Studienarbeit:

Fließbandverarbeitung in der Vorverarbeitung von Sprachsignalen

Bernhard Suhm
(suhm@ira.uka.de)

Zusammenfassung

Die Vorverarbeitung von Sprachsignalen wurde für das Spracherkennungssystem JANUS von einer sequentiellen Ausführung auf Fließbandverarbeitung umgestellt. Dadurch ist es möglich geworden, die Vorverarbeitung in Echtzeit auszuführen.

am
Institut für Logik, Komplexität und Deduktionssysteme.

6. April 1992

Inhaltsverzeichnis

1 Einführung

In der Mustererkennung werden in der Regel die von den Sensoren aufgenommenen Eingabedaten zunächst aufbereitet. Diese Vorverarbeitungsschritte dienen dazu, diejenigen Merkmale besonders herauszuarbeiten, anhand derer im nachfolgenden Erkennungsschritt die Klassifikation vorgenommen wird.
Im Fall des Spracherkennungssystems JANUS[1] ist die Energieverteilung im Spektrum des Sprachsignals das entscheidende Merkmal.

Ein Schwerpunkt der Forschung auf dem Gebiet der Spracherkennung am Institut für Logik, Komplexität und Deduktionssysteme an der Universität Karlsruhe ist die Erkennung und Übersetzung gesprochener Sprache *in Echtzeit.*
Im Rahmen einer Diplomarbeit [4] wurde bereits die Implementierung auf einer SIMD–Architektur[2] untersucht. Sie ist allerdings an die Verfügbarkeit hochparalleler SIMD–Rechner gebunden.
Im Sinne eines möglichst weiten Einsatzfeldes ist es allerdings wünschenswert, das Spracherkennungssystem auch auf Workstations so zu implementieren, daß eine Verarbeitung in Echtzeit erreicht wird.

Die bisher für JANUS verfügbare Implementierung konnte diesen Anforderungen nicht genügen; allein die Vorverarbeitung der Sprachsignale benötigte auf einer leistungsfähigen Workstation[3] eine Zeit, die in der Größenordnung der Dauer der Spracheingabe selbst lag.

Tabelle 1 stellt das Ausführungsprofil der sequentiellen Signalvorverarbeitung an einem Beispielsatz *(Could you give me your name and address ?)* dar. Man kann ihr entnehmen, daß die Berechnung des Signalspektrums mit Hilfe der Fourieranalyse, die im wesentlichen in den Routinen "fht", "fht_pow_spec" und "ham" durchgeführt wird, den Hauptanteil an für die Vorverarbeitung erforderlichen Ausführungszeit ausmacht.

Die Berechnung des Signalspektrums erfolgt in der Weise, daß aus dem Rechner in digitisierter Form vorliegenden Signal mit Hilfe einer Fensterfunktion kurze Teilabschnitte ausgeblendet werden und für jeden dieser Abschnitte eine Fouriertransformation durch-

[1] JANUS ist eine Kooperation der Universität Karlsruhe mit der Carnegie–Mellon University (Pittsburgh), ATR Interpreting Telephone Laboratories (Osaka) und der SIEMENS AG.
[2] MasPar MP-1
[3] DEC 5000/200

Zeit in %	Summe %	Routine	Funktionalität
72.4	72.4	fht	Hartley Transformation
8.0	80.4	fht_pow_spec	Leistungsspektrum
4.5	84.9	ham	Hamming Fenster
3.6	88.5	ptp_amp	Clipping
3.6	92.1	main	Hauptprogramm
3.1	95.2	mkcoeff	*melscale*-Koeffizienten
1.8	97.0	log	*melscale*-Koeffizienten

Tabelle 1: Ausführungsprofil der sequentiellen Vorverarbeitung für einen Beispielsatz

geführt wird. Eine solche Vorgehensweise eignet sich sehr gut für die Anwendung der Fließband–Technik.
In dieser Studienarbeit wurde untersucht, wie dies auf einer Workstation implementiert werden kann.

Im folgenden wird ein kurzer Überblick über den Inhalt dieser Ausarbeitung gegeben.
In Kapitel 2 werden die der Vorverarbeitung von Sprachsignalen zugrundeliegenden Verfahren und Algorithmen vorgestellt, bevor im Kapitel 3 die mit der Implementierung als Fließband zusammenhängenden Probleme diskutiert werden. In Kapitel 4 werden die Ergebnisse kurz dargestellt und ein Ausblick auf die Fortführung der Fließbandverarbeitung im eigentlichen Spracherkenner gegeben. Im Anhang werden einige Hinweise für den praktischen Umgang mit der im Rahmen dieser Studienarbeit erstellten Software gegeben.

Mehr Informationen über das JANUS–Projekt findet der interessierte Leser in [5].

2 Vorverarbeitung von Sprachsignalen

Sprachlaute lassen sich in erster Linie anhand ihrer spektralen Zusammensetzung unterscheiden. Daher wird zur automatischen Erkennung von gesprochener Sprache nicht das Zeitsignal des Mikrofons (Schalldruckverlauf) ausgewertet, sondern es wird das Frequenzspektrum des Sprachsignals herangezogen. In Anlehnung an das menschliche Gehör muß also eine Spektralanalyse wesentlicher Bestandteil der Vorverarbeitung sein.

Gesprochene Sprache ist außerdem durch ein hohes Maß an Variabilität gekennzeichnet. Sie ist nicht nur von der Umgebung und dem Sprecher abhängig, sondern auch von dessen augenblicklicher Konstitution. Ferner ändern sich die Bedingungen, unter denen das Sprachsignal aufgenommen wird, ständig – z.B. der Abstand zwischen Mund und Mikrofon. Damit der Spracherkenner in etwa vergleichbare Daten erhält, muß daher das gewonnenene Spektrum geeignet normalisiert werden.

Aus dem praktischen Umgang mit einem Spracherkennungssystem ergibt sich ein weiteres Problem: Nachdem der Sprecher dem System mitgeteilt hat (meist per Tastendruck), daß er zu sprechen beabsichtigt, beginnt er meistens nicht sofort mit dem ersten Wort, sondern es vergeht noch eine gewisse Zeit der „Stille“, die je nach äußeren Umständen mit Hintergrundgeräuschen angefüllt sein kann. Es ist aus Effizienzgründen (man strebt eine Spracherkennung in Echtzeit an) wünschenswert, "Stille" am Anfang und Ende des Sprachsignals abzuschneiden.

2.1 Wahl der Darstellung des Sprachsignals

Die Darstellung der Sprachsignale muß gewährleisten, daß die Merkmale, anhand derer die Spracherkennung vorgenommen werden soll, durch die Vorverarbeitung in ihrem Bedeutungsgehalt nicht verändert werden. Diese Merkmale bezeichnet man auch als *auditive Hinweise*. Sie sind im allgemeinen sowohl vom Sprecher selbst abhängig als auch vom Kontext, der durch den Satz gebildet wird.

Sprach-Datenkompressionssystem	Spracherkennungssysteme
Signal muß rekonstruierbar sein	nicht erforderlich
Sprecher-Variabilität erhalten	unterdrücken
Sprach-Variabilität erhalten	Sprach-Variabilität unterdrücken
Sprecher-/Sprachcharakteristiken erhalten	nur Sprachcharakteristiken erhalten
jede Äußerung für sich behandeln	gemeinsame Merkmale interessant
Darstellungsproblem	Mustererkennungsproblem
Modelle des Sprechtraktes als Grundlage	auditive Hinweise als Grundlage

Tabelle 2: Gegenüberstellung der Anforderungen

Die Sicherheit, mit der der Mensch Sprachsignale zu erkennen vermag, rechtfertigt es, sich bei der Auswahl der Darstellung an das natürliche Vorbild zu halten.
Schalldruckwellen werden über die Ohrmuschel und den Schädelknochen an das Trommelfell geleitet, welches als Membrane dient. Anschließend wird in der spiralförmigen Chornea des Innenohrs eine Frequenzanalyse durchgeführt. Das derart vorverarbeitete Sprachsignal wird über die Nervenfasern der Hörbahn – nach Frequenzbereichen getrennt – den für das Sprachverständnis zuständigen Bereichen der Großhirnrinde (der sogenannten „Wernicke–Sprachregion") codiert als Folge von Nervenimpulsen zugeleitet.

In der Natur hat sich also das Spektrogramm, d.h. der Energiegehalt in bestimmten Frequenzbereichen, als Darstellungsform bewährt. Sein Informationsgehalt ist offensichtlich für die Sprach*erkennung* ausreichend.

An die Vorverarbeitung von Sprachsignalen für Spracherkennungssysteme und für Systeme zur Kompression von Sprachdaten sind unterschiedliche Anforderungen zu stellen. In [2] werden die in Tabelle 2 dargestellten Kriterien genannt.

2.2 Spektralanalyse

Sprachsignale sind nichtstationäre Vorgänge, die nur über kürzere Zeitabschnitte als hinreichend stationär angesehen werden können. Die spektralen Eigenschaften ändern sich folglich auch als Funktion der Zeit.
Im folgenden wird beschrieben, wie der zeitliche Verlauf des Spektrums erfaßt wird. Die Grundidee ist, aus dem im Rechner in digitisierter Form vorliegenden Signal mit Hilfe einer Fensterfunktion kurze Teilabschnitte auszublenden und für jeden dieser Abschnitte eine Fourieranalyse durchzuführen.

2.2.1 Hamming-Fenster

An kurz aufeinanderfolgenden Analysezeitpunkten[4] wird ein Teilabschnitt des Sprachsignals, über den der Frequenzgehalt des Signals als konstant angesehen werden kann, ausgeblendet.

[4] Das Zeitintervall zwischen zwei Analysen ist eine Konstante des Programms; für Sprachsignale haben sich 10 ms als ausreichend erwiesen.

