Linear Prediction Coder – Sprachkodierverfahren mit niedrigen Datenraten

Martin Schmidt und Marcel Wagner

Zusammenfassung

In dieser Arbeit werden Sprachkodierer für sehr niedrige Bitraten vorgestellt, die auf der Technik der lineraren Vorhersage beruhen. Zunächst wird ein Überblick über den theoretische Zusammenhang zwischen Spracherzeugung und linearer Vorhersage gegeben. Dann werden Experimente mit zwei konkreten Varianten dieser Art von Kodierern beschrieben, dem LPC-10 und dem CELP Kodierer. Abschließend wird die Integration einer TCP/IP Schnittstelle beschrieben.

1 Einleitung

Sprache ist in ihrer natürlichen Form ein analoges Audiosignal. Unter einem Audiosignal versteht man die zu Schall angeregte Luft, die vom menschlichen Ohr wahrgenommen werden kann. Durch Ausnutzung von perzeptiven Eigenschaften des Ohres kann ein Audiosignal pro Sekunde durch 44100 Abtastwerte mit 16 Bit pro Wert für einen menschlichen Zuhörer verlustfrei dargestellt werden. Man benötigt also eine Datenrate von ca. 700000 Bit/s.

Sprache hat jedoch als Ziel die Übermittlung von Informationen, d.h. es ist i.allg. nicht notwendig, Sprache mit sehr guter Qualität wiederzugeben. Um verständliche Sprache zu repräsentieren, reicht es, pro Sekunde 8000 Abtastwerte, 8 Bits pro Wert, zu verwenden, man kommt also mit 64kBit/s aus.

Sprachkodierverfahren sind Verfahren, deren Ziel es ist, Sprache in möglichst kompakter Form, d.h. mit geringeren Datenraten als 64kbit/s, zu repräsentieren. Ein Sprachkodierverfahren besteht aus einem Kodierer, der Daten transformiert und einem Dekodierer, der aus den kodierten Daten Sprache rekonstruiert. Sprachkodierverfahren sind durch Berücksichtigung der Spracherzeugungseigenschaften und Wahrnehmungseigenschaften des menschlichen Ohres effizienter als allgemeine Kodierverfahren (wie LZW oder Huffman - Coding). Einfache Sprachkodierverfahren (wie *differential pulse code modulation (dpcm)*) erreichen Kompressionsraten von 2-4, also Datenraten zwischen 16 und 32kBit/s.

In den Bereichen Mobilfunk und Internet Telefonie ist die Übertragungsbandbreite sehr stark eingeschränkt. Deshalb benötigt man Verfahren mit einer noch niedrigen Datenrate als 16kBit/s.

Die vorliegende Arbeit beschäftigt sich mit Verfahren, die Sprache bei einer sehr niedrigen Datenrate kodieren (2.4 - 4.8kBit/s). Um diese geringen Datenraten zu erreichen, wird Sprache auf Kodiererseite analysiert und als eine Folge von Parametern dargestellt, mit deren Hilfe auf Dekodiererseite Sprache synthetisiert wird.

Basis dieser Verfahren ist die lineare Vorhersage der Abtastwerte und die Ausnutzung vieler Eigenschaften der Spracherzeugung und -wahrnehmung. Die lineare Vorhersage realisiert die Signalformung mit einem *linearen Modell der Spracherzeugung* (auch Quelle-Filter Modell), auf dem alle Kodierer der vorgestellten Klasse basieren.

Diese Arbeit behandelt neben dem linearem Modell der Spracherzeugung und Theorie der linearen Vorhersage zwei konkrete Kodierverfahren für sehr niedrige Datenraten.

Zum einen ein Verfahren, welches als *pitch-excited linear prediction* bezeichnet wird. Es kodiert Sprache bei 8000Hz (8000 diskrete Abtastwerte pro Sekunde) mit 2400 Bit/s und hat damit eine Kompressionsrate von 26.67:1.

Das zweite Verfahren wird als *code-excited linear prediction (CELP)* bezeichnet. Es kodiert Sprache bei 8000Hz mit 4800 Bit/s und erreicht damit eine Kompressionsrate von 13.33:1.

Diese Arbeit wurde im Rahmen des Projekts *Adaptive techniques for image and image sequence coding* in der Arbeitsgruppe von Herrn Prof. Saupe im Institut für Informatik, Universität Freiburg, angefertigt. Das Ziel dieses Projekts ist die Entwicklung von Kodierverfahren (Bild und Ton) für Videokonferenzen mit Hilfe der adaptiven Vektorquantisierung.

2 Lineares Modell menschlicher Spracherzeugung

Viele Sprachkodierverfahren implementieren das lineare Modell der menschlichen Spracherzeugung, welches auch als Quelle-Filter Modell (*source-filter model*) bekannt ist. Auch die beiden in Abschnitt 4 vorgestellten Verfahren versuchen möglichst exakt Sprache diesem Modell nach zu synthetisieren. Dieser Abschnitt beschreibt die Abstraktion der realen menschlichen Sprachsynthese in diesem Modell, beginnend mit den Grundlagen zur menschlichen Spracherzeugung.

2.1 Menschliche Spracherzeugung

Die menschliche Spracherzeugung unterteilt sich im wesentlichen in zwei Abschnitte. Der erste Abschnitt besteht aus einer Schallanregung (Stimmgebung) des aus der Lunge strömenden Luftschwalls an den Stimmbändern. Der zweite Abschnitt ist die Signalformung (Resonanzbildung) durch den Vokaltrakt.

Als Vokaltrakt werden Rachen-, Mund- und Nasenraum bezeichnet.

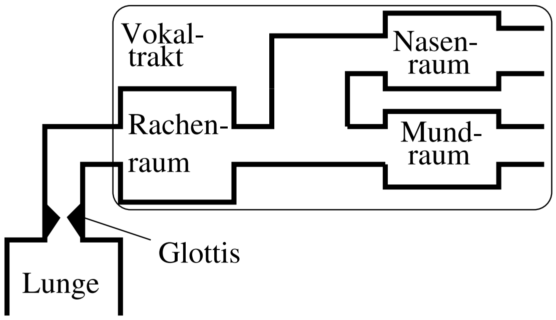

Abbildung 1:
Schematische Darstellung des menschlichen Artikulationssystems

Beim Prozess der Stimmgebung, der auch Phonation genannt wird, strömt die Luft durch Luftröhre und Bronchien zum Kehlkopf. Dort wird sie durch Passieren der Glottis, einer kleinen Öffnung, deren Größe von den Stimmbändern gesteuert wird, unterschiedlich angeregt. Diese angeregte Luft wird anschließend im Vokaltrakt zu einem Klang geformt. Sprache ist dann eine systematische Zusammensetzung dieser Klänge, wobei die Klangvielfalt selbst von den unterschiedlichen Anregungsformen abhängt.

