

Colloquium Signaalanalyse en Spraak

Citation for published version (APA):

ten Bosch, L. F. M., Eggen, J. H., Verhelst, W. D. E., & Willems, L. F. (editors) (1990). *Colloquium Signaalanalyse en Spraak: 22 en 23 oktober 1990 : reader*. (IPO rapport; Vol. 765). Instituut voor Perceptie Onderzoek (IPO).

Document status and date:

Gepubliceerd: 03/10/1990

Document Version:

Uitgevers PDF, ook bekend als Version of Record

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Instituut voor Perceptie Onderzoek
Postbus 513, 5600 MB EINDHOVEN

LtB/lTB 90/13
03.10.1990

Rapport no. 765

Reader COLSAS 1990
22-23 oktober 1990

L.F.M. ten Bosch
J.H. Eggen
W.D.E. Verhelst
L.F. Willems

Colloquium Signaalanalyse en Spraak

Colloquium **Signaalanalyse
en Spraak**

22 en 23 okt. '90

r e a d e r

Georganiseerd door het IPO m.m.v.

Louis ten Bosch

Berry Eggen

Werner Verhelst

Lei Willems

logo SAS D.J.M. Weenink (Amsterdam)
N.E.Tebak (IPO Eindhoven)

lay-out M.E. de Vette (IPO Eindhoven)

Alle bijdragen zijn door de auteurs voor opname
in deze reader ter beschikking gesteld.
Vermenigvuldiging van de bijdragen is niet
toegestaan zonder toestemming van de
desbetreffende copyright-houder.

Inleiding

Voor u ligt de reader van het vierde Colloquium Signaalanalyse en Spraak, dat plaats vindt op 22 en 23 oktober 1990. Na de succesvolle Colloquia in Amsterdam, Den Haag en Utrecht is het de beurt aan het IPO om dit jaar gastheer te zijn. Het Colloquium is bedoeld als discussieplatform voor deskundigen op het gebied van signaalanalyse enerzijds en spraak anderzijds. Met deze doelstelling voor ogen hebben wij een aantal onderwerpen in het programma opgenomen die passen in de trend van het huidige spraakonderzoek. Wij hopen dat deze trend voldoende in de keuze van de artikelen naar voren komt.

Het Colloquium is verdeeld in vier secties: Codering, Bijzondere onderwerpen, Spraakherkenning, Spraaksynthese.

codering In de sectie Codering zal worden ingegaan op spraak- en muziekcodering (maandagochtend), en op de toepassing in spraak van Scale Space Codering (SSC), een techniek die al langer uit de beeldcodering bekend is. Ook gaan we dieper in op de principes achter de Singular Value Decomposition (SVD).

bijzondere onderwerpen Een aantal onderwerpen is samengevat onder de noemer 'Bijzonder' (maandagmiddag). Onder deze noemer wordt aandacht gevraagd voor stromingsverschijnselen, mogelijke toepassingen van fractals, de nieuwe PSOLA-manipulatietechniek, en de achtergronden van regelsets bij spraaksynthese.

spraakherkenning In de sectie Spraakherkenning (dinsdagochtend) zal het vooral gaan om een vergelijking tussen de concurrerende benaderingen van Hidden Markov Models (HMM) en Neural Networks (NN), en de prestaties van HMM.

spraaksynthese De sectie Spraaksynthese (dinsdagmiddag) is in het bijzonder gericht op toepassingen (Spraakmaker, SPICOS), waarbij ook human interface-aspecten nader worden belicht. Daarnaast is er aandacht voor de methodologie van het vinden van duurregels in spraak.

Het organisatiecomité.

Inhoud

codering	R.J. Sluyter (Philips Research Eindhoven)	
	Spraakcodering	1
	R.N.J. Veldhuis (Philips Research Eindhoven)	
	Muziekcodering	2
	J.B. Martens (IPO)	
	Scale space codering	3
	E. Deprettere (TU Delft)	
	Toepassingen van SVD in spraak	4

bijzondere onderwerpen	A. Hirschberg (TU Eindhoven)	
	Geluidsproductie door stroming	5
	J. Grasman (LU Wageningen)	
	Fractale structuren in tijdreeksen: zelfsimilariteit, willekeur en chaos	6
	L. Vogten (IPO)	
	PSOLA-manipulatie van spraak	7
	L. ten Bosch (KU Nijmegen, UvA)	
	Data-regel-conversie voor spraak	8

spraakherkenning	C. Wellekens (Philips Research Louvain-la-Neuve)	
	Hidden Markov Models versus Multi-Layer Perceptrons	9
	P. van Alphen (Research Neher Lab. (PTT), UvA)	
	Hidden Markov Models en hun toepassing in spraakherkenning	10
	J.P. Martens (Univ. Gent)	
	Neurale netwerken in spraakherkenning	11
	D. v. Compernelle (Univ. Leuven, IPO)	
	Spraakherkenning in ruis-omgeving	12

spraaksynthese	B. van Coile (Univ. Gent)	
	Duurregels in spraaksynthese	13
	H. van Leeuwen (IPO)	
	Spraakmaker: van tekst naar spraak	14
	F. van Nes (IPO)	
	Toepassing van spraak bij mens-machine communicatie	15
	R. Collier (IPO)	
	SPICOS II: mens-machine dialoog	16

Codering

Perceptieve foutmaskering sleutel tot betere spraakkwaliteit

Digitalisering van spraak voor mobiele telefonie

In een cellulair radiosysteem communiceren de mobiele abonnees niet rechtstreeks, maar loopt de communicatie via een vast basisstation in het midden van een cel. Zo'n cel is een verzorgingsgebied met een bepaalde oppervlakte (in het verleden een stad of een deel van een provincie), waarbinnen bepaalde radiofrequenties voor de tweerichtingscommunicatie zijn toegewezen. Basisstations zijn onderling gekoppeld via schakelcentra die weer toegang hebben tot het openbare geschakelde telefoonnet (PSTN) en in de toekomst tot het "diensten geïntegreerde digitale netwerk (ISDN)".

Het "cellulaire radio"-concept biedt de mogelijkheid tot hergebruik van radiofrequen-

R.J. SLUIJTER

De auteur is werkzaam bij het Philips Natuurkundig Laboratorium te Eindhoven.

ties. Frequenties van de ene cel kunnen opnieuw gebruikt worden in op enige afstand gelegen cellen. Frequentiehergebruik vergroot de netwerkcapaciteit binnen een gegeven radioband. Er is een trend te bespeuren naar steeds kleinere cellen om steeds meer abonnees te kunnen bedienen. Bij kleine cellen - bijvoorbeeld met een diameter van en-

kele kilometers - wordt de kans groot dat gebruikers zich tijdens een gesprek van de ene cel naar de andere verplaatsen, zodat van basisstation en dus van zend- en ontvangsfrequentie moet worden omgeschakeld. Cellulaire radiosystemen met kleine cellen vereisen daarom een geavanceerd besturings- en beheersysteem om er voor te zorgen dat dit omschakelen ongemerkt gebeurt.

Nieuwe ontwikkelingen op het gebied van cellulaire radio gaan in de richting van digitale in plaats van analoge systemen. Digitale radiosystemen bieden een aantal voordelen namelijk:

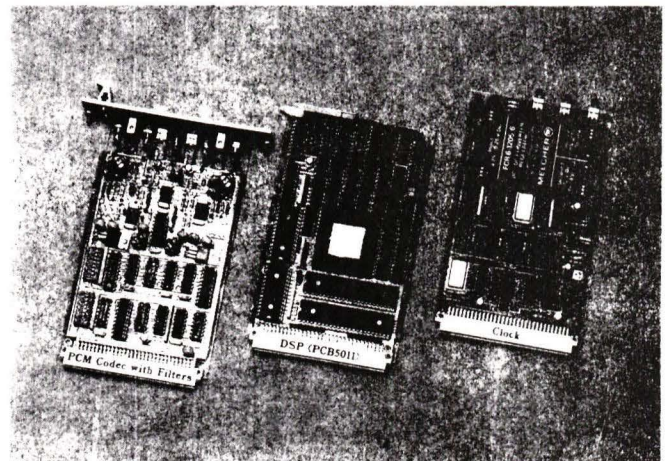
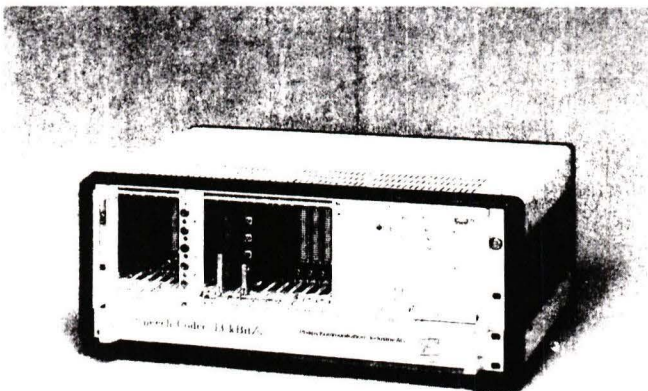
- verminderde gevoeligheid voor interferentie van nabuurlagen ("co-channel interference"). Frequenties kunnen daardoor eerder, dat wil zeggen op kortere afstand, worden hergebruikt, wat resulteert in een meer efficiënt gebruik van het radiospectrum;

Mobiele telefonie bestaat al een halve eeuw, maar slechts weinigen konden tot nu toe van deze dienst profiteren. In de toekomst zal echter iedereen, om het even waar, mobiele telefonieverbindingen tot stand kunnen brengen. Digitale cellulaire radio biedt de hiervoor noodzakelijke technische mogelijkheden.

- potentieel betere spraakkwaliteit;
- minder invloed van veldsterktevariëaties ("fading") ten gevolge van meerwegpropagatie door het gebruik van geavanceerde digitale signaalbewerking voor kanaalcodering en van spraakinterpolatie om periodes van diepe fading te overbruggen;
- meer "privacy" door versleuteling van digitale spraak;
- eenvoudige(r) integratie van spraak- en datadiensten (ISDN) en

Foto a,b:

Experimentele hardware van de GSM-spraakcodec uitgevoerd met een Philips signaalprocessor van het type PCB 5011. Een realisatie in de vorm van drie maat-IC's is op dit moment mogelijk. In de toekomst zal de codec op een enkele chip kunnen worden ondergebracht (foto's: Philips Kommunikations Industrie, Neurenberg, Duitsland).



- uitzicht op geavanceerde signaalbewer- kings- en besturings-IC's tegen lage kosten door gebruik van VLSI- technieken.

In 1991 zal in Europa een begin worden ge- maakt met de installatie van een systeem voor digitale mobiele radio in de 900MHz- band. De ontwikkeling van dit Pan-Europe- se systeem (GSM) zal gepaard gaan met la- gere kosten als gevolg van de mogelijkhe- den van massafabricage. Ook de internatio- nale bewegingsvrijheid van de abonnees zal erdoor worden vergroot.

Eisen spraakcodec

De spraakcodec (codeer/decodeerschake- ling) moet aan een aantal eisen voldoen te weten:

- een bitsnelheid van ongeveer 16 kbit/s als compromis tussen de prestaties van de huidige spraakcodecs en de vereiste radiospectrum-efficiëntie;
- een gemiddeld betere spraakwaliteit (onder praktische operationele condi- ties) dan de huidige, analoge 900MHz- systemen;
- transparantie voor de auditieve signale- ringssignalen van het netwerk naar de gebruiker zoals kiestoon, beltoon, bezet- toon enzovoort. Dit geldt ook voor even- tueel achtergrondlawaai dat bij de in- gangsspraak is opgeteld;
- minimale vertraging om echo's te voor- komen. Als bovengrens voor de ver- traging in een cascade van spraakcoder en decoder geldt een waarde van 65 ms;
- bitfoutenkansen tot 1% mogen geen al te grote invloed hebben op de gemiddelde spraakwaliteit en
- lage complexiteit. Bij voorkeur moet de codec in één IC te implementeren zijn.

Voor de selectie van een GSM-codec zijn een zestal kandidaat-systemen uit Frankrijk, Duitsland, Italië, Noorwegen, Zweden en het Verenigd Koninkrijk, onder meer via luisterproeven, intensief met elkaar verge- leken [1,2]. Uiteindelijk is de keus gevallen op het Philips LPC/RPE-systeem ("linear predictive coding/regular pulse excitation") aangevuld met het LTP-systeem ("long term prediction") van IBM. Alvo- rens het voorkeurssysteem in meer detail te bespreken, zal in het kort worden ingegaan op wat basiskennis over het spraakpercep- tiemodel.

Perceptiemodel

Figuur 1 toont een algemeen gebruikt model

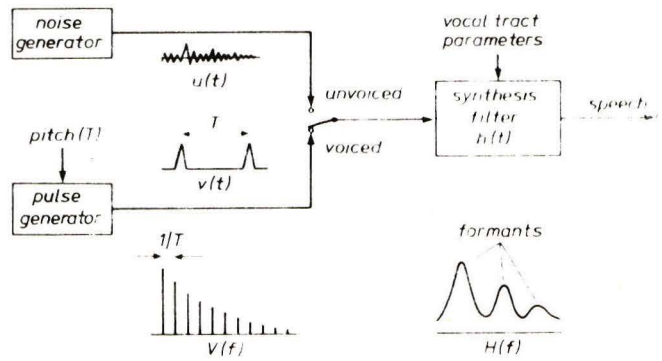


Fig. 1. Model van het menselijk spraakproductie- mechanisme.

van het spraakproductiemechanisme [3]. Dit model is deels gebaseerd op kennis van de menselijke spraakorganen en deels op de karakteristieke vormen van spraaksignalen. Het model bevat twee signaalbronnen: een (witte) ruisgenerator als stimulus voor het opwekken van stemloze klanken en een pulsgenerator voor stemhebbende klanken. De pulsgenerator, die in wezen model staat voor de menselijke stembanden, levert een serie periodieke pulsen $v(t)$, waarvan de herhalingsfrequentie $1/T$ overeenkomt met de fundamentele frequentie ofwel de *toon- hoogte* van de spraak. Het frequentiespec- trum $V(f)$ van dit signaal vertoont een lijnen- spectrum met componenten op de freque- tie-afstand $1/T$. De harmonische structuur hiervan wordt vaak aangeduid als de *spec- trale fijnstructuur* van de spraak. In het model wordt de *spectrale omhullende* van de geselecteerde bron bepaald door het synthe- sefilter, wat qua functie vergelijkbaar is met het menselijke stemkanaal dat bestaat uit mond-, neus- en keelholte. De frequentie- overdracht van het filter vertoont karak- teristieke resonantiepieken die *formanten* worden genoemd. In de spraakband van 0-4000 Hz kunnen zo'n drie tot zes formanten voorkomen. Iedere formant kan met twee parameters (frequentieligging en band- breedte) worden beschreven, zodat het syn- thesefilter met acht tot veertien parameters is te karakteriseren inclusief enkele additio- nele globale spectrale parameters. Het men- selijk spraakproductiemechanisme wordt door spierbewegingen bestuurd, hetgeen betekent dat de parameters slechts langzaam kunnen variëren met een tijdconstante in de orde van grootte van 10 ms.

Spraak nemen we waar met het oor. Over het functioneren van het menselijk oor is veel minder bekend dan over het spraakpro- duktiemechanisme. Desondanks is de ken- nis voldoende om met succes toegepast te worden in spraakbewerkingssystemen. Zo weten we bijvoorbeeld dat het oor relatief ongevoelig is voor de fase van de verschil- lende frequentiecomponenten in de spraak. Een van de meest nuttige eigenschappen is *auditieve masking*: sterke geluiden ver- minderen het vermogen tot waarnemen van

zwakke geluiden speciaal als ze in dezelfde frequentieband liggen. Dit kan men uitbui- ten om ruis en vervorming onder het signaal zelf te "begraven". Daar de formanten het leeuwendeel van de spraakenergie uitma- ken, spelen ze een belangrijke rol bij de ruismasking.

GSM-codec

In praktische systemen wordt de spraak, voorafgaande aan de verschillende bewer- kingsstappen, in segmenten verdeeld die achtereenvolgens worden bewerkt. De tijds- duur van deze segmenten is in de orde van grootte van 10 ms. Het synthesefilter bestaat vaak uit een recursief, tijddiscreet filter, waarvan de orde overeenkomt met het aan- tal formanten dat moet worden opgewekt. De overdrachtsfunctie $1/A(z)$ van zo'n syn- thesefilter heeft de vorm:

$$\frac{1}{A(z)} = \frac{1}{1 - \sum_{i=1}^p a(i)z^{-i}} \quad (1)$$

waarin p de orde van het filter is. De filter- coëfficiënten $a(i)$ zijn te bepalen op basis van *linear predictive coding* (LPC): een techniek die gebruik maakt van de "korte termijn correlatie" over p bemonsterperio- den in het spraaksignaal als gevolg van de formanten [4]. Zoals later zal blijken, kan dezelfde techniek worden toegepast om de spectrale fijnstructuur weer te geven in ter- men van de "lange termijn correlatie" over een aantal perioden van de grondfrequentie in het spraaksignaal. De resulterende pa- rameters worden aangeduid met de termen "korte termijn"- respectievelijk "lange ter- mijn"-parameters.

Figuur 2 geeft het blokdiagram van de GSM-codec, die werkt bij een bitsnelheid van 13 kbit/s [5]. De bemonsterfrequentie van het in- en uitgangsspraaksignaal is 8 kHz. De decoder bevat een spraaksynthesi- zer die bestaat uit een excitatiegenerator, een *long term prediction* (LTP) synthesefil- ter voor het opwekken van de spectrale fijnstructuur (toonhoogte) en een *short term prediction* synthesefilter op basis van LPC- technieken voor het genereren van de spec-

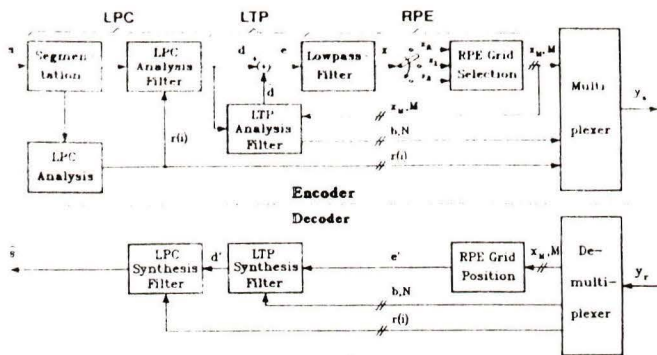


Fig. 2. Blokdiagram van de GSM-spraakcodec.

trale omhullende. Het LTP-synthesefilter heeft als overdrachtsfunctie:

$$\frac{1}{P(z)} = \frac{1}{1 - bz^{-N}} \quad (2)$$

waarin b de LTP-coëfficiënt [6] en N de periode van de grondfrequentie is. De "korte termijn"-parameters worden elke 20 ms bijgesteld; de "lange termijn"-parameters b en N elke 5 ms.

In de uiteindelijke spraakcodec is het synthesefilter als 8^e-orde ladderfilter uitgevoerd en niet in de directe vorm volgens vergelijking (1). Ladderfilters hebben betere stabiliteitseigenschappen in het bijzonder als de coëfficiënten zijn gekwantiseerd. De juiste set reflectiecoëfficiënten $\{r(i)\}$ van het ladderfilter is echter één-op-één gerelateerd aan de set $\{a(i)\}$. Het excitatiesignaal bestaat uit een serie equidistante pulsen die wordt verkregen door het bemonsterraster van 8 kHz met een factor 3 te decimeren. Deze excitatiemethode staat bekend als *regular pulse excitation* (RPE). Het RPE-systeem werkt met 5ms-subrasters. De decimatie kan in verschillende "fasen" worden gedaan die de posities van de pulsen binnen het raster bepalen. Deze fase of rasterpositie M en de amplitudes x_M van de pulsen worden per subraster van 5 ms ontvangen.

In de encoder worden achtereenvolgens de volgende bewerkingsstappen uitgevoerd:

- "linear predictive coding (LPC)";
- "long-term prediction (LTP)" en
- bepaling van de "regular pulse excitation (RPE)" pulsreeks.

Als eerste stap splitst men het signaal in segmenten van 20 ms. Vervolgens berekent men met LPC-analyse de acht reflectiecoëfficiënten $r(i)$ die gezamenlijk worden gekwantiseerd in 36 bits. De set reflectiecoëfficiënten wordt gebruikt in het LPC-analysefilter van het ladderfilter dat een overdrachtsfunctie $A(z)$ heeft. Dit verwijderd de "korte termijn"-correlatie van het spraaksignaal of, met andere woorden, zorgt voor een vlakke spectrale omhullende. Als twee-

de stap wordt de "lange termijn"-correlatie ofwel de spectrale fijnstructuur verwijderd met behulp van een adaptief LTP-filter met overdrachtsfunctie $P(z)$. Iedere 5 ms worden er nieuwe waarden van N en b berekend en gekwantiseerd in respectievelijk 7 en 2 bits. Na de uitgangssignalen d en \hat{d} van beide filters van elkaar afgetrokken te hebben, houden we een residu-signaal e over, dat veel lijkt op witte ruis, omdat er nauwelijks nog correlatie in het signaal aanwezig is. In de derde stap splitst men het residu-signaal weer op in subrasters van 5 ms. Elk subraster wordt gefilterd in een laagdoorlaatfilter en uit het uitgangssignaal x worden vier gedecimeerde kandidaatreeksen gegenereerd, waarvan er twee slechts aan de uiteinden verschillen volgens:

$$\begin{aligned} x_m(i) &= x(m+3i) \\ i &= 0, 1, \dots, 12 \\ m &= 0, 1, 2, 3 \end{aligned} \quad (3)$$

De rasterpositie M wordt bepaald aan de hand van de reeks met de grootste energie en in 2 bits gecodeerd. De pulsamplitudes x_M worden eenvoudigweg gehandhaafd en genormaliseerd met de maximum amplitude in elk blok van 5 ms. Elke genormaliseerde pulsamplitude wordt gekwantiseerd in 3 bits en het maximum van het blok in 6 bits. De parameters $r(i)$, b , N , en M alsmede de monsters x_M en de maximum amplitude worden tenslotte gemultiplext tot een bitstream met een bitfrequentie van 13 kbit/s.

Conclusie

Het basis-idee van de codeertechniek is het extraheren van de belangrijkste spraakarakteristieken in termen van filterparameters en wel in een zodanige vorm dat de spraak gereconstrueerd kan worden met behulp van een grof gekwantiseerd excitatiesignaal van lage bitsnelheid. De RPE-kwantiseringsmethode is gebaseerd op een perceptieve minimalisering van de codeerfout. Daar de analyse in wezen een spectrumvlakkende bewerking is en de kwantiseringsfout van het residu-signaal in principe ook een vlak spectrum heeft, krijgen zowel het spraaksignaal als de kwantiseringsruis in de synthesizer een identieke formantstructuur, waardoor een effectieve maske-

ring van de ruis optreedt. Het laagdoorlaatfilter in de RPE-coder heeft een speciaal gekozen frequentieresponsie om de ruismaskeringseigenschappen nog verder te optimaliseren. Een gedetailleerde beschrijving van dit maskeringsfenomeen is te vinden in referentie [7].

De GSM-spraakcodec levert een goede spraakkwaliteit die aanzienlijk beter is dan die van de huidige analoge cellulaireradio-systemen. De codec is in hoge mate transparant, zodat niet alleen de signaleringstonen, maar ook achtergrondlawaaï op natuurlijke wijze worden overgebracht. Circa 70% van de overgezonden informatie bestaat uit het predictieresidu. Daardoor is de foutgevoeligheid van de codec vergelijkbaar met die van de notoir robuuste golfvormcodecs. De complete, experimentele codec is met een Philips VLSI-sigitaalprocessor van het type PCB 5011 en wat standaardcomponenten gerealiseerd. Op basis van de huidige VLSI-technologie kan de hardware in drie "maat-IC's" worden ondergebracht namelijk een IC voor de A/D- en D/A-omzetting alsmede de filtering, een digitale signaalprocessor en een ASIC voor klokregeneratie en wat logische bewerkingen. In een komende generatie VLSI-technologie behoort een één-chips uitvoering tot de mogelijkheden. Aan het prototype met signaalprocessors is een signaalvertraging van 22 ms gemeten. In totaal zal de vertraging van de codec, inclusief extra foutcorrectie, zo'n 70 ms bedragen. ●

Dit artikel is gebaseerd op een voordracht tijdens het symposium "Mobiële Communicatie" van de TU Delft, 11 oktober 1989.

Dank is verschuldigd aan Karl Hellwig (PKI Neurenberg) voor zijn hulp bij het tot stand komen van dit artikel.

Referenties

[1] J. E. Natvig; *Evaluation of Six Medium Bit-rate coders for the Pan-European Digital Radio System*; IEEE Journal on Selected Areas in Communications, Vol. 6, No. 2, februari 1988.

[2] GSM Recommendation 06.10, Full Rate Speech Transcoding.

[3] R.J. Sluijter; *Digitalisering van Spraak*; Philips Techn.T. Jaargang 41, 1983, No. 7/8.

[4] J. Makhoul; *Linear Prediction: A Tutorial Review*; Proc. IEEE, Vol. 63, april 1975.

[5] P. Vary, K. Hellwig, R. Hoffmann, R.J. Sluijter, C. Galand en M. Rosso; *Speech Codec for the European Mobile Radio System*; Proc. of the IEEE Int. Conf. on ASSP, New York, 1988.

[6] P. Kroon en E.F. Deprettere; *A Class of Analysis-by-Synthesis Predictive Coders for High Quality Speech Coding at Rates between 4.8 and 16 kbit/s*; IEEE Journal on Selected Areas in Communication, Vol. 6, No. 2, februari 1988.

[7] P. Kroon, E.F. Deprettere en R.J. Sluijter; *A Novel Approach to Effective and Efficient Multipulse Coding of Speech*; IEEE Trans. on ASSP, Vo. 34, No. 5, 1986.

Subband Coding of Digital Audio Signals Without Loss of Quality*

Raymond N.J. Veldhuis, Marcel Breeuwer, Robbert van der Waal
Philips Research Laboratories, P.O. Box 80.000
5600 JA Eindhoven, The Netherlands

Abstract

A subband coding system for high-quality digital audio signals is described. To achieve low bit rates at a high quality level, it exploits the simultaneous masking effect of the human ear. It is shown how this effect can be used in an adaptive bit-allocation scheme. Results obtained with a low-complexity and a high-complexity system are discussed.

1 Introduction

Transmission and storage of high-quality digital audio is becoming important for the audio industry, for instance in the case of digital radio and new applications for optical disks. The bit rate of a high-quality stereophonic digital audio signal is about 1.4 Mbit/s. For some transmission channels or storage media this is too high and therefore source coding is required. Since digital audio is associated with high quality, a perceptible loss of quality cannot be tolerated.

Source coding of audio signals at low bit rates generally introduces errors. This paper describes a coding system that attempts to keep coding errors inaudible by exploiting the *simultaneous masking effect*. This is the perceptive phenomenon that a weak signal, e.g. quantization noise, is masked (= made inaudible) by a stronger signal, e.g. a pure tone in the audio signal. Simultaneous masking is briefly explained in Section 2.

Simultaneous masking is most effective if both masked and masking signal are in a rather narrow frequency band. This suggests the use of subband coding, where the signal is first split up into frequency bands which are then quantized. The structure of the subband coding system is given in Section 3.

Quantization should be such that the quantization noise is masked by the audio signal. This is achieved by using uniform APCM quantization [1]. The subband signals are split up into blocks. Each block is scaled to a unit level and then quantized by a uniform quantizer. Quantized data and scale factors are transmitted. In this manner the power of the quantization noise can be controlled by allocating a certain amount of bits to each quantizer.

*This paper has previously been published in [5].

In a subband coding system we can distinguish *in-band* masking, where both masked and masking signal are in the same subband, and *out-of-band* masking, where masking and masked signal are in different subbands. Both are exploited in the system described in this paper. Section 4 explains how for each subband the maximum power of the quantization noise that is masked, called the *masked power*, can be estimated.

Once the masked powers have been computed for all subbands, bits are allocated to the quantizers. Ideally the amount of bits for each quantizer should be such that the quantization noise is completely masked. However, the masked powers are signal-dependent and therefore the amount of bits needed to ensure complete masking varies in time. Because the coding system described here has a fixed bit rate, the bits must be divided over the subbands in such a way that the audible degradation of the output signal is minimal. This requires an adaptive bit-allocation technique, that is described in Section 5.

There is a trade-off between quality, bit rate, and complexity. Complexity is largely determined by the splitting and merging subband filters. It can be kept low by keeping the number of subbands low and their minimum bandwidth high. At a fixed quality level, the lowest bit rate achievable with a ‘low-complexity’ system is higher than with more complex systems with more and narrower subbands. This is explained in Section 4. Results obtained with a simple and a complex system are discussed in Section 6.

2 Simultaneous masking

Simultaneous masking is the effect that a weak signal is made inaudible by a simultaneously occurring stronger signal. Masking is discussed in great detail in [2,3]. The use of masking in subband coding is described in [4,6]. First consider the simple case of a pure tone as a masking signal. A signal component with a certain frequency is masked if the ratio of its power and the power of the masking tone is below the *masking threshold*. The masking threshold is a function of frequency. Figure 1 shows a stylistic approximation (on a dB scale) of the masking threshold for a pure tone of 1000 Hz at a sound pressure level of 0 dB. In general the masking threshold for a pure tone can be approximated by

$$T(f_m, f) = \begin{cases} T_{\max}(f_m) \left(\frac{f}{f_m}\right)^{28}, & f \leq f_m, \\ T_{\max}(f_m) \left(\frac{f}{f_m}\right)^{-10}, & f > f_m. \end{cases} \quad (1)$$

In this expression f_m is the frequency of the masking signal and $T_{\max}(f_m)$ is the masking threshold at this frequency. To simplify the masking model it is assumed that masking thresholds for tones of all frequencies have the same shape. However, $T_{\max}(f_m)$ depends on the frequency of the masking signal [4]. It is also assumed that the masking threshold is independent of the power of the masking signal. Furthermore, it is assumed that masking is additive: the masking threshold for a signal containing more than one frequency component can be obtained by adding the masking thresholds of the components.

The masking model used here is a simplification of reality. Coding systems based on it may show unexpected and unwanted effects. To avoid this, they must be tested

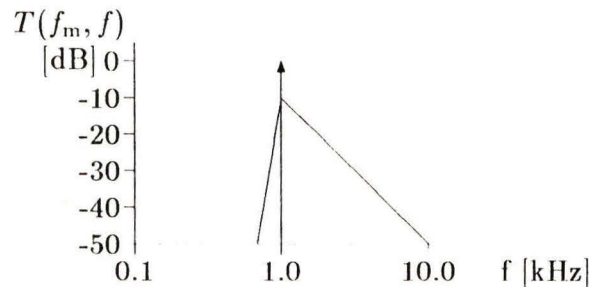


Figure 1: *Masking threshold as a function of frequency*

and optimized in extensive listening experiments.

3 Subband coding

It is clear from Figure 1 that masking is strongest for frequencies close to the frequency of the masking signal. This suggests that the masking phenomenon can be well exploited in a subband coding system. In such a system the signal is split up into frequency bands, called subbands, which are then quantized. The splitting of the signal into subbands and the merging of the subbands into a replica of the original signal are done by decimating and interpolating filter banks, such as quadrature-mirror (QMF) or conjugate quadrature-mirror filter banks (CQF) [7,8]. Due to the decimation, the sampling frequency of a subband signal equals twice the subband's bandwidth. Therefore the total sample rate after splitting is the same as the sample rate at the input. Because the 'bandwidth' of the masking threshold as given by (1) increases with the frequency of the masking signal, the bandwidths of the subbands also have to increase (or be at least non-decreasing) with frequency.

Quantizing signals means adding quantization noise to them. If the filter banks have good separating properties, the additional noise will remain in the subband it was added to. It was assumed in Section 2 that masking occurs if the signal-to-noise ratio is above a certain threshold. This implies that the quantizers must operate at a predetermined signal-to-noise ratio. This can be achieved with uniform APCM [1] quantizers. In this type of quantizer the signal is first divided into blocks. Of these blocks the maximum absolute values, called *peak values*, are computed. By dividing the samples in the blocks by the peak values, they are scaled to a unit level. The scaled blocks are then quantized with a uniform quantizer. After dequantization the signal-to-noise ratio is proportional to the number of bits used in the quantizer. In this way the signal-to-noise ratio of a quantizer can be predetermined by allocating a certain amount of bits to it.

Figure 2 shows a diagram of a coding system with 20 subbands. As can be seen, quantized samples as well as coded peak values and side information to indicate the number of bits used for quantization are transmitted.