Dazu wird das Signal $f(\tau)$ im Zeitbereich mit der am Analysezeitpunkt t gespiegelten Fensterfunktion $w(t-\tau)$ multipliziert. In Abbildung 1 ist diese Vorgehensweise für $w(\tau) = e^{-\tau}$ dargestellt.

Die Abtastung und Quantifizierung des Sprachsignals erfolgt durch einen A/D–Wandler, der mit einer Abtastrate von 16 KHz[5] abtastet und dem Rechner die Werte als 16-Bit Integer–Zahlen[6] über einen Bus zur Verfügung stellt.
Das abgetastete Zeitsignal kann man also mit folgender abkürzender Schreibweise bezeichnen:

$$\mathrm{s(n)} = \mathrm{f(n}\ \triangle\mathrm{t)}$$

wobei die diskrete Zeitvariable mit n bezeichnet ist und das Abtastintervall $\triangle t$ sich als Kehrwert der Abtastrate berechnet.

Einen entscheidenden Einfluß hat offensichtlich die Wahl der Festerfunktion $w(\tau)$. Sie soll einerseits den für die Spracherkennung relevanten Frequenzbereich (ungefähr 0 – 8000 Hz) unverändert lassen, d.h. im Frequenzbereich möglichst rechteckig sein; andererseits soll sie aber auch im Zeitbereich schnell abklingen. In der Praxis hat sich das sogenannte "Hamming–Fenster" bewährt:

$$w(k) = 0.54 + 0.46\cos(2\pi k/N) \qquad \text{für } -N/2 \leq k \leq N/2 - 1 \tag{1}$$

Den Verlauf des "Hamming–Fensters" im Zeit– und Frequenzbereich kann man Abbildung 2 entnehmen.

Das ausgeblendete Teilstück für ein diskretes, digitales Spektrum über eine Periode mit N Abtastwerden berechnet sich also gemäß

$$\tilde{s}(n) = \sum_{k=-N/2}^{N/2-1} s(k)w(n-k) \tag{2}$$

2.2.2 Fourier Transformation und Hartley Transformation

Zur Bestimmung des Frequenzspektrums eines diskreten, zeitabhängigen Signals wird meist die diskrete Fourier Transformation (DFT) benutzt; angewendet auf das Signal $f(\tau)$ im Bereich $0 \leq \tau \leq N-1$ lautet sie:

$$F(\nu) = \frac{1}{N}\sum_{\tau=0}^{N-1} f(\tau)\exp\left(-i\frac{2\pi\nu\tau}{N}\right) = \frac{1}{N}\sum_{\tau=0}^{N-1} f(\tau)\left(\cos\left(\frac{2\pi\nu\tau}{N}\right) - i\sin\left(\frac{2\pi\nu\tau}{N}\right)\right) \tag{3}$$

wobei ν ebenfalls im Bereich von $0 \leq \nu \leq N-1$ definiert ist.

[5] Die Abstastrate ist eine Konstante des Programms. 16 kHz stellen sicher, daß die obere Grenzfrequenz des für die Spracherkennung relevanten Teil des Spektrum (7250 Hz, vergleiche mit Tabelle 3 in Kapitel 2.2.4) noch korrekt abgetastet wird.

[6] Der Typ der Abtastwerte ist durch die verwendete Hardware festgelegt.