Man unterscheidet in drei verschiedene Anregungsformen, wobei auf Mischungen, die insbesondere bei Übergängen zwischen unterschiedlichen Klängen auftreten, hier nicht näher eingegangen wird.

Stimmhafte Anregung Die stimmhafte Anregung dient zur Erzeugung von Schwingungen mit unterschiedlichen Frequenzen und wird mittels Öffnen und Schließen der Glottis (Spalte in den Stimmbändern) erreicht. Diese Anregung ist die Basis für Vokale (a,e,i,o,u).

Stimmlose Anregung (Reibe- oder Friktionsgeräusch) Bei stimmloser Anregung durchströmt die Luft bei geöffneten Stimmbändern eine Engstelle im Mund- oder Rachenraum, so daß dadurch eine turbulente Strömung entsteht, die als Rauschen wahrgenommen wird. Diese Anregung ist die Basis für Konsonanten (s,f,h, usw.).

Transiente Anregung (Plosivgeräusch) Transiente Anregung entsteht, wenn im supraglottalem System ein Verschluß gebildet wird, so daß der Luftstrom kurzzeitig unterbrochen wird und sich hinter der Verschlußstelle durch den subglottalem Druck ein Überdruck aufbaut. Das plötzliche Öffnen des

Verschlusses und die dadurch schnell entweichenden Luft dient als Anregung für Plosivgeräusche (k, g, z, usw.).

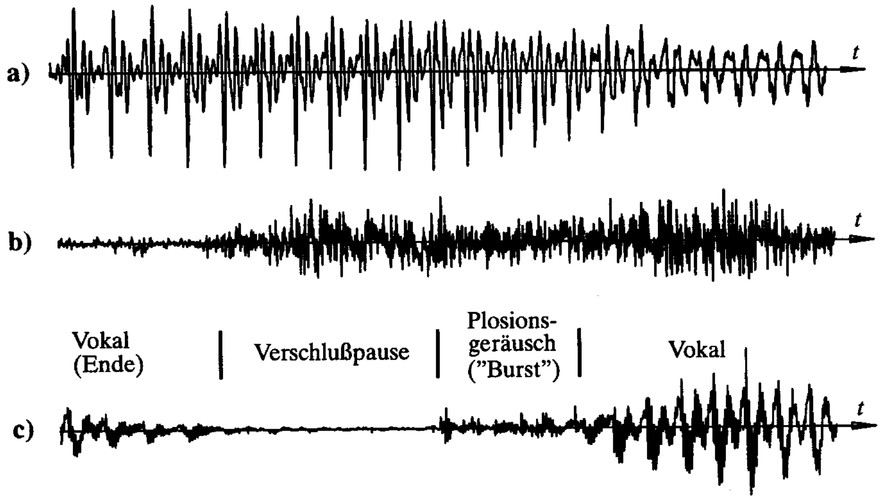

Abbildung 2:
Sprachsignalbeispiele für die einzelnen Anregungsarten (100ms/Zeile)
a) Stimmhaft (Vokal mit Übergang zu einem stimmhaften Konsonanten)
b) Stimmlos (Frikativ)
c) Übergang Vokal-Plosiv-Vokal

2.2 Lineares Modell der Spracherzeugung

Das lineare Modell der Spracherzeugung (auch Quelle-Filter Modell) ist eine naheliegende Abstraktion der menschlichen Sprachsynthese, betrachtet man die Unabhängigkeit von Stimmgebung durch die Stimmbänder und Resonanzbildung im Vokaltrakt. Abbildung 3 ist eine schematische Darstellung des Modells.

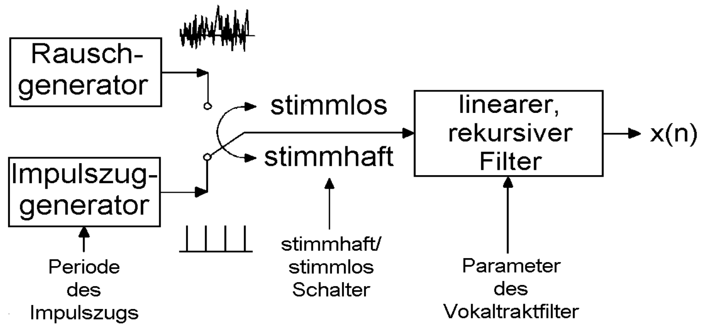

Abbildung 3:
Lineares Modell der Sprachsynthese

Sprache ist demnach die Reaktion eines linearen Filters (Modellierung des Vokaltraktes) auf ein zeitdiskretes Eingangssignal $u(n)$ (Modellierung des Anregungssignals).

Bei der Modellierung des Anregungssignals verzichtet das Modell auf die Erzeugung von transienter Anregung (Plosivgeräusche), d.h. es wird nur zwischen stimmhafter und stimmloser Anregung unterschieden. Bei stimmhafter Anregung wird mittels Impulszuggenerator ein Impulszug mit der Grundfrequenz $f_0 = \frac{1}{T_0}(T_0 = \text{Impulsperiode})$ erzeugt. Im Falle von stimmloser Anregung wird ein weißes Rauschen mit Varianz $\sigma_u^2 = 1$ erzeugt.

Betrachtet man einen festen Zeitpunkt n, dann läßt sich dieses Modell für das ausgehende Sprachsignal $x(n)$ folgendermaßen mathematisch beschreiben.

$$x(n) = \sum_{k=1}^{P} a_k x(n-k) + u(n) \tag{1}$$

Dabei ist das ausgehende Signal $x(n)$ die Summe aus dem Anregungssignal $u(n)$ und einer Linearkombination vergangener Sprachsignale $x(n-k)(k = 1, \ldots, P)$. Dem entspricht im Frequenzbereich die Übertragungsfunktion

$$X(z) = \frac{U(z)}{1 - A(z)} \tag{2}$$

die durch den Vokaltrakt-Filter

$$H(z) = \frac{1}{1 - A(z)}$$

charakterisiert wird. Der Vokaltrakt-Filter $H(z)$ besteht selbst im wesentlichen aus dem Koeffizientenfilter

$$A(z) = \sum_{k=1}^{P} a_k z^{-k} \tag{3}$$

Viele Sprachkodierverfahren basieren auf diesem Modell, wie auch die beiden in Abschnitt 4 vorgestellten Verfahren pitch-excited LP und CELP. Beide stützen sich dabei auf die Theorie der linearen Vorhersage, um die erforderlichen Parameter zur Steuerung des Vokaltraktfilters und zur Erzeugung des Anregungssignals zu erhalten.