In this section the number of subbands and the amount of bits for the quantizers have not been determined. In Section 4 it is shown how for a given division of the signal into subbands the masking model of Section 2 can be used to determine the

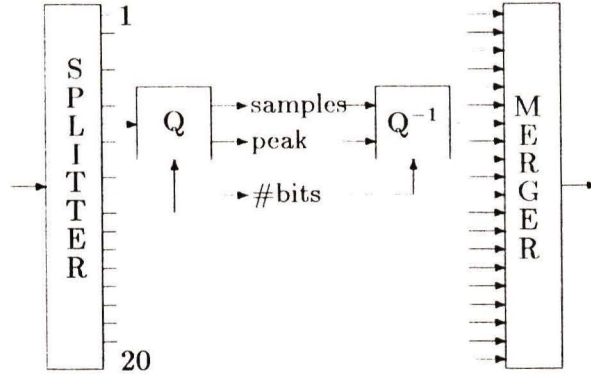


Figure 2: Subband coding system with 20 subbands

masked power in the subbands. It is also explained how the final bit rate depends on the division into subbands. In Section 5 it is shown how the number of bits allocated to each quantizer is computed.

4 Masking and subband coding

It can be seen from Figure 1 that signals with a frequency lower than the frequency of the masking signal are hardly masked. Therefore only two kinds of masking are considered: in-band masking, this is masking within a subband, and masking of signals in subbands at higher frequencies. For both cases the masked power is computed as a function of the powers of the subband signals.

Firstly it is assumed that there is only a signal in the subband with index i . This subband ranges from $f_{l,i}$ to $f_{u,i}$. The signal power is $\sigma_{s,i}^2$. The quantization noise is assumed to have a flat spectrum in the subband. The worst-case situation for in-band masking occurs when the masking signal is a pure tone with a frequency $f_{u,i}$. In this case the power of the quantization noise in subband i that is masked by a signal with power $\sigma_{s,i}^2$ in the same subband must be less than $\sigma_{s,i}^2 T(f_{u,i}, f_{l,i})$. This situation is illustrated in Figure 3.

The worst-case situation for the masking of noise in subbands at higher frequencies occurs when the masking signal in subband i is a pure tone with a frequency $f_{l,i}$. For this case the power of the quantization noise in subband j that is masked by a signal with power $\sigma_{s,i}^2$ in subband i must be less than $\sigma_{s,i}^2 T(f_{l,i}, f_{u,j})$. This situation is illustrated in Figure 4.

In this way the contribution of a subband to the masked power in all subbands can be computed. Because masking is assumed to be additive, the masked power in a subband can be obtained by adding all contributions.

The lowest achievable bit rate at a certain quality level depends on the division into subbands. Firstly, the computations of the masked powers are based on worst-case assumptions. The real masked powers can be substantially higher. If the subbands

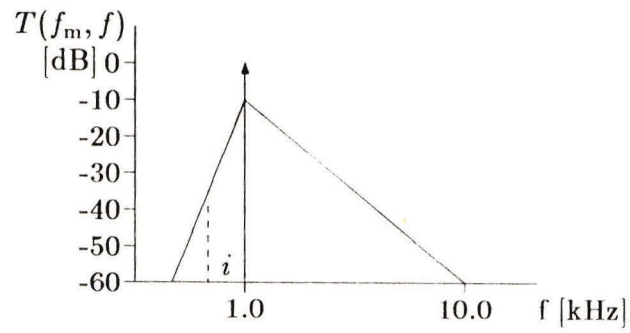


Figure 3: *In-band masking*

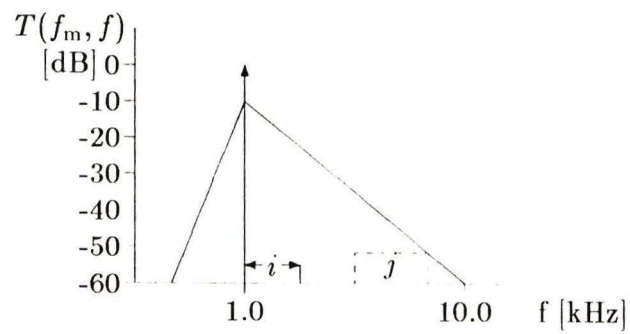


Figure 4: *Masking of subbands at higher frequencies*

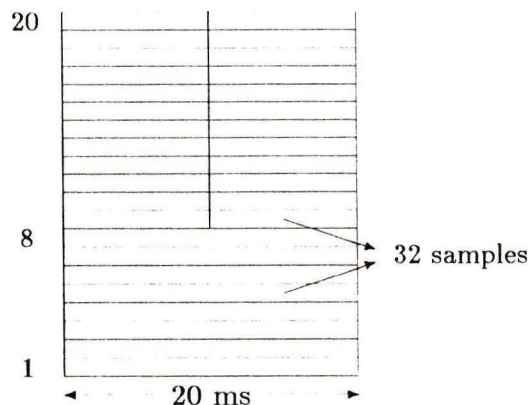


Figure 5: *Allocation window for a 20-band system*

are narrower the results of these computations will, on average, be closer to the real masked powers. Secondly, it follows from Figure 3 that the contribution of in-band masking to the masked power is higher if the subbands are narrower. The latter effect, however, is limited because in reality the top of the curve of Figures 1, 3, and 4 is flatter than is depicted [4], so that there is no use in decreasing the bandwidth of the subbands beneath a certain point. Due to the two effects mentioned here, narrower subbands lead to higher masked powers and consequently the amount of bits required for quantization can be lower.

The results of this section are only valid if the distribution of signal power over the subbands is stationary. In reality this is not true. Therefore the computations must be repeated periodically. As a consequence of this instationarity the amount of bits needed to quantize each subband will also vary in time. The allocation of bits to the quantizers on the basis of the masked powers is discussed in Section 5.

5 Bit allocation

The quantizers in the subbands are uniform APCM quantizers. For the block length M a rather arbitrary value of 32 has been chosen. Before quantization, the peak value and the power are computed for each block. The power in a block is obtained as the average of the squares of the samples of the block. The masked powers are now computed for every block, instead of for every subband as was done in the previous section. Before the masked power is computed, the blocks are arranged in an *allocation window*. An allocation window contains all subband samples during a period of time. This period is chosen in such a way that it contains one block of samples from the most decimated subband. This is in general the subband at the lowest frequency. An example of an allocation window for a 20-band system is shown in Figure 5. If the input sample frequency is 44100 Hz, subbands 1–8 have a bandwidth of 689 Hz, subbands 9–20 have a bandwidth of 1378 Hz.

It can now be computed how much each block in an allocation window contributes to its own masked power and to the masked powers in the blocks in higher subbands. A block only contributes to the masked power in blocks that lie within the time-interval of the masking block.

Before allocating bits to the blocks, blocks with powers below their masked powers can be excluded. The signals in those blocks will be masked. Only codes indicating that they are empty have to be transmitted. Assume that the amount of blocks in an allocation window is N and that the blocks are numbered from 1 to N . The masked power in the i^{th} block is denoted by $\sigma_{m,i}^2$, and the peak value by p_i . Ideally, one would choose the amount of bits b_i for this block such that the quantization noise is completely masked, which means that

$$\frac{1}{12} \left(\frac{2p_i}{2^{b_i} - 1} \right)^2 \leq \sigma_{m,i}^2.$$

This leads to a varying amount of bits per allocation window that can be higher than what is available. Therefore the bits must be allocated under the constraint

$$\sum_{i=1}^N b_i = B, \quad (2)$$

where B is the number of bits available. This number can be derived from the desired bit rate, taking into account that also a small number of bits is required to code peak values and to code the number of bits allocated to each block.

The allocation procedure is such that the total *noise-to-mask ratio*, given by

$$\sum_{i=1}^N \frac{1}{12} \left(\frac{2p_i}{2^{b_i} - 1} \right)^2 \frac{1}{\sigma_{m,i}^2}$$

is minimized under the constraint (2). A further constraint is that all b_i must be integers with $2 \leq b_i \leq 16$. The solution to this constrained integer minimization problem is given in [9].

6 Results

The ideas explained in this paper have been applied in two coding systems, a complex system splitting up the signal into 26 subbands, approximately one third of an octave wide, and a simpler 20-band system of which the bandwidths were given in Section 5. In both systems the adaptive bit-allocation method of [9] is used. Both systems have been designed for coding stereophonic 16-bit compact disc signals with a sample frequency of 44.1 kHz. Left and right channel are coded independently. With the 26-band system high-quality results can be obtained at bit rates of 220 kbit/s. With the 20-band system similar results can be obtained at bit rates of 360 kbit/s. The complexity of the systems is largely determined by the memory requirements of the filter banks. These are substantially higher for the 26-band system.

The filtering and coding delay is determined by the maximum decimation factor, the filter banks used, and the quantization block length. For the 26-band system the

decimation factor is 256 and the total delay can be as high as 800 ms. For the 20-band system it is 32 and a typical value for the total delay is 80 ms. Other types of filter banks and shorter quantization blocks may give lower values.

References

- [1] N.S. Jayant, and P. Noll. *Digital Coding of Waveforms*, Prentice-Hall, Englewood Cliffs, NJ, 1984.
- [2] E. Zwicker, and R. Feldtkeller, *Das Ohr als Nachrichtenempfänger*, S. Hirzel Verlag Stuttgart, 1967.
- [3] B. Scharf, in *Foundations of Modern Auditory Theory*, ed. J.V. Tobias, Academic Press, NY, 1970.
- [4] M.A. Krasner, 'The critical band coder', *Proc. of ICASSP 1980*, pp. 327–331.
- [5] R.N.J. Veldhuis, M Breeuwer, R.G. van der Waal, 'Subband coding of digital audio signals without loss of quality', *Proc. of ICASSP 1989*, pp. 2009–2012.
- [6] G. Theile, M. Link, and G. Stoll, 'Low bit rate coding of high-quality audio signals', Preprint 2431 (C-1), 82nd convention of the AES, London, 1987.
- [7] P.P. Vaidyanathan, 'Quadrature Mirror Filter Banks, M-Band Extensions and Perfect-Reconstruction Techniques', *IEEE-ASSP Magazine*, vol.4–3, 1987, pp. 4–20.
- [8] M.T.J. Smith, and T.P. Barnwell, 'Exact reconstruction techniques for tree-structured subband coders', *IEEE Trans. on ASSP*, vol.34–3, 1986, pp. 434–441.
- [9] Y. Shoham, and A. Gersho, 'Efficient bit allocation for an arbitrary set of quantizers', *IEEE Trans. on ASSP*, vol.36–9, 1988, pp. 1445–1453.

Image Processing with Hierarchical Polynomial Transforms

Jean-Bernard Martens

Institute for Perception Research,
P.O. Box 513, 5600 MB Eindhoven, Netherlands

1 INTRODUCTION

The development of polynomial transforms^{14,15} was motivated by a number of factors. First, we wanted an image description technique which gave a better description of the early visual system than is currently available. Second, we were not satisfied with the existing discrepancy between image processing techniques in computer vision and image coding. In computer vision, one is interested in finding important image features. This is often accomplished by fitting templates^{9,17}, such as polynomials, to the image. Hence, local *spatial* descriptions of the image are used extensively in computer vision. In image coding, the main objective is to find image representations that are efficient, i.e. reduce the amount of information, and complete, i.e. allow for a reconstruction of the original image. Currently, most image coding techniques are based on a harmonic signal analysis^{7,11,20}, i.e. use a *frequency* description of the image. This distinction between processing techniques in image coding and computer vision is remarkable and undesirable, as it obstructs the transfer of ideas between these related domains.

A polynomial transform^{14,15} performs an approximation of an image by a sum of local polynomials. A local polynomial is the product of a window function and a polynomial, and hence has some similarity to a Gabor signal⁷, which is the product of a window function with a sine or cosine. The forward polynomial transform maps the image into a set of polynomial coefficients, while the inverse transform resynthesizes the image from these coefficients.

The basic mathematics of polynomial transforms are discussed in section 2. In section 3, we elaborate on the major properties of polynomial transforms and discuss some of the implications for practical applications such as coding. An important choice for a polynomial transform is the window to be used. Section 4 argues that there is usually no window that is optimal for the entire image, but that the window should be adjusted to the image contents. One possible approach to solving this window selection problem is discussed.

2 POLYNOMIAL TRANSFORMS

In this section we describe how signals can be locally approximated by polynomials. The process involves two steps. First, the original signal is localized by multiplying it by a window function. The signal within the window is subsequently approximated by a polynomial. We concentrate first on one-dimensional(1D) signals, and subsequently extend the theory to two dimensions.

2.1 Polynomial transforms in 1 dimension

In order to localize the signal $L(x)$, we apply a *window* function $V(x)$ to it. A complete description of the signal requires that this localization process is repeated at a sufficient number of window positions. We consider the case of equidistant spacing between local window functions.

From the localized window function $V(x)$, we can construct a *weighting* function

$$W(x) = \sum_k V(x - kT) \quad (1)$$

by periodic repetition. The weighting function is itself periodic with period T . Provided $W(x)$ is nonzero for all x , we get

$$L(x) = \frac{1}{W(x)} \sum_k L(x) \cdot V(x - kT), \quad (2)$$

so that we are guaranteed that the localized signals $L(x) \cdot V(x - kT)$ for all different window positions kT contain sufficient information about the original signal.

The next step consists in approximating the signal within the window $V(x - kT)$ by a polynomial. As basis functions for the polynomial expansion, we take the polynomials $G_n(x)$, $\text{degree}[G_n(x)] = n$, that are orthonormal with respect to $V^2(x)$, i.e.

$$\int_{-\infty}^{+\infty} V^2(x) G_m(x) G_n(x) dx = \delta_{mn}. \quad (3)$$

These polynomials are uniquely determined by $V^2(x)$ ^{8,16}. Examples of orthogonal polynomials for different window functions are listed in Abramowitz & Stegun¹.

Under very general conditions⁴ for the original signal $L(x)$, we get that

$$V(x - kT) \left[L(x) - \sum_{n=0}^{\infty} L_n(kT) \cdot G_n(x - kT) \right] = 0, \quad (4)$$

with

$$L_n(kT) = \int_{-\infty}^{+\infty} L(x) \cdot G_n(x - kT) V^2(x - kT) dx. \quad (5)$$

Hence, the polynomial coefficients $L_n(kT)$ can be derived from the original signal $L(x)$ by convolving with the *filter* functions

$$D_n(x) = G_n(-x) V^2(-x), \quad (6)$$

followed by sampling at multiples of T . This mapping from the original signal $L(x)$ to the coefficients $L_n(kT)$ is called a forward polynomial transform.

If $L(x)$ is analytic and finite for all x , then the convergence of the series expansion in equation (4) can be guaranteed for most window functions. Hence, for a broad class of signals and window functions, the polynomial approximation error can be made arbitrarily small by taking the maximum degree of the expansion sufficiently high. This in turn implies that the description of the localized signal $L(x) \cdot V(x - kT)$ can, up to an arbitrary small approximation error, be reduced to specifying a finite set of polynomial coefficients $L_n(kT)$. The latter signal description is much easier to handle than the original signal itself. For instance, the signal energy within the window can be expressed in terms of the coefficients of the expansion, i.e.

$$\int_{-\infty}^{+\infty} L^2(x) V^2(x - kT) dx = \sum_{n=0}^{\infty} L_n^2(kT), \quad (7)$$

hence reducing the continuous integral to a discrete sum. This is the generalization of Parseval's theorem to orthonormal polynomials.

Combining equations (2) and (4), we get the following expansion for the complete signal

$$L(x) = \sum_{n=0}^{\infty} \sum_k L_n(kT) \cdot P_n(x - kT). \quad (8)$$

This signal reconstruction, which is called a inverse polynomial transform, consists of interpolating the coefficients with the *pattern* functions

$$P_n(x) = G_n(x)V(x)/W(x), \quad (9)$$

and adding terms.

The forward and inverse polynomial transforms are illustrated in figure 1. Note that a polynomial transform is completely specified by the window function $V(x)$ and the sampling distance T . Section 4 will discuss how these parameters can be selected.

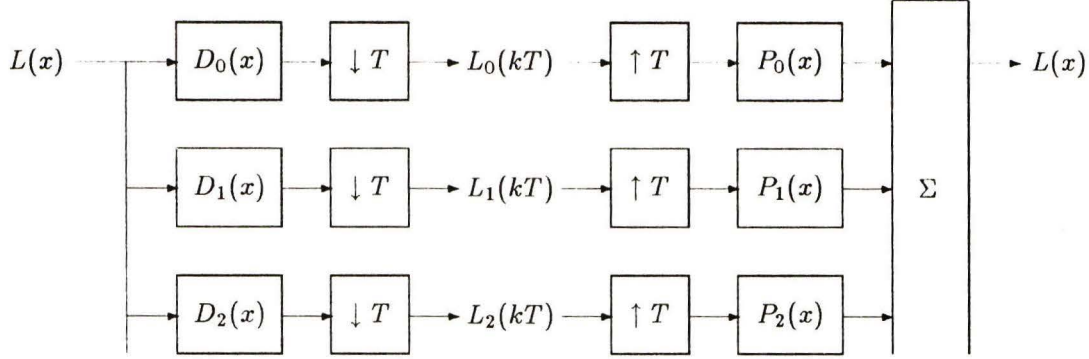


Figure 1: Polynomial transform.

If the window function is Gaussian, then the polynomial transform is called a Hermite transform^{14,15}. The Hermite transform has a number of interesting special properties. Most noticeably, the forward polynomial transform involves the use of derivatives of Gaussians. These filters are used extensively in computer vision^{2,3,18} because of their interesting operational characteristics. Moreover, it has been demonstrated that these Gaussian derivatives also provide a good description of cortical receptive fields²¹, and are hence relevant to visual perception.

2.2 Polynomial transforms in 2 dimensions

The polynomial transform technique can be easily generalized to two dimensions. Given a local window function $V(x, y)$, the orthonormal polynomials $G_{m,n-m}(x, y)$, where m and $n - m$ are the degrees with respect to x and y respectively, are uniquely determined¹⁰ by

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} V^2(x, y) G_{m,n-m}(x, y) G_{j,i-j}(x, y) dx dy = \delta_{ni} \delta_{mj}, \quad (10)$$

for $n, i = 0, 1, \dots, \infty$, $m = 0, \dots, n$ and $j = 0, \dots, i$.

The decomposition of two-dimensional(2D) signals into localized polynomials becomes

$$L(x, y) = \sum_{n=0}^{\infty} \sum_{m=0}^n \sum_{(p,q)} L_{m,n-m}(p, q) \cdot P_{m,n-m}(x - p, y - q), \quad (11)$$

where the coordinates (p, q) belong to a 2D sampling lattice. The polynomial coefficients $L_{m,n-m}(p, q)$ are derived by convolving the image with the filter functions

$$D_{m,n-m}(x, y) = G_{m,n-m}(-x, -y)V^2(-x, -y), \quad (12)$$

followed by sampling at the coordinates (p, q) of the lattice. The pattern functions are defined by

$$P_{m,n-m}(x, y) = G_{m,n-m}(x, y)V(x, y)/W(x, y), \quad (13)$$

where

$$W(x, y) = \sum_{(p,q)} V(x - p, y - q) \quad (14)$$

is the 2D weighting function. The necessary condition for applying the polynomial transform is that the weighting function $W(x, y)$ is different from zero for all coordinates (x, y) .

An interesting special case arises when the window function is separable, i.e. $V(x, y) = V_x(x) \cdot V_y(y)$, and the sampling lattice is rectangular. The filter and pattern functions are then also separable, and can hence be implemented very efficiently. An important example of a separable window is the 2D Gaussian window.

3 PROPERTIES OF POLYNOMIAL TRANSFORMS

It is a well-known analogy that a 2D signal $L(x, y)$ can be regarded as a 2D surface in a three-dimensional space by interpreting the signal strength as the height above the (x, y) -plane. This input description does however not make explicit the relationships that may exist between different points on the surface. Differential geometry¹⁹ describes how such a pointwise characterization of a surface can be converted into a parametrized description, so that important characteristics such as surface orientation and tangent plane, curvature, surface area, ... can be derived. Differential geometry is based on the ability to investigate how surface descriptions depend on the orientation and position of the reference frame. We argue that polynomial descriptions are ideally suited for this purpose.

Given a polynomial description of degree n in a reference frame (x, y) , then translating and rotating this reference frame to

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \quad (15)$$

will result in a polynomial description of the same degree n . Moreover, the polynomial coefficients in the new reference frame depend linearly on the coefficients in the old reference frame. For example, a simple polynomial description of degree 1

$$L(x, y) = L_{00} + L_{10} \cdot x + L_{01} \cdot y \quad (16)$$

in the original (x, y) -frame is converted into

$$\begin{aligned} L(x, y) &= [L_{00} - \Delta x(L_{10} \cos \theta + L_{01} \sin \theta) - \Delta y(-L_{10} \sin \theta + L_{01} \cos \theta)] \\ &+ (L_{10} \cos \theta + L_{01} \sin \theta) \cdot x' + (-L_{10} \sin \theta + L_{01} \cos \theta) \cdot y' \end{aligned} \quad (17)$$

for the transformed (x', y') -frame. This property of polynomials of being closed under translation and rotation is unique and has important consequences for image coding (and image processing).

The first aim in image coding is to compact the signal energy in as few components as possible. For instance, in a uniform region of the image this is realized (by most image description techniques) in a simple way because the mean value is a sufficient description. The higher order (polynomial) coefficients are then all equal to zero. Another class of image features that are very important, and that occur frequently, are local 1D structures, such as edges and lines. By locally reorienting the reference frame along the direction of the 1D structure, we can make a large number of the polynomial coefficients equal to zero¹⁵.

The second aim in image coding is to reduce the number of bits needed for coding the (non-zero) coefficients by making predictions for these coefficients, based on previously transmitted information. In

this case, a prediction for the polynomial description, centered at a given point in the image, has to be derived from polynomial descriptions in nearby points. This implies translating the reference frame of the latter descriptions to the point of interest. Again, this can be done fairly easy with polynomial descriptions by simple linear transformations.

4 HIERARCHICAL IMAGE DESCRIPTIONS

It is unrealistic to expect that a single image representation technique can be optimal for all images of interest. In this section we therefore address the problem of how several techniques, such as for instance polynomial transforms for different windows and sample structures, could be integrated.

A useful interpretation of many image processing algorithms is that they convert an image into an image description. In practice, this is most often done by determining the optimum approximation of the input image by a weighted sum of a priori selected patterns. The weights of the contributing patterns then constitute the description of the input image. With these coefficients an approximation of the original image can be synthesized. Application of such a processing algorithm in image coding is straightforward and illustrated in figure 2.

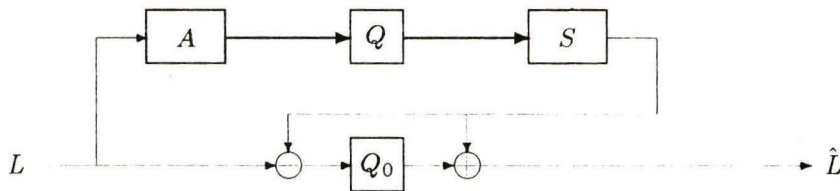


Figure 2: The analysis A converts the input signal into a set of coefficients. These coefficients are coded in Q . An approximation of the original signal is synthesized in S , using the coded coefficients. The remaining differences between the input signal and the approximated signal are coded in Q_0 .

In order for the image description to be efficient, it is necessary that the a priori selected patterns are well-adapted to the input image, i.e. that the original image can be (locally) approximated with a limited number of patterns. However, because of the large variety of image features that can arise in natural images, it is very unlikely that one set of patterns can offer an efficient description for all these features. For instance, distinct spatial structures such as homogeneous regions, edges, lines, textural regions, ... are likely to require distinct descriptions. The situation is further complicated by the fact that these structures can have many different orientations and sizes (or spatial scales), and that they can move in different directions within a range of velocities (or temporal scales).

One possible approach is to use several sets of patterns, inspired by the different features that can arise in an image. For example, in many existing coding schemes, we distinguish sets that differ in spatial scale. For each set of patterns an image analysis and resynthesis can be performed. The image is subsequently segmented in regions by selecting the most efficient description at each image point. This approach is for instance adopted in segmentation-based coding¹¹. Additional recursive processing of the raw segmentation data is however usually required to limit the number of segmentation regions.

This general scheme can be simplified in the case that the different sets of patterns can be ordered in a hierarchical way. For instance, in the case of two sets of patterns with decreasing complexity and descriptive power, we start by examining the image description by the simpler second set. This set clearly offers the most efficient description, when applicable. The approximated image, after synthesis with the second set of patterns, is subsequently compared with the original image by decomposing both on the more

complex first set of patterns. Only the differences in the two descriptions have to be coded. Application of this approach in coding is illustrated in figure 3.

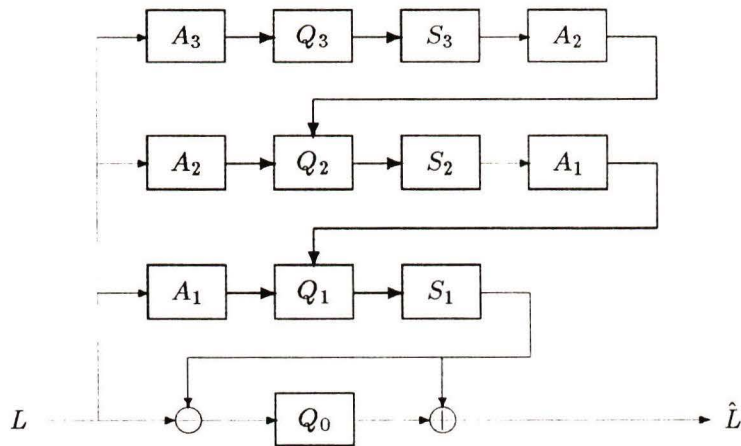


Figure 3: Hierarchical coder: The analysis A_3 maps the input signal into a coefficient description for the simplest set of patterns. These coefficients are coded in Q_3 . An approximation of the original signal is synthesized in S_3 , using the coded coefficients. The analysis A_2 maps both the original and approximated signal into a coefficient description for the next set of patterns. The coefficients of the original image are coded in Q_2 , conditional on the coefficients of the approximated image. This usually results in substantial savings over a direct coding of these coefficients. The above process can subsequently be repeated at a number of levels. The description for the most complex set of patterns is compared to the original signal.

The hierarchical scheme of figure 2 can for instance be applied in the case of multiscale analysis, where the different sets of patterns differ only in (spatial and/or temporal) scale and sample spacing. This includes wavelet descriptions^{6,12}, with pyramid coding^{5,13} and sub-band coding²⁰ as special cases, but also polynomial transforms for windows of different sizes¹⁵. The orthogonality conditions that are used in wavelet descriptions result in a specific kind of prediction between the different levels of the hierarchical coder. More precisely, a perfect prediction is made for some coefficients in the description, while no prediction is made for the remaining coefficients. It is however not obvious a priori that this approach is better than the general case where an imperfect prediction is made for all coefficients. Although the number of coefficients to be coded is larger in the general case, the total information content may be smaller because the conditional coding of the coefficients is more efficient than the unconditional coding. The application of hierarchical coding to polynomial transforms is an illustration of the general case.

5 REFERENCES

1. M. Abramowitz & I. Stegun, *Handbook of Mathematical Functions*, Dover Publications, New York, 1965.
2. H. Asada & M. Brady, The Curvature Primal Sketch, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8, pp. 2-14, January 1986.
3. J. Bevington & R. Mersereau, Differential Operator Based Edge and Line Detection, *ICASSP Proceedings*, pp. 249-252, 1987.
4. R. Boas & R. Buck, *Polynomial Expansions of Analytical Functions*, Springer-Verlag, Berlin, 1958.
5. P. Burt & E. Adelson, The Laplacian Pyramid as a Compact Image Code, *IEEE transactions on Communications*, 31, pp. 532-540, April 1983.

6. I. Daubechies, Orthonormal Bases of Compactly Supported Wavelets, *Communications on Pure and Applied Mathematics*, *XLI*, pp. 909-996, 1988.
7. J. Daugman, Entropy Reduction and Decorrelation in Visual Coding by Oriented Neural Receptive Fields, *IEEE Transactions on Biomedical Engineering*, *36*, pp. 107-114, January 1989.
8. G. Freud, *Orthogonal Polynomials*, Pergamon Press, Oxford, 1971.
9. R.M. Haralick, Ridges and Valleys on Digital Images, *Computer Vision, Graphics and Image Processing*, *22*, pp. 28-38, 1983.
10. D. Jackson, Formal Properties of Orthogonal Polynomials in Two Variables, *Duke Mathematical Journal*, *2*, pp. 423-434, 1936.
11. M. Kunt, M. Benard & R. Leonardi, Recent Results in High-Compression Image Coding, *IEEE Transactions on Circuits and Systems*, *34*, pp. 1306-1336, November 1987.
12. S. Malath, A Theory for Multiresolution Signal Decomposition: The Wavelet Representation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*, pp. 674-693, July 1989.
13. J.B. Martens & G.M.M. Majoor, The perceptual relevance of scale-space image coding, *Signal Processing*, *17*, pp. 353-364, 1989.
14. J.B. Martens, The Hermite Transform - Theory, *IEEE Transactions on Acoustics, Speech and Signal Processing*, *38*, pp. 1595-1606, September 1990.
15. J.B. Martens, The Hermite Transform - Applications, *IEEE Transactions on Acoustics, Speech and Signal Processing*, *38*, pp. 1595-1606, September 1990.
16. G. Szego, *Orthogonal Polynomials*, American Mathematical Society, Colloquium Publications, 1959.
17. C.H. Teh & R. Chin, On Image Analysis by the Method of Moments, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*, pp. 496-512, July 1988.
18. V. Torre & T. Poggio, On Edge Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *8*, pp. 147-163, March 1986.
19. J.A. Thorpe, *Elementary Topics in Differential Geometry*, Springer-Verlag, New-York, 1979.
20. J. Woods, Sub-band Coding of Images, *IEEE Transactions on Acoustics, Speech and Signal Processing*, *34*, pp. 1278-1288, October 1986.
21. R. Young, The Gaussian Derivative Theory of Spatial Vision: Analysis of Cortical Cell Receptive Field Line-Weighting Profiles, *General Motors Research Laboratories report 4920*.

SVD AND SIGNAL PROCESSING

Algorithms, Applications and Architectures

Edited by

Ed. F. DEPRETTERE

Department of Electrical Engineering
Delft University of Technology
Delft, The Netherlands



1988

NORTH-HOLLAND
AMSTERDAM • NEW YORK • OXFORD • TOKYO

Bijzondere onderwerpen

SOME FLUID DYNAMIC ASPECTS OF SPEECH

A.Hirschberg, Vakgroep Transportfysica, W&S 0-54, TUE, Postbus:513, 5600 MB Eindhoven.

Paper presented at the fourth Colloquium Signaalanalyse en Spraak, COLSAS 22-23 oct. 1990, held at the Instituut voor Perceptie Onderzoek, Eindhoven.

1. Introduction

Papers by Teager and Teager [1,2] and Kaiser [3] on nonlinear production mechanisms and flow in the vocal tract provide us with questions without answers. I cannot answer most of these questions because I know too little about speech. I will restrict myself to some fluid dynamic aspects of the problem. In particular I would like to give some general considerations on the character of flow induced sound sources. I will try to give a feeling for the use of these general concepts in speech by considering some simple examples: glottis oscillation, human whistling and sound production by turbulence. Finally I will give some information about the state of the art in fluid dynamics.

I will start by proposing a definition of "sound" and by discussing the relationship between flow and acoustic field (section 2).

We discuss three basic types of sound sources:

- the monopole (volume injection) [+]
- the dipole (force) [+ -]
- the quadrupole [$\begin{matrix} + & - \\ - & + \end{matrix}$] or [+ - - +]

The periodic volume flow through oscillating vocal cords acts as a monopole on the supra-glottal (downstream) part of the vocal tract. Vortex shedding induces an aero-acoustic dipole.

Turbulence in free space induces a quadrupole.

In section 3, I will explain why the type and position of the source is crucial for sound production.

In the literature one often assumes that vocal cords oscillation is driven by a Bernoulli force due to the interaction of the cords with the local flow. In section 4 we discuss the classical quasi-stationary flow model of the Bernoulli force.

As an example of vortex driven sound we consider in section 5 a model for human whistling proposed by Wilson e.a.[4].

Finally sound production by turbulence is discussed in section 6.

2 Flow and acoustic field

The flow generated by speech is locally incompressible because the pressure differences driving the flow are very small compared with the atmospheric pressure (1 bar). The sound field consists of deviations from the incompressible or constant density approximation which can be observed on the length scale of the acoustic wave length c/f . Where c is the speed of sound and f a characteristic frequency. These deviations are characterised by density fluctuations ρ' and pressure fluctuations $p' = c^2 \rho'$ which propagate as waves. In free space the acoustic (particle) velocities u_a associated with the wave propagation are much smaller than the incompressible flow velocities v_i ($u_a \approx p'/\rho c$). Therefore one may in general neglect the feedback from the acoustic field to the incompressible flow in free space. Acoustic waves are per definition assumed to be linear [5,6,7].