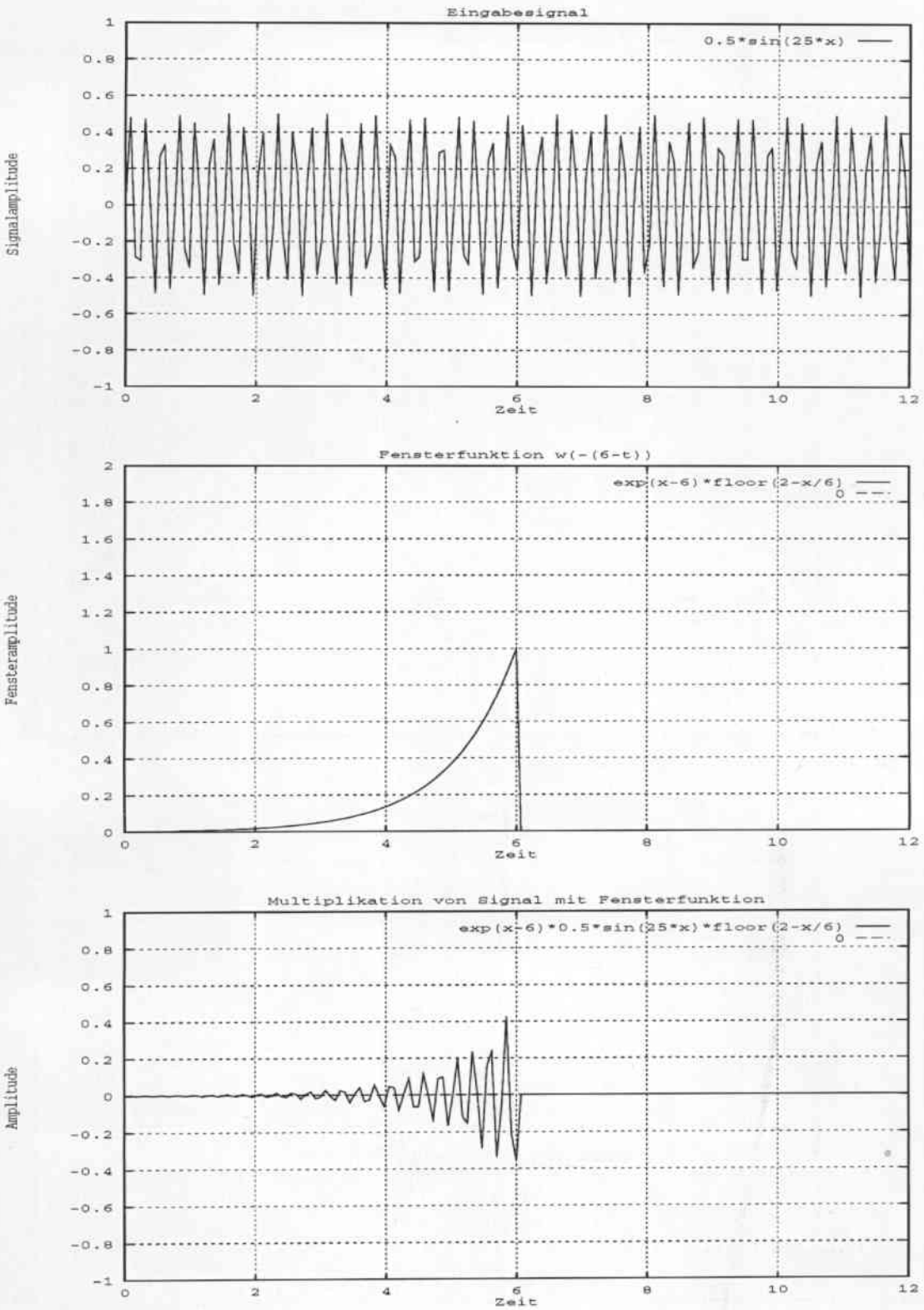

Abbildung 1: Ausblenden eines Teilstücks des Sprachsignals

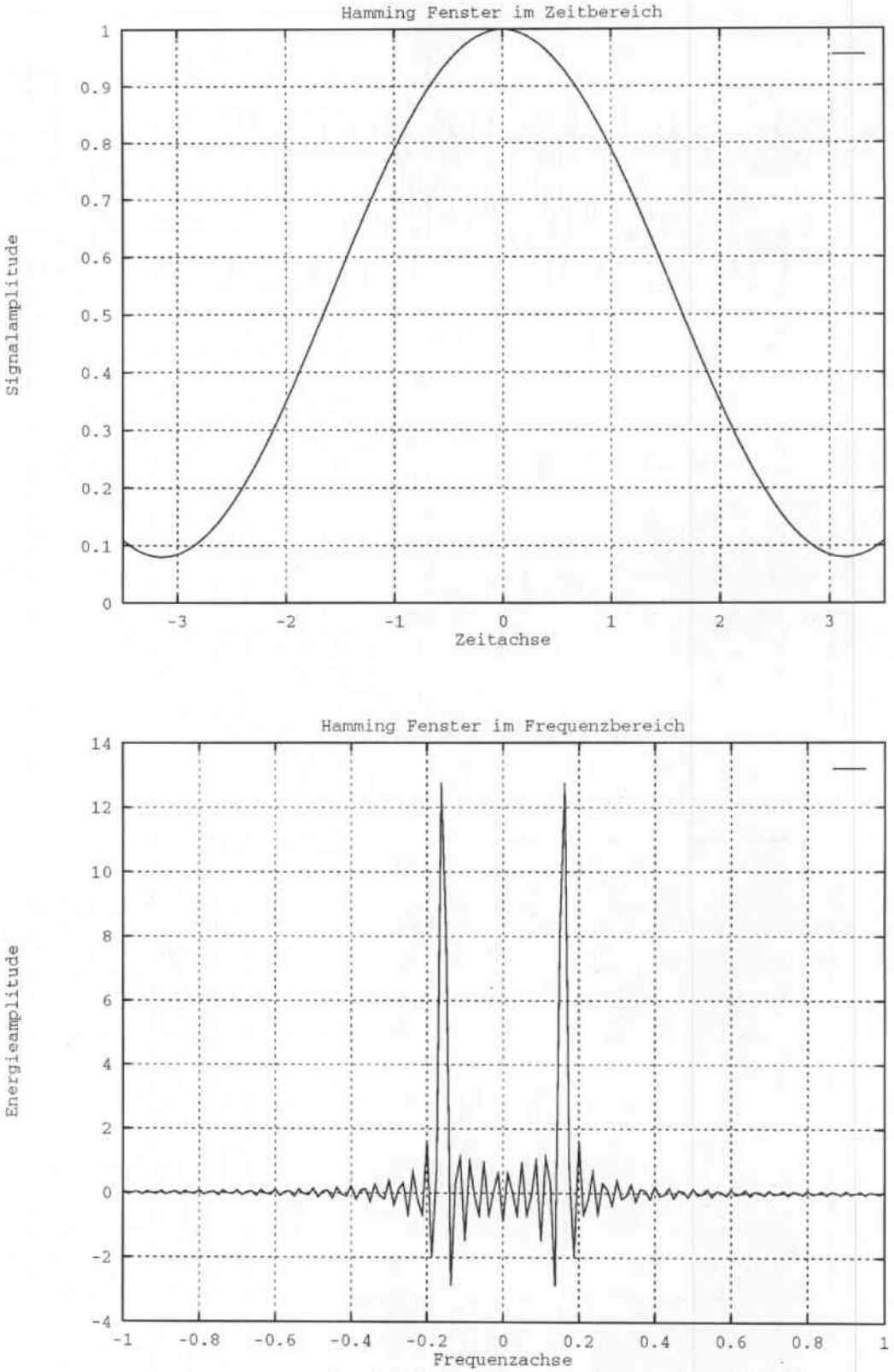

Abbildung 2: Hamming–Fensterfunktion im Zeit– und Frequenzbereich

Wenn wie in der Anwendung auf Sprachsignale τ eine Zeitvariable ist, dann stellt ν eine Frequenzvariable dar. Ist das zeitliche Intervall zwischen zwei aufeinanderfolgenden Elementen der Folge $f(\tau)$ eine Sekunde, dann enspricht der Abstand zwischen zwei aufeinanderfolgenden Werten von ν einem Frequenzintervall von $\frac{1}{N}$ Hz. Durch Inkrementieren von ν steigt die Frequenz entsprechend, jedoch nur bis $\nu = N/2$. Über diesen Wert hinaus entspricht ν der Frequenz $(N-\nu)/N$ bis zu der Frequenz Null für $\nu = N$[7].

Wenn das Signal nur reellwertige Werte annehmen kann, führt die (komplexwertige) DFT doppelt so viele Multiplikationen aus, wie zur Berechnung des Frequenzspektrums notwendig sind; die diskrete Hartley Transformation (DHT) stellt ein reellwertiges Äquivalent zur DFT für den Fall reellwertiger Eingabedaten dar. Sie lautet:

$$H(\nu) = \frac{1}{N}\sum_{\tau=0}^{N-1} f(\tau)\mathrm{cas}\left(\frac{2\pi\nu\tau}{N}\right) = \frac{1}{N}\sum_{\tau=0}^{N-1} f(\tau)\left(\cos\left(\frac{2\pi\nu\tau}{N}\right) + \sin\left(\frac{2\pi\nu\tau}{N}\right)\right) \quad (4)$$

Man kann die DHT in ihre gerade *(Even)* und ungerade *(Odd)* Komponente zerlegen:

$$H(\nu) = E(\nu) + O(\nu) \qquad \text{wobei}$$

$$E(\nu) = 1/2\,(H(\nu) + H(-\nu)) \qquad \text{und} \qquad O(\nu) = 1/2\,(H(\nu) - H(-\nu))$$

Dann kann man die Äquivalenz von Fourier Transformierter F und Hartley Transformierter H in den folgenden Gleichungen (5) und (6) ausdrücken:

$$F(\nu) = E(\nu) - iO(\nu) \quad (5)$$

$$H(\nu) = Re\{F(\nu)\} - Im\{F(\nu)\} \quad (6)$$

2.2.3 FFT und FHT

Ein Vergleich der Gleichungen (3) und (4) legt die Vermutung nahe, daß es einen schnellen Algorithmus zur Berechnung der Hartley Transformation gibt, ähnlich der schnellen Fourier Transformation (FFT). Im folgenden soll der in [3] vorgeschlagene FHT–Algorithmus dargestellt werden.

Die FFT beruht auf einem *Divide and Conquer* Algorithmus. Dabei wird – unter Ausnutzung inhärenter Symmetrien – die Folge von Signalwerten $\{f(\tau) \mid 0 \leq \tau \leq N-1\}$, wobei N eine Zweierpotenz ist, in zwei Teilfolgen aufgespalten *(Divide)*, für jede Teilfolge getrennt die FFT (durch einen rekursiven Aufruf) berechnet und aus den Spektralkoeffizienten der Teilfolgen diejenigen der Folge von Signalwerten berechnet *(Conquer)*.

Diese Vorgehensweise liegt auch der FHT zugrunde. Der Rekombination der Spektralkoeffizienten aus den Koeffizienten der Teilfolgen liegen im Fall der Hartley Transformation die folgenden Theoreme zugrunde, die man sich durch elementare Rechnung aus der Definitionsgleichung der DHT (4) herleiten kann.

[7] Das Aufnahmeprogramm für JANUS arbeitet mit einer Abtastrate von 16 KHz und einer Fenstergröße $N = 256$. Daraus ergibt sich als Einheit für ν die Frequenz 62.5 Hz.

Dehnungstheorem der DHT:
Sei $DHT\{f(\tau) \mid 0 \leq \tau \leq N-1\} = \{H(\nu) \mid 0 \leq \nu \leq N-1\}$
Dann ist

$$DHT\{f(0), 0, f(1), 0, \ldots, f(N-1)\} = 1/2\{H(0), H(1), \ldots, H(N-1), H(0), H(1), \ldots, H(N-1)\} \quad (7)$$

Verschiebungstheorem der DHT:

$$DHT\{f(\tau - T) \mid 0 \leq \tau \leq N-1\} = \{\hat{H}(\nu) \mid 0 \leq \nu \leq N-1 \quad \text{und} \quad \hat{H}(\nu) = \cos\left(\frac{2\pi\nu T}{N}\right) H(\nu) - \sin\left(\frac{2\pi\nu T}{N}\right) H(-\nu)\} \quad (8)$$

Nach diesen Vorüberlegungen läßt sich – in Analogie zur einschlägigen FFT – folgender Algorithmus zur schnellen Berechnung der diskreten Hartley Transformation angeben:

Eingabe: Folge von Signalwerten $\bar{f} = \{f(\tau) \mid 0 \leq \tau \leq N-1\}$ mit $N = 2^p$

Ausgabe: Folge der Spektralkoeffizienten $\hat{F} = \{F(\nu) \mid 0 \leq \nu \leq N-1\}$

1. *Schritt*: Divide
   Fasse die Folgen der Signalwerte mit geradem bzw. ungeradem Index als Teilfolgen auf.
   $\bar{g} = \{f(0), f(2), \ldots, f(N-2)\}$
   $\bar{k} = \{f(1), f(3), \ldots, f(N-1)\}$

2. *Schritt*:
   Berechne $FHT\{\bar{g}\} = \hat{G}$ und $FHT\{\bar{k}\} = \hat{K}$.

3. *Schritt*: Conquer
   Aus dem Dehnungstheorem (7) folgt

   $$DHT\{f(0), 0, f(2), 0, \ldots, f(N-2)\} = 1/2\{G(0), G(1), \ldots, G(\tfrac{N}{2}-1), G(0), G(1), \ldots, G(\tfrac{N}{2}-1)\}$$

   und analog

   $$DHT\{f(1), 0, f(3), 0, \ldots, f(N-1)\} = 1/2\{K(0), K(1), \ldots, K(\tfrac{N}{2}-1), K(0), K(1), \ldots, K(\tfrac{N}{2}-1$$

   Wenn man das Verschiebungstheorem auf $\hat{K}$ anwendet erhält man für $DHT\{0, f(1), 0, f(3), \ldots, 0, f(N-1)\}$ die Spektralkoeffizienten

   $$\{\tilde{K}(\nu) \mid 0 \leq \nu < N \quad \text{und} \quad \tilde{K}(\nu) = \cos\left(\frac{2\pi\nu}{N}\right) K(\nu) + \sin\left(\frac{2\pi\nu}{N}\right) K(-\nu)\}$$

   wobei $\tilde{K}(\nu + N/2) = \tilde{K}(\nu)$ ist für $0 \leq \nu < N/2$.

Damit erhält man aus den Spektralkoeffizienten $\hat{G}$ und $\hat{K}$ die gesuchten Koeffizienten $\hat{F}$ gemäß

$$F(\nu) = G(\nu) + \cos\left(\frac{2\pi\nu}{N}\right) K(\nu) + \sin\left(\frac{2\pi\nu}{N}\right) K(-\nu)$$

Man bricht die Rekursion im Fall $N = 4$ ab und bestimmt die Spektralkoeffizienten durch direktes Einsetzen in die Definitionsgleichung der DHT (4):

$$\begin{aligned} H(0) &= 1/4(f(0) + f(1) + f(2) + f(3)) \\ H(1) &= 1/4(f(0) + f(1) - f(2) - f(3)) \\ H(2) &= 1/4(f(0) - f(1) + f(2) - f(3)) \\ H(3) &= 1/4(f(0) - f(1) - f(2) + f(3)) \end{aligned}$$

In der Vorverarbeitung von Sprachsignalen muß man über die gesamte Aufnahme äquidistant alle 5 ms ein Hamming–Fenster schieben und die DHT mit einer festen Fensterbreite berechnen. Daher kann man eine weitere Beschleunigung dadurch erzielen, daß man für die benötigten Sinus- und Kosinusfunktionswerte, sowie für die Einerkomplemente der Zahlen $0, \ldots, N-1$ Nachschlagetabellen im voraus für die gegebene Fensterbreite N berechnet.