3 Lineare Vorhersage

Die lineare Vorhersage definiert eine Klasse von Kodierverfahren, welche eine Eigenschaft von autoregressiven Systemen (wie z.B. Sprache) ausnutzen. Diese Eigenschaft ist die Tatsache, daß in autoregressiven Systemen, ein Signal i.allg. nicht vollständig unabhängig von bisherigen Signalen ist. Die lineare Vorhersage versucht auf Dekodiererseite das neue Signal auf Basis vorausgegangener Signale vorherzusagen, d.h. die kodierte Sprache besteht aus Steuerungsparameter für den Dekodierer.

Der Zusammenhang zwischen linearer Vorhersage und Sprachkodierverfahren wird näher in Abschnitt 4 beleuchtet. In diesem Abschnitt soll der mathematische Hintergrund zur linearen Vorhersage, insbesondere auch der Minimierung des durch die Vorhersage entstehenden Fehlers erläutert werden.

3.1 Die Theorie der linearen Vorhersage

Bei der linearen Vorhersage geht man davon aus, daß ein zeitdiskretes Signal $x(n)$ durch eine Linearkombination seiner P Vorgänger vorhergesagt werden kann. Dies läßt sich mathematisch durch folgende Gleichung

$$\tilde{x}(n) = \sum_{k=1}^{P} b_k x(n-k) \tag{4}$$

darstellen.
Im Frequenzbereich (z-transformiert) ist diese Gleichung darstellbar durch

$$\tilde{H}(z) = \frac{1}{1 - P(z)} \tag{5}$$

wobei

$$P(z) = \sum_{k=1}^{P} b_k z^{-k}$$

das sogenannte Prädiktorfilter darstellt. Die $b_k (k = 1..P)$ werden Prädiktorkoeffizienten genannt, P die Prädiktorordnung.
Betrachtet man einen festen Zeitpunkt n, so lassen sich Koeffizienten finden, so daß der Fehler

$$\epsilon(n) = x(n) - \tilde{x}(n) = x(n) - \sum_{k=1}^{P} b_k x(n-k) \tag{6}$$

gleich null ist.

Dies ist bei festen Koeffizienten für einen kurzen Zeitbereich von N Signalen i.allg. nicht der Fall, so daß sich für den i-ten Abschnitt in einer Signalfolge ein kummulierter Fehler folgendermaßen ergibt

$$\begin{aligned} E_i &= \sum_{n=0}^{N-1} (\epsilon_i(n))^2 \\ &= \sum_{n=0}^{N-1} (x_i(n) - \tilde{x}_i(n))^2 \\ &= \sum_{n=0}^{N-1} (x_i(n) - \sum_{k=1}^{P} x_i(n-k))^2 \end{aligned} \tag{7}$$

Die Definition von x_i führt zu unterschiedlichen Berechnungsmethoden, auf die weiter unten eingegangen wird (siehe Kovarianzmethode und Autokorrelationsmethode).

Die Koeffizienten lassen b_k dadurch errechnen, daß man die Ableitungen von E_i nach b_j gleich null setzt ($\partial E_i / \partial b_j = 0$) und das dadurch entstehende lineare Gleichungssystem von P Gleichungen löst. Es ergeben sich die P Gleichungen $(k = 1, \ldots, P)$

$$\begin{aligned} 0 &= \tfrac{\partial E_i}{\partial b_j} \\ \Leftrightarrow \quad \textstyle\sum_{k=1}^{P} b_k \sum_{n=0}^{N-1} x_i(n-j) x_i(n-k) &= \textstyle\sum_{n=0}^{N-1} x_i(n) x_i(n-j) \end{aligned}$$

Durch geeignete Substitution mit

$$\varphi_i(j,k) = \sum_{n=0}^{N-1} x_i(n-j) x_i(n-k)$$

lassen sich diese Gleichung vereinfachen zu

$$\sum_{k=1}^{P} b_k \varphi_i(j,k) = \varphi_i(j,0) \tag{8}$$

bzw. in Vektor-Matrix-Darstellung

$$\Phi_i \vec{b} = \vec{\varphi}$$

Durch Berechnung der Werte $\varphi_i(j,k)$ für $0 \leq j \leq N-1$ und $1 \leq k \leq P)$ erhält man resultierend ein lineares Gleichungssystem mit P Gleichungen, welches mittels Methoden der Numerik lösbar ist.

Betrachtet man die Gleichungen (6) und (7) genauer, stellt man fest, daß bei der Berechnung auch Werte außerhalb eines Abschnitts benötigt werden, d.h. x_i muß für die Werte $-P \leq n \leq N-2$ definiert sein.

Zwei bekannte Möglichkeiten x_i zu wählen, sollen hier kurz vorgestellt werden. Es handelt sich zum einen um die sogenannte Autokorrelationsmethode (*autocorrelation*) und zum anderen um die Kovarianzmethode (*covariance*).

3.1.1 Autokorrelationsmethode

Bei der Autokorrelationsmethode werden die Werte außerhalb des i-ten Abschnitts für die Berechnung ausgeblendet, d.h. gleich 0 gesetzt. Die Werte innerhalb des i-ten Abschnitts werden mit einer Fensterfunktion $w(n)$ multipliziert. Üblicherweise ist $w(n)$ ein Hamming-Fenster, d.h. Werte am Rande des Abschnitts werden abgeflacht, um auftretenden Fehler durch die Einbeziehung von Nullwerten außerhalb des Abschnitts weniger stark zu bewerten.

$$x_i(n) = \begin{cases} x(n+iN)w(n), & 0 \leq n \leq N-1 \\ 0, & sonst \end{cases}$$

Bei dieser Wahl von x_i erhält man eine symmetrische Toeplitz Matrix. Damit läßt sich Gleichung (6) sehr effizient mit dem rekursiven Algorithmus nach Durbin lösen, welcher die Eigenschaften einer Toeplitz Matrix ausnutzt ([1], S.92).