The vocal tract is a resonator in which acoustic energy can accumulate. In a resonator the acoustic velocities may become larger than the incompressible flow velocities exciting the sound field. In a clarinet the pressure amplitude p' of the acoustic field at the reed can be larger than the mean pressure difference over the reed which is driving the system. In such a case during part of the oscillation period there is a flow through the reed directed from the pipe (resonator) towards the mouth of the player [8]! Clearly there is under such circumstances a strong (non-linear) interaction between the flow through the reed and the acoustic field.

Another type of (non-linear) interaction between incompressible flow and acoustic field is shown in Fig. 1. The oscillating velocity of the acoustic field in combination with the mean flow velocity results into periodic vortex shedding at the end of a pipe driven by a clarinet mouth piece. The vortex rings are very similar to the smoke rings generated by a smoker when impulsively blowing. The shedding of vortices is controlled by centrifugal forces and viscous forces at a curved surface [9]. At a sharp edge (teeth), when the radius of curvature is zero, there is always flow separation. Flow separation results into the formation of a layer separating a low speed region from a high speed region : the so called shear layer. A stream tube bounded by shear layers is called a free jet. Such a free jet is formed downstream of the glottis and was observed by Teager [1].

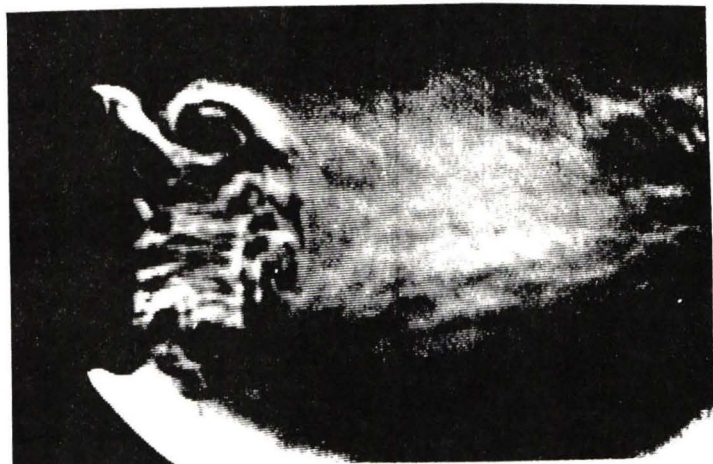
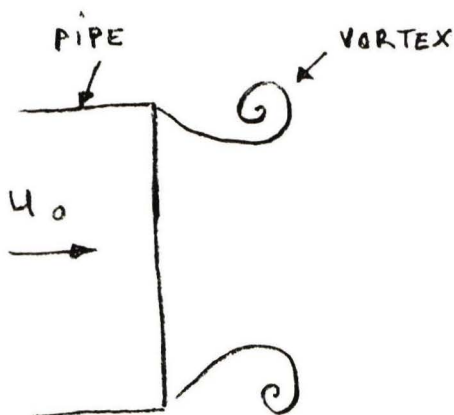


Fig.1 Vortex shedding induced at a pipe termination acoustic resonance (pipe driven by clarinet mouth piece, flow visualization by CO₂ injection).

A shear layer can be considered as a row of vortices. Simple kinetic considerations demonstrate that a shear layer is unstable for low frequency perturbations. Perturbation of the rate of vorticity shedding at the separation point, induces a rolling up of the shear layer into a coherent structure called ring vortex. Acoustic perturbation induces periodic vortex shedding.

The basis of the modern theory of aero-acoustics [5,6,7] is the assumption that once the strength and the path of the vortices as been calculated by means of an incompressible flow approximation, one can calculate the generated acoustic field using the "vortex sound" theory of Howe [5]. When the vortex shedding depends on the acoustic field this procedure should be carried out iteratively.

In fluid mechanics we make a strong distinction between vortex shedding and turbulence [9]. Vortex shedding is the process described above. 2-D vortices are very persistent in absence of main flow. (Note: ring vortices are 2-D structures in cylindrical coordinates). Turbulence is a chaotic behaviour of the flow triggered by the non linear convective forces in the flow. This occurs at high velocities when viscous forces are not sufficient to stabilize the flow. In turbulence energy extracted from the mean flow at large length scales is transferred to smaller length scales by the process of "vortex stretching". By a so called "cascade " process of successive vortex stretching the energy is transferred to decreasing length scales. When the energy has reached a critical length scale (Kolmogorov length scale) it is dissipated by viscous forces (the fluid is heated up). Due to this very effective dissipation process, in absence of a non-uniform main flow from which energy can be extracted turbulence dies fast.

At typical conditions encountered in speech the flow in the oscillating glottis is not turbulent but the free jet in the vocal tract is expected to be turbulent. The interaction between turbulence and the acoustic field in a pipe is weak [6].

When we measure a pressure fluctuation in the vocal tract with a microphone we have two contributions one from the acoustic field and one from the incompressible flow. The contribution from the incompressible flow is the "pseudo-sound" which consists out of pressure disturbances which do not propagate with the speed of sound but with the flow velocity (think of the low pressure in a tornado). Because turbulence is a chaotic flow with a broad band pressure fluctuation spectra, it can be distinguished from the acoustic field by spectral analysis as suggested by Cranen and Boves [10]. This is not true for the pseudo-sound of periodic vortex shedding. However vortex "rings" generated at the glottis are not expected to live long. Due to turbulence they soon should diffuse and annihilate. The influence of vortex shedding on pressure measurements depends strongly on the distance from the glottis at which the microphone is placed.

3 Excitation of a resonator by basic types of sound sources

As stated in the introduction aero-acoustic sound sources in the vocal tract can have the character of a monopole (oscillating flow through the glottis), dipole (vortex shedding) and quadrupole (turbulence).

The supra-glottal part of the vocal tract is a resonator which we represent for simplicity as a pipe segment of length L , closed at one end (glottis) and open at the other end (mouth). We will now show that the capability of a sound source to excite such a resonator depends strongly on the frequency of oscillation of the source and its position in the resonator.

At frequencies below the cut off frequency of the pipe, the acoustic field can be approximated by two plane waves with equal amplitudes and opposite propagation directions. Interference of these waves results into standing waves. For given acoustic source the acoustic field can be considered as build up out of a series of standing waves with wave length λ_n :

$$\lambda_n = 4 L / (1 + 8n) \quad ; \quad n=0,1,2,\dots \quad (1)$$

Each of these standing waves is a so called mode which behaves as an independant harmonic oscillator (acoustic mass/spring system) with a resonance frequency $f_n = c / \lambda_n$.

(Note: in speech resonances of the vocal tract are responsible for the formation of formants.). The closed end corresponds to a node of the acoustic velocity distribution and a maximum of the pressure amplitude in the standing waves. At the open end the acoustic pressure is almost zero $p' \simeq 0$ (pressure node).

Let us place a monopole (pulsating sphere) in the resonator. The volume flow injected is $Q = dV/dt$, where V is the volume of the sphere. The source performs acoustic work given by:

$$W = \int p' dV = \int_0^t p' (dV/dt) dt' = \int_0^t p' Q dt' \quad (2)$$

We see from this formula that placing a monopole at the open end ($p'=0$) will not excite the resonator. Please note that direct injection of Q in free space without vocal tract would be a very ineffective way of producing sound because the sound source cannot perform much work ($p' \simeq 0$). The vocal tract is not only a filter it is also an impedance matching between the source and free space. The injection of Q at the closed end can excite a mode of the resonator if we adjust the oscillation frequency to that of the mode. If we neglect losses and non-linear effects we see from equation (2) that p' will increase indefinitely with increasing time (resonance). The higher p' the more work the source can perform.

A dipole corresponds with two monopoles of equal strength Q but with opposite phases, placed at a small distance δ from each other along the pipe. δ should be small compared to the wave length c/f . We will show that this corresponds to a force excitation. Assume for simplicity that the flow between the two monopoles is uniform. The velocity of the fluid in this region is given by Q/S , where S is the cross section area of the pipe. The momentum of the fluid in the region is $[\rho_0 (Q/S) S \delta]$, where ρ_0 is the fluid mean density. From Newton's law we know that the rate of change in momentum corresponds to a force F :

$$F = d[\rho_0 Q \delta] / dt \quad (3)$$

In words: the air between the monopoles is, like a cat in a bag, jumping up and down. This results into a force F on the "bag". In the case of the pipe the force F is provided by the surrounding air in the form of a pressure jump $\Delta p = F/S$ over the region where the dipole is placed.

Note that when a dipole is placed in the middle of the pipe perpendicular to the pipe axis it will not radiate for frequencies below the cut off frequency for the first transverse mode of the pipe. A dipole placed close to the wall and directed perpendicular to the wall will have a very weak radiation field even at high frequencies. In the present section we only consider low frequencies and dipoles directed along the pipe axis.

The acoustic work W performed by the force F is given by:

$$W = \int F dx = \int_0^t F (dx/dt) dt' = \int_0^t F u_a' dt' \quad (4)$$

where u_a' is the acoustic velocity. We see from formula (4) that a dipole sound source like vortex shedding at the closed pipe end ($u_a'=0$) will not excite the modes of the pipe. A dipole placed at the open end where the amplitude of u_a' is maximum will strongly interact with the acoustic field in the pipe. Hence the vortex shedding illustrated in Fig. 1 is expected to be an effective sound source.

A quadrupole is obtained by placing two opposite dipoles at a small distance from each other. ($\delta \ll c/f$). In a pipe with uniform cross section a quadrupole is an very ineffective sound source, what ever its position along the pipe.

We have considered here only the supra-glottal part of the vocal tract as a resonator. This can be a useful approximation, however there is a priori no reason to exclude the coupling with the sub-glottal part and the lungs. In particular low frequency oscillations might be due to resonance of the entire system. We will consider this further in the next section.

4 Vocal cords oscillation

In the literature the vocal cords oscillation is assumed to be mainly controlled by the interaction of the flow with the vocal cords. Reviews of such models are given by Cranen [11] and Titze [12]. The model is based on the assumption that the flow in the glottis is not only locally incompressible but also frictionless and quasi-stationary. Under such circumstances the mechanical energy of a fluid particle is conserved as it is convected along a streamline. This conservation law is the Bernoulli equation relating the velocity v to the pressure p :

$$\frac{1}{2} \rho_0 v^2 + p = \text{constant} \quad (5)$$

where the integration constant is determined by the boundary conditions.

An essential assumption in most models is that the flow separation responsible for the formation of a free jet does not occur at the narrowest cross section of the glottis but a little bit further downstream as shown in figure 2.

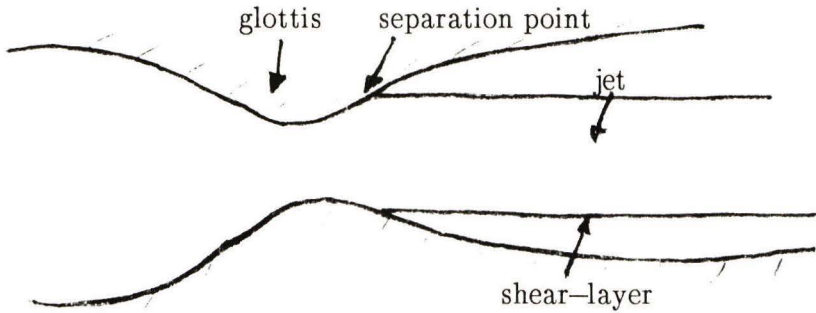


Figure 2. Formation of a free jet in the glottis

Note that as the free streamlines (shear layers) at the boundaries of the jet are like flexible walls which cannot sustain any pressure difference the pressure in a stationary free jet is determined by its surrounding. In the present case it is equal to the pressure in the supra-glottal region. Therefore we can use Bernoulli's equation to relate the supra-glottal pressure to the pressure in the glottis.

Using the conservation law of mass we see that for a stationary flow the velocity at the narrowest cross section should be higher than in the jet. Hence applying Bernoulli's equation we conclude that the local pressure at the narrowest cross section must be lower than the pressures just upstream and downstream of the glottis. This low pressure will "suck" the vocal cords towards each other. This sucking force is the so called Bernoulli force F_B .

As noted by Cranen [11] and Titze[12], assuming that the Bernoulli force depends only on the aperture h of the glottis, all the work performed by this force upon closing the glottis will be released when the glottis opens. Integrated over a period $T = 1/f$ of oscillation we have :

$$W = \int_0^T F_B (dh/dt) dt = 0 \quad (6)$$

Hysteresis is needed to explain a transfer of energy from the flow to the vocal cords oscillation. This can be achieved in various ways:

- change in vocal cords geometry (two mass model or wave motion along the vocal cords surface [11,12]).
- Time dependence of the separation point ([13],[14]).
- Contribution of inertial forces in the flow ([13],[15],[16]).

The first effect is well known in the literature.

Flow visualization experiments by Gupta [13] did not show large changes in the position of the separation point. Gupta therefore assumes that this effect is negligible. One should however consider this conclusion with care. When the vocal cords are almost closed a shift in separation point must occur as a result of increasing importance of viscous forces. This effect which was observed in our experiments is also illustrated by the stationary flow measurements of Scherer and Titze [17].

The effect of inertia in the glottal flow can be significant [13].

One of the statements of Teager and Kaiser is that acoustical feedback and loading may also be important. By acoustical feedback we mean an influence of the acoustic field on the glottis oscillation. By loading we mean that acoustic pressure fluctuations at the glottis cannot be neglected compared to the transglottal pressure drop. This last effect simply implies that the glottis is not a ideal volume source (Q depends on p'). Titze [12] gives a description of the acoustical loading by the supra-glottal part of the vocal tract. Teager [2] illustrates the importance of acoustic feedback on speech by describing the effect of replacing air by Helium in our lungs. Teager reports an increase of the formant frequencies by a factor 1.6 instead of the factor 3 expected in a source filter model. Contamination of Helium with 30% air would explain such an effect. Hence the "evidence" of Teager's conclusion is doubtful. However Gupta e.a. [13] show that the sub-glottal part of the system (trachea) and the lungs may strongly influence the vocal cords oscillations. In their model self sustained oscillations are achieved without Bernoulli force as a result of acoustical feedback. The acoustic model used by Gupta [13] for the lungs seems unrealistic and should be reconsidered. The problem is therefore still open.

Note that the existence of an interaction between the vocal cords oscillation and the acoustic field is not necessarily in contradiction with a source/filter model. Let's assume that the low frequency oscillations of the entire acoustic system (lungs, trachea and vocal tract) sustains the oscillation of the glottis. The low frequency determines the fundamental frequency of speech. Changes of the geometry of the supra-glottal part responsible for speech might not modify strongly the low frequency resonance of the entire system.

5 Human whistling

As stated above periodic vortex shedding induced by strong acoustic oscillations is a dipole type sound source. The acoustic dipole corresponding to a vortex ring is directed perpendicular to the plane of the ring [7].

In general, vortex shedding extracts energy from the acoustic field [6]. The reduction with increasing mean flow velocity of the energy reflection coefficient (Fig.3) at an open end of a pipe with sharp edges can be interpreted in terms of vortex sound [18]. When however the pipe is terminated by a horn there can be a net sound production by vortex shedding if the travel time of the vortex in the horn matches the oscillation frequency (Fig. 3). This travelling time is determined by the mean flow velocity.

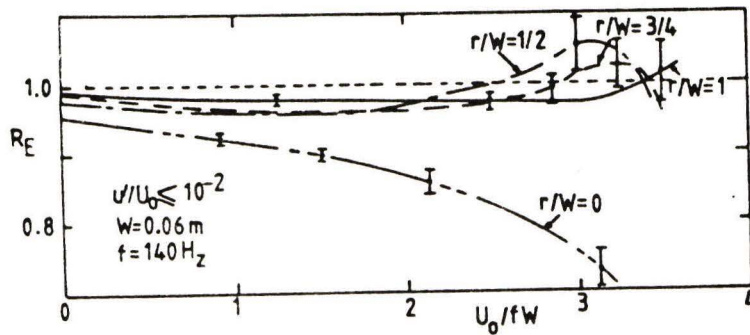


Fig. 3 Influence of the mean flow velocity U_0 on the energy reflection coefficient R_E at a pipe termination with sharp edges ($r/W=0$) and with horns (horn radius of curvature r , pipe width W). [18]

When we whistle, we form an acoustic mass spring system or Helmholtz resonator with our mouth [4]. We then adjust the blowing velocity to the resonator frequency. This results in self-sustained oscillation due to periodic vortex shedding at our lips (horn-shaped pipe termination). Of course, this effect cannot be described by a source/filter model as used in speech. Such effects are also expected in the typical speech of a person missing a tooth.

Note that the model described above is very similar to the model proposed by Shadel [19]. In the model of Shadel, however, the lips have sharp edges and vortex shedding occurs at the entrance of the flow channel formed by the lips.

6 Sound production by turbulence

Turbulence in free space is a quadrupole type of aero-acoustic sound source [7]. It is a very weak sound source at speeds v low compared with the speed of sound. The power produced decreases with $(v/c)^8$. When an object small compared with the wavelength is placed in the neighbourhood of the flow, the power decreases only with $(v/c)^6$ because the object induces a dipole contribution.

The "transformation" of a quadrupole into a dipole by a cylinder of radius R is easily understood by using the method of images as shown in Fig. 4. The figure is obtained by using the basic law: that the image of a source Q placed at a normalized distance r/R from the axis of the cylinder is a source Q at the inverse point R/r and a sink $-Q$ on the axis of the cylinder [20]. As shown in Fig. 4 the right hand dipole from the original quadrupole forms a quadrupole with its image because it is close to the cylinder. The image of the left hand dipole is a negligible small dipole near the axis of the cylinder. Hence the left hand dipole acts almost as a free dipole.

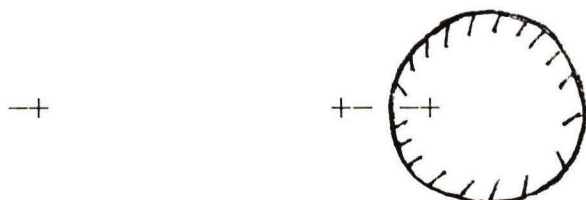


Fig. 4 Longitudinal quadrupole near a cylinder.

This partially explains why wind blowing in a forest is much louder than in free space. At 10 m/s the difference in sound power is about a factor $(v/c)^2 \approx 10^3$. (Note: also periodic vortex shedding at branches contributes quite significantly to sound production). For sharp edges (teeth) on an object (man) large compared with the wave length this effect is even stronger: a $(v/c)^5$ dependence is expected [7].

A free space approximation as used above can only be used in the vocal tract for frequencies high compared to the cut off frequency for the first transversal mode.

There is little information about sound production by turbulence in a tube at low frequencies. In a uniform tube for frequencies below the cut off frequency of the first transversal mode we expect a $(v/c)^6$ dependence of the sound power produced by turbulence. In a series of experiments with an organ pipe we found that when the jet was directed below the labium so that no self-sustained oscillation occurred, the sound power produced by the turbulence increased with $(v/c)^4$. This is the behaviour expected for an aero-acoustical dipole sound source in a tube at low frequencies [7]. The dipole character is due to the presence of a sharp edge (labium) in the neighbourhood of the turbulent jet. Unlike in free space we found a non-uniform spectral distribution. The turbulent noise is modulated by the resonances of the tube (Fig.5).

The dramatic increase with speed and the very low efficiency of turbulence as a sound source is clearly illustrated by fricative noise in the vocal tract. We must blow much harder with open glottis to reach a much lower sound level than when the sound is produced by vocal tract oscillation. Also the effect of sharp edges (teeth) becomes clear when we compare blowing with open mouth and with almost closed teeth. The increase in sound production is a combination of improvement of sound production by the ("already existing") turbulence in the main flow and additional vortex shedding at sharp edges.

The importance of obstacles in the flow and the dipole character of turbulence induced sound sources in the vocal tract is discussed in detail by Shadle [19,21].

Sound produced by vortex shedding can easily be distinguished from turbulence noise. Vortex shedding produces sharp peaks in the sound spectra. The quality factor of these peaks is much larger than typical quality factors of vocal tract resonance. When the frequency is determined by the flow (aeolian tone of a cylinder [7,9]) the frequency is proportional with the flow velocity. When acoustical feedback induces whistling by lock-in of the vortex shedding the frequency is close to a resonance frequency of the mouth or a part of the mouth. It increases then only very slowly with increasing flow velocity.

The discussion above we have only considered low frequencies. However in fricative sounds the high frequencies contribute substantially to the signal [19,21]. Above the cut off frequency of the first transversal mode the dipoles directed perpendicular to the pipe axis will also radiate. This effect explains the sudden increase at the cut off frequency of sound produced by a turbulent jet blowing below the labium of an organ pipe (Fig.5).

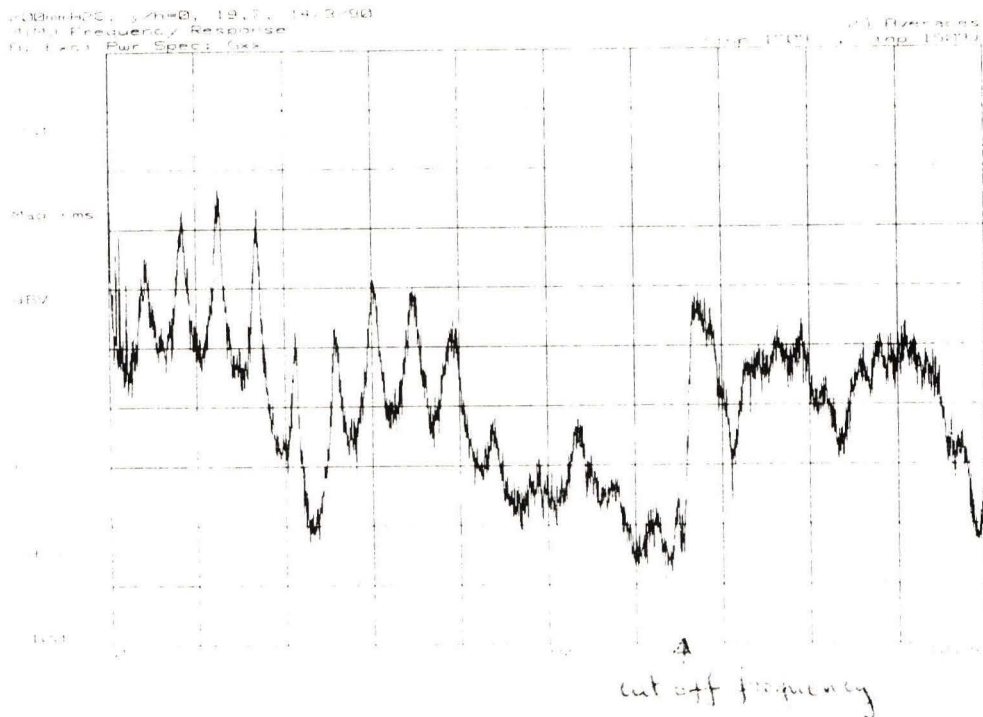


Fig. 5 Internal pressure spectrum of an organ pipe excited by a turbulent jet (pipe length 28.2 cm, width 2.0 cm).

Because both the cut off frequency and the position of the flow obstacles depend on the shape of the vocal tract a simple low frequency (one dimensional) source/filter model will not be accurate. Both the position and the strength of the sources will have to be modified when the filter is modified in order to obtain an accurate description of fricative sounds.

7 Conclusion

Some of the flow effects described by Teager [1,2] and Kaiser[3] may be significant for speech.

Acoustical "feedback" or "loading" can affect vocal cords oscillations and even sustain the oscillations in absence of Bernoulli force. This statement is however not necessarily in contradiction with the source/filter model.

Vortex shedding is the source of sound responsible for acoustic energy production in human whistling. Whistling occurs when the vortex shedding is controlled by a coupling to the oscillation of an acoustic mode of the cavity. This cannot be described by a source/filter model. Vortex shedding without coupling to the acoustic field may also be a significant sound source. This depends strongly on the position of sound source with respect to the acoustic resonator.

The turbulence in the jet in the supra-glottal part of the vocal tract, generated during vocal cords oscillations, will not be of major importance for the sound production.

Fricative sound is difficult to describe accurately with a simple one dimensional model of the vocal tract because high frequencies are important. Further even at low frequencies the "source" depends on the geometry of the vocal tract (position of flow obstacles).

At the present time 3-D numerical calculations of unsteady turbulent flow at high velocities are so crude that they cannot be used for the estimation of sound production by the flow. Hence exact modelling is impossible. Interesting numerical simulations of the laminar flow in the glottis have been presented by several researchers [22-24]. These calculations should be considered as numerical experiments. They cannot be used for real time sound production.

The use even of simple non-linear models is extremely time consuming. Very interesting results based on simple non-linear models have been achieved for the clarinet [25]. In this case non-linearity (like beating of the reed) appeared to be an essential feature. Fortunately these simple models reproduce the typical sound of the music instruments without a detailed model of the flow. The use of similar models for speech simulation seems to be more promising than the detailed study of the flow proposed by Teager and Kaiser.

Acknowledgments:

I wish to thank C. d'Alessandro, P. Badin, B. Cranen, J.H. Eggen and J. Smith for teaching me the little I know about speech.

References

- [1] Teager, H.M. and Teager, S.M.
 "The effects of separated air flow on vocalizations", in *Vocal Fold Physiology*, ed. by D.M. Bless and J. Abbs, College Hill, San Diego, CA (1983) pp.124–145.
- [2] Teager, H.M. and Teager, S.M.
 "Evidence for nonlinear production mechanisms in the vocal tract", in *Speech Production and Speech Modelling*, ed. by W.J. Hardcastle and A. Marchal, Kluwer Academic Pub., Dordrecht, The Netherlands (1990).
- [3] Kaiser, J.F.
 "Some observations on vocal tract operation from a fluid flow point of view", in *Vocal Fold Physiology*, ed. by I.R. Titze and R.C. Scherer, Denver Center for Performing Arts, Denver, CO (1983) pp. 358–386.
- [4] Wilson, T.A.; Beavers, G.S.; De Coster, M.A.; Holger, D.K. and Regenfuss, J.D.
 "Experiments on the fluid mechanics of whistling", *JASA*, Vol.50 (1971) pp.366–372.
- [5] Howe, M.S.
 "Contributions to the theory of aerodynamic sound, with application to excess jet noise and the theory of the flute", *J.Fluid Mech.*, vol.71 (1975) pp. 625–673.
- [6] Howe, M.S.
 "On the absorption of sound by turbulence and other hydrodynamic flows", *IMA J. of Applied Math.*, vol.32 (1984) pp.187–209.
- [7] Blake, W.K.
 "Mechanics of flow-induced Sound and Vibration", Vol. I & II, *Applied Math.and Mech.*, Academic Press, Orlando (1986).
- [8] Schumacher, R.T.,
 "Ab-initio calculations of the oscillations of a clarinet", *Acustica*, vol. 48 (1981) pp. 71.
- [9] Lugt, H.J.
 "Vortex Flow in Nature and Technology", John Wiley & Sons, NY (1983).
- [10] Cranen, B. and Boves, L.
 "On the measurement of glottal flow", *JASA*, vol. 84 (1988) pp.888–900.
- [11] Cranen, B.
 "The Acoustic impedance of the glottis"
 Phd thesis, Katholieke Universiteit Nijmegen (1987).
- [12] Titze, I.R.
 "The physics of small-amplitude oscillation of the vocal folds"
JASA, vol.83 (1988) pp.1536–1552.
- [13] Gupta, V.; Wilson, T.A. and Beavers, G.S.
 "A model for vocal cord excitation", *JASA*, vol.54 (1973) pp.1607–1617.
- [14] Hirschberg, A.; van de Laar, R.W.A.; Marrou-Maurieres, J.P; Wijnands, A.P.J.; Dane, H.J.; Kruijswijk, S.G. and Houtsma, A.J.M.
 "A quasi-stationary model of air flow in the reed channel of single reed woodwind instruments", *Acustica*, vol.70(1990) pp. 146–154.
- [15] St.Hilaire, A.O.; Wilson, T.A. and Beavers, G.S.
 "Aerodynamic excitation of the hermonium reed", *J.Fluid Mech.*, vol.49 (1971) pp.803.
- [16] Kolkman, P.A.
 "Flow induced gate vibrations", Waterloopkundig Lab. Delft, pub. no 164 (1976).
- [17] Scherer, R.C. and Titze, I.R.,
 "Pressure-flow relationships in a model of the laryngeal airway with diverging glottis", in *Vocal Fold Physiology, Contemporary Research and Clinical Issues*, ed. by D.M. Bless and J.H. Abbs, College Hill, San Diego, Ca (1983) pp. 177–193.
- [18] Hirschberg, A.; Bruggeman, J.C., Wijnands, A.P.J. and Morgenstern, M.
 "The whistler nozzle and horn as aero-acoustic sound sources in pipe systems", *proceedings of the Institute of Acoustics*, vol.10, part 2(1988)pp. 701–708.

- [19] Shadle, C.H.
"The acoustics of fricative consonants", PhD thesis, Dept. of Elect. Engr. and Comp. Sci. MIT, Rsch. Lab. Elect. Report No 506 (1985)
- [20] Milne-Thomson, L.M.
"Theoretical Aerodynamics", Fourth ed. (1966) republication by Dover ed. (1973) NY.
- [21] Shadle, C.H.
"Models of fricative consonants involving sound generation along the wall of a tube", ICA 12, A3-4 (1985?).
- [22] Thomas, T.J.
"A finite element model of fluid flow in the vocal tract", Computer Speech and Language, vol. 1 (1986) pp. 131-151.
- [23] Iijima, H., Miki, N. and Nagai, N.
"Viscous flow analyses of the glottal model using a finite element method", The Second Joint Meeting of ASA and ASJ, Nov. (1988).
- [24] Hegerl, G.C.
"Neue ansatze zur numerischen simulation der glottesanregung und -stromung", Fortschritte der Akustik DAGA (1989) p. 323.
- [25] Ducasse, E.
"Modelisation d'instruments de musique pour la synthese sonore: application aux instruments a vent", Colloque de physique, C2, vol.51 (1990) pp. C2-837-840.

FRACTALE STRUCTUREN IN TIJDREEKSEN: ZELFSIMILARITEIT, WILLEKEUR EN CHAOS

Johan Grasman

Vakgroep Wiskunde, Landbouwniversiteit Wageningen

Vele typen fysische processen laten zich beschrijven door middel van differentiaalvergelijkingen zoals b.v. de slinger. Er zijn echter problemen waarvan we de differentiaalvergelijkingen niet kunnen afleiden vanwege de gecompliceerdheid van het proces. Voorbeelden daarvan zijn fysiologische en ecologische processen. De vraag die wij ons stellen is of in een fysisch proces een chaotische attractor onderkend kan worden zonder in te gaan op de bijbehorende differentiaalvergelijkingen. Mocht dit lukken dan kan men op deze wijze ook minder exact te formuleren processen analyseren op de aanwezigheid van zo'n chaotische attractor. De dynamica van een dergelijk proces wordt geregistreerd door middel van een waarnemingsvariabele, in b.v. een ecologisch probleem kan dat de dichtheid van een diersoort zijn. Door de invloed van seizoenen zal men in zo'n geval eenmaal per jaar de dichtheid registreren. De opeenvolgende waarden vormen een tijdreeks. Ook voor continu te meten variabelen wordt veelal op discrete tijden de waarnemingsvariabele opgeslagen.

Zelfsimilariteit

Van een fysisch biologisch of economisch proces beschikken we over een waarnemingsvariabele $y(t)$. De registratie van y vindt plaats op discrete tijdstippen t_0, t_1, \dots, t_N . Als van het proces bekend is dat het de zelfsimilariteitseigenschap heeft dan kan men op basis hiervan een invulling van de waarde van y geven op de intervallen tussen de waarnemingspunten.

Zelfsimilariteit houdt in dat een variabele zoals bij voorbeeld de rente op de kapitaalmarkt een zodanig verloop heeft dat de structuur van het diagram voor jaargemiddelden sterk gelijkert op die van maandgemiddelden. Het enige verschil is dat het renteverloop zich over een groter interval uitstrekt dan dat van maandelijks gemiddelden.