2.2.4 Von Spektralkoeffizienten zu *melscale*-Koeffizienten

Die mit dem oben beschriebenen FHT–Algorithmus pro Zeitpunkt berechneten Spektralkoeffizienten werden allerdings noch nicht in dieser Form für die Erkennung des Sprachsignals verwendet.

Zunächst wird aus dem Frequenzspektrum das Leistungsspektrum berechnet gemäß

$$\begin{aligned} \bar{P}(\nu) = Re\{F(\nu)\}^2 + Im\{F(\nu)\}^2 & \\ = E(\nu)^2 + O(\nu)^2 & \\ = \frac{H(\nu)^2 + H(-\nu)^2}{2} & \end{aligned} \tag{9}$$

Man beachte, daß $-\nu = N - \nu \quad$ für $\quad N/2 \leq \nu < N$.

In einem weiteren Vorverarbeitungsschritt werden aus den so berechneten 128 [8] Leistungs–Koeffizienten 16 *melscale*-Koeffizienten berechnet. Dabei handelt es sich um eine logarithmische Quantisierung der Frequenzdaten. Die Idee ist, die frequenzabhängige Empfindlichkeit des menschlichen Ohrs nachzuahmen.
Die *melscale*-Koeffizienten berechnen sich gemäß Tabelle 3, wobei sich die Angaben zum Frequenzintervall auf eine Abtastrate von 16 KHz beziehen.

[8] Alle im folgenden auftretenden Fensterbreiten umfassen 256 Abtastpunkte, entsprechend der Implementierung für das JANUS – Spracherkennungssystem. Die Fenstergröße in Abtastpunkten ergibt sich als Produkt von Abtastrate (16 Muster pro Millisekunde) und Fenstergröße in Millisekunden (16 ms). Aus den 256 Spektralkoeffizienten werden 128 Leistungskoeffizienten berechnet.

Index	*melscale*-Koeffizient	Frequenzintervall
0	$m(0) = H(0) + H(1) + H(2)/2$	0 - 125 Hz
1	$m(1) = H(2)/2 + H(3) + \cdots + H(6)/2$	125 - 375 Hz
2	$m(2) = H(6)/2 + H(7) + \cdots + H(10)/2$	375 - 625 Hz
3	$m(3) = H(10)/2 + H(11) + \cdots + H(14)/2$	625 - 875 Hz
4	$m(4) = H(14)/2 + H(15) + \cdots + H(18)/2$	875 - 1125 Hz
5	$m(5) = H(18)/2 + H(19) + \cdots + H(22)/2$	1125 - 1375 Hz
6	$m(6) = H(22)/2 + H(23) + \cdots + H(26)/2$	1375 - 1625 Hz
7	$m(7) = H(26)/2 + H(27) + \cdots + H(30)/2$	1625 - 1875 Hz
8	$m(8) = H(30)/2 + H(31) + \cdots + H(35)/2$	1875 - 2187.5 Hz
9	$m(9) = H(35)/2 + H(36) + \cdots + H(41)/2$	2187.5 - 2562.5 Hz
10	$m(10) = H(41)/2 + H(42) + \cdots + H(48)/2$	2563.5 - 3000 Hz
11	$m(11) = H(48)/2 + H(49) + \cdots + H(57)/2$	3000 - 3562.5 Hz
12	$m(12) = H(48)/2 + H(49) + \cdots + H(68)/2$	3562.5 - 4250 Hz
13	$m(13) = H(68)/2 + H(69) + \cdots + H(81)/2$	4250 - 5062.5 Hz
14	$m(14) = H(81)/2 + H(82) + \cdots + H(97)/2$	5062.5 - 6062.5 Hz
15	$m(15) = H(97)/2 + H(98) + \cdots + H(116)/2$	6062.5 - 7250 Hz

Tabelle 3: Interpretation der *melscale*-Koeffizienten im Frequenzbereich

2.3 Anfangs- und Endpunktbestimmung

Es ist aus mehreren Gründen wünschenswert, daß innerhalb des durch die Abtastung gewonnenen Vektors der Signalwerte Anfang und Ende der eigentlichen Äußerung bestimmt werden:

- Die Analyse von Stille verzögert den Vorgang der Spracherkennung unnötig.
- Ein Vergleich von sprachlichen Äußerungen, die in verschiedenen Umgebungen gemacht wurden, ist nur möglich, wenn störende Hintergrundgeräusche abgeschnitten sind, die zu Beginn und am Ende mitaufgenommen wurden.

Die beste Lösung wäre natürlich, die Vorverarbeitung der abgetasteten Signale erst dann anlaufen zu lassen, wenn der Benutzer tatsächlich zu sprechen beginnt.
Eine solche automatische Detektion von Anfang bzw. Ende des Signals ist noch nicht implementiert[9]. Vielmehr muß der Benutzer dem System per Tastendruck mitteilen, daß die Aufnahme beginnen soll. Erst nach Beendigung der Aufnahme (ebenfalls per Tastendruck) werden anhand der gesamten Abtastwerte diejenigen Teile abgeschnitten *(clipping)*, deren Aussteuerungspegel unter einem Schwellwert liegt.

Die Schwierigkeiten beim Entwurf eines solchen Clipping–Algorithmus verbergen sich hinter mehreren Problemkreisen:

[9] Diese und die folgenden Aussagen beziehen sich auf das Aufnahmeprogramm für JANUS.

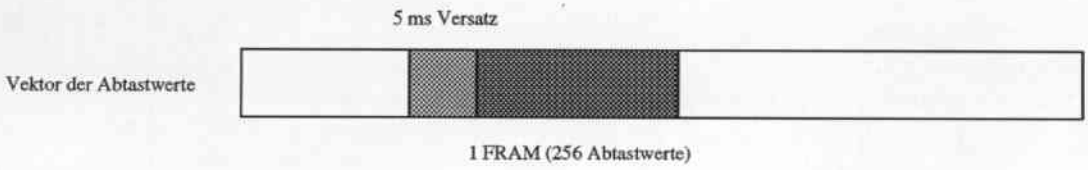

Abbildung 3: Schieben des Fensters über das Sprachsignal

- Unterscheidung zwischen Sprache, die auch aus schwachen oder geräuschähnlichen Teilen wie z. B. „fünf" bestehen kann, und Hintergrundgeräuschen; denn aufgrund des variierenden Mikrofonabstandes kann das Verhältnis in der Aussteuerung von Sprachsignal und Hintergrundgeräusch beträchtlich schwanken.
- Entscheidung, ob zwei Intervalle von Sprachsignal zusammengehören, wie z.B. Sprechpausen, die zwischen zwei aufeinanderfolgenden Worten entstehen.

Im folgenden soll der in der Vorverarbeitung für JANUS eingesetzte Clipping–Algorithmus kurz erläutert werden. Über das gesamte Sprachsignal wird dazu äquidistant im Abstand von 5 ms ein Fenster der Größe 256 geschoben, wie in Abbildung 3 dargestellt. Ein solches Fenster soll mit „FRAM" bezeichnet werden.

Eingabe: Tabelle mit allen Abtastwerten

Ausgabe: Index des ersten und letzten Werts, die als zum Sprachsignal gehörig angesehen werden.

1. *Schritt*: Punkt–zu–Punkt Amplituden
   Bestimme für jeden FRAM den minimalen und den maximalen Abtastwert und speichere die Differenz als „Punkt–zu–Punkt Amplitude" in einer Tabelle ab.
2. *Schritt*: Normierung
   Normiere die Punkt–zu–Punkt Amplituden auf das Intervall $[-1; +1]$.
3. *Schritt*: Detektion des Anfangs
   Suche nach einem Sprung innerhalb der Punkt–zu–Punkt Amplituden in einem zeitlichen Abstand von 100 ms, beginnend mit dem ersten Abtastwert. Dies liefert den Index des ersten FRAMs, der zum Sprachsignal gerechnet wird.
4. *Schritt*: Detektion des Endes
   Suche nach einem Sprung innerhalb der Punkt–zu–Punkt Amplituden in einem zeitlichen Abstand von 130 ms, rückwärts beginnend mit dem letzten Abtastwert. Dies liefert den Index des FRAMs, der den Abschluß des Sprachsignals bildet.

Abbildung 4 illustriert die Vorgehensweise.

Dieser Algorithmus basiert auf dem Signal im Zeitbereich. Man kann aber auch Schwellenbedingungen entwickeln, denen Parameter im Frequenzbereich zugrundeliegen.

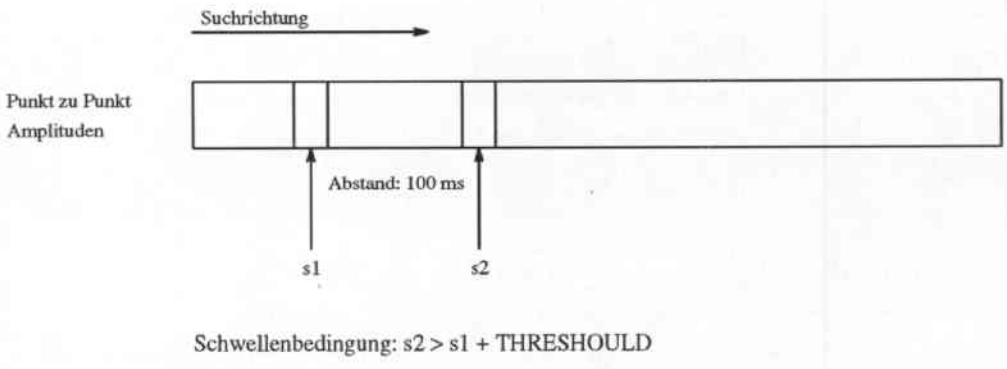

Abbildung 4: Schwellenbedingung für das Clipping

In [2] wird eine Möglichkeit, diesen Ansatz zu implementieren, näher ausgeführt. Folgende Vorteile des Verfahrens werden genannt:

- Die Empfindlichkeit gegen Rauschen ist geringer.
- Die Entscheidung ist unabhängig von der Signalamplitude.
- Das Verfahren ist robuster, da gemittelte Koeffizienten verwendet werden können.
- Die Schwellen sind leichter zu bestimmen.