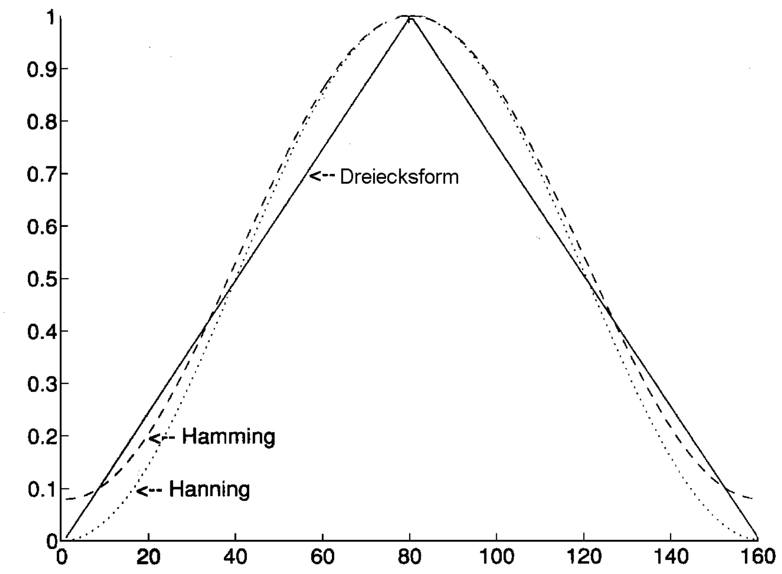

Abbildung 4:
Von LPC verwendete Fensterfunktionen (Autokorrelationsmethode), bei einer Abschnittslänge von 160 Signalen

3.1.2 Kovarianzmethode

Bei der Kovarianzmethode werden die Werte außerhalb des i-ten Frames in die Berechnung integriert, d.h.

$$x_i(n) = x(n + iN), \quad -P \leq n \leq N - 1$$

Bei dieser Wahl von x_i erhält man zur Berechnung des Prädiktorkoeffizienten eine symmetrische Matrix Φ_i mit

$$\Phi_i[j,k] = \sum_{k=1}^{N-1} x(n - j + iN)x(n - k + iN)$$

Auch für die Lösung von linearen Gleichungsystemen mit symmetrischer Koeffizientenmatrix existieren effiziente Algorithmen. Dennoch wird i.allg. die Autokorrelationsmethode zur Berechnung der Prädiktorkoeffizienten eingesetzt, auch wenn diese modifizierte Werte verwendet (Fensterfunktion): Die Verwendung der Autokorrelationsmethode garantiert im Gegensatz zur Kovarianzmethode stabile Vokaltraktfilter([1], S.94).

4 Linear Prediction Coder (LPC)

Verfahren, die mittels der *Linearen Vorhersage* Sprache kodieren, nennt man Linear Prediction Coder. Zwei Verfahren dieser Klasse von Sprachkodierverfahren sollen hier nun vorgestellt werden. Sie unterscheiden sich im wesentlichen nur in der Wahl des Anregungssignals und implementieren die Modellierung des Vokaltrakts durch einen Prädiktorfilter:

In Abschnitt 2 wurde das lineare Modell der Spracherzeugung vorgestellt:

$$X(z) = \frac{U(z)}{1 - A(z)}$$

Dabei war $U(z)$ die Z-Transformation des Anregungsignals und

$$H(z) = \frac{1}{1 - A(z)}$$

der Vokaltraktfilter.
Vergleicht man den Koeffizientenfilter des Modells

$$A(z) = \sum_{k=1}^{P} a_k z^{-k}$$

mit dem Prädiktorfilter

$$P(z) = \sum_{k=1}^{P} b_k z^{-k}$$

so ist eine große Ähnlichkeit zu bemerken. Tatsächlich läßt sich nachweisen, daß die Prädiktorkoeffizienten b_i eines Signals $x(n)$, welches durch die Anregung mittels weißen Rauschens oder eines Einheitsimpuls entstanden ist, identisch mit den Koeffizienten des Vokaltraktfilters im Modell sind, d.h. der durch die Prädiktion entstehende Fehler stimmt mit der Anregung überein.

Ist das lineare Modell der Spracherzeugung eine gute Approximation der menschlichen Spracherzeugung, so läßt sich Sprache mittels Erzeugung des Anregungssignals und anschließender Formung durch den Vokaltraktfilter (im Falle der linearen Vorhersage, dem Prädiktorfilter) synthetisieren.

4.1 Pitch-excited linear prediction

Das Sprachkodierverfahren, welches das Anregungssignal im stimmhaften Fall als periodisches Impulssignal bzw. im stimmlosen Fall als weißes Rauschen erzeugt (*pitch-excited linear prediction*), ist eine sehr genaue Nachbildung des linearen Modells der Spracherzeugung.

Auf Transmitterseite werden die ankommenden zeitdiskreten Signale zunächst in Abschnitte (*frames*) unterteilt und für jeden Abschnitt werden die zur Erzeugung der Sprache benötigten Parameter ermittelt. Dazu gehören zum einen die Parameter für den Vokaltraktfilter, welche mittels der linearen Vorhersage analysiert werden. Zum anderen gehören dazu die Parameter zur Erzeugung des Anregungssignals, d.h. Energie des Abschnitts, die Entscheidung ob der Abschnitt stimmhaft oder stimmlos ist und im stimmhaften Fall die Impulsperiode.

Zur Bestimmung der Parameter des Vokaltraktfilters wird auf Abschnitt 3 verwiesen. An dieser Stelle wird die Analyse zur Bestimmung der Parameter zur Erzeugung des Anregungssignals vorgestellt.

4.1.1 Berechnung der Energie

Die Energie eines diskreten Signals für einen Abschnitt berechnet sich durch

$$E = \sum_{k=0}^{N-1} (x(n))^2$$

Damit läßt sich der Faktor errechnen, der zur Multiplikation des Anregungssignal benötigt wird, damit dieses die gleiche Energie besitzt. Möchte man vermeiden, daß große Werte aufgrund der Quadrierung den Wert über ein Segment zu stark beeinflussen, so kann man auch auf die Funktion

$$E = \sum_{k=0}^{N-1} |x(n)|$$

zurückgreifen.

4.1.2 Entscheidung, ob ein Segment stimmlos bzw. stimmhaft ist

Die Entscheidung, ob ein Zeitabschnitt als stimmlos oder stimmhaft gekennzeichnet wird, läßt sich auch über die Berechnung der Energie feststellen. Ein stimmhafter Abschnitt besitzt eine wesentlich höhere Energie als ein stimmloser Abschnitt.

Die Entscheidung zwischen stimmlos und stimmhaft nimmt für die Sprachsynthese einen hohen Stellenwert ein. Wird eine falsche Anregung gewählt, so weist das erzeugte Sprachsegment i.allg. keine Ähnlichkeit mit der Realität auf, ein periodisches Segment wäre aperiodisch und umgekehrt. Der Klang wäre nicht mehr korrekt. Deshalb zieht man als zusätzliches Entscheidungskriterium die Anzahl der Nulldurchgänge hinzu. Ein stimmhafter Abschnitt hat wesentlich weniger Nulldurchgänge als ein stimmloser (siehe Abbildung 2). Ein Nulldurchgang zählt genau dann, falls gilt

$$sign(x(n)) \neq sign(x(n+1))$$

4.1.3 Ermittlung der Impulsperiode (stimmhafter Fall)

Es gibt verschiedene Möglichkeiten, die Impulsperiode zu ermitteln. Eine effiziente Möglichkeit ist die sogenannte AMDF (*average magnitude difference function*).

$$AMDF(\lambda) = \sum_{k=0}^{N-1} |x(k) - x(k+\lambda)|$$

Diese Funktion hat ein deutliches Minimum, falls λ mit der Grundperiodendauer T_0 übereinstimmt.