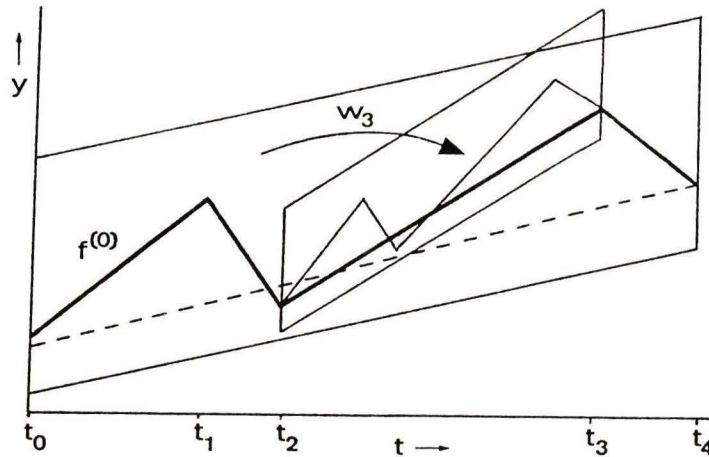
Van een proces kennen we een variabele $y(t)$ de waarde op enkele tijdstippen:

$$y(t_0) = y_0, y(t_1) = y_1, \dots, y(t_N) = y_N. \quad (1)$$

De eenvoudigste invulling van tussenliggende waarden is deelsgewijze lineaire interpolatie $y = f^{(0)}(t)$ (zie figuur 1), hiermee wordt $y(t)$ benaderd op $I = [t_0, t_N]$. Bij fractale interpolatie wordt binnen het deelinterval $I_n = [t_{n-1}, t_n]$, $n = 1, \dots, N$ de lineaire functie vervangen door een deelsgewijze lineaire functie. In figuur 1 bijvoorbeeld worden in $I_3 = [t_2, t_3]$ de tijdstippen

$$t_{20} = t_2, t_{21}, t_{22}, t_{23}, t_{24} = t_3$$

bepaald met de tussenliggende intervallen in dezelfde verhouding als I_n , $n = 1, \dots, 4$. De waarde van y in de nieuwe interpolatiepunten wordt als volgt bepaald: de afstand tussen $f^{(0)}$ en de stippellijn g in het punt t_n wordt overgebracht naar t_{2n} met een vermenigvuldigingsfactor d . In t_{2n} is $f^{(0)}$ de lijn welke het gehele deelinterval I_3 doorloopt zoals g dat doet op het interval I . In de keuze van $d \geq 0$ is men nog vrij. Het fractale interpolatieproces is convergent voor $d < 1$. Voeren we de interpolatie uit voor elk van de deelintervallen dan verkrijgen we de deelsgewijze lineaire approximatie $f^{(1)}(t)$. Hoe gaan we nu verder? De bedoeling is dat de grafiek $f^{(1)}(t)$ wederom naar elk van de deelintervallen afgebeeld wordt.



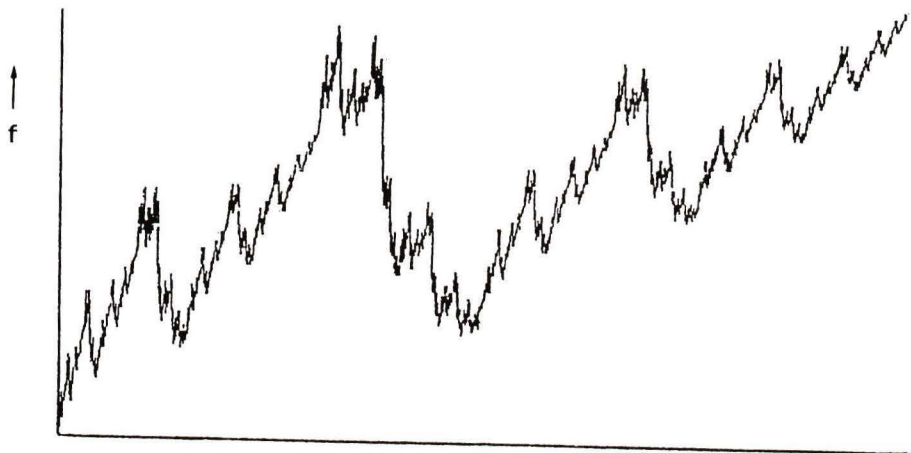
Figuur 1: De afbeeldingen naar de subintervallen in het fractale interpolatieproces.

Het is aantrekkelijk om de afbeeldingen te beschrijven als afbeeldingen van \mathbf{R}^2 in zichzelf. Voor de afbeelding van het gehele interval naar het deelinterval I_n definiëren we

$$w_n: I \times \mathbf{R} \rightarrow I_n \times \mathbf{R}. \quad (2)$$

De afbeelding w_n is samengesteld uit een lineaire afbeelding en een translatie. De werking van w_n is af te lezen in figuur 1. Het parallellogram dat zich over I uitstrekt wordt afgebeeld naar het parallellogram over I_n (in figuur 1 alleen voor $n = 3$ geschetst). Zoals gebruikelijk bij de beschrijving van een iteratieproces wordt $f^{(k)}(t)$ bekend verondersteld en wordt de formule gegeven die $f^{(k+1)}(t)$ levert uit $f^{(k)}(t)$.

Het resultaat voor $k \rightarrow \infty$ wordt bepaald met het random iteratie-algoritme, zie figuur 2. Dit werkt als volgt. We nemen een punt op de interpolatiecurve, b.v. één van de steunpunten. We kiezen random een getal uit de verzameling $\{1,2,3\}$ (met kans $p_i > 0$, $p_1 + p_2 + p_3 = 1$). Dit getal s_1 geeft aan welke afbeelding toegepast wordt: w_{s_1} . Dit wordt herhaald voor het nieuwe punt en alle punten worden getekend.



Figuur 2: Fractale interpolatie levert f^∞ .

Chaos en fractale dimensie

We beschouwen het voorbeeld van een signaal $y(1), y(2), \dots$ dat gegenereerd wordt door een iteratie-afbeelding:

$$x(n+1) = 1 - ax(n)^2 + y(n), \quad (3a)$$

$$y(n+1) = bx(n). \quad (3b)$$

Voor $n \rightarrow \infty$ doorlopen de punten $(x(n), y(n))$ de zogeheten Henon-attractor. Voor bijvoorbeeld $(a,b) = (1.4, 0.3)$ heeft deze attractor een chaotisch gedrag. De verzameling punten $\{(x(n), y(n)), n = 1, 2, \dots, N\}$, heeft een fractale dimensie D , welke volgt uit de correlatie-integraal

$$C(\epsilon) = \lim_{N \rightarrow \infty} \frac{\text{aantal puntenparen met onderlinge afstand} < \epsilon}{N^2}.$$

Er geldt namelijk

$$D = \lim_{\epsilon \rightarrow \infty} \frac{\log C(\epsilon)}{\log \epsilon}.$$

Voor genoemde parameterwaarden: $D = 1.2$.

Reconstructie van een dynamisch systeem

Van vele fysische en biologische processen wordt met behulp van de reconstructie-methode de dimensie van een mogelijk aanwezige chaotische attractor bepaald. Daarbij wordt ervan uitgegaan dat het proces zich laat beschrijven door een niet-lineair iteratieproces

$$x(t+h) = F(x(t)) \quad (4)$$

Het systeem wordt uitgelezen aan de hand van een observatie-variabele b.v. de eerste toestandsvariabele $x_1(t)$ op tijdstippen $t_n = hn$. Het gereconstrueerde systeem heeft als toestandsvector

$$z(t_n) = \{x_1(t_n), x_1(t_{n-1}), x_1(t_{n-2}), \dots, x_1(t_{n-d+1})\}.$$

In de d -dimensionale toestandsruimte met d voldoende groot wordt de chaotische attractor gerepresenteerd door de punten $z(t_1), z(t_2), \dots$.

We illustreren dit aan de hand van een tijdreeks geproduceerd door de Hénon attractor, waarbij we aannemen dat niet bekend is dat het de Hénon attractor betreft. Ook de dimensie van het systeem wordt niet bekend verondersteld. De afbeelding (3ab) genereert de punten z_n volgens

$$z_{n+1}^{(1)} = 1 - a(z_n^{(1)})^2 + bz_n^{(2)},$$

$$z_{n+1}^{(2)} = z_n^{(1)},$$

...

$$z_{n+1}^{(d)} = z_n^{(d-1)}.$$

Voor $d \geq 2$ levert dit, zoals beschreven, voor de verzameling punten z_1, z_2, z_3, \dots in \mathbf{R}^d een correlatiedimensie $D = 1.2$. Voor $d = 1$ gaat het fout en kan een systeem van het type $z_{n+1}^{(1)} = F(z_n^{(1)})$ niet de observatiewaarden genereren. In dat geval zou de verzameling $z_1^{(1)}, z_2^{(1)}, z_3^{(1)}, \dots$ op \mathbf{R} als correlatiedimensie een waarde kleiner of gelijk 1 hebben, hetgeen onjuist is. Dit geeft aan hoe we bij de tijdreeks $x_1(t_1), x_1(t_2), x_1(t_3), \dots$ van een systeem met onbekende dynamica (4) en onbekende dimensie van de toestandsruimte te werk gaan. We bepalen $D(d)$ voor $d = 1, 2, 3, \dots$ en gaan na wanneer de verzadiging optreedt, d.w.z. wanneer D niet meer stijgt bij toenemende d .

Voorspellen

Op basis van de tijdreeks $x_1(t_j)$, $j = 0, 1, 2, \dots$ kan met behulp van het model van vertraagde coördinaten z een voorspelling gemaakt worden van de waarde van x_1 in de toekomst; zeg op het tijdstip $t + T$. We doorlopen de gegeven tijdreeks over interval $t_j \in [0, t)$ en registreren de waarden $z(t_{j_i})$, $i = 1, 2, \dots, k$ van de k het dichtst bij $z(t)$ liggende waarden. Een geschikte interpolatie van $z(t_{j_i} + T)$, $i = 1, 2, \dots, k$ levert een voorspelling van $z(t+T)$. De kracht van een dergelijk voorspellingsmodel ligt in het behoud van het niet-lineaire karakter in het voorspelsysteem, nl. de oplossingskurven uit een verleden. Een dergelijk voorspellingsmodel kan dus in principe plotselinge toekomstige veranderingen verwerken. Echter gezien het sterke divergente karakter van oplossingen kan het voorkomen dat $z(t_{j_i} + T)$, $i = 1, \dots, k$ zich over twee ver uit elkaar liggende subdomeinen van de toestandsruimte uitspreidt, zodat het antwoord sterk van de gekozen interpolatietechniek afhangt: een "gemiddelde" oplossing kan dan in een domein terecht komen dat ver van de attractor verwijderd is. Farmer en Sidorowich (1987) geven een methode om tot een geschikt interpolatieschema te komen.

Literatuur

Barnsley, M. (1988), *Fractals everywhere*, Adac. Press, New York.

Crutchfield, J.P., J.D. Farmer, N.H. Packard and R.S. Shaw (1986), *Chaos*, Scientific American **225**, 38-49.

Farmer, J.D., J.J. Sidorowich (1987), *Predicting chaotic time series*, Phys. Rev. Letters. **59**, 845-848.

Grasman, J. (1990), *Fractale structuren in de dynamica van systemen: zelfsimilariteit, willekeur en chaos*, in Dynamische Systemen en Chaos, H.W. Broer en F. Verhulst (red.), Epsilon Uitgaven, Utrecht, 172-195.

Jürgens, H., H.-O. Peitgen en D. Saupe (1990), *The language of fractals*, Scientific American, August, 40-47.

PSOLA manipulatie van spraak

L.L.M. Vogten, IPO-Eindhoven

1 Inleiding

Voor het spraakonderzoek zijn technieken om prosodische eigenschappen van spraakgeluid te manipuleren van grote betekenis. Bij die manipulatie willen we graag een zo hoog mogelijke kwaliteit en natuurlijkheid behouden. De conventionele LPC-parameterbeschrijving van spraakgeluid met een bron-filtermodel levert goede mogelijkheden tot manipulatie, omdat alle modelparameters onafhankelijk van elkaar als functie van de tijd zijn gespecificeerd. Veranderingen van de grondtoon F_0 in een spraakuiting worden gerealiseerd door bij stemhebbende klanken de herhalingsfrequentie van de excitatie (het bronsignaal) te wijzigen. Ook de duur van spraaksegmenten kan eenvoudig worden gemanipuleerd door de frameduur, zoals die bij de analyse is gekozen, bij resynthese van de afzonderlijke frames te verlengen of te verkorten. Immers, de tijdsduur waarover de impulsresponsie van het filter voor ieder frame wordt berekend kan willekeurig verlengd of verkort worden, ongeacht de herhalingsfrequentie van de excitatie.

Kwaliteit en natuurlijkheid van de aldus LPC-gesynthetiseerde en gemanipuleerde spraak laten echter soms te wensen over. Dat komt ten dele vanwege fundamentele beperkingen in het bron-filtermodel en ten dele door de wijze waarop in de standaard LPC-techniek de modelparameters worden bepaald. Zo voorziet het model niet in stemhebbende wrijfklanken, als combinatie van periodiek en ruisig geluid, en worden plofklanken, die intrinsiek niet stationair zijn, bij LPC-analyse met een venster van enkele tientallen ms soms hoorbaar aangetast. Daarnaast kan een te laag gekozen aantal filtercoëfficiënten klinkers en nasalen aantasten, waardoor het timbre verandert en de resynthese soms zoemend ("buzzy") klinkt. Verder leiden fouten in de automatische toonhoogtemeting en stem-stemloosbeslissing vaak tot een minder goede resynthese en kunnen eveneens bijdragen tot het "buzzy" karakter van de geresynthetiseerde spraak. Tenslotte hebben ook een minder geschikte spreekstem of ongunstige akoestische omstandigheden waaronder de microfoonopnamen zijn gemaakt (nagalm, achtergrondlawaai) een negatieve invloed op de resynthesekwaliteit.

Voor zulke gevallen zou een andere methode van manipulatie wellicht tot betere spraakkwaliteit leiden, die minder verschilt van het origineel. Recent werk van Charpentier & Moulines (1989) van het franse CNET wijst erop dat de door hen ontwikkelde techniek voor manipulatie in de golfvorm

van spraak- (en ander) geluid inderdaad tot natuurlijker spraak leidt. Deze techniek, PSOLA ("Pitch Synchronous OverLap and Add") geheten, is door Verhelst (1990) in het IPO geïntroduceerd en daarmee is inmiddels enige ervaring opgedaan.

In deze bijdrage aan het Colloquium SignaalAnalyse en Spraak (COLSAS) zullen we die ervaringen weergeven. Eerst gaan we kort in op het principe van de PSOLA-methode. Daarna demonstreren we enkele praktische resultaten, aan de hand van een korte beschrijving van twee in het IPO ontwikkelde interactieve programma's voor het manipuleren van toonhoogte en duur. We besluiten met een vooruitblik op mogelijke toepassingen in het licht van de pro's en contra's van de PSOLA-techniek vergeleken met de conventionele manipulatie via de LPC-techniek.

2 De PSOLA-methode

In de analysefase wordt de golfvorm op regelmatige afstanden ("pitch synchron") voorzien van markeringen, bijv. op de positieve nuldoorgangen van het signaal aan het begin van de iedere grondtoonperiode. In stemloze stukken waarin geen duidelijke periodieke structuur aanwezig is, worden de markeringen op vaste tijdsafstanden aangebracht. Ter weerszijden van iedere markering wordt de golfvorm "uitgepoort" volgens halve "raised cosine" vensters (Hanning, zonder "stoepje"), zodat buiten de vensters het signaal nul is. De vensters hebben dus een breedte van twee periodes van de grondtoon en zijn bij niet-constante toonhoogte (licht) asymmetrisch. Daarmee is de oorspronkelijke golfvorm geanalyseerd in een rij korte golfjes, hier verder frames genoemd, die elkaar overlappen en die zijn verkregen via pitch synchron tijdsvensteren. In fig. 1 is dit schematisch aangegeven.

De synthese bestaat slechts uit het bij elkaar optellen van alle frames. Zolang de tijdsafstand tussen de frames daarbij niet verandert t.o.v. de originele, krijgen we uiteraard de oorspronkelijke golfvorm weer terug. Veranderingen in bijv. de toonhoogte verkrijgen we door die afstanden te verkorten (verhoging van de grondtoon F_0) of te verlengen (verlaging van F_0). Dat is schematisch weergegeven in fig. 2. Zodra de nieuwe F_0 minder dan de helft is van de originele, wordt de golfvorm tussen de opeenvolgende nieuwe grondtoonperiodes nul. De lengte van die "lege stukken" neemt toe met afnemende F_0 . Ook bij zeer sterke verhoging van F_0 zal de golfvorm van de nieuwe periodes sterk wijzigen t.o.v. de oude.

Duurveranderingen kunnen worden gerealiseerd door bij gelijkblijvende onderlinge afstanden hele frames weg te laten (verkorte duur) of selectief te herhalen (verlengde duur), zoals in fig. 3 is weergegeven. Bij vertragingen met factoren 2 of meer kunnen in de stemloze stukken door de herhaling van (ruis)frames ongewenste toonhoogte-effecten worden optreden. Die kunnen worden tegengegaan door de stemloze frames bij de opeenvolgende herhalin-

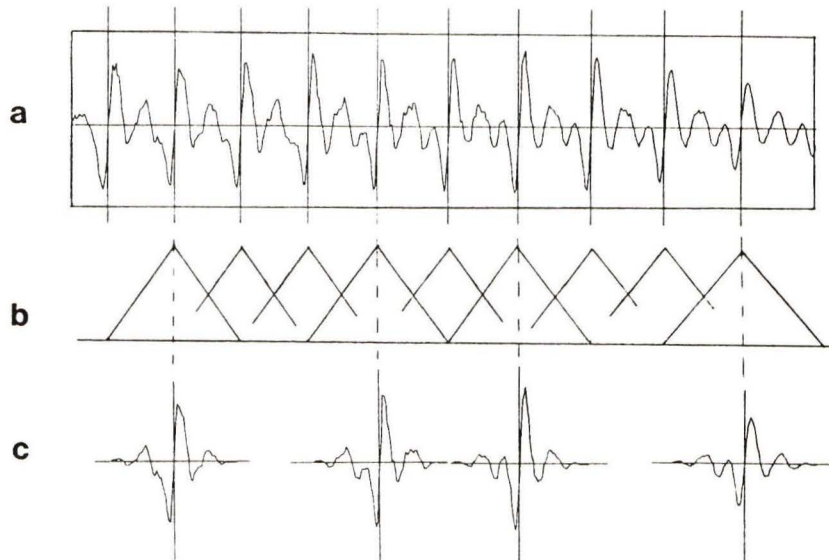


Fig. 1: Schematische weergave van de PSOLA analyse. In de oorspronkelijke golfvorm (a) worden "pitch synchroon" markeringen aangebracht, waaromheen de golfvorm wordt uitgepoort (b) tot korte golfjes "frames" (c). De vensters zijn hier schematisch als lineair weergegeven; in werkelijkheid verlopen ze volgens een \cos^2 -functie.

gen in de tijd om te keren (Charpentier & Moulines, 1989). Zeer sterke vertragingen (van factoren 4 of meer) leiden ertoe dat de veelvuldige herhalingen van eenzelfde frame hoorbaar worden als stationaire stukken met abrupte overgangen naar een volgend stationair stuk. Een simpele lineaire interpolatie tussen de opeenvolgende frames zou dit effect kunnen vermijden.

3 Implementatie in het IPO

De techniek van PSOLA-manipulatie is in het IPO praktisch toegepast in twee grafisch interactieve programma's: MOP ("MODify Pitch", ontwikkeld door Allain) en MAD ("MANipulate Duration", ontwikkeld door Eggen). De onderzoeker kan daarmee een willekeurig segment met cursor/duimwielen specificeren en beluisteren. In MOP kan binnen zo'n segment de toonhoogte lineair worden geïnterpoleerd, vermenigvuldigd met een bepaalde factor of worden gecopieerd uit een al bestaande LPC-parameter-file waarin F_0 al gestileerd is, bijv. met een ander automatisch programma. Het resultaat kan direct worden beluisterd en vergeleken met de oorspronkelijke golfvorm. In MAD kunnen op soortgelijke wijze binnen willekeurig aangewezen segmenten zowel de duur als de amplitude lineair worden geïnterpoleerd of met een vrij te kiezen factor worden vermenigvuldigd. In beide programma's kan de gebruiker kiezen of hij de manipulaties op alleen de stemhebbende, alleen de stemloze of op beide wil uitvoeren. Fig. 4 en 5 geven een indruk van het-

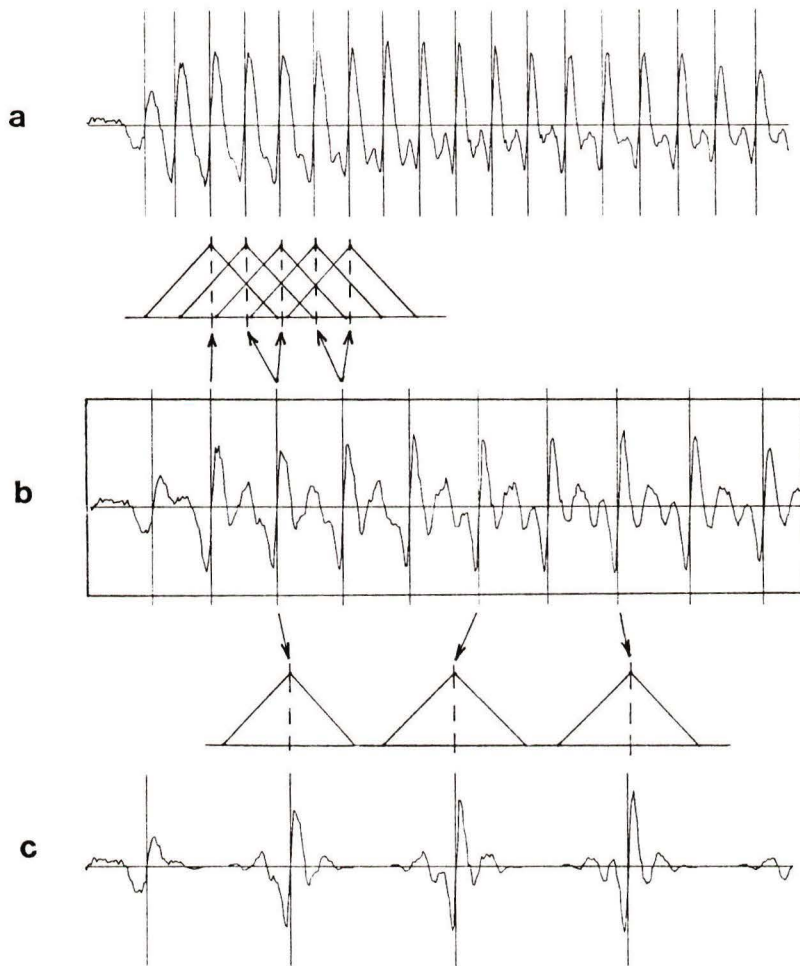


Fig. 2: Toonhoogteveranderingen met de PSOLA techniek. Verhogen van F_0 (hier met een faktor 1.9) vindt plaats door de tijdsafstand tussen de oorspronkelijke markeringen (b), aangebracht bij de analyse, overeenkomstig de nieuwe F_0 te verkleinen, daarbij eventueel frames te herhalen, en dan de frames bij elkaar op te tellen (a). Verlagen van de grondtoon (hier met een faktor 0.4) gebeurt door die afstanden te vergroten, zo nodig met weglating van frames, en op te tellen (c).

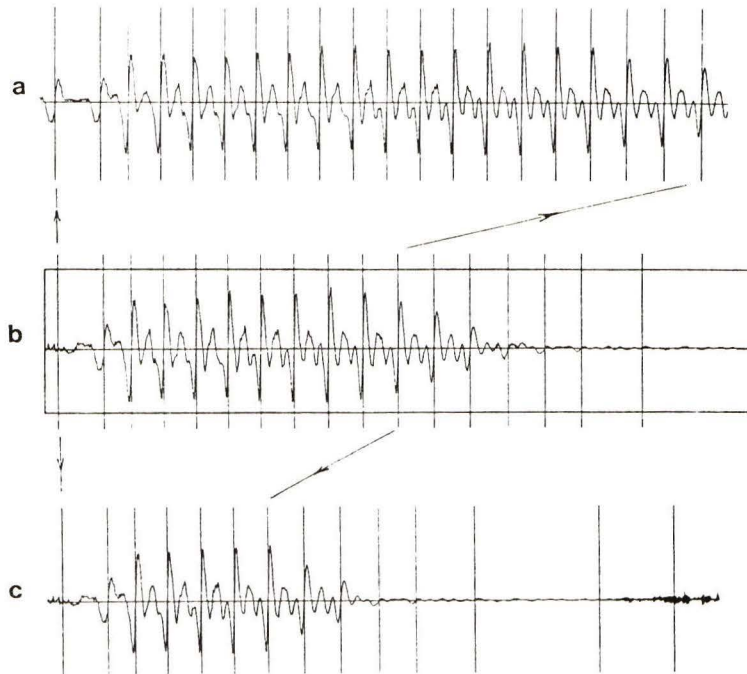


Fig. 3: Duurveranderingen worden met PSOLA gerealiseerd door zoveel frames te herhalen of selectief weg te laten als nodig is om de nieuwe duur van het betreffende segment te verkrijgen. In dit voorbeeld is bij (a) de duur verlengd (faktor 1.9) en bij (c) verkort (faktor 0.6).

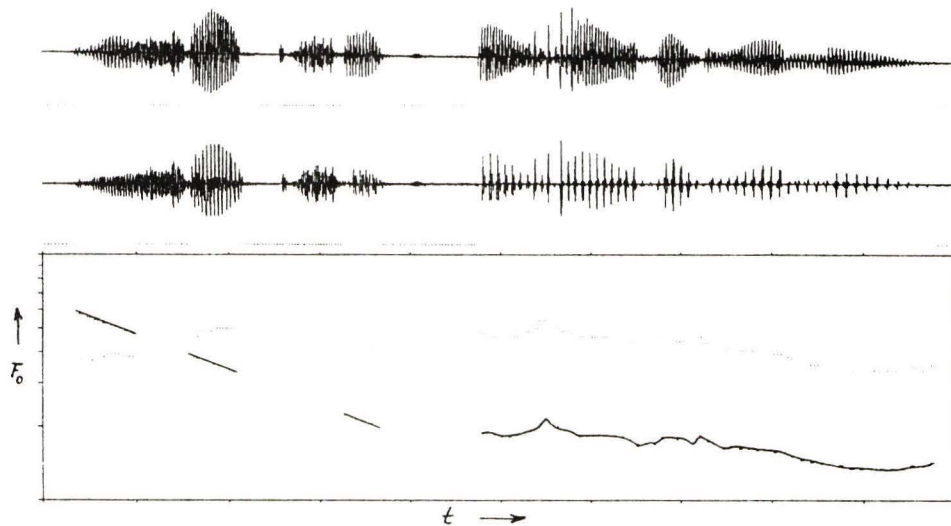


Fig. 4: Voorbeeld van interactieve verandering van de toonhoogte via PSOLA met het programma MOP. Boven: originele golfvorm met daaronder een indicatie voor de stemloze fragmenten. Daaronder de F_0 -gemanipuleerde golfvorm met eveneens de stemloze stukken. De onderste helft van de figuur toont het oorspronkelijke F_0 -verloop (gestippeld, op log schaal) in de tijd, en de veranderingen daarin aangebracht (doorgetrokken). In het eerste deel van de uiting is F_0 lineair geïnterpoleerd, in het tweede deel verlaagd met een faktor 0.4.

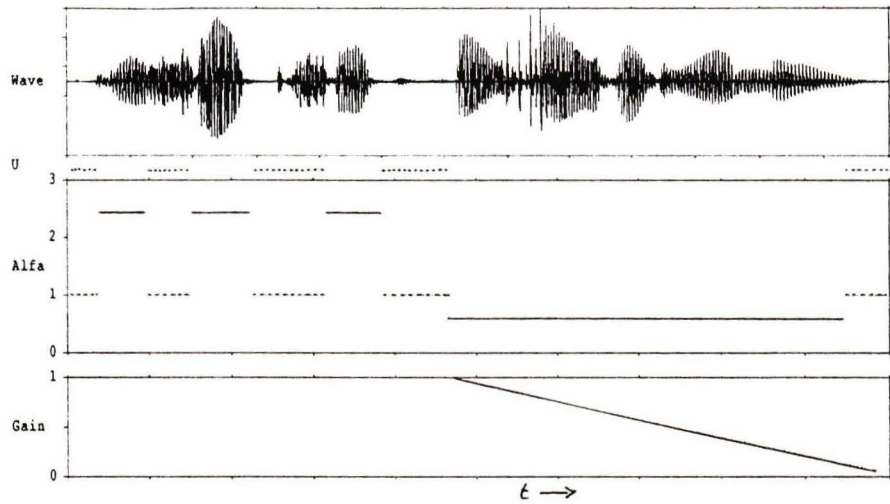


Fig. 5: Voorbeeld van interactieve verandering van de duur via PSOLA met het programma MAD. Boven: originele golfvorm met daaronder een indicatie voor de stemloze fragmenten. Daaronder het verloop van de faktor (alpha) waarmee het te kiezen fragment in duur vermenigvuldigd wordt bij de synthese. In dit voorbeeld zijn de stemhebbende stukken van het eerste deel verlengd (faktor 2.5) en is het tweede deel van de zin in duur verkort (faktor 0.6). Het onderste deel toont het amplitudeverloop ("gain"), dat in het tweede deel van de zin lineair afneemt van 1 naar 0.

geen de onderzoeker voor zich in beeld heeft. Tijdens COLSAS zullen o.m. deze programma's gedemonstreerd worden.

4 Toepassingen: PSOLA versus LPC

Een aantrekkelijk aspect van de PSOLA golfvorm-manipulatie techniek is de mogelijkheid om toonhoogte en duur te manipuleren met behoud van zeer goede natuurlijkheid (kwaliteit) van het spraaksignaal. Bovendien is ze zeer eenvoudig en vergt relatief weinig rekentijd. Daarom zal ze binnen het spraak- onderzoek een belangrijk hulpmiddel kunnen worden, met name voor het intonatie- en duur-onderzoek. In die gevallen waarin de oorspronkelijke spraak onder akoestisch slechte omstandigheden is opgenomen (grote microfoonafstand, nagalm, achtergrondlawaai) zal de PSOLA-techniek in principe tot aanzienlijk betere resynthese kunnen leiden dan LPC. Bij de ontwikkeling van duurregels voor toepassing in o.m. automatische systemen voor tekst-naar-spraakomzetting biedt ze zeer goede vooruitzichten. Ook bij de spraaksynthese d.m.v. difoonconcatenatie kan ze wellicht tot betere spraakkwaliteit leiden. Aanwijzingen daarvoor komen van het franse CNET, waar de PSOLA-difoonsynthese op basis van golfvormconcatenatie, kwalitatief duidelijk superieur is t.o.v. de LPC-versies (Hamon et.al., 1989).

Toch kleven er ook nadelen en problemen aan de PSOLA-techniek. Uiteraard kost golfvormmanipulatie een factor 10 meer opslagcapaciteit dan een standaard LPC-techniek en 100 keer meer dan zuinig gecodeerde LPC. Dat weegt bij een goede LPC niet altijd op tegen het (niet altijd spectaculaire) verschil in kwaliteit en natuurlijkheid.

Een veel belangrijker hinderpaal voor toepassing van de PSOLA-techniek op grote schaal is het feit dat de methode pitch synchroon werkt. De markeringen moeten zorgvuldig, correct en consistent in de golfvorm worden aangebracht. Fouten in de markering leiden bij manipulatie tot slechte, hese spraak, "dubbele" toon- hoogtes enz. Visuele correctie van die fouten vereist vooralsnog veel interactief handwerk, is in de praktijk vaak lastig en bij onregelmatige golfvormen soms zelfs onmogelijk. Daar komt nog bij dat eventuele fouten in de markeringen pas na manipulatie en synthese manifest worden. Pas dan blijkt bijv. dat een constante frame-afstand bij synthese niet de gewenste constante toonhoogte oplevert en dat de oorspronkelijke F_0 er soms dwars doorheen klinkt.

5 Conclusie

Eerste ervaringen in het IPO opgedaan met de PSOLA techniek openen goede perspectieven voor golfvormmanipulatie van toonhoogte en duur, met behoud van zeer goede spraakkwaliteit en natuurlijkheid. Daarom alleen al

vormt de PSOLA techniek een nuttige aanvulling op het arsenaal gereedschappen voor het spraakonderzoek. Duurmanipulatie kan plaats vinden zonder enig verlies van natuurlijkheid. Toonhoogtemanipulatie is voornamelijk slechts op beperkte schaal toepasbaar omdat bij grote verschillen tussen oorspronkelijke en nieuwe F_0 vaak storende effecten hoorbaar zijn.

In de analysefase is het correct aanbrengen van de markeringen in de golfvorm van zeer groot belang voor F_0 -manipulatie in de synthesefase. Het vinden van goede, robuuste algoritmes voor automatische markering is dan ook essentieel voor ruimere toepassing van de PSOLA techniek in intonatieonderzoek en in de spraaksynthese.

6 Referenties

- Charpentier, F., Moulines, E. (1989) Pitch synchronous waveform processing technique for text to speech synthesis using diphones, Proc. Eurospeech Paris, 13-19.
- Hamon, C., Moulines, E., Charpentier, F. (1989) A diphone synthesis system based on time domain modifications of speech, Proc. FASE Int. Conf., Edinburgh, 238-241.
- Verhelst, W. (1990) An implementation of the PSOLA/KDG waveform synthesis technique, IPO report 733.