2.4 Normalisierung

Eine Normalisierung der berechneten *melscale*-Koeffizienten ist erforderlich, um einerseits den als konstant angenommenen Pegel an Hintergrundgeräusch zu elemenieren und um andererseits dem Spracherkenner die Aufgabe nicht noch durch die stark unterschiedliche Aussteuerung der Aufnahmen zusätzlich zu erschweren.

Man kann sich viele Algorithmen ausdenken, die diese Aufgaben mehr oder weniger gut bewältigen. Im folgenden soll das Verfahren kurz vorgestellt werden, das der sequentiellen Datenvorverarbeitung für das JANUS–Spracherkennungssystem zugrunde lag.

Eingabe: Tabelle der 16 *melscale*-Koeffizienten für jeden FRAM

Ausgabe: Tabelle normalsierter *melscale*-Koeffizienten für jeden FRAM

1. *Schritt*: Mittelung
   Die *melscale*-Koeffizienten werden über die Zeit paarweise gemittelt. Als Ausgabe erhält man so für alle 10 ms[10] Spracheingabe, die im folgenden als "FRAME" bezeichnet werden sollen, einen Merkmalsvektor mit 16 Koeffizienten.

2. *Schritt*: Bestimmung des Skalierungsfaktors und des Grundpegels
   Unter allen 16 Koeffizienten aller FRAMEs werden die globalen Extrema gesucht. Das Minimum wird als Maß für den durch das Hintergrundgeräusch verursachten Grundpegel genommen, die Differenz von Maximum und Minimum bildet den Skalierungsfaktor.

3. *Schritt*: Normalisierung
   Die Koeffizienten aller FRAMEs werden um den Grundpegel verringert und durch den Skalierungsfaktor dividiert. Dadurch werden die gesamten Daten der Aufnahme auf das Intervall [0; 1] normiert.

2.5 Sequentielle Vorverarbeitung in JANUS

Zum Abschluß dieses Kapitels, das die Grundlagen der Vorverarbeitung von Sprachsignalen erläuterte, soll noch einmal zusammenhängend dargestellt werden, wie der Ablauf in JANUS bisher implementiert war.

Das Sprachsignal wurde mit 16 kHz abgetastet und in einem 14 Bit A/D–Wandler in digitale Daten gewandelt. Die gesamten digitalen Sprachdaten wurden an einem Stück aus dem Pufferbereich des Aufnahmegerätes in die Workstation geladen, auf der die Vorverarbeitung und Spracherkennung zur Ausführung kam. Über die gesamte Aufnahme wurde äquidistant alle 5 ms ein Fenster konstanter Breite geschoben und die Minima und Maxima des Sprachsignals innerhalb der Fenster berechnet (Punkt–zu–Punkt Amplituden). Anschließend wurden diejenigen Teile am Anfang und am Ende abgeschnitten, die unter einem bestimmten Aussteuerungspegel liegen. Mit den verbleibenden Daten wurde eine Spektralanalyse durchgeführt (wie oben beschrieben), die berechneten *melscale*-Koeffizienten normalisiert und als Ausgabedaten an einem Stück in eine Datei geschrieben, um dann vom Erkenner weiterverarbeitet zu werden.

In Abbildung 5 ist der Ablauf der bisherigen sequentiellen Vorverarbeitung in JANUS dargestellt.

[10] Für Anwendungen in der Spracherkennung hat sich dieses Abtastintervall für den Frequenzbereich bewährt.

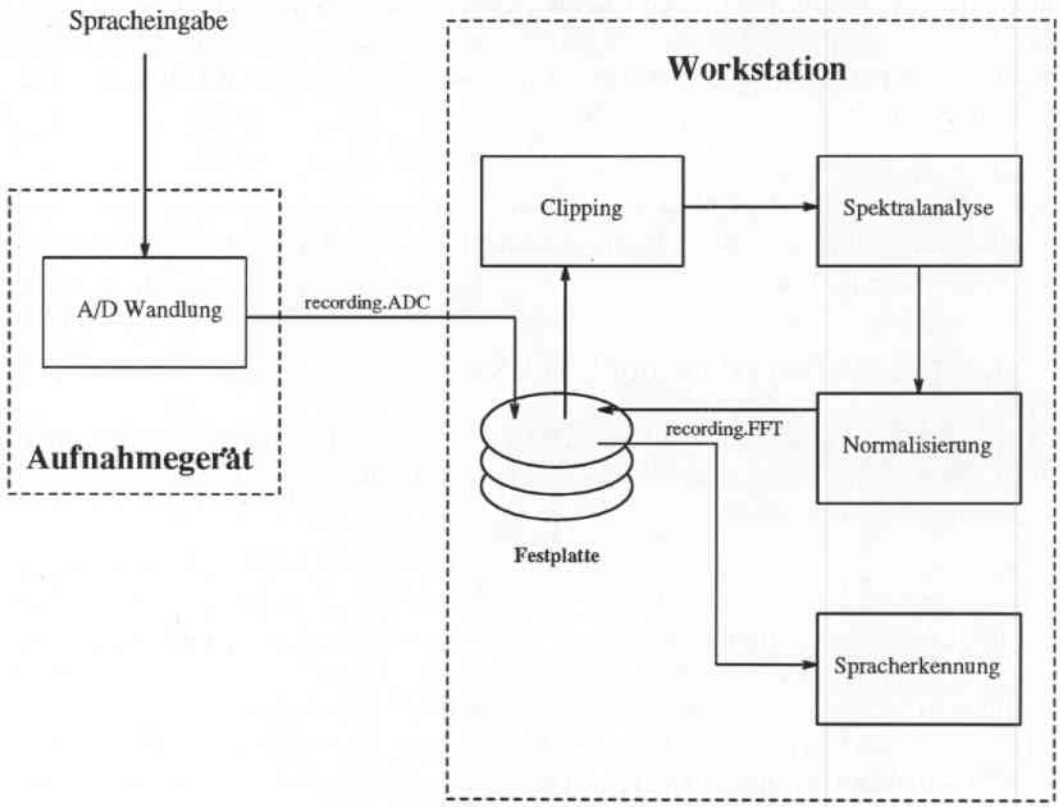

Abbildung 5: Sequentielle Vorverarbeitung in JANUS

3 Fließbandverarbeitung in der Vorverarbeitung

Die sequentielle Vorverarbeitung der Sprachsignale in JANUS begann mit der eigentlichen Bearbeitung des digitisierten Signals erst, nachdem der Sprecher die Aufnahme beendet hat. Dies hatte zur Folge, daß in einer beschleunigten Implementierung als Teil einer Diplomarbeit [4] die Verzögerung durch die Vorverarbeitung immer noch in der Größenordnung der Dauer der Spracheingabe selbst lag.
Eine Vorverarbeitung in Echtzeit konnte also nur durch die Anwendung von Fließbandverarbeitung erwartet werden.

In diesem Kapitel wird für jeden Vorverarbeitungsschritt im einzelnen diskutiert, ob überhaupt und in welchem Umfang die Umstellung auf Fließbandverarbeitung möglich ist. Zum Abschluß wird die in dieser Studienarbeit implementierte Fließband erläutert.

3.1 Digitisierung des Sprachsignals

Das Aufnahmegerät, das für Spracheingaben in JANUS verwendet wird, puffert die Abtastwerte als 16-Bit quantisierte Werte in einem geräteeigenen RAM–Bereich. Für die Weiterverarbeitung der Abtastwerte ist es also erforderlich, sie vom Aufnahmegerät in die Workstation zu übertragen. Dies ist auf zwei Arten möglich:

- nach Abschluß der Aufnahme alle Abstastwerte an einem Stück
- noch während der laufenden Aufnahme paketweise mit einstellbarer Paketgröße.

Beide Übertragungsmodi sind als Bibliotheksroutinen in der Software des Aufnahmegerätes implementiert; der erstere wurde in der sequentiellen Vorverarbeitung eingesetzt, der letztere ist Voraussetzung dafür, die Vorverarbeitung überhaupt auf Fließbandverarbeitung umzustellen.

3.2 Spektralanalyse

Die Spektralanalyse umfaßt das Ausblenden einer Folge von Fenstern aus dem Sprachsignal, das Berechnen der FHT über jedem Fenster und die Umrechnung der Spektralkoeffizienten in die *melscale*-Koeffizienten.
Da für jede dieser Operationen nur die zu einem Fenster gehörenden Daten erforderlich sind, ist die gesamte Spektralanalyse geradezu für Fließbandverarbeitung prädestiniert.

Die einzige Schwierigkeit ist die Kooperation zwischen Aufnahmegerät und Workstation auf den gemeinsamen Sprachdaten. In dieser Studienarbeit konnte das gesamte Fließband in einem Prozeß implementiert werden, da die Routinen, die die Übertragung der Sprachdaten vom Aufnahmegerät zur Workstation ausführen, und die Routinen, die die Spektralanalyse durchführen, zu einem Modul zusammengebunden werden konnten.
Die Sprachdaten, über denen die Kooperation durchgeführt wird, liegen in einem Puffer im RAM der Workstation. Auf sie werden über zwei Indizes zugegriffen; zum einen durch die

Übertragungsroutinen und zum anderen durch die Routinen, die aus den Sprachsignalen die Spektralkoeffizienten berechnen. Das Fließband kann also in einer Schleife folgender Struktur implementiert werden:

```
WHILE Aufnahme nicht gestoppt
DO IF neues Paket liegt zur Übertragung bereit
     THEN Übertrage Paket und hänge es an den beschriebenen Pufferbereich an
   FI
   Fahre mit der Spektralanalyse nicht bearbeiteter Sprachsignale im Puffer fort
OD
```

Eine (anspruchsvollere) Implementierungsalternative ist, für jedes Glied des Fließbandes einen eigenen Prozeß vorzusehen und deren Interaktion als Erzeuger-/Verbraucher–Beziehung zu gestalten.
Das Aufnahmegerät erzeugt die Abtastwerte, die von dem Prozeß "verbraucht" werden, der die Spektralanalyse durchführt, u.s.w.

Dieser zusätzliche Aufwand lohnt sich z.B., wenn die Ausführung von Übertragungsroutinen und FHT–Routinen durch die Workstation zusammen länger dauert als die dazu parallel durch das Aufnahmegerät durchgeführte Digitisierung der Spracheingabe. Dies würde zur Folge haben, daß die Vorverarbeitung des letzten Pakets Spracheingabe noch nicht abgeschlossen ist, wenn das nächste Paket schon im Puffer des Aufnahmegerätes zur Abholung bereit liegt.
Dies ist allerdings in der Implementierungsumgebung dieser Studienarbeit[11] nicht der Fall. Ferner kann es aus technischen Gründen wünschenswert sein, die Spektralanalyse nicht auf dem Rechner auszuführen, an dem das Aufnahmegerät angeschlossen ist. Dies kann z.B. dann der Fall sein, wenn man mehrere Mikrophone an das Aufnahmegerät angeschlossen hat und dadurch die Workstation zum Engpaß wird.

3.3 Normalisierung

Die Umstellung der Normalisierung auf Fließbandverarbeitung bereitet mehr Schwierigkeiten.

Der triviale Algorithmus – anstelle über das gesamte Sprachsignal nur über ein Paket zu normieren – ist ungeeignet: Aufgrund des sich ständig ändernden Abstandes zwischen Mikrofon und Mund sowie der Aussprache des Sprechers schwankt die Aussteuerung des Sprachsignals innerhalb einer Aufnahme stark. Dadurch würden sich die Skalierungsfaktoren über die gesamte Aufnahme beträchtlich ändern.

Es ist möglich, die Normalisierung wie bisher sequentiell durchzuführen, d.h. nachdem das gesamte Sprachsignal aufgenommen und spektral analysiert wurde und bevor die Ergebnisdaten an einem Stück als Datei auf die Festplatte der Workstation geschrieben werden.

[11] DECstation 5000/200

Aufgrund des geringen Anteils der Normalisierung an der für die Vorverarbeitung erforderlichen Gesamtrechenzeit verzögert diese Vorgehensweise den Erkennungsprozeß kaum und ist daher auch vertretbar; allerdings nur solange die Spracherkennung noch sequentiell durchgeführt wird, d.h. erst beginnt, wenn die Spektralkoeffizienten der gesamten Aufnahme (in einer Datei) verfügbar sind.
Die angestrebte Spracherkennung in Echtzeit ist aber nur realisierbar, wenn auch die Spracherkennung auf Fließbandverarbeitung umgestellt wird. Daher wurden im Rahmen dieser Studienarbeit auch Methoden entwickelt, die es ermöglichen, die Normalisierung parallel zur Aufnahme durchzuführen.

Beim Vergleich der Skalierungsfaktoren, die das sequentielle Normalisieren verschiedener Aufnahmen eines Sprechers berechnet, fällt auf, daß diese – bei unveränderter Lautstärke–Regelung des Aufnahmegerätes – nur sehr wenig schwanken. Allerdings sind zwischen verschiedenen Sprechern, und vor allem wenn die Lautstärke–Regelung verändert wird, erhebliche Unterschiede festzustellen.
Dies legt folgende Vorgehensweise nahe:

1. *Schritt*: Einlernen der erforderlichen Normalisierungsparameter (Skalierungsfaktor und Grundpegel) anhand weniger unter gleichen Bedingungen aufgenommener Sätze, die sequentiell normalisiert werden.

2. *Schritt*: Normalisierung der folgenden Aufnahmen paketweise mit den eingelernten Parametern, ggf. mit Erfolgskontrolle und Korrektur der Parameter, falls sie sich als ungeeignet erweisen.

Die Implementierung dieses Algorithmus zeigt, daß eine Korrektur der eingelernten Parameter – die eine Wiederholung der Aufnahme notwendig machen würde – nur sehr selten erforderlich ist. Allerdings bleibt noch zu untersuchen, ob und wie weit die Erkennungsrate durch das neue Normalisierungsverfahren absinkt.
Es wäre im Sinne der für eine sichere Spracherkennung erforderlichen Unterdrückung der Sprechervariabilität wünschenswert, die Normalisierungsparameter während der laufenden Aufnahme an die sich ändernde Aussteuerung anzupassen, und zwar als zumindest stetige, besser noch differenzierbare Funktion der Zeit.
Eine Möglichkeit, diese adaptive Normalisierung zu implementieren, soll im folgenden angedeutet werden. Sie benötigt als Eingabedaten nur die Spektralkoeffizienten weniger FRAMEs, also deutlich weniger, als ein Fenster umfaßt[12].

1. Bestimme minimalen und maximalen Spektralkoeffizient der nächsten n FRAMEs.
2. Normalisiere den aktuellen FRAME anhand dieser Parameterschätzung.
3. Fahre mit dem auf den aktuellen FRAME folgenden FRAME fort.

3.4 Anfangs- und Endpunktbestimmung

Der im Kapitel 2.3 vorgestellte Clipping–Algorithmus ist für eine Umstellung auf Fließbandverarbeitung ungeeignet, da das verwendete Schwellenkriterium sich auf die über das

[12] Eine Implementierung für die Vorverarbeitung in JANUS steht noch aus.

gesamte Sprachsignal berechneten Punkt–zu–Punkt Amplituden bezieht.
Bereits dieses Verfahren schneidet in manchen Fällen zu viel vom Sprachsignal ab, z.B. wenn kurz vor Ende der Aufnahme noch sehr laut in das Mikrofon gesprochen wird und die letzten Laute nur noch ganz leise (daß diese Artikulation böswillig ist, sei dahingestellt). Das Clipping kann daher noch nicht als zufriedengestellt gelöst betrachtet werden.

Allerdings ist in der Befehlsbibliothek des Aufnahmegerätes eine Routine implementiert, die ohne Tastendruck durch den Sprecher die Digitisierung automatisch beginnt und wieder abschaltet, sobald die Aussteuerung einen bestimmten Pegel über– oder unterschreitet. Man könnte eine ähnliche Aufnahmesteuerung auch mit Hilfe der Übertragungsroutine implementieren, die die digitisierten Sprachsignale paketweise vom Aufnahmegerät in die Workstation überträgt, indem man in den übertragenen Sprachsignalen nach einem Sprung in der Aussteuerung sucht – ähnlich wie dies im Clipping–Algorithmus geschieht – und die eigentliche Vorverarbeitung erst nach Auffinden eines solchen Sprungs anlaufen läßt[13].

3.5 Entwurf des Fließbandes

Die Implementierung der Vorverarbeitung im Rahmen dieser Studienarbeit geht von folgenden Voraussetzungen aus:

- Der Spracherkenner arbeitet noch sequentiell; er erwartet die Spektralkoeffizienten der gesamten Aufnahme in einem File "recording.FFT".
- Die Rechenleistung der Workstation ist so hoch, daß die einzelnen Vorverarbeitungsschritte (Spektralanalyse und Normierung) in einem Prozeß ausgeführt werden können.

Abbildung 6 zeigt den Entwurf schematisch.

4 Ergebnisse

4.1 Was hat's gebracht?

Ziel dieser Studienarbeit war, die Vorverarbeitung von Sprachsignalen in Echtzeit ausführen zu können.

Aus Tabelle 4, die auf Zeitmessungen an einer DEC 5000/200 beruht, läßt sich entnehmen, daß dieses Ziel als erreicht angesehen werden kann[14]. Wenn die Fließbandverarbeitung konsequent in den Spracherkenner fortgeführt wird, entfällt das Clipping und die Notwendigkeit, die Spektraldaten des gesamten Sprachsignals in eine große Datei zwischenzuspeichern. Ferner käme nur noch paralleles Normalisieren in Betracht. Es würde

[13] Diese Idee wurde im Rahmen dieser Studienarbeit noch nicht implementiert.

[14] Verzögerungen werden ab dem Ende der Spracheingabe gerechnet und aufaddiert. Die Option -n beinhaltet sequentielles oder paralleles Normalisieren; die Option -c beinhaltet Clipping.

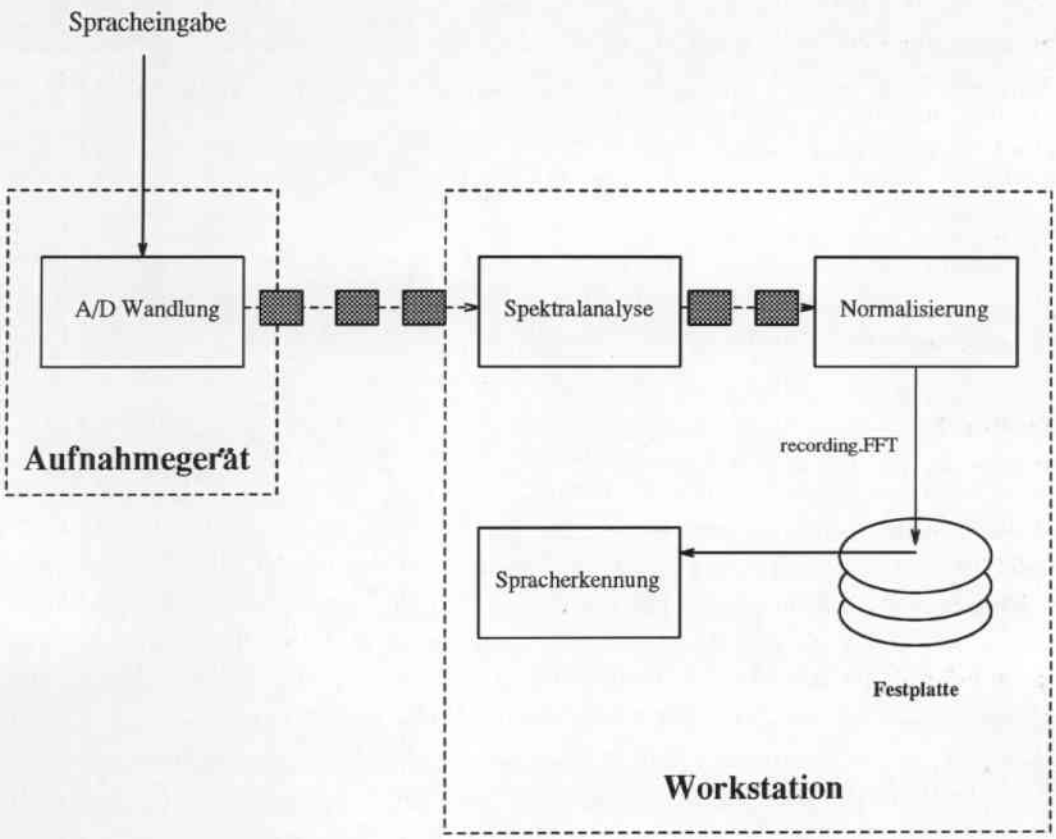

Abbildung 6: Vorverarbeitungs-Fließband für JANUS

Optionen	Aufnahmezeit	Verzögerung bis FHT	Verzögerung bis FFT–Datei
-nS	1.92	0.12	0.43
-nS	5.13	0.12	0.55
-nS -cC	1.93	0.12	0.77
-nS -cC	5.38	0.12	1.00
-nP	2.20	0.12	0.37
-nP	5.39	0.12	0.46
-nP -cC	1.93	0.12	0.70
-nP -cC	4.59	0.13	0.86

Tabelle 4: Ausführungszeiten für die Vorverarbeitung in Fließbandtechnik

also nur noch die Verzögerung bis zum Abschluß der Berechnung der FHT anfallen (siehe dritte Spalte in Tabelle 4).

Ferner wurde mit der Einführung eines Verfahrens zum parallelen Normalisieren die Voraussetzung dafür geschaffen, die gesamte Spracherkennung unter Anwendung von Fließbandverarbeitung in Echtzeit auf einer Workstation zu implementieren.

4.2 Ausblick

Aus dem bisher Gesagten folgt als nächste Aufgabe, auch den Spracherkennungsprozeß auf Fließbandverarbeitung umzustellen.
Diese Aufgabe stellt wesentlich höhere Anforderungen an die Implementierungstechnik; denn die Rechenleistung heutiger Workstation wird es nicht zulassen, auch den gesamten Erkennungsalgorithmus mit nur einem Prozeß zu realisieren und trotzdem Echtzeitanforderung zu erfüllen. Es wird vielmehr erforderlich sein, für Vorverarbeitung und Erkennung mehrere Prozesse vorzusehen, die über gemeinsamen Daten kooperieren.

Eine Nebenbemerkung sei mir zum Schluß noch erlaubt: Keine Wissenschaft ist so kurzlebig wie die Informatik. Man muß in seiner Forschungsarbeit ständig darauf gefaßt sein, daß neue Entwicklungen die eigene Arbeit zu Makulatur werden lassen.
Dieses Schicksal steht auch der im Rahmen dieser Studienarbeit erstellten Software bevor; denn die Firma Gradient, die das in JANUS eingesetzte Aufnahmegerät herstellt, hat inzwischen eine Ergänzungsplatine mit einem digitalen Signalprozessor entwickelt, der in C formulierte Routinen in Fließbandtechnik abarbeiten kann.

A Hinweise zur Benutzung der Software

Im folgenden sollen Aufruf und Benutzerführung der im Rahmen dieser Studienarbeit erstellten Software kurz erläutert werden. Es sei darauf hingewiesen, daß die Software nicht konzipiert wurde, um eine möglichst komfortable Mensch–Maschine Schnittstelle zu besitzen, sondern sich darauf beschränkt, die wesentliche Funktionalität so zur Verfügung zu stellen, daß möglichst wenig Interaktion mit dem Benutzer erforderlich ist.

A.1 Aufruf des Aufnahmeprogramms

Name
pipe_record - Aufnahme und Vorverarbeitung von Spracheingaben mit dem ∇ Gradient DeskLab [15]

Syntax
pipe_record [**options**]

Beschreibung
Spracheingaben werden über ein Mikrofon aufgenommen, durch das ∇ Gradient DeskLab digitisiert, paketweise in die Workstation über den SCSI–Bus übertragen, dort vorverarbeitet und standardmäßig in die Datei "recording.FFT" des aktuellen Verzeichnisses geschrieben. Der Abschluß der Vorverarbeitung einer Aufnahme wird durch die Semaphordatei mit der Bezeichnung "recording.FFT-ready" angezeigt. Dies dient zur Synchronisation zwischen Aufnahmeprogramm und Spracherkenner.
Die verschiedenen Aufnahmeoptionen können teilweise über ein Menü geändert werden.

Mit Rücksicht auf die Kollegen an der Carnegie Mellon University, die mit diesem Aufnahmeprogramm ebenfalls umgehen müssen, ist der gesamte Benutzerdialog in Englisch gehalten.

Optionen

[-a]
Erzeugen einer Datei mit den unverarbeiteten Abtastwerten der Aufnahme.
Diese Datei erhält standardmäßig den Namen "recording*(aktuelle Nummer)*.ADC", wobei die Aufnahmen seit dem Start des Programms automatisch hochgezählt werden. Sie wird in das aktuelle Verzeichnis geschrieben.
Der Name der Datei wird allerdings über die Option **-f** angegeben.

[-f *Dateiname*]
Ändern des standardmäßigen Namens der Datei, die die Spektralkoeffizienten der gesamten Aufnahme enthält. Die Namen der Semaphordatei und ggf. der Datei, in die die Abtastwerte geschrieben werden, werden entsprechend geändert.
Die Namen der Dateien werden gebildet, indem an die angegebene Zeichenkette *Dateiname*, die auch eine gültige Pfadangabe enthalten darf, die Endungen ".FFT", ".FFT-ready" bzw. ".ADC" angehängt werden.

[15] ©1990, Gradient Technology, Inc.

[-g *Verstärkung*]
Die standardmäßige Voreinstellung der Verstärkung für die Aufnahme kann durch Angabe eines Wertes aus dem Bereich 0 bis 255 geändert werden.
Die Standardeinstellung beträgt 100.

[-r]
Durch Angabe dieser Option wird der gesprochene Satz dem Benutzer noch einmal vorgespielt, nachdem die Datei mit den Spektralkoeffizienten erzeugt worden ist.

[-l]
Diese Option bewirkt, daß während der Aufnahme die Aussteuerung kontrolliert wird. Liegt sie außerhalb des Bereiches von 20 % bis 99 %, so wird die gesamte Aufnahme zurückgewiesen.

[-n*Normalisierungsmodus*]
Je nach Angabe des *Normalisierungsmodus* wird die Normalisierung **(S)**equentiell oder **(P)**arallel durchgeführt. Die Standardeinstellung ist sequentielle Normalisierung.

[-c*Clippingmodus*]
Es wird an die Vorverarbeitung noch eine Detektion des Anfangs und des Endes des Sprachsignals angehängt, je nach Angabe des *Clippingmodus* mit einer feinen **(F)**ine oder einer groben **(C)**oarse Schwellenbedingung.
Standardmäßig wird auf das Clipping verzichtet.

A.2 Benutzung des Aufnahmeprogrammes

Die Benutzerführung des Programmes wurde so konzipiert, daß

- mit einem Minimum an Eingaben die entscheidenden Parameter beeinflußt werden können und
- alle für den mit dem System Vertrauten wichtigen Informationen über den Programmzustand prägnant dargestellt werden.

Die drei Hauptelemente der Benutzerführung sind:

1. das Steuermenue
2. das Menue zum Einstellen der Optionen und Programmparameter
3. Kurzmeldungen über den Programmzustand

Im folgenden soll der Benutzerdialog innerhalb dieser Hauptelemente kurz erläutert werden.

1. Das Steuermenue

[Wann erscheint es?]
vor jeder neuen Aufnahme

[Aufbau]

```
Press
  <RETURN> to begin recording
  <m> to switch to menue
  <e> to exit
```

[Beschreibung]
Mit <**RETURN**> beginnt unmittelbar die Abtastung der mit dem Mikrofon aufgenommenen Signale.
Mit <**m**> erreicht man das Menue, das das Einstellen von Programmparametern wie Vorverstärkung *(gain)*, Normalisierung und Clipping ermöglicht.
Mit <**e**> verläßt man das Programm.

