich) vielen, vielen Lesern schuldig macht, die mein Geschriebenes ungeprüft übernehmen.

Bevor Sie mich Hals über Kopf mit Beschwerden zupflastern, noch ein paar Bemerkungen. Nicht, daß ich nicht gerne Briefe lese, ich mag nur nicht wieder und wieder hören, was ich selbst schon weiß. Deshalb nehmen Sie sich bitte folgendes zu Herzen:

Bei Fehler-Berichten sollten Sie sicher sein, den entsprechenden Abschnitt im Guide gelesen zu haben. Habe ich den Fehler schon selbst erwähnt, reicht eine kurze Bemerkung von Ihnen, daß Sie dieser Fehler oder jene Unzulänglichkeit genauso stört wie mich.

Beim Berichten undokumentierter Fehler ist es wichtig, daß Sie die Voraussetzungen zum Auftreten des Fehlers möglichst genau eingrenzen. Dementsprechend ist Ihre Rechnerkonfiguration, die installierten Patches usw. aufschlußreich. Besonders bei Anlaufschwierigkeiten, d.h. der Assampler will gar nicht erst starten, geben gelegentlich SnoopDos-Protokolle Aufschluß über die Fehlerquelle.

Für Verbesserungsvorschläge empfehle ich ebenfalls das Studium des zugehörigen Abschnittes und der vorgesehenen Erweiterungen. Am besten ist, Sie nehmen die Stichworte aus der Erweiterungen-Liste und schreiben hinter diese Zahlen, je nach dem wie wichtig Ihnen die Umsetzung dieses Punktes erscheint. Mit "0" (Null) wird diejenige Sache gekennzeichnet, die ich Ihrer Meinung nach zuerst machen sollte. Mit "1" (Eins) das, was danach ansteht usw. "-" (oder unendlich, dessen Zeichen allerdings nicht im ISO 8859-1 enthalten ist) bleibt den Dingen vorbehalten, die Sie für völlig überflüssig erachten.

Nach diesen Pflichtteilen können Sie sich der freien Meinungsäußerung hingeben, so Ihnen noch nicht die Lust daran vergangen ist. Allerdings: Wenn Sie Assampler für viel zu kompliziert und realitätsfern konzipiert halten - schauen Sie auch einmal zur Konkurrenz!

Alles für mich interessante adressieren Sie an:

E-Mail: henning.thielemann@student.uni-halle.de

URL: http://home.pages.de/~lemming

Briefe: Henning Thielemann

Veilchenweg 34

06118 Halle

Germany

1.67 Bitte keine Werbung einwerfen

Werbung

Trotz, daß mich die Arbeit am Assampler rund um die Uhr beschäftigt, habe ich noch andere Programme erschaffen. Diese sind allesamt im AmiNet und auf den CDs (bislang AmiNet 17-20 = AmiNetSet 5) zu finden.

NomenEstOmen (misc/sci/NomenEstOmen.lha, benutzt MUI)

Hiermit können Sie organische Moleküle editieren und danach per Knopfdruck den Nomenklaturnamen nach IUPAC-Regeln bestimmen lassen.

FotoBestellung (biz/dbase/FotoBestellung.lha, benutzt MUI)

Nützlich für Leute, die für Bekannte Fotos nachbestellen wollen. Programm zählt Fotonummern aus, bedruckt Fototaschen mit Nachbestelliste und ordnet später Fotos den Bestellern zu.

MasterMindSolve (game/think/MasterMindSolv.lha)

Fünf verschiedene Algorithmen, jeder in einer anderen Sprache, zum Knacken von MasterMind-Codes. Sprich: Das beliebte Mastermind-Spiel, nur daß Computer Codes des Benutzers entschlüsselt. Allerdings nur Shell-Version und daher recht farblos.

Compare (util/cli/CompareHT.lha)

Dateien und Verzeichnisse vergleichen

SplitFile (util/cli/SplitFile.lha)

Datei auf mehrere Datenträger verteilen

StripUmlauts (util/cli/StripUmlauts.lha)

Deutsche Umlaute in ae, oe, ue usw. konvertieren

StripCRLF (util/cli/StripCRLF.lha)

PC-typische Zeilenenden (CR+LF) durch normale LFs ersetzen

MemTest (util/moni/MemFlipTest.lha)

Speicher auf Fehlerfreiheit testen

MUI Interface (dev/mui/MuiCluster.lha)

Cluster-Interface für MUI

... und nicht vergessen, meine ausufernde HomePage zu besuchen!

1.68 Adreß-, Daten- und Handelsregister

Registration

Die ersten Versionen von Assampler werden Freeware sein, auf lange Sicht peile ich aber Shareware oder kommerziellen Vertrieb an. Deswegen können Sie sich bei mir noch nicht registrieren lassen, aber ich bin für jede Zuschrift dankbar und ich registriere dann zumindest Ihr Interesse :-)

Als nächstes brauchen Sie wohl meine Adresse.

1.69 Wer kreditwürdig ist

Wer und was war indirekt am Zustandekommen von Assampler beteiligt?

Funktionsbibliotheken - was bei Ihnen zum Einsatz kommt

MUI (MagicUserInterface), Stefan Stuntz, Bedienoberfläche

MCC_Listtree, Klaus Melchior, Mui-Custom-Class zur Anzeige von Baumstrukturen

AHI (AudioHardwareInterface), Martin Blom, Sound-Ausgabe u.a. für Soundkarten

datatypes (V45), Christian Buchner/Roland Mainz, Import/Export verschiedener Datenformate

ReqTools, Nico François + Magnus Holmgren, Bibliothek für Requester

Cluster-OFunctions.mod, Viona, Funktionsinterpreter

iffparse, AmigaOS3.1

SearchGuide, LSAGS, Suchfunktion für AmigaGuides

Entwicklungsumgebung - womit ich Assampler entwickelt habe

Cluster, Viona/MOM, Programmiersprache (exzellente Weiterentwicklung von ModulaII)

PhxAss, Frank Wille, Assembler (so wird das Gerät richtig geschrieben!), damit entwickle ich die synthesizer.library

ADis, Martin Apel, Disassembler

gcc, ziemlich viele Autoren, C-Compiler

sobja, Ray Burr, damit kann man die vom gcc erzeugten Linkobjekte in das Amiga-typische Linkformat konvertieren

Scout, Andreas Gelhausen, Systemmonitor

Enforcer, Mike Sinz, erkennt fehlerhafte Speicherzugriffe

MungWall, Commodore Amiga Inc., erkennt fehlerhafte Speicherzugriffe, provoziert Fehler, die ansonsten unentdeckt bleiben

PoolWatch, Magne Østlyngen, wie MungWall nur mit den Speicher-Pools, die seit OS3.0 unterstützt werden

MemWatcher, Frank Brandau, beobachtet Größe des belegten Speichers

BarFly debug, Ralph Schmidt, Assembler debugger

gccfaq, Lynn Winebarger, Zusammenstellung von Fragen und Antworten zur GNU-C-Portierung für den Amiga

ADocReader, Gilles Masson, komfortable Anzeige für AutoDocs

Amiga Developer CD, Amiga Technologies GmbH, alles rund um die Amiga-Programmierung, kostet nicht viel, ist aber viel drauf

HappyEnv, Martin Gierich, ein sich selbst füllendes ENV:-Verzeichnis, was sich beim Programmieren ebenfalls als guter Objekt-Modul-Cache (so was wie Link-Libraries bei C) erwiesen hat; sozusagen das HappyEnd für das ENV:-Verzeichnis

KingCon, David Larsson, zwar "nur" ein Console-Handler, erleichtert aber Arbeit mit Tools mit Shell-Interface enorm

GoldEd, Dietmar Eilert, ein Editor mit sehr vielen Funktionen speziell für Programmierer, aber leider mit einer mindestens genauso langen Liste von ärgerlichen Schönheitsfehlern (z.B. dem selbstgestrickten GUI-System)

MuiBase, Steffen Gutmann, Datenbank, damit verwalte ich die Beschreibungen aller Soundklassen

Localizer, Massimiliano Origgi, Unterstützung bei Erstellung und Übersetzung der Sprachkataloge

1.70 MUI - magic user interface

MUI - MagicUserInterface

Version 3.8

(c) Copyright 1992-97, Stefan Stuntz

MUI is an object oriented system to create and maintain graphical user

interfaces. From a programmers point of view, using MUI saves a lot of

time and makes life much easier. Thinking about complicated terms like

window resizing or font sensitivity is simply not neccesary.

On the other hand, users of MUI based applications have the ability to

customize nearly every pixel of a programs interface according to their personal taste.
Please click on the Install-MUI icon to install MUI on your system or to update from previous versions. Latest news and support can be found on Internet at www.sasg.com.
****
Attention
If you’re still not registered for MUI, check the new and unique shareware lottery and great discounts at www.sasg.com. Give it a try!

1.71 AHI - audio hardware interface

Overview
****
The Amiga has always had excellent sound capabilities. In 1986, they were awesome. Today, well.... Perhaps not awesome, but still very good. The OS interface, audio.device has however never been as good as it could have been. It is tied hard to the underlying hardware, and doesn’t work very well for music. This has led to a situation where most audio programs only use audio.device to allocate the audio resource, and then poke around in the hardware registers--making it next to impossible to replace the Paula chip (1).
There have been attempts to write an audio.device clone that uses a sound card instead of Paula, but so far nobody has succeeded. It is definitely possible, but the question is if it is worth the trouble--too many of the programs bang the hardware.
Entering AHI (2). AHI is a new audio subsystem, designed to be flexible, hardware independent, expandable and future safe. It is designed with real-time applications in mind. It is designed to play modules (3) and sound effects as efficient as possible, taking advantage of modern DSP-based sound cards.
Yet AHI allows applications that don’t need full control over the audio hardware to share the resource, so that many different programs can play and record sound at the same time, without conflicts.
As a user you will hopefully not see much of AHI, other than the audio mode requesters. They works almost exactly like screen mode requesters.
AHI was never supposed to be the standard for hardware independent audio. It was meant as a temporary solution until Amiga Technologies

delivered an official standard. However, the situation looks worse and
worse for every day that passes by, and this may be all you will ever
get.
---------- Footnotes ----------
(1) Paula is one of the custom chips, and she is responsible for
the sound (and more). Unfortunately, this chip has not been updated
since the very first Amiga was released.
(2) The name AHI was chosen because the functions in the system had
to have a prefix, and the author couldn't come up with anything better
than Audio Hardware Interface, something that he has regretted ever
since. The suggested pronunciation is "atchii", as in "God bless!".
(3) Originally designed in 1986 by Karsten Obarski, modules have
become a de facto standard for game and demo music. The original format
has been improved many times, and many new music formats have--more or
less--been derived from it, including the popular S3M and XM formats.

1.72 Konkurrenz

Die Konkurrenz - mehr als 50 Gründe, Assampler nicht zu benutzen

Tja bei so vielen Soundprogrammen fragt man sich doch, wie der Amiga jemals zum Ruf eines Grafikcomputers gelangen konnte ...

Ich bin mir sicher, daß es weit da draußen in der Welt noch viel viel mehr Sound-Programme gibt, von denen wir nie etwas gehört und gesehen haben. Aber wenn Sie solche Außenseiter kennen, auch wenn die Projekte lange gestorben sind, verraten Sie es mir!

Auflistung erfolgt in alphabetischer Reihenfolge, mit Name, Quelle, Autor, Kommentar. Version und Quelle müssen nicht immer die aktuellsten sein. Einige Namen habe ich nur aufgeschnappt, andere Programme bzw. deren Demos schon persönlich begutachtet, das erkennt man am Umfang der Kommentare.

Bearbeitung, Synthese, Analyse von Sample-Sounds

AmigaSynth 1.0, Amok 35, Jürgen Zimmermann

FM-Synthese mit 32 Operatoren

AudioEd 4, Meetings Pearls III, Christian A. Weber

Sample-Editor, einfache Effekte, MUI

AudioEvolution, AmiNet, Davy Wentzler

HardDisk-Sample-Editor, einfache Effekte

Audiomaster 4, Aegis

Samplebearbeitung, einfache Effekte auf Sample-Sounds oder in Echtzeit

AudioLab16, AmiNet, Maurizio Ciccione

Sample-Bearbeitung + Sound-Processor

AudioScope, Fred Fish, Richard Horne

Frequenzanalyse in Echtzeit

AuralSynthetica, AmiNet, Nicola Blachford

Sound-Processor, Zusammenschalten von Operatoren

DeluxeSound

Samplesoftware zu Sampler

FMSynth, Amok/FredFish/Aminet, Christian Stiens

FM-Synthese mit 6 Operatoren, Einspielen über MIDI, Austausch Algorithmen mit DX7

MasterSound (?)

ein Sampler zu dem es eigentlich auch eine SampleSoftware geben müßte

MSPL Music Synthesis Programming Language, Antonello Biancalana

der Name ist Programm (im wahrsten Sinne des Wortes), wird vom Autor wohl nur noch auf Unix entwickelt

Lichtorgel, Amok 77, Christian Stiens

8-Kanal-Lichtorgel in Echtzeit

PerfectSound

Sample-Bearbeitung

Samplitude, AmiNet, früher SEK'D, jetzt bei A.C.T.

Sample-Bearbeitung

SampleWrench, AmiNet, Dissidents

Sound-Processor

SinED, AmiNet, Jarkko Vatjus-Anttila

Sound-Processor

SoundEffect, Fred Fish/AmiNet, Sven Bühling

Sound-Processor

SoundFX, AmiNet, Stefan Kost

Sound-Processor

SoundProbe, Oberland

Sound-Processor

Spectrogram, Fred Fish, Richard Horne

Frequenzanalyse, nicht in Echtzeit

Spectroscope, Amok 87, Christian Stiens

Frequenzanalyse in Echtzeit

Synthia, Demo auf CAM #70, Bob Dayley and The Other Guys

Sound-Processor, so ziemlich erster mir bekannter

TechnoSound, New Dimensions, Jon Wheatman

Samplesoftware zu Sampler

TB303 Emu, Aminet:misc/emu/303emuV2_2, Jeroen Schellekens

Emulation des Kultsynthesizers TB303

WaveBeast, Aminet:mus/edit, Jan Krutisch

Sound-Processor

Wavetracer, Time/AmiNet, Michael Pfeiffer

Sound-Processor

Musikprogramme

AlgoRhythms, AmiNet, Thomas E. Janzen

automatische Rhythmus-Erzeugung

AProSys, Aminet, Petter A. Urkedal

Tracker mit Prozessorsounds für Hardcore Hex-Freaks

ArtOfNoise

Tracker

CyberTracker, AmiNet, Andreas Öhman

Tracker

DeluxeMusic, Electronic Arts

DigiBooster, AmiNet, Tomasz&Waldemar Piasta

Tracker, AHI

DigitalSymphony

FaceTheMusic

InstantMusic

MusicLine, AmiNet, John Carehag

Tracker mit vielen Echtzeiteffekten

NewTracker, AmiNet, Bjornar Henden

Tracker

NoiseTracker

Tracker

Octalyzer

Tracker, achtstimmig

OctaMED, AmiNet, Teijo Kinnunen

Tracker, achtstimmig, MIDI

ProTracker

Tracker

QuadraComposer

SAM

StoneTracker

Tracker

SonicArranger

Sonix

Musikprogramm mit Noten, Echtzeiteffekte

SoundTracker

Tracker

StarTracker

Tracker

SuperJam

TFMX, Chris Huelsbeck

THX Sound System

TME - The Musical Enlightment, UGA Development

MIDI-Sequencer

Bars&Pipes, The Blue Ribbon SoundWorks

Camouflage, I.S.M. Martin Endres

MidiTracker, AmiNet:mus/midi, Sven Thoennissen

Mignon, SEK'D

Soundscape Pro Midi Studio

Samplekonverter

AmiSOX, AmiNet, Lance Norskog

Opus8, FredFish, John A. Safranek

konvertiert Macintosh-8-Bit-Roh-Sample-Sounds in IFF-8SVX

SoundBox, AmiNet, Richard Körber

konvertiert zwischen RAW, IFF-8SVX, IFF-16SV (möchte mal wissen, wer das eingeführt hat, es gab doch schon lange das IFF-SAMP), WAVE, VOC, AIFF, MAUD, Maestro

SoundMachine, FredFish, Syd L. Bolton

konvertiert zwischen RAW, IFF-8SVX, WAV, VOC

ungezählte Song-Ripper und -Player

CDPlayer14 (CyberSound), AmiNet, Christian Buchner

DeliTracker, FredFish, Peter Kunath und Frank Riffel

EaglePlayer

MultiPlayer, FredFish

MultiRipper, FredFish

ExoticRipper, AmiNet

sonstige Programme, die Krach machen

Bugz

EasyPlay, AmiNet, Jimmy Rosenholm

beide Programme erzeugen recht chaotische Musik, wenn man bunte Punkte auf der Bildfläche verteilt und sich Punkte darüberhinwegbewegen

1.73 Die Polizei warnt: Neue Stilblüten aufgetaucht

Stilblüten - wer zuletzt lacht, hat's nicht eher begriffen

Es gibt eine ganze Reihe von Dokumentationen und Workshops zum Thema Soundbearbeitung, die ich im Laufe der Zeit konsumiert habe. So finden sich nach und nach recht gegensätzliche Beschreibungen zu den gleichen Sachverhalten, so daß ein Laie danach entweder genau so viel weiß wie vorher, oder aber schlimmer, sich etwas falsches merkt. Deswegen seien Sie mir bitte nicht böse, wenn ich hier alle kleinen Ausrutscher durch den Kakao ziehe.

Amiga Special 8/9-1997, Seite 40

Hier findet man eine Abbildung, die die 8-Bit- und 16-Bit-Auflösung durch 8 bzw. 16 Kästchen hohe Raster zu veranschaulichen sucht.

unicum oder academix, Ende 1997, weiß nicht mehr genau

Enthält Erklärung, daß bei Soundkarten 16-Bit-Sound 16 Kanäle bedeuten würde.

Dr. Holger Blaar, Vorlesung Datenkommunikation an der MLU, 11.5.1999

Dieser Dozent unterrichtet, daß Sampling mit 8 Bit bedeutet, daß man 256 verschiedene ... Töne darstellen kann.

Rügheimer, Spanik, AmigaBASIC, 7. Auflage, Databecker, 1986, Düsseldorf

Aus der Beschreibung des Befehls WAVE auf der Seite 729 (=3^6 :-), der die Wellenform des abzuspielenden Tones angibt:

... In diesem Integerfeld ist der Verlauf einer beliebigen Wellenform gespeichert. Die Werte liegen zwischen -128 und 127 und geben die Amplitude der Welle zum Zeitpunkt des Samplings an. ...

Bleek, Hecht, Litzkendorf, Das große Buch zu GFA-BASIC / Amiga, 1. Auflage, Databecker, 1990, Düsseldorf

Diese Autoren führen diesen Gedanken auf Seite 141 bei der Beschreibung des WAVE-Befehls, der an sich funktionsgleich zu seinem AmigaBASIC-Namensvetter ist, konsequent weiter:

Mit WAVE läßt sich für jeden der vier Tonkanäle des Amiga (Kanal = 0 bis 3) eine sogenannte Hüllkurve festlegen. Diese bestimmt den Klangverlauf des über diesen Kanal ausgegebenen Tones. ...

Philip Grabbert, "Der gute Ton" - Soundausgabe mit Assembler, Amiga Special, Ausgaben 9/1992 - 12/1992

Wie ein roter Faden zieht sich die geballte Unkenntnis von einem zum nächsten Teil des Workshops.

Zum Beispiel nennt der Autor eine Frequenz Hertz-Zahl und eine Abtastperiode (Register $DFF0A6) Amiga-Frequenzwert und wundert sich dann, warum ein größerer Wert zu tieferen Tönen führt. Dazu liefert er das wahrscheinlich umständlichste Programm zum Umrechnen von Frequenz in Abtastperiode gezählt in DMA-Zyklen - normalerweise ist das mit der Division 3.58MHz/f abgehakt:

Zunächst lasse man alle Einheiten weg und ersetze die einfache Division durch einen Doppelbruch:

1/(f/3580000) = 1/(f*2.79365e-7)

Nun runde man f auf ein Vielfaches von Hundert, damit die nachfolgende Multiplikation mit 2.8 innerhalb der ganzen Zahlen exakt durchgeführt werden kann. Diese Multiplikation wird zerlegt in 2*f+0.8*f, wobei die Multiplikation mit 0.8 um Gottes willen nicht mit einer Hardware-Multiplikation durchgeführt wird, sondern mit einer Division durch 10 und einem anschließenden Shift um 3 Stellen nach links. Dafür haben wir ja auf Vielfache von Hundert gerundet. Ob Vielfache von zehn auch gereicht hätten? Na, sicher ist sicher, also Vielfache von Hundert. Nicht etwa, daß wir durch 5 geteilt und dafür nur um 2 Stellen nach links verschoben hätten, das wäre viel zu genau.

Dann teile man 10 Millionen durch diesen Wert und zwar nicht mit der Hardwaredivision divu (viel zu schnell!) und auch nicht mit der Schuldivision sondern mit Abzählen (= wie oft kann ich x von y abziehen?).

Als weitere Neuheit weiß der Autor zu berichten, daß der Vibrato-Effekt eine Amplitudenmodulation ist, wohl als Konsequenz daraus, daß manche dem Tremolo-Effekt auch ein bißchen Frequenzmodulation zuschreiben.

Weiterhin erwartet der Autor, daß ihm das arithmetische Mittel aller Signalwerte die durchschnittliche Amplitude liefert (der Fehler wird auch im RKRM/Hardware begangen), nimmt man dafür nicht die Beträge der Signalwerte?

So geht das munter weiter: Wir erfahren, daß Verstärken um gebrochene Faktoren besonders schwierig ist (ob wir vielleicht die gleiche Divisions-Multiplikation wie oben verwendet haben?), daß rauschende, verzerrte und kratzende Ergebnisse bei Verzerr-Effekten wohl auf das 8-Sampling zurückzuführen sind, daß man zwei Signale am einfachsten mischt, in dem man ihre Signalwerte verzahnt und die Abtastfrequenz verdoppelt, daß man zu einem Gemisch aus zwei Klängen besser keinen weiteren Klang mischt, weil man das menschliche Ohr damit nicht überlisten kann (liegt's vielleicht daran, daß die beiden ersten Klänge im Gesamtgemisch jeweils nur noch mit dem Lautstärkeanteil 1/4 vorliegen, der letzte aber mit 1/2 ?)

Michael Pfeiffer, Autor von WaveTracer Mark IV, Aminet 27, Oktober 1998

Neben anderen interessanten Behauptungen ist folgender Ausschnitt aus der WavetracerMarkII-Beschreibung besonders gut gelungen:

Bei der Erzeugung der Grundwellenformen wurde auf hohe Genauigkeit Wert

gelegt. So wird eine Welle nicht durch einmaliges Berechnen einer

Periode und mehrfaches Kopieren dieser erzeugt. Vielmehr wird diese Welle

durch laufende Berechnung von Perioden erstellt. Dies ist u.a. bei Sinus-

kurven zweckmäßig, da hier eine Periode niemals genau dem eingestellten

Wert der Periodendauer entsprechen kann (ich sag nur Konstante Pi:

3.141592654blablablaetceterappundsoweiterundsoweiterundsofort...).

Das muß dann selbst dem Autor komisch vorgekommen sein und er hat diesen Abschnitt im Guide zum WaveTracerMarkIV entfernt. Erhalten bleibt uns aber folgende zum Kompressionsverfahren aufgebauschte Selbstverständlichkeit (aus Programmierung.guide des WT-IV):

Die 3Byte-Kompression ist eigentlich nichts weiter als eine reine Runtime-

Kompression. Die 24Bit-Daten des WaveTracer sind vorzeichenbehaftete 32Bit-

Zahlen, deren Bits folgendermaßen genutzt werden:

Bit 0 bis Bit 22 - Datenbits

Bit 23 bis Bit 30 - frei

Bit 31 - Vorzeichenbit

Daraus folgt, das die 24Bit-Daten sich im Bereich von ca. -8388600 bis

8388600 (-MAX24..MAX24) bewegen.

Dieses Kompressionsverfahren ist ein verlustfreies Verfahren, da die

Daten wieder zu 100% hergestellt werden.

Die Kompression:

Die 24Bit-Daten werden durch Addition einfach um 8388600 (+ MAX24) an-

gehoben. Jetzt sind, da das Bit 31 nicht mehr als Vorzeichenbit genutzt

wird, die oberen 8 Bit frei. Folglich brauchen nur noch die Bits 0 bis

23, d.h. wirklich nur die unteren 24 Bit, abgespeichert werden.

Die Dekompression:

Hier wird einfach alles in umgekehrter Reihenfolge erledigt: Die 24Bit

werden in die unteren 3 Bytes eines Longwords geladen und durch

Subtraktion von MAX24 wieder in das korrekte Longword-Format für den

WaveTracer umgewandelt.

Da fällt noch eine kleine Übungsaufgabe für Sie ab: Was unterscheidet eine Runtime-Kompression von anderen Kompressionsarten?

Sollten Sie bei der einen oder anderen Sache nicht so recht wissen, wo da die Pointe liegt, fragen Sie lieber noch mal nach.

1.74 Fährmann, übersetze!

Wörterbuch

Wer sich schon länger mit Computern befaßt, weiß daß es lange Zeit überhaupt nicht üblich war, Programme in anderen Sprachen als Englisch anzubieten. Bildschirmplatz war schon immer knapp und da kamen die durchweg kurzen englischen Begriffe gerade richtig, um in einer Bedienoberfläche untergebracht zu werden.

Heute ist das ein bißchen anders. Seit man es sich speicher- und platzmäßig leisten kann, werden Programme in verschiedenen Landessprachen angeboten. Beim Amiga wird das sogar durch die locales.library unterstützt. Allerdings fühlen sich nun viele

Experten auf ihrem eigenen Amiga angesichts der unheitlichen Übersetzungen und unmöglichen Abkürzungen gar nicht mehr heimisch, und die Bedienung erfolgt wirklich nur noch intuitiv, nach dem Motto "Ich weiß nicht genau, was dieser Knopf auslöst, aber von der Positionierung im GUI her müßte es der Quit-Button sein." Also werden sicher einige die Kombination aus englischem GUI und muttersprachlicher Dokumentation wählen, mit dem Erfolg, daß in der Dokumentation immer noch die übersetzten Bezeichnungen stehen.

Die Brücke zum englischen GUI soll nun diese kleine Liste bieten:

deutsch englisch

Abweichung Deviation

Anstieg Tangent

Anzeige Display

Ausdruck Expression

Auswahl Selection

Bearbeiten Edit

Beenden Quit

Bereich Range

Bruder Brother

Datei File

Einfügen Insert

Einschieben Paste

Ersatz Replace

Erzeugen Create

Faden Thread

Farbstift Pen

Fenster Window

Freistellen Isolate

Hilfe Help

Kinder Children

Klang Sound

Klangwahl SoundSelect

Klon Clone

Laden Load

Leeren Clear

Löschen Delete

Öffnen Open

Parameter Parameter

Primzahl Prime

Rechner Calculator

Referenz Reference

Schleife Loop

Schließen Close

Schnitt Cut

Schwingung Wave

Sichern Save

Stützstelle Node

Tangente Tangent

Über About

Variable Variable

Vielfach Multiply

Verschmelzen Merge

Welle Wave

Wert Value

Wiedergabe Play

Zeiger Pointer

Zurück Restore

Zwischenablage Clipboard

1.75 Du sollst nicht Stahl dieben!

Das gibt es rechtlich zu beachten!

1. Assampler darf in dieser Version frei kopiert werden.

2. Für die kommerzielle Nutzung ist eine Lizenz erforderlich.

3. Assampler darf verkauft werden oder in eine PublicDomain-Softwaresammlung, auf Heftdisketten oder CDs u.ä. aufgenommen werden, so lange beim Verkauf der Preis für Datenträger + Kopiergebühr einen Euro nicht übersteigt.

4. Zwingen kann ich niemanden dazu, aber jeder der Workshops zum Assampler schreibt (soll ja vorkommen), möchte den Text vorher von mir absegnen lassen. Ich möchte nicht, daß Irrtümer durch einen "professionellen" Workshop noch mehr Gewicht erhalten.

5. Für Schäden, die durch die Benutzung des Programmes entstehen, kann ich nicht haftbar gemacht werden. Ich kann zumindest dafür garantieren, keinen Code eingebaut zu haben, der Festplatten formatiert oder Assampler-Code vervielfältigt. Assampler ist ebenfalls kein trojanisches Pferd. Für Programme, die sich ebenfalls Assampler nennen, aber nicht von mir stammen, gilt diese Garantie natürlich nicht.

6. Es ist ausdrücklich erlaubt, Assampler für den Eigengebrauch zu modifizieren. Nicht gestattet ist es hingegen, derart veränderten Code weiterzugeben.

7. Mit Assampler erzeugte Dateien dürfen selbstverständlich frei weitergegeben werden.

1.76 Bücher, die die Welt bedeuten

Literatur

Einen Blick hinter die Kulissen erlauben diese Bücher:

Hans-Jochen Schulze, Georg Engel

Moderne Musikelektronik

Praxisorientierte Elektroakustik und Geräte zur elektronischen Klangerzeugung

VEB Militärverlag der DDR, Berlin

1989

ISBN 3-327-00772-1

- trotz der inzwischen vergangenen 10 Jahre immer noch ziemlich modern

- die ganze Palette der analogen Effekt- und Synthesizertechnik

- von den Effektideen bis zur Leiterplattenbestückung alles dabei (wobei sich die Mangelsituation inzwischen umgekehrt hat, d.h. damals wurden Bauteile westlicher Produktion gemieden, dafür bekommt man heute nichts mehr aus der Ostproduktion)

- mit vielen Zeichnungen, Fotos, Schaltplänen, Leiterplattenentwürfen

Hans-Jürgen Kowalski

Berechnung und Aufbau aktiver RC-Filter

VEB Militärverlag der DDR, Berlin

1988

ISBN 3-327-00552-4

- alle möglichen analogen Filter zur Klangsteuerung (weniger Effektfilter)

- BASIC-Programme zur Filterdimensionierung

Richard W. Hamming

Digitale Filter

VCH Verlagsgesellschaft

1987

ISSN 0931-3060

ISBN 3-527-26463-9

- nichtrekursive und rekursive Filter

- Spektrum, schnelle Fouriertransformation (FFT)

- ziemlich detailliert, vermittelt Grundlagen

- dennoch nur mit soliden Mathematik-Kenntnissen genießbar

Günter Jorke, Bernhard Lampe, Norbert Wengel

Arithmetische Algorithmen der Mikrorechentechnik

VEB Verlag Technik Berlin

1989

ISBN 3-341-00515-3

- flotte maschinennahe Rechenroutinen

- Zahlendarstellungen, Berechnung von Standardfunktionen, geometrische Probleme

- fertige Programme für Z80 und 8086

Commodore-Amiga, Inc.

AMIGA(R) ROM Kernel Reference Manual: Libraries

Addison-Wesley

1992

ISBN 0-201-56774-1

- Standardwerk für jeden Amiga-Programmierer

- detailliert, leicht verständlich, so gut wie fehlerfrei

- Warnungen vor häufigen Fehlern

- auf der AmigaDeveloper-CD V1.2 erhältlich

Günther Eisenreich, Ralf Sube

Wörterbuch Mathematik Englisch->Deutsch

Verlag Harri Deutsch, Thun und Frankfurt am Main

1994

ISBN 3-87144-939-3

- leider nur in Englisch->Deutsch zu haben

- die nächstgrößere Ausgabe umfaßt dann vier Sprachen, deren Stichworte über Nummern aneinander gekoppelt sind

1.77 ARexx, bzw. was es mal werden könnte

ARexx

Zukunft

Ja, Assampler hat (noch) keinen ARexx-Port. Ich komme aber nach und nach zu der Einsicht, daß da echt was fehlt. Zwar sind eine Menge Möglichkeiten gegeben, mehrere Aktionen mit einem Griff zu erledigen, ohne programmieren zu müssen. Aber nicht alles kann mit Sound-Gruppen und Algorithmen automatisiert werden und alles von einer Bedienoberfläche aus machen zu wollen, ist wahrscheinlich nicht der richtige Weg, da mit jeder neuen Funktion im GUI die Übersichtlichkeit leidet.

Bevor ich mich damit aufhalte, Ihnen die Vorteile ARexx darzulegen, wollen wir lieber mal gemeinsam überlegen, was so ein ARexx-Port alles beherrschen müßte:

Da wären alle Funktionen des Hauptfensters, mit dem Unterschied, daß immer Klang und für Schnittfunktionen Bereiche angegeben werden müssen. Weiter ist es wichtig, mit dem ARexx-Port Klänge erzeugen und Werte verändern zu können. Damit wäre es dann möglich Algorithmen zusammenzubauen.

Trotz aller Perspektiven kann der ARexx-Port nur eine Erweiterung, kein Ersatz zu den Assampler-Algorithmen sein. Denn mit ARexx kann man immer nur einen Prozeß auf einmal ausführen (abgesehen von durch ARexx erzeugten Algorithmen). Assampler-Algorithmen haben der schrittweisen Abarbeitung voraus, daß der Prozeß förmlich in der Luft gehalten wird, weil das Endergebnis erst zum Schluß zurückgeschrieben wird. Anders wäre es gar nicht möglich Megabytes von Daten zu verarbeiten.

Beispiel: Ein- und Ausblenden eines 4-minütigen Musikstückes digitalisiert mit 44 kHz in 16 Bit auf Festplatte. ARexx-Lösung (falls Assampler nicht seine Algorithmen anböte): Anlegen einer weiteren Datei der Größe des Musikstückes für die Hüllkurve (21 MB schreiben) und Berechnen der Hüllkurve (nochmal 21 MB schreiben). Aufprägen der Hüllkurve auf Musikstück verlangt Lesen von Hüllkurve und Musik (2*21MB) und Schreiben des Ergebnisses (21 MB). Macht zusammen 105 MB auf Festplatte umhergeschobene Daten und erfordert vorübergehend zusätzlich 21 MB Festplattenspeicher. Assampler-Algorithmus-Lösung: Anlegen eines Algorithmus der Hüllkurve als Zusammenschnitt von Steuerkurven oder als Spline einem Verstärker zuführt, der auf das Musikstück angesetzt wird. Muß für Lesen und Schreiben zusammen nur 42 MB Festplattendaten schaufeln und kommt ohne weiteren Festplattenspeicher aus.

Ein echter Vorteil wäre ein ARexx-Port zum Beispiel beim automatischen Nachbearbeiten von digitalisierten Klängen. Das könnte ungefähr so aussehen:

Für alle Dateien eines Verzeichnisse tue

Datei einladen

Klang von Gleichstromoffset befreien ("Anheben" mit Differenz "-ZeroLevel")

Pausen an Anfang und Ende entfernen ("Freistellen" mit Bereichsgrenzen "SoundBegin" und "SoundEnd")

Datei speichern

Klang löschen

Schleifenende

Jetzt sind Sie dran: Was für Dinge erwarten Sie von dem ARexx-Port? Was wollen Sie damit anstellen, was ohne nicht geht?

1.78 Alles Schnee von gestern

Was ist bisher passiert?

Es gab von Assampler bereits eine Version, die ich 1992-1993 in GFA-Basic entwickelt hatte. Durch Verwendung des Assembler-Codes den ich in der synthesizer.library unterbrachte, waren damals die Berechnungen schon so schnell wie heute. Trotzdem war das Programm mit Intuition-Gadgets (war nämlich unter GFA-Basic nicht so einfach) ausgestattet und daher recht komfortabel zu bedienen.

Mit dieser Version bin ich dann zu Jugend forscht 1993 angetreten und immerhin bis zu einem 2. Platz in der Landesrunde gekommen. Unter die Amiga-Benutzer habe ich diese Version allerdings nie gebracht, weil ich es einfach nicht geschafft habe, ein lauffähiges Programm zu kompilieren. Wer mir hier helfen kann, sollte das unbedingt tun. Vielleicht findet das Programm unter den Besitzern von Classic-Amigas noch seine Freunde. Aufgrund von ein paar Schnitzern funktioniert die GFA-Basic-Version unter AmigaOS-Versionen ab 2.0 nicht mehr vollständig.

1.79 No future! No simple present! No past tense!

Was bringt die Zukunft?

Sequencer

Das objektorientierte Konzept bietet hervorragende Möglichkeiten zur Komposition von Musik. Viele Ideen habe ich schon (auch mit anderen Amiga-Benutzern) durchgespielt, aber es ist immer noch nicht zu konkreten Ergebnissen gekommen.

Wer hat Lust daran mitzuwirken, oder besser gleich den ganzen Sequenzer-Teil zu übernehmen?

ARexx

Trotz daß man mit den Klangalgorithmen schon fast richtig programmieren kann/muß, reicht das noch nicht aus. Einen ARexx oder einen Rebol-Port, wie auch immer, sollten zukünftige Assampler-Versionen besitzen.

Digitalisieren

Auf jeden Fall werde ich das Digitalisieren (Samplen) von Klängen integrieren. Die Möglichkeiten der Echtzeitverarbeitung, die sich damit innerhalb des Assamplers ergeben, sind riesig.

PlugIn

Bislang sind alle Klangklassen in Assampler festverdrahtet. Das ist sehr einfach zu programmieren und führt zu effizientem aber auch sehr großem Code. Besser wäre ein modulares Konzept, das zum Beispiel auf dem datatypes-System oder dem (zumindest heute 03.09.1999) noch nebulösen ARTAS aufbauen könnte und dadurch für jeden Programmierer erweiterbar wäre.

Diese Änderung zieht aber unheimlich viele weitere nach sich. Denn dann müßte ich auch andere Teile des Programmes auslagern, Schnittstellen schaffen, dokumentieren etc. Das betrifft die Verwaltung der Variablen, die Formelauswertung, Rechnen mit einheitenbehafteten Werten, verschiedene bislang private MUI-Klassen.

Nicht zuletzt muß ich dafür garantieren, daß sich das Protokoll, mit dem sich die Prozesse untereinander auf günstige Signalattribute einigen, in Zukunft nicht mehr wesentlich ändert. Zur Zeit ist es noch nicht ausgereift, und wird sich daher garantiert ändern.

Zum Schluß darf das alles auch nicht zu kompliziert werden. Kein Programmierer neuer Klangklassen will sich stundenlang einarbeiten und die Schnittstellen dürfen den Code der Klangklassen nicht so weit aufblähen, daß die Summe aller Module den jetzigen Umfang des Assampler-Hauptprogrammes um ein Vielfaches übersteigt.

Taschenrechner

Soll das Taschenrechnerfenster verbessert werden?

PPC

PowerPC-Unterstützung ist zumindest für die Klangberechnungen denkbar. Dazu müßte ein Ersatz oder Patch für die synthesizer.library geschaffen werden. Da dieser aber weder von der Parallelität der Prozessoren Gebrauch machen könnte und er mit Festkommazahlen seine Stärken nicht voll zur Geltung bringen könnte, ist die Beschleunigung womöglich nicht zufriedenstellend.

Auch hier bin ich auf Unterstützung angewiesen, da ich in meinen Amiga500 kaum mehr einen PowerPC einbauen werden kann.

Amigas der nächsten Generation

Was die Amigas der nächsten Generation bringen werden, ist noch ungewiß. Ob mein nächster Computer und damit die Entwicklungsgrundlage zukünftiger Assampler-Versionen ein AmigaInc-Amiga wird oder ein phase5-Amirage mache ich davon abhängig, welcher Rechner die bessere Sound-Unterstützung mitbringt, die bessere Entwicklungsumgebung etc.

Weitgehend ohne Einfluß auf meine Entscheidung wird bleiben, welcher Rechner voraussichtlich die besseren Marktchancen hat, denn ich möchte bestimmt nicht eine sich selbst erfüllende Prognose füttern.

1.80 Intellenztest-Anlösung

Verstärker, Phasen- und Frequenzmodulation sind lineare Operationen, es ist also egal, ob man erst mischt/verstärkt und dann eine dieser Operationen anwendet oder umgekehrt. Probieren Sie es aus!

Anheben ist im allgemeinen nicht linear, Ausnahme: Anhebung ist gleich 0. Anheben ist gleichbedeutend mit dem Hinzumischen eines Tones der Frequenz 0. Das heißt im Ausgangssignal ist der Ton mit der Frequenz null vorhanden, obwohl das im Eingangssignal nicht der Fall war. Oder um mit Definition 2 zu arbeiten: Verstärken Sie ein Signal mit dem Faktor 0, dann sehen Sie sofort, daß Verstärkung und Anhebung nicht vertauscht werden können.

Mischen ist kein Filter, schon weil der Mischer mehr als einen Eingang besitzt, Ausnahme: hinzugemischte Signale heben sich auf. Ansonsten kann man mit Mischen insbesondere auch anheben.

Gleichrichten ist nichtlinear. Beispiel: Zwei Rechteck-Schwingungen, beide genau versetzt. Mischt man erst, löschen sie sich aus, nach Gleichrichten bleibt Signal null. Richtet man erst gleich und mischt dann ist das Ergebnis ein konstantes Signal > 0.

1.81 gleitendes Maximum

Gibt's noch nicht :-(

1.82 Konstante

Gibt's noch nicht :-(

1.83 Der Klassenbaum - Fluch oder Segen der Gentechnik?

Der Baum der Soundklassen

Klang

1 Datenbasierter Klang

1.1 Digitalisierter Klang

1.1.1 Digitalisierter Klang im Speicher

1.1.2 Digitalisierter Klang auf Datenträger

1.2 Freiformkurve

1.3 Spektrum

2 Klanggruppe

2.1 Listengruppe

2.2 Stereogruppe

1.84 Was für ein Klang!

Klang

Unterklassen

Datenbasierter Klang

Klanggruppe

Prozeß

Funktion

Die Klasse der Klänge steht in Assampler über allen Dingen. Die Begriffe Sound, Klang oder Objekt sind in diesem Dokument zumeist identisch. Egal ob Sample-Sounds, Gruppen, Algorithmen oder Prozesse, alles das sind Klänge, gehören also der Klasse Klang an.

Werte

Die Werte eines Klanges können im Informationsfenster eingesehen und verändert werden. Beachten Sie auch die Möglichkeit, Variablen vorübergehend abweichende Werte (Parameter) zuzuweisen. Wenn mehrere Klassen gleiche Werte mit der gleichen

Bedeutung benutzen, dann wird dieser Wert nicht jedesmal neu aufgeführt und erklärt. Stattdessen wird er einmal in der übergeordneten Klasse erläutert.

Folgende Konventionen gelten für Klangwerte:

Anteile werden immer mit (rationalen) Zahlen von 0 bis 1 angegeben, oder entsprechend 0..100% Beispiel: Lautstärkeabfall (Decay) beim Echo-Prozeß.

Werte sind meistens nicht an bestimmte Einheiten gebunden! Sie werden nicht vom Programm ergänzt, wie das in anderen Anwendungen häufig der Fall ist, sondern müssen vom Benutzer angegeben werden und Assampler testet später nur noch, ob sie in den jeweiligen Kontext passen.

Die korrekte Einheit eines Wertes entnehmen Sie der Beschreibung der betrachteten Klangklasse. Diese steht hinter dem Variablentyp. Manchmal ist eine Einheit nur implizit vorgegeben, d.h. sie muß in einer bestimmten Weise mit den Einheiten anderer Werte zusammenpassen. Eckige Klammern kennzeichnen die Einheit des geklammerten Ausdrucks. [x] ist die Einheit der x-Achse des Ausgangssignals, [y] die Einheit der y-Achse. [Input.x] ist die Einheit der x-Achse des Klanges der durch die Input-Variable gegeben ist, [Frequency] ist die Einheit der Variable Frequency usw.

Die Zahl 0 soll hier zu den natürlichen Zahlen zählen und jede Zählung beginnt bei 0. Stellt eine Auswahlvariable n Texte zur Auswahl so sind diese von 0 bis n-1 durchnumeriert.

Einstellungen

... sind der Übersichtlichkeit wegen in zwei Teile getrennt. Der erste betrifft die Bedienung, der zweite die Klangeinstellungen.

Bedienung

Variablen

Sollen Variablen extra eingerahmt werden, oder reicht es wenn der Referenz-Knopf getrennt dargestellt wird? Eine Änderung wird erst bei neu geöffneten Informationstafeln sichtbar. Das kann ärgerlich werden, weil die Vorlage-Seiten der Klangklassen nur einmal pro Programmstart angelegt werden, eine Änderung hier also erst nach erneutem Aufruf von Assampler Erfolg zeigt.

Knopfleiste herausnehmbar

Falls aktiviert, kann man die Knopfleiste in Anzeigefenstern herausnehmen und zusammenfalten. Dazu wird MUI-Erweiterung TearOff.mcc benötigt. (Aminet:mui/dev/MCC_TearOff.lha)

Tastenkürzel

Für die besonders oft benötigten Funktionen "Wiedergabe" und "Anzeigen" bestimmter Bereiche eines Klanges können Tastenkürzel angegeben werden. Dabei kann es zu Überschneidungen kommen, da in Informationsfenstern die Variablen über Tasten aktiviert werden können. Es ist nicht vorgesehen, alle Funktionen von Assampler auf diese Weise zu erreichen, es geht nur bei diesen.

Klang/Datei

Laden IFF/Datatypes/Roh

Bestimmt, auf welche Formate beim Laden von Dateien getestet werden soll.

Beispiel 1: alle drei Formattypen sind aktiviert

Wird jetzt eine IFF-Datei geladen, durchläuft sie zuerst die Assampler-interne Laderoutine. Handelt es sich um eine 8-Bit (8SVX) oder 16-Bit (16SV) IFF-Sampledatei oder um eine andere aus Assampler abgespeicherte Datei (Spectrum, Algorithmus etc.), wird sie sofort von Assampler geladen. Handelt es sich nicht um ein IFF-Format oder kein direkt unterstütztes Format, versucht Assampler die Datei mit Hilfe eines Datatypes zu lesen. Haben Sie z.B. einen WAV- oder einen VOC-Datatype installiert, werden von Assampler auch diese auf PC's verbreiteten Dateien geladen. Schlägt auch das fehl, wird die Datei als Roh-Datei behandelt. Je nach den Lade-Einstellungen bei Sample-Sounds wird die Rohdatei auf verschiedene Aufbauformen geprüft.

Beispiel 2: IFF und Datatypes sind aktiviert

Sie besitzen Dateien in Fremdformaten, wollen aber nicht, daß Assampler unbekannte Sachen stillschweigend als Rohdateien betrachtet. In diesem Falle würden nämlich die Zusatzinformationen (meistens am Anfang einer Datei) als Teil des Klanges angesehen und hinterließen einen Störgeräusch.

Beispiel 3: nur Roh ist aktiviert

Sie können beschädigte Dateien laden, deren Formatinformationen nicht korrekt sind, möglicherweise stürzt Assampler beim Laden als IFF-Dateien gar ab. Beim Lesen von Rohdaten werden Formatinformationen ignoriert, durch Ausschneiden der fehlerhaften Daten können Sie retten, was noch zu retten ist.

Zur Zeit werden mit Datatypes und Rohformaten nur Samples eingelesen. Trotzdem befinden sich diese Einstellungen nicht bei den Sample-Einstellungen, da in Zukunft auch andere Soundtypen z.B. als Rohdaten eingelesen werden könnten.

Sichern Anzeigefenster/Informationsfenster/Zustand

In einer Klangdatei (sofern IFF) können weitere Informationen gespeichert werden. Wenn Sie in einer Sitzung eine optimale Anordnung der Anzeige- und Informationsfenster gefunden haben, können Sie diese im abgespeicherten Klang verewigen. Dazu muß die Option Sichern-Fenster aktiv sein.

Ebenso wird der offen/geschlossen-Zustand von Gruppen in einer Datei vermerkt, wenn die Option Sichern/Zustand (in Zukunft könnten weitere folgen) eingestellt ist. Unentbehrlich sind diese Optionen, wenn Sie einen ganzen Sound-Baum abspeichern, der bei jedem Programmstart von Assampler geladen werden soll (siehe unten).

Dateiauswahl Pfad

Der Pfad der bei Programmstart in das Dateiauswahlfenster eingetragen werden soll.

Dateimuster beim Laden

Gibt Ihnen Kontrolle darüber, welche Dateien aus Verzeichnissen geladen werden sollen. Wenn Sie ein ganzes Verzeichnis mitsamt seinen Klangdateien laden wollen, sichert die Einstellung "~(*.info)" daß die Piktogramme der Klangdateien nicht eingeladen werden.

Für geladenen Klang

Was soll die Bedienoberfläche unternehmen, sobald ein Klang erfolgreich eingeladen wurde?

tue nichts - ebendies

wechsele Anzeige - im aktuellen Anzeigefenster wird zum geladenen Klang gewechselt

öffne Fenster - öffnet neues Fenster für den geladenen Klang

Piktogramm

Piktogrammdatei, die für jeden von Assampler abgespeicherten Klang kopiert werden soll.

Klang/Funktionalität

Zeige zu Beginn

Der Klang, der nach dem Start im Anzeigefenster angezeigt werden soll.

Sitzungsstruktur

Die hier ausgewählte Datei, wird zu jedem Programmstart von Assampler als Wurzelgruppe geladen. Damit steht Ihnen die Möglichkeit offen, sofort zu Programmbeginn auf eine erweiterte oder völlig umgekrempelte (wovon allerdings aus Gründen der Austauschbarkeit mit Dateien von anderen Benutzern abzuraten ist) Klangpalette zugreifen zu können. Möchten Sie selbstkreierte Wellenformen, Übertragungsfunktionen, Filterfenster, Algorithmen, oder bestimmte Anzeigefenster (z.B. der Wurzelgruppe) sofort parat haben, ist das mit dieser Einstellung kein Problem.

Achten Sie darauf, daß diese Datei existiert und einen Gruppensound enthält, andernfalls verwendet Assampler seine Grundeinstellung. Beachten Sie auch, daß die Wurzelgruppe nicht mehr austauschbar ist, sobald das Programm einmal läuft. Es hat keinen Sinn andauernd die Wurzelgruppe ändern zu wollen, oder von mir die Verwaltung mehrerer Wurzelgruppen zu verlangen. Die zu Beginn erscheinende Gruppenstruktur, die voreingestellte Klänge beinhaltet, ist als Gerüst für alle weiteren Operationen anzusehen. Mehrere gleichzeitig zu bearbeitende Projekte sind als Angehörige einer Gruppe anzulegen.

Beim Beenden sichern

Sollen beim Beenden des Programms alle gerade in Assampler geladenen Klänge in dieser Datei gespeichert werden?

Zwischenablage

Die Zwischenablage ist die Gruppe, in der alle Bruchstücke landen, die bei der Kopier- und der Schneidefunktion anfallen. Die Einfügefunktionen benutzen wiederum den ersten in dieser Gruppe liegenden Klang zum Einfügen in den aktuell bearbeiteten Klang.

Interpolation

Legt den Grad der Interpolation fest, die automatisch beim Angleichen von quantisierten Sounds (Sample, Spektrum) angewandt werden soll.

Berechnungspuffer

Die Größe der beim Berechnen verwendeten Puffer.

Berechnungspriorität

Priorität des Berechnungsfadens.

Klang/Wiedergabe

Wiedergabepuffer

Die Größe der beim Abspielen verwendeten Puffer.

Wiedergabepriorität

Priorität des Wiedergabefadens.

Wiedergabe Modus + Abtastrate

Öffnet AHI-AudioMode-Requester für Einstellung von Ausgabegerät (FileSave, Paula, Soundkarten), Kanalanzahl und Qualität der Ausgabe. Requester blockiert den Rest der Assampler-Bedienoberfläche!

fixiert

Sollen Sounds immer mit der im AHI-Requester eingestellten Abtastrate wiedergegeben werden, oder soll bevorzugt eine Abtastrate nahe der klangeigenen verwendet werden? Ist der Schalter nicht gesetzt, hängt die zur Wiedergabe benötigte Rechenzeit und die erreichte Qualität maßgeblich von der Abtastfrequenz der jeweiligen Klänge ab. Anderfalls kann eine konstant (gute oder schlechte) Qualität erzwungen werden. Wobei die "gute" Qualität mit Vorsicht zu genießen ist, da jede Anpassung an andere - auch höhere - Abtastraten Qualitätsverluste mit sich bringt.

runden

Die Soundausgabehardware beherrscht womöglich die angeforderte Ausgabefrequenz nicht. Normalerweise wird dann der Sound auf die Ausgabefrequenz umgerechnet (= Qualitätsverlust). Um dies zu verhindern kann man Schalter "runden" setzen. Dann wird eine Verstimmung in Kauf genommen, dafür wird der Sound (jedenfalls in der Qualität) unverfälscht wiedergegeben.

Es ist sehr ungewöhnlich, daß Runden und Fixieren gleichzeitig aktiv sind, denn das würde bedeutet, daß alle Sample-Sounds auf ein und derselben Wiedergabefrequenz abgespielt werden, welche mit der ursprünglichen Klang-Abtastung nichts zu tun hat.

Vorgehensweise:

1. Im Audio-Mode-Requester den richtigen Treiber aussuchen. Dort erfährt man auch, welche Abtastfrequenzen die entsprechende Hardware unterstützt. Diese Liste ist aber z.B. im Falle der Amiga-internen Ausgabe (Paula-Chip) nicht vollständig!

2. Soundkarten besitzen meist nur eine kleine Auswahl an Wiedergabefrequenzen. Liegen diese immer um das doppelte auseinander, wäre die Auswirkung eines gesetzten Runden-Schalters verheerend, falls die abgespielten Sounds auf einer Frequenz mitten zwischen zwei Hardware-Frequenzen liegen, z.B. ein 16kHz-Sample auf einer Hardware, die 11kHz, 22kHz und 44kHz wiedergeben kann.

Arbeitet man sowieso nur mit Sample-Sounds der unterstützten Frequenzen (z.B. die mit der gleichen Hardware digitalisiert wurden), hilft die Rundung unnötiges Umrechnen der Sounds zu vermeiden, falls Abtastfrequenzen doch nicht exakt übereinstimmen. Die Frequenzfixierung muß dann natürlich abgeschaltet sein.

Normalerweise würde ich aber sagen, daß bei Soundkartenbesitzern der Runden-Schalter eher aus und die Fixierung eher eingeschaltet sein wird.

3. Die Amiga-interne Soundausgabe unterstützt eine sehr breite und fein unterteilte Palette von Wiedergabefrequenzen. Deshalb ist das Runden hier immer zu bevorzugen, besonders bei niedrigen Abtastfrequenzen werden Sie ganz dankbar für diese Option sein. Eine Frequenzfixierung dürfte hier eher selten sein.

Maximale Lautstärke

Ordnet der von der Soundhardware maximal erzeugbaren Auslenkung einen einheitenbehafteten Wert zu. Beispiel: maximale Lautstärke sei 2V, dann muß jeder abzuspielende Klang die y-Einheit V haben und wird bis zu 2V verzerrungsfrei wiedergegeben.

Bitte nicht verwechseln mit einem Hauptregler für die Lautstärke (Master Volume), denn hier verhält es sich genau anders herum. Machen Sie die maximale Lautstärke größer, wird die Klangausgabe leiser.

Maximale CPU-Last

Welchen Anteil an der gesamten Rechenleistung darf die Klangwiedergabe höchstens einnehmen? Wird dieser Rechenzeitanteil überschritten, wird die Klangwiedergabe mit einer Fehlermeldung abgebrochen.

Dieser Wert ist absolut wichtig um zu vermeiden, daß Sie den Amiga nicht mehr bedienen können, weil die Klangwiedergabe alle Rechenzeit beansprucht. Werte über 50% sind gefährlich.

1.85 Wir sind die Morsedaten und ziehen mit Baudraten durch's Rohr

Datenbasierter Klang

aus der Klasse Klang

Unterklassen

Digitalisierter Klang

Freiformkurve

Spektrum

Funktion

Ein datenbasierter Klang speichert sein Signal im wesentlichen in einer großen Wertetabelle.

Die Klasse der datenbasierten Klänge ist eine abstrakte Klasse. Sie können keinen solchen Klang erzeugen! Die Klasse dient nur der Vereinheitlichung der Funktionen, die Sample-, Spline- und Spectrum-Sounds bieten.

Werte

Die Werte Volume, SampleFreq, Begin bestimmen direkt die entsprechenden Signal-Attribute

Begin (Formel) [x]

Legt fest, an welchen Anfangspunkt die Daten verschoben werden sollen. Beispiel: Begin = 1s bedeutet bei einem Sample-Sound, daß erst nach einer Sekunde Pause der erste Sample-Wert ausgegeben wird. Anderes Beispiel: Mischt man Sample-Sound A mit Begin 0 und Sample-Sound B mit Begin 1s enthält das Resultat eine Sekunde lang nur Klang A und dann setzt Klang B ein und überlagert Klang A.

Length (Formel) [x]

Länge des Klanges. Ändert man diesen Wert manuell (<ENTER> nicht vergessen!), wird die Länge des Klänges entsprechend angepaßt. Wird der Klang dadurch länger, wird der Rest mit Nullen aufgefüllt, wird er kürzer gehen die Daten am Ende verloren.

SampleFreq (Formel) 1/[x]

Abtastfrequenz des Klanges. Alle x-Werte sind intern ganze Zahlen. Erst durch die Abtastfrequenz wird angegeben wie schnell die Signaldaten ausgegeben werden sollen. Je schneller, desto präziser ist das Abbild von der Wirklichkeit im Rechner und umso höher ist der Speicherverbrauch.

Volume (Formel) [y]

Die Lautstärke des Klanges legt fest, welche Einheit die y-Werte des Klanges haben sollen und wie groß die maximale Auslenkung ist. Stellen Sie hier zum Beispiel 3m ein, enthält der "Klang" Längenangaben als Werte, welche sich innerhalb des Bereiches -3m bis +3m bewegen können.

Anzeige

Anzeigen von Datensounds erkennt man an den Scrollbalken unten und rechts.

sichtbarer Bereich

Schon bei Samplebearbeitungsprogrammen hat sich die Vorgehensweise bewährt, Sample-Sounds in verschiedenen Vergrößerungsstufen darzustellen. Je mehr vergrößert wird, desto mehr Details sind zu sehen, aber desto weniger Sample-Werte sieht man. Der Ausschnitt, welcher betrachtet wird, kann mit dem unteren Scrollbalken und den dazugehörigen Pfeiltasten verschoben werden.

Die Größe des Ausschnittes kann mit den Anzeige-Funktionen aus dem Hauptfenster festgelegt werden. Darüberhinaus existiert die Möglichkeit die Größe des Balkens direkt zu ändern. Außerdem kann die Anzeigenlänge im Zahlengadget AL (Anzeigenlänge) abgelesen und geändert werden. (<ENTER> nicht vergessen!)

Auch in der Höhe kann der Ausschnitt variiert werden, um z.B. Details in leisen Passagen betrachten zu können. Änderung von Größe und Position des Höhenausschnittes ist nur direkt am Proportionalgadget möglich.

markierter Bereich

Insbesondere für die Schnittfunktionen ist es wichtig, Bereiche markieren zu können. Schnittfunktionen und einige Möglichkeiten zur Bereichsmarkierung finden sich im Hauptfenster.

Individuelle Bereiche können markiert werden, indem man mit der Maus in die Anzeige eines Klanges klickt, die linke Maustaste hält, die Maus zum Ende des anvisierten Bereiches schiebt, und dort die Maustaste losläßt. Ziehen über die Grenzen der Anzeige hinaus führt zum Weiterschieben des sichtbaren Ausschnittes, das umso schneller, je weiter die Maus über der Grenze steht. Nachträglich kann die Endmarke verschoben werden, wenn bei gehaltener Umschalttaste (voreingestellt ist Shift) mit der linken Maustaste geklickt wird.

Die Bereichsmarken können als Zeitangaben in den Zahlengadgets rechts unter der Anzeige abgelesen und eingegeben werden. Es bedeuten BB - Bereichbeginn, BE - Bereichsende, BL - Bereichslänge.

Ansichtenteilung

Für die Suche von günstigen Schleifenpunkten unentbehrlich: Das Teilen der Anzeige in Ausschnitte gleicher Größe aber unterschiedlicher Position. Wurde aus praktischen Erwägungen auf vier Teile beschränkt. Die Anzahl der Ansichten wird durch den Auswahlknopf rechts ganz unten im Anzeigefenster festgelegt. Für mehrere Ansichten unterschiedlicher Vergrößerung sollten weitere Anzeigefenster vom gleichen Klang geöffnet werden.

Einstellungen

Anlegen in

In dieser Gruppe werden datenbasierte Klänge abgelegt, die durch Doppelklick im Klassenfenster erzeugt wurden.

Umschalttasten in Anzeigen von datenbasierten Klängen

In beiden Tastenbeschreibungen muß "lbutton" vorkommen, obwohl es zur Zeit auch ohne gehen würde, weil immer die linke Maustaste erwartet wird. Sollte es in späteren Versionen auch mal ohne linke Maustaste gehen, dann funktioniert Ihre jetzige Einstellung trotzdem, sofern sie "lbutton" dazuschreiben.

Proportionalgadget-Zoom

Taste, die gehalten werden muß, um an den Proportionalgadgets statt den Ausschnitt zu verschieben, die Größe des Ausschnitts zu verändern.

Bereichserweiterung

Taste, die gehalten werden muß, um einen schon ausgewählten Bereich nachträglich zu vergrößern oder zu verkleinern.

Seite bei Erweiterung

Welche der beiden Bereichgrenzen (also linke oder rechte) soll bei Bereichserweiterung mit der Maus wieder aufgenommen werden? Die Stelle, an der man losgelassen hat oder an der, die der Maus gerade am nächsten ist?

ständig aktuelle Knopfleiste

Wenn angekreuzt, dann werden in der Anzeige ständig sämtliche Angaben aufgefrischt, die vom markierten Bereich abhängen, während man einen Bereich markiert.

1.86 Hier geht es um simple Samples

Digitalisierter Klang

aus der Klasse Datenbasierter Klang

Unterklassen

Digitalisierter Klang im Speicher

Digitalisierter Klang auf Datenträger

Funktion

Der Sample-Sound entspricht dem herkömmlichen Begriff von Computerklang. Ein Sample-Sound enthält die Werte, die ein Signal mit der Zeit durchläuft, wobei sich die Genauigkeit der x-Werte und y-Werte variieren läßt. Ein Sample-Sound entspricht gewissermaßen einer endlosen Zahlenkolonne, wobei jeder Wert angibt wie weit zum Beispiel eine Membran zu aufeinanderfolgenden Zeitpunkten ausgelenkt werden soll.

Mit Sample (engl. Probe) bezeichnen die einen einen einzelnen Wert, andere meinen damit den kompletten Klang. Der Klarheit wegen verwende ich hier Sample-Sound oder digitalisierten Klang für den ganzen Klang und mit Sample-Wert meine ich einen einzelnen Wert.

Werte

BaseFreq (Formel) 1/[x]

Die Frequenz des Tones der durch den Sample-Sound repräsentiert werden soll. Ob der Sound diesen Ton wirklich enthält ist damit nicht gesagt. Manchmal ist es auch gar nicht möglich eine einzelne Frequenz anzugeben, weil der Klang zum Beispiel aus Rauschen oder einem Akkord besteht.

Wirklich gebraucht wird der Wert nur für Schwingungsanimationen.

LoopBegin (Formel) [x]

Kennzeichnet den Schleifenbeginn einer Klangschleife.

LoopLength (Formel) [x]

Kennzeichnet die Schleifenlänge einer Klangschleife.

Anwendung

Sample-Sounds sind meistens das Endprodukt einer Berechnung. Wann immer Sie einen Effekt berechnen, werden Sie als Ziel einen Sample-Sound angeben. Sample-Sounds lassen sich abspeichern und in anderen Programmen weiterwenden und umgekehrt - für alle anderen Klangklassen im Assampler gilt das nicht!

Tips

Schwingungsform-Animation

Die Schrittweite, um die die Pfeiltasten am horizontalen Schieber den Ausschnitt verschieben, können Sie bei Samplesounds über die Grundfrequenz (BaseFreq) beeinflussen. Wissen Sie die Grundfrequenz des Tones, dann sollten Sie diese als BaseFreq im Informationsfenster eintragen. Halten Sie eine Pfeiltaste gedrückt, müßte bei jedem Weiterrücken jede Schwingung den Platz ihrer benachbarten Schwingung einnehmen. Es ergibt sich eine Animation, in der man erkennen kann, wie sich die Wellenform mit der Zeit verändert. Wählen Sie einen ganzzahligen Teil (Hälfte, Drittel ...) der Grundfrequenz, werden mehrere Phasen übersprungen und die Animation wird schneller.

Verwechseln Sie jetzt aber nicht die Grundfrequenz des durch den Samplesound dargestellten Tones mit der Abtastfrequenz des Samplesounds, denn diese muß immer deutlich größer sein. Also wenn Ihnen ein Sample eines eingestrichenen C eines Instruments vorliegt, dann erhalten Sie dessen Frequenz über Tone(c1). Natürlich funktioniert der Animationseffekt nur bei sauberen Tönen, bei Akkorden oder gar Rauschen dürfte man nicht allzu viel sehen, bei frequenzmodulierten Tönen könnten einem die Wellen weglaufen.

Einstellungen

Datei/Laden

Bits

Endian

Vorzeichen

Hier kann man auswählen, welche Aspekte des Sample-Formates beim Einladen von Roh-Daten von Assampler ermittelt werden sollen und welche man selbst vorgeben möchte.

Analyse

Bei großen Dateien kann die Ermittlung des Sample-Formates sehr langwierig werden. Meistens reicht es schon ein paar Bytes zu Beginn zu analysieren. Hier kann man angeben wieviele.

Datei/Sichern

Typ

IFF:

Assampler speichert Sample-Sounds mit eigenen Routinen ab, im IFF-8SVX oder IFF-16SV. Dabei werden auch alle Zusatzinformationen gesichert, wie benutzerdefinierte Variablen und Informationen über geöffnete Fenster. Diese Zusatzinformationen werden von anderen Programmen ignoriert, sofern sie nicht andere Erweiterungen mit gleicher Kennung benutzen. Sauber ist es von mir jedenfalls nicht, solche Zusätze eigenmächtig in das Format einzubringen. Sie gehen im übrigen verloren, wenn man Assampler-Dateien in andere Programme lädt und von dort wieder abspeichert.

Roh:

Assampler speichert die blanken Sample-Daten ab.

Datatypes:

Assampler speichert die Sample-Daten mit Hilfe eines im Betriebssystem installierten Datentyps ab. Leider unterstützen die wenigsten Datentypen diesen Vorgang, dann werden die Daten ohne weitere Warnung im IFF-8SVX abgelegt. Zur Zeit unterstützen datatypes nur 8-Bit-Daten.

Bits

Endian

Vorzeichen

Beim Abspeichern von Rohdaten, kann man alle Aspekte des Sample-Formates vorgeben. Beim IFF-Format entscheidet die Anzahl der Bits zwischen dem 8SVX- und dem 16SV-Format.

Datentyp

Der zum Abspeichern verwendete Datentyp.

Datei/Funktionalität

Blockgröße

Maximale Größe der Blöcke, in die die Sample-Daten zerlegt werden. Kleine Blöcke sorgen für zerstückelten Speicher (Fragmentierung), große Blöcke können Editieroperationen ausbremsen.

Datei-Puffer

Die Daten aus der Datei eines Sample-Sounds auf einem Datenträger werden stückweise in den Speicher geladen. Die Puffer-Größe gibt an, wie groß diese Stücken sind.

Kopierablage

Das Verzeichnis, in dem die Dateien für die Zwischenablage-Klänge angelegt werden. Nur relevant für Sample-Sounds auf Datenträgern.

Farbauswahl für

Kurvenzug

Nulllinie

Hintergrundmuster

Bereichsmarkierung

Wiedergabe-Zeiger

Kurve (Auswahl)

Linie - es wird ein Kurvenzug durch alle Punkte gelegt

Punkte - es werden die Punkte selbst gezeichnet

Füllen - es wird die Fläche unter der Kurve gefüllt

Nulllinie (Schalter)

Mit dem Schalter kann man Nulllinie ein- oder ausblenden.

schnelle Anzeige

Kurve wird erst in einen internen Puffer gezeichnet und dann mit einem Schwung zur Anzeige gebracht. Selbst wenn das Verfahren nicht schneller als das direkte Zeichnen ist, flackert das Bild wenigstens nicht so.

1.87 Hier geht es um simple Samples

Digitalisierter Klang im Speicher

aus der Klasse Digitalisierter Klang

Funktion

Sample-Sounds in dieser Form werden im Speicher abgelegt, werden also bei Beendigung des Programmes gelöscht. Ihre Bearbeitung geht besonders schnell, dafür sind Sample-Sounds im Speicher durch die Größe desselben in ihrem Umfang eingeschränkt.

1.88 Hier geht es um simple Samples

Digitalisierter Klang auf Datenträger

aus der Klasse Digitalisierter Klang

Funktion

Sample-Sounds können auch direkt auf Datenträgern bearbeitet werden, wenn sie dort in einem Roh-Format vorliegen. Die Bearbeitung zerstört das Originalmaterial, eine Rücknahme von Operationen ist nicht möglich! Legen Sie deshalb vorsichtshalber Kopien an, oder verwenden Sie solche Sounds nur als Eingänge für Algorithmen. Um ganz sicher zu gehen, können Sie mit dem Protect-Befehl im CLI oder mit einem vergleichbaren Befehl eines Dateiverwaltungsprogrammes (DirectoryOpus, MegaDir, DiskMaster) das Überschreiben und Löschen einer Datei verbieten.

Werte

Die folgenden Werte sind keine echten Variablen, und können daher nicht parametrisiert werden.

Path

Die Datei aus der die Daten bezogen bzw. in die Daten hineingeschrieben werden sollen.

Bits, Endian, Sign

Bestimmen, als welches Sampleformat die Daten in der Datei angesehen werden sollen.

Technik

Legt man eine Kopie eines Disk-Sample-Sounds mit Hilfe der Gruppen-Schnittfunktionen an, wird ein neues File auf der Disk angelegt. Problem: Welchen Namen soll diese Datei bekommen? Ich habe es so eingericht, daß, wann immer eine Datei Ziffern im Namen enthält, die Ziffern am weitesten rechts als Zahl betrachtet werden. Diese wird dann um eins erhöht. Besitzt eine Datei keine solche Nummer, wird eine dreistellige Nummer eingefügt, und zwar genau vor der Dateinamensendung (wie ".smp"), falls vorhanden, sonst am Ende des Namens. Achten Sie darauf, daß es hier keine Überschneidungen mit Ihrer eigenen Bezeichnungsweise gibt, sonst können Ihnen dadurch andere Dateien verlorengehen.

1.89 Ganz schön spleenig

Freiformkurve

aus der Klasse Datenbasierter Klang

Funktion

Erlaubt das intuitive Erstellen von Hüllkurven, Steuerkurven und Wellenformen. Freiformkurven sollen das Freihand-Editieren von Sample-Sounds ersetzen, da das mit einer einfachen Maus meist nicht besonders gut gelingt.

Die Freiformkurve wird durch einen kubischen Spline realisiert, also aneinandergereihte Kurven von kubische Polynomen. Die Stützstellen, d.h. die Intervallgrenzen und die Ableitungen in den Stützstellen geben Sie vor.

Werte

Criterion (Auswahl)

Das Kriterium mit dem beurteilt wird, wie gut die Freiformkurve das urspringliche Signal annähert:

ErrorSquare - die Summe der Quadrate der Abweichungen (Fehlerquadratsumme)

MiniMax - die maximale Abweichung

Dieses Kriterium wird nur eingesetzt, wenn man in mehreren Iterationen eine verbesserte Stützstellenwahl berechnet haben möchte. Die Anpassung der Freiformkurve an das Original bei festem Stützstellenschema erfolgt immer unter der Bedingung einer minimalen Fehlerquadratsumme.

Nodes (Formel) 1

Nur beim Erzeugen einer Freiformkurve von Interesse: Gibt die Anzahl der zu Beginn vorhandenen Knoten vor.

Optimization (Schieber)

Wenn beim Konvertieren in eine Freiformkurve auch die Stützstellen (seitlich) verschoben werden sollen um eine optimale Anpassung zu liefern, muß man hier einen ganzzahligen Wert größer 0 angeben. Dieser Wert gibt die Anzahl der Versuche an, mit denen Assampler probiert, das Ergebnis zu verbessern.

Der Algorithmus dafür ist noch nicht sehr ausgegoren, man braucht in der Regel sehr viele Iteration und bekommt in schwierigen Fällen auch dann noch kein vernünftiges Ergebnis. Sie machen also nichts falsch, wenn Sie den Wert auf 0 setzen und die Stelle der Stützpunkte selbst vorgeben, so daß Assampler nur noch die Höhe der Punkte bestimmen muß. - Das geht nämlich exakt und in überschaubarer Zeit.

Anzeige

Es wird der Verlauf der Freiformkurve dargestellt, wobei die Stützstellen durch senkrechte Striche markiert sind. Ein Klick auf ein solches Stützstellenkreuz läßt ein Gebilde erscheinen, daß aus drei miteinander verbundenen Kästchen besteht. Verschiebt man das mittlere Kästchen, ändert man den Stützpunkt. Die beiden äußeren Kästchen erlauben es, linksseitigen und rechtsseitigen Anstieg der Kurve in der Stützstelle vorzugeben.

Kontextmenü

Neue Stützstelle

Fügt rechts neben der aktuellen Stützstelle eine neue ein.

Lösche Stützstelle

Entfernt aktuelle Stützstelle.

Lineare Verbindung

Ändert die Anstiege in den Stützstellen so ab, daß die Kurve zwischen dem aktuellen Stützpunkt und seinem rechten Nachbarn zur linearen Funktion wird. Kann auch Auswirkungen auf die direkt angrenzenden Intervalle haben.

Gemeinsame Tangente

Wenn der Schalter gesetzt ist, werden linksseitiger und rechtsseitger Anstieg immer auf dem gleichen Wert gehalten. Das garantiert einen glatten Kurvenverlauf in diesem Stützpunkt.

Einstellungen

Tangente

Sollen die Tangenten an den Stützstellen bei neu erzeugten Freiform-Kurven gekoppelt oder unabhängig voneinander sein?

Knotengröße

Größe der Kästchen für die Stützstellen in Bildpunkten

Knotenradius

Abstand der Tangenten-Kästchen zum Stützstellen-Kästchen in Bildpunkten

Farbauswahl für

Kurvenzug

Stützstellen-Strich (unangewählt)

Stützstellen-Kästchen (angewählt)

Nulllinie

Hintergrundmuster

Bereichsmarkierung

Nulllinie (Schalter)

Mit dem Schalter kann man Nulllinie ein- oder ausblenden.

1.90 Ein weitgefächertes Spektrum an Einsatzmöglichkeiten

Spektrum

aus der Klasse Datenbasierter Klang

Funktion

Der Spektrum-Klang enthält Daten im Umfang wie der Sample-Sound, speichert sie aber in einer Form, die etwas intuitiver ist und bestimmte (aber nicht alle) Operationen vereinfacht. Er unterteilt ein Signal zeitlich in Blöcke und merkt sich mit welcher Intensität und Phasenlage (Zeitverzögerung) jede mögliche Frequenz in diesem Block vorkommt.

Werte

BaseFreq (Formel) 1/[x]

Grundfrequenz - Die Periode dieser Frequenz gibt die Blocklänge an, mit der das Signal zerlegt wird. Enthält der Klang ohnehin nur einen Ton, ist es zweckmäßig, die Grundfrequenz auf die Frequenz dieses Tones einzustellen, oder auch auf einen subharmonischen Anteil. (Subharmonisch zur Frequenz f bedeutet f/n, n natürliche Zahl>1.) Ausschlaggebend ist, was zum Schluß im Feld Wellenlänge steht, also <ENTER> nicht vergessen!

Je kleiner die Grundfrequenz desto genauer ist die Auflösung der Frequenzen, umso mehr Platz in senkrechter Richtung braucht die Anzeige und umso schlechter ist die Auflösung in der Zeit.

Deviation (Formel) 1

Relative Abweichung innerhalb der nach der günstigsten Primzahlzerlegung gesucht werden soll. Beispiel: BaseFreq 100Hz, Deviation 10% dann <ENTER> sucht von 90Hz bis 110Hz nach einer Grundfrequenz, welche zu einer optimalen Primzahlzerlegung führt.

MaxPrime (Formel) 1

Der Aufwand der Frequenzanalyse kann gering gehalten werden, wenn man eine günstige Blocklänge wählt. Dabei ist eine geeignete Primzahlzerlegung derselben wichtig. Gibt man in MaxPrime eine kleine Zahl ein (+<ENTER>), sucht Assampler nach einer Blocklänge, die einerseits wenig von der durch BaseFreq vorgegebenen Blocklänge abweicht und zum anderen nur Primfaktoren enthält, die MaxPrime nicht überschreiten.

MergeWaves (Formel) 1

Die Zerlegung des Signals in Blöcke hat den Nachteil, daß die Blöcke später womöglich nicht zusammenpassen (es gibt Sprünge beim Wechsel zum nachfolgenden Block), wenn Operationen jeden Block für sich bearbeiten. Deshalb kann man hier angeben, wie viele Blöcke insgesamt innerhalb einer Grundperiode angelegt werden sollen. 1 bedeutet es gibt keine Überlappungen, 2 bedeutet es werden doppelt so viele Blöcke wie notwendig verwendet usw.

Anzeige

In einem Feld werden die Frequenzanteile als Helligkeitswerte visualisiert. Die Phaseninformationen werden bei der Anzeige ignoriert. Von links nach rechts verläuft die Zeitachse, von unten nach oben die Frequenzachse. Die Amplitude der Frequenz 0 (Gleichstromoffset) kann man also am unteren Rand ablesen.

Der Schiebebalken am rechten Rand ist nach wie vor für die y-Werte zuständig, nur daß diese jetzt als Punkte unterschiedlicher Helligkeit zu sehen sind. Ist das Bild zu dunkel, muß man den Balken verkleinern, ist es zu hell, muß man ihn vergrößern.

Es ist nicht möglich in senkrechter Richtung feiner oder gröber aufzulösen.

Anhand der Anzeige kann man in etwa sehen, wie sich ein Klang anhört. Helle Punkte im oberen Bereich deuten auf helle, spitze, scharfe Klänge hin, helle Punkte im unteren Bereich dagegen auf dumpfe, matte Töne. Chaotische Punktwolken weisen auf Rauschen hin, klare waagerechte Linien dagegen auf Töne. Auch die Wirkung von Filtern läßt sich hier gut nachvollziehen: Kammmuster entstehen durch Phaser, verschwommene Kurvenzüge können durch modulierte Bandpassfilter erzeugt werden.

Einstellungen

Bereich

Farbe der Bereichsmarkierung

Zeiger

Farbe des Positionsstriches beim Abspielen (Nennen Sie das Ding meinetwegen sonstwie, ich meine jedenfalls den Strich, der während der Wiedergabe eines Klanges über's Bild zuckelt.)

Farbverlauf

Die Farben, die die Amplituden der Frequenzanteile beim Anzeigen eines Spektrums repräsentieren. Die Farbe ganz links bedeutet eine nicht vorhandene Frequenz, die Farbe ganz rechts eine Frequenz mit maximaler Amplitude.

Damit Sie nicht jede einzelne Farbe des Farbverlaufs einstellen müssen, wird der Farbverlauf nur an wenigen Stützstellen festgemacht. Zur Zeit müssen sie mit fünf Stützstellen, d.h. vier Intervallen auskommen. Wenn Sie meinen, daß Sie es brauchen, könnte ich mich auch zu einer Lösung mit flexibler Anzahl der Stützstellen durchringen. Das wäre nur deutlich aufwendiger zu programmieren.

In den Stringgadgets tragen Sie die Intervallbreiten ein. Eine Intervallbreite schließt immer eine Stützstelle mit ein, so daß die Summe aller Intervallbreiten plus 1 genau die Anzahl der benötigten Farben ergibt. Achten Sie darauf, daß Sie nicht mehr Farben von einem Bildschirm erwarten, als er besitzt. Darüber hinaus sind acht Farben eines Bildschirms vorbelegt und passen höchstwahrscheinlich nicht in Ihren Farbverlauf. Deshalb sollte die Summe der Intervallbreiten die Anzahl der freien Bildschirmfarben nicht überschreiten. Mißachten Sie diese Faustregel, werden die Farben nicht optimal verteilt - kaputtmachen können Sie allerdings nichts.

Sind Ihnen vier Intervalle sogar schon zuviel, können Sie den Farbverlauf abrechen, in dem Sie im Stringgadget unter der zuletzt zu benutzenden Farbe eine 0 eintragen.

Beispiel 1: Bildschirm mit 32 Farben

Farben: schwarz, blau, cyan, weiß

Intervalle: 5, 5, 5, 0 -> benötigt 16 Farben, bleiben noch 16 für andere Zwecke (Zeiger, Bereichsmarkierung, Spline-Stützstellen ...)

Zwischen den vier Farben werden jeweils 4 Farben eingefügt um den Farbverlauf feiner zu gestalten.

Beispiel 2: Bildschirm mit 256 Farben

Farben: schwarz, rot, gelb, weiß

Intervalle: 90, 70, 50, 0 -> benötigt 211 Farben, bleiben noch 45 für andere Zwecke

In den oberen Intervallen werden weniger Farben eingefügt, weil dort weniger Helligkeitsunterschiede zu überwinden sind.

1.91 Sammlung von Klängen

Klanggruppe

aus der Klasse Klang

Unterklassen

Listengruppe

Stereogruppe

Algorithmus

Funktion

Die Hauptaufgabe dieser Sounds besteht im Verwalten anderer Sounds. Die Art der Gruppe legt fest, als was die Gruppenangehörigen zu betrachten sind.

Die Klasse der Gruppen ist eine abstrakte Klasse. Sie können keine reine Gruppe (nicht verwechseln mit Listengruppen) erzeugen! Die Klasse dient nur der Vereinheitlichung der Funktionen, die Listen-, Stereo- und Algorithmus-Gruppen bieten.

Einstellungen

Anlegen in

In dieser Gruppe werden Gruppen abgelegt, die durch Doppelklick im Klassenfenster erzeugt wurden.

Sichern als

Sollen Gruppen mitsamt ihren untergeordneten Klängen in eine Datei geschrieben werden, oder sollen die Gruppe und etwaige Untergruppen als Verzeichnisse abgespeichert werden. Im letzteren Fall gehen sämtliche Detailinformation der Gruppe verloren, also Art der Gruppe (Liste, Stereo, Algorithmus), benutzerdefinierte Variablen und Informationen über geöffnete Fenster.

1.92 Er siegte nicht mit Gewalt sondern mit Liszt

Listengruppe

aus der Klasse Klanggruppe

Funktion

Die Listengruppe erlaubt es, verschiedene Klänge aufzunehmen. Damit ist eine Listengruppe so etwas wie ein Verzeichnis auf Diskette.

Führt man eine Berechnung an einer Listengruppe aus, wird die Berechnung stattdessen an allen von ihr verwalteten Klängen ausgeführt. Haben dabei beteiligte Prozesse selbst Gruppen als Eingänge, werden diese wiederum der Reihe nach durch ihre untergeordneten Klänge ersetzt. Beispiel: Startet man einen Verstärkerprozeß mit Volume=2 auf einer Listengruppe, dann werden alle untergeordneten Klänge um den Faktor zwei verstärkt.

Für jede Teilberechnung bekommt ein Unterklang in der Zielgruppe die temporäre Variable Index, welche die Position innerhalb der direkt übergeordneten Gruppe enthält (von null an gezählt). Damit ist es möglich, Werte in der Berechnung an die einzelnen Zielklänge anzupassen.

Spielt man eine Listengruppe ab, werden alle enthalten Klänge der Reihe nach abgespielt.

Werte

Multipitch (Schalter)

Kennzeichnet Gruppe als Sammlung von mehreren Sounds des gleichen Instruments mit unterschiedlicher Tonhöhe. Der Schalter wird automatisch beim Einladen von Mehr-Oktav-Samples gesetzt. Er besitzt keine Funktionalität, sondern informiert nur über die Herkunft der Gruppe. In Zukunft soll er helfen, diese Klänge über datatypes wieder als Mehroktav-Klänge exportieren zu können. Zur Zeit kann man nämlich keine Mehr-Oktav-Samples abspeichern.

Anzeige

Mit einfachem Klick lassen sich Einträge aus der Liste markieren. Diese Markierung entspricht im wesentlichen der Bereichsmarkierung der Datenklänge.

Ein Klick auf das Bildchen links neben einem Gruppennamen macht die dort eingeordneten Klänge unsichtbar oder auch wieder sichtbar. Gruppen einzuklappen ist manchmal der einzige Weg, sich in einer verwinkelten Verzeichnisstruktur einen Überblick zu verschaffen. Der Faltungszustand von Gruppen läßt sich speichern.

Ein Doppelklick auf einen Eintrag öffnet ein Fenster des entsprechenden Sounds. Welche Fensterart (Anzeige oder Information) verwendet wird, entscheidet Assampler allerdings selbst.

Kontextmenü

Öffne Anzeige-/Informationsfenster

Öffnet ein Fenster für den gerade markierten Klang.

Drag&Drop

In der Anzeige ist es möglich die untergeordneten Klänge mittels Drag&Drop neu anzuordnen. Weiterhin kann ein Eintrag auch in eine andere Listengruppen- oder Algorithmus-Anzeige verschoben werden, damit wird der Klang selbst verschoben, oder der Eintrag wird in ein Klangwahl-Gadget geschoben, dann wird dort der entsprechende Sound eingestellt.

Geschichte

Ursprünglich gab es noch eine weitere Gruppenklasse, die direkt für die Aufnahme von Mehroktav-Klängen vorgesehen war (siehe Multipitch-Schalter). Jedoch brachte das keine Vorteile, da sie keinerlei Funktionalität über die einer normalen Listengruppe hinaus benötigte und so wurde sie wieder entfernt, ehe sie jemand erblicken konnte.

Früher benutzten manche Musikprogramme diese Besonderheit des IFF-8SVX um die begrenzte Wiedergabefrequenz der Amiga-Hardware (damals 28kHz) zu überwinden. Dem eigentlichen Sample-Sound werden noch zwei oder vier weitere in der gleichen Datei hinzugefügt, wobei die zusätzlichen Sample-Sounds den gleichen Klang eine oder zwei Oktaven höher bzw. niedriger darstellen. Die Längen dieser Zusatz-Klänge sind vorgeschrieben auf das doppelte, vierfache bzw. halbe, viertel. Daran erkennt man, daß diese Erweiterung nur für algorithmisch transponierte Klänge (resampling) taugt. Damit war diese Erweiterung meines Erachtens von Anfang an überflüssig, denn das Umrechnen hätten auch die Musikprogramme erledigen können.

Trotz allem ist eine derartige Erweiterung nicht vom Tisch. Es wäre sehr sinnvoll von digitalisierten Klängen Proben in verschiedenen Tonhöhen zu nehmen und in einer Datei zu speichern, es müssen auch nicht unbedingt Oktavabstände sein. Ein Instrument wie die Violine klingt in verschiedenen Tonhöhen wirklich anders, jedenfalls so, daß es durch einfaches Strecken/Stauchen (resampling) der Daten nicht ersetzt werden kann.

1.93 Die Stereo-/Mehrkanalgruppe

Stereogruppe

aus der Klasse Klanggruppe

Funktion

Die Stereogruppe (oder besser Mehrkanalgruppe) verwaltet mehrere Klänge (Kanäle) der gleichen Klasse, welche insbesondere eine Datenklang-Klasse sein muß. D.h. ein Stereosound kann zur Zeit nur aus Samplesounds bestehen, oder nur aus Spektren oder nur aus Freiformkurven. Der wesentliche Unterschied zur Listengruppe ist, daß die Klänge hier eher als zeitlich parallel betrachtet werden, während in der Listengruppe die Klänge zeitlich hintereinander angeordnet sind.

Enthält so eine Gruppe zwei Kanäle, haben wir es mit einem konventiellen Stereosound zu tun. Sind es vier Kanäle könnte man von einem Quadrophonie-Klang sprechen. Auch wenn Assampler anbietet, die Kanäle mit Standardnamen zu belegen, behandelt er alle Kanäle gleich, wenn es der Benutzer nicht ausdrücklich anders anordnet. Spezielle Stereophonie- oder Quadrophonie-Effekte sind im Assampler nicht integriert. Sie können diese entweder selbst basteln oder auf die Beispielalgorithmen zurückgreifen.

Werte

Die Werte entsprechen denen der Datensounds. Beim Erzeugen eines Stereosounds werden alle Variablen der Kanäle so initialisiert, daß sie die Werte der Stereogruppe verwenden. Trotzdem ist es möglich Stereogruppe und Kanälen verschiedene

Abtastfrequenzen, Längen u.ä. zu geben. Das System ist noch nicht ganz ausgegoren und man sollte keine Annahmen darüber machen, wie Assampler auf verschiedene Einstellungen jetzt und in Zukunft reagiert.

Erzeuge

Anzahl und Klasse der Kanäle die gleich beim Anlegen des Stereosounds erzeugt werden sollen. Die Einstellungen für die Kanäle müssen bei den Vorgaben für die entsprechende Klasse vorgenommen werden.

Technik

Bei der Wiedergabe werden die einzelnen Kanäle auf der ganzen Panorama-Breite von ganz links nach ganz rechts gleichmäßig verteilt. Bei einem Stereosound haut das genau hin, der linke Kanal ist nur auf dem linken Lautsprecher zu hören, der rechte Kanal nur auf dem rechten Lautsprecher. Bei mehr Kanälen gibt es ein Mischmasch, wenngleich alle Kanäle zu hören sind.

Technisch wäre es nun jedenfalls möglich, für eine Quadrophonie-Soundkarte einen AHI-Treiber zu schreiben, der Kanäle anhand des Panorama-Parameters zuweist und somit einen Quadrophonie-Klang tatsächlich auf vier getrennten Lautsprechern wiedergibt.

Ist eine Stereogruppe Ziel einer Berechnung, wird die Berechnung nacheinander auf allen Kanälen ausgeführt. Dabei erhält jeder Kanal vorübergehend die Variable Channel, welche die Nummer des Kanals enthält. Der Zugriff aus einer Formel heraus erfolgt mit "_.Channel". Somit kann die Berechnung für jeden Kanal verschieden sein. Da die Berechnungen hintereinander ausgeführt werden, ist z.B. beim Erreichen des zweiten Kanals der erste bereits modifiziert. Das ist bei solchen Effekten wie Stereo-Echos zu beachten. Dieses Verhalten wird in späteren Assampler-Versionen korrigiert, bis jetzt muß man in solchen Fällen eine zweite Stereogruppe als Ziel verwenden.

Theorie

Der Mensch besitzt von seinen Ohren gleich zwei Stück, so daß er in der Lage ist, Geräusche räumlich wahrzunehmen. Deshalb hat man sich wohl überlegt, daß das Aufzeichnen und Wiedergeben von zwei Tonspuren ausreichend sei und man gab dem Verfahren den Namen Stereophonie = Raumklang.

Aber ganz so einfach geht es doch nicht. Spielt man das Geräusches über eine Stereoanlage ab, hört das linke Ohr auch Signale, die nur für das rechte Ohr bestimmt sind und umgekehrt. Na gut, dem kann man mit Kopfhörern abhelfen.

Wie muß man nun ein Geräusch bearbeiten, damit es sich anhört, als ob es von einer vorgegeben Position im Raum stammt? Wir wollen der Einfachheit halber annehmen, daß unsere Ohren Empfänger sind die aus allen Richtungen gleich gut empfangen. Dann unterscheiden sich die Signale, die am linken und am rechten Ohr ankommen in Lautstärke und Phasenlage. Die Abnahme der Lautstärke (Oberfläche der sich ausbreitenden Schallkugel verhält sich quadratisch zum Radius) und die Zeitverzögerung (Weg durch Schallgeschwindigkeit) lassen sich mit entsprechenden Modellen recht leicht simulieren. Beim Panorama/Balance-Effekt an der Stereoanlage wird nur die Lautstärke beeinflußt. Besonders stereophon klingt dieser Effekt aber nicht, was wohl daran liegen dürfte, daß die Phaseninformation mindestens genauso wichtig ist.

Schauen wir mal wie sich die Phasenlage maximal unterscheiden kann. Den größten zeitlichen Versatz der empfangenen Signale erreichen wir sicherlich wenn Schallquelle und beide Ohren auf einer Geraden liegen. Unsere beiden Ohren haben in etwa einen Abstand von 2 dm. Das ergibt eine Signallaufzeit von 0.2 m / mach = 0.2 m / (330 m/s) = 0.6 ms. Damit man noch unterscheiden kann, ob ein Signal von links oder von rechts kommt, müßte man die Periode nochmals doppelt so groß wählen, also 1.2 ms, das entspricht einer Frequenz von 825 Hz. Nach meiner Modellrechnung wäre das die maximale Frequenz, bei der man anhand der Phasendifferenz die Schallherkunft orten könnte. Das wäre aber schlecht, denn der Bereich, in dem unsere Sprache angesiedelt ist und in dem wir besonders gut hören ist um die 3 kHz. Mal sehen, was ich da wieder falsch gemacht habe ...

Eine besseres Modell erhält man, wenn man die verknorkelte Form des Ohres berücksichtigt, welche bekannterweise dazu dient, den Klangereignissen aus den unterschiedlichen Richtungen verschiedene Charakteristiken aufzuprägen. Die müßte man synthetisch nachbilden können. Ich glaube, bis heute ist das noch nicht so richtig gelungen.

Was ist, wenn mehrere Personen dem Klangerlebnis beiwohnen sollen? Hierfür bietet sich dann Quadrophonie an, also das Auftrennen eines Klanges in vier Kanäle. Noch mehr Kanäle können die Qualität sicher nur noch steigern. Mit vielen unabhängigen Lautsprechern kann man sich die theoretischen Handstände um die richtige Geräuschbearbeitung fast sparen, denn wenn durch die Vielzahl der Lautsprecher der Klang tatsächlich aus der Richtung kommt, aus der er kommen soll, läßt sich davon jedes Ohr betrügen, egal wo es sich befindet (im Gegensatz zum berühmten Stereodreieck).

Einstellungen

Standard-Kanalnamen

Wenn gesetzt, bekommen alle Kanalklänge abhängig von ihrer Position innerhalb der Stereogruppe vordefinierte Namen, die da sind:

left

right

left back

right back

1.94 Algorhythmus

Algorithmus

aus der Klasse Klanggruppe

Funktion

Ein Algorithmus beinhaltet Prozeße, die komplex verschaltet sein können. Besonders interessant ist hier das Anlegen von benutzerdefinierten Variablen, da Sie damit eine Schnittstelle schaffen können, über die Sie bequem alle wichtigen Parameter des Algorithmus steuern können.

Das Zusammenspiel der Prozesse ist nicht auf Algorithmengruppen angewiesen, das funktioniert alles auch mit Listengruppen oder auch völlig ohne. Die Vorteile eines Algorithmus sind allerdings die bequeme Eingabe der Algorithmen als Flußdiagramme. Außerdem dürfen Sie sicher sein, daß zukünftige Assamplerversionen noch genauer zwischen den einzelnen Gruppenarten unterscheiden werden.

Werte

Output (Klang)

Legt fest, welcher Prozeß das Ausgabesignal für diesen Algorithmus liefert. Dieser Prozeß sollte direkt im Algorithmus enthalten sein (nicht etwa in einem Algorithmus, der in diesem enthalten ist). Sein Ausgang wird in der Regel unbelegt sein und bei übersichtlicher Anordnung der Prozesse ganz rechts stehen.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Anwendung

Algorithmen mit Benutzerschnittstelle

Sie kreieren zunächst einen Algorithmus nebst Anzeige (Doppelklick in den Klassenbaum). In diesem Algorithmus legen Sie die benötigten Prozesse an (Drag&Drop vom Klassenbaum in die Algorithmus-Anzeige) und verbinden sie wie gewünscht. Dann öffnen Sie die Informationsfenster der Prozesse (Doppelklick auf Kästchen im Flußdiagramm), um allen Variablen geeignete Werte zu geben.

Bislang müssen wir noch jeden Prozess einzeln anzeigen, um die Eigenschaften des Algorithmus zu beeinflussen. Wir wollen aber erreichen, daß alle wichtigen Werte (Zahlen und Klangangaben) des Algorithmus in dessen Informationsfenster einsehbar und veränderbar sind.

Dazu öffnen Sie das Informationsfenster des Algorithmus, stellen für die Output-Variable den letzten Klang in ihrer Prozeßkette ein. Öffnen Sie nun das Variablenfenster für den Algorithmus. Überlegen Sie sich, welche Variablen der Prozesse Sie vom Informationsfenster des Algorithmus aus steuern wollen. Ziehen Sie die Variablennamen der ausgewählten Variablen aus den Prozessfenstern in das Variablenfenster des Algorithmus. Wählen Sie eine sinnvolle Reihenfolge, und entscheiden Sie in welcher Spalte jede Variable dargestellt werden soll. Schließen Sie das Variablenfenster. Alle Variablen sind nun im Informationsfenster zu sehen.

Nun wollen wir es so einrichten, daß die Prozesse die Werte von der Algorithmus Benutzer-Schnittstelle übernehmen. Für jede benutzerdefinierte Variable des Algorithmus setzen Sie den Referenzschalter (befindet sich zwischen Variablenname und Variableninhalt) der zugehörigen Variablen eines Prozesses und ziehen den Variablennamen aus dem Algorithmus-Fenster in das sichtbar gewordene Stringgadget. Dort wird dann der Pfad eingetragen, der den Prozess-Wert vom Algorithmus-Wert abhängig macht.

Anzeige

Die Anzeige besteht aus einer virtuellen Gruppe (ein spezielles MUI-Objekt). Sie enthält Kästchen (ebenfalls MUI-Objekte), welche die Unterklänge des Algorithmus repräsentieren. Jeder Kasten ist mit dem Namen des Klanges beschriftet und ist an den Rändern mit Buchstaben besetzt, welche die Anschlüsse darstellen.

Ein Doppelklick auf eine Box öffnet ein geeignetes Fenster des Klanges.

Auf der rechten Seite sind immer O's (wie Out) zu finden, welche Ausgänge kennzeichnen. Assampler sorgt dafür, daß immer ein nicht verbundenes O dabei ist. Die Anzahl der Ausgänge wächst also automatisch mit den Ansprüchen des Benutzers. Die anderen Seiten zeigen Eingangsanschlüsse. Folgende Bedeutungen werden dabei von den fest verdrahteten Assampler-Prozessen gebraucht und sind ebenfalls für Algorithmen empfohlen, die Sie selbst entwickeln:

Links: Eingangssignale, also die Signale, die dem Ausgangssignal noch am meisten ähneln

Oben: Steuersignale, also Signale die die Parameter des Prozesses über die Zeit hinweg ändern

Eingangs- und Steuersignale werden meistens die gleiche x-Einheit wie das Ausgangssignal haben

Unten: sonstige, darunter fallen z.B. Wellenformen, Funktionstabellen, Filterfenster

Sie verbinden zwei Anschlüsse, in dem sie auf einen unbelegten Anschluss klicken und die Maustaste halten, dann den Mauszeiger über den Zielanschluß schieben und dort loslassen. Falls dort schon eine Verbindung bestand, wird sie gekappt. Beginnt man bei einem belegten Anschluß wird die Leitung entsprechend umgelegt. Verbindungen werden aufgelöst, wenn man den Eingang, in den die Leitung führt aufgreift und irgendwo zwischen den Kästen fallen läßt.

Es ist klar, daß Ausgänge immer nur mit Eingängen verbunden sein können. Bei Analogtechnik kann man da echt etwas kaputt machen. Weil es die Verwaltung in Assampler aber gar nicht anders erlaubt, haben Sie nichts zu befürchten.

Es ist nicht vorgesehen, Leiterzüge zusammenzuführen oder verzweigen. (In der Elektrotechnik werden solche Kreuzungen mit Punkten markiert.) Haben Sie vor, Leitungen zusammenzuführen, dann wollen Sie vermutlich deren Signale mischen. In diesem Fall benutzen Sie bitte tatsächlich einen Mischer. Beabsichtigen Sie, Leitungen zu verzweigen, nutzen Sie stattdessen die Möglichkeit, von einem Prozeß mehrere Ausgänge anzuzapfen.

Tastatur

Algorithmen nur über die Tastatur zu erstellen, ist so ohne weiteres nicht möglich. Verknüpfungen kann man nur über den Umweg Prozeß-Fenster öffnen + Klang-Variable einstellen bearbeiten.

Drag&Drop

Erzeugen von Klängen: Ziehen Sie Eintrag aus Klassenbaum in Algorithmus-Anzeige.

Verschieben zwischen Algorithmen und Gruppen und Anordnen von Klängen: Ziehen Sie Kasten innerhalb Algorithmus-Anzeige herum oder in andere Algorithmus- oder Gruppen-Anzeige. Beachten Sie, daß in diesem Fall Drag&Drop nur mit Umschalttaste möglich ist (z.B. Control, je nachdem was in den MUI-Einstellungen eingestellt ist).

Ersetzen von Prozessen: Ziehen Sie Kasten oder Gruppen-Eintrag auf anderen Kasten. Besitzen ersetzter und ersetzender Klang gleichnamige Variablen, werden übereinstimmende Variablen vom ersetzten Klang in den ersetzenden übernommen. Das sichert z.B. daß alle Verbindungen erhalten bleiben. Problem: Die Output-Variable von Algorithmen wird auch ersetzt, obwohl das wenig sinnvoll ist. Problem wird später beseitigt, versprochen!

Einstellungen

Reserviere Rand links/rechts

Algorithmus-Anzeige achtet darauf, daß immer mindestens soviele Buchstabeneinheiten links und rechts neben den äußersten Kästen (genaugenommen deren Mittelpunkten) Platz bleiben. Damit ist garantiert, daß der zur Verfügung gestellte Platz mit der Größe des Flußdiagrammes wächst.

Reserviere Rand oben/unten

das gleiche für den Abstand nach oben und nach unten

Farbauswahl für

Beschriftung der Kästen

Leiterbahnen

Gummiband während Leitungen gezogen werden

Anschluß offen

Anschluß belegt

Hintergrundmuster für Kästen

Hintergrundmuster für Flußdiagramm

1.95 Hier wird ihnen ein Prozeß gemacht

Prozeß

aus der Klasse Klang

Unterklassen

Hilfsmittel

Erzeuger

Bearbeiter

Analyse

Funktion

Prozesse generieren neue Klänge oder verändern vorhandene. Ihr Signal liegt nicht statisch vor, sondern muß erst auf Anfrage berechnet werden.

Anwendung

Obwohl für alle Soundklassen zutreffend, möchte ich hier noch einmal auf folgende praktische Einrichtungen hinweisen:

Durch Doppelklick in den Klassenbaum kann man einen Prozess erzeugen. Von diesem wird dann das Informationsfenster geöffnet.

Das Informationsfenster ist im Prinzip genauso zu bedienen wie die Requester vergleichbarer Soundeditoren. Man trägt ein paar Werte ein und startet dann mit dem Start-Knopf die Berechnung, welche auf dem Sound im aktuellen Anzeigefenster ausgeführt wird. Unterschied: Das Fenster wird nach Start der Berechnung nicht geschlossen, man kann eine Berechnung mehrmals starten - mit unterschiedlichen Werten und auch dann, wenn die aktuelle Berechnung noch läuft! Außerdem wird der Prozeß beim Schließen des Fensters nicht gelöscht.

Anzeige

Damit im Anzeigefenster von Prozessen überhaupt etwas sinnvolles erscheint, prangt derzeit nur ein großer Wiedergabe-Knopf vom Bildschirm, welcher beim Betätigen genau das tut, was man vom ihm erwartet: Er macht das Prozeßsignal hörbar.

Tips

Wie erhält man klangliches Grundmaterial

Oszillator, FM Operator,

Rauschen, Knattern

Wie erzeugt man Steuerkurven (z.B. Hüllkurven)

Splines, konstante Funktion, lineare Funktion,

gerade oder ungerade gebrochenrationale Funktionen, Exponentialfunktion, Tieffrequenz-Oszillator (LFO)

Wie prägt man Klängen Hüllkurven (inklusiv Effekte) auf

Verstärker, Filter aller Art,

Begrenzer, Schrumpfer, Umklappen

Wie gewinnt man Hüllkurven von fertigen Klängen zurück

Klang zuerst gleichrichten, dann glätten mit

rekursivem oder nichtrekursivem Tiefpassfilter, Konvertierung in Spline, gleitendes Maximum

Wie befreit man Klänge von Störgeräuschen

Hochfrequentes Rauschen mit Tiefpass herausfiltern.

Konstante Nullpunktverschiebung durch entsprechende Anhebung beseitigen.

Variable Nullpunktverschiebung durch einen Hochpass mit niedriger Grenzfrequenz (etwa 10 Hz) eliminieren.

Rauschen in Pausen durch Dämpfen der Pause. Dazu Hüllkurve abnehmen (s.o.), triggern und das gewonnene Signal wieder als Hüllkurve aufprägen.

Geräusch von Pause an Anfang und Ende befreien: Bereichsmarkierung so wählen (oder besser: lassen), daß nur eigentliches Geräusch darin enthalten ist, dann Freistellen.

Pfeifen entfernen durch Bandsperre.

Brummen entfernen: Brummen zunächst durch Echo verstärken (Delay gleich der Periode = 1/Frequenz des Brummens setzen), das erzeugte Signal vom Original subtrahieren (Mixer mit Volume 1 für Original und -1 für Brummen).

Einstellungen

Anlegen in

In dieser Gruppe werden Prozesse abgelegt, die durch Doppelklick im Klassenfenster erzeugt wurden.

1.96 hier werden sie geholfen

Hilfsmittel

aus der Klasse Prozeß

Unterklassen

Umleitungsprozesse

Schnitt

Funktion

Hilfsprozesse lösen Probleme, die es ohne Assampler nie gegeben hätte. Sie erzeugen selbst keine Klangeffekte.

1.97 Der chinesische Verkehrsminister: Um Lei Tung

Umleitungsprozesse

aus der Klasse Hilfsmittel

Unterklassen

Auswahl

Weiterleiten

Puffer

Ablage

1.98 Das Brot sollst Du brechen (würg) und unter den Armen verteilen (schmier, schmier)

Auswahl

aus der Klasse Umleitungsprozesse

Funktion

Erlaubt die Anwahl eines Eingangs, der durchgeschleift werden soll.

Werte

Array (Feld)

Array.Input (Klang)

mögliche Eingangssignale

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Choice (Formel)

Nummer des ausgewählten Eingangs

1.99 Weiterleiten

Weiterleiten

aus der Klasse Umleitungsprozesse

Funktion

Schleift das Eingangssignal unverändert durch.

Werte

Input (Klang)

Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Tips

Möchten Sie die Klänge einer Gruppe in einer anderen Reihenfolge oder nur einen Teil der Klänge auslesen (d.h. für eine andere Berechnung verwenden), können Sie sich eine Gruppe von Weiterleitungsobjekten anlegen und mit Hilfe dieser Objekte die Positionen innerhalb der Gruppe den Klängen zuordnen.

1.100 Hm lecker, Kartoffelpuffer

Puffer

aus der Klasse Umleitungsprozesse

Funktion

Speichert Signal über bestimmten Zeitraum. Wann immer ein Ausgangssignal auf mehrere Nachfolgeprozesse aufgefächert werden soll, sollte man einen Puffer dazwischen schalten, welcher sicherstellt, daß das Signal nicht für jeden der Nachfolgeprozesse neu berechnet werden muß.

Irgendwelche Parametrisierungen des Eingangsklanges, die bei jedem Auslesen des Puffer-Prozesses verschieden sind, werden dementsprechend nicht beachtet, sollten daher von vorneherein vermieden werden.

Dieser Prozeß dient allein der Minimierung von Rechenzeit und Speicherbedarf.

Werte

Buffered (Formel) [Input.x]

Der Puffer hebt sich nur begrenzt Daten auf. Dieser Wert gibt an, über welche Zeit zurückliegende Daten gespeichert werden sollen.

Reicht dieser Umfang nicht aus, gibt es eine Fehlermeldung. Das Problem: Eigentlich können Sie gar nicht wissen, wie groß Sie die Zahl wählen müssen. Selbst wenn sie zu klein gewählt wird, kann es funktionieren. Und in der nächsten Assampler-Version tut es das vielleicht nicht mehr. Das ist also keine befriedigende Lösung, aber ich kenne einfach noch keine bessere.

Das sei Ihnen als Faustregel mitgegeben: Geht das Signal aus dem Puffer nur an Prozesse, die nicht in das Zeitverhalten eingreifen, reicht ein kleiner Sicherheitswert von vielleicht 0.1s. Sind Phasenmodulationen nachgeschaltet (auch weiter entfernt) nehmen Sie einen Wert, der die maximale Verzögerung und maximale Vorschau dieser Prozesse berücksichtigt.

Input (Klang)

Zu puffernder Klang

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn fixiert

Technik

Mit dem Puffer-Prozeß habe ich mir einen Haufen Schwierigkeiten an Land gezogen. Eines davon ist, daß der Puffer immer puffert. Manchmal setzt man aber intuitiv voraus, daß er es nur manchmal tut. Zum Beispiel habe ich einen Puffer-Prozeß an den Eingang des Phasing-Algorithmus gesetzt. Zur Erinnerung: Der Phasing-Effekt mischt das Originalsignal mit einem verzögerten Signal. Um zu verhindern, daß das Eingangssignal für beide Strecken neu berechnet werden muß (das Eingangssignal kann ja von einem beliebig komplexen Algorithmus stammen), habe ich den Puffer dahin gesetzt. Man stelle sich aber vor, daß ein Mischer den Ausgang des Phasers mehrmals anfordert, jeweils mit anderem Phaser-Eingang als Parameter - kommen Sie noch mit? - dann puffert der Puffer nur einen dieser Phaser-Eingänge.

Eine Lösung des Problems steht noch aus.

1.101 Ganz schön sedimental

Ablage

aus der Klasse Umleitungsprozesse

Funktion

Speichert das einkommende Signal zu Beginn im Puffer-Klang und läßt alle nachgeschalteten Prozesse aus diesem lesen. Wo die Puffer-Klasse aufgrund ihres beschränkten Speicherumfangs versagt, kann manchmal die Ablage weiterhelfen.

Werte

Buffer (Klang)

Klang, in den Zwischenergebnis geschrieben wird. Sinnvollerweise ein datenbasierter Klang und wahrscheinlich fast immer ein Sample-Sound.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Input (Klang)

zu puffernder Klang

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Technik

Um genau zu sein, trennt dieser Prozeß die Berechnungskette auf und zerlegt sie in zwei einzelne Berechnungen. Die Berechnung, die den Puffer-Klang auffüllt wird einmal vor allen anderen Berechnungen ausgeführt. Parametrisierungen durch den Klang, der den Ablageklang aufruft, sind daher wirkungslos.

Selbstverständlich kann man die Ablage mehrmals innerhalb einer Berechnung verwenden. Diese Berechnung wird durch die Ablagen in mehrere Etappen unterteilt. Ein Klang darf mehrmals zum Puffern verwendet werden. Insbesondere darf auch der Zielklang der gesamten Berechnung benutzt werden. Dann müssen Sie aber den Überblick behalten, um zu verhindern, daß etwa ein Prozeß in einer Etappe einen Puffer-Klang verwendet, darin aber das Ergebnis einer anderen als der vorhergehenden Etappe erwartet.

Tips

Mit der Ablage ist es möglich Zwischenergebnisse aus einer Berechnung abzuzweigen. Beispiel: Sie wollen zwei Klänge mischen und dann einen Echo-Effekt darüber legen. Wenn Sie einen Algorithmus basteln, der das erledigt, bekommen Sie nur das Endergbnis: Den gemischten Klang mit Echoeffekt.

Wenn Sie sowohl den gemischten Klang als auch das Gemisch mit Echo-Effekt haben wollen, können Sie entweder von Hand Mixer und Echo auf zwei verschiedene Sample-Sounds als Ziel anwenden oder Sie bauen in Ihrem Algorithmus zwischen Mixer und Echo das Ablage-Objekt ein. Mit letzter Variante können Sie alles in einem Rutsch erledigen, was sich auf jeden Fall lohnt, wenn Sie noch ein paar Parameter ausprobieren wollen.

1.102 Chirurgen sind doch alle Aufschneider

Schnitt

aus der Klasse Hilfsmittel

Unterklassen

Ausschnitt

Aneinanderreihen

1.103 Das muß einmal clip und klar gesagt werden

Ausschnitt

aus der Klasse Schnitt

Funktion

Blendet Signal um einen bestimmten Zeitbereich herum aus, eine Zeitverschiebung findet nicht statt.

Werte

Begin (Formel) [Input.x]

Legt den Zeitpunkt fest, ab dem das Eingangssignal durchgeführt werden soll. Davor wird mit Nullen aufgefüllt.

Input (Klang)

Durchgeschleifter Klang

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn fixiert

Length (Formel) [Input.x]

Legt fest, wie lange das Eingangssignal nach Begin durchgeschleift werden soll. Danach wird wiederum mit Nullen aufgefüllt.

Tips

Durch Nachschalten dieses Prozesses kann jeder Klang als Wellenform für den Oszillator oder als Kennlinie für den Abbildungsprozess eingesetzt werden.

1.104 Camping-Anhänger von Camping-Anhängern

Aneinanderreihen

aus der Klasse Schnitt

Funktion

Reiht Ausschnitte aus verschiedenen Eingangssignalen aneinander.

Werte

Array (Feld)

Array.Begin (Formel) [Input.x]

Stelle, ab der aus dem Eingangsklang gelesen werden soll.

Array.Input (Klang)

durchgeschleifter Klang

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Array.Length (Formel) [Input.x]

Wie lange soll aus dem Eingangsklang gelesen werden.

Technik

Bei entkoppeltem Ausgang hängen das Lautstärke- und das Samplefrequenz-Attribut vom ersten Sound der Liste ab, der gelesen wird. Dessen Lautstärke sollte also nicht zu klein (insbesondere nicht null) sein. Kann das nicht erfüllt werden, müssen mit nachgeschaltetem Begrenzer bzw. Quantisierer die gewünschten Attribute erzwungen werden. Der Beginn des Ausgangssignals ist immer null.

Tips

Benutzt man nur einen Eingangsklang, entspricht dieser Prozeß dem Ausschnitt-Prozeß bis auf den kleinen Unterschied, daß beim Ausschnittsprozeß das Ausgangssignal bei Begin beginnt, während der Ausgang beim Aneinanderhängen immer bei 0 beginnt.

Mal abgesehen davon, daß er nicht sehr intuitiv zu bedienen ist, kann man mit diesem Prozeß bereits virtuellen Schnitt betreiben.

1.105 Der Produzent

Erzeuger

aus der Klasse Prozeß

Unterklassen

Oszillator

Rauschen

Steuerkurven

Funktion

Erzeuger-Prozesse berechnen ohne Vorlagen Signale.

1.106 Oszillator/Tongenerator

Oszillator

aus der Klasse Erzeuger

Unterklassen

einfacher Oszillator

FM-Operator

Funktion

Ein Oszillator erzeugt Schwingungen, im Gegensatz zum Oszilloskop, das welche sichtbar macht und zum Oszillograph, der welche aufzeichnet. Akustisch sind Schwingungen Töne, und genau zur Tonerzeugung setzen wir den Oszillator hier ein. Solche schlichten periodischen Schwingungen sind fast immer Grundbaustein eines Melodieinstruments.

Der Vorteil der digitalen Synthese allgemein und des Assamplers im Besonderen liegt darin, daß man die Wellenform beliebig wählen kann. Im Assampler kann dafür jeder bereits vorhandene Sound genutzt werden, sofern ihm zuvor eine Zahlenwert-Variable mit dem Namen Length verpaßt wurde. Length gibt dem Oszillator Auskunft über die Länge des Signals, das von dem Sound als Wellenform genutzt werden soll. (Signale sind ja prinzipiell unbeschränkt in der Länge.)

Werte

Frequency (Formel) 1/[x]

Frequenz des Tones in der Regel mit der Einheit Hertz (Hz = 1/s = Schwingungen pro Sekunde).

Neben der menschlichen Hörschwelle ist eine andere Begrenzung nach oben die Abtastfrequenz des Zielklanges. Aufgrund der Rückfaltung kann die Ton-Frequenz maximal halb so groß wie die Abtastfrequenz sein.

Nützlich ist die Tone-Funktion, wenn Sie den Oszillator einen bestimmten Ton der temperierten Tonleiter generieren lassen wollen oder einen Ausdruck wie "1/_.Length" wenn der Oszillator als LFO eingesetzt werden soll.

Phase (Formel) 1°

Bestimmt den Startzustand des Oszillators. Wenn Sie bereits einen Ton mit Phase 0 erzeugt haben, dann sieht der gleiche Ton mit Phase 90° wie um eine Viertelperiode nach links verschoben aus. Empfohlen sind Zahlen im Bereich 0 - 2 Pi rad bzw. 0°-360°, so Sie das Grad-Zeichen nicht vergessen.

Wenn Phase = 0 erhalten Sie als Wellenformen ungerade Funktionen (sofern Sie die voreingestellten Wellenformen, ausgenommen das periodische Rauschen, verwenden). Daraus folgt, daß Ihre Fourierzerlegung ohne Cosinus-Anteile auskommt, da die Cosinus-Funktion bekanntlich eine gerade (nicht etwa eine Gerade!) ist. Sinn und Zweck des Kopfstandes ist, der Sinus-Funktion ähnliche Wellenformen zu haben, die leicht untereinander austauschbar sind und bei der Überlagerung mit verstimmten Tönen das gleiche Schwebungsverhalten aufweisen.

Bei hörbaren Tönen hat die Phase allein hat noch keinen Einfluß auf den Klang. Bei sehr langen Schwingungen ist der Unterschied, ob bei einer Schwingung, deren Periode doppelt so lang wie der Zielklang ist, die Schwingung von 0 über 1 nach 0 kommt (Phase 0) oder von 1 über 0 nach -1 (Phase Pi/2 rad), schon gewaltig.

Die andere Sache ist ein bewußt herbeigeführter Phasenschieber-Effekt. Beginnen beide Oszillatoren mit Phase 0, beginnt das gemischte Signal mit maximaler Verstärkung. Beginnt ein Oszillator mit Phase 0, der andere mit Phase 180°, beginnt das Schwebungsmuster mit der Auslöschung.

Volume (Formel) [y] = [Wave.y] * [Volume]

Ausgabelautstärke des Tongenerators

Wave (Klang)

Hier kann man die Wellenform der Schwingung auswählen. Die Sounds in der Waveforms-Gruppe sind immer die richtige Wahl, es dürfen aber auch Splines sein, wenn Sie auf Freihand-erstellte Wellenformen schwören oder Spektren mit einem Block, wenn Sie die harmonische Synthese bevorzugen. Auch vor Prozessen brauchen Sie nicht halt zu machen, wenn Ihnen der Sinn nach gefilterten Impulsen oder Rauschen steht. Wie Sie Wellenformen von natürlichen Quellen wiederverwenden, erfahren Sie im Tip-Abschnitt weiter unten.

Kopplung Lautstärke gekoppelt, Abtastrate entkoppelt, Beginn entkoppelt

Technik

Die Wellenform wird vor dem Oszillieren komplett in Sample-Daten umgewandelt. Daraus folgt, daß es nicht möglich ist, die Wellenform während eines Tones zu ändern. Die Oszillatoren sind darauf beschränkt statische Töne zu erzeugen! Es sei denn man übertreibt es mit der Frequenzmodulation. Diese Methode macht man sich tatsächlich bei der FM-Synthese zu nutze.

Vermeiden Sie Sounds, die vollständig berechnet viel Speicher beanspruchen würden, denn die Wellenform wird in einem zusammenhängenden Stück Speicher abgelegt. Insbesondere bei Freiformkurven und Prozessen verlieren Sie das schnell aus den Augen. Normalerweise kümmert sich eine Freiformkurve nicht um ihre Abtastfrequenz - sie ist ja nicht quantisiert. Aber hier ist es von Bedeutung: Eine Freiformkurve der Länge 10 s mit Sample-Rate 16726 Hz nimmt bereits 334.52 KB in Beschlag, zum Vergleich eine Standardwellenform dagegen nur 512 Bytes.

Obwohl es ganz im Sinne der Assampler-Philosophie wäre, ist der Oszillator in der derzeitigen Implementation nicht zum Erzeugen von Musik-Schleifen geeignet.

Theorie

Tja, das habe ich nun alles in die Wertebeschreibungen einfließen lassen.

Tips

Für die Erstellung von noch nicht vorhandenen Wellenformen bietet sich folgendes an: Man digitalisiert den Ton eines natürlichen Instrumentes und kopiert eine komplette Schwingung des Tones in die Zwischenablage. Von dort zieht man den Klang am besten in die vorgegebene Wellenform-Gruppe.

Allerdings verändert sich bei den meisten Instrumenten die Wellenform über die Zeit hinweg, dann sollte man nicht eine Wellenform herausgreifen, sondern so eine Art Durchschnitt berechnen. Das erledigt der AverageWave.algo- Beispielalgorithmus. Man muß nur noch die Länge einer Schwingungsperiode messen. Das bedeutet: In der Sample-Anzeige eine Welle markieren, dann das Bereichslänge-Gadget aus dem Anzeigefenster über das WaveLength-Gadget ziehen.

Bei der Vermessung ist äußerste Präzision erforderlich, deshalb sollten Sie die Sample-Anzeige noch über den 1:1-Faktor hinaus vergrößern. Probieren Sie es mit leicht verschiedenen Werten, und verwenden Sie letztendlich den, der das lauteste Ergebnis erzielt (siehe auch Resonanz). Vom Ergebnis der Berechnung verwenden Sie dann eine der letzten Wellen (also ganz rechts) des erzeugten Klanges.

1.107 Oszillator/Tongenerator

einfacher Oszillator

aus der Klasse Oszillator

Funktion

Dieser Oszillator ist die einfache und schnelle Variante des Tongenerators, die sich lediglich in der Frequenz modulieren läßt.

Werte

Depth (Formel)

Die übliche Bedeutung

Interpolation (Auswahl)

Nur wenn Sie einen superschnellen Rechner haben und sich ganz sicher sind, Ihre Kreationen niemals an Benutzer langsamerer Amigas weiterzugeben, sollten Sie hier immer die höchste Interpolation einstellen. Die Resourcen dosiert einsetzen heißt zumindest meine Devise, da unser Gehör eben nicht auf die dritte Nachkommastelle genau hören kann.

Ich empfehle null Interpolation für hohe Töne (Melodieinstrumente), lineare Interpolation für tiefe Töne (Bassinstrumente) und kubische Interpolation für unhörbar tiefe Töne (Steuersignale eines LFO).

Modulation (Klang)

Ein Sound der als Steuerkurve für die Frequenz fungiert. Prinzipiell könnte man auch eine Frequenzmodulation hinter einen unmodulierten Tongenerator schalten, allerdings wäre dann die Qualität nicht so gut und negative Frequenzen wären so gut wie ausgeschlossen.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Round (Schalter)

Beim Arbeiten mit niedrigen Sample-Frequenzen (20kHz abwärts) macht es sich unschön bemerkbar, wenn die Perioden nicht ganzzahlige Anzahlen Sample-Werte überstreichen. Bei kantigen Wellenformen wie Sägezahn oder Rechteck sind im Endeffekt die Perioden einmal so lang, dann mal wieder einen Sample länger, dann mal wieder einen kürzer, gerade so, daß zum Schluß die Statistik stimmt und gebrochene Anzahlen Samples pro Periode aufweist.

Die eingeschaltete Rundungs-Option erzwingt eine Verstimmung des Tones, so daß die Periode genau ein Vielfaches der Sampleperiode ist. Vorsicht ist bei Phasing-Effekten geboten. Wenn durch Rundung zwei leicht verstimmte Frequenzen zusammenfallen, ist er hinüber.

Scale (Formel)

Die übliche Bedeutung

1.108 Ambos, der Operator

FM-Operator

aus der Klasse Oszillator

Funktion

Der FM Operator besteht aus einem Oszillator der durch einen Verstärker in der Lautstärke moduliert wird. Damit hätten wir gerade erst einen Baustein, der sich auch aus zwei anderen zusammensetzen ließe.

Das Besondere ist jetzt aber, das es erlaubt ist, Rückkopplungen einzusetzen, die nur über FM Operatoren hinweg führen. Damit eignen sich die FM Operatoren hervorragend für die FM-Synthese (daher eigentlich auch der Name der Operatoren).

Werte

Array (Feld)

Es sind mehrere Modulationen erlaubt, diese überlagern sich in ihrer Wirkung auf diesen Operator.

Array.Depth (Formel) [Frequency]/[Modulation.y] bei Frequenzmodulation, 1/[Frequency]/[Modulation.y]

Gibt Modulationstiefe für Operator an, größere Modulationstiefe bedeutet stärkere Modulation bis hin zur unbrauchbaren Verzerrung.

Modulationssignale bedeuten:

bei Phasenmodulation:

-1/.Frequency : Oszillator wird um eine Schwingungsdauer verzögert

0 : Oszillator unbeeinflußt

1/.Frequency : Oszillator wird um eine Schwingungsdauer beschleunigt

bei Frequenzmodulation:

-.Frequency : Oszillatorfrequenz ist null (keine Schwingung)

0 : Oszillator unbeeinflußt

.Frequency : Oszillatorfrequenz wird verdoppelt

Array.Modulation (Klang)

Der FM-Operator oder ein anderer Sound, der diesen Operator in Phase oder Frequenz modulieren soll. Bei einem FM-Operator sind Rückkopplungen erlaubt. Beliebt sind zu sich selbst rückgekoppelte Operatoren. Ist kein Sound eingestellt, entfällt diese Modulation.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Envelope (Klang)

Hüllkurve (Lautstärkeverlauf) für diesen Operator oder auch den Modulationsverlauf falls Operator als Modulator auftritt. Envelope wirkt wie das Nachschalten eines modulierten Verstärkers. Ist kein Klang eingestellt, wird keine Hüllkurve aufgeprägt.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

FbDelay (Formel) [x]

Falls Sie Rückkopplungen verwenden, können Sie einstellen, wie lange ein Signal verzögert wird, bis es wieder zurückgekoppelt wird (Feedback delay). Der Wert sollte in genau einem der Operatoren einer Schleife gesetzt sein.

Die Rückkopplungsverzögerung ist wichtig, damit das Signal unter anderen Abtastfrequenzen reproduziert werden kann. Würde man nämlich immer die schnellste Verbindung zurück benutzen, würde die Rückkopplung mit steigender Abtastfrequenz effektiv höher.

Deshalb wird folgende Vorgehensweise empfohlen: Für einen FM-Sound unter einer bestimmten Abtastfrequenz die richtigen Werte austüfteln und am Schluß den reziproken Wert der Abtastfrequenz (also die Abtastperiode) in FbDelay eintragen.

Modulate (Auswahl)

Entscheidet ob Frequenz oder Phase moduliert wird. Diese Extravaganz hat eher geschichtliche Hintergründe und macht die Sache insgesamt nur komplizierter und langsamer.

Volume (Formel) [y] = [Wave.y] * [Envelope.y] * [Volume]

Bestimmt die Lautstärke des Operators und ist die letzte Chance die Einheit des Ausgangssignales zurechtzubiegen.

Wave (Klang)

Hier kann man die Wellenform der Schwingung auswählen. Bei der Verschaltung mehrerer FM-Operatoren sind Sinusschwingung zu empfehlen, da hier keine Klang-Sprünge entstehen, wenn man mit der Modulation herumspielt.

Kann man sich schlecht vorstellen, wenn man's nicht gehört hat, also probieren Sie's mal mit anderen Wellenformen.

Kopplung Lautstärke gekoppelt, Abtastrate entkoppelt, Beginn entkoppelt

Technik

Verwenden Sie möglichst keine Parametrisierung. Wegen der Spezialbehandlung (z.B. für Rückkopplung) von FM-Operatoren wird davon ausgegangen, daß derselbe FM-Operator auch immer die gleichen Eigenschaften besitzt.

1.109 Rauschen

Rauschen

aus der Klasse Erzeuger

Unterklassen

Weißes Rauschen

Impulse

Theorie

Weißes und rosa Rauschen

Das sind die zwei Arten des Rauschens, die am meisten verwendet werden und daher auch spezielle Namen bekommen haben. Die Bezeichnungen wurden in Anlehnung an das Spektrum von weißem bzw. rosa Licht gewählt. Weiß sieht Licht aus, in dem alle Frequenzanteile gleich aber zufällig verteilt sind. Während Rosa nur die tieferen Frequenzen des sichtbaren Lichts enthält. Weißes Rauschen klingt wie das Rauschen aus dem Radio auf unbelegten Kanälen. Während das rosa Rauschen ein ganz dumpfes Rauschen ist, welches auch mehr für Steuerzwecke gedacht ist, z.B. als Frequenzverlauf für einen Ton.

Rauschen und Chaos

Wenn man's nicht besser weiß, kann man schnell dem Irrtum unterliegen, daß Rauschen chaotisch und nicht näher beschreibbar sei. Wahr ist aber, daß Rauschen sehr wohl strukturiert ist, z.B. durch die Häufigkeit bestimmter Werte. Das Spektrum von weißem Rauschen ist wiederum weißes Rauschen, sollte also - als Sample-Sound abgespielt - mindestens ebenso gleichmäßig rauschen.

Um von der Struktur eine Vorstellung bekommen, machen Sie folgendes Experiment: Erzeugen Sie weißes Rauschen von einer Sekunde Länge und definieren eine Schleife eben dieser Länge. Achten Sie beim Abspielen darauf, ob sich der Sound wiederholt. Sie werden merken, wie er an manchen Stellen vielleicht leiser oder lauter wird oder wie bestimmte Frequenzen herausgehoben werden und wieder im Einheitsbrei verschwinden. Nichts spektakuläres zwar, aber man hört die Wiederholungen. Ab einer bestimmten Länge verläuft sich dieser Effekt natürlich, weil Sie sich nach einer Minute nicht mehr an den Anfang erinnern würden.

Umgedreht kann man den Effekt verstärken, in dem man die Periode so kurz wählt, daß wir in den Bereich hörbarer Frequenzen kommen. Dann nämlich entsteht ein klarer konstanter Ton. Verwenden Sie Rauschen als Wellenform für einen Tongenerator, klingt der Ton jedesmal anders, weil das Rauschen jedesmal neu und zufällig erzeugt wird - mal dominieren die Oberwellen, mal kommt die Grundfrequenz mehr zum Tragen, aber es ist immer ein konstanter Ton. Die Regelmäßigkeit kann man in diesem Fall auch sehen. Nur das Oberwellenspektrum trägt den Charakter von Rauschen. Der Sound dieser Töne ist meistens ein sehr spitzer, weil in natürlichen Klängen die Oberwellen mit steigendem Faktor immer schwächer auftreten, die Rauschwelle dagegen einige Oberwellen haben kann, die lauter sind als die Grundwelle. Weil der Sound so typisch ist, finden Sie ihn auch unter den voreingestellten Wellenformen.

Rauschen kann man auch durch Filter seiner Homogenität berauben. Es ist schon ein bißchen schwer zu glauben, daß nach ein paar Additionen und Multiplikationen aus dem Rauschen etwas anderes als genau das gleiche Rauschen herauskommen soll, aber man kann Rauschen genau wie allen anderen Sounds die typischen Filtereffekte verpassen: Frequenzen herausscheiden, abschwächen oder anheben.

Filtert man Rauschen mit einem Bandpass kann man typische Windgeräusche imitieren. Mit dem Phaser kann man aus Rauschen wunderbare Flugzeuggeräusche produzieren.

Rauschen und Reproduzierbarkeit

Zufallsgeneratoren im Computer basieren meistens auf Zahlenfolgen, die vom Angucken und Anhören her völlig ungeordnet und willkürlich erscheinen, in Wirklichkeit aber exakt berechnet und reproduzierbar sind. Diese Folgen müssen trotzdem ein paar Bedingungen erfüllen, um als Allzweck- Zufallsgeneratoren durchgehen zu können. Die berechneten Zahlen müssen ein vorgegebenes Intervall gleichmäßig abdecken, dürfen also nicht gehäuft um bestimmte Zahlen herum auftreten und die Folge sollte nicht in einen allzu kurzen Zyklus eintreten, so daß man z.B. nur noch eine Folge aus 10 Zahlen erhält, die sich immer wiederholt. Eine Zahlenfolge, die diese Kriterien erfüllt und deshalb fast immer als Zufallsfolge herhalten muß, ist die Folge a(n) = pˆn mod q, wobei mod den Rest bei der Division pˆn durch q berechnet. (In Computerkreisen durchaus übliche Notation, aber mathematisch ist das "mod" schon für Zahlenkongruenzen vergeben.)

Im Amiga kann man unter Ausnutzung unbelegter Ports wie für Analogjoysticks echte Zufallsfolgen erzeugen. Aber wir brauchen nicht unbedingt echte Zufallsfolgen. Es reicht völlig, wenn das resultierende Signal rein subjektiv wie Rauschen klingt.

1.110 Assampler-Vollrausch

Weißes Rauschen

aus der Klasse Rauschen

Funktion

Erzeugt weißes gleichverteiltes Rauschen.

Werte

Frequency (Formel)

Da der Computer nicht wirklich weißes Rauschen erzeugen kann, weil seine Ausgabe auf einen endlichen Frequenzumfang beschränkt ist, kann nur quantisiertes weißes Rauschen erzeugt werden. Damit diese Quantisierung unabhängig von der Ziel-Abtastrate ist, muß man hier eine Abtastfrequenz angeben. Diese sollte immer kleiner als die Ziel-Abtastrate sein.

Davon daß die Quantisierung mehr als theoretische Spielerei ist, kann man sich leicht überzeugen, indem man weißes Rauschen mit vorgegebener 22kHz Quantisierung einmal in einen 8363 Hz- und dann in einen 44100 Hz-Sample-Sound ausrechnet. Es klingt einfach völlig anders. Selbst wenn man jetzt mit einem Tiefpassfilter das Rauschen von hohen Frequenzen befreit, so daß beide Sample-Sounds etwa gleichklingen, bleibt immer noch ein gravierender Lautstärkeunterschied.

Benutzen Sie daher keine Werte wie "_.SampleFreq" und benutzen sie niemals höhere Werte, als Sie auf ihrem Amiga als Abtastrate ausprobiert haben.

Volume (Formel)

Aussteuerung des Rauschens.

Tips

Um Rosa-Rauschen zu erhalten schalten sie einen Tiefpass oder Bandpass mit tiefer Filterfrequenz nach.

1.111 Geigelzähler

Impulse

aus der Klasse Rauschen

Funktion

Erzeugt zufällig zeitlich verteilte Impulse, die Impulse selbst sind immer gleichgeformt. Der knatternde oder tickende Klang erinnert an Geigerzähler (die korrekt Geiger-Müller-Zählrohre heißen).

Werte

Density (Formel) 1/[x]

Wieviel Impulse pro Zeiteinheit sollen durchschnittlich (allerdings nicht im arithmetischen, sondern im harmonischen Mittel :-) erzeugt werden.

Height (Formel) [y]

Die Höhe jedes einzelnen Impulses.

Stray (Formel) 1

Maß wie ungleichmäßig die Impulse im Rahmen der durchschnittlichen Dichte verteilt sein sollen. Wert 0 bedeutet, daß die Impulse in gleichen Abständen erfolgen. Maximalwert 1 bedeutet, daß die Impulse mal schneller, mal langsamer aufeinander folgen und eben auch mal direkt nacheinander auftreten.

Width (Formel) [x]

Die Länge jedes einzelnen Impulses.

Theorie

Ein infinitesimal schmaler Impuls enthält alle möglichen Frequenzen. Die Impulse im Computer haben immer eine endliche Breite, und enthalten nur noch die Frequenzen, die ganze Teiler der reziproken Impulsbreite sind.

Tips

Durch nachgeschaltete Filter, Bandpässe bevorzugt, läßt sich der Klang der Impulse nachgeträglich ausprägen.

1.112 Herausgezogenes Lenkrad = Steuererhöhung

Steuerkurven

aus der Klasse Erzeuger

Unterklassen

lineare Kurve

Exponentialkurve

gerade Funktion

ungerade Funktion

1.113 Zieh Leine!

lineare Kurve

aus der Klasse Steuerkurven

Funktion

Erzeugt eine lineare Kurve.

Werte

Begin (Formel) [x]

An der durch Begin gekennzeichneten Stelle nimmt die lineare Kurve den Wert BeginY an.

BeginY (Formel) [y]

Wert der bei Begin durchlaufen wird.

EndY (Formel) [y]

Wert bei Begin+Length. Der größere Betrag von BeginY und EndY wird als maximale Auslenkung verstanden, wenn diesem Prozeß ein Lautstärke-entkoppelnder Prozeß nachgeschaltet ist. Im Prinzip ist das falsch, da alle linearen Funktionen außer den konstanten unbeschränkt sind, aber es ist irgendwie am sinnvollsten - und wahrscheinlich wollten Sie es so genau gar nicht wissen :-)

Length (Formel) [x]

Nach dieser Distanz gemessen ab Begin nimmt die lineare Kurve den Wert EndY an.

1.114 Exponentialkurve

Exponentialkurve

aus der Klasse Steuerkurven

Funktion

Erzeugt eine exponentielle Kurve oder mit anderen Worten eine Funktion der Form eˆ(t*x).

Werte

Begin (Formel) [x]

Die Stelle, an der der Wert BeginY angenommen wird.

BeginY (Formel) [y]

Der Wert, der an der Stelle Begin durchlaufen wird. Wird zugleich als maximale Auslenkung verstanden, wenn Lautstärke-entkoppelnde Prozesse nachgeschaltet sind. Hier wie auch bei der linearen Kurve ist diese Festsetzung abgesehen vom Fall BeginY=0 unkorrekt aber sinnvoll.

Decay (Formel) [x]

Die Zeit, in der die Kurve auf den Anteil 1/e also etwa auf ein Drittel abfällt.

Ich habe lange überlegt, ob nicht die Halbwertszeit der intuitivere Wert wäre. Aber da es in der jetzigen Variante möglich ist, direkt die R-C-Konstante aus einer elektronischen Schaltung zu übernehmen, habe ich ihr den Vorrang gegeben.

1.115 gerade gebrochenrationale Funktion

gerade Funktion

aus der Klasse Steuerkurven

Funktion

Erzeugt symmetrische Funktion mit einer glockenartigen Form. Die Funktionswerte sind also zunächst nahe bei null, erreichen dann das Maximum, und fallen wieder ab. Die verwendete gebrochen-rationale Funktion ist bis auf Parameter 1/(xˆ2+1).

Werte

MaximumX (Formel) [x]

Stelle, an der das Maximum angenommen wird

MaximumY (Formel) [y]

Maximalwert

Width (Formel) [x]

Abstand zwischen Maximalstelle und der Stelle, an der der halbe Maximalwert angenommen werden soll

1.116 ungerade gebrochenrationale Funktion

ungerade Funktion

aus der Klasse Steuerkurven

Funktion

Erzeugt ungerade symmetrische Funktion mit einem schnell ansteigenden und langsam abfallenden Verlauf. Die verwendete gebrochen-rationale Funktion ist bis auf Parameter x/(xˆ2+1).

Werte

Begin (Formel) [x]

Stelle des Nulldurchgangs

MaximumX (Formel) [x]

Stelle, an der das Maximum angenommen wird

MaximumY (Formel) [y]

Maximalwert

1.117 Der Sachbearbeiter

Bearbeiter

aus der Klasse Prozeß

Unterklassen

X-Modulation

Y Werte

Filter

Spektrum

Funktion

Bearbeiter-Prozesse berechnen anhand Vorlagen Signale. Die Klänge sollten nach der Bearbeitung noch eine gewisse Ähnlichkeit (dazu zählt auch die Länge) zum Original haben.

Werte

Einige Prozesse können im Verlaufe der Zeit ihre Eigenschaften ändern. Man spricht dann von Modulation. Die Modulation gibt die Abweichung von einem Grundwert (z.B. Frequenz beim Oszillator) an.

Es werden von den Prozessen Werte für die Variablen Modulation, Depth und Scale (nicht immer alle) verlangt. Deren Bedeutung ist im Prinzip immer gleich.

Depth (Formel)

Modulationstiefe für die Modulation. Um nicht immer einen Verstärker dazwischen schalten zu müssen, bieten alle Prozesse die Eingabe der Modulationstiefe an. Im Prinzip wird einfach jeder Wert der Modulation mit der Modulationstiefe multipliziert, das Ergebnis wird dann der eigentlichen Modulation zugeführt.

Es bietet sich an, Steuerkurven grundsätzlich mit einer Maximal-Lautstärke von 1 auszurüsten. Es kommt nämlich häufig vor, daß eine Steuerkurve mehrere Prozesse steuern soll. Wirkliche Modulationstiefe nebst Einheit werden dann mit Depth nachgereicht. Vorteil: Sie haben mit dem Informationsfenster eines Prozesses alles im Blick, und müssen nicht noch darauf achten, was andere Prozesse für Ausgangslautstärken besitzen. In den Steuerkurven-Prozessen ist übrigens die Lautstärke 1 voreingestellt.

Wegen der hochoptimierten Festkomma-Assembler-Routinen (Schulter-Klopf) ist nicht jede Modulationstiefe realisierbar. Ich habe zwar versucht den Bereich so nah wie möglich an der Praxis zu wählen (macht man ihn größer, wird immer auch die absolute Genauigkeit geringer, klar), aber ich bin mir sicher, daß der Wert, den Sie jetzt unbedingt dringend brauchen, gerade nicht mehr erreicht wird.

Modulation (Klang)

Der Sound dessen Verlauf die Modulation angibt. Tatsächlich tritt erst eine Modulation ein, wenn eine Modulation und eine andere als die konstante Skalierung und eine Modulationstiefe ungleich 0 gewählt wurde.

Ist kein Sound gewählt, wird nicht moduliert. Dann liegt es aber immer noch im Interpretationsspielraum des Prozesses, ob er das als Nullstellung (wie beim Oszillator) oder als Maximalstellung (wie beim Verstärker) auffaßt. In zukünftigen Assampler-Versionen wird diese Unterscheidung auch im Programm transparenter werden.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Scale (Auswahl)

Die Skalierung legt fest, wie die Gesamtmodulation (also Modulation*Depth) ausgewertet werden soll.

constant: konstante (sprich: gar keine) Modulation

linear: lineare Modulation, die Gesamtmodulation wird einfach zum Grundwert addiert

Beispiel: Oszillator, Frequency=500Hz, Modulation.Volume=1, Depth=200Hz

Damit können Sie die Frequenz im Bereich von 300Hz bis 700Hz linear abgestuft steuern.

exponential: exponentielle Modulation, es wird die Zweierpotenz der Gesamtmodulation berechnet und das Ergebnis mit dem Grundwert multipliziert. Konsequent wäre hier zwar die Basis e gewesen, aber gerade bei Frequenzen ist die 2 doch praktischer. Wollen Sie dennoch e als Basis haben, genügt die Anwendung des wohlbekannten Logarithmengesetzes aˆb = Exp (b Ln(a)), d.h. Sie dividieren Ihren für die Exp-Funktion bestimmten Exponenten durch Ln(2).

Beispiel: Oszillator, Frequency=500Hz, Modulation.Volume=1, Depth=3

Die Frequenz läßt sich damit um 3 Oktaven sowohl nach oben als auch nach unten beugen.

In Zahlen: Untere Grenze 500Hz*2ˆ-3 = 500Hz/8 = 62.5Hz, obere Grenze 500Hz*2ˆ3 = 500Hz*8 = 4000Hz

Technik

In der Reihenfolge wie die Modulationsarten schwerer zu verstehen sind, sind auch die dafür nötigen Algorithmen langsamer (also konstant < linear < exponentiell). Klarer Fall, Computer sind auch nur Menschen.

1.118 X-Modulation

X-Modulation

aus der Klasse Bearbeiter

Unterklassen

Frequenzmodulation

Phasenmodulation

Quanteln in X-Richtung

Umkehren

Funktion

Diese Prozesse wirken auf die zeitliche Reihenfolge der Signalwerte.

1.119 Ein stark frequentierter Effekt

Frequenzmodulation

aus der Klasse X-Modulation

Funktion

Moduliert die Frequenz der Töne im Eingangssignal.

Werte

Depth (Formel)

Die übliche Bedeutung

Input (Klang)

Zu modulierendes Eingangssignal.

Kopplung Lautstärke gekoppelt, Abtastrate entkoppelt, Beginn gekoppelt

Interpolation (Auswahl)

Interpolationsart die auf das Eingangssignal angewandt wird.

Modulation (Klang)

Modulationskurve für die Frequenz, oder nichts, falls nicht moduliert werden soll.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Ratio (Formel) [x] = [Input.x] / [Ratio]

Frequenzverhältnis von Ausgangssignal zu Eingangssignal.

Werte >1 bedeuten Anheben der Frequenzen, Werte <1 bedeuten Absenken der Frequenzen.

Wert 2 bedeutet z.B. Transposition um eine Oktave nach oben.

Scale (Formel)

Die übliche Bedeutung

Technik

Die Frequenzmodulation wird durch zeitliches Strecken und Stauchen des Eingangssignals erreicht, wie man es auch bei leiernden Kassetten erlebt oder bei Spielereien am laufenden Plattenspieler. Das zieht unweigerlich nach sich, daß das Erhöhen der Frequenz den Klang verkürzt, wogegen das Verringern der Frequenz den Klang verlängert. Diesem Phänomen kann man mehr oder weniger gut mit den Beispiel-Algorithmen PitchShift und HunkTranspose begegnen. Diese Algorithmen springen in gewissen Abständen vor oder zurück, sobald das Ausgangssignal zu kurz oder zu lang zu werden droht.

Versuchen Sie nicht, mit Modulationskurven zu arbeiten, die Impulse von genau einem Sample-Wert Länge enthalten. Sie könnten das zum Beispiel wollen, um in bestimmten Abständen im Eingangssignal vor- oder zurückzuspringen. Dafür nehmen Sie lieber den Phasenschieber mit sägezahnförmigen Modulationskurven, denn die Frequenzmodulation tendiert hier zu Ungenauigkeiten (jeder Fehler summiert sich, und das Signal läuft Ihnen mit der Zeit weg) und die besagten Impulse entsprechen nicht dem Ideal, der Reproduzierbarkeit unter verschiedenen Abtastfrequenzen.

Theorie

Die Frequenzmodulation entspricht einer Phasenmodulation, sofern man für die Phasenmodulation die integrierte Modulationskurve der Frequenzmodulation verwendet.

Tips

Man kann die Frequenzmodulation dafür mißbrauchen, einen Klang an einer bestimmten Stelle zu stoppen und später an der gleichen Stelle fortzusetzen. Das entspricht in etwa der Einfügen-Editierfunktion. Sie benötigen hierzu eine Steuerkurve, die in den Phasen, wo der Klang unverändert durchgeschleift werden soll, eins ist und in den Pausen null. Diese Steuerkurve setzen Sie als Modulation, die Modulationstiefe Depth auf 1 und das unmodulierte Frequenzverhältnis Ratio auf 0.

1.120 Jeder Effekt hat mal seine Phase

Phasenmodulation

aus der Klasse X-Modulation

Funktion

Verzögert das Eingangssignal und moduliert damit die Phasenlage der Töne. Das wird für physikalische Simulationen benötigt, für den legendären Phasing-Effektfilter und für einmalige Echos (Delay-Effekt).

Werte

Depth (Formel)

Die übliche Bedeutung

Input (Klang)

Zu modulierendes Eingangssignal.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn mit Distanz

Interpolation (Auswahl)

Interpolationsart die auf das Eingangssignal angewandt wird.

Modulation (Klang) [x] = [Modulation.y] * [Depth]

Modulationskurve für die Phasenverschiebung, oder nichts, falls konstant mit dem Wert Depth verzögert werden soll.

Negative Modulationswerte bedeuten eine echte Verzögerung, positive Werte bewirken ein Vorwegnehmen zukünftiger Schallereignisse.

Beispiel: Die Auslenkung des Zielklanges zum Zeitpunkt 1s entspricht dem Wert des Eingangsignals zum Zeitpunkt 1.1s, wenn die aktuelle Modulation 0.1s beträgt.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Theorie

Die Phasenmodulation entspricht einer Frequenzmodulation, sofern man für die Frequenzmodulation die differenzierte Modulationskurve der Phasenmodulation verwendet.

Es ist nicht das gleiche, wenn man einmal Steuerkurven A und B überlagert und damit die Phasenmodulation steuert und wenn man zum anderen zweimal Phasenmodulation anwendet, zuerst mit Steuerkurve A und dann mit Steuerkurve B. Daraus ergibt sich, daß man einen einmal erzeugten Phasenverschiebungseffekt nicht wieder aufheben kann, in dem man über das gewonnene Ergebnis nochmal die Phasenmodulation mit negierter Steuerkurve laufen läßt.

Damit brauche ich wohl fast nicht mehr zu erwähnen, daß das Vertauschen von zwei hintereinander ausgeführten Phasenmodulationen im allgemeinen zu verschiedenen Resultaten führt.

Wo liegt das Problem?

Sei f(x) unser Eingangssignal als Funktion von x betrachtet und sei g(x) unsere Steuerkurve.

Dann berechnet die Phasenmodulation die Funktion p(x) = f(x+g(x)).

Was passiert, wenn wir auf dieses Ergebnis die Phasenmodulation mit negativer Steuerkurve anwenden? Wir erhalten

p1(x) = p(x-g(x)) = f(x-g(x)+g(x-g(x))

Was wir eigentlich erhalten wollten, war aber f(x-g(x)+g(x)) = f(x).

Das ganze ist doch ein recht kniffliges Problem, obgleich es zunächst nicht sehr anwendungsnah erscheint. Man könnte in vielen Fällen ganz brauchbare Näherungen finden, man kann aber auch Fälle konstruieren, wo die Phasenmodulation wirklich nicht eindeutig umkehrbar ist.

1.121 Käsequanten

Quanteln in X-Richtung

aus der Klasse X-Modulation

Funktion

Quantisiert das Eingangssignal in x-Richtung. Man stelle sich vor, daß das Signal gleichmäßig in Intervalle zerlegt wird, und daß innerhalb eines Intervalles das Signal einen konstanten Wert bekommt, nämlich den der vorher am linken Rand des Intervalles war. In etwa das gleiche passiert bekanntlich beim Digitalisieren von natürlichen Geräuschen.

Werte

Depth (Formel)

Die übliche Bedeutung

ForceFreq (Schalter)

Wenn dieser Schalter gesetzt ist, wird dem Eingangssignal die Quantisierungsfrequenz als Abtastfrequenz aufgezwungen. Damit können Sie gezielt Korrekturen vornehmen, wenn die automatische Anpassung der Abtastfrequenz im Assampler versagt.

Ist der Schalter nicht gesetzt, wird das Eingangssignal mit der gleichen Abtastfrequenz wie das Ausgangssignal ausgelesen. Die Abtastung des Modulationssignals ist dagegen immer an die Abtastung des Ausgangssignals gekoppelt.

Frequency (Formel) 1/[x]

Gibt an, wie das Intervallgitter gelegt werden soll. Zur Zeit wird immer der Betrag der Frequenz benutzt, das ist aber unlogisch und wird sich in Zukunft vermutlich ändern.

Input (Klang)

Bestimmt das zu quantisierende Eingangssignal.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

Das Gitter muß nicht durchgehend gleiche Abstände haben. Mit diesem Steuersignal ist es möglich, das Gitter mal enger und mal weiter zu ziehen.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Phase (Formel) 1°

Legt fest, in welcher Phase das Gitter beginnen soll. 270° bedeutet zum Beispiel, daß der erste Wert noch eine Viertelperiode gehalten wird, bevor ein neuer Wert übernommen wird.

1.122 Umkehren

Umkehren

aus der Klasse X-Modulation

Funktion

Kehrt einen Klang zeitlich um.

Werte

Buffer (Klang)

Da Assampler die Signale normalerweise von vorn nach hinten geradewegs durcharbeitet, ist das Umkehren eines Klanges ein besonders heißes Eisen. Um diesen sehr grundlegenden Effekt dennoch bewerkstelligen zu können, arbeitet die Umkehrung ähnlich wie das Ablageobjekt. Zuerst wird das Eingangssignal im Puffer-Klang abgelegt, dieser wird dann umgekehrt, und der Puffer-Inhalt wird an die Nachfolgeprozesse weitergeleitet. Auch hier darf der Puffer der Zielklang der Berechnung sein.

Nachteil des Verfahrens ist, daß bei einer einfachen Umkehrung, also ohne vor- oder nachgeschaltete Prozesse, ein paar überflüssige Kopieroperationen anfallen.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Input (Klang)

das umzukehrende Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

1.123 Y Werte

Y Werte

aus der Klasse Bearbeiter

Unterklassen

linear

nicht-linear

Funktion

Diese Prozesse verändern die Signalwerte unabhängig voneinander.

1.124 linear

linear

aus der Klasse Y Werte

Unterklassen

Verstärker

Mischer

Anheben

Negieren

1.125 Wir benötigen Verstärkung

Verstärker

aus der Klasse linear

Funktion

Was Verstärker machen, kann man am Volume-Regler seiner Stereo-Anlage leicht nachvollziehen: Man kann Klänge lauter oder auch leiser machen. Zunächst wollen wir zwischen konstantem und steuerbarem Verstärker unterscheiden.

Der konstante Verstärker sollte z.B. verwendet werden, um ein Sample maximal auszusteuern. Will man einen berechneten Sound in 8-Bit-Auflösung speichern, ist es der Qualität wegen wichtig, daß die ohnehin wenigen 8 Bit gut ausgenutzt werden. Unterläßt man in diesem Fall die Verstärkung, werden leise Stellen bei der 16->8-Bit-Konvertierung zu sehr verzerrt.

Der Verstärker kann andersherum auch zum Abschwächen eines Samples eingesetzt werden, wenn man weiß, daß eine nachfolgende Operation zu einer Übersteuerung führen würde.

Interessant wird es beim steuerbaren Verstärker, denn damit lassen sich nicht nur Schönheitskorrekturen am Sample vornehmen, sondern auch Effekte setzen.

Die Steuerkurve eines Verstärkers wird auch Hüllkurve genannt. Man kann sich das leicht vorstellen, denn das Signal wird entsprechend des Verlaufes der Hüllkurve gedämpft, so daß an keiner Stelle die Lautstärke des Klanges die Hüllkurve überschreitet. Das Eingangssignal wird also von dieser Steuerkurve "eingehüllt".

Ein paar gängige Anwendungen des steuerbaren Verstärkers sollen nun gezeigt werden:

ADSR

Verwendet man als Steuerkurve einen ADSR-Generator kann man Lautstärkeverläufe erzielen, wie sie für natürliche Instrumente typisch sind. Im Falle einer kurzen Attack-Phase mit längerer Decay- und fehlender Sustain-Phase, erhält man perkussive Klänge, wie sie Klavier, Gitarre usw. eigen sind.

Wählt man dagegen kurze Attack- und Decay-Phase mit lauter Sustain-Phase, kann man diese für Blasintrumente aller Art oder Orgel-Klänge nutzen.

LFO

Nimmt man langsame Sinus-Schwingungen zur Lautstärke-Modulation erhält man den Tremolo-Effekt. Man kann sich mit Musikern herrlich darüber streiten, was Tremolo genau bedeutet - für uns soll es eine reine Lautstärke-Modulation sein im Gegensatz zur Frequenz-Modulation beim Vibrato-Effekt.

Bei den analogen Synthesizern gibt es noch Unterschiede zwischen LFO (LowFrequencyOscillator) und VCO (VoltageControlledOscillator). Zum einen sind LFOs nicht steuerbar (also in ihrer Frequenz jeweils fest und nicht modulierbar), zum anderen liegen die LFO-Frequenzen, wie der Name vermuten läßt, unter der Hörschwelle. Bei der digitalen Synthese brauchen wir diese Unterscheidung nicht vorzunehmen, da Computer hohe wie tiefe Frequenzen gleich gut erzeugen (ok, tiefe sogar qualitativ besser).

Werte

Input (Klang)

das zu verstärkende Eingangssignal

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Inverse (Schalter)

Wenn aktiviert, wird die Modulation nicht mit dem Eingangssignal multipliziert, sondern letzteres wird durch die Modulation dividiert, aber weiterhin mit Volume multipliziert. Enthält Modulation die Hüllkurve des Eingangssignals, läßt sich hiermit das Eingangssignal von seiner Hüllkurve befreien, hat danach also einen gleichbleibend lauten Klang.

Modulation (Klang)

Die Hüllkurve, die auf das Eingangssignal geprägt werden soll. Es wird keine Hüllkurve aufgeprägt, falls Modulation nicht gesetzt.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Volume (Formel) [y] = [Input.y] * [Modulation.y] * [Volume]

Verpaßt dem Eingangssignal nach dem Aufprägen der Hüllkurve eine konstante Verstärkung.

Theorie

Ich möchte darauf hinweisen, daß die Verwendung von AlphaChannels bei Klängen (siehe WaveTracer) keine besonders glorreiche Idee ist. Erstens kann man den Speichervorteil von Steuerkurven gut mit den Spline-Objekten des Assamplers erreichen. Zweitens greift der Verstärker, wie wir gleich sehen werden, in das Frequenzspektrum des Klanges ein, so daß die Bearbeitung mit bzw. ohne AlphaChannel im allgemeinen zu unterschiedlichen Ergebnissen führt!

Jetzt müßten Sie noch einmal nachhaken: Verstärker ändern das Frequenzspektrum? Ja, das ist fein beobachtet! Erinnern Sie sich an den Phasenschieber-Effekt? Dort haben wir durch Mischen von Tönen Lautstärkemodulationen erhalten. Das geht natürlich auch umgekehrt. Wir können durch Lautstärkemodulation neue Frequenzen erzeugen.

Verwendet man als Hüllkurve eine Siunsschwingung deren Frequenz dicht bei der Frequenz unseres Tones oder bei einem Vielfachen derselben liegt, tritt die Lautstärkemodulation im herkömmlichen Sinne völlig in den Hintergrund. Der Effekt wird mitunter als Ringmodulation bezeichnet, aber meine Quellen widersprechen sich da, also genießen Sie es mit Vorsicht. Um die Möglichkeiten dieser Modulationsart auszuloten, bleibt einem nichts weiter als Ausprobieren übrig. Folgendes Additionstheorem sollte Ihnen helfen, gezielt vorzugehen:

sin (f1*t) * sin (f2*t) = ½ * (cos ((f1-f2)*t) - cos ((f1+f2)*t))

Das bedeutet, daß die Lautstärke-Modulation eines Tones mit Frequenz f1 mit einer Sinusschwingung der Frequenz f2 zu zwei neuen Frequenzen, die symmetrisch bezüglich f1 liegen, führt.

1.126 Moulinette

Mischer

aus der Klasse linear

Funktion

Wie Sie's wahrscheinlich geahnt haben, mischt der Mixer mehrere Klänge zusammen. Oder überlagert die Signale, wenn Ihnen das geläufiger ist.

Werte

Die Werte sind wieder einmal in einer der berüchtigten Listen untergebracht, das bedeutet, daß Sie die Anzahl der Werte ändern können.

Array (Feld)

Array.Input (Klang)

die Klänge, die miteinander gemischt werden sollen

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Array.Volume (Formel) [y] = [Input.y] * [Volume]

die Lautstärke mit der jeder einzelne Klang in das Endergebnis einfließen soll

Technik

Die Eingangsklänge werden tatsächlich in einem Abwasch gemischt. Ich betone das, weil ich den Mixer zuerst durch Kaskaden realisiert hatte, d.h. Signal 1 + Signal 2 -> Signal 2', Signal 2' + Signal 3 -> Ausgangssignal. Warum das nicht ganz korrekt ist, erfahren Sie unter dem Punkt Theorie.

Optimierung

Beschleunigung, falls keiner, einer oder zwei Kanäle gemischt werden.

Theorie

Mischt man mehrere Signale, indem man das Mischen auf mehrere Mischvorgänge mit je 2 Eingangssignalen zerlegt (Kaskade), taucht das Problem des begrenzten Zahlenbereichs beim Speichern der Sample-Werte auf. Ein derartiger Mixer als Operation betrachtet wäre nicht assoziativ, d.h. (Signal1 + Signal2) + Signal3 = Signal1 + (Signal2 + Signal3) ist nicht immer erfüllt, obwohl es das eigentlich sein sollte.

Ein Beispiel gefällig? Seien Signal 1 und 2 Rechteckschwingungen der gleichen Frequenz und gleicher Amplitude 1 und sei Signal 3 das negierte Signal 1. Probieren Sie beide Wege durch: Signal 1 + Signal 2 -> Rechteckschwingung mit Amplitude 2. Durch unsere Meßskala können aber maximal Amplituden von 1 auftreten, das Rechtecksignal wird auf Amplitude 1 begrenzt. Mischt man dazu Signal 3, löschen sich beide Schwingungen aus. Mischt man zuerst Signal 2 mit Signal 3 ist das Ergebnis ein leerer Klang. Mischt man dazu Signal 1 erhält man wieder Signal 1, also die Rechteckschwingung.

Sie sehen, man kommt auf zwei verschiedenen aber prinzipiell gleichen Wegen zu stark verschiedenen Ergebnissen. Der Mixer mit mehreren Eingängen unterbindet dies durch eine intern höhere Wortbreite. Nichtsdestotrotz steht es Ihnen weiterhin offen, Mixer mit je 2 Eingängen hintereinanderzuschalten, Assampler würde das sogar korrekt berechnen, aber aus Speicher- und Übersichtlichkeitsgründen halten Sie sich lieber an den Mischer mit mehreren Eingängen.

Tips

Möchte man die Grundfrequenz eines Klanges ermitteln, um den Klang zum Beispiel zu stimmen, kann einem der Mixer in Verbindung mit einem Oszillator helfen. So wählt man als einen Eingang den Oszillator und als anderen den zu stimmenden Ton und dann kann man das Klanggemisch abspielen. Ist die Oszillatorfrequenz nah genug an der Grundfrequenz hört man die typischen Schwebungen. So kann man sich in mehreren Schritten an die Grundfrequenz des Klanges heranarbeiten. Durch Transponieren (resample) läßt sich dann der Klang einstimmen.

1.127 Anheben

Anheben

aus der Klasse linear

Funktion

Erhöht das Eingangssignal um einen bestimmten Wert. Für die Elektrotechniker: Es wird ein Gleichspannungsoffset hinzugefügt.

Werte

Input (Klang)

das Eingangssignal, das angehoben werden soll

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Level (Formel) [y]

Größe des Offsets

Theorie

Eine konstante Funktion entspricht einem Ton der Frequenz 0 und ist für das menschliche Ohr nicht zu hören. Entsprechend verändert dieser Prozeß abgesehen von Begrenzungsverzerrungen den Klangeindruck nicht. Allerdings ist der Wechsel zwischen verschiedenen Offsets als Knacks hörbar, und daher sollte der Gleichspannungsoffset immer null sein, sonst knackt es beim Abspielen des Klanges zu Beginn und am Ende.

1.128 Darf man nicht so negativ sehen

Negieren

aus der Klasse linear

Funktion

Negiert alle Werte des Eingangssignals.

Werte

Input (Klang)

das zu negierende Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Theorie

Wenn man sich das Signal aus Sinusschwingungen aufgebaut vorstellt, verschieben sich alle Sinusschwingungen um 180°.

Die Phasenlage eines Mono-Signals wird vom Ohr bekanntlich nicht wahrgenommen. Erst wenn das negierte Signal wieder mit anderen zusammengebracht wird, z.B. durch Mischen oder durch Modulation anderer Prozesse, wird die Veränderung hörbar.

1.129 nicht-linear

nicht-linear

aus der Klasse Y Werte

Unterklassen

Begrenzen

Umklappen

Schrumpfen

Quanteln in Y-Richtung

Abbildung

1.130 Weisen Sie den Klang in seine Grenzen

Begrenzen

aus der Klasse nicht-linear

Funktion

Begrenzt das Signal in der Lautstärke, zu große Auslenkung werden einfach abgeschnitten. Das ist der Effekt, der E-Gitarren den bekannten rauhen, kratzigen Sound verleiht, dort auch Fuzz-Booster oder Begrenzerverstärker genannt.

Werte

Input (Klang)

das zu verzerrende Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

Falls hier ein Klang eingestellt ist, wird die Begrenzung über die Zeit hinweg variiert. Die Begrenzung ist dann Modulation.y * Volume.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Volume (Formel) [y]

Wird dieser Wert von einem Eingangssignalwert betragsmäßig überschritten, wird der Signalwert auf Volume (mit dem originalen Vorzeichen, versteht sich) gesetzt.

1.131 Umklappen

Umklappen

aus der Klasse nicht-linear

Funktion

Klappt Signalwerte um, wenn Sie die Begrenzung erreichen. Sollte ein Signalwert dadurch die gegenüberliegende Begrenzung überschreiten wird er wieder umgeklappt usw. usf.

Werte

Input (Klang)

das zu verzerrende Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

Falls hier ein Klang eingestellt ist, wird die Begrenzung über die Zeit hinweg variiert. Die Begrenzung ist dann Modulation.y * Volume.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Volume (Formel) [y]

Wird dieser Wert von einem Eingangssignalwert betragsmäßig überschritten, wird der Signalwert an dieser Begrenzung gespiegelt.

1.132 Schrumpfen

Schrumpfen

aus der Klasse nicht-linear

Funktion

Verschiebt Signalwerte symmetrisch zur x-Achse.

Werte

Input (Klang)

das zu verzerrende Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

Falls hier ein Klang eingestellt ist, wird die Verschiebung über die Zeit hinweg variiert. Die Distanz ist dann Modulation.y * Volume.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Volume (Formel) [y]

Gibt die Differenz an, um die die Signalwerte verschoben werden sollen. Positive Werte bedeuten eine Verschiebung von der x-Achse weg, negative Werte bedeuten eine Verschiebung zur x-Achse hin. Werte, die durch die Verschiebung die x-Achse überschreiten würden, werden auf Null gesetzt.

1.133 Käsequanten

Quanteln in Y-Richtung

aus der Klasse nicht-linear

Funktion

Quantisiert das Eingangssignal in der y-Richtung. Ähnliches passiert beim Digitalisieren von natürlichen Tonquellen. Der Unterschied zwischen 8-Bit-, 12-Bit- und 16-Bit-Sample-Sounds besteht in der Genauigkeit, in der die Auslenkungen erfaßt werden. Sie können die Qualität der 16-Bit-Sample-Sounds im Assampler mit Hilfe der Y-Quantifizierung nachträglich verschlechtern. Das ergibt zum einen einen interessanten Effekt und kann zum anderen für die Bearbeitung von Steuerkurven eingesetzt werden.

Mathematisch gesehen wird der Wertevorrat auf ein paar gleichmäßig verteilte Werte eingeschränkt, und alle Signalwerte auf den nächsten dieser Werte abgerundet.

Werte

Input (Klang)

Das zu verarbeitende Eingangssignal. Kein Sound eingestellt bedeutet Benutzung des jeweils aktiven Sounds.

Die Lautstärkeanpassung versagt manchmal, wenn man den Ausgang an einen lautstärke-entkoppelnden Prozeß anschließt. Falls Ihnen dadurch unerwartete Effekte widerfahren, schalten Sie einen ausreichend dimensionierten Begrenzer nach.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Offset (Formel) [y]

ein Wert, der im Wertevorrat auf jeden Fall enthalten sein soll

OffsetMod (Klang)

Falls Klang eingestellt, wird Offset moduliert. Die zeitabhängigen Offsetwerte sind dann OffsetMod.y * Offset.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Step (Formel) [y]

Schrittweite von einem zum nächsten Wert des eingeschränkten Wertevorrats

StepMod (Klang)

Falls ein Klang eingestellt ist, wird die Schrittweite moduliert. Die zeitabhängigen Schrittweiten sind dann StepMod.y * Step.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

1.134 Abbildung

Abbildung

aus der Klasse nicht-linear

Funktion

Die Abbildung stellt die Verallgemeinerung aller Verzerrungen dar, ist aber im Gegensatz zu den Prozessen für die Spezialfälle nicht modulierbar. Für jeden Signalwert wird in einer Wertetabelle nachgeschlagen und der Wert aus der Wertetabelle in das Ausgangssignal übernommen.

Werte

Input (Klang)

das Signal, das der Abbildung unterzogen wird

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Interpolation (Auswahl)

auf welche Weise Werte ermittelt werden sollen, die nicht in der Wertetabelle stehen

irrelevant für Umkehrabbildung

Inverse (Schalter)

Wenn aktiviert, wird die Umkehrabbildung berechnet. D.h. es wird in der Wertetabelle die Stelle gesucht, an der der aktuelle Eingangssignalwert angenommen wird. Diese Stelle ist nur eindeutig bestimmt, wenn Table eine streng monotone Funktion (also nur fallend oder nur wachsend) ist. Was bei nicht umkehrbaren Funktionen passiert ist Glücksache.

Prinzipiell hebt das Nachschalten der Umkehrabbildung hinter die Abbildung den Effekt vollständig auf. Praktisch allerdings nicht, denn die Umkehrabbildung bedient sich eines sehr ungenauen, aber schnellen Verfahrens.

Table (Klang)

Dieser Klang enthält die Funktion, über die abgebildet wird. Aus ihm wird dann eine Wertetabelle erstellt. Beispiel: Tabellen-Klang sei Freiformkurve mit SampleFreq = 100/V, Begin = -1V, Length = 2V, Volume = 1m. Das bedeutet, daß die Abbildung für Argumente von -1V bis 1V (-1V+2V) definiert ist und 100 Werte pro 1V zur Verfügung stellt. Die y-Einheit des Eingangssignals an der Abbildung muß V sein und für das Ausgangssignal ist sie 1m.

Der Klang muß im Falle einer umgekehrten Abbildung eine Variable Length enthalten und einen Beginn besitzen. Falls es sich bei der Tabelle nicht gerade um einen datenbasierten Klang handelt, erreichen Sie das am einfachsten durch Dazwischen-Schalten eines Ausschnittprozesses.

Kopplung Lautstärke gekoppelt, Abtastrate fixiert, Beginn fixiert

1.135 Zigarette? Aber nur mit Filter!

Filter

aus der Klasse Bearbeiter

Unterklassen

nicht-rekursiv

rekursiv

Theorie

Viel Aberglaube herrscht auf dieser Welt, was den Begriff des Filters angeht. Wenn es allerdings konkret um Signalfilter geht, wollen wir nur das als Filter bezeichnen, was in einem Signal bestimmte Frequenzen um beliebige (aber vom Eingangssignal unabhängige) Faktoren verstärkt. Überträgt man diese Faktoren in ein Frequenz-Faktor-Diagramm, erhält man eine Anschauung für die Übertragungsfunktion, manchmal auch Frequenzgang genannt. Durch die Übertragungsfunktion ist ein Filter eindeutig bestimmt.

Diese Definition ist Ihnen noch am ehesten geläufig, aber es gibt noch eine einfachere, die nicht darauf aufbaut, daß man jedes Geräusch in Sinusschwingungen zerlegen kann. Nach dieser Definition sind Filter und nur Filter lineare Operationen. Mathematisch bedeutet das: T(a*x+b*y)=a*T(x)+b*T(y), wobei T(x) die Filteroperation auf das Signal x bezeichnet. Für Klänge bedeutet das, daß es egal ist, ob man erst verstärkt und mischt und dann filtert, oder ob man erst filtert und dann verstärkt und mischt. Klingt zunächst einmal völlig aus der Luft gegriffen und scheint wenig mit der vorangehenden Definition zu tun zu haben. Aber prüfen Sie es ruhig an Beispielen für natürliche Filter nach.

Die zweite Begriffserklärung geht allerdings noch etwas weiter als die erste, denn nach der ersten müßte man das, was wir intuitiv modulierte Filter nennen, ausschließen. Es ergeben sich aber gerade interessante Effekte, wenn man die Parameter eines Filters mit der Zeit ändert. Damit verläßt man allerdings langsam den Boden des klassischen Filters, was man im Extremfall daran sieht, daß dann ein Filter auch durch sein Eingangssignal moduliert werden kann. Das Ergebnis hat nur noch wenig mit dem intuitiven Begriff eines Filters zu tun - ist deswegen aber keineswegs nutzlos.

Abbildungen zur Wirkungsweise eines Filter zeigen manchmal Frequenzdiagramme (also Peaks in regelmäßigen Abständen, an dieser Stelle vermißt man wieder mal schmerzlich die nicht vorhandene Einbindung von Bildern in AmigaGuides) über das eine Hüllkurve gelegt wird und alle überstehenden Peaks abgeschnitten werden. Abgesehen davon, daß zur Veranschaulichung auch mal Details unterschlagen werden dürfen, suggeriert diese Darstellung, daß mehrmaliges Anwenden eines Filters auf ein Signal keine weitere Wirkung zeigt. Unterstützt wird dieser Irrtum durch Formulierungen wie "Frequenzen oberhalb der Grenzfrequenz

werden abgeschnitten" (klar werden z.B. die hohen Frequenzen irgendwie von den tiefen abgeschnitten (zumindest beim idealen Filter), aber die Frequenzen werden nicht etwa in der Amplitude begrenzt.) Ganz im Gegenteil ist die Hintereinanderausführung (Kaskadierung) mehrerer Filter ein beliebtes Mittel um die Filterwirkung zu verstärken.

Schlaue Menschen haben herausgefunden, daß sich Filter berechnen lassen, indem man jeden Wert des Ausgangssignals als Linearkombination von Eingangswerten und vorhergehenden Ausgangswerten ermittelt. Alles was sich Filter nennen möchte, muß sich auf diese Form bringen lassen. Verwertet man nur die Eingangswerte spricht man nichtrekursiven Filtern. Ihr Entwurf (also das Bestimmen der Faktoren der Linearkombinationen) ist vergleichsweise einfach, dafür ist die Berechnung recht langsam, da schärfere Frequenztrennungen und das Erfassen tiefer Frequenzen größere Filterfenster (= einzubeziehende Werte = Rechenaufwand) erfordert.

Bezieht man in die Rechnung zusätzlich die schon berechneten Ausgangswerte mit ein, erhält man rekursive Filter. Ihr Entwurf ist nicht ganz so trivial, dafür sind sie schön schnell, aber wiederum schwer im Zaum zu halten (Stichwort Stabilität), da sie auch Resonanzkatastrophen (im Rechner Zahlenüberlauf -> Datenmüll) heraufbeschwören können. Das macht Ihren Rechner zwar nicht kaputt, klingt aber nicht gut, und sollte auch nicht als Rauschgenerator eingesetzt werden! Im übrigen lassen sich die Filter der Analogtechnik mit rekursiven Filtern simulieren, nichtrekursive Filter gibt es da, so weit ich weiß, nicht. In gewisser Weise kann man die analogen Filter als Fortsetzung der rekursiven Filter betrachten, wenn man die Abtastrate ins unendliche erhöht.

Nach der oben gegeben Definition entpuppen sich auch andere Prozesse des Assamplers als Filter. Ziehen wir für einen Filter nur den jeweils aktuellen Wert des Eingangssignals heran, dann ergibt das einen normalen Verstärker. Die Übertragungsfunktion ist diesem Falle eine konstante Funktion, d.h. alle Faktoren für die Frequenzen sind gleich.

Benutzt man den aktuellen und noch einen früheren Eingangswert entsteht ein Phaser, dessen typischer Klang sich einstellt, wenn man die Zeitdistanz zwischen beiden Werten variiert. Verwendet man als früheren Wert statt des Eingangswertes einen Ausgangswert, bekommen wir einen rekursiven Filter, bei ausreichend großer Zeitspanne als Echo aufgefaßt.

Eine andere Einteilung der Filter, ist die nach dem ungefähren Frequenzgang. Betont ein Filter tiefe Frequenzen oder schneidet hohe weg ist es ein Tiefpass (bedeutet: die Tiefen läßt er passieren), umgekehrt ist ein Filter welches hohe Frequenzen gegenüber den tiefen verstärkt ein Hochpass. Schält ein Filter aus einem Frequenzgemisch die Frequenzen (Frequenzband) um eine bestimmte Frequenz (Bandmitte) heraus, ist es ein Bandpass, unterdrückt es ein Frequenzband nennt man es Bandsperre. Eine zunächst unsinnig erscheinende Filterart ist der Allpass, er läßt alle Frequenzen gleichermaßen hindurch, verändert aber deren Phasenverschiebung, wobei verschiedene Frequenzen unterschiedlich beeinflußt werden können. In der Analogtechnik, die ja mit dem Manko leben muß, daß man Analogsignale nur schwer speichern kann, benutzt man die Allpässe zur Signalverzögerung.

Sucht man in seiner Umgebung aufmerksam nach Vorgängen, die sich mit Filtern vergleichen lassen, wird man schnell fündig. Ich glaube besonders leicht sind Tiefpass-Filter zu finden. Überall wo irgendetwas geglättet wird oder etwas träge ist, kann man den Vorgang als Tiefpass beschreiben. Z.B. das Tacho, das nicht jeder kleinen Geschwindigkeitsänderung folgen kann und auch gar nicht soll, weil Sie keinen zitternden Zeiger sehen wollen. Das, was das Tachometer anzeigt, kann als geglättete (=tiefpassgefilterte) Kurve der Momentangeschindigkeiten angesehen werden. Oder nehmen Sie Regen, wie er von den Bäumen zurückgehalten wird. Er kommt zwar früher oder später doch zum Boden, aber kurze Huschen fangen die Baumkronen ab und geben sie verteilt wieder ab.

Hochpässe findet man zum Beispiel in Mikrophonen. Deren Grenzfrequenz ist aber so tief, daß man deren Effekt nicht hört. In einem Mikrophon befindet sich nämlich am Ende ein Luftloch, das für einen Druckausgleich sorgt. Fehlt es, kann bei unterschiedlichen Luftdrücken (Unterschied z.B. zwischen Berg- und Flachland) die Membran auch ohne Schalleinwirkung dauerhaft ausgelenkt sein, was zu einer konstanten Spannung am Verstärker führt. Diese Spannung nennt man Spannungsoffset und ist im Prinzip ein Signal der Frequenz 0 oder zumindest extrem tieffrequent (wenn sie einmal im Jahr zwischen Berg und Tal pendeln beträgt die Frequenz 32 nHz :-) Dieser Offset kann z.B. beim Digitalisieren stören, da dann der Wertebereich unsymmetrisch wird und Sie bei den in der Regel nach oben und unten gleich ausgesteuerten Signalen Auflösung verschenken.

Auch an Bildern läßt sich herumfiltern, wie Sie vielleicht von diversen Bildverarbeitungsprogrammen (ImageProcessing) wissen. Dort betrachtet man die Helligkeit als Funktion von der x- und der y-Koordinate, bei Farben die Intensitäten der roten, grünen und blauen Farbanteile. Bei diesen Bildverarbeitungen kommen eigentlich nur nicht-rekursive Filter zum Einsatz - rekursive wären zwar auch nicht schlecht, aber das ist wohl (noch) nicht so populär.

Wußten Sie auch, daß selbst beim menschlichen Sehen ein Hochpass im Spiel ist? Zeichnen Sie einfach mit einem Malprogramm einen Farbverlauf über den ganzen Bildschirm mit 16 Abstufungen, von weiß nach blau meinetwegen. Sie können auch eines jener Tools verwenden, das einen Farbverlauf per Copper auf der Workbench installiert. Sehen Sie, wie in jedem der 16 Streifen der Helligkeitsverlauf dem des gesamten Bildes entgegengesetzt zu sein scheint. Vielleicht haben Sie sich schon öfter daran gestört, daß Sie mit den 16 Farbabstufungen der alten (OCS-)Amigas keinen absolut weichen Übergang hinbekommen konnten. Das liegt daran, daß das Auge die Konturen verstärkt. Normalerweise soll es einem diese Einrichtung ermöglichen, auch bei wenig kontrastreichen Bildern Gegenstände erkennen zu können.

Tja, wo findet man nun Bandpässe? Zum Beispiel Brücken. Sie kennen vielleicht die galoppierende Gertie, die Brücke die kurz nach ihrem Aufbau durch einen Sturm eingerissen wurde, aber nicht, weil der Sturm übermäßig heftig war, sondern weil die Brücke in Schwingung geriet, sich selbst immer weiter aufgeschaukelt hat, bis sie auseinanderriß. Das ist die berüchtigte Resonanzkatastrophe, die ich vorhin erwähnte. Die Frequenz bei der das passiert, nennt man die Resonanzfrequenz - allerdings kann man Bandpässe auch so konstruieren, daß sie keine Resonanzkatastrophe produzieren können, dort nennt man eben die Frequenz, auf die das Bandpass-Filter am meisten anspricht, Resonanzfrequenz.

Wenn wir Radio hören, hat das Signal ebenfalls bereits einen Bandpass passiert. Es werden nämlich die Musik- oder Sprachsignale beim Radiosender auf eine Trägerfrequenz (allgemein als die Sender-Frequenz bekannt) moduliert, entweder per Amplituden- oder Frequenzmodulation. Nun können Sie abschätzen, daß bei hörbaren Signalen Frequenzen von 0 bis 20khz auftreten, folglich sollte das Sendefrequenzband 2*20kHz breit sein. Das Radio empfängt nun alle Frequenzen von allen Sendern der Umgebung, und muß aus diesem Frequenzgemisch einen bestimmten Sender herausfischen, das geschieht mit einem Bandpass. Anschließend muß das Signal noch demoduliert werden.

Und auch wieder an uns selbst finden wir Bandpässe. Wenn Sie mal in ein Ohr schauen (probieren Sie's mal an Ihrem eigenen, hehe), ähm, da werden Sie wahrscheinlich auch bloß nichts sehen können. Jedenfalls befinden sich in unseren Ohren Härchen, die von den Tönen aus der Luft zum Schwingen angeregt werden. Jedes Härchen reagiert auf ein kleines Frequenzband und so ist schon vor der Weitergabe der Informationen ans Gehirn eine erste Auswertung erfolgt und das Gehirn weiß bereits, welche Töne in dem ankommenden Geräusch enthalten sind.

Nur natürliche Allpässe zu finden ist mir nicht gelungen. Auch die anderen Filter können natürlich Phasenverschiebungen hervorrufen, daß Sie aber die Frequenzzusammensetzung ändern, läßt sich kaum ausschließen.

Intellenztest

Welche der folgenden Operationen sind (modulierte) Filter?

Verstärken, Anheben (um konstanten Wert), Mischen, gleichrichten, Phasenmodulation, Frequenzmodulation

Antwort

1.136 nicht-rekursiv

nicht-rekursiv

aus der Klasse Filter

Unterklassen

nichtrekursiver Tiefpass

allgemeines nichtrekursives Filter

Ableitung

1.137 Damit alles glatt geht

nichtrekursiver Tiefpass

aus der Klasse nicht-rekursiv

Funktion

Ein nichtrekursiver Tiefpass mit einem langen Rechteckimpuls als Fenster.

Werte

Depth (Formel)

Die übliche Bedeutung

Input (Klang)

der zu filternde (=glättende) Klang

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

Modulationskurve für die Fensterbreite, oder nichts, falls der Filter nicht moduliert werden soll.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Scale (Formel)

Die übliche Bedeutung

Width (Formel)

Die Weite des Filterfensters, welche umgekehrt proportional zur Filter-Grenzfrequenz ist. Bei stückweise konstanten Signalen kann man den Wert so interpretieren: Erst wenn ein neues Signalniveau solange anhält, wie das Filterfenster breit ist, erreicht der Filterausgang das Eingangsniveau.

Betrachtet man die Übertragungsfunktion des Filters, sieht man, daß er die Frequenzen unterhalb der Grenzfrequenz passieren läßt, je näher an der Grenzfrequenz desto schwächer bis zur Grenzfrequenz, die komplett eliminiert wird. Über der Grenzfrequenz werden auf einmal wieder Frequenzen hindurchgelassen (aber nur schwach) und nur ganzzahlige Vielfache der Grenzfrequenz werden wirklich exakt ausgelöscht.

Technik

Speicherverbrauch wächst proportional zur Fensterbreite, mithin umgekehrt proportional zur Filterfrequenz, d.h. tiefe Frequenzen brauchen mehr Speicher als hohe.

Geschichte

Früher wurde statt der Fensterbreite die Filterfrequenz abgefragt. Sobald der Filter modulierbar wurde, erwies sich das aber als unpraktisch, denn bei linearer Skalierung wird der Einfachheit halber linear bezüglich der Fensterbreite moduliert. Um die Konfusion in Grenzen zu halten habe ich die Frequenzangabe durch die Filterfensterweite ersetzt.

1.138 Wenn Ihr Klang erste Falten bekommt ...

allgemeines nichtrekursives Filter

aus der Klasse nicht-rekursiv

Funktion

Ein nichtrekursives Filter, bei dem die Impulsantwort in Form eines anderen Sounds frei gewählt werden kann. Der mathematisch Begriff für diesen Vorgang heißt Falten (engl. Convolution).

Werte

Input (Klang)

Der zu filternde (=faltende) Klang.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Volume (Formel)

Ein Verstärkungsfaktor, mit dem man die Lautstärke des meist zu leise ausfallenden Ausgangssignals anheben kann.

Window (Klang)

Ein Klang, der die Impulsantwort beschreibt. Dieser Klang muß eine Variable namens "Length" besitzen, der die Länge desselben festlegt. Ursprünglich war vorgesehen, daß dieser Sound das Filterfenster repräsentiert, daher der Variablenname Window. Das Filterfenster ist immer zeitlich umgekehrt zur Impulsantwort, und genau das hat sich als unpraktisch erwiesen.

Woher bekommt man so eine Impulsantwort?

1. Man digitalisiert die Antwort auf einen Impuls. Dazu erzeugt man einen solchen Impuls und nimmt den Nachhall und ggf. das Echo auf. So einen richtig idealen Impuls bekommt man natürlich nicht so leicht, aber vielleicht tut's auch ein Fingerschnippen. Die Impulsantwort läßt sich dann dazu verwenden, die Klangcharakteristik des betrachteten Raumes in jeden beliebigen Sound einzuarbeiten.

2. Man entwirft eine Übertragungsfunktion, also einen Sound der Frequenzen auf einheitenlose Werte (Faktoren) abbildet. Aus dieser Übertragungsfunktion erhält man dann über eine Frequenztransformation die benötigte Impulsantwort.

3. Man benutzt einen Klang als Filterfenster, dessen Eigenschaften man dem Eingangssignal aufprägen möchte. Nimmt man zum Beispiel eine Schwingung der Rechteckwellenform und als Eingang Rauschen, erhält man Rauschen, das eben ein bißchen nach einer Rechteckschwingung klingt. Nimmt man mehrere Rechteckschwingungen hintereinander in das Filterfenster auf, wird der Effekt stärker (und die Berechnung gähnend langsam).

Die y-Einheit des Fensters ist übrigens egal, da das Fenster zuvor durch die von ihm eingeschlossene Fläche normiert wird.

Dafür, daß man das Filterfenster frei wählen kann, besteht das Handicap, daß die Impulsantwort über die Zeit hinweg gleich bleiben muß.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn entkoppelt

Technik

Bei kleineren Fenstern (derzeit bis 500 Samples) wird die Faltung von Hand vorgenommen, danach kommt eine FFT zum Einsatz. Die Qualität läßt dann sichtbar nach. Die Fenstergrenze ist gerade so gewählt, daß auf meinem Rechner immer die schnellere von beiden Methoden angewendet wird. Auf anderen Rechnern muß das nicht erfüllt sein, es wäre dort sogar möglich, daß man die Berechnung durch Verlängern des Fensters beschleunigt.

1.139 Zwischen manchen Begriffen muß man genau differenzieren

Ableitung

aus der Klasse nicht-rekursiv

Funktion

Berechnet die Ableitung des Eingangssignals. Gemeint ist hier tatsächlich der mathematische Begriff des Differenzierens. Es handelt sich natürlich nur um ein numerisches Verfahren, hier wird nicht mathematisch korrekt abgeleitet.

Die Einheit des Ergebnissignals ist folglich [Input.Y] / [Input.X].

Das Ableiten entspricht einem nicht-rekursiven Filter, konkret einem Hochpass. Gedacht ist es aber zum Einsatz bei Simulationen.

Werte

Input (Klang)

Der abzuleitende Sound.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

1.140 rekursiv

rekursiv

aus der Klasse Filter

Unterklassen

rekursives Filter 1. Ordnung

Universalfilter

Integration

Echo

1.141 rekursives Filter 1. Ordnung

rekursives Filter 1. Ordnung

aus der Klasse rekursiv

Funktion

Dieser Prozeß vereint den einfachsten (=schnellsten) Tiefpass (entspricht Glättung) und den einfachsten Hochpass, den wir im Angebot haben. Entspricht einem RC-Tiefpass bzw. RC-Hochpass erster Ordnung.

Werte

Depth (Formel)

Die übliche Bedeutung

FilterType (Auswahl)

Wählt zwischen Tiefpass und Hochpass.

Frequency (Formel) 1/[x]

Die Grenzfrequenz des Filters, also die Frequenz, von der ab aufwärts der Tiefpass die Frequenzen (je höher desto stärker) unterdrückt und von der ab abwärts ein Hochpass die Frequenzen unterdrückt. Für die Physiker: Die Frequenz bei der Blind- und Wirkwiderstand gleich sind.

Zur Simulation eines RC-Filters wäre hier sowas wie 1/(2Pi * 1Ohm * 1nF) einzutragen.

Input (Klang)

der zu filternde Klang

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

Modulationskurve für die Filterfrequenz, oder nichts, falls das Filter nicht moduliert werden soll. Der Wert Null in der Modulationskurve bedeutet die normale Filterfrequenz.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Theorie

Der geglättete Klang wird so verstärkt, daß er im Extremfall gerade die maximale Amplitude erreicht. Das ist übrigens gleichzeitig die Skalierung, bei der ein Impuls durch die Filterung in eine Exponentialkurve mit der gleichen eingeschlossenen Fläche übergeht. Den Grund kann man sich zur Übung ruhig mal überlegen.

1.142 Universalfilter / State-Variable-Filter (Filter zweiter Ordnung)

Universalfilter

aus der Klasse rekursiv

Funktion

Dieser Filter vereint in sich Hochpass, Tiefpass, Bandpass und Bandsperre mit wählbarer Grenzfrequenz/Bandmittenfrequenz und Güte. Besonderheit: Auch Hoch- und Tiefpass verstärken ihre Grenzfrequenz bei höherer Güte mehr als die Frequenzen im Durchlaßbereich.

Werte

FilterType (Auswahl)

Der Typ des Filters (siehe Funktion)

FreqDepth (Formel)

Die übliche Bedeutung

FreqMod (Klang)

Modulationskurve für die Filterfrequenz, oder nichts, falls dieser Filterparameter nicht moduliert werden soll.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

FreqScale (Formel)

Die übliche Bedeutung

Frequency (Formel) 1/[x]

die Grenz- bzw. Bandmittenfrequenz des Filters

Input (Klang)

der zu filternde Klang

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Precision (Auswahl)

Falls eine Modulation eingestellt ist, gibt dieser Wert an, wie genau die internen Parameter des Filters aus den anschaulichen Parametern Filterfrequenz und Güte berechnet werden sollen. Die Präzision ist immer "exact", wenn nicht moduliert wird.

exact - das Filter wird ohne bewußte Rundung gesteuert, dauert am längsten, kann aber wichtig sein, wenn z.B. gefiltertes Rauschen mit einem Ton gleicher Frequenz gemischt werden soll

approximate - durch ein paar Rundungen wird etwas höhere Geschwindigkeit erkauft, Unterschiede zu exact sollten nicht hörbar sein

sketch - Filter wird nur pi mal Daumen mit Parametern bestückt, das reicht für manche Effekte, in denen es nur um ein Auf- und Ab an der richtigen Stelle geht, wie bei pfeifendem Wind oder Drums

QualDepth (Formel)

Die übliche Bedeutung

Quality (Formel)

die Güte des Filters

QualMod (Klang)

Modulationskurve für die Güte, oder nichts, falls dieser Filterparameter nicht moduliert werden soll.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

QualScale (Formel)

Die übliche Bedeutung

Volume (Formel) [y] = [Input.y] * [Volume]

Ein Lautstärkefaktor, um zu leise oder zu laut gewordene Ergebnisse optimal aussteuern zu können.

Hängt das Universalfilter in einer Kette von Prozessen, ist es wichtig, das Ergebnis des Filter bereits testweise in einem Sample-Sound ausgerechnet zu haben, da das Filter selbst keine brauchbare Voraussage für die Lautstärkeentwicklung des Ausgangssignals treffen kann. Anhand dieses Samplesounds sollte man einen Wert für Volume finden, der das Signal gut aber nicht allzu knapp am Limit aussteuert.

Technik

Auch hierzu gibt es ein Äquivalent in der Analogtechnik. Dort lassen sich die Eigenschaften Filterfrequenz und Güte direkt einstellen. Mit den gequantelten Daten im Computer klappt das leider vorne und hinten nicht, deswegen muß besonders viel gerechnet werden, wenn das Filter moduliert wird. Kurz: Die Berechnung des Filters dauert deutlich länger, wenn man seine Eigenschaften moduliert.

1.143 Die Integ-Ration

Integration

aus der Klasse rekursiv

Funktion

Integriert das Eingangssignal fortlaufend. Die Einheit des Ergebnissignals ist folglich [Input.Y] * [Input.X].

Das Integrieren entspricht einem rekursiven Filter, konkret einem Tiefpass. Gedacht ist es aber zum Einsatz bei Simulationen.

Werte

Input (Klang)

der zu integrierende Sound

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Technik

Konstante Funktionen werden durch Integration bekanntlich zu linearen Funktionen. Vermeiden Sie deshalb Nullpunktverschiebungen im Eingangssignal. Selbst die Korrektur des Offsets reicht nicht aus, da damit immer nur in ganzen Schritten bezüglich der 16-Bit-Arithmetik verschoben wird.

Mit Integration gelangt man schnell an die Grenzen der 16Bit-Berechnung. Versuchen Sie nicht die Integration als Umkehrfunktion der Differentiation zu verwenden und umgekehrt.

1. Sind die Algorithmen nicht aufeinander abgestimmt.

2. Gibt das die 16-Bit-Arithmetik meistens nicht her.

1.144 Hallo Echo! - Hallo Otto!

Echo

aus der Klasse rekursiv

Funktion

Erzeugt Echos vom Eingangssignal und mischt sie hinzu.

Werte

Component (Auswahl)

Da Assampler nur mit einkanaligen Signalen umgehen kann, muß irgendein Weg gefunden werden, um auch Prozesse, die eigentlich mehrere Kanäle im Ausgangssignal haben, zu unterstützen. Mit Component wählen Sie bei komplexem Echos zwischen dem Real- und Imaginärkanal des Ausgangssignals.

Bei Anwendung auf Stereogruppen kann man hier zum Beispiel die Referenz _.Channel angeben.

Decay (Formel) 1

Lautstärkeabfall von einem zum nächsten Echo.

0 - das erste Echo tritt bereits nicht mehr auf

0.5 - jedes Echo ist genau halb so laut wie das vorangehende

1 - das erste Echo ist wieder genau so laut wie das ursprüngliche Schallereignis

Interessant wird es bei komplexen Decay-Werten. Mit ihnen ist es möglich, Echos innerhalb ihres Abfalls periodisch lauter und leiser werden zu lassen. Der Betrag der komplexen Zahl gibt den Gesamtabfall an, das Argument bestimmt die Geschwindigkeit, mit der die Echos lauter und leiser werden.

Beispiel: Decay = 0.8 * Exp (pi i/7) erzeugt Echos, die mit einer Periode von 7 lauter und leiser werden (das siebente Echo ist dann zwar genau negiert, aber das hört man nicht), insgesamt aber mit dem Faktor 0.8 leiser werden. Das siebente Echo ist also nur noch 0.8^7 mal so laut wie das Original.

Damit der Effekt funktioniert, müssen Sie auch einen komplexen Delay-Wert angeben, Input1 hingegen muß nicht belegt sein.

Delay (Formel) [x]

Bestimmt die Verzögerung zwischen zwei aufeinanderfolgenden Echos. Ist der Decay-Wert komplex, sollte es auch der Delay-Wert sein. Am einfachsten sind solche Ausdrücke wie 1s * (1+i). Man kann Real- und Imaginärteil auch verschieden wählen. Was dann passiert kann man schwer in Worte fassen, probieren Sie's einfach aus.

Input0 (Klang)

das Eingangssignal, welches mit Echos versehen werden soll

Es liefert die Realteile bei komplexen Echos.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Input1 (Klang)

das zweite Eingangssignal, welches mit Echos versehen werden soll

Es liefert die Imaginärteile bei komplexen Echos.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Volume (Formel) [y] = [Input0.y] * [Volume]

Eingangsverstärkung

Technik

Optimierung

Beschleunigung, falls der Decay-Wert reell ist und daher keine Mischung zweier Kanäle notwendig ist.

Theorie

Trotz daß ich mir von Anfang an sicher war, damit nur Kopfschütteln zu ernten, habe ich das Echo bei den rekursiven Filtern eingeordnet.

Der theoretische Grund: So wie der Echo-Prozess arbeitet, entspricht er genau der Definition eines rekursiven Filters.

Der praktische Grund: Wählt man eine ganz kurze Verzögerungsrate, klingt der Effekt tatsächlich nicht wie ein Echo sondern wie ein Filter. Das Prinzip ist aber für lange Verzögerungszeiten dasselbe, nur daß man die betroffenen tiefen Frequenzen beim besten Willen nicht hören kann.

Der Hintergrund: Hätte ich das Echo nicht bei den rekursiven Filtern eingeordnet, wäre sein Platz allein irgendwo zwischen anderen Verzweigungen gewesen, und da hätten Sie's auch nur nicht gefunden.

1.145 Spektrum

Spektrum

aus der Klasse Bearbeiter

Unterklassen

spektrumbasiertes Filter

spektrumbasierte Faltung

Funktion

Diese Prozesse arbeiten mit der spektralen Darstellung des Signals.

Werte

BaseFreq (Formel) 1/[x]

die Grundfrequenz wie beim Spektrum-Klang

Hier müssen Sie allerdings selbst auf eine Grundfrequenz achten, die eine schnelle Transformation erlaubt. Verwenden Sie deshalb die RoundPrimeSum() oder RoundPrimeMax()-Funktion.

Merge (Schieber)

die Anzahl der sich überlappenden Blöcke wie beim Spektrum-Klang

Volume (Formel) [y] = [Input.y] * [Volume]

Verstärkungsfaktor

1.146 spektrumbasiertes Filter

spektrumbasiertes Filter

aus der Klasse Spektrum

Funktion

Filtert Eingangssignal mit frei wählbarer Übertragungsfunktion, die außerdem mit der Zeit über den Frequenzbereich verschoben werden kann. Sie können damit Tiefpässe, Hochpässe, Bandpässe, Bandsperren und völlig neuartige Filter realisieren, welche zudem in der Filterfrequenz modulierbar sind.

Werte

Depth (Formel) 1/[x] = [Modulation.y] * [Depth]

Die übliche Bedeutung

Frequency (Formel) 1/[x]

Die Frequenz die die Stelle 0 in der Übertragungsfunktion repräsentieren soll. Hat die Übertragungsfunktion gerade an der Stelle 0 ihr Maximum bzw. den Übergang von 0 auf 1, dann entspricht Frequency der Bandmittenfrequenz des Bandpasses bzw. der Grenzfrequenz des Hoch- oder Tiefpasses.

Input (Klang)

das zu filternde Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

Hiermit moduliert man den Frequency-Wert und kann auf diese Weise die Übertragungsfunktion mehr in hohe oder mehr in tiefe Frequenzbereiche verschieben.

Die Modulation ist immer linear skaliert.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

TransFunc (Klang)

Die Übertragungsfunktion als Klang. Sowohl x- als auch y-Achse sollten einheitenlos sein. Die Übertragungsfunktion gibt für jede Frequenz den Verstärkungsfaktor an, der auf diese Frequenz wirken soll. Sie kann durch die Werte Frequency und Width noch in den richtigen Frequenzbereich abgebildet werden.

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Width (Formel) 1/[x] = [TransFunc.x] * [Width]

Gibt einen Streckungsfaktor in x-Richtung für die Übertragungsfunktion an. Beispiel: Die Übertragungsfunktion enthalte einen Peak, der Filter ist also ein Bandpass. Dann bedeuten kleine Width-Werte hohe Filtergüten. Dieser Parameter kann nicht moduliert werden.

1.147 spektrumbasierte Faltung

spektrumbasierte Faltung

aus der Klasse Spektrum

Funktion

Prägt die Klangcharakteristik eines Klanges auf einen anderen auf. Sehr bekannt ist der Vocoder-Effekt, bei dem man ein menschliche Sprache auf ein Trägersignal prägt. Hiermit kann man Effekte wie sprechende Gitarren oder Mundharmonikas erzeugen.

Werte

Input (Klang)

das Trägersignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Modulation (Klang)

der Klang, dessen Klangcharakteristik dem Eingangssignal aufgeprägt wird

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Smooth (Formel) 1/[x]

die Grenzfrequenz des Tiefpasses der benutzt wird, um die Frequenz-Informationen des Modulationssignals zu glätten

1.148 Analysiere das, D3!

Analyse

aus der Klasse Prozeß

Unterklassen

Detektoren

Frequenzspektrum

Funktion

Analyseprozesse arbeiten bestimmte Eigenschaften von Klängen heraus. Ihre Ausgangssignale sind in der Regel nicht mehr direkt musikalisch verwertbar, sondern dienen der Weiterwendung als Steuerkurven.

1.149 Trigger Dir einen

Detektoren

aus der Klasse Analyse

Unterklassen

Schwellwert-Trigger

Nulldurchgang-Trigger

1.150 Vorsicht Schwelle!

Schwellwert-Trigger

aus der Klasse Detektoren

Funktion

Ein Trigger hat zwei Eingangskippunkte und zwei Ausgangszustände. Unterschreitet das Eingangssignal den unteren Kippunkt (ThresholdHL) kippt der Ausgang des Triggers auf den einen Wert (FillL), überschreitet das Eingangssignal dann wieder den anderen Kippwert (ThresholdLH) springt der Ausgangswert auf FillH.

Das ganze wird fortwährend wiederholt und zeichnet dadurch einen stark quantisierten Verlauf des Eingangssignals nach.

Das Äquivalent in der Elektrotechnik heißt übrigens Schmitt-Trigger.

Werte

FillH (Formel)

Ausgangswert nach Überschreiten von ThresholdLH.

FillL (Formel)

Ausgangswert nach Unterschreiten von ThresholdHL (und vor erneutem Überschreiten von ThresholdLH natürlich).

Initialize (Auswahl)

Entscheidet in welcher Phase der Trigger beginnen soll, also ob zuerst FillL oder zu erst FillH ausgegeben werden soll.

Input (Klang)

das zu verarbeitende Eingangssignal

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

ThresholdHL (Formel)

Schwelle des Eingangswertes bei dessen Unterschreiten der Ausgang des Triggers auf FillL kippt.

ThresholdLH (Formel)

Schwelle des Eingangswertes bei dessen Überschreiten der Ausgang des Triggers auf FillH kippt.

Theorie

Die Differenz zwischen beiden Kippschwellen wird gewählt, damit der Ausgang nicht hin und her schaltet, weil das Eingangssignal für kurze Zeit minimal um den Kippwert schwankt. Meistens sind die Eingangskurven längst nicht so glatt, wie sie in der Anzeige aussehen. Durch die Differenz wird das sofortige Zurückkippen unterbunden, das Eingangssignal gewissermaßen entprellt.

Damit wir den Begriff auch mal genannt haben: Wenn man den Graph einer Parameterfunktion zeichnet, deren x-Komponente durch das Eingangssignal und deren y-Komponente durch Ausgangsignal gegeben sind, entsteht ein Gebilde das im Idealfall (und der Computertrigger ist ideal) wie ein Rechteck mit zwei überstehenden Strecken aussieht. Das Gerät heißt Hystereseschleife.

1.151 Nulldurchgang-Trigger

Nulldurchgang-Trigger

aus der Klasse Detektoren

Funktion

Setzt an jeder Stelle, an dem im Eingangssignal ein Nulldurchgang auftritt einen Impuls der Höhe Abtastfrequenz (oder eines Faktors davon) des Eingangssignales.

Die eigenartige Einheit der Impulshöhe erklärt sich damit, daß das Ausgangssignal bei einfachen Wellenformen (z.B. Sinus, Dreieck, Rechteck, Sägezahn) zur Bestimmung des Frequenzverlaufes eines Tones herangezogen werden kann. Bei natürlichen Klängen dient das Ausgangssignal dieses Prozesses als Indikator für Rauschanteile.

Da das Ergebnis stark von einem Gleichstrom-Offset abhängt ist es zweckmäßig, einen Hochpass vorzuschalten.

Werte

FillH (Formel) [y] / [x]

Die Höhe des Impulses der erzeugt wird, wenn das Eingangssignal von negativ zu positiv wechselt.

Die Höhe wird als Verhältnis zur Abtastfrequenz aufgefaßt, ist sie 1 (voreingestellt) ist sie also gerade so groß wie die Zielabtastfrequenz.

FillL (Formel) [y] / [x]

Die Höhe des Impulses der erzeugt wird, wenn das Eingangssignal von positiv zu negativ wechselt.

Input (Klang)

das zu analysierende Eingangssignal

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Theorie

Nur wenn die Impulshöhe in der oben geschilderten Weise von der Abtastfrequenz abhängt, erhält man insgesamt von der Abtastfrequenz unabhängige Ergebnisse. Das klingt zunächst völlig unlogisch, entpuppt sich aber als sehr sinnvoll, wenn man die Fläche unter der Kurve betrachtet: Geht man davon aus, daß die Anzahl der Impulse nicht von der Abtastfrequenz abhängt, bleibt immer noch das Problem, daß der kürzest mögliche Impuls bei einer höheren Abtastfrequenz auch eine geringere Fläche zur x-Achse hin einschließt. Gerade die ist aber ausschlaggebend, um bei weiteren Bearbeitungen mit Filtern zu Abtast-unabhängigen Ergebnissen zu gelangen. Deshalb muß bei höherer Abtastfrequenz die Impulshöhe proportional steigen.

Tips

Mit dem aus vielen Impulsen bestehenden Ausgangssignal kann man erstmal nicht viel anfangen, möchte man damit Modulationen vornehmen. Deshalb empfiehlt es sich, nachträglich einen Tiefpass anzuwenden, der eine halbwegs stetige Kurve liefert. Mit dieser läßt sich dann ein anderer Prozess (Frequenzmodulation, Filter, Trigger für stimmhaft/stimmlos-Detektor) speisen.

1.152 Frequenzspektrum

Frequenzspektrum

aus der Klasse Analyse

Unterklassen

Fouriertransformation

Absoluter Betrag

Winkel der komplexen Zahlen

Polar-zu-Komplex-Konvertierung

1.153 Foyertanzformation

Fouriertransformation

aus der Klasse Frequenzspektrum

Funktion

Berechnung des Frequenzspektrums eines Klanges oder das ganze umgekehrt.

Werte

Buffered (Schalter)

Trotz, daß bei der Frequenztransformation zwei Klänge anfallen (Real- und Imaginärteil), kann aufgrund des Assampler-Konzepts in einem Schritt immer nur ein Sound ausgelesen werden. Um für weitere (parallele) Anfragen die Frequenztransformation nicht erneut berechnen zu müssen, erlaubt der Buffered-Schalter, das Ergebnis zwischenzuspeichern.

Das verlangt von Ihnen ein bißchen Disziplin, da nun Änderungen der Eingabesounds nicht mehr berücksichtigt werden können. Hängt ein Eingabeklang von äußeren Parametern ab (referenzierte Werte), ist es normalerweise möglich, daß bei Berechnung von Real- oder Imaginärteil des Spektrums trotz des gleichen Klangobjekts als Eingabe verschiedenen Daten transformiert werden.

Component (Auswahl)

Gibt an, welche Komponente (Real- oder Imaginärteil) ausgegeben werden soll.

Direction (Auswahl)

Transformationsrichtung:

Analyse: Zeitsignal -> Frequenzspektrum

Synthese: Frequenzspektrum -> Zeitsignal

Beide Richtungen unterscheiden sich im wesentlichen durch ein paar Vorzeichen und Faktoren, ansonsten kann man sie gar nicht so recht unterscheiden.

Input0 (Klang)

Eingangssignal für den Realteil oder unbelegt, falls konstant Null

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Input1 (Klang)

Eingangssignal für den Imaginärteil oder unbelegt, falls konstant Null. Wird oft nicht benötigt.

Dieser Eingabeklang wird in Lautstärke und Abtastfrequenz an den nullten angepaßt.

Kopplung Lautstärke entkoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Length (Formel) [x]

Entspricht der Länge der Ausgabe. Beim Analysieren wird das eine Frequenz sein, beim Synthetisieren eine Zeit.

Falls der Eingabeklang eine feste Abtastfrequenz besitzt, wird diese benutzt und Length ignoriert.

Im Falle der Analyse bedeutet das, daß die höchste Frequenz im Spektrum der Abtastfrequenz des Quellklanges entspricht. Das heißt, so ganz stimmt das nicht, denn aufgrund der Rückfaltung nimmt man lieber den Bereich minus halbe Abtastfrequenz bis halbe Abtastfrequenz.

Volume (Formel) [y] = [Input0.y] * [Volume]

Ausgangsverstärkung

Anwendung

Zuerst zur Frage, wie man die Frequenztransformation anwendet:

Quell- und Zielklang haben zueinander vertauschte Abtastrate und Länge, d.h. Länge der Quelle = Abtastrate des Ziels und Abtastrate der Quelle = Länge des Ziels.

Daß ein Klang (... diese Bezeichnung wird an dieser Stelle unpassend und wir gehen zu Spektrum über ...) eine Frequenz als Länge hat, kann man sich gerade noch vorstellen, aber eine Zeit als Abtastfrequenz?

Schreiben Sie z.B. 2 s lieber als 2/(1Hz), was rein rechnerisch das gleiche ist. Das bedeutet dann: Verwende 2 Abtastwerte pro einem Hertz im Spektrum, oder auch: lege das Spektrum mit einer Skaleneinteilung von 1/2 Hz ab.

Und nicht vergessen: Wegen der Rückfaltung gibt es keine Frequenzen über der halben Abtastfrequenz, deswegen den Zielsound bei der Analyse mit einem Begin von -.Length/2 ausstatten!

Wie gibt man das ganze am bequemsten ein? Zu einem Sound erzeugen wir einen Sound für das Spektrum, indem wir den Originalklang klonen (entweder über die Schnittfunktionen der Klanggruppen oder über das Fenster mit den Soundklassen). Zu diesem Sound öffne man das Informationsfenster und ziehe das Length-Gadget über das SampleFreq-Gadget und fertig! (Ok, Begin nicht vergessen!)

Dann zur Frage, wofür man die Frequenztransformation einsetzt:

1. Tonhöhenbestimmung: Zu einem Klang erzeugen Sie das Spektrum, und können dann in selbigem nach Peaks suchen. Markieren Sie an diesen Stellen einen Bereich, können sie bequem Frequenz und Bandbreite ablesen. Über den Daumen gepeilt, läßt sich sagen, daß große Bandbreiten auf schnell an- und abschwellende Töne hindeuten und umgekehrt.

2. Filtern: Die Frequenztransformation bietet die ideale Möglichkeit zum Filtern, denn erstens können Sie sehen, wo etwas zu viel oder zu wenig ist, zweitens ist das was Sie im Spektrum herauslöschen (immer "Leeren", um Himmels Willen niemals ausschneiden, d.h. die Länge des Spektrums verändern) definitiv weg.

Sind Sie Netzbrummen in Ihrem Klang auf der Spur, suchen Sie um die 50 Hz und löschen diesen Bereich. Wollen Sie allgemein Rauschen eliminieren, leeren sie alle Bereiche zwischen den Peaks (kann man bequem mit Verstärker und Oszillator als Hüllkurve erreichen), denn dort befindet sich hörbares Rauschen, auch wenn man es im Spektrum kaum sieht.

Die ganzen Operationen haben dennoch einen Haken: Auch die Frequenztransformation arbeitet mit Qualitätsverlust, da auch sie nur mit 16Bit-Werten arbeitet :-(

3. Transponieren: Transponieren ohne Längenveränderung bewerkstelligt man, in dem man alle Peaks proportional verrückt. Resampling nützt nichts, die Peaks müssen erhalten bleiben und nur in Ihrer Position verschoben werden, und zwar um einen konstanten Faktor, so daß alle Peaks danach wieder gleichen Abstand zu einander haben. Auch das läßt sich bequem mit Phasenmodulation durchführen.

Für die beiden letzten Beispiele habe ich übrigens Algorithmen mitgeliefert.

Technik

Die Frequenztransformation wird über die Fouriertransformation abgewickelt. Auch hier sind die Begriffe bezüglich der Flexibilität schon schön von (herkömmlichen?) Programmen abgenutzt worden:

Programm beherrscht Fouriertransformation = es existieren eine oder ein paar mehr Routinen, die jeweils auf einen bestimmten Datenumfang (z.B. 256, 4096, jedenfalls Zweierpotenzen) fixiert und (hoffentlich) optimiert sind

Programm beherrscht Fouriertransformation für beliebigen Datenumfang = Routine transformiert Daten vom Umfang 2^k (k kleine natürliche Zahl)

Nun ist es aber so, daß Assampler die Daten in wirklich absolut frei wählbarem Datenumfang transformieren kann. Das Problem liegt hier: Die normale unoptimierte Fouriertransformation benötigt O(n^2) Zeit, das bedeutet: verdopple ich die zu transformierende Datenmenge bestehend aus n Samplewerten, vervierfacht sich die benötigte Rechenzeit. Praktisch bedeutet das: Bereits bei Datenaufkommen von vielleicht 10000 Samples (ist ja nicht viel, wenn man bedenkt, daß das einer Länge von einer halben Sekunde bei 20kHz entspricht) warten Sie ein halbe Ewigkeit auf das Ergebnis.

Optimierung

Schlaue Leute haben einen Algorithmus gefunden, der die Berechnung rapide beschleunigt, und haben ihn Schnelle Fouriertransformation genannt. Der Trick ist, wie so oft, das Teile-und-Herrsche-Prinzip, das heißt, daß die gesamte Transformation im wesentlichen in mehrere kleine Transformationen unterteilt wird. Das geht in diesem Falle aber nur bei exakten Teilungen, teilt man die Daten in z.B. drei nur ungefähr gleich große Teile, haut es nicht mehr hin.

Den Hauptgewinn hat man gezogen, wenn die Daten vom Umfang einer Zweierpotenz n=2^k sind, dann nämlich geht die Berechnung mit O(n log n) bzw. O(k*2^k) am schnellsten und die Programmierung am einfachsten. Deshalb helfen die meisten Programmierer dem Zufall nach und stutzen oder strecken die Daten auf die benötigte Länge (und behaupten in ganzen schlimmen Fällen noch, die Schnelle Fouriertransformation funktioniere nur für 2^k).

Es gibt zwei Gründe, warum ich mir für Assampler die Mühe für eine wirklich flexible und trotzdem schnelle Fouriertransformation gemacht habe: Der erste ist, daß das ganze Berechnungskonzept des Assamplers davon ausgeht, daß die Länge eines Klanges keine Rolle spielt, also schon gar nicht auf bestimmte Längen fixiert ist. Der zweite Grund ist theoretischer Natur, siehe nächsten Punkt.

Trotz daß der Assampler nun alle Datenlängen schluckt, ist es nicht zu empfehlen, daß in jedem Fall auszukosten, denn im schlimmsten Fall hat man wieder die Komplexität von n^2. Wie unterbindet man nun diesen Fall? Man kommt nach wie vor nicht um Stutzen oder Strecken der Daten herum, nur daß wir lange nicht so drastisch eingreifen müssen, denn von einer Zahl zur nächsten Zweierpotenz kann es sehr weit sein (maximal die relative Abweichung 1.414 = Sqrt 2), zur nächsten günstigen Primzahlzerlegung ist es dagegen meist nicht weit. Unser Problem ist also jetzt, den Datenumfang auf eine Zahl mit günstiger Primzahlzerlegung abzuändern, denn anhand der Primfaktoren wird das Gesamtproblem in Teilprobleme zerlegt, je kleiner die Primfaktoren desto besser.

Nun setzen Sie sich natürlich nicht hin und bestimmen Primzahlzerlegungen, dafür gibt es zum Glück schon die Funktionen RoundPrimeMax und RoundPrimeSum. Wie setzt man die nun ein?

1. Klang -> Spektrum

Die höchste Frequenz im Spektrum ist durch die Abtastfrequenz der Quelle festgelegt, daran können Sie nichts ändern. Bleibt die Auflösung (=SampleFreq) welche mit z.B. RoundPrimeMax (FT.Input0.Length, 5, 1/FT.Input0.SampleFreq) korrigiert werden kann. (FT sei der Name unseres Frequenztransformationsprozesses)

2. Klang -> Spektrum -> Modifikation -> Klang

Seien zwei Frequenztransformationsprozesse, einer zur Analyse und der andere zur Synthese, zwischen die z.B. ein Verstärker geschaltet ist. Dann dürfen der Quell- und der Zielklang der ganzen Berechnung identisch sein, die modifizierte Länge wird nur im Synthese-Prozess eingestellt, z.B. RoundPrimeMax (_.Length, 5, 1/_.SampleFreq). Die Länge des Analyse-Prozesses sollte auf _.SampleFreq gesetzt werden.

Theorie

Naturgemäß wird die Fouriertransformation mit komplexen Zahlen durchgeführt. Wie so oft, wird durch Anwendung der komplexen Zahlen vieles leichter (wenn man es einmal gegessen hat), und sie ermöglichen überhaupt erst die Schnelle Fouriertransformation. Diesen Vorteil reiche an Sie weiter, indem ich Ihnen die Einstellung sowohl von Realteil als auch Imaginärteil der Daten im FT-Prozess anbiete. Meistens werden Sie das nicht brauchen, wollen Sie aber die gleiche (Filter-)Operation auf zwei gleichlange Sounds anwenden, bekommen Sie so eine Filteroperation bei gleicher Rechenzeit praktisch geschenkt.

Die Realteile aller Werte befinden sich in einem Signal und die Imaginärteile an den entsprechenden Stellen im zweiten Signal. Sowohl die Eingabe als auch die Ausgabe bestehen auf diese Weise aus komplexen Zahlen.

Wollen wir noch ein paar Eigenschaften des Frequenzspektrums festhalten:

Ist der Imaginärteil des Quellklanges konstant null, das heißt der Klang ist rein reell, gilt:

1. Der Imaginärteil des Spektrums spiegelt die Amplituden der Sinusschwingungen wider, der Realteil des Spektrums die der Cosinusschwingungen.

2. Der Betrag des Spektrums (als Funktion mit komplexen Werten betrachtet, Berechnung mit absolute-Prozeß) ergibt die absoluten Amplituden aller enthaltenen Frequenzen.

3. Der Winkel des Spektrums (wie oben, Berechnung mit angle-Prozeß) gibt Aufschluß über die Phasenlage aller Frequenzen.

4. Negiert man im Spektrum eines Klanges den Imaginärteil, erhält man bei der Rücktransformation das zeitlich umgekehrte Signal.

Das funktioniert, weil die Sinusschwingungen (die durch den Imaginärteil dargestellt werden) ungerade Funktionen sind, d.h. negieren einer Sinusschwingung führt genau zum zeitlich um den Nullpunkt gespiegelten Signal. Cosinusschwingungen verändern sich beim Spiegeln jedoch nicht. Schlußfolgerung: Durch Negieren des Imaginärteils werden alle Frequenzbestandteile zeitlich umgekehrt, daher auch der zurückgewonnene Klang.

allgemein gilt:

1. Überlagert (addiert, mixt etc.) man zwei Signale, so addieren sich auch die Frequenzspektren (diese Eigenschaft nennt man Additivität).

2. Verstärkt (multipliziert) man ein Signal um einen konstanten (insbesondere komplexen) Faktor, geschieht das gleiche mit dem Spektrum.

3. Filtert man ein Signal (mathematisch: Faltung), führt das zum Aufprägen der Filterübertragungskurve auf das Spektrum.

4. Spezialfall Allpass: Verzögert man einen Klang (besser wäre rotieren = was hinten herausfällt wird vorne wieder hineingesteckt), verändert sich im Spektrum nur die Phase jeder Frequenz nicht aber ihre Amplitude.

5. Das Spektrum das Spektrums ist bis auf einen konstanten Faktor und der zeitlichen Umkehrung identisch zum Originalsignal.

Damit können wir auch begründen, warum das Stutzen von Daten sehr behutsam getan werden sollte: Das Kürzen oder Strecken ist ja sowas wie das Aufprägen einer rechteckigen Hüllkurve, die gerade dort 1 ist, wo die Daten bleiben und 0 dort, wo die Daten weggeschnitten werden. Im Spektrum erhalten wir dann also gefilterte (d.h. geglättete oder welligere, auf jeden Fall aber verfälschte) Daten.

Hinzu kommt, daß die (diskrete) Fouriertransformation einen Datensatz immer als Ausschnitt aus einem periodischen Vorgang, und zwar genau von der Länge der Periode, betrachtet. Bekommt man durch die rechteckige Hüllkurve starke Anstiege in den

Klang, müssen hohe Frequenzen im Spektrum auftauchen um das zu realisieren. Prägt man dem Spektrum dann zwecks Filterung eine Hüllkurve auf, fallen die hohen Frequenzen wahrscheinlich am ehesten weg, und es füllt sich im Zurücktransformierten der vorher freie Platz mit Geräusch an.

Geschichte

Ursprünglich sollte der Frequenztransformationsprozess die Kürzung der Datenmenge auf schnell berechenbare Zerlegungen eigenmächtig durchführen, weil ich dachte, daß Sie in diesem Falle ganz gerne auf die Transparenz des Verfahrens (verbunden mit dem Zwang sich mit der recht komplizierten Materie befassen zu müssen) verzichtet würden.

Nur leider hätten dann in den Zielsounds die Frequenzen auf der x-Achse nicht mit der Skalierung des Spektrums übereingestimmt und dann wäre es sehr ärgerlich gewesen, wenn Sie die Frequenztransformierte zur Bestimmung der Grundfrequenz einsetzen wollten.

1.154 Absoluter Betrag

Absoluter Betrag

aus der Klasse Frequenzspektrum

Funktion

Faßt jeweils korrespondierende Werte aller Eingänge zu einem Tupel zusammen und bestimmt davon die euklidische Norm, also die Quadratwurzel aus der Quadratsumme.

Spezialfälle

1 Eingang: Gleichrichter

2 Eingänge: Beträge von komplexen Zahlen

Werte

Array (Feld)

Array.Input (Klang)

Eingangssignal

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Volume (Formel) [y] = [Array.Input.y] * [Volume]

Verstärkungsfaktor

Technik

Optimierung

Besonders schnell geht es bei einem oder zwei Eingangssignalen.

1.155 Winkel der komplexen Zahlen

Winkel der komplexen Zahlen

aus der Klasse Frequenzspektrum

Funktion

Bestimmt Paare korrespondierender Werte in den Eingangssignalen und betrachtet diese als komplexe Zahlen. Davon wird das Argument bestimmt, also der Winkel zur positiven reellen Achse.

Das Ausgangssignal hat auf der y-Achse die maximale Auslenkung 180°.

Werte

Input0 (Klang)

Eingangssignal mit den Realteilen

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Input1 (Klang)

Eingangssignal mit den Imaginärteilen

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

1.156 Polar-zu-Komplex-Konvertierung

Polar-zu-Komplex-Konvertierung

aus der Klasse Frequenzspektrum

Funktion

Bestimmt Paare korrespondierender Werte in den Eingangssignalen und betrachtet diese als Polarkoordinaten. Von den so definierten Punkten kann man sich entweder die x- oder die y-Koordinaten ausgeben lassen.

Werte

Angle (Klang)

liefert die Winkel

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Component (Auswahl)

entscheidet zwischen Ausgabe von Real- und Imaginärteil

Input (Klang)

liefert die Beträge

Kopplung Lautstärke gekoppelt, Abtastrate gekoppelt, Beginn gekoppelt

Theorie

An sich ist der Prozeß überflüssig, da man ihn recht einfach nachbilden kann: Das Winkelsignal führe man in einen Abbildungsprozeß, der im Falle der x-Koordinate den Cosinus und im Falle der y-Koordinate den Sinus berechnet, das Ergebnis führe man zu einem Verstärker, der mit dem Betragssignal moduliert.

Aber da dieser Prozeß im Zusammenhang mit der Frequenztransformation häufiger gebraucht werden könnte, habe ich ihn vorsichtshalber fest verdrahtet.