RULE EXTRACTION FOR ALLOPHONE SYNTHESIS

Louis ten Bosch

Department of Language and Speech, Phonetics Section
Erasmusplein 1, P.O. Box 9103
6500 HD Nijmegen, The Netherlands

ABSTRACT

A method will be presented for extracting rules from a labelled speech database in order to find context-dependent allophone rules. The classical approach to this problem involves parameter fitting by e.g. least-squares error minimization on a sufficiently large dataset. In that approach, the type of interaction is often to be chosen beforehand. Moreover, there remains a general problem, how to improve the output of linearly ordered rule sets.

The present algorithm searches for the interaction in a broad class which can be modified interactively. It is partly based upon the classical approach, and partly on (non-linear) matrix manipulation. From the theory, we will discuss the question, how to construct an 'optimal' linearly ordered rule set.

1 Introduction

In this paper, we present a method for the extraction of 'rules' from 'data'. By data we mean in general parameter data obtained from natural speech or from a speech database, such as diphone sets. 'Rules' are algorithms used e.g. in rule-based allophone synthesis systems, and specified by a focus, a context specification and an action (cf. Loman et al., 1989). Such a rule extraction meets the problem, how to find underlying structure in data. In most cases, such a problem is untractable, unless we have a *model* that can be fitted to the available data.

A rule extraction method may be a useful tool for the improvement of rule sets used in allophone synthesis applications, as it may accelerate the optimization process. In the following, we go into detail on the theory and background of the present method. The material extends the presentation in Ten Bosch (1989).

2 Preliminaries

In this section, we first consider a simple example.

Suppose we have a labelled speech database in which [a] occurs sufficiently many times. The first formant F_1 of [a] is on the average equal to 700 (Hz),

except for [a] in unstressed syllables where the average F_1 equals 650. We say that the *parameter* F_1 depends on the *function* [stress]. For simplicity, we here assume that this function is binary-valued only. Other functions are e.g. [height], [front], [round], [plosive], [nasal], [voice], [fric], etc. Other parameters are e.g. the formant frequencies F_i , bandwidths B_j , the energy, LPC-parameters, etc. Functions and parameters are phonologically and phonetically inspired, respectively.

The corresponding rule might read:

if focus = [a] and syllable is unstressed then	$F_1 := 650$
if focus = [a] and syllable is stressed then	$F_1 := 700$

or

if focus = [a] then	$\begin{cases} F_1 := 650 \\ F_1 := F_1 + 50 \text{ if } [\text{stress}](\text{syll}) \end{cases}$
---------------------	--

Here '[stress]' is formulated as a predicate with 'syll' as an argument, defined by

$$[\text{stress}](\text{syll}) = \begin{cases} 1 & \text{if the syllable syll is stressed} \\ 0 & \text{if the syllable syll is unstressed} \end{cases}$$

In this example, [stress] is a binary-valued function. If the two values are 0 and 1, we call it a *boolean*. Other functions (e.g. [height]) may attain more different values.

Also for the focus, we can introduce the predicate 'focusA' with argument 'focus' by defining

$$[\text{focusA}](\text{focus}) = \begin{cases} 1 = \text{true} & \text{if 'focus' equals [a]} \\ 0 = \text{false} & \text{otherwise} \end{cases}$$

Accordingly, the above rule has a 'numerical' variant:

If [focusA](focus) then	$F_1 := 650 + 50 [\text{stress}](\text{syll})$
-------------------------	--

or, explicitly,

$$\begin{aligned} F_1 &:= (650 + 50[\text{stress}](\text{syll})) \cdot ([\text{focusA}](\text{focus})) \\ &= 650 \cdot ([\text{focusA}](\text{focus})) + 50 \cdot [\text{stress}](\text{syll}) \cdot ([\text{focusA}](\text{focus})) \end{aligned} \quad (1)$$

More difficult rules can be formulated by using more functions, by involving the same type of logic. Also, many functions can be decomposed into other ones. For example, [focusA] can be factored out into [vowel] \times [low] \times (1-[nasal]) \times [back].

Table 1: A representation of a speech database

Functions				Parameters		
[focusA]	...	[stress]	F_1	...
0	...	0	510	...
0	...	1	350	...
1	...	1	700	...
1	...	1	710	...
1	...	0	660	...
⋮						
0	...	0	405	...

We see that the final, ‘numerical’ rule 1 corresponds to the initial rule ‘in words’ we started with. We found a translation from a phonological rule to a numerical rule. It will be evident that many phonological rules can be translated in this manner. One can also observe that there exist a trade-off between context specification and action specification in the rules. In the rule we started with

if focus = [a] and syllable is unstressed then	$F_1 := 650$
if focus = [a] and syllable is stressed then	$F_1 := 700$

the context specification is rather elaborate (‘focus must be [a]’, ‘syllable must be stressed or unstressed’); the action was very simple: $F_1 := 650$ or $F_1 := 700$. In the last rule:

$$F_1 = 650 \cdot ([\text{focusA}](\text{focus})) + 50 \cdot [\text{stress}](\text{syll}) \cdot ([\text{focusA}](\text{focus}))$$

we do not have any context specification, but the action is rather elaborate. In fact, the context specification is put into the action section of the rule by using the feature functions for the action specification. This phenomenon is very general and can be studied on its own (ten Bosch, 1990).

We now pose the question, how to find such ‘numerical’ rules in a ‘more or less’ automatic way. In this paper we will present an idea to cope with that problem. Our method is based on a trick with respect to the interpretation of the solution of a minimization problem. The method presented here is still to be refined and improved.

Before we explain our approach, we first represent the speech database in a tractable way. One possible representation is shown in table 1.

The above F_1 -rule (rule 1) can in principle be found automatically from table 1 by considering all lines in the table that contain a ‘1’ in the function column [focusA], by minimization of a vector expression $\|Ax - b\|$. Here A is an unknown matrix derived from the left-hand side of table 1, consisting of the columns [focusA] and [stress] and their product; b is a known parameter

vector containing the parameters in the F_1 -column in the right-hand side, and x is the unknown weighting vector.

After minimization of $\|Ax - b\|$, the expression Ax will be an approximation of the parameter vector b . In general, Ax will have the form (p denotes the parameter, [fun] a function)

$$p := \underbrace{a_0}_{\text{constant}} + \underbrace{a_i[\text{fun}]_i + \dots}_{\text{linear terms}} + \underbrace{a_{ij}[\text{fun}]_i [\text{fun}]_j + \dots + \dots}_{\text{quadratic terms}} \quad (2)$$

The algorithm searches for the matrix A and the vector x such that Ax optimally approximates b (Golub & Van Loan, 1983). The columns of matrix A correspond to values of functions, or to values of powers or products of them. These products correspond to compound conditional statements. For example,

$$F_1 := 340 - 50[\text{height}][\text{front}](\text{focus}) + 20[\text{stress}](\text{syll})$$

corresponds to

$$\begin{array}{l}
 F_1 = 340 \text{ (from table)} \\
 \text{if focus is front then} \left\{ \begin{array}{ll}
 \text{no action} & \text{if } [\text{height}](\text{focus}) = 0 \\
 F_1 := F_1 - 50 & \text{if } [\text{height}](\text{focus}) = 1 \\
 F_1 := F_1 - 100 & \text{if } [\text{height}](\text{focus}) = 2 \\
 F_1 := F_1 - 150 & \text{if } [\text{height}](\text{focus}) = 3 \\
 \vdots & \\
 \text{if focus is contained in stressed syllable then } F_1 := F_1 + 20
 \end{array} \right.
 \end{array}$$

In general, equation 2 corresponds to a default parameter setting and a sequence of interchangeable (eventually compound) if-statements.

The problem how to compose the matrix A out of the left-hand data in table 1 is probably very hard ('NP-hard', Lenstra, personal communication). It corresponds to the search of the shortest non-trivial vector in a (high-)dimensional lattice. In our case, however, we can incorporate intuitive phonetic knowledge to simplify our task and to reduce the dimension and size of the solution space.

The solution vector x will have phonetic 'sense' only if the number of columns will be restricted to a certain maximum M . For practical purposes, M was set to 5. $M = 5$ corresponds to a parameter dependence of maximally five functions or to an if-nesting of five levels.

The approximation Ax to b in equation 2 is assumed to be adequate as in practice parameter settings will be of the three following forms: (1) $p := p + c$, (2) $p := c$, or (3) $p := \tilde{p} + c$, where c denotes some context-dependent positive or negative constant, and \tilde{p} a parameter unequal to p . In case (3), parameters at the right side of table 1 depend on other parameters at the right side of that table. All the three cases (1), (2) and (3) are subcases of equation 2.

The algorithm allows a column of A to correspond to $[\text{fun}]_i^{e_i}$, where e_i is a positive integer. In order to reduce CPU-time, e_i must be limited to some upperbound E . In the algorithm, $E = 3$ was taken. This is a safe upperbound. If $[\text{fun}]$ is a boolean, we have $[\text{fun}]^2 \equiv [\text{fun}]$, so $[\text{fun}]^k \equiv [\text{fun}]$ for all $k \geq 1$. Accordingly, in the case of booleans, putting $e_i > 1$ is senseless. In other cases, where $[\text{fun}]$ is not a boolean, parameters are never observed to depend on functions to more than degree 2. In one actual example (the Dutch allophone synthesis), E can be set to 1, due to the fact that contexts are specified sufficiently narrowly.

M , the number of columns in A , and E , the maximal exponent of the functions used, depend on the width of the focus and the context of the rule in question. The smaller the domain of the rule, the more specific the rule can be, and the lower M and E can be, as possible context dependencies can be put into the constants a_i, a_{ij}, \dots , etc. This is again a consequence of the trade-off rule between context and action we mentioned before.

The algorithm has now been applied to the phonemes /a/, /i/, /u/, /p/, /t/, /k/ in different contexts, yielding a rule-based versions of utterances /pit/, /put/, /kap/, etc. The data were taken from a set of accented ('full') diphones available from a Dutch professional speaker. In the presentation, we will give a demonstration of such rule-based 'allophone' speech.

It must be observed that the least-squares minimization does not ensure that the *quality* of the resulting allophone speech actually increases. The relations between speech quality and speech parameters are too complex, and, in any case, non-linear. The current optimization only deals with the interpretation of the data as found in the database. As a consequence, rule optimization without thorough perceptual feedback will most likely be impossible. The present algorithm however produces a first 'good guess', based upon speech data, for an allophone rule set.

3 Application in rule sets

By means of the algorithm, it is possible to detect regularities in data that can be represented by rules. Those rules have to be of special form, as we have seen above. Each approximation Ax to b yields a rule, which is denoted R_i . R_i is decomposable into one parameter setting and a sequence of compound if-statements. A linear rule set is represented by (R_1, R_2, \dots, R_k) , where R_{i+1} appears 'later' than R_i . D_i denotes the domain of the rule R_i . In order to avoid rule correlation, the domains D_i in the ideal case must obey the following constraint: if $i < j$ then either $D_j \subset D_i$ or $D_i \cap D_j = \emptyset$. In other words: a 'later' rule possibly corrects the output of a former rule only on a subdomain. For the sake of rule improvement, such a rule set behaves best. In practical cases, this structure is often not possible. For example, suppose that at first the labials are dealt with, and that rules concerning plosives occur later in the rule set. If the [p] results incorrectly and other labials and plosives result correctly, we are likely to be forced to adopt a [p]-rule of which the domain is a subdomain of both the labial-rule and the plosive-rule. The construction of such 'auxiliary' rules however may be inspired by perceptual

rather than by numerical findings, and depends on how the rule set precisely acts on the input phoneme string (Ten Bosch, 1990).

The solution for this problem in the present algorithm is, to take such dependencies of two functions together in one global rule, which is found on the domain of labials as well as plosives. This yields a rule with a compound statement involving constants a_i , a_{ij} , as we have seen above. After having found that global rule, severe mismatches may remain for specific function values, e.g. [plosive] = 1, [voice] = 0 and [labial] = 1, pointing to [p]. By the algorithm, an auxiliary [p]-rule is then suggested to correct the output of the former global rule.

In the present implementation, the procedure can be dealt with interactively. Domain specification, focus, and the upperbounds M and E can be defined before and during the minimization process.

4 Conclusion

A method is presented for the extraction of rules from data for the purpose of allophone synthesis. The algorithm is based upon the classical minimization principles, but extended with some essential features. It searches for an optimal additive-multiplicative model within user-defined restrictions. We made a relation between the different formats of rules appearing in rule sets on the one hand, and the numerical variants as found from numerical algorithms on the other. The present experimental implementation allows us to construct an 'initial guess' for the allophone rule set from a diphone data set. Perceptual evaluation must lead to further optimization of this set.

Acknowledgement

This research was sponsored by the Dutch Program for the Advancement of Computer Science on Analysis and Synthesis of Speech SPIN-ASSP.

References

- Ten Bosch, L.F.M. (1989). From diphones to allophone rules. Proceedings AFN, Nijmegen. Vol. 13, p. 41-49.
- Ten Bosch, L.F.M. (1990). On the application of rule sets on linear symbol strings. (in preparation).
- Loman, H., Kerkhoff, J., and Boves, L. (1989). A working environment for speech synthesis by rule. Proceedings AFN, Nijmegen. Vol. 13, p. 51-63.
- Golub, G.H., and Van Loan, C.F. (1983). Matrix computations. North Oxford Academic.

Spraakherkenning

Speech Recognition Using Connectionist Methods

Christian J. Wellekens

Philips Research Laboratory Brussels, Av. Em. Van Becelaere 2, Box 8

B-1170 Brussels, Belgium.

Phone: (+32) 2 6742275, E-mail: wlk@prlb2.uucp

Abstract—*The difficulties of speech recognition are analyzed. Classical approaches are described and their weak points are brought out: non discriminant training, difficulty to incorporate contextual information, insufficient flexibility of the architecture.*

A recurrent multilayer perceptron is shown to be able to advantageously replace the classical Markov models and to circumvent their drawbacks.

However, dynamic programming remains the basic tool for connected speech recognition.

Connectionist methods raise strong expectations in recognition and research must be devoted in parallel to this new field as well as to the more classical Markov models. Moreover, speech recognition constitutes a large scale application which will lead to a better understanding of the connectionist mechanisms themselves.

Keywords—Speech recognition, connectionism, multilayer perceptrons, Markov models, Boltzmann machine, neural networks, discrimination, classifiers, dynamic programming.

1. INTRODUCTION

Speech recognition is only a part of speech understanding which is a much more ambitious task: while recognition only provides a sequence of words satisfying a given syntax, understanding requires the transformation of such a sequence into a semantic representation which can enrich a data base or which can immediately activate a process such as robot, textprocessing, baggage sorting, air flight reservations, data base query ... Achieving speech understanding implies the combination of several knowledge sources as phonetics, lexical properties, syntax, semantics and even pragmatics. The latter one restricts the search for a recognized sentence to the domain of interest and is crucial for the disambiguation of the anaphoras. The quality of the phonetic or lexical recognition could be significantly improved by the cooperation of all levels which restricts the search domain. The implementation of such interactions is particularly difficult and research goes on to solve the problem by using Artificial Intelligence methods such as blackboards and knowledge sources as in the early

HEARSAY system. Most of the high level techniques developed for natural language processing rely on error-free low level inputs. As a consequence, the understanding is often split in two steps: recognition and natural language processing. The rôle of recognition which extends from the acoustic data acquisition to the syntactical level is to provide the most reliable input as possible to the higher levels.

2. MODELS OF SPEECH UNITS

The simplest method occurring in the mind of an engineer for solving the problem of speech recognition is to identify typical features of the signals to be classified and then to try and detect these features in a recognition phase. Each word is then identified by its acoustic signature. The early methods of speech recognition were based on such a pattern recognition approach but formed the weak link of ambitious projects such as HEARSAY. The first step requires indeed expert knowledge in phonetics. However, the utterance of a speech signal differs so strongly from one speaker to another and its variability even for the same speaker is so important that it is a tremendous work to specify the characteristic features of a speech unit such as a word, a syllable and a fortiori a phoneme, particularly when embedded in a discourse. Moreover, industrial production tends to avoid the need of human expertise each time a different application must be designed.

The representation of the speech signal is obtained by a preprocessing that achieves some data compression. Spectral or cepstral analysis are applied to sliding windows of the time signal. The extracted parameters form an *acoustic vector* associated with each time slot. Linear predictive coding is also commonly used but requires a special metric, known as the Itakura distance, contrary to the other representations which are compatible with an Euclidean metric.

A supplementary data compression may be obtained by vector quantization. Each acoustic vector is replaced by the nearest prototype. Prototypes are computed by a clustering algorithm such as K-MEANS on a large acoustic vector data base.

2.1 Template Models

In small vocabulary applications, the word could be selected as the basic unit for speech recognition. Typical records of words, named templates, are memorized and uttered signals are compared with them. Variability of speech production is taken into account by storing several templates per word uttered by the same speaker (for speaker dependent recognition) or by different speakers (for the multispeaker mode).

For large vocabulary applications, smaller sub-units must be used in order to limit their number. The smallest sub-unit set able to cover an unlimited vocabulary is the phoneme set but phonemes are highly coarticulated so that they cannot be easily and accurately excised out of a continuous text.

A significant improvement for the recognition was the *Dynamic Time Warping* (DTW) (Vintsyuk, 1968; Sakoe & Chiba, 1978). This algorithm based on the dynamic programming principles achieves the optimal matching between an acoustic vector sequence and the stored models. The nonlinear distortions of the time axis between a reference template and an actual utterance are smoothed out.

To cope with the high variability of the speech signals, reference templates have been replaced by statistical models.

2.2 Hidden Markov Models

Stochastic models of speech units (Jelinek, 1976) able to be automatically trained on a speech data base have been developed. A *Hidden Markov Model (HMM)* is constituted by a set of states connected together by transition edges. A probability density (named local probability) to produce an acoustic vector is associated with each transition. The probability to produce an acoustic vector sequence with a model, often known as score, is the sum of the cumulated probabilities corresponding to each path in this model from a given initial state to a final one (Bahl et al., 1983; Bourlard et al., 1985). A usual simplification is to reduce this sum to its dominant term. Under this assumption, the score of the best path can be easily computed by using a dynamic programming algorithm similar to the *DTW* and known as the *Viterbi* algorithm. The *Viterbi* algorithm achieves the optimal matching between an acoustic vector sequence and the states of the model. It also plays an essential role in connected speech recognition: indeed, the determination of the best path with possible discontinuities at the boundaries of the word models simultaneously provides the segmentation and the labeling of words (Bridle et al., 1982; Sakoe, 1978, Myers & Rabiner, 1981; Ney, 1984).

The parameters of the models are estimated by training. Known sentences are matched against the corresponding chain of models and the parameters are tuned to maximize the global probability over the complete training data set. Such an "embedded training" avoids the tedious a priori phonetic segmentation and the coarticulation effects are incorporated by the stochastic nature of the representation (Bourlard et al., 1985). *HMM*'s, coupled with the *Viterbi* algorithm, are particularly well suited for modeling the sequential nature of speech.

Hidden Markov models led to a breakthrough in the recognition score improvement. However, during the last two years, any supplementary progress is obtained only at the price of an excessively increased complexity (Aubert et al., 1988). The main difficulties lie in the lack of discrimination between models of different speech units and in the ignorance of the contextual speech information.

To make the models context-sensitive, two solutions are possible: either the data structure can be modified by replacing each acoustic vector by a larger one containing also its left and right neighbours (Furui, 1986; Marcus, 1981) or a new probability density can be defined which depends on the neighbours (Wellekens, 1987). In both cases, the number of parameters of the densities becomes so large that a meaningful training would require an unrealistic amount of data. Moreover, in the second case, any modification of the contextual window width would require an explicit reformulation.

The most commonly used learning criterion, known as the *Maximum Likelihood Estimation* (Bahl et al., 1983; Bourlard et al., 1985) maximizes the global probability of the model associated with the training set (Baum-Welch algorithm) but does not simultaneously minimize the probabilities associated with all other models and which could yield false recognitions. The score of the correct path differs very slightly from that of the second candidate: no discrimination occurs. Alternative criteria have been proposed as the *Maximum Mutual Information* (Bahl et al., 1986) but they are difficult to work out and often at the price of simplifying hypotheses (Brown, 1987). Another approach is a modification of the *HMM*'s

by the definition of discriminant local probabilities (Bourlard & Wellekens, 1988b) which allow Bayes classification of the acoustic vectors (Fukunaga, 1972). Such discriminant *HMM*'s are closely related to connectionist networks as *Multilayer Perceptrons (MLP)* as will be shown in section 4.

3. CONNECTIONIST REPRESENTATIONS

These drawbacks are easily circumvented by using connectionist machines such as *Boltzmann machines* or *Multilayer Perceptrons*. Just like the *HMM*'s, connectionist machines are trained on examples. The explicit formulation of the characteristics of the processed information is not required but its main features are automatically captured and stored in a distributed and hidden way in the parameters of the machines. This property is essential when the underlying structure of the signals is not available as for example in speech recognition or in vision: indeed, it is impossible to explicit the rules used by a listener or an observer to come to a decision on the signal.

Another advantage of these machines is their *generalization* property which is closely related to their non-linear characteristics. It plays an important role as it is impossible in a training data base to include all possible speech events. Moreover, the hidden units of these machines allow the construction of an internal representation of the speech. This self-organization is useful as important hidden features of the internal structure of speech can be so captured during the training phase.

The particular architecture of the connectionist networks make them suitable for parallel computation so that the use of dedicated hardware will be a step towards real time speech recognition.

3.1 Boltzmann machines

Boltzmann machines (Hinton et al., 1984) form a particular family of connectionist machines which provide a solution by running to an equilibrium point which corresponds to the minimum of an energy function. In the context of speech applications, "solution" means recognition of a phoneme, a word or a sentence.

Boltzmann machines are constituted by a very large number of elementary computational units with binary states. If some units are specialized to constitute an input and an output field (visible units), they can be used as recognizers: the input units are clamped to represent a stimulus the identification of which is displayed on the output field. The remaining units are the hidden units. *Boltzmann machines* are said to be "stochastic" since the output of a unit is "on" or "off" according to a probability depending on the weighted sum of the outputs of the other units and of a control parameter known as the "temperature". The energy minimization is performed by a *simulated annealing* algorithm (Kirkpatrick et al., 1982; Aarts & Korst, 1988). While in a simple descent algorithm, a unit will change its state only if it results in a decrease of the energy, a temporary increase of the energy is statistically accepted in a simulated annealing process: the higher the temperature, the more probable are the upward energy jumps. If the temperature is slowly decreased, a global minimum of the energy is eventually reached.

The weights, known as *synaptic weights*, are bidirectional and symmetrical; their values are determined by the training phase. A Kullback-Leibler criterion is minimized which tends to equalize, for a given input field, the probabilities to observe an equilibrium point with an output field clamped on desired states on the one hand and with free outputs on the other hand. During this training, an internal representation of the input/output mapping is automatically built on the hidden units.

A huge amount of CPU time is required as well for the training as for the dynamic energy minimization. That may be acceptable for the training phase since it is performed off-line but a real-time convergence to the solution is compulsory in the recognition phase.

Although *Boltzmann machines* exhibit attractive features for simulating human recognition processes, the required optimizations constitute a major hindrance to practical implementations.

3.2 Multilayer Perceptrons

The architecture of *multilayer perceptrons (MLP)* is much simpler (Rumelhart et al., 1986). Units are organized into layers; the last one is the output field and the other layers are *hidden* for the outside world. The input of a unit is the weighted sum of the output activations of the units of the preceding layer. The synaptic weights are thus unidirectional and for that reason, no iterative process is required in the recognition phase. Thus real time applications can be considered in particular by using a hardware implementation taking full advantage of the parallel structure. This property justifies the fact that *MLP's* presently stand in the focus of the connectionist research in speech recognition.

The training of *MLP's* is based on the gradient minimization of a criterion: the one most commonly used is the least mean squared error between the desired and the actually observed outputs. This minimization is cleverly programmed as an *error back propagation* (Rumelhart et al., 1986). A forward computation provides the activations of the output layer and the differences with the desired outputs are computed. Then in a backward computational wave, corrections are applied to the weights. However, just like for the *Boltzmann machine* (although it is based on a different algorithm), the training is very time consuming since it requires a very large number of iteration cycles during which a large data base is forwarded in the input field with the corresponding desired activations on the output units.

Multilayer perceptrons have been used as discriminant nonlinear classifiers (Lippmann, 1987). They are able to classify data having nonlinear separating curves or even forming non connected sets in the feature space.

A particular application in speech recognition is the representation of the whole set of phonemes of a lexicon by a *single MLP* with one hidden layer (Bourlard & Wellekens, 1987, 1988a). Since this machine is globally trained on the whole data set contrary to what happens for *HMM* models which are separately trained class by class, it can provide simultaneous information on the probability that the input is a member of a class and on the probability that it does not belong to the others. In other words, discrimination can be easily enforced.

The input field may include contextual information in the same spirit as the famous NETtalk network (Sejnowski & Rosenberg, 1987) which learns and memorizes in its

synaptic weights the English language pronunciation rules which are indeed fundamentally context-dependent.

In this particular configuration, each letter of a written word is supplied to the input field surrounded by several left and right neighbouring letters (or silences). The associated output is the phoneme that must be associated in a normal English pronunciation. The input field has thus the structure of a shift register. During the training, the synaptic weights are iteratively updated. After a large number of iteration cycles, NETtalk is able to deliver the correct phonetic labeling of any written word. In an impressive demonstration by Sejnowski, this machine activates a speech synthesizer showing that it has thus learned to read aloud.

In a speech recognition application, the letters are replaced by acoustic vectors and still more variability appears in the input/output mapping: in our case between the acoustic input and the phonemic labeling. However, it must be noticed that, even with quantized acoustic vector inputs, the number of possible different stimuli increases exponentially with the width of the contextual window: it is thus impossible to estimate the local probabilities by counting as in a *HMM* due to the excessive size of the data set that would be required for a meaningful training. The generalization property plays here a key role: indeed, during the training, the synaptic weights have been tuned in order to generate at the outputs of the hidden units linear combinations of cross-products between the inputs. Activations at the output layer are obtained by recombining these cross-products (Bourlard & Wellekens, 1987, 1988a). Thus, a stimulus never observed in the input field, will preferably activate an output corresponding to an input that possesses in common as many cross-products as possible. On the contrary, with a discrete *HMM*, the activation would be zero or artificially fixed at a lower bound.

4. RECURRENT AND CONTEXT-SENSITIVE MLP

But classification is a static application. For speech recognition, a weakness of the connectionist structures is the difficulty to cope with the sequential nature of the speech signal or signal dynamics. Addition of *feedback loops* provides a solution to this shortcoming.

For instance, Prager, Harrison and Fallside (Prager et al., 1986) have described a *Boltzmann machine* supplied with a feedback loop that, at the thermal equilibrium, brings the decisions taken at the preceding instant in the input field. Unfortunately, this attractive principle suffers from the drawbacks of the *Boltzmann machines*, i.e. excessive CPU requirements even in the recognition phase.

The same principle has been applied to *MLP's*. Several applications for isolated word recognition have been proposed in the literature but most of them are restricted to low size lexicons and thus still remain at the level of research experiments.

In this section, a recurrent and context-sensitive *MLP* is described that can be used as local probability generator (Bourlard & Wellekens, 1988b).

The input signal is a quantized acoustic vector sequence. Part of the inputs of the *MLP* represents a current prototype vector. The output field contains a number of units equal to the number of phonemes in the lexicon and it is fed back as such or after recoding to form the second part of the input field.

Such a recurrent architecture allows the simulation of some useful properties of the discriminant hidden Markov models. In the discriminant *HMM's*, the local probability is defined as the probability to be on state (say q_k) under the condition that the current observed

vector is y_i and that the precedingly visited state was q_m . The estimators of these local probabilities are precisely the *optimal* output activations of the recurrent *MLP* (Bourlard & Wellekens, 1988b). The optimal activations do not depend on the internal topology of the machine but are immediately related to the data base statistics. They are actually the observed outputs of the *MLP* provided the machine contains enough parameters. This condition is generally not satisfied by nonlinear classifiers without hidden layers (Bourlard & Wellekens, 1986).

The input field can also be extended to the neighbouring vectors to take the context into account as explained in the preceding section.

The training of this recurrent machine is totally supervised. Indeed, the data base is constituted by prototype vectors and associated phonemes so that even the correct feedback can be supplied to the input.

As such, this machine performs phonemic labeling. To solve more realistic tasks as large lexicon connected speech recognition, it is useful to resort to the Dynamic Time Warping procedure.

5. CONNECTED SPEECH RECOGNITION

Since the outputs of the recurrent context sensitive *MLP* are equivalent to the local probabilities of a discriminant hidden Markov model, classical recognition packages based on the Viterbi algorithm accept their activations as local probabilities. The *multilayer perceptron* plays thus here the role of a local contribution generator able to force discrimination between phonemes and to include wide contextual acoustic information.

A large tableau is built (Bourlard et al., 1985). Its column index represents the time. The rows are associated with states and are ordered to form state sequences that correspond to words. A one level *Viterbi* algorithm (Bridle et al., 1982; Ney, 1984) is used to find in the tableau the best path between the beginning of a word model at the first instant to the end of a word model at the last instant. Constraints are applied that forbid path discontinuities inside a word. This path provides the word segmentation and the word labeling of the utterance.

Transition probabilities as in the *HMM* are no longer required in the *DTW* tableau itself since they have been learned by the *MLP* which provides dynamical and context-sensitive contributions.

6. CONCLUSIONS

Speech recognition still requires a huge amount of research before reaching a high level application field such as natural language recognition with large vocabulary, without speaker constraints and in a noisy environment. Meanwhile, less ambitious applications, based on classical techniques as *HMM*'s have been commercialized.

Connectionist methods must be considered as one alternative technique towards achievement of more difficult tasks. Results obtained up to now do not transcend the level of preliminary studies. Thanks to the flexibility of their architecture which allows an easy incorporation of contextual sensitivity and of recursion, the use of discriminant criteria and the generalization ability, *MLP*'s constitute extremely promising tools. Nevertheless, research

is going on to improve *HMM* performances, viz. discrimination, state occupation duration, model topology and nature of the local probability densities.

Connectionist models have not yet proved any definitive superiority versus classical models but become everyday more challenging and raise expectations for increased speech recognition scores. Conversely, their use in recognition allows a better understanding of the connectionist mechanisms by applying their principles on a real industrial task.

REFERENCES

- Aarts E.H.L., & Korst J.H.M. (1988). *Simulated Annealing and Boltzmann Machines*, Wiley, (to appear)
- Aubert X., Boulard H., Kamp Y., & Wellekens C.J. (1988). Improved Hidden Markov Models for Speech Recognition. *Philips Journal of Research*, vol.43, nos 3/4, 224-245.
- Bahl L.R., Jelinek F., & Mercer R.L. (1983). A Maximum Likelihood Approach to Continuous Speech Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no 2, 179-190.
- Bahl L.R., Brown P.F., de Souza P.V., & Mercer R.L. (1986). Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition, *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 49-52, Tokyo.
- Boulard H., Kamp Y., Ney H., & Wellekens C.J. (1985). Speaker-Dependent Connected Speech Recognition via Dynamic Programming and Statistical Methods, in M.R. Schroeder (Ed.), *Speech and Speaker Recognition*, Zurich: Karger.
- Boulard H., & Wellekens C.J. (1986). Discriminant Functions for Connected Speech Recognition, in I.T. Young, J. Biemond, R.P.W. Duin & J.J. Gerbrands, (Eds.), *Proceedings of the European Signal Processing Conference*, 507-510, The Hague (The Netherlands).
- Boulard H., & Wellekens C.J. (1987). Multilayer Perceptrons and Automatic Speech Recognition, *Proceedings of the First International Conference on Neural Networks*, IV, 407-416, San Diego, CA.
- Boulard H., & Wellekens C.J. (1988a). Speech Pattern Discrimination and Multilayer Perceptrons, *Computer, Speech and Language*, vol.3, 1-19.
- Boulard H., & Wellekens C.J. (1988b). Links between Markov Models and Multilayer Perceptrons, *Proceedings of the Neural Information Processing Systems Conference 1988, Denver, CO*, to appear.
- Bridle J.S., Brown M.D., & Chamberlain R.M. (1982). An algorithm for connected word recognition, *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 899-902, Paris.
- Brown P. (1987). *The Acoustic-Modeling Problem in Automatic Speech Recognition*, Ph.D. thesis, Computer Science Department, Carnegie-Mellon University.
- Furui S. (1986). Speaker Independent Isolated Word Recognizer Using Dynamic Features of Speech Spectrum, *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-34, 52-59.

- Fukunaga, K. (1972). *Introduction to Statistical Pattern Recognition*, Academic Press, NY.
- Hinton G.E., Sejnowski T.J., & Ackley D.H. (1984). Boltzmann Machines: Constraint Satisfaction Networks that Learn, *Technical Report CMU-CS-84-119*, Carnegie Mellon University.
- Jelinek F. (1976). Continuous Recognition by Statistical Methods, *Proceedings IEEE*, vol.64, no.4, 532-555.
- Kirkpatrick S., Gelatt C.D. jr., & Vecchi M.P. (1982). Optimization by Simulated Annealing, *IBM Research Report RC 9355*.
- Lippmann R.P. (1987). An Introduction to Computing with Neural Nets, *IEEE Magazine on Acoustics, Speech and Signal Processing*, vol. 4, 4-22.
- Marcus S.M. (1981). ERIS-context sensitive coding in speech perception, *Journal of Phonetics*, 9, 197-220.
- Myers C.S., & Rabiner L.R. (1981). Connected Digit Recognition Using a Level-Building DTW Algorithm, *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-29, 351-363.
- Ney H. (1984). The use of a one-stage dynamic programming algorithm for connected word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-32, 263-271.
- Prager R.W., Harrison T.D., & Fallside F. (1986). Boltzmann machines for speech recognition, *Computer, Speech and Language*, vol.1, 3-27.
- Rumelhart D.E., Hinton G.E., & Williams R.J. (1986). Learning Internal Representations by Error Propagation, in D.E.Rumelhart & J.L. McClelland, (Eds.), *Parallel Distributed Processing. Exploration in the Microstructure of Cognition, vol.1: Foundations*, MIT Press.
- Sakoe H., & Chiba S. (1978). Dynamic programming algorithm optimization for spoken word recognition, *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-26, 43-49.
- Sakoe H. (1979). Two-Level DP Matching. A Dynamic Programming-Based Pattern Matching Algorithm for Connected Word Recognition, *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-27, 588-595.
- Sejnowski T.J., & Rosenberg C.R. (1987). Parallel Networks that Learn to Pronounce English Text, *Complex Systems*, vol.1, 145-168.
- Vintsyuk K. (1968). Speech Discrimination by Dynamic Programming, *Kibernetika (Cybernetics)*, 4, 81-88.
- Wellekens C.J. (1987). Explicit Time Correlation in Hidden Markov Models for Speech Recognition, *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 10.7.1-10.7.3, Dallas, TX.

Hidden Markov Models and their application in speech recognition ¹

Paul van Alphen

Institute of Phonetic Sciences, University of Amsterdam
Herengracht 338, 1016 CG Amsterdam, The Netherlands

introduction

In current speech recognition research the use of Hidden Markov Models becomes more and more popular and successful. However, for those who know little about (Hidden) Markov Models, an accessible introduction is hard to find. What is a Markov Model? How can Markov Models be used as recognizers? Why is a Hidden Markov Model hidden? In this contribution an attempt will be made to answer these questions step by step.

Starting from Markov processes that are described by Markov chains, we will first show what the Markov Models look like. Next the way Markov Models can be used for recognition is demonstrated (the Viterbi algorithm), and the algorithm for training these models is presented (the Baum Welch reestimation algorithm). The intention of this training (or learning) algorithm is to extract representative characteristics from the input process; these characteristics are then modelled in the parameters of the Markov Models. Next we will extend the Markov Models theory to Hidden Markov Models. Hidden Markov Models inherit all properties of the Markov Models, but the states in the model are no longer associated with one observation only, but with a probability distribution over all possible observations. A new set of parameters is introduced, describing this distribution, and changes in recognition and training algorithms are explained. Changes can be quite drastic, because the hidden states raise the complexity of the models and algorithms. One of the reasons why Hidden Markov Models are introduced, is that the system becomes too large for normal Markov Models, if structures to be recognized become more and more complex. Multiple observation sequences turn out to be very useful to overcome this problem. Next we will adapt the models to be capable to deal with more than one feature describing the objects (multiple symbol distributions). This means that, for instance apart from spectra, one can also specify the energy or LPC coefficients. After this mathematical foundation, we will proceed with the application of Hidden Markov Models to speech. In order to increase the readability, no references are given; instead we have included a list of recommended literature.

¹ This reader-contribution is a summary of a paper in IFA Proceedings 13 (1989), with D. van Bergem as co-author.

Markov Models

A Markov chain is a stochastic process, of which the outcome is a sequence of T observations $O(1), O(2) \dots O(T)$ that satisfy the following assumptions:

1. Each observation belongs to a finite set of N states $\{ S_1, S_2, \dots, S_N \}$. If the outcome at time t is S_i , then we say that the system is in state S_i at that time.
2. Any observation depends only upon the immediately preceding observation and not upon any other previous observation. For each pair of states $\{ S_i, S_j \}$ the number a_{ij} denotes the probability that S_j occurs immediately after S_i occurs.

In figure 1 a simple example is given of a Markov chain. The outcome of this Markov chain might be the following observation sequence: "a", "a", "b", "a", "b", "b", "a", ...

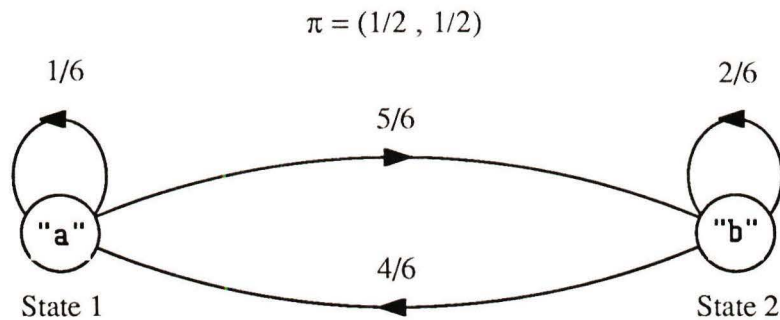


Figure 1. Example of a Markov chain.

The numbers a_{ij} , called the state **transition probabilities**, can be arranged in a matrix

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & \dots & a_{1N} \\ a_{21} & a_{22} & a_{23} & \dots & \dots & a_{2N} \\ a_{31} & a_{32} & a_{33} & \dots & \dots & a_{3N} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & a_{N3} & \dots & \dots & a_{NN} \end{pmatrix}$$

called the transition matrix, where N is the total number of states.

The **initial state probability distribution**, i.e. the probability distribution when the process begins ($t=1$) can be arranged in a vector

$$\Pi = (\pi_1, \pi_2, \pi_3, \dots, \pi_N)$$

One can calculate the probabilities that the chain has produced the observations using an algorithm called the *forward procedure*. For this procedure we define the probability $\alpha_i(t)$ as the probability of being in state S_i at time t . Thus the vector $\alpha(t)$ denotes the probability distribution at time t .

$$\alpha_j(t) = \sum_{i=1}^N \alpha_i(t-1) \cdot a_{ij}$$

If one is interested in the single best path with the highest probability, the *Viterbi algorithm* has to be used, which finds this path based on a Dynamic Programming method. We define $\delta_j(t)$ as the best score (highest probability) at time t along a single path that ends in state S_j . This score can be recursively calculated with the formula:

$$\delta_j(t) = \max_{1 \leq i \leq N} (\delta_i(t-1) \cdot a_{ij})$$

The Viterbi algorithm is similar to the forward procedure. However, a maximization over previous states is used instead of the summing procedure used in the forward calculation. If we store the *index* i of the $\delta_i(t-1)$ that maximizes $\delta_j(t)$ in a vector $\psi_j(t)$, the optimal path can be found by backtracking. This vector $\psi_j(t)$ is simply a pointer to the 'best' preceding state S_i .

$$\psi_j(t) = \operatorname{argmax}_{1 \leq i \leq N} (\delta_i(t-1) \cdot a_{ij})$$

Hidden Markov Models

Suppose we want to model a stochastic process that generates 100 colours (in modelling speech we will actually use 256 different 'symbols', which can be considered to be acoustical events). With a 'normal' Markov Model we would need 100 states, one for each colour. To obtain such a model we would have to train 10000 (100^2) transition probabilities, which would require an enormous amount of training data. The number of states can be drastically reduced with the use of **Hidden Markov Models** (HMM's), although at the cost of a greater arithmetic complexity as we will see.

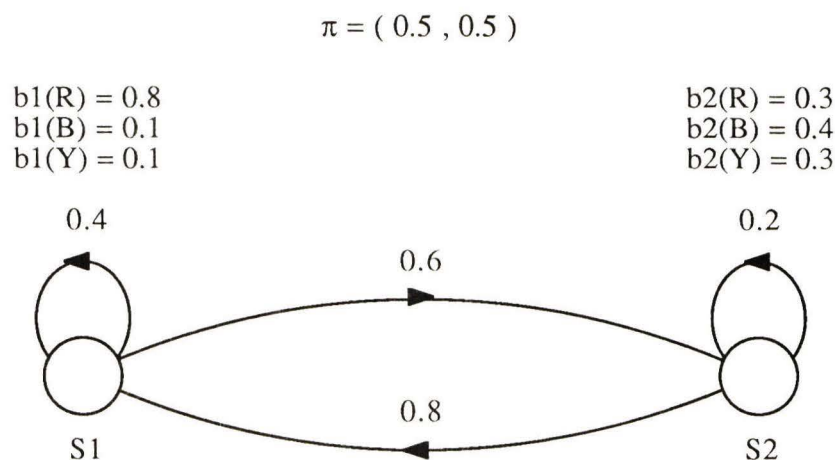


Figure 2. A simple Hidden Markov Model with two states and three observation symbols (colours).

Consider the Markov Model of figure 2. This model has two states S_1 and S_2 . However, the states are not associated with one coloured ball any more, but with an urn that is filled with an infinite number of balls. For reasons of clarity we will use three colours only, although it is possible to have balls with a large number of different colours. There are red balls, blue balls and yellow balls, each with a certain probability of occurrence, being different for each state (see figure 2). This **observation symbol**

probability distribution can also conveniently be arranged in a $N \times M$ matrix, in which N denotes the number of states and M the number of observation symbols (colours in our example):

$$B = \begin{pmatrix} b_{11} & b_{12} & b_{13} & \dots & \dots & b_{1M} \\ b_{21} & b_{22} & b_{23} & \dots & \dots & b_{2M} \\ b_{31} & b_{32} & b_{33} & \dots & \dots & b_{3M} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ b_{N1} & b_{N2} & b_{N3} & \dots & \dots & b_{NM} \end{pmatrix}$$

called the observation matrix.

The recursion formulas for $\alpha_j(t)$, $\delta_j(t)$ and $\psi_j(t)$ become:

$$\alpha_j(t) = \sum_{i=1}^N (\alpha_i(t-1) \cdot a_{ij}) \cdot b_j(O(t))$$

$$\delta_j(t) = \max_{1 \leq i \leq N} (\delta_i(t-1) \cdot a_{ij}) \cdot b_j(O(t))$$

$$\psi_j(t) = \operatorname{argmax}_{1 \leq i \leq N} (\delta_i(t-1) \cdot a_{ij})$$

How do we train the Hidden Markov Models? The following estimates are proposed for a_{ij} , $b_j(m)$ and π_i (Baum Welch):

$$\tilde{a}_{ij} = \frac{\left(\begin{array}{l} \text{Probability of being in state } S_i \\ \text{and making a transition from state } S_i \text{ to state } S_j \end{array} \right)}{\text{Probability of being in state } S_i}$$

$$\tilde{b}_j(m) = \frac{\text{Probability of being in state } S_j \text{ and observing symbol } m}{\text{Probability of being in state } S_j}$$

$$\tilde{\pi}_i = \text{Probability of being in state } S_i \text{ at } t = 1$$

Hidden Markov Models for speech

Of the many topics that remain when one wants to apply HMM's to speech, the following are the most prominent:

- **Isolated word recognition** versus **continuous speech recognition**.
- **Discrete observations** (via a codebook) versus **continuous observations**. Continuous observations (e.g. a spectral vector) are described in terms of Gaussian densities, with mean and variance as parameters.

- The choice of the **speech-unit** to be modelled and the HMM **topology** used.

Figure 3 shows the HMM, we used in our Rexy system to model phone-like units.

The continuous speech recognition system Rexy, which is developed in our institute in Amsterdam, is trained and tested for one (male) speaker, and has a vocabulary of 240 words. At the colloquium some of the recognition results will be presented.

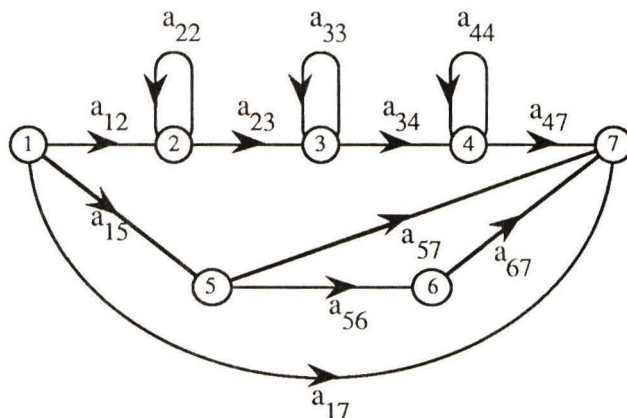


Figure 3. Transition probabilities of the phone model.

RECOMMENDED LITERATURE

- Gupta, V.N., Lennig, M. & Mermelstein, P. (1987). "Integration of acoustic information in a large vocabulary word recognizer", Proc. IEEE-ICASSP '87, Dallas, 697-700.
- Lee, C.H., Juang, B.H., Soong, F.K. & Rabiner, L. R. (1989). "Word recognition using whole word and subword models", Proc. IEEE ICASSP 89, 683-686.
- Lee, C.H., Rabiner, L. R., Pieraccini, R. & Wilpon, J.G. (1990). "Acoustic modeling for large vocabulary speech recognition", Computer Speech and Language, Vol. 4(2), 127-165.
- Lee, K.F. (1989). Automatic Speech Recognition: The Development of the SPHINX System, Kluwer Academic publishers, Boston, 207 pages.
- Lee, K.F. (1989). "Hidden Markov Models: past, present, and future", Proc. Eurospeech '89, book 1, 148-155.
- Levinson, S.E. (1985). "Structural methods in automatic speech recognition", Proceedings of the IEEE, Vol. 73(11), 1625-1650.
- Levinson, S.E. (1986). "Continuously variable duration Hidden Markov Models for automatic speech recognition", Computer Speech and Language, Vol. 1(1), 29-45.
- Levinson, S.E. (1987). "Continuous speech recognition by means of acoustic/phonetic classification obtained from a Hidden Markov Model", Proc. IEEE-ICASSP '87, Dallas, 93-96.
- Lloyd, E. (1980). "Processes in discrete time: Markov chains", in Ledermann, W. (editor), Handbook of Applicable Mathematics, Probability, Vol. II, John Wiley & Sons Ltd., Chichester, 373-401.
- Lipschutz, S. (1966). "Markov chains", in Schaum's Outline Series: Theory and problems of finite mathematics, Schaum Publishing co., New York, 233-256.
- Mari, J.F. & Roucos, S. (1985). "Speaker independent connected digit recognition using Hidden Markov Models", Proc. SPEECH TECH '85, New York, 127-132.
- Paul, D.B. (1990). "Speech recognition using Hidden Markov Models", The Lincoln Laboratory Journal, Vol. 3, Number 1, 41-62.
- Picone, J. (1990). "Continuous speech recognition using Hidden Markov Models", IEEE ASSP Magazine, Vol. 7(3), 26-41.

- Rabiner, L.R., Levinson, S.E. & Sondhi, M.M. (1983). "On the application of Vector Quantization and Hidden Markov Models to speaker-independent, isolated word recognition", *The Bell System Technical Journal* 62(4), 1075-1105.
- Rabiner, L.R. & Juang, B.H. (1986). "An introduction to Hidden Markov Models", *IEEE ASSP Magazine* 3(1), 4-16.
- Rabiner, L.R. (1988). "Mathematical foundations of Hidden Markov Models", *NATO ASI Series, Vol. F46, Recent Advances in Speech Understanding and Dialog Systems*, Springer-Verlag, Berlin Heidelberg, 183-205.
- Rabiner, L. R., Wilpon, J.G. & Soong, F.K. (1989). "High performance connected digit recognition using Hidden Markov Models", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-37(8), 1214-1225.
- Silverman, H.F. & Morgan, D.P. (1990). "The application of Dynamic Programming to connected speech recognition", *IEEE ASSP Magazine*, Vol. 7(3), 7-25.
- van Alphen, P. & van Bergem, D.R. (1989). "Markov Models and their application in speech recognition", *Proc. Institute of Phonetic Sciences Amsterdam 1989*, Vol. 13, 1-27.
- van Alphen, P. (1990). *Speech recognition by computer*, PhD thesis, Institute of Phonetic Sciences, Amsterdam, to be published in December 1990.

Artificial neural nets and their applications in speech recognition

J. P. Martens

Research Associate of the National Fund for Scientific Research

University of Ghent

Laboratory for Electronics and Metrology

St.-Pietersnieuwstraat 41, B-9000 Gent (Belgium)

1 Introduction

In this paper we describe some applications of artificial neural networks (NN's) in the field of speech processing. It has been suggested to use NN's in speech synthesis (to learn pronunciation rules [12]), speech analysis (to perform voiced/unvoiced classification [1]) and speaker verification [2]. Nevertheless, the bulk of applications is found in the domain of automatic speech recognition.

Obviously, speech recognition involves a great deal of acoustic pattern classification, and neural networks have been shown to have excellent pattern discrimination capabilities. Therefore it was quite normal to try and use neural networks in the acoustic-phonetic decoding part of a speech recognizer. In this contribution we discuss the two types of neural networks that are most often used in this respect, namely, self-organizing feature maps (SOFM's) and multi-layer perceptrons (MLP's). We indicate how they can be integrated in the recognition process, and investigate their performance. Do NN-based recognizers perform better than recognizers based on more established statistical models such as Hidden Markov Models (HMM's)? Can NN's and HMM's be used together in one system?

2 Static pattern recognition

The classical pattern recognition paradigm is a static pattern recognition scheme. It assumes that the input pattern can be classified correctly without taking into account the context (the past and/or the future) in which it occurs. In a static pattern recognition task, the neural network observes an input pattern represented by a feature vector $\mathbf{X} = \{X_1 \dots X_N\}$, and it generates an output vector $\mathbf{O} = \{O_1 \dots O_M\}$ representing evidences for the classes $1..M$ to be distinguished (fig. 1). If the neural network is doing its job well, it generates one active (high) output for each input vector, and this active output corresponds to the class of the input pattern.

If the neural network incorporates feedback loops, it will take several time steps before the output becomes stable. Therefore, networks with feedback are hardly used in speech recognition (which is essentially a real-time process). The NN's are always feed-forward or recurrent neural networks.

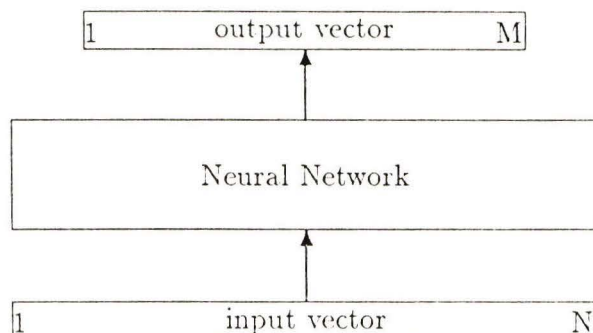


Figure 1: Pattern recognition with a neural network

Recurrent nets have memory (information about the past), feed-forward networks don't.

In the subsequent sections we describe two pattern classification paradigms that can easily be implemented with connectionist models. We outline the basics of the models, and we indicate how the models are integrated in the speech recognition process.

3 The Nearest Neighbour classifier

One of the most popular pattern classifiers is the nearest neighbour classifier. It operates according to the following two principles:

1. The set of all possible input patterns can be represented adequately by a finite set of labelled representatives, called templates.
2. The class of test input feature vector is assumed to be equal to that of the nearest template in the feature space.

The distance between the input pattern and the reference templates is computed according to a particular distance metric, and the reference templates are usually obtained by means of a clustering procedure, e.g., a K-means [13] or a C-means [4] clustering procedure. It has been shown by Kohonen [5] that the clustering can also be accomplished by means of a self-organizing

feature map. Such a map is represented on fig. 2. It consists of one layer

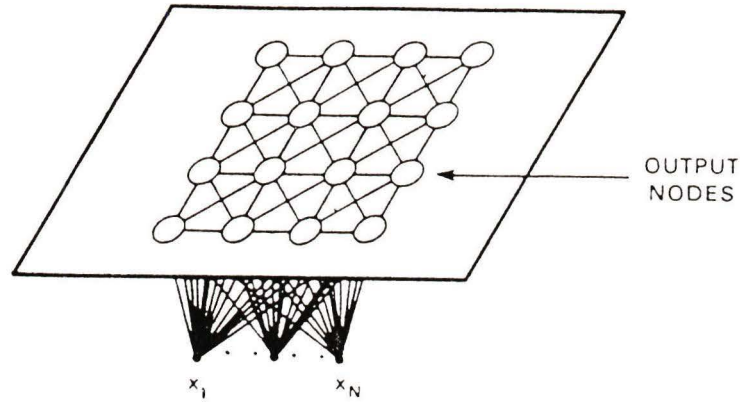


Figure 2: A self-organizing feature map

of nodes connected to the input features. Each node k is represented by a vector $\mathbf{W}_k = \{W_{k1}..W_{kN}\}$. The elements of this vector are the weights of the connections to the input vector. Furthermore, each node k computes some kind of a dissimilarity measure between its representation \mathbf{W}_k and the input vector \mathbf{X} . During the training phase there are also imaginary connections between adjacent nodes on the map. The meaning of these connections will become apparent from the description of the training algorithm, an algorithm which can be described in 4 steps:

1. Examine the set of available training input vectors \mathbf{X}_p , ($p = 1..P$), and select K of these vectors as the representations of the map nodes. Notice that K must be larger than the number of classes M to distinguish.
2. Compute for each training pattern \mathbf{X}_p , the dissimilarity scores

$$y_{pk} = \sum_{j=1}^N (W_{kj} - X_{pj})^2 \quad k = 1..K \quad (1)$$

and determine the node with the smallest score. This node (k_0) is called the winner.

3. Update the weights of all nodes k which are located in a certain neighbourhood of k_0 . If n represents the iteration step, the weights are updated according to

$$W_{kj}(n+1) = W_{kj}(n) + \alpha(n) \cdot [X_{pj} - W_{kj}(n)] \quad (2)$$

It is easy to verify that if the same input pattern were presented again, the nodes in the neighbourhood of k_0 would have a response

$$y_{pk}(n+1) = y_{pk}(n) \cdot [1 - \alpha(n)]^2 \quad (3)$$

Consequently, if $\alpha(n)$ is in the range from 0 to 1, the responses are reduced, while the responses of the cells far away from the winner are not affected. The node k_0 will thus attract input patterns which are similar to its current representation (=clustering).

4. The above procedure is repeated for all the training patterns, and the average outputs of the winner are accumulated to constitute an overall error measure. As long as no convergence is established the procedure is repeated from step 2.

The critical parameters are the learning rate $\alpha(n)$ and the definition of the neighbourhood of the winner. It is recommended to start with a large neighbourhood and to reduce it gradually to a single point near the end of the training procedure. For the learning rate, it is advised to ensure that the following relations are satisfied

$$\sum_{n=1}^{\infty} \alpha(n) = \infty \quad \sum_{n=1}^{\infty} \alpha^2(n) < \infty \quad (4)$$

The above training algorithm was shown to be a very powerful clustering algorithm which is entirely unsupervised (no labelled data required). It tries to minimize the total distortion which would arise from substituting the input vectors by the node representations.

Once the clustering has been accomplished, one can assign a meaning to each node k by estimating the conditional probabilities of the different classes under the condition that node k was found to be the winner. If node k is the winner, the hypothesis corresponding to the highest of these probabilities will be emitted.

Kohonen [6] has used the feature map to recognize Finnish and Japanese continuous speech utterances. In his recognizer, a spectral feature vector is generated every 10 ms, and the corresponding phonetic hypothesis is determined by the self organizing map. The obtained sequence of phonetic labels is then smoothed in order to remove isolated errors. Finally, successive labels who are identical are substituted by a single label representing one phonetic segment. These labels constitute a phonetic transcription of the utterance. Although Kohonen claims to obtain excellent result, there is a lot of scepticism in the rest of the speech recognition research community. In fact, the Kohonen map does something very comparable to a classical vector quantizer, and the applied smoothing algorithm does not offer the same temporal modelling capabilities as HMM's do. So far I didn't come along any study which demonstrates that Kohonen's strategy can compete with the standard HMM approach.

4 The maximum a posteriori classifier

In a Maximum A Posteriori (MAP) classifier is operating according to the following principles:

1. Estimate the a posteriori probabilities of the classes given the observation (the input pattern).
2. Select the class corresponding to the largest a posteriori probability

It has been demonstrated [9] that a Multi Layer Perceptron (MLP) can be trained in such a way that its outputs become estimates of these probabilities.

A MLP is a feed-forward network which is organized in successive layers. These layers are numbered from 0 (input layer) to $L + 1$ (output layer). The layers 1.. L are the so called hidden layers (fig. 3). The nodes of a particular layer different from the output layer are connected to all the nodes of the next layer. If we adopt the notation

$$\begin{aligned}
 N_l &= \text{the number of nodes on layer } l \\
 y_{li} &= \text{output of node } i \text{ of layer } l \quad (l = 1..L + 1) \\
 w_{lij} &= \text{weight of the connection from node } j \text{ on layer } l - 1 \\
 &\quad \text{to node } i \text{ on layer } l \\
 w_{li0} &= \text{bias input for node } i \text{ on layer } l
 \end{aligned}$$

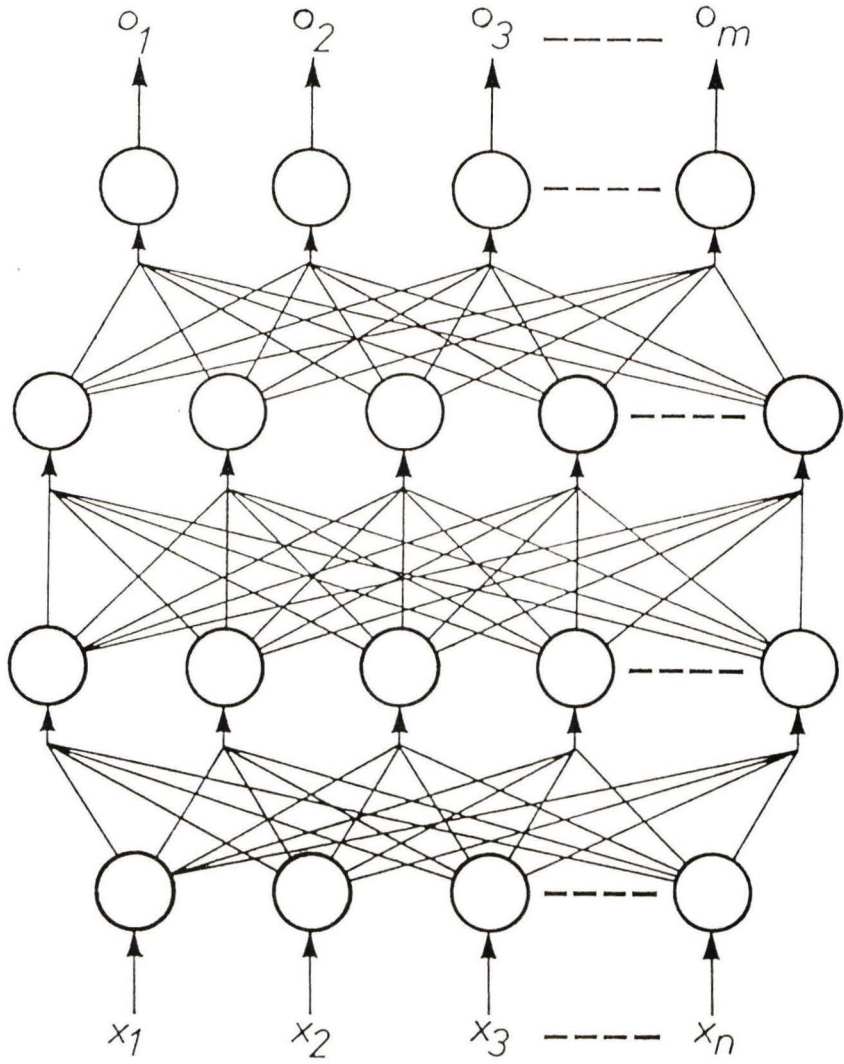


Figure 3: The Multi Layer Perceptron architecture

then the MLP is described by the following equations

$$y_{li} = F(a_{li}) \quad (5)$$

$$a_{li} = \sum_{j=1}^{N_{l-1}} w_{lij} \cdot y_{l-1,j} + w_{li0} \quad (6)$$

$$F(a) = \frac{1}{1 + e^{-a}} \quad (7)$$

The neural network nodes are stimulated by the activation a_{li} , and the activation function $F(a)$ approximates the threshold function

$$F_T(a) = 1 \quad \text{if } a \geq 0 \quad (8)$$

$$= 0 \quad \text{if } a < 0 \quad (9)$$

From the pattern classification viewpoint, the MLP computes a set of discriminant functions

$$o_m = g_m(\mathbf{X}) \quad m = 1..M \quad (10)$$

to make a decision about the class of the observation \mathbf{X} . Thanks to the non-linear activation function the discriminant functions can be highly nonlinear.

As the discriminant functions for a particular observation are entirely determined by the weights of the network connections, it is for the training algorithm to determine an optimum set of weights for the task at hand. However, it is not sure that there exists a set of discriminant functions (within the family of functions that can be computed by the MLP) which can correctly classify all the observations. It is interesting to notice for instance that

1. A MLP without hidden layers can only distinguish between classes that are linearly separable in the input space. As the activation function is monotonic, the discriminant functions computed by the MLP can be regarded as linear: $F(a) > F(b)$ is equivalent to $a > b$.
2. A MLP with hidden layers, but with a linear activation function would be equivalent to another MLP having no hidden layers.

From the paper of Lippmann [7] we have taken a figure (fig. 5) which clearly shows the kind of decision areas that can be constructed in the input space by MLP's with 0, 1 and 2 hidden layers.


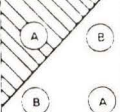



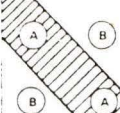
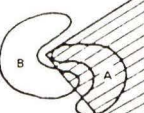


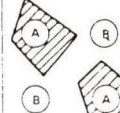
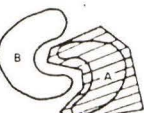

STRUCTURE	TYPES OF DECISION REGIONS	EXCLUSIVE OR PROBLEM	CLASSES WITH MESHED REGIONS	MOST GENERAL REGION SHAPES
SINGLE-LAYER 	HALF PLANE BOUNDED BY HYPERPLANE			
TWO LAYER 	TYPICALLY CONVEX			
THREE LAYER 	ARBITRARY (Complexity Limited By Number of Nodes)			

Figure 4: Decision regions that can be formed by MLP's

5 Training the MLP

Starting from random initial settings, the training algorithm has to update the weights until the MLP performs well enough on a representative set of labelled training examples. If the training examples are denoted \mathbf{X}_p ($p = 1..P$) and if their classes are known to be m_p , the expected output of the MLP given the input pattern \mathbf{X}_p is defined as

$$T_p = \{T_{p1}..T_{pM}\} \quad (11)$$

with M being the number of classes to distinguish, and

$$T_{pm} = 1 \quad \text{if } m = m_p \quad (12)$$

$$= 0 \quad \text{if } m \neq m_p \quad (13)$$

The expected output vector will be used as a teaching vector in the Error Back Propagation (EBP) training algorithm. This algorithm will try to minimize an accumulated error

$$E = \frac{1}{2} \sum_{p=1}^P \sum_{m=1}^M [O_{pm} - T_{pm}]^2 \quad (14)$$

Notice that any other function accumulating the discrepancies between the actual and the expected outputs would be equally acceptable. From a mathematical viewpoint, the EBP algorithm is a steepest descent (gradient) search

procedure in the weight space. The weights are updated according to

$$\Delta w = -\eta \frac{\partial E}{\partial w} \quad (15)$$

with η representing the learning rate. From the mathematics it follows that the gradients can be computed as follows:

1. For the weights to the output layer ($l = L + 1$) it emerges that

$$\frac{\partial E}{\partial w_{lij}} = \sum_{p=1}^P \delta_{pli} \cdot y_{p,l-1,j} \quad (16)$$

with

$$\delta_{pli} = (O_{pi} - T_{pi}) \cdot F'(a_{pli}) \quad (17)$$

and F' representing the derivative of F .

2. For the weights to a hidden layer l one obtains

$$\frac{\partial E}{\partial w_{lij}} = \sum_{p=1}^P \delta_{pli} \cdot y_{p,l-1,j} \quad (18)$$

with

$$\delta_{pli} = F'(a_{pli}) \cdot \sum_{k=1}^{N_{l+1}} \delta_{p,l+1,k} \cdot w_{l+1,ki} \quad (19)$$

It emerges that the gradients for the hidden layers can be expressed in terms of the δ 's which were computed for the higher layer. In other words, starting from the output layer the errors must be propagated back to the actual layer to compute the gradient.

As a steepest descent procedure usually converges very slowly, many techniques have been proposed to speed up the training process. We briefly mention some of them:

1. On-line training

Instead of presenting all the training patterns before updating the weights, one can update them every time a single example is presented (at least in the the beginning of the training).

2. Adaptive training

One can try to adapt the learning rate by regularly estimating the properties of the error function in the surroundings of the weight vector.

3. Smoothing

Especially in case of on-line training it is recommended to update the weights according to the following equation

$$\Delta w(n) = -\eta \frac{\partial E}{\partial w} + \beta \Delta w(n-1) \quad (20)$$

In the subsequent sections we describe how MLP's can be applied in isolated and continuous speech recognition.

6 Isolated word recognition

If we consider an entire word utterance as the input pattern presented to the MLP, the word recognition problem can be reduced to a static pattern recognition problem. The MLP will then perform a spatial coding of the temporal relations inside the utterance.

A problem may be that different utterances of different words are bound to have different lengths. Consequently, the utterances will be represented by a different number of feature vectors (one every 10 ms). Two simple strategies can be applied to convert the variable length input patterns into fixed length patterns:

1. The padding method
2. The trace segmentation method

The padding strategy appends silence vectors to the utterances until an expected maximum length is reached. The trace segmentation method reduces the length of the input patterns to an expected minimum length. It does so according to a non-linear time warping transformation which is based on the idea that successive feature vectors of an utterance u draw a trace in the feature space. The length of this trace is computed as

$$D_u = \sum_{n=1}^{N_u-1} d(\mathbf{X}_n, \mathbf{X}_{n+1}) \quad (21)$$

with N_u being the number of feature vectors in the utterance, and $d(\mathbf{X}, \mathbf{Y})$ representing the distance between two feature vectors \mathbf{X} and \mathbf{Y} . The trace segmentation method will select N feature vectors ($N < N_u$) which are more or less equidistantly spaced along the trace.

Some results recently published in the literature indicate that MLP's can attain the same level of performance as HMM's [11]. However, recent studies on vowel recognition under noisy circumstances [10] indicate that MLP's are more robust against noise.

7 Continuous speech recognition

In continuous speech recognition one is forced to recognize so called speech units such as phones or diphones. Unfortunately, there are no pauses between these units so that they can't be isolated from the surrounding units. In fact, it is generally accepted that the units cannot be recognized accurately without taking into account the properties of the surrounding units (the phonetic context). Obviously, continuous speech recognition cannot be reduced to a simple static pattern recognition scheme. Nevertheless, parts of the problem may be solved by static pattern classifiers, and MLP's may be the obvious tool to design them.

Some researchers [3] have proposed the following two stage phone recognition scheme:

1. Use a MLP to generate an output vector (representing phonetic evidences) for each feature vector that comes in
2. Supply these vectors to a dynamic time warping (DTW) process which is responsible for modelling the dynamics of the speech signal.

Speaker dependent tests [3] have demonstrated that the MLP + DTW combination clearly outperforms the standard HMM approach (using context independent phone models that is).

More recently, people have tried to integrate MLP's directly into the HMM paradigm. In that case, the MLP's are used to estimate the emission probabilities of the Markov states [9]. The training of the MLP's must then be embedded in the HMM training, which is not trivial but feasible.

In an accompanying paper [8], we discuss a hierarchical approach in which a first MLP is responsible for generating broad phonetic class evidences once every 10 ms. The evidence vectors are then supplied to a dynamic programming algorithm which generates a multi-level segmentation of the utterance. Once the segment boundaries are known, context-dependent descriptions of the segments can be supplied to specialized MLP's which perform the final phonetic classification of the segments. The obtained phonetic hypotheses are then supplied to a second dynamic programming stage which will search for the optimum word sequence.

References

- [1] A. Bendixsen and K. Steiglitz (1990). "Neural networks for voiced/unvoiced speech classification," Proceedings ICASSP 90, 521-524.
- [2] Y. Bennani, F. Soulie, P. Gallinari (1990). "A connectionist approach to automatic speaker recognition," Proceedings ICASSP 90, 265-268.
- [3] H. Bourlard and C. Wellekens (1989). "Speech pattern discrimination and multilayer perceptrons," Computer, Speech and Language 3, 1-20.
- [4] R. Cannon, J. Dave and J. Bezdek (1986). "Efficient implementation of the fuzzy c-means clustering algorithm," IEEE Trans. Pattern Analysis and Machine Intelligence, PAMI 8, 258-255.
- [5] T. Kohonen (1984). *Self-organization and associative memory*. Series in Information Sciences, vol 8 (Springer Verlag, Berlin).
- [6] T. Kohonen (1988), 'The Neural Phonetic Typewriter', IEEE Computer Magazine, 11-22.
- [7] R. Lippmann (1987), 'An Introduction to Computing with Neural Nets', IEEE ASSP Magazine, 4-22.
- [8] J.P. Martens and L. Depuydt (1990). "Broad phonetic classification and segmentation based on neural networks and dynamic programming", to appear in Speech Communications 10 (number 1).

- [9] N. Morgan and H. Bourlard (1990). "Continuous speech recognition using multi layer perceptrons with Hidden Markov Models," Proceedings ICASSP 90, 413-416,
- [10] K. Paliwal (1990). "Neural net classifiers for robust speech recognition under noisy environments," Proceedings ICASSP 90, 429-432.
- [11] S. Peeling, R. Moore and A. Varga (1987). "Isolated digit recognition using the multi layer perceptron," in *Recent advances in speech understanding and dialog systems* (edt. Niemann), 261-265.
- [12] T.J. Sejnowski and C.R. Rosenberg (1987), "Parallel networks that learn to pronounce English text", *Complex Systems* 1, 145-168.
- [13] J. Wilpon and L. Rabiner (1985). "A Modified k-Means Clustering Algorithm for use in Isolated Word Recognition", *IEEE Transactions on Acoustics Speech and Signal Processing* 33, 587-594.

Speech Recognition in Noisy Circumstances

Dirk Van Compernelle¹

K.U. Leuven - ESAT
Kardinaal Mercierlaan 94
B-3001 Heverlee
Belgium

E-mail: compi@esat.kuleuven.ac.be

I. INTRODUCTION

Variation in acoustic environments is huge due to variety in room reverberation characteristics and background noises. Robustness against this variability could be built into recognition systems by spreading out the data gathering for training over many different environments - much in the same way that speaker independence is achieved by averaging data from hundreds or even thousands of speakers. However the acoustic variability is even larger than speaker variability, limiting such a statistical approach to small vocabulary systems. Furthermore, success cannot be guaranteed, as inherent robustness might require adequate processing in the first stages of the recognizer.

Sadly enough, much of the work in speech recognition and speech research in general is still based on recordings made in a quiet room. Such recordings are essential in speech production research, but they are fully misleading, however, if one wants to define salient features for speech recognition and understanding. As a consequence many glamorous systems are characterized by a total absence of acoustic robustness. For potential users of speech recognition systems it is frustrating to see the performance of a system drop by several orders of magnitude when one switches from a lip microphone to a far talk microphone or when the air conditioning in a room turns itself on.

In this paper we will first review the properties of human performance in (not so) noisy conditions. Then starting from a prototype speech recognition system features are added which have shown their potential in enhancing the machine robustness significantly. No technical details are presented, these can be found in the references.

II. HUMAN SPEECH UNDERSTANDING IN NOISE

Background Noise *Noise* is most often defined in a subjective way as *anything except the desired signal*. For speech recognition purposes it could as well be defined as anything that hampers optimal recognition. A bothersome noise level is something very subjective and is very different indeed for HI-FI listening conditions vs. office communication conditions or optimal conditions for your automatic speech recognition system.

¹Research Associate of the National Fund for Scientific Research of Belgium (N.F.W.O.)

Before thinking further about machines it is good to see how different conditions affect human performance. Background noise doesn't affect speech understanding at all as long as the SNR (signal to noise ratio) is better than 20dB. This is so in most daily situations. Most people barely notice if the SNR is 30 or 40 or only 20 dB. With lower SNRs things progressively get worse, but basic conversation is still possible at SNRs around 0dB. A most common situation in which human recognition is affected by background noise is inside a driving car, where an SNR of 6dB is common.

Room acoustics Absorption coefficients of many materials are highly frequency dependent. Hence reverberant recordings in a normal room vs. close talk recordings undergo filtering aside from strong phase shifts. Varying room acoustics is one of those typical effects that we are well aware of in "music listening mode", but rarely in "speech listening mode". The information in a speech signal is highly redundant and understanding of speech filtered by filters as narrow as one octave and less poses little problems. Sensitivity of ASR systems to acoustics is much greater, largely due to the use of speech modeling instead of hearing modeling strategies inside the recognizer.

A Classification of Noisy Conditions For ASR systems any environment in which the recorded signal does not well fit a signal generated from a speech production model can be considered as noisy. This deviation can be due to room reverberation, to background noise, limited bandwidth recording channel, stress condition etc. Different typical noise conditions are best illustrated by a number of examples.

- **Clean Conditions:** Any situation with an SNR of 45dB or above, i.e. the noise is at level of quantization noise in a 12 bit A/D system, can be considered as clean for speech recognition purposes. This situation is achieved in free field recordings in a silent room and with the use of a lip microphone in a normal office or living room.
- **Office Environment:** The office environment is the prototype environment where noise levels are low enough not to affect human hearing, but nevertheless pose difficult situations to automatic speech recognition systems. The noise level in an office rarely reaches an annoying level with

typical SNRs between 20 and 50dB. Most common noise sources are computer fans, airconditioning and (lowpass) filtered road noise which are rather broadband but mainly low frequency noises. Furthermore there are impulsive noises due to clicks, typing, slamming doors, telephone rings etc.. Another large variability in offices exists in their reverberation characteristics which also can have a significant filtering effect on the speech signal. Problems posed by office environments are due to variability in conditions rather than absolute noise levels.

- **Telephone Quality:** The telephone network filters speech between 300Hz and 3300Hz. The amount of filtering however is fully dependent on the actual connection. One simplifying factor with telephone speech is the relative absence of reverberation and the rather predictable noise level (25-30dB), though these statements do not hold for hands free telephone and mobile telephone connections. Another annoying side effect of telephone communication is nonlinear distortion resulting from the many switches in the telephone network.
- **Noisy Conditions:** Medium level stationary background noise ($6dB < SNR < 18dB$) is typical for hands free telephone operation in a driving car and speech recognition in factory environments.
- **Lombard Speech:** High background noise levels and stress conditions cause speech distortion on top of additive noise. This situation is intrinsically different from the previous ones and will be dealt with briefly.

This paper concentrates on methods which are most efficient in the not so noisy but highly variable office environment, but the general approach should be inspiring for all possible situations.

III. A PROTOTYPICAL SPEECH RECOGNITION SYSTEM

In order to illustrate the different strategies it is easy to start from a rather prototypical speech recognition system consisting of:

1. short time spectral estimation (feature extraction)
2. vector quantization (rough classification)
3. hidden Markov model recognition system (high level recognition)

A system as described above is prototypical of most of the existing large vocabulary systems (IBM Tangora, Dragon Dictate, etc.). Two aspects of these systems are of critical importance:

1. Large vocabulary systems are isolated word systems and depend heavily on accurate silence detection for word separation. The word separation is not explicit but imbedded in the recognizer, as it looks for a most likely word string of the type "silence-word-silence-word- ... -silence".
2. Parameters of a speech recognition system are set (adapted) during a "training session", often not lasting longer than a couple of minutes. A system can only recognize what it has learned !!! It will adapt to speech characteristics and room acoustics at the same time, unless you manage to separate both aspects. And the "silence model" will be a model of the silence seen during training.

It is not surprising that these systems are *not* sensitive to *absolute* noise level, as long as training and testing conditions are very similar, but *most* sensitive to small *variations* in the background conditions. The IBM Tangora 5000, first demonstrated at ICASSP '86, was insensitive to absolute SNR over a range of 50 to 25dB, but could not tolerate variations of 6dB or more between training and testing[1].

The task of building acoustically robust speech recognition systems is one of environment normalization and not just noise removal !!

IV. ROBUST SPECTRAL ESTIMATION

By robust spectral estimation I mean obtaining an "environment independent" spectral estimate as required in step 1. of the prototypical recognizer. It is proposed to achieve this robustness by the combination of three operations: background noise removal by spectral subtraction, channel equalization and masking of the variable residual noise from spectral subtraction by noise which is prototypical for the recognizer.

Speech in Noise Model Two basic, though reasonable assumptions, are used in almost all robust spectral estimation techniques. The speech and noise signals are assumed to be additive and uncorrelated so that their crosscorrelation is zero and the filtering due to room acoustics and recording equipment is assumed to be linear. Given these assumptions free field power spectral estimation can be written as:

$$y(t) = g(t) * s(t) + n(t) \quad (1)$$

$$Y^2(f, k) = G(f) \cdot S^2(f, k) + N^2(f) \quad (2)$$

in which t is time, f is the frequency and k a frame index indicating the window over which the spectral estimate was computed. $G(f)$ and $N^2(f)$ are only slowly time-varying and to a first order approximation independent of the frame index k .

Spectral Subtraction Spectral subtraction as described by S. Boll in [2] has been the basis of many

noise suppression algorithms. In communication systems spectral subtraction has been able to increase SNR significantly, but only in a few specific cases has it been able to improve speech intelligibility. However for speech recognition spectral subtraction has been very successful due to its normalizing effect over a wide variety of backgrounds. An estimate of the noise spectrum is obtained by averaging power spectra estimates over periods assumed to be speech free. A least squares estimate of the speech power spectrum is then obtained as:

$$\hat{S}^2(f, k) = Y^2(f, k) - \bar{N}^2(f) \quad (3)$$

As power spectral estimates must always be positive, hard clipping at 0 or another appropriate measure is required.

Spectral Subtraction Enhancements Least squares estimation of power spectra is far from optimal with respect to properties of the ear and common distortion metrics within a speech recognizer. Therefore modifications can be made to the original spectral subtraction formulation such that minimization of the estimation error isn't in the power spectrum domain but rather in the log spectrum or another appropriate domain [3], [4],[5].

Acoustic Channel Normalization Spectral subtraction doesn't take into account the variable transferfunction $G(f)$ from the recording channel, in which the channel is the combination of the acoustic room transferfunction and the recording electronics (microphone, amplifier, etc.). Once measured, it is easy to compensate for this rather slowly varying filter.

$$\hat{S}^2(f, k) = G^{-1}(f)[Y^2(f, k) - \bar{N}^2(f)] \quad (4)$$

Room acoustics, however, don't just vary from session to session, but even over a short time span, e.g. opening of a door. Hence an adaptive estimation of this transferfunction is required. A reasonable estimate can be derived from peaks in frequency channel histograms, measured over a few seconds (e.g. 10). This relies on two assumptions: first, that typical conversation varies quickly in acoustic phonetic content and that a variable acoustic phonetic content provides a good spread of spectral peaks over the whole frequency range.

Noise Masking If only consistency between training and testing matters, then one could ask: "Why not add always the same prototypical white noise to mask out the natural environment?" The idea is far from silly, indeed, and can be used to one's advantage. Of course this masking noise will mask speech as well as the background, leading to the key question of how much noise can be added before the bad effects outweigh the normalizing effect. Straightforward reasoning gives the answer:

- Speech dynamics of roughly 20-25dB is adequate for excellent speech communication. Noise at 25 db or more below the peak speech level will only mask very low energy speech events which are detectable in quiet environments and not in noisy ones and which are not critical. Hence one should avoid that a speech recognition system learns about them. Thus, the amount of added noise should be adaptive and relative to the speaking level. Masking noise levels at 15-25 dB below the peak speech level are reasonable choices; the different levels will yield a tradeoff between increased robustness and optimal performance in clean conditions.
- Masking noise will only efficiently mask events that are 6dB lower and more. The residual noise of spectral subtraction is considerably lower than the input noise, though less predictable. Using spectral subtraction before adding masking noise will therefore be very helpfull and increase the effectiveness of this method by a margin of 12dB.

As final "robust spectral estimation" we hence propose

$$\hat{S}^2(f, k) = G^{-1}(f)[Y^2(f, k) - \bar{N}^2(f)] + N_m^2(f) \quad (5)$$

in which $N_m^2(f)$ is set such that it is about 20dB below the peak speech level.

The addition of spectral subtraction, channel normalization and noise masking to the IBM Tangora 5000 improved worst case recognition scores (training in clean, decoding in noise) from above 50% in the original system to about 8%[1].

V. NOISE ADAPTIVE PROTOTYPES

The previous approach embedded environment normalization into the signal processing. Another approach exists in delaying the environment normalization until the vector quantization[6]. If transferfunction and noise level are known then original VQ prototypes can be mapped into noise adapted prototypes. Mapping of the prototypes can be exact but cluster boundaries will not be mapped precisely from one environment to another. This distortion, if not too large, will be quite gracefully handled by the stochastic nature of HMMs.

VI. NOISE ADAPTIVE MODELS

Ultimately it is possible to implement environment adaptation in the HMMs themselves. Reestimation of the parameters of the noise models can be through an independent speech silence discrimination algorithm[1]. This can be done quite accurately as no 100% correct word segmentation is required, though only average noise statistics. In isolated word recognition systems this will guarantee a good silence detection. However, in order to be effective not only the noise model parameters but also the speech parameters must be adjusted. This is possible in continuous density HMMs

using dominance principles. Output probability densities are modified on the basis of the maximum noise level from training and recognition such that events that fall below this level cannot be taken into account [7],[8]. To some extent this method can be seen as the equivalent of noise adaptive prototypes in VQ based systems.

VII. MULTISTYLE TRAINING

Totally different problems arise in military applications of speech recognition where background noise levels of 140dB SPL have to be dealt with. Human communication is difficult, even with the use of lip-microphones and shielding helmets. strong g-force and stress have an important impact on the speech production itself. Due to the bad conditions a speaker will attempt to compensate and shout rather than speak. This, combined with other stress symptoms, which might be due to fighting conditions or g-force, distorts the speech heavily. Obviously the additive speech in noise model is not valid any longer.

The one way out in these circumstances is to use "multistyle training", i.e. train the recognizer with data recorded in different noise and stress conditions[9]. HMMs are obviously a most suitable method for these multistyle training procedures as no intrinsic modifications to training or recognition algorithms are required, only the data base collection has to be modified. It is this expensive and bothersome data collection that limits applications to small command sets, both speaker dependent and independent.

VIII. MULTI-MICROPHONE NOISE SUPPRESSION

All preceding approaches assume more or less stationary interference. There is however a large class of acoustic interferences which are far from stationary: competing speakers, and all the impulsive noises: clicks, typing, door slam, etc.. Speech beamforming systems have recently been developed and are promising for a wide array of applications[10]. Easy delay and sum beamforming is very robust but the SNR gain is theoretically limited to $10\log_{10}M$ dB with M the number of microphones. Higher SNR gains must be obtained from single channel postprocessing, direct multichannel frequency estimation or multichannel adaptive filtering. The latter technique is quite expensive in terms of computations and must be implemented with a lot of care in order to avoid undesirable speech distortion.

IX. CONCLUSIONS

Many existing speech recognition systems are incredibly sensitive to changes in the acoustic environment, as too much of speech research is based on speech production rather than on hearing principles. Techniques such as spectral subtraction, channel normalization, noise masking and adaptive VQ prototypes have

matured substantially over the past couple of years. Except for added complexity there is no reason not to incorporate one or several of these technique into a speech recognition system. Multi-microphone noise suppression is a recent evolution and especially promising with respect to non-stationary interferences.

References

- [1] D. Van Compernelle. Noise adaptation in a hidden markov model speech recognition system. *Computer Speech and Language*, vol3.2:151-168, 1989.
- [2] S. Boll. Suppression of Acoustic Noise in Speech Using Spectral Subtraction. *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP 27 No. 2:113-120, 1979.
- [3] J.E. Porter and S.F. Boll. Optimal Estimators for Spectral Restoration of Noisy Speech. In *Int. Conf. Acoust. Speech & Signal Processing*, pages 18.A.2.1-4, 1984.
- [4] D. Van Compernelle. Spectral Estimation Using a Log-error Criterion With Applications to Speech Recognition in Noisy Environments. In *Int. Conf. Acoust. Speech & Signal Processing*, pages 258-261, 1989.
- [5] Adoram Erell and Mitch Weintraub. Estimation using Log-Spectral-Distance Criterion for Noise-Robust Speech Recognition. In *Int. Conf. Acoust. Speech & Signal Processing*, pages 853-856, 1990.
- [6] A. Nadas, D. Nahamoo and M.A. Picheny. Speech Recognition using Noise-Adaptive Prototypes. *IEEE Transactions on Acoustics Speech and Signal Processing*, 37.10:1495-1503, 1989.
- [7] D.H. Klatt. A Digital Filter Bank for Spectral Matching. In *Int. Conf. Acoust. Speech & Signal Processing*, pages 573-576, 1979.
- [8] A.P. Varga and R.K. Moore. Hidden Markov Model Decomposition of Speech and Noise. In *Int. Conf. Acoust. Speech & Signal Processing*, pages 844-848, 1990.
- [9] D. Paul. A Speaker-Stress Resistant HMM Isolated Word Recognizer. In *Int. Conf. Acoust. Speech & Signal Processing*, pages 713-716, 1987.
- [10] D. Van Compernelle, W. Ma, F. Xie and M. Van Diest. Speech Recognition in Noisy Environments with the Aid of Microphone Arrays. *Speech Communication*, 5.6:-, nov 1990. to be published.

Spraaksynthese

Duurregels voor spraaksynthese

B. Van Coile

Universiteit Gent,
Laboratorium voor elektronica en meettechniek,
St. Pietersnieuwstraat 41, B-9000 Gent, België

1 Introductie

Men kan de verstaanbaarheid en natuurlijkheid van synthetische spraak verbeteren door aan ieder foneem een gepaste duur toe te wijzen. Dit betekent dat men in een tekst-naar-spraak systeem nood heeft aan een goed duurmodel waarin de factoren die de foneemduur bepalen, worden beschreven en gequantificeerd.

Vele studies over duurfenomenen zijn gebaseerd op woordenlijsten met gewone of nonsenswoorden die ofwel geïsoleerd ofwel steeds in dezelfde draagzin werden uitgesproken (onder andere [3,5,7]). Om een duurmodel te ontwikkelen voor continue spraak, hebben sommige onderzoekers zich gebaseerd op een analyse van de foneemduur in een tekst (onder andere [2,4,6]). In wat volgt bespreken we een dergelijke aanpak voor het Nederlands [9]. We gaan dieper in op de ontwikkeling van het klinkergedeelte van ons duurmodel.

2 Basismateriaal

Een tekst met 90 zinnen en een totale duur van meer dan 8 minuten werd voorgelezen door een vrouwelijke spreker. De tekst werd op band opgenomen, gedigitaliseerd en in de computer opgeslagen voor verdere verwerking. Het volledige corpus werd manueel gesegmenteerd op basis van de visuele inspectie van de golfvorm, het energiecontour en het spectrum. De foneemgrenzen werden op de gebruikelijke manier bepaald ter hoogte van de discontinuïteiten in het signaal, de excitatie en/of de formantstructuur [7,4,6]. Bij twijfel omtrent de ligging van een bepaalde foneemgrens werd deze grens

als *onzeker* gemarkeerd. De fonemen die werden afgebakend door een onzekere grens, werden bij de verwerking van de resultaten niet gebruikt. Van de volledige tekst werd een brede fonetische transcriptie gemaakt. In deze fonetische transcriptie werden drie accentsymbolen gebruikt: één voor de primaire lexicale accenten, één voor de secundaire accenten en één voor de zinsaccenten. Ieder woord (waarin een klinker optrad verschillend van een schwa) kreeg minstens één accent toegewezen. De zinsaccenten werden door de spreker en de onderzoeker onafhankelijk van elkaar toegekend. Terwijl ze naar de tekst luisterden, duiden ze elk de woorden aan die dominant leken. De overeenkomst tussen de resultaten van beiden was werkelijk erg goed. Als er toch verschillen optraden, werd de lexicale optie verkozen. Van ieder woord werd de woordsoort bepaald alsook de frequentie van voorkomen in het Nederlands. Dit laatste gebeurde aan de hand van het corpus Uit den Boogaart (1975).

3 Methode

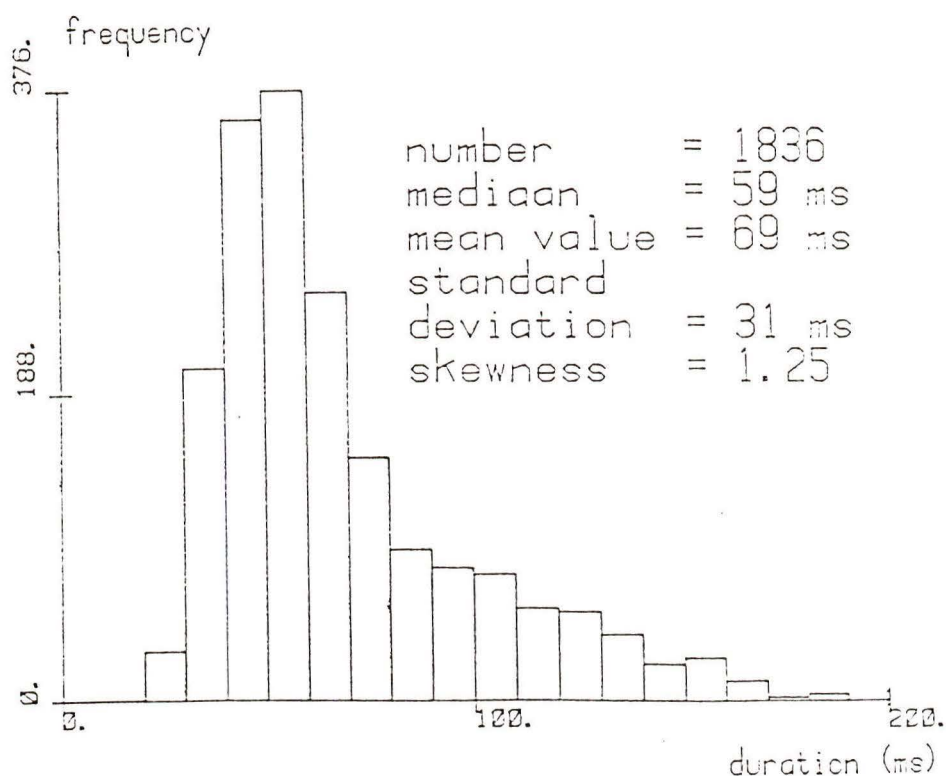
Door middel van een stapsgewijze selectie van factoren, hebben we verschillende duurmodellen opgesteld met toenemende complexiteit en accuraatheid. Tijdens dit proces werd de theorie van de kleinste kwadraten gebruikt om de parameters van deze modellen te bepalen.

3.1 De klinkerduur

Figuur 1 toont het histogram voor de duur van alle klinkers uit de tekst. De variantie in deze distributie wordt gebruikt als referentiewaarde voor de evaluatie van de verschillende duurmodellen. Het meest eenvoudige model dat we op basis van de resultaten in figuur 1 kunnen voorstellen, geeft aan iedere klinker een duur van 69 ms.

3.2 De intrinsieke klinkerduur

Elk foneem heeft een eigen intrinsieke duur. Figuur 2 geeft een overzicht van de gemiddelde waarden (en standaard afwijkingen) voor de klinkerduur per klinker. Voor de berekening van deze waarden werden *alle* realisaties van eenzelfde klinker samengenomen. Als intrinsieke waarde voor de verschillende klinkers kunnen bijvoorbeeld de gemiddelde waarden uit figuur 2 worden gebruikt. Een duurmodel dat aan iedere klinker deze intrinsieke duur toekent,



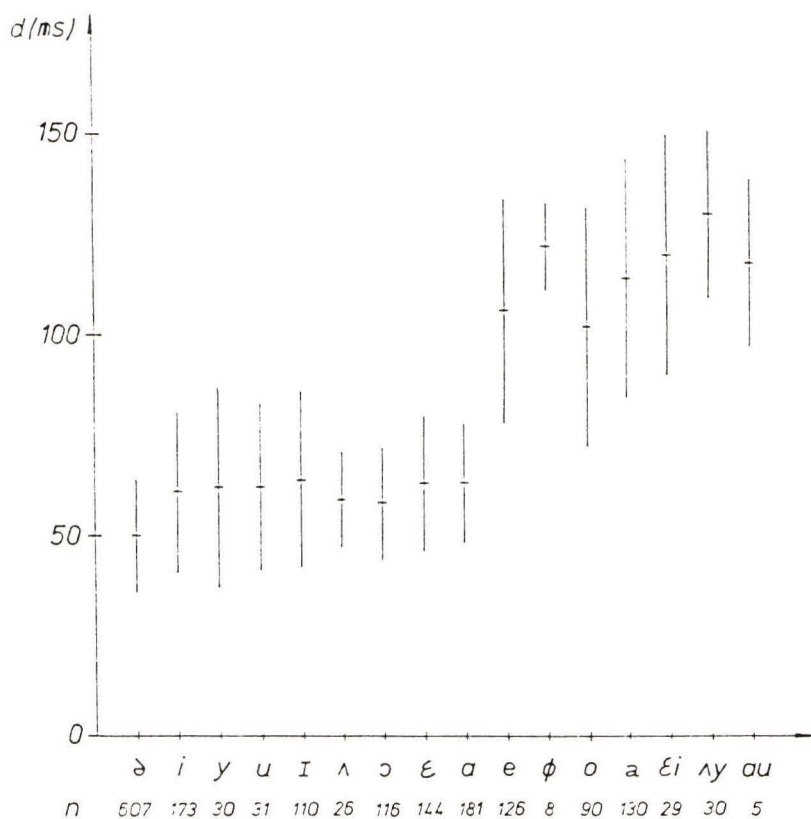
Figuur 1: Het histogram voor de duur van alle klinkers uit de tekst

verklaart reeds 56% van de originele variantie. Het is belangrijk op te merken dat er een duidelijk verschil in duur bestaat tussen twee verzamelingen klinkers: de lange klinkers en de korte klinkers. De schwa is duidelijk de kortste klinker. Een eerste, erg eenvoudig duurmodel dat aan lange klinkers, korte klinkers en aan de schwa telkens een verschillende waarde toekent, verklaart 54% van de totale variantie:

[kort] ---> 60 ms
 [lang] ---> 104 ms
 [schwa] ---> 48 ms

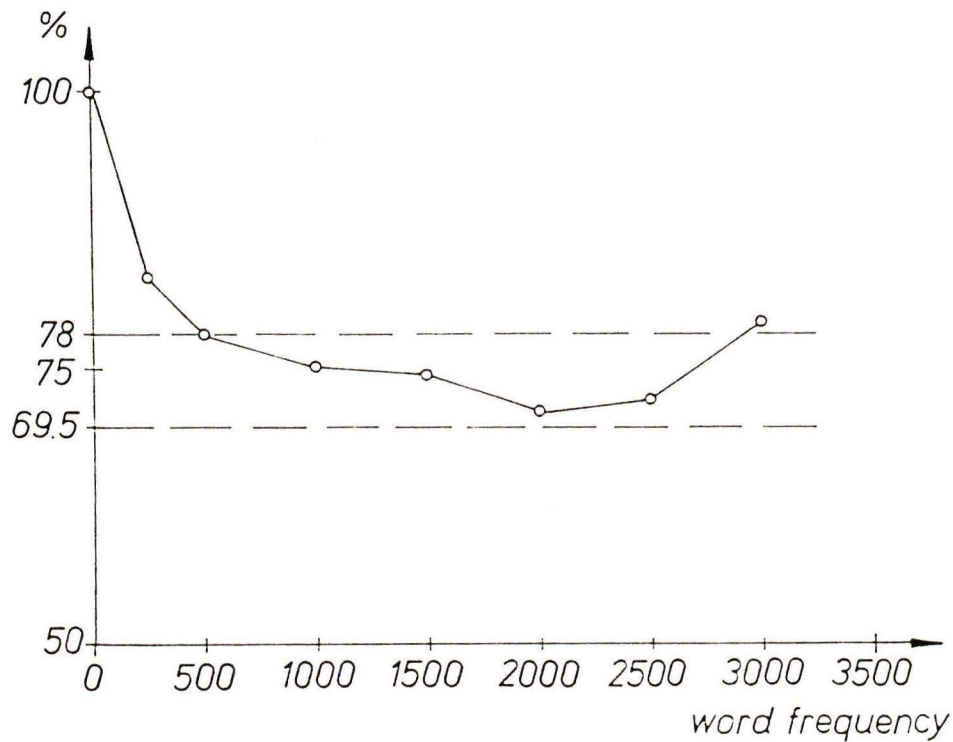
3.3 Klemtoon, functiewoorden, inhoudswoorden

Een belangrijke factor die de duur van klinkers beïnvloedt en te maken heeft met klemtoon, is het onderscheid tussen functiewoorden en inhoudswoorden. De fonemen in functiewoorden zijn over het algemeen korter dan dezelfde fonemen in inhoudswoorden. De opsplitsing in functiewoorden en inhouds-



Figuur 2: De gemiddelde duur voor iedere klinker (verticale lijn = 2.standaard afwijking)

woorden kan bijvoorbeeld op basis van de woordsoort gebeuren [6]. Als inhoudswoorden beschouwt men dan alle woorden uit de open woordklassen. Als functiewoorden gebruikt men de woorden uit de gesloten woordklassen (voorzetsels, voegwoorden, hulpwerkwoorden, voornaamwoorden en lidwoorden). De opsplitsing in functie- en inhoudswoorden kan ook gebeuren op basis van de frequentie van voorkomen van de woorden. Om de meest geschikte classificatiemethode te bepalen, hebben we de volgende procedure uitgevoerd: alle niet-prepauzale klinkers met een lexicaal accent werden geselecteerd. De variantie van deze groep klinkers werd tijdens het experiment als referentie gebruikt. Zoals reeds werd vermeld, haalden we de woordfrequenties uit het werk van Uit den Boogaart (1975). De klinkers werden in 2 groepen onderverdeeld op basis van een drempelwaarde voor de woordfrequentie. In figuur 3 zien we het overblijvende deel van de variantie voor verschillende drempelwaarden. Gebruiken we een frequentiedrempel 2000, dan wordt 29% van de variantie verklaard. Als we alle persoonlijke voornaamwoorden ook als functiewoorden beschouwen, krijgen we een iets beter resultaat (31%). Een indeling op basis van de woordsoorten verklaart slechts 22% van de totale

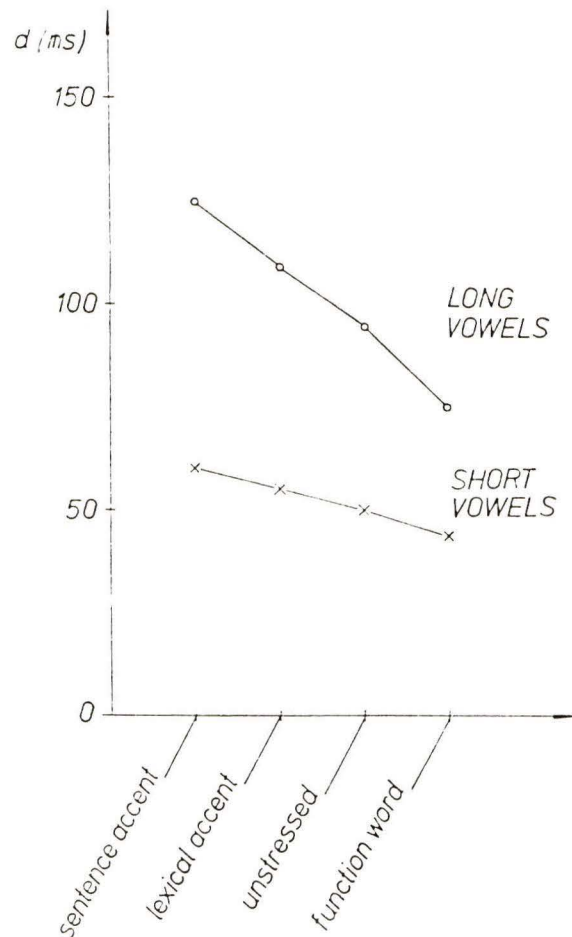


Figuur 3: Het overblijvende gedeelte van de variantie, indien de woorden op basis van hun frequentie worden opgesplitst in functie- en inhoudswoorden.

variantie. In wat volgt, werd gebruik gemaakt van de optimale opsplitsing in functie- en inhoudswoorden.

In figuur 4 is duidelijk de invloed te zien van belangrijkheid en accentuering op de klinkerduur. Er wordt een onderscheid gemaakt tussen klinkers in functiewoorden en klinkers in inhoudswoorden. De klinkers uit de laatste groep werden onderverdeeld in drie categorieën afhankelijk van hun accent-type. We onderscheiden klinkers met een zinsaccent, klinkers met een lexicaal accent en onbeklemtoonde klinkers. We kunnen nu een tweede eenvoudig duurmodel opstellen dat reeds 64% van de totale variantie verklaart:

[kort, zinsaccent]	--->	68 ms
[kort, lexicaal accent]	--->	63 ms
[kort, geen accent]	--->	58 ms
[kort, functiewoord]	--->	52 ms
[lang, zinsaccent]	--->	123 ms
[lang, lexicaal accent]	--->	107 ms
[lang, geen accent]	--->	90 ms
[lang, functiewoord]	--->	70 ms
[schwa]	--->	48 ms



Figuur 4: De invloed van belangrijkheid en accentuering op de klinkerduur

3.4 De plaats van de klinker in het woord en in de zin

Het derde duurmodel, dat ongeveer 73% van de totale variantie verklaart, houdt rekening met de plaats van de klinker in het woord en binnen de zin. Er worden verschillende aanpassingsregels gebruikt: deze regels wijzigen de toegekende duurwaarde met een bepaalde factor. Uit de literatuur blijkt dat er verlengingen optreden in de lettergrepen net voor een belangrijke syntactische grens, ook al worden deze grenzen niet door een pauze gemarkeerd [4]. Aangezien we in ons tekst-naar-spraak systeem over omzeggens geen syntactische informatie beschikken, hebben we de verlengingseffecten slechts bestudeerd in prepauzale posities. De volgende fenomenen werden geobserveerd:

- klinkers zijn langer in prepauzale posities dan in andere posities.

- klinkers met klemtoon (lexicaal accent of zinsaccent) worden korter naarmate het aantal lettergrepen dat nog moet worden uitgesproken, toeneemt. Dit effect is vooral te zien in prepauzale posities. In [5] werd een gelijkaardige effect vastgesteld in geïsoleerde Nederlandse woorden.
- onbeklemtoonde klinkers zijn langer in woord-finale posities dan in overige woordposities.

Deze tendensen werden gequantificeerd. De resultaten zijn in het duurmodel van figuur 5 weergegeven. In dit model wordt ook rekening gehouden met het feit dat de klinkers *i*, *y* en *u*, zich voor een consonant *r* als een lange klinker gedragen.

{i,y,u}	--->	[lang] /_r	
[kort, zinsaccent]	---	>	69 ms
[kort, lexicaal accent]	---	>	64 ms
[kort, - accent]	---	>	59 ms
[kort, functiewoord]	---	>	52 ms
[lang, zinsaccent]	---	>	120 ms
[lang, lexicaal accent]	---	>	104 ms
[lang, - accent]	---	>	92 ms
[lang, functiewoord]	---	>	70 ms
[schwa]	---	>	47 ms
[+accent, +prepauzaal, syl=0]	---	>	* 1.40
[+accent, +prepauzaal, syl=1]	---	>	* 1.25
[-accent, -schwa, +prepauzaal, syl=0]	---	>	* 1.56
[-accent, -schwa, +prepauzaal, syl>0]	---	>	* 1.05
[-prepauzaal, syl>0]	---	>	* 0.95
[schwa, +prepauzaal, syl=0]	---	>	* 1.20

Figuur 5: Duurmodel 3.

Syl = *x* betekent dat in het woord nog *x* syllaben op de klinker volgen.

Met [+accent] worden zowel lexicaal beklemtoonde klinkers als klinkers met een zinsaccent aangeduid.

3.5 De invloed van de volgende medeklinker

In prepauzale posities is duidelijk de invloed te zien van de medeklinker die op de klinker volgt. In andere omstandigheden is het effect erg klein. Om de invloed van postvocale medeklinkers in rekening te brengen, werd de volgende regel aan model 3 toegevoegd:

```
{[+accent, prepauzaal, syl<2],  
 [-accent, prepauzaal, -schwa, syl=0]}  
  /_[stemloze plosief]      ---> * 0.85  
  /_{[nasaal],[liquid]}    ---> * 0.95  
  /_[stemhebbende plosief] ---> * 1.04  
  /_[stemhebbende fricatief] ---> * 1.14
```

Dankzij de toevoeging van deze regels, verklaart het nieuwe model (model 4) 76% van de totale variantie. Dit is een (kleine) stijging van 3% ten opzichte van het vorige model. Dat deze regel wel degelijk van belang is, hebben we als volgt proberen aan te tonen. We hebben alle klinkers geselecteerd die aan de volgende voorwaarden voldoen: [prepauzaal, -schwa]. Binnen deze groep werd de resterende variantie gemeten wanneer gebruik wordt gemaakt van model 3. Door de introductie van de regel die rekening houdt met de invloed van de volgende medeklinker, werd 24% van deze resterende variantie verklaard.

3.6 Performantie van het model

Om een goed idee te krijgen over de performantie van de voorgestelde modellen, zou men in principe de performantie moeten meten op een tekst die *niet* werd gebruikt voor het afleiden van de modellen. Gelet op het feit dat het voorbereidende werk (opnamen, digitalisatie, segmentatie,...) erg tijdrovend is, hebben we dit niet gedaan. Om toch een inzicht te verwerven over de bruikbaarheid van de modellen, hebben we doormiddel van een rotatiemethode de performantie gemeten. Zo een rotatiemethode gaat als volgt in zijn werk:

- neem een deelverzameling van observaties als een testverzameling (x %).
- bepaal de parameters van de modellen met de overblijvende observaties.
- bepaal de performantie van het model op de testverzameling.

- herhaal deze werkwijze $100/x$ keer zodat iedere observatie éénmaal in een testverzameling optreedt.
- bepaal de gemiddelde waarde van de verschillende performantiemetingen.

In tabel 1 zijn de performanties weergegeven die werden bekomen met een rotatie van 20% en 50%. Gelet op de relatief kleine daling in performantie ten

model	geen rotatie	rotatie (grootte testverzameling)	
		20%	50%
2	64.2	63.5	63.0
3	73.0	71.4	70.5
4	75.9	73.6	72.1

Tabel 1: Performantie van de duurmodellen 2, 3 en 4

opzichte van de waarden die men bekomt zonder roteren, kunnen we stellen dat de voorgestelde modellen ook in staat zijn het belangrijkste deel van de variantie van de klinkerduur in een willekeurige tekst te verklaren.

4 Bespreking

We hebben verschillende modellen voorgesteld voor de klinkerduur. Deze modellen hadden een toenemende complexiteit en accuraatheid. Ze werden ontwikkeld op basis van een analyse van productiedata gemeten in een voorgelezen Nederlandse tekst. Aangezien we slechts één spreker hebben gebruikt, is het onduidelijk of bepaalde aspecten (en vooral de parameterwaarden) algemeen geldig zijn of enkel gelden voor onze spreker. Voor de tekst-naarspraak omzetting volstaat het echter een goed duurmodel te hebben voor de spreker die voor het aanmaken van het systeem wordt gebruikt. Het is in dit licht dat de duuranalyse van de tekst moet worden gezien.

De verschillende modellen werden perceptief *niet* geëvalueerd. Het verschil tussen de gemeten klinkerduur en de duur zoals die wordt gegenereerd met model 4, vertoont een standaardafwijking van 15 ms. Dit is over het algemeen minder dan het juist waarneembare duurverschil voor een enkelvoudige verandering van de segmentale duur in zinnen [1]. De betere modellen (3 en

4) verklaren het grootste gedeelte van de variaties in de klinkerduur zoals die in een tekst worden geobserveerd. Derhalve zijn ze zeker bruikbaar bij tekst-naar-spraak synthese van zinnen. Dit wordt ook bevestigd door de *informele* perceptieve beoordeling van synthetische spraak.

Referenties

- [1] J. Allen, S. Hunnicutt, en D. Klatt (1987), *From Text to Speech: The MITalk System*, Cambridge: Cambridge University Press.
- [2] T.H. Crystal, en A.S. House (1982), "Segmental Durations in Connected Speech Signals: Preliminary Results," *JASA*, vol. 72 (3), pp. 705-716.
- [3] D. Klatt (1974), "The Duration of [s] in English Words," *Journal of Speech and Hearing Research*, vol. 17, pp. 51-63.
- [4] D. Klatt (1975), "Vowel Lengthening Is Syntactically Determined in a Connected Discourse," *Journal of Phonetics*, vol. 3, pp. 129-140.
- [5] S.G. Nooteboom (1972), "Production and Perception of Vowel Durations," *Philips Research Reports*, Suppl.5.
- [6] D. O'Shaughnessy (1984), "A Multispeaker Analysis of Durations in Read French Paragraphs," *JASA*, vol. 76 (6), pp. 1664-1672.
- [7] G.E. Peterson, en I. Lehiste (1960), "Duration of Syllable Nuclei in English," *JASA*, vol. 32(6), pp. 693-703.
- [8] P.C. Uit den Boogaart (1975), *Woordfrequenties in geschreven en gesproken Nederlands*, Utrecht: Oosthoek, Scheltema en Holkema.
- [9] B.M. Van Coile (1987), "A Model of Phoneme Durations Based on the Analysis of a Read Dutch Text," *Proceedings "European Conference on Speech Technology"*, *Edinburgh*, vol. 2, pp. 233-236.

Speech Maker: a Framework for Text-to-Speech Systems

Hugo C. van Leeuwen*

Abstract

Speech Maker is a general framework in which one can implement a text-to-speech system. It is being developed at the Institute for Perception Research (IPO), and in the SPIN-ASSP project it is used as the framework in which a text-to-speech system for Dutch is being implemented. In this paper three important aspects of this system are discussed: the data structure which is used to store relevant data for the text-to-speech process, the general architecture of the entire system, and a formalism which can be used to manipulate the data structure. The implementation of the Dutch text-to-speech system is used in this paper as an example to illustrate these three aspects.

Multilevel synchronized structure

Most text-to-speech (TTS) systems consist of a serial control structure and a linear data representation (Carlson & Granström (1976), Kerkhoff, Wester & Boves (1984), Kulas & Rühl (1985), Allen, Hunnicutt & Klatt (1987), Van Rijnsoever (1988)). A serial control structure means: the various modules (such as grapheme-to-phoneme conversion or intonation contour generation) are called upon in a certain order and do not interact. A linear data representation means: the information which is transferred from one module to the other is coded in a linear way, often in a string of characters. Such a string may need to contain information of different types. For instance, a string presented for grapheme-to-phoneme conversion may need to contain information on sentence accent, morphological structure and orthography. The Dutch word 'partijvoorzitterschap' can be represented at this level as 'partij#voor%zitt%er%schap', where '' denotes sentence accent, '#' a morphological boundary between compounds and '%' a boundary between stems and affixes.

In Speech Maker a serial control structure is maintained, but the linear data representation is replaced by a multi-level, synchronized data structure. In this structure, information of different types is represented on different levels. The information on the different levels is synchronized by so-called sync marks, which are placed between all data items on each level. For instance, 'partijvoorzitterschap' can be represented as follows in Speech Maker:

accent:						+				
morph:		stem		prefix		stem	-	suffix		suffix
graph:		partij		voor		zitt		er		schap

Here, the morpheme types are coded directly instead of indirectly by the different types of boundary, and sentence accent is indicated by a more insightful code ('+' instead of '');

*Institute for Perception Research, PO Box 513, 5600 MB Eindhoven, The Netherlands

a quote occurs rather frequently in unrestricted text in a function which seldom means: put sentence accent on this word).

This somewhat more complex way of data representation, which has first been introduced by Hertz, Kadin & Karplus (1985), is chosen for two main reasons.

1. We believe that it is more transparent. Each level represents a different type of information. The representation of the information at a certain level need not necessarily be different from the representation of information at other levels. For instance, the presence of sentence accent (i.e. which words are to receive accent in the sentence) and word stress (which syllables carry word-internal stress) may both be indicated by '+' since the level itself disambiguates.

Moreover, text-to-speech systems will have to be expanded for some time to come, since they are not yet perfect. More information of linguistic nature will become available to the system, which will be increasingly difficult to code transparently in a linear string. For instance, word class determination is closely related to morphological analysis. Word class is needed for syntactic analysis, which in turn is needed for determination of sentence accent and pause position. Even if it were possible to code such information transparently in a linear string, separating the different types of information explicitly will considerably enhance its transparency. And we believe a transparent data representation increases comprehension of the system's internal state. This in turn increases the speed with which a user can develop new modules or improve existing ones, given the correct tools to interact with such a structure.

2. Separate modules in the TTS system become more independent of each other. A certain module is developed at a certain time. It is designed to deal with certain kind of input. When a new module is added to the TTS system it is well possible that an existing module will be confronted with input it cannot deal with. For instance, if the word class of a certain word is 'noun', and if this is represented by '[n]' preceding the orthographic representation of the word, the grapheme-to-phoneme conversion will probably not be able to recognize this as information which is not to be pronounced, unless the module is altered once word class information is added to the system. This is undesirable, since this means that each module in the entire TTS system may need to be adjusted when the system is expanded.

General architecture

The multi-level synchronized structure is the heart of Speech Maker. All data transferred between modules pass through this structure. Also, the data in the structure are the only data directly accessible for the user. Therefore, in the context of Speech Maker the multi-level synchronized structure is called the central data structure (CDS).

The general architecture of Speech Maker is therefore as follows: the first module reads the input from a file or from the terminal and writes it into the central data structure. Each consecutive module reads its data from the CDS. Often, not all data present in the CDS are necessary for a certain module, so a selection is made. This is done by a module-specific interface which collects the relevant data, and transforms them to the input format of the module. In the same way, the output of the module is collected by the interface and written

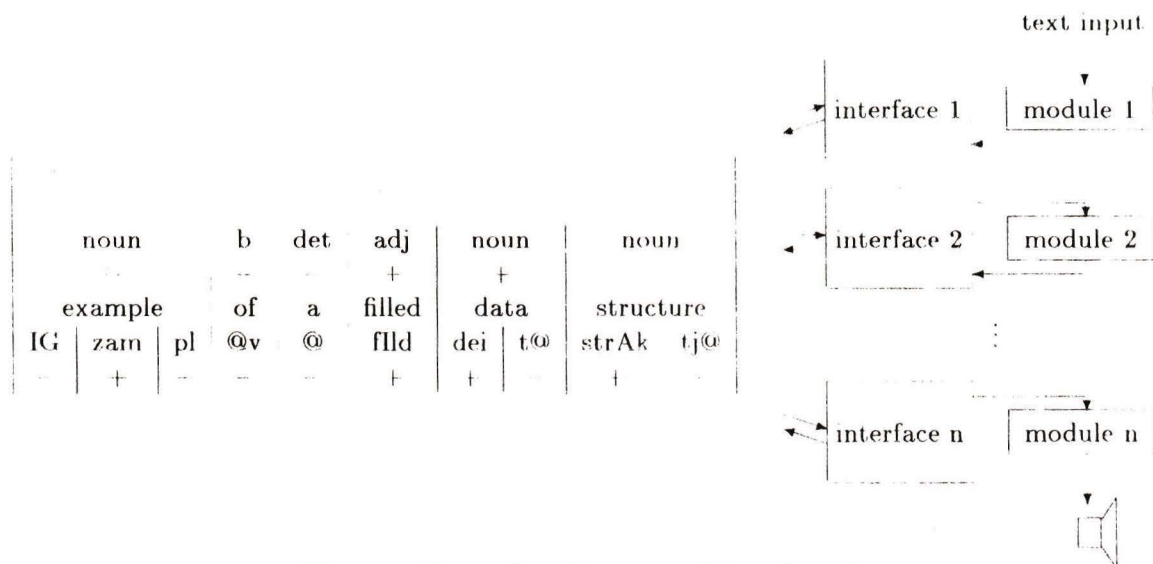


Figure 1: General architecture of Speech maker.

into the CDS and synchronized with the other levels. Some modules will fill a new level with information (e.g. a grapheme-to-phoneme conversion module fills the phoneme level), some modules will alter an already filled level (e.g. a phonological module may alter the phoneme level). The last module, however, only reads its data from the CDS, and its output is sent to a loudspeaker. This architecture is depicted in figure 1.

The Dutch text-to-speech system

The general Speech Maker architecture will have to receive a specific implementation for a specific language. In the SPIN-ASSP project a text-to-speech system for Dutch is under development which will be referred to as 'Spraakmaker'. In Spraakmaker the following levels of information and linguistic modules are implemented.

The most important levels of information to be distinguished are:

- a) Sentence: indicates the begin and end of the sentence and its type (declarative/interogative).
- b) Intonation phrase: indicates the begin and end boundaries of an intonation phrase. Between the intonation phrases a pause will be added and a declination reset occurs
- c) Word: indicates the begin and end of a word and contains all information concerning words. Not all information is of the same type. Therefore, several sub-levels are introduced. The information differs from sub-level to sub-level, but the synchronization is the same for all sub-levels. Therefore the whole can be considered as one level. The most important sub-levels are:
 - type: indicates whether a word is lexical (the 'normal' words, to be dealt with by the morphological analyser), or a number or an acronym (to be dealt with by specialized rule-based expansion modules).
 - class: gives the part of speech (or word class) of the word.
 - accent: indicates whether or not the word is to receive sentence accent

- d) Morphology: indicates the morphological structure of a word and the type of the morphemes (prefix/stem/suffix).
- e) Syllable: gives the syllable structure of a word and the stress level of the syllables (primary/secondary/none).
- f) Graphemes: gives the orthography of the word
- g) Phonemes: gives the phonemic representation of the word. Attached to each phoneme is an indication of its duration.
- h) Pitch movement: gives the type of pitch movement according to the Dutch intonation grammar of 't Hart & Cohen (1973). There are four additional sublevels which specify the movement parameters:
 - anchor: gives the anchor of the pitch movement (vowel onset/end of voicing).
 - onset: gives the timing onset relative to the anchor (e.g. a pitch lending rise in Dutch usually starts 70 ms before vowel onset).
 - duration: gives the duration of the pitch movement
 - excursion: gives the excursion of the pitch movement (for instance in semitones).

At the time of publication of this paper most of the above levels have already been implemented in Spraakmaker, except for the phoneme duration sublevel and the pitch movement level. When all of the analysis modules have operated (and the input sentence has been uttered) the CDS will look as follows¹ (suppose the input sentence was "De bal vloog over de schutting").

sentence:	declarative									
int_phrase:										
word.type:	lex	lex	lex	lex	lex	lex	lex			
.class:	det	noun	verb	b	det	noun				
.accent:	-	+	-	-	-	+				
morph:	stem	stem	stem	stem	stem	stem	stem	suffix		
syllable:	-	+	+	+	-	-	+	-		
grapheme:	de	bal	vloog	o	ver	de	schu	tt	ing	
phoneme:	d@	bAL	vlox	o	v@r	d@	sxU	T	IN	

(1)

The second part of Spraakmaker's architecture is the succession of modules which operate on the CDS. The following modules are implemented, and are called by Spraakmaker in the order in which they are presented here.

1. Label: reads text from a file or from the terminal. Marks the beginning and the end of a sentence and labels the words in the sentence. A wide range of labels is used to characterize the words (abbreviations, telephone numbers, amounts of money, etc.). Thus, the grapheme, word.type and sentence levels are initiated.
2. Expand: deals with all input that has received a special label, and expands or rewrites the input to one of the three output types, lexical, number, acronym. For instance, after application of LABEL, "tel. 050-123456" is represented in the CDS as:

¹Phoneme duration and pitch movements have been not included in this display, since they are not yet included in Spraakmaker.

word.type:	abbr	tel nr
graphemes:	tel	050 123456

The expander will alter these data into:

word.type:	lex	d	d	d	d	d
graphemes:	telefoon	0	50	12	34	56

3. **Word:** deals with several aspects of grapheme-to-phoneme conversion on the word level. This comprises phonemic representation, word stress, morphological and syllable structure and word class. These different output types are included in one module since it is relatively efficient to determine them at the same time. Lexical items must be analysed morphologically to arrive at the correct pronunciation. For this purpose a morpheme lexicon is needed, and once one has a lexicon, one can also store phoneme representation, syllable and stress information. Combining the morphemes renders word class. Changes to the pronunciation due to morphological structure (e.g. 'reduction' ↔ 'reduce') will be dealt with by the next module. Non-lexical items (numbers and acronyms) are dealt by specialized, rule-based modules which determine above output types in a more modular manner. The input is the orthography (the grapheme level), the output is written into the phoneme, syllable, morphology and word.class levels.
4. **Phonology:** adjusts phonemic representation. Since the previous module is word based it cannot deal with phonologic effects such as assimilation which transcend morpheme and word boundaries. These effects are dealt with by this phonologic module. Input and output are both the phoneme and syllable levels.
5. **Prosodic analysis:** determines sentence accents and the begin and end of intonation phrases. Takes its input from the word class sublevel and fills word accent and intonation phrase (sub)levels.
6. **Duration²:** determines durations of the phonemes. Takes as input the phonemes, syllable structure, word stress and sentence accent. Output is written to the duration level.
7. **Pitch movements²:** determines the relevant pitch movement parameters. Input: sentence range, intonation phrase, sentence accent, word stress; the output is written to the pitch movement level.
8. **Speech synthesis:** determines the speech waveform. Takes as input the phonemes, syllable structure, segmental duration, intonation phrase and pitch movement parameters. The output (a sample data file) can be written to a file, and/or is sent to a loudspeaker via a DA-converter.

Speech Maker Formalism

As can be seen in figure 1, each module needs a specific interface to communicate with the CDS. This is necessary if the module expects linear input and produces linear output. This

²Not yet implemented

will be the case for most modules which already exist, e.g. modules extracted from a 'linear' TTS system. New modules, however, can be developed according to the philosophy of the multi-level synchronized data structure. Modules which can directly access the data structure do not need an interface.

For this purpose a formalism has been devised which enables the rule-writer to directly manipulate the CDS. This formalism is called SMF which stands for "Speech Maker Formalism". With SMF, a linguist can algorithmically manipulate the CDS directly.

The central idea in SMF is that the rule-writer can access and modify the CDS in a way that resembles the way in which Speech Maker presents it to the user, viz. in a two-dimensional display as depicted in (1). The rules to be used are two-dimensional too.

Every rule consists of a pattern and an action. The pattern specifies a state which should be present in the CDS. The action part specifies how the CDS should be altered. The desired state (specified in the rule) is matched to the actual state of the CDS, and if it matches the action is performed. Both the pattern and the action part can encompass one or more levels.

An example of an SMF-rule is given below. Its purpose is to attach a certain pitch movement (indicated by '1&a') to a stressed syllable in an accented word in sentence final position:

```

sentence:      |
word.acc:  |< + |
syllable:  | + |  -->  pitch : | 1 & a |
           ^   ^         ^         ^
    
```

(2)

Here, the arrow '-->' separates the pattern part (left) from the action part (right). The pattern part specifies constraints on three levels: the 'sentence' and 'syllable' level, and the 'word.acc' sublevel. The levels of information are called streams in the SMF environment.

These constraints are synchronized by means of the vertical bars, '|'. These denote sync marks which must be present in the specified stream(s). If they are placed exactly beneath each other this means that the streams must be synchronized at that point.

In the syllable stream a '+' must be found preceding the sync mark that is synchronized with the sentence and the word level. This means that it must be the last syllable in the sentence. The syllable which carries stress must also be part of an accented word. This is indicated by the 'search left' operator, '|<'. This operator must be interpreted as follows. If, in the current example, one starts at the sync mark to the left of the '+' in the syllable stream, and one 'goes up' one level to the word stream, one should search for the first sync mark to the left of this point. This may be the same sync mark (if the original sync mark is also present in the word stream) or it may be another one (the original one is not present). Thus, in rule (2), between the sync mark thus found and the end-of-sentence sync mark a '+' in the word.acc stream must be found (i.e. the word must be accented). The search left operator and its counterpart, the search right operator '>|' are powerful operators to specify special relations between streams³, such as 'a stressed syllable in an accented word'.

The focus marks '^' in the line below the stream constraints serve two purposes. In the first place they serve to relate the sync marks in the action part to the pattern part. In

³The search right operator is not needed here since we know that the stressed syllable concerns the sentence final syllable, and therefore the end-of-sentence sync mark must be present in the word stream.

rule (2) the sync marks enclosing the '1 & a' are the same sync marks enclosing the '+' in the syllable stream. This means that if those sync marks are present in the pitch (movements) stream, '1 & a' will be inserted between those sync marks. If they are not, SMF will first insert them in that stream after which the data insertion takes place. '1', '&' and 'a' are three different data items, which are also called tokens. In general, tokens are data items in a stream that are enclosed between two sync marks. Between these tokens automatically new sync marks will be inserted, i.e. sync marks which are not present in any other stream of the CDS.

The second function of the sync marks is to indicate how the matching process should proceed after the matching of the rule. A rule is always part of a rule set. This is an ordered set of rules which are matched from first to last. A rule set also has a 'focus stream', this is the stream through which the anchor proceeds. The anchor is a pointer to a sync mark which indicates where a rule will be matched against the CDS. Starting from one side of the CDS, the anchor will move sync mark by sync mark to the other side. For each rule set the user can choose whether the matching direction should be from left to right or from right to left.

In left to right matching the first focus mark '+' of the pattern is mapped onto the anchor. If the pattern matches, the new position of the anchor will be the sync mark to which the second focus mark has been mapped. If the pattern does not match, the next rule will be tried, starting at the same position of the anchor, or if there is no next rule, the position of the anchor is shifted to the right by one sync mark in the focus stream⁴. Vice versa, in right to left matching the second focus mark is mapped to the anchor, and the anchor is updated in an analogous way.

Thus, suppose the CDS is in the following state⁵:

sentence:	declarative					
word.type:	lex	lex	lex	lex	lex	
accent:					↑	
syllable:	-	+	+	-	-	+
graphemes:	de	man	had	een	ge	weer
pitch:						

The arrow '↑' indicates the anchor. Suppose, also, that rule (2) is part of a rule set which matches from right to left and that the syllable stream is the focus stream. Then, as can be seen, the rule matches, and when it has been applied, the CDS will be in the following state:

sentence:	declarative					
word.type:	lex	lex	lex	lex	lex	
accent:	-	-			↑	
syllable:	-	+	+	-	-	+
graphemes:	de	man	had	een	ge	weer
pitch:						1 & a

⁴This is true for token-by-token matching. In rule-by-rule matching the anchor is shifted one position directly, and the *same* rule is matched for the new position. If the final sync mark is reached the anchor is reset to the first sync mark and the next rule can be matched.

⁵Only the relevant portion is shown.

It falls outside the scope of this paper to discuss all possibilities of SMF. Some, however, have been shown in the example. In general one can use the pattern part to access data and synchronization information in the CDS, and manipulate this information with the action part. An important characteristic that has not yet been mentioned is the possibility to invoke a new rule set as part of the action which is performed when a rule matches. In this way one can build 'programs' to manipulate the CDS, and thus build a full module for Speech Maker. One can view rule sets as subroutines, the matching order of rules within a rule set as a repetition-statement, and rules as if-statements. One can perform variable instantiation in patterns and use the variables in expressions in the action part. Thus, several important aspects of general purpose programming languages are available in SMF, not as flexible and explicit as in those languages, but more implicit and dedicated to the task of manipulating the CDS. Therefore, almost everything the user needs to type in to define a Speech Maker module is directly relevant to the process of altering the CDS. Moreover, it can be formulated in a two-dimensional manner which corresponds to the mental picture one develops of the CDS when working with the system. By only having to define information which is directly relevant one can concentrate on the task of manipulating the CDS without being bothered or distracted by the syntactical aspects of general purpose languages (consider, for instance, how one would specify rule (2) in C or Pascal). It is our hope that this kind of transparency in the rule formalism improves both the speed of development and the quality of the new Speech Maker modules.

References

- Allen, J., Hunnicutt, S. & Klatt, D. (1987). *From Text to Speech: The MITalk system*. Cambridge University Press, Cambridge.
- Carlson, R. & Granström, B. (1976). A text-to-speech system based entirely on rules. *Proceedings of ICASSP 76*, 686-688.
- 't Hart, J. & Cohen, A. (1973). Intonation by rule: a perceptual quest. *Journal of Phonetics*, **1**, 309-327.
- Hertz, S.R., Kadin, J. & Karplus, K. (1985). The Delta rule development system for speech synthesis from text. *Proceedings of the IEEE*, **73**(11), 1589-1601.
- Kerkhoff, J., Wester, J. & Boves, L. (1984). A compiler for implementing the linguistic phase of a text-to-speech conversion system. *Linguistics in the Netherlands*, 111-117.
- Kulas, W. & Rühl, H.W. (1985). Syntax--unrestricted conversion of text-to-speech for German. *New Systems and Architectures for Automatic Speech Recognition and Synthesis*, 517-535.
- Van Rijnsoever, P.A. (1988). A multi-lingual text-to-speech system. *IPO Annual Progress Report 1988*, 34-40.

Multimedia workstations for the office

F.L. van Nes

Abstract

Human factors research was carried out on the application of speech in three areas of man-computer communication: instruction, voice commands for system control and annotation of documents. As to instruction, learning was found to proceed equally fast with speech and text: a number of subjects preferred speech to text. Secondly, in speech-to-text conversion, subjects preferred voice to manual commands for layout and typographic control, although text input was slower with voice than with manual commands. Thirdly, voice annotations are more readily made than script annotations, but processing times may be longer for voice than for script annotations. In conclusion, speech is a valