Echounterdrückung

Autor: Andre Adrian, Brunnenstr. 48, 65618 Selters

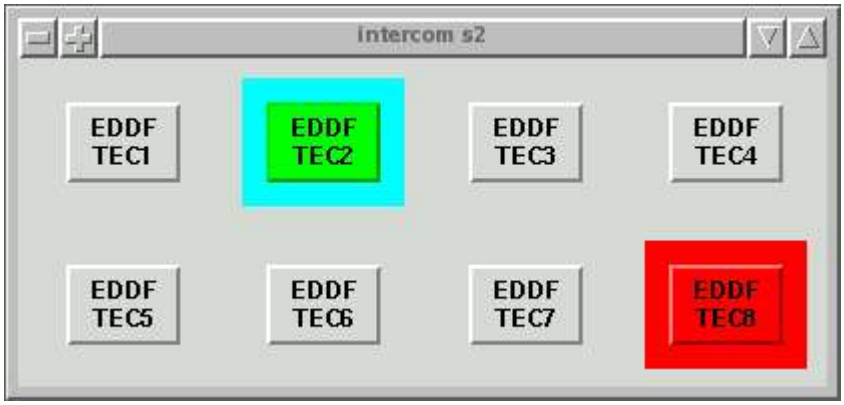
Tel tagsüber: 069 69766 176, FAX: 069 69766 175

E-Mail: adrianandre aT compuserve dOt de oder Andre aT Adrian aT dfs dOt de

Version: 24sep2005

Einleitung

„Wie heißt der Bürgermeister von Wesel“ - „Esel“. Dieses Spiel mit Echo kennt wohl jeder. Hören wir Echo bei unseren Telefongesprächen sind wir weniger amüsiert. Bei Freisprecheinrichtungen im Auto und besonders bei Voice-over-IP Freisprechtelefonen ist Echounterdrückung oder AEC (Acoustic Echo Cancellation) nötig. Wie AEC funktioniert, und wie man als Programmierer selbst AEC machen kann will der Artikel zeigen.



Die intercom Benutzeroberfläche ist sehr einfach. Aktuell wird mit der Station EDDF-TEC2 gesprochen, die Station EDDF-TEC8 ist ausgefallen.

Echoverzögerung

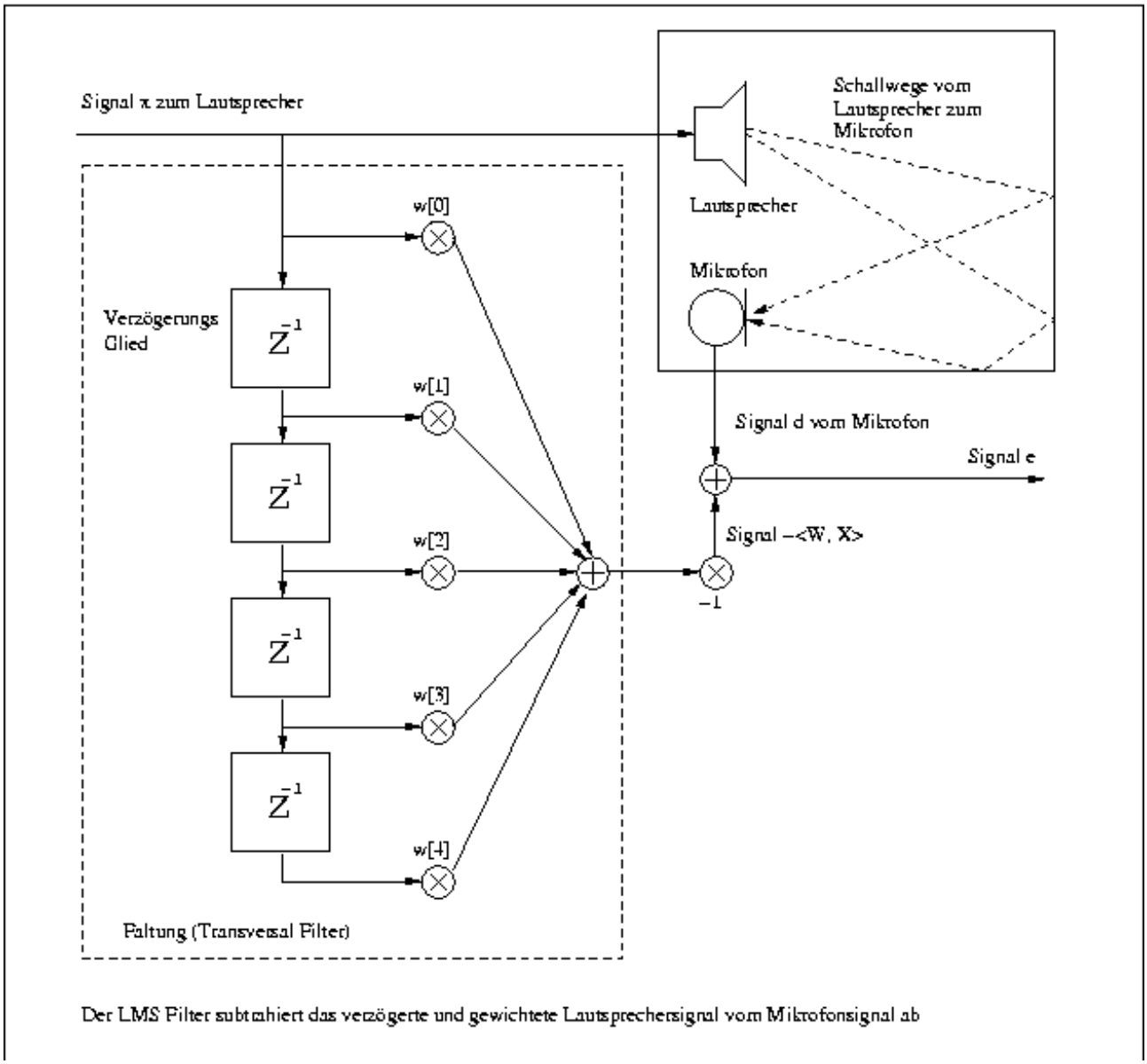
Eine Art von Echounterdrückung führen wir aus, seit wir sprechen können: wir können eine zweite Person hören, auch wenn wir gerade selbst sprechen. Diese Fähigkeit hat unser Gehirn für Echoverzögerungen bis 20ms (Millisekunden, Tausendstel Sekunden). Wird in einem Test die Echoverzögerung gesteigert, bemerken die Testpersonen zuerst einen kathedralenhaften, halligen Klang, dann störendes Echo. Der Kathedralen Effekt durch Echo wird im Musikstudio eingesetzt um Aufnahmen natürlicher klingen zu lassen. Siehe c‘t 8/2004 Audioeffekte: Hall mit Impulsfaltung.

Bei VoIP (Voice-over-IP) liegt die Echo-Laufzeit oder die zweifache Mund-zu-Ohr Übertragungszeit bei 100ms bis über 400ms. Die vom Programm ping gemeldete round-trip-time RTT ist die Daten-Transportzeit „einmal hin und wieder zurück“. Die VoIP Echo-Laufzeit ist ungefähr RTT plus 6 mal der Zeitabstand zwischen VoIP-Sprachpaketen. Mit RTT von 50ms und Sprachpaketen-Zeitabstand von 20ms ergibt sich eine Echo-Laufzeit von 170ms, bzw. eine einfache Mund-zu-Ohr Übertragungszeit von 85ms.

Ursache des Echos

Hörbares Echo bei Sprecher A entsteht weil auf der Seite von Sprecher B das Mikrofon den Schall des Lautsprechers aufnimmt. Kann nun dieser akustische Weg bei B blockiert werden, hört A kein Echo mehr. Die weiterhin bestehende Laufzeit stört uns Menschen nicht, solange diese unter 200ms einfacher Weg bleibt. Ein Headset (Kopfhörer-Mikrofon Garnitur) blockiert durch „den Kopf dazwischen“ den akustischen Weg. Bei einem Handset (Telefonknochen) kann durch Konstruktion und Dämmmaterial der akustische Weg von Hörmuschel zu Sprechmuschel durch den Handapparat blockiert werden – deshalb sind die Knochen von guten VoIP-Telefonen so wuchtig. Bei Freisprech-Telefonen muss der „akustische Kurzschluß“ mit elektronischen Mitteln bekämpft werden.

Hierzu wird vom elektrischen Mikrofonsignal das elektrische Lautsprechersignal abgezogen. Vor dieser einfachen Subtraktion muss das elektrische Lautsprechersignal aufwendig manipuliert werden. Und zwar genau so, wie der Raum das akustische Lautsprechersignal auf dem Weg zum Mikrofon verändert hat.



Der LMS Filter subtrahiert das verzögerte und gewichtete Lautsprechersignal vom Mikrofonsignal ab

Aufgrund der Schallgeschwindigkeit von 340m/s entsprechen 1ms akustische Laufzeit 34cm Schallweg. Schallwellen im Raum werden mehrfach zwischen Wänden, Decke und Boden hin- und her geworfen. Deshalb dauert die Impulsantwort (bis ein Knall verklingt) eines Raumes oft 100 Millisekunden und mehr. In schallarmen Räumen frisst der Wandbekleidung die Schallwellen schnell auf – man fühlt sich in diesen Räumen wie in Watte eingepackt.

Tapped-Delay-Line

Das Herzstück jeder Echounterdrückung ist die Verzögerungskette mit Abgriffen oder tapped-delay-Line. Die Verzögerungskette stellt die vergangenen Lautsprechersignale zur Verfügung, welche ja für die elektrische Nachbildung der akustischen Ausbreitung benötigt werden. Ein einzelnes Verzögerungselement (tap) entspricht einer Abtastung des Digital/Analog Wandlers. Für Telefonqualität gibt es 8000 Abtastungen pro Sekunde, für Wideband Qualität gibt es 16000 Samples pro Sekunde.

Eine Tapped-Delay-Line für 100ms Echokompensation benötigt bei Wideband 1600 Verzögerungselemente. Da jedes Sample mit der gesamten Tapped-Delay-Line verrechnet wird, entsteht ein ordentlicher Rechenaufwand für AEC. Bei Wideband muß 16000 mal pro Sekunde für jeweils 1600 taps die Echounterdrückungs-Rechnung ausgeführt werden. Bei einem Embedded Computer wie Intel Pentium-M mit 600MHz oder VIA Ezra mit 667MHz benötigt dies 23% der gesamten Rechenleistung.

Die Echounterdrückungs-Rechnung besteht aus der Faltung (Convolution) und der Anpassung (Update) der Filter-Koeffizienten. Die Faltung wird durch das Skalarprodukt (dot product) errechnet. Dabei enthält der Vektor X die Lautsprecher-Signale und der Vektor W die Koeffizienten. N steht für die Anzahl der Taps.

```
/* Skalarprodukt */
float dotp(float X[], float W[])
{
int j;
float sum0 = 0.0, sum1 = 0.0;
for (j = 0; j < N; j += 2) {
sum0 += X[j] * W[j];
sum1 += X[j + 1] * W[j + 1];
  }
return sum0 + sum1;
}
```

Aus mathematischer Sicht ist die Aufteilung in sum0 und sum1 nicht nötig. Es hilft aber den Rechenwerken in der CPU besser parallel zu arbeiten. Die SSE Befehle der Intel CPU erlauben die Verarbeitung von 4 Fließkommawerten in einem Assemblerbefehl. Durch Intrinsic Funktionen ist es beim Intel und beim GNU C/C++ Compiler leicht möglich mit SSE Kommandos zu arbeiten:

```
/* Skalarprodukt SSE Version */
#include <xmmintrin.h>
float dotp(float X[], float W[])
{
  int j;
  float sum[4];
  __m128 xmm0, xmm1, xmm2;
```

```
  sum[0] = sum[1] = sum[2] = sum[3] = 0.0;
  xmm2 = _mm_load_ps(sum);
  for (j = 0; j < N; j += 4) {
    xmm0 = _mm_loadu_ps(&X[j]);
    xmm1 = _mm_loadu_ps(&W[j]);
    xmm0 = _mm_mul_ps(xmm0, xmm1);
    xmm2 = _mm_add_ps(xmm2, xmm0);
  }
  _mm_store_ps(sum, xmm2);
  return sum[0] + sum[1] + sum[2] + sum[3];
}
```

Leider arbeitet die SSE Version nicht schneller bei aktuellen CPUs. Zwei Ursachen sind möglich: Die Übersetzung von Assemblerbefehle auf Mikro-Opcodes in der CPU nutzt in beiden Fällen die verfügbaren Rechenwerke gleichgut aus oder der Algorithmus ist datenflußbegrenzt, d.h. Die Daten können nicht schnell genug zwischen CPU und Arbeitsspeicher transportiert werden.

Die Faltung findet im Zeit-Bereich (time-domain) statt. Durch eine Fourier-Transformation ist es möglich die Faltung im Frequenz-Bereich (frequency-domain) mit weniger Rechenaufwand auszuführen. Per inverse Transformation wird das Ergebnis wieder zurückgerechnet. Der Quelltext des Open-Source Speex Codec (www.speex.org) von Jean-Marc Valin enthält eine Implementierung des Multidelay Block Frequency Adaptive Filter von Soo und Pang in der Datei mdf.c. Ob Frequency-Domain AEC Lösungen die Time-Domain Lösungen verdrängen, oder ob bei gleicher AEC Qualität der gleiche Aufwand bei beiden Ansätzen nötig ist, ist eine offene Frage. Wenigstens den älteren Frequency-Domain Lösungen wird schlechte Stabilität und langsame Anpassung an Änderung der Schallwege nachgesagt.

Least Means Square Algorithmus

Die Filter-Koeffizienten für die Faltung müssen irgendwie berechnet werden. Bei einem üblichen Finite-Impulse-Response Filter sind diese Koeffizienten Konstanten. Mit FIR Filtern lassen sich Tiefpässe, Hochpässe und Bandpässe bauen – Filter, die gewisse Tonhöhen besser durchlassen als andere Tonhöhen. Für Echounterdrückung müssen die Filter-Koeffizienten Variablen sein, welche sich an die unterschiedlichen Gegebenheiten anpassen. Wenn der Sprecher seinen Oberkörper bewegt, ändert sich der Schallweg und die Filter-Koeffizienten müssen sich anpassen oder adaptieren.

Die Herren Widrow und Hoff haben 1960 im Rahmen ihrer Künstlichen Intelligenz Forschung den LMS Algorithmus erfunden. Eigentlich soll der Adaline, so der ursprüngliche Name, das Verhalten einer Nervenzelle nachbilden. Der LMS Algorithmus optimiert, KI typisch, den Fehler. Der Fehler bei der Echounterdrückung ist der Unterschied zwischen dem echten Signal am Mikrofon wenn nur der Lautsprecher aktiv ist und dem nachgebildeten Signal welches die Faltung berechnet hat. „Wenn einer schweigt, der andere spricht dann nennt man dieses Adaline Unterricht“.

Der Fehler e wird berechnet aus Mikrofon Signal d minus Skalarprodukt, d.h. Mit

$e = d - \langle X, W \rangle$

Mit diesem Fehler e und einem weiteren Parameter mikro können jetzt die Koeffizienten adaptiert werden:

W[i+1] = W[i] + mikro*e*X[i]

Dabei steht W[i] für den alten Wert und W[i+1] für den neuen Wert des Koeffizienten-Vektors. Der Parameter mikro kontrolliert die Stabilität der Adaption. Bei einem Filter mit konstanten Koeffizienten ist die Stabilität leicht sicherzustellen. Nicht so bei der Echounterdrückung. Verliert der Filter die Stabilität hört man ein ekelhaft lautes Kreischen aus dem Lautsprecher, welches schlimmer als das ursprüngliche Problem, das Echo, ist.

Obwohl die LMS Formeln sehr einfach sind, ist die Sicherstellung der Stabilität sehr schwierig. Das Problem ist, neben den schon bekannten Werten gibt es keine weiteren. Münchhausen muß sich also am eigenen Schopf aus dem Sumpf ziehen. Seit 1965 bekannt ist die NLMS (Normalized LMS) Berechnung von mikro welche nur das aktuelle Skalarprodukt der Lautsprecher-Samples benötigt:

mikro = 1 / (<X, X> + delta)

Der Parameter delta begrenzt den Wertebereich von mikro und vermeidet eine Division durch Null. Der Wert von delta ergibt sich aus dem noise floor (Signal zu Rausch Abstand) und der Länge N des NLMS Filters in Taps. Typische Werte für SNR sind -55dB. Berechnung von delta:

delta = N * noisefloor * noisefloor

Pre-Whitening Filter

Die bis jetzt aufgebaute Echounterdrückung funktioniert prima mit weissem Rauschen als Lautsprechersignal. Betrachtet man die Frequenzverteilung von Sprache bemerkt man kräftige Tiefen und einen mehr oder minder gleichmäßigen Abfall zu hohen Tönen. Leitet man Sprache durch einen Hochpass erster Ordnung mit Eckfrequenz von 2 Kilohertz (2000 Schwingungen pro Sekunde) sieht die Frequenzverteilung schon deutlich mehr nach weissem Rauschen aus. Deshalb auch der Name Pre-Whitening Filter: wasche ein Signal bis es wie weisses Rauschen aussieht.

Natürlich soll das Weiss-Waschen dem LMS Filter helfen bessere Echounterdrückung auszuführen, aber nicht das Mikrofon Signal verzerren. Deshalb gibt es den Lautsprecher-Signal Vektor X in zwei Varianten: einmal ohne Weisswaschen für die Faltung, und dann weissgewaschen für die LMS Filter Adaption.

Ef = hochpass(e)

Xf = hochpass(X)

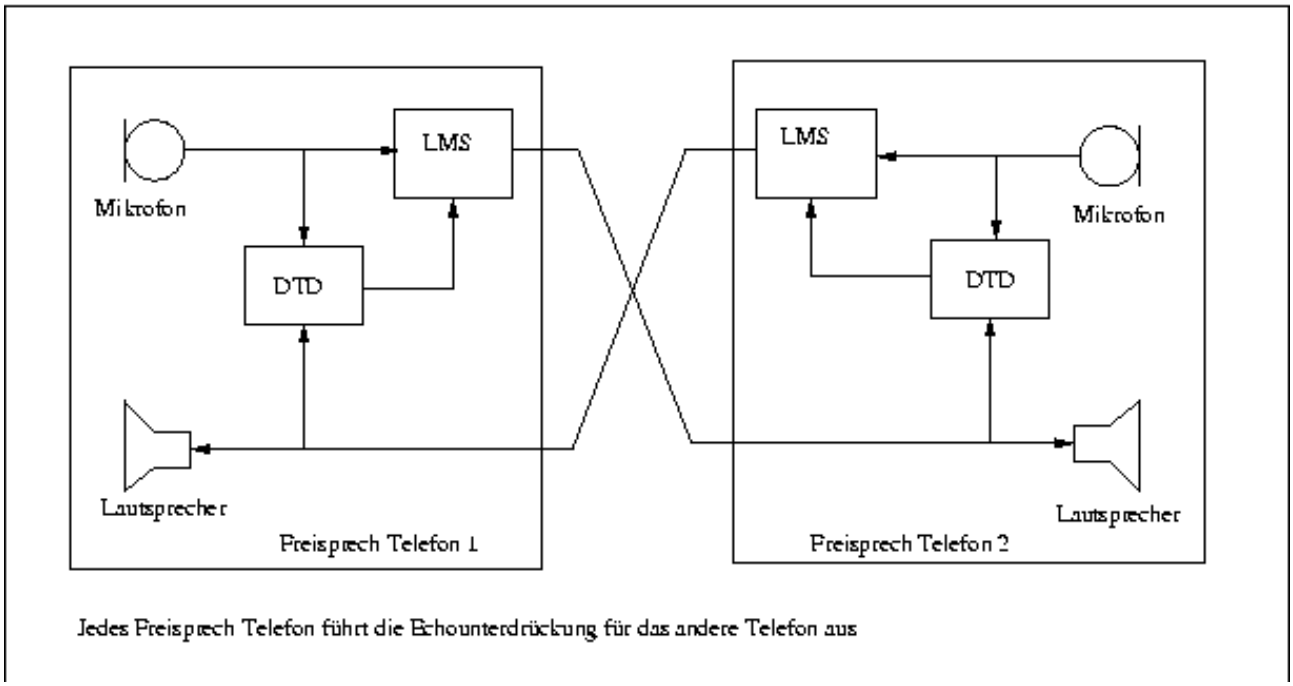
mikro = 1 / (<Xf, Xf> + delta)

W[i+1] = W[i] + mikro*ef*Xf[i]

Geigel Double-Talk-Detector

Nach Stabilität und Weiss-Waschen kommen wir nun zum Höhepunkt der Echounterdrückungs-Kunst. Alles wäre so einfach wenn sich die Nutzer an „einer schweigt, der andere spricht“ halten würden. Reden nun beide Nutzer gleichzeitig,

fehlt dem LMS Filter das Adaptions-Ziel. Das vom Mikrofon aufgenommene Signal stammt nun nicht nur vom Lautsprecher, sondern ein Störsignal ist überlagert. Dies ist auch der Grund warum Echounterdrückung in einer ruhigen Umgebung besser funktioniert. Fast jeder Gesprächspartner eines Automobil-Freisprech-Nutzers wird bestätigen, daß AEC geschwindigkeitsabhängig ist. Einfach weil mehr Geschwindigkeit mehr Fahrgeräusch bedeutet.



Jedes Freisprech Telefon führt die Echounterdrückung für das andere Telefon aus

Durch den Vergleich von Lautstärke des Mikrofon-Signals zu Lautstärke des Lautsprecher-Signals wird erkannt ob ins Mikrofon gesprochen wird. Wie beim ganzen Thema Echounterdrückung dürfen nicht die älteren Lautsprecher-Signale vergessen werden. Mit dem aktuelle Mikrofon-Signal d und den Lautsprecher-Signalen x ergibt sich die Double-Talk-Detector Formal nach Geigel:

$$|d| >= 0.5 * \max(|x[0]|, |x[1]|, .., |x[N-1]|)$$

Dabei bedeutet |d| der Betrag oder Absolutwert von d. Der Parameter 0.5 entspricht einer Dämpfung von 6 dB. Wird dieser Parameter verändert funktioniert Echounterdrückung entweder besser oder schlechter. Oft geschieht sogar beides: das Verhalten bei Single-Talk (nur einer spricht) wird besser, das Verhalten bei Double-Talk wird schlechter. Wird Double-Talk erkannt, so wird für eine hangover Zeit von z.B. 30ms die LMS Filter Adaption unterbunden. Die Echounterdrückung führt in dieser Zeit die Faltung aus, aber nicht die Anpassung.

Adrian Double-Talk-Detector

Der Geigel DTD hat Probleme wenn im Double-Talk Fall der eine Gesprächspartner leise, der andere laut redet: Die 6dB Schwelle wird nicht überschritten und Double-Talk wird nicht erkannt. Eine Idee aus der Börsen Chart-Analyse soll weiterhelfen. Wie bekannt werden 200-Tage und 30-Tage gleitende Mittelwerte der Aktienkurse gebildet. Wenn nun die „schnelle“ Kurve die „langsame“ Kurve überholt, dann sollen

Aktien gekauft oder verkauft werden. Anstelle von Aktienkurswerten haben wir bei der Echounterdrückung das Verhältnis von Mikrofon- zu Lautsprechersignal. Mit Exponential Smoothing Formeln lassen sich schnelle und langsame gleitende Mittelwerte von Mikrofon- und Lautsprecher-Signalen bilden:

```
dfast += 2e-3f * (fabsf(d) – dfast);
xfast += 2e-3f * (fabsf(x) – xfast);

dslow += 1e-6f * (fabsf(d) – dslow);
xslow += 1e-6f * (fabsf(x) – xslow);
```

Die vier gleitenden Mittelwerte können zu einem Verhältnis der Verhältnisse verrechnet werden:

```
ratio = (dfast / xfast) / (dslow / xslow);
```

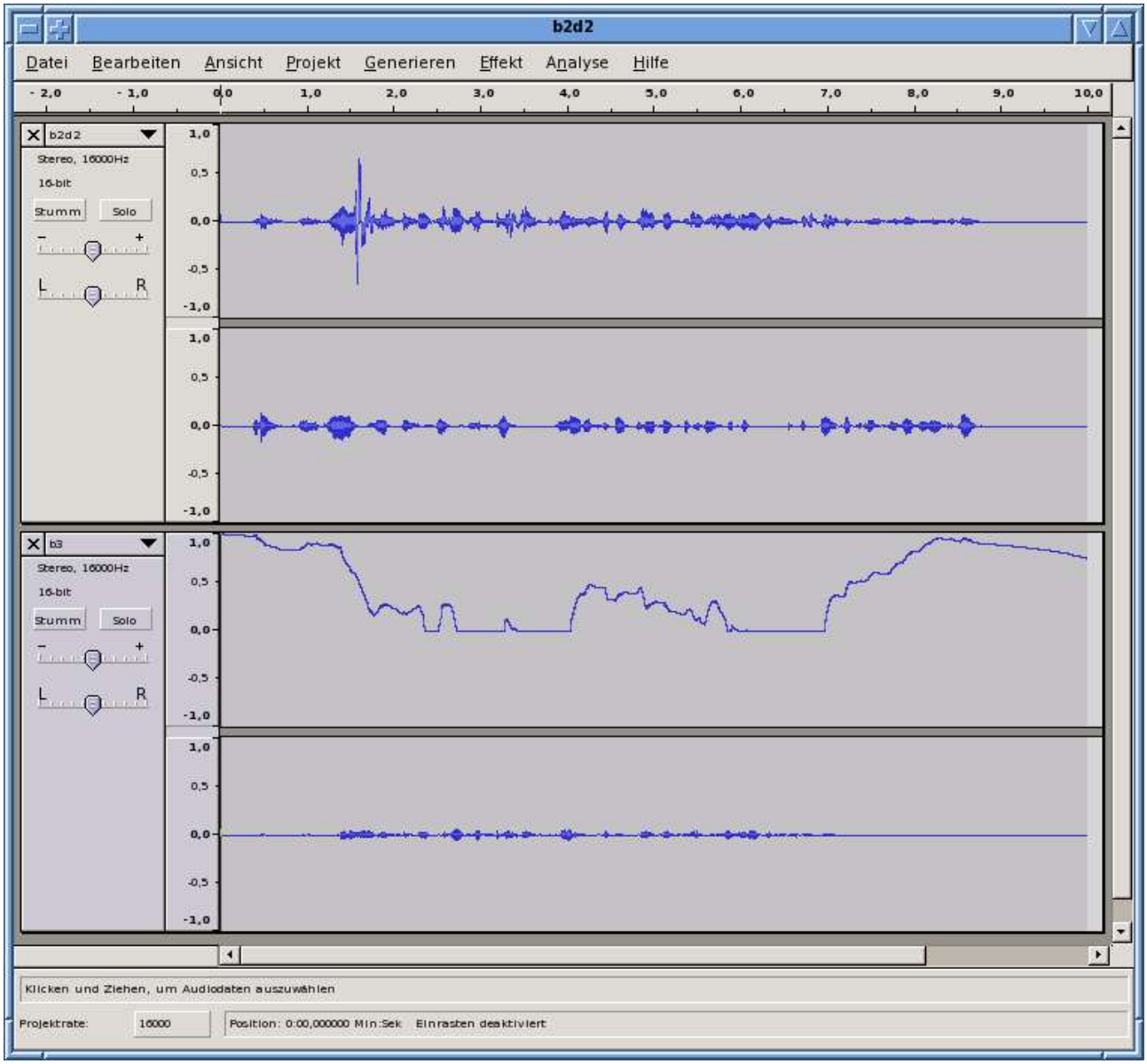
Bei Single-Talk liegt ratio bei 1. Je grösser ratio wird, umso mehr Double-Talk ist vorhanden. Mit ratio kann ein soft decision DTD realisiert werden: Anstelle einem Ein-/ Ausschalten der LMS-Filter Adaption kann mehr oder weniger Adaption ausgeführt werden. Aus der Konstanten 1 in der mikro Formel wird die Variable stepsize mit dem Wertebereich von 0 bis 1. Die Umrechnung von ratio auf stepsize erfolgt über eine Kennlinie. Dabei führen grosse Werte von ratio (viel Double-Talk) zu kleinen Werten von stepsize und kleine Werte von ratio zu grossen Werten von stepsize. Mit der Punktrichtungsform der Geradengleichung und Begrenzung gegen zu grosse oder zu kleine Werte ergibt sich:

```
// linker Punkt auf ratio nach stepsize Kennlinie const float X1 = 1.0,
Y1 = 1.0;
// rechter Punkt auf ratio nach stepsize Kennlinie const float X2 = 2.0,
Y2 = 0.0;
// Geradensteigung
const float M = (Y2 – Y1) / (X2 – X1);

if (ratio < X1) {
  stepsize = Y1;
} else if (ratio > X2) {
  stepsize = Y2;
} else {
  // Punktrichtungsform der Geraden
  stepsize = M * (ratio – X1) + Y1;
}
```

Die Formel für mikro im Fall des soft decision DTD lautet somit:

mikro = stepsize / (<Xf, Xf> + delta)



Double-Talk Test. Von oben nach unten wird Mikrofonsignal, Lautsprechersignal, Verlauf der Variablen stepsize und Signal e angezeigt (Bildschirmphoto Audio-Editor audacity).

Hardware

Für beste Echounterdrückung sollten die Lautsprecher- und Mikrofon-Signale Sample-synchron verarbeitet werden. Mit PC Hardware ist dies leichter gesagt als getan. Dank ALSA (Advanced Linux Sound Architecture) gibt es unter Linux folgenden Trick: Die Aufnahme arbeitet im Stereo-Modus. Der linke Kanal nimmt das Mikrofon-Signal auf, der rechte Kanal das Lautsprecher-Signal. Soundchips die hardware-mäßig dem AC97 Standard entsprechen, beherrschen diesen besonderen Stereo-Modus. Getestet wurde erfolgreich mit den Onboard Soundchips Analog Devices AD1985, ICEnsemble ICE1232, Realtek ALC650 und ALC655, SigmaTel STAC9750/51 und VIA VT1612A. Bei

den PCI-Soundkarten war der Test nur mit der heute nicht mehr lieferbaren Soundblaster PCI128 erfolgreich. Soweit bekannt funktioniert der Trick unter MS-Windows nicht. Die MS-Windows-API erlaubt nicht die Einstellung des speziellen Stereo-Modus. Wird das Mikrofon am linken Line-In Eingang der Soundkarte angeliefert, können MS-Windows Programmierer durch ein Kabel von Line-Out auf rechtes Line-In die sample-synchrone Anlieferung von Mikrofon- und Lautsprechersignal erreichen.

Der Toningenieur weiß, die Qualität fängt beim Mikrofon an, und hört bei schlechten Mikrofonen auch dort gleich wieder auf. Aus dem Billigsegment ist von den getesteten Mikrofonen nur das Labtec Verse 333 empfehlenswert (Labtec Verse 524 wurde nicht getestet). Für die semi-professionelle Nutzung wird eine Sennheiser ME34 Mikrofonkapsel mit MZH3040 Schwanenhals und Behringer MIC100 Mikrofonverstärker verwendet. Dieser Mikrofonverstärker versorgt das Kondensator-Mikrofon mit der Phantomspannung von 48 Volt, enthält einen Limiter und verstärkt das Mikrofonsignal auf Line-In Pegel. Dadurch wird der oft schlechte Mikrofonverstärker im PC umgangen, die übliche Schwachstelle Nummer 2 in der Audiokette. Als aktiver Lautsprecher wird Edirol MA-5A eingesetzt. Obwohl Preis und Aussehen sehr an PC-Brüllwürfel erinnern, hat der MA-5A schon ein wenig Studio-Monitor-Lautsprecher Qualität. Natürlich darf man keine Wunder bei dem kleinen Gehäuse erwarten, aber die Sprache klingt schön deutlich. Die Audio Hardware stammt von Musikhaus Thomann (www.thomann.de).

Als Entwicklung- und Test-PC wird beim Autor ein leiser Office-PC im Mini-Tower Gehäuse mit Intel 865 Mainboard und 2,6GHz Celeron CPU der Firma ICO (www.ico.de) benutzt. Als Zielhardware soll ein 3 Höheneinheiten (19-Zoll Technik) Touch-Input LCD PanelPC mit Via Ezra 667MHz CPU der Firma Delta Components (www.delta-components.de) eingesetzt werden. Der LCD hat 640x480 Pixel Auflöung bei 6,4 Zoll Diagonale. Betriebssystem wird SuSE Linux sein.

Software

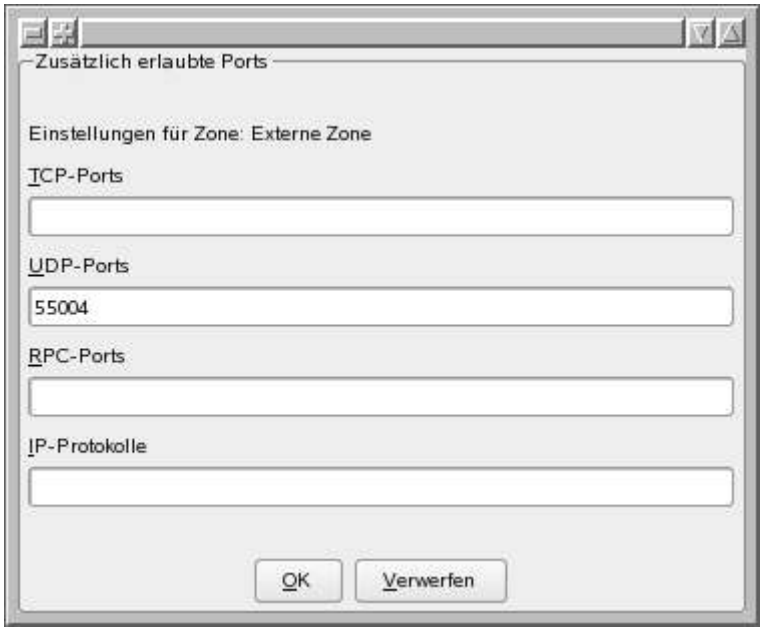
Die vorgestellten Funktionen stammen aus intercom, einer Open Source Voice-Over-IP Gegensprech Applikation. Neben Echounterdrückung ist auch Telefon-Konferenz mit maximal 6 Teilnehmern realisiert. Neben Telefonie Codecs wie G.711, G.726 und iLBC arbeitet intercom auch mit dem Speex Wideband Codec. Skype Sprachqualität ist dank Wideband möglich. Quelltext und RPM-Paket getestet mit x86 SuSE Linux 9.0 bis 9.3 liegen auf: http://home.arcor.de/andreadrian/intercom/

Als Programmiersprache wird C++ für den Audio Daemon intercomd und Tcl/Tk für das Benutzeroberflächen-Programm intercom benutzt. Beide Programme reden per TCP localhost Socket miteinander. Zum Transport der Sprachdatenpakete wird RTP (Real Time Protocol) nach IETF RFC3550 eingesetzt. Zur Ansteuerung der Audio-Hardware aus dem C++ Programm heraus wird das OSS (Open Sound System) API verwendet. Bei Open Sound ist die Audio-Blockgrösse beschränkt auf Werte von 2ms, 4ms, 8ms, ... (Zweierpotenzen). Die Codecs verlangen 20ms Blockgrösse. Deshalb wurde 4ms als Blockgrösse festgelegt. Damit stellt die Applikation eine hohe Anforderung an das Echzeitverhalten des Betriebssystems, welches der Linux Kernel ab Version 2.6 gut erfüllt. Mit dem Programm setpriority aus dem RPM Paket rtstools kann und sollte das Programm intercomd auf FIFO Realtime Prozess Priorität gestellt werden.

Seit kurzem wird die hier vorgestellte Echounterdrückung auch in dem SIP Softphone minisip eingesetzt. Siehe http://www.minisip.org/develop_build.html für die Anleitung um minisip unter Linux aus dem Subversion Repository zu bauen. Minisip wird unter GPL/LGPL Lizenz veröffentlicht.

Programmstart

Wird SuSE 9.3 eingesetzt, muss in der Firewall der UDP Port 55004 freigegeben werden, sonst ist die Signallisierung (Taste leuchtet gelb, magenta usw.) fehlerhaft. Über YaST, Sicherheit und Benutzer, Firewall, Erlaubte Dienste, Erweitert, UDP-Ports 55004 wird der Port freigeschaltet. Der UDP Port 5004 für Sprachübertragung (RTP) muss ebenfalls frei sein.



Bei SuSE 9.3 muss der UDP Port 55004 freigegeben werden.

Mit intercom wird das Programm gestartet. Beim ersten Programmstart erscheint die Meldung die Datei ~/.intercom.conf anzupassen. Dies erfolgt mit einem beliebigen Editor wie kate, gedit oder vi. Die Zeile guiconfig t1 "EDDF\nTEC1" 192.168.1.1 belegt die Taste t1 mit dem zweizeiligen Text EDDF TEC1. Nach Betätigung der Taste wird die IP-Adresse 192.168.1.1 angesprochen.

Die Programmoptionen sind -m für Mikrofonanschluss über Mic-In und -l für Mikrofonanschluss über Line-In, linken Kanal. Falls -m oder -l nicht funktioniert oder noch Echo hörbar ist, kann mit der Option -b und einem Dezibelwert der Echo-Suppressor aktiviert werden. Sinnvoll ist -b -24 oder -b -12. Ohne Option wird der iLBC Codec verwendet. Mit -A wird G.711 A-law, mit -U wird G.711 u-law und mit -S wird der Wideband Speex Codec eingesetzt. Die Option -t erlaubt Telefonkonferenz. Eine Telefonkonferenz entsteht spontan indem mehrere intercom Stationen einen Sprechweg zu einer Station aufbauen. Diese Station übernimmt das Mischen und Verteilen der Sprache damit jeder Teilnehmer jeden anderen Teilnehmer hören kann. Um Bandbreite zu sparen wird alle 80ms ein Sprachdatenpaket übertragen. Mit -f 1 wird alle 20ms übertragen.

Literatur

- Simon Haykin, Adaptive Filter Theory, 4. Edition, Prentice Hall, 2002
- Michael Hutson, Bachelor Thesis, Acoustic Echo Cancellation using Digital Signal Processing, University of Queensland, 2003, http://innovexpo.itee.uq.edu.au/2003/exhibits/s365914/