2. Einstellen der Programmparameter

[Wann erscheint es?]
nach Wahl von <m> im Steuermenue

[Aufbau]

```
Choose
  <g> to set the gain
  <n> to change normalization mode (not, sequential or parallel)
  <c> to change clipping mode (not, coarse or fine)
  <r> to change replaying mode
```

[Beschreibung]
Mit <**g**> kann man die Vorstärkung des Aufnahmegerätes einstellen auf einen Wert zwischen 0 und 255.
Mit <**n**> kann man die Art der Normalisierung beeinflußen. Sequentielle Normalisierung bedeutet, daß erst nach Abschluß der Aufnahme die Daten an einem Stück normalisiert werden; parallele Normalisierung erfolgt dagegen als ein Glied des Fließbandes – nach einer Einlernphase, in der die Parameter anhand sequentieller Normalisierung eingelernt werden.
Mit <**c**> kann man die Art des Clipping beeinflußen. Feines Clipping benutzt eine schwächere Schwellenbedingung und schneidet daher einen kleineren Teil des Sprachsignals am Anfang und am Ende ab als das grobe Clipping.
Mit <**r**> kann man erreichen, daß die Aufnahme über den Kopfhörer nocheinmal eingespielt wird.

3. Meldungen während des Programmablaufs

Das Programm gibt zahlreiche Meldungen, die über den aktuellen Zustand informieren, Warnungen ausgeben oder Fehler anzeigen.

- *Meldungen von initialen Tests*

  [Wann erscheinen sie?]
  direkt nach dem Programmstart

  [Aufbau] `>>>>> WARNING: The file recording.FFT already exists. <<<<<`

  [Beschreibung]
  Sie haben einen Dateinamen angegeben, der schon belegt ist. Wenn Sie daher verhindern wollen, daß die bestehende Datei überschrieben wird, müssen Sie das Programm unterbrechen (mit CTRL-C).

  [Aufbau]

  `>>>>> ERROR: Please run <source gstart> to initialize the system . <<<<<`

  [Beschreibung]
  Sie haben vergessen, das Aufnahmegerät zu initialisieren. Das Programm bricht ab, um Ihnen Gelegenheit zu geben, dies nachzuholen.

- *Kurzmeldungen*
  Im folgenden soll für die wichtigsten Kurzmeldungen kurz Zeitpunkt und Art ihres Erscheinens erläutert werden.

  [Aufbau]

  `>>>>> Press <RETURN> to continue <<<<<`

  [Wann erscheint sie?]
  zwischen den Aufnahmen, vor Erscheinen des Steuermenues

  [Beschreibung]
  Das Programm wartet bei blockierter Tastatur, damit die CPU bis kurz vor Beginn der nächsten Aufnahme nicht durch das Abfragen der nicht–blockierenden Tastatur unnötig belastet wird.

  [Aufbau]

  `recording (blocksize is 4096 samples) .....(done)`

  [Wann erscheint sie?]
  während einer Aufnahme

  [Beschreibung]
  Jeder Punkt zeigt die Bearbeitung eines Blocks Sprachdaten an. Der Abschluß der Aufnahme wird mit "(done)" angezeigt.

[Aufbau]

```
>>>>> Normalization done <<<<<
```

[Wann erscheint sie?]
nach der Aufnahme, nur bei sequentiellem Normalisieren oder in der Einlernphase des parallelen Normalisierens

[Aufbau]

```
>>>>> Clipping begun <<<<<
      total number of samples: 65200
      after clipping: first sample = 4230
                      last sample = 60150
```

[Wann erscheint sie?]
nach der Aufnahme, wenn *clipping* angeschaltet ist

[Aufbau]

```
>>>>> Replaying what you said <<<<<
```

[Wann erscheint sie?]
nach der Aufnahme, wenn *replaying mode* angeschaltet ist

[Aufbau]

```
>>>>> Writing recording.ADC (done) <<<<<
```

[Wann erscheint sie?]
nach der Aufnahme, wenn das zusätzliche Abspeichern der unverarbeiteten Sprachdaten als Option gewählt wurde.

[Aufbau]

```
>>>>> Writing recording.FFT <<<<<
```

[Wann erscheint sie?] nach der Aufnahme um anzuzeigen, daß die Datei mit den verarbeiteten Sprachdaten auf die Festplatte geschrieben werden.

- *Warnungen und Fehlermeldungen*
  Während des Programmlaufs können zahlreiche Warnungen und Fehlermeldungen erscheinen; die meisten von ihnen treten aber nur bei Systemfehlern auf, die vom Programm aus nicht behoben werden können.
  Auf eine Beschreibung der möglichen Gegenmaßnahmen wird im Rahmen dieser Ausarbeitung verzichtet.

B Kurzbeschreibung der Software

Die im Rahmen dieser Studienarbeit erstellte Software ist modular aufgebaut, damit die verschiedenen Funktionalitäten leicht wiederverwendbar sind. Im einzelnen gliedert sie sich in die folgenden Module:

- pipe_record.c
- FFT.c
- clipping.c
- terminal.c

Im folgenden sollen die einzelnen Module und ihre wichtigsten Parameter kurz beschrieben werden.

Hierbei werden alle Prozedurbezeichner fett und alle Konstantenbezeichner kursiv dargestellt.

B.1 pipe_record.c und terminal.c

Das Modul "terminal.c" stellt Routinen zur Verfügung, die ein Schreiben von Dateien unabhängig vom Typ der Workstation (RT_SUN, PMAX oder VAX) im gleichen Format ermöglichen. [16]
Ferner ermöglichen die Routinen **block_terminal** bzw. **deblock_terminal** das An- und Abschalten des blockierenden Tastaturzugriffs.

Das Modul "pipe_record.c" enthält das Hauptprogramm mit der Aufnahmeschleife. Es wurde darauf geachtet, daß der gesamte Benutzerdialog möglichst in eigenen Routinen eingekapselt ist, so daß ein Einbinden in X–Umgebung keine großen Schwierigkeiten machen sollte.

Im folgenden sollen die wichtigsten Programmkonstanten kurz vorgestellt werden:

- *DSP_WND_SIZ* — Größe des Fensters
- *SAMPL_RATE* — Abtastrate in kHz
- *BUFFER_LEN* — Anzahl der Abtastwerte in einem Paket
- *FRAM_RAT* — Abstand zwischen zwei aufeinanderfolgenden Fenstern in ms
- *MAX_TIME* — maximale Länge einer Aufnahme in Sekunden
- *LEARN* — Anzahl der Aufnahmen, anhand derer beim parallelen Normalisieren die Parameter gelernt werden

B.2 FFT.c

Dieses Modul enthält die für die Berechnung der FHT erforderlichen Routinen.

DSP_WND_SIZ legt die Größe des Fensters fest. Sie muß nicht notwendig eine Zweierpotenz sein; in diesem Falle berechnet **find_log** die nächst kleinere Zweierpotenz.

Die Funktion der anderen Routinen sind:

[16] Dazu wird die Reihenfolge von High- und Lowbyte gegebenenfalls vertauscht.

- **init_fht** initialisiert die Nachschlagetabellen der Sinus- und Kosinuswerte sowie inversen Elemente modulo der Fenstergröße, die für die nachfolgende Hartley Transformation benötigt werden.
- **ham** schneidet ein Teilstück aus dem Signal an der momentanen Position mit Hilfe einer Hamming–Fensterfunktion.
- **fht** führt die Hartley Transformation auf dem aktuellen Fenster durch.
- **fht_pow_spec** berechnet aus den Spektral- die Leistungskoeffizienten.

B.3 clipping.c

Dieses Modul enthält alle Routinen, die für die Detektion von Anfang und Ende des Sprachsignals erforderlich sind.

Die Routine **clip** führt die erforderlichen Schritte nacheinander durch. Sie bekommt die Abtastwerte der gesamten Spracheingabe im Array **ad_buf** übergeben.

Zuerst werden in **ptp_am** die Extrema in den Abtastwerten innerhalb jedes Fensters berechnet, das man erhält, wenn man es äquidistant alle 5 ms über das Sprachsignal schiebt.

begend_det bestimmt anhand von zwei Schwellwertkriterien je zwei Indizes für den Anfang und das Ende des Sprachsignals innerhalb von **ad_buf**. Folgende Konstanten bestimmen die Schwellwertkriterien:

- *TIM_BEG_INT* gibt den Abstand in ms an, in dem nach einem Sprung in den oben bestimmten Punkt–zu–Punkt Amplituden vom Anfang her gesucht wird; *TIM_END_INT* gibt das entsprechende für das Signalende an.
- *THRESH1* bestimmt das feine Schwellenkriterium; *THRESH2* das grobe.

Je nach Wahl des Clipping–Modus wird anschließend entweder die grobe oder die feine Schwelle benutzt, um den Signalanfang und das Signalende festzulegen.

C Listing

Auf den folgenden Seiten wird das Listing der im Rahmen dieser Studienarbeit erstellten Software wiedergegeben. Einige Routinen wurden aus bestehenden Programmen übernommen; die verschiedenen Autoren sind in den jeweiligen Prozedurköpfen dokumentiert.

Literatur

[1] G. Ruske: *Automatische Spracherkennung,* München, Wien: Oldenbourg (1988)

[2] A. Waibel, B. Yegnanarayana: *Comparative Study of Nonlinear Time Warping Techniques in Isolated Speech Recognition Systems,* Carnegie-Mellon University (1981)

[3] R. N. Bracewell: *Schnelle Hartley-Transformation,* München, Wien: Oldenbourg (1990)

[4] T. Sloboda: *Algorithmen der Spracherkennung auf massiv parallelen SIMD - Rechnern,* Diplomarbeit Universität Karlsruhe (Januar 1992)

[5] A. Waibel, A. Jain, A. McNair, H. Saito, A. Hauptmann, J. Tebelskis: *JANUS: A speech-to-speech Translation System Using Connectionist and Symbolic Processing Strategies, ICASSP 91, Volume 2, IEEE Mai 1991*