4.1.4 Federal Standard 1015 (FS1015)

Der Federal Standard 1015 (FS1015) - auch bekannt als LPC-10e - ist ein Standard für pitch-excited LP und wurde vom Departement of Defense (DoD) der Vereinigten Staaten herausgegeben. Der Standard legt die Größen für die Parameter zur Erzeugung und Analyse der Sprache (auf Empfänger- und Senderseite) fest.
Die Parameter wurden wie folgt festgelegt:

Abschnittslänge	20ms (160 Signale)
Prädiktorordnung	10

4.2 Code-excited Linear Prediction (CELP)

Das sogenannte Code-excited Linear Prediction ist ebenfalls ein Verfahren, welches als Kern einen linearen Prädiktor besitzt, dessen Prädiktorkoeffizienten abschnittsweise (i.allg. 240 Signale) berechnet und übermittelt werden. Es unterscheidet sich vom pitch-excited LP nur durch die Wahl des Anregungssignals.

Bei CELP wird bei der Berechnung des Anregungssignals ein Abschnitt (*frame*) in mehrere Unterabschnitte (*subframe*) geteilt und für jeden Unterabschnitt ein optimales Anregungssignal ermittelt. CELP besitzt als Auswahl des Anregungssignals Vektoren, deren Dimension genau der Länge eines Unterabschnitts entspricht. Jeder Vektor steht auf Kodierer- und Dekodiererseite zur Verfügung, so daß nur der Index des Vektors und Verstärkungsfaktor übermittelt werden.

Der optimale Anregungsvektor wird mittels Vektorquantisierung festgestellt, wobei bei der Minimierung des Fehlers ein Maß verwendet wird, welches die perzeptiven Eigenschaften des Ohres ausnutzt (*perceptually-weighted error function*). Dazu muß Sprache auf Basis eines Anregungsvektors zunächst künstlich erzeugt werden, um anschließend künstliche und reale Sprache entsprechend dem Maß zu vergleichen. Verfahren, die zur Bestimmung des besten Anregungssignals auf Kodiererseite Sprache künstlich erzeugen und mit dem realen Sprachsignal vergleichen nennt man Analyse-durch-Synthese (*analysis-by-synthesis*) Verfahren.

4.2.1 Das Modell der Anregung

Das Anregungssignal beim CELP-Kodierer ergibt sich als Summe zweier Vektoren aus zwei verschiedenen Codebüchern, einem statischem Codebuch und einem adaptiven Codebuch.

Das statische Codebuch enthält K_s Vektoren, die nach einem festen Algorithmus erzeugt werden, welcher Kodierer- und Dekodiererseite bekannt ist. Damit ergibt sich die Menge der Vektoren als

$$\mathcal{F}_s = \{f_\gamma | \gamma = 1, \ldots, K_s\}$$

Das adaptive Codebuch enthält K_a Vektoren, die aus dem bisherigen Anregungssignalen resultieren, beginnend beim vergangenen Unterabschnitt (Länge $= d$ Signale) und den Anregungsvektoren jeweils um ein Signal verschoben, d.h. ist $e(n)$ das Anregungssignal, dann ergibt sich die Menge der Vektoren aus dem adaptiven Codebuch als

$$\mathcal{F}_a(n) = \{e(n-\gamma) | \gamma = d, \ldots, (d + K_a)\}$$

Das Anregungssignal $e(n)$ ergibt sich dann als

$$e(n) = \beta_a e(n - \gamma_a) + \beta_s f_{\gamma_s}$$

wobei γ_a und γ_s die Indizes des optimalen Anregungsvektors in adaptiven und statischen Codebuch sind und (β_a, β_s) das Paar der Verstärkungsfaktoren.

Perceptually weigthed error function Innerhalb der Analyse-durch-Synthese Schleife wird zur Verbesserung der Qualität nicht der Fehler zwischen künstlicher und realer Sprache als Maß verwendet, sondern ein Fehlermaß, welches die perzeptiven Eigenschaften des menschlichen Ohres ausnutzt.

Die Idee dabei ist, das Spektrum des Rauschsignals an das Spektrum des Sprachsignals anzupassen, so daß die wahrgenommene Lautstärke des Rauschens durch das Sprachsignal überdeckt wird und in den Frequenzbereichen mit wenig Energie nicht mehr oder weniger wahrgenommen wird.

Erreicht wird dies mit dem gewichteten Filter

$$W(z) = \frac{1 - A(z)}{1 - A(\frac{z}{\alpha})} = \frac{1 - A(z)}{1 - F(z)}$$

mit der Gewichtskonstante $0 \leq \alpha \leq 1$.

Abbildung 5 zeigt den schematischen Aufbau eines CELP-Kodierers mit Einsatz des gewichteten Filters $W(z)$ (Zur Vereinfachung ohne Langzeitprädiktion).

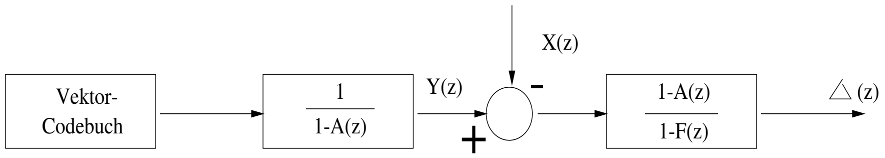

Abbildung 5:
Schematischer Aufbau eines CELP-Kodierers (ohne Langzeitprädiktion).

Der CELP-Kodierer realisiert gleichzeitig einen CELP-Dekodierer (*Analyse-durch-Synthese Prinzip*) mit dem Ausgangssignal $Y(z)$. $\Delta(z)$ ist bei diesem Aufbau der zur Minimierung verwendete, spektral gewichtete Fehler. Für das Ausgangssignal $Y(z)$ auf Dekodiererseite (bei Verwendung des optimalen Vektors) gilt:

$$\begin{aligned} \Delta(z) &= (Y(z) - X(z))\tfrac{1-A(z)}{1-A(\frac{z}{\alpha})} \\ \Leftrightarrow \quad Y(z) &= X(z) + \Delta(z)\tfrac{1-A(\frac{z}{\alpha})}{1-A(z)} \end{aligned}$$

$\Delta(z)$ weist i.allg. keine Korrelation auf. Zur Vereinfachung kann man annehmen, daß $\Delta(z)$ ein gleichverteiltes Rauschen.

Gilt $\alpha = 1$, so wird der Fehler am Ausgang des Dekodierers nicht gewichtet ($W(z) = 1$) und tritt als gleichverteiltes Rauschen auf.

Gilt $\alpha = 0$, so ist $W(z) = 1 - A(z)$, d.h. in diesem Fall wird das weiße Rauschen gleich dem Anregungssignal geformt.

Durch verschiedene Werte von α kann man zwischen diesen beiden Extremen variieren. Den Einsatz von $0 < \alpha < 1$ nennt man spektrale Färbung.

Die wirksame spektrale Färbung des Fehlers wird in folgender Abbildung verdeutlicht. Im logarithmischen Maßstab sind dargestellt das Betragsquadrat des Synthesefilters (Linie) und der Verlauf des Leistungsdichtespektrums des wirksamen Fehlers (gestrichelte Linie).

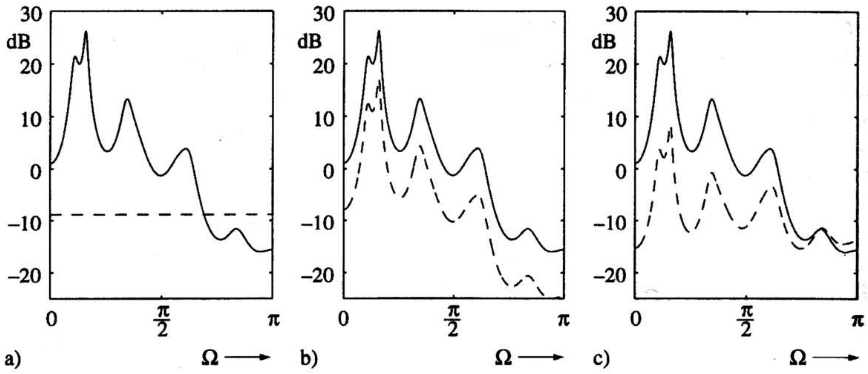

Abbildung 6:
Spektrale Formung des Filters am Beispiel des Vokals 'o'
a) Rückwärtsprädiktion ($\alpha = 1$)
b) Vorwärtsprädiktion ($\alpha = 0$)
c) Rauschfärbung ($\alpha = 0.7$)

Die Wirkung dieser Parameterwahl läßt sich auch anhand der Nullstellen z_{0i} der Übertragungsfunktion des Analysefilters erklären. Betrachtet man $1 - A(z)$ in der Produktform

$$1 - A(z) = \frac{1}{z^n} \prod_{i=1}^{n} (z - z_{0i})$$

so folgt für $1 - A(\frac{z}{\alpha})$

$$1 - A(\frac{z}{\alpha}) = \frac{1}{z^n} \prod_{i=1}^{n} (z - \alpha z_{0i})$$

Durch die Variation des Parameters α werden die Nullstellen des Filters

$$1/W(z) = \frac{1 - A(\frac{z}{\alpha})}{1 - A(z)}$$

(auf Dekodiererseite) zwischen Ursprung und Polstellen skaliert (siehe Abbildung 7).

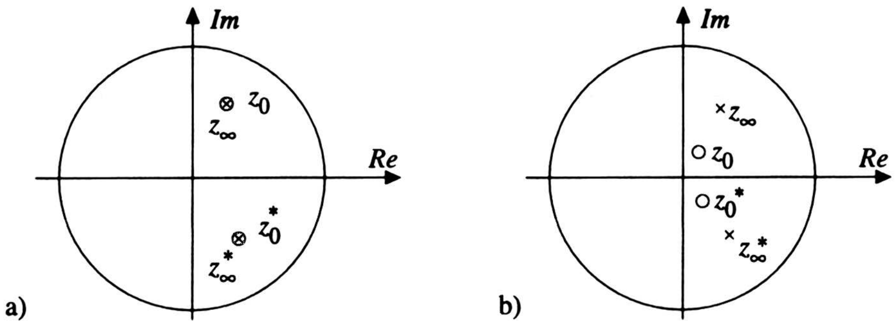

Abbildung 7:
a) Nullstellen (z_0) und Polstellen (z_∞) ($\alpha = 1$)
b) Nullstellen und Polstellen ($0 \leq \alpha \leq 1$)

Damit läßt sich die Wirkung der Polstellen auf die Werte auf dem Einheitskreis ($z = e^{i\Omega}, 0 \leq \Omega \leq 2\pi$) skalieren. Dies sind die Werte im Frequenzbereich, d.h. die Extrema der Signalform (Übertragungsfunktion) prägen auch die Extrema der Signalform des Fehlersignals. Das Fehlersignal paßt sich der Form des Sprachsignals an und wird von diesem überdeckt.

4.2.2 Federal Standard 1016 (FS1016)

Der Federal Standard 1016 (FS1016) ist ein Standard für CELP, herausgegeben vom Department of Defense (DoD) der Vereinigten Staaten. Er beinhaltet die benötigten Parameter zur Implementation eines Sender-Empfänger-System auf Basis von CELP.
FS1016 legt die Parameter wie folgt fest:

Abschnittlänge (frame interval)	30ms (240 Signale)
Unterabschnittslänge (subframe interval)	(60 Signale)
Prädiktorordnung	10
Länge des stochastischen Codebuchs (Erzeugung wird im Standard mittels Fortran-Programm beschrieben)	512
Länge des adaptiven Codebuchs	147
Gewichtsfaktor α	0.9

4.3 LPC vs CELP: Qualität der dekodierten Sprache

In dieser Arbeit wurde der theoretische Hintergrund von zwei Kodierverfahren - LPC und CELP - vorgestellt. Beides sind Verfahren, die auf der linearen Vorhersage basieren.

Um eine Aussage zur Qualität beider Verfahren zu machen, wurden parallel die Implementationen der Standards FS1015 (LPC10e) und FS1016

(CELP) getestet.

Beide Verfahren wurden mit einer männlichen, einer weiblichen Stimme und Musik getestet. Alle Daten wurden mit einer Abtastfrequenz von 8000Hz und einer Auflösung von 8 Bit pro Abtastwert aufgenommen. Qualitativ wurden dabei folgende Ergebnisse erzielt (alle Werte in dB):

	LPC		CELP	
	SNR	SEGSNR	SNR	SEGSNR
Männliche Stimme	10.80	11.27	11.05	12.94
Weibliche Stimme	10.93	11.51	10.13	12.63
Musik	2.01	2.95	7.12	8.63

Diesen Ergebnissen nach zu urteilen, gibt es bei menschlicher Sprache nur geringfügige Unterschiede zwischen den beiden Verfahren LPC und CELP ($\leq$ 2dB). Deutlicher ist der Unterschied bei Musik (zwischen 5-6dB). Daraus kann man auch schließen, daß Hintergrundgeräusche bei CELP weniger störend sind als bei LPC. Zwischen männlicher und weiblicher Stimme gibt es keine markanten Unterschiede ($\leq$ 1dB).

Meiner Meinung nach ist die dekodierte Sprache bei CELP qualitativ hochwertig. Es bestehen nur geringe Unterschiede zum Original. Bei der Musik war bei CELP im Gegensatz zu LPC die Stimme der Sängerin noch zu verstehen.

Die von LPC erzeugte Sprache ist verständlich - auch Sprecher - Unterschiede, insbesondere zwischen männlicher und weiblicher Stimme können unterschieden werden.

Dennoch weist dieses Kodierverfahren ganz erhebliche Schwächen durch das Fehlen der Erzeugung von transitiven Geräuschen auf. Die Sprache klingt damit an manchen Stellen nicht klar.

Unabhängig davon klingt LPC im Gegensatz zu CELP sehr künstlich und hallt auch sehr stark nach.

5 Schlußwort

Diese Arbeit dient als Einführung in Verfahren mit niedrigen Datenraten. Das sind Verfahren, die Sprache auf Basis der linearen Vorhersage auf Dekodiererseite synthetisieren und damit sehr viele Prinzipien der menschlichen Spracherzeugung nachbilden.

In dieser Arbeit wurden Implementationen der beiden Standards FS1015 und FS1016 untersucht, um abschätzen zu können, inwiefern sich diese Verfahren für die Übermittlung von Audiodaten in einem Videokonferenzsystem eignen. Darüberhinaus wurde eine TCP/IP-Verbindung implementiert, um Kodierer und Dekodierer unter möglichst realistischen Bedingungen testen zu können.

Bei dem Verfahren *pitch-excited linear prediction (FS1015)* fiel auf, daß die dekodierte Sprache sehr künstlich und an manchen Stellen (transitive Geräusche) nicht klar ist. In der Literatur wird zusätzlich darauf hingewiesen, daß dieses Verfahren sehr empfindlich auf Hintergrundgeräusche und Mehrsprecherumgebungen reagiert. Pitch-excited LP ist demnach für Videokonferenzsysteme ungeeignet.

Bei *code-excited linear prediction (FS1016)* ist die Qualitätsminderung nicht auffällig. Durch die verschiedene Wahl der Anregungssignale reagiert dieses Verfahren auch nicht so empfindlich auf Hintergrundsgeräusche und mehrere Sprecher. So wurde beim Test von Musik festgestellt, daß die Stimme der Sängerin noch verständlich war. CELP ist demnach eher für Videokonferenzsysteme geeignet.

Ob CELP eingesetzt werden kann, hängt aber von der Realisierung der Vektorquantisierung ab. In einem Videokonferenzsystem erfordert die Kodierung der Bildsignale eine große Rechenkapazität. Da CELP durch die Verwendung einer rechenintensiven Vektorquantisierung sehr große Rechenkapizitäten benötigt, kann es deshalb u.U. nicht verwendet werden.

Die Untersuchung der benötigten Rechenkapazitäten bei verschiedenen Implementationen einer Vektorquantisierung und damit die Entscheidung für oder gegen den Einsatz von CELP in einem Videokonferenzsystem bleibt einer fortführenden Arbeit vorbehalten.

Literatur

[1] Thomas P. Barnwell III, Kambiz Nayebi, Craig H. Richardson. *Speech Coding - A Computer Laboratory Textbook.* John Wiley & Sons, 1996.

[2] Isabel M. Trancoso. *An Overview of Different Trends on CELP Coding* aus Speech Recognition and Coding - New Advances and Trends, S.351-368. Springer, 1995.

[3] A.Gersho. *Concepts and Paradigms in Speech Coding*aus Speech Recognition and Coding - New Advances and Trends, S.369-186. Springer, 1995.

[4] Joseph P. Campbell Jr., Thomas E. Tremain, Vanoy C. Welch. *The Proposed Federal Standard 1016 4800bps Voice Coder: CELP.* Speech Technology Magazine April/May 1990, S. 58-63.

[5] *Federal Standard 1016* . General Services Administration Office of Information Resources Management, 1991.

[6] Ronald W. Schafer, Lawrence R. Rabiner. *Digital Representations of Speech Signals* . IEEE, 1995.

[7] P.Vary, U.Heute, W.Hess. *Digitale Sprachsignalverarbeitung.* Teubner Stuttgart, 1998.

[8] Alan V. Oppenheim, Alan S. Willsky. *Signals & Systems* , 2. Auflage. Prentice Hall,1997.

[9] Heinz Kronmüller. *Digitale Signalverarbeitung.* Springer 1991.

[10] E.G. Schukat-Talamzzini. *Automatische Spracherkennung - Grundlagen, statistische Modelle und effiziente Algorithmen.* Vieweg, 1995.

A TCP/IP Programme als Quellcode

Im Rahmen dieser Arbeit wurde die Übertragung von Sprache zwischen zwei Rechnern über eine TCP/IP-Verbindung realisiert. Für die Kodierung von Sprache wurde der Standard FS1015 (LPC10e) verwendet. Die vorliegenden Programme als Quellcode realisieren auf Kodiererseite die folgenden drei Schritte: Die Aufnahme von Sounddaten via Sound Bibliothek des SGI-Systems (*digital media*), die Kodierung mit Hilfe einer C-Bibliothek (Herausgegeben vom U.S. DoD) und die Versendung via TCP/IP mit System-Biblitheken, die i.allg. jedem Unix-System zur Verfügung stehen. Der Dekodierer realisiert den Empfang der Daten via TCP/IP, die Dekodierung und das Senden der dekodierten Sprachsignale an die Soundkarte des SGI-Systems.

A.1 Kodierer

```
/* Name: encodeLPCServer.c
   Author: Martin Schmidt
   Date: September 1999
   Type (executable, library): executable
   Purpose: Contains Server-application, which sends data lpc-encoded
            to the listening Clients
   References: (none)
   Functions:
     main
 */

#include <stdio.h>   /* standard io-library */
#include "lpc10.h"   /* lpc-10 library (used for decoding) */
#include "inet.h"    /* contains read/write functions for
                        TCP/IP Connections */

#include <signal.h>  /* internal */
#include <dmedia/audio.h>      /* audio library for indy sound system*/

void sendEncodedAudioData (int sockfd)
```

```
{
   /** encoder - declarations */
   float speech[LPC10_SAMPLES_PER_FRAME]; /* decoded speech */
   signed char samples[LPC10_SAMPLES_PER_FRAME];
   INT32 bits[LPC10_BITS_IN_COMPRESSED_FRAME];
    /* parameters for one frame
      1bit = voiced/unvoiced decision
      6bit = quantized pitch period
      11 LPC coefficients and gain
       c1-c4 (5 bit quantized)
       c5-c8 (4 bit quantized)
       c9 (3 bit quantized)
       c10 (2 bit quantized)
       gain (5 bit quantized)
     */

    struct lpc10_encoder_state *st; /* encoder state (lpc internal) */
    char sendBuffer[7];

   /** audio - declarations*/
    ALport        port;  /**  audio port */
    int i;


   /** initialize encoder-state */
    st = create_lpc10_encoder_state();
   /** open the audio port for read */
    port = OpenALPort(0);

   /* encoding loop
       1) read 180 samples from Port
       2) encode speech to 54 bits via LPC-10e
       3) write these bits into sendBuffer
       4) send one frame via tcp/ip (54 bits, 7 bytes)
    */
   while (1)     {
     ALreadsamps(port, samples, LPC10_SAMPLES_PER_FRAME);
     for (i = 0; i<LPC10_SAMPLES_PER_FRAME; i++)
       { speech[i] = ((float)samples[i] - 0.5) / 128.0; }
     lpc10_encode(speech, bits, st);
     writeBitsToBuffer(sendBuffer, bits, LPC10_BITS_IN_COMPRESSED_FRAME);
     writen(sockfd, sendBuffer, 7);
    }
```

```
    /** close */
    ALcloseport(port);
    exit(0);
}

/* Procedure Name: main
   Author: Martin Schmidt
   Date:   September 1999
   Action: Establish Server, waits for client, starts sending
           audio data
    Input: (none)
    returns: (none)
 */
void
main(int argc, char *argv[]) {

  int sockfd, newsockfd, clilen, childpid;
  struct sockaddr_in cli_addr, serv_addr;
  pname = argv[0];

  /* get socket id (socketfd) */
  if ((sockfd = socket(AF_INET, SOCK_STREAM,0)) < 0 )
    { printf("Error while establishing socket \n"); exit(0); }

  /* get socket adress */
  bzero((char *) &serv_addr, sizeof(serv_addr));
  serv_addr.sin_family =AF_INET;
  serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
  serv_addr.sin_port = htons(SERV_TCP_PORT);

  /* bind socketfd with socket adress */
  if (bind(sockfd, (struct sockaddr *) &serv_addr, sizeof(serv_addr)) < 0)
    { printf("Error while binding socket and type \n"); exit(0); }

  /* establish listener */
  listen (sockfd, 5);

  for ( ;;)
   {/* create connection with Client/Server via newsocketfd */
    clilen = sizeof(cli_addr);
    newsockfd = accept (sockfd, (struct sockaddr *) & cli_addr, &clilen);

    if (newsockfd < 0)
      { printf("Error while accepting \n"); exit(0); }
```

```
    if ((childpid = fork()) < 0)
      { printf("Error calling fork \n"); exit(0); }

    /* start sending encoded audio data */
    if (childpid == 0)
      { close (sockfd);
        sendEncodedAudioData(newsockfd);
        exit(0); }
       }

    exit(0);
}
```

A.2 Dekodierer

```
 /*
 Name: decodeLPCClient.c
 Author: Martin Schmidt
 Date: September 1999
 Type (executable, library): executable
 Purpose: receiving lpc-encoded data via TCPIP from
          Server SERV_HOST_ADDR (inet.h),
          decodes received data and sends to audio port
 References: (none)
 Functions: main
 */

#include <stdio.h> /* standard io-library */
#include "lpc10.h" /* lpc-10 library (used for decoding) */
#include "inet.h"  /* inet library (used for receiving and sending data) */

#include <signal.h> /* internal */
#include <dmedia/audio.h>     /* audio library for indy sound system*/

/* Procedure Name: receiveWriteAudioData
   Author: Martin Schmidt
   Date:   September 1999
   Action: receives, decodes and plays data
   Input:
      sockfd = id to tcpip socket
   returns: (none)
 */
void receiveWriteAudioData (int sockfd)
```

```
{  /** decoder - declarations */
   float speech[LPC10_SAMPLES_PER_FRAME]; /* decoded speech */
   signed char samples[LPC10_SAMPLES_PER_FRAME];
   INT32 bits[LPC10_BITS_IN_COMPRESSED_FRAME];
    /* parameters for one frame
      1bit = voiced/unvoiced decision
      6bit = quantized pitch period
      11 LPC coefficients and gain
       c1-c4 (5 bit quantized)
       c5-c8 (4 bit quantized)
       c9 (3 bit quantized)
       c10 (2 bit quantized)
       gain (5 bit quantized)
     */
   struct lpc10_decoder_state *st;

   char readBuffer[7];
   int i;

  /** audio - declarations*/
   ALport port; /** audio port */

  /** initialize decoder state */
   st = create_lpc10_decoder_state();

  /** open the audio port for write */
   port = OpenALPort(1);

  /* decoding loop
       1) read 7 Bytes (54 bits, one frame) via tcpip
       2) read bits from buffer
       3) decode to speech
       4) translate to linear pcm (8000Hz, 8 bits, mono)
       5) write decoded speech to audio port
     */
   while (1)     {
     readn(sockfd, readBuffer, 7);
     readBitsFromBuffer(readBuffer, bits, LPC10_BITS_IN_COMPRESSED_FRAME);
     lpc10_decode(bits, speech, st);
     for (i = 0; i<LPC10_SAMPLES_PER_FRAME; i++)
        { samples [i] = speech[i] * 127.5 - 0.5; }
     ALwritesamps(port, samples, LPC10_SAMPLES_PER_FRAME);
     writeSamples(outfile, speech, LPC10_SAMPLES_PER_FRAME);
    }
```

```
  /** close and exit */
   ALcloseport(port);
   exit(0);
}

/* Procedure Name: main
   Author: Martin Schmidt
   Date:   September 1999
   Action: establishes tcpipsocket to Server SERV_HOST_ADDR (inet.h)
   Input: (none)
   returns: (none)
 */
void
main(int argc, char *argv[]) {

 int sockfd;
  struct sockaddr_in serv_addr;
  pname = argv[0];
  bzero((char *) &serv_addr, sizeof(serv_addr));
  serv_addr.sin_family = AF_INET;
  serv_addr.sin_addr.s_addr = inet_addr(SERV_HOST_ADDR);
  serv_addr.sin_port = htons(SERV_TCP_PORT);

  if ((sockfd = socket (AF_INET, SOCK_STREAM, 0)) < 0)
    {  printf("Error while establishing socket \n"); exit(0); }

  if (connect (sockfd, (struct sockaddr *) &serv_addr,
        sizeof(serv_addr)) < 0)
    {  printf("Server couldn't be connected \n"); exit(0); }

  receiveWriteAudioData(sockfd);

  close(sockfd);
  exit(0);
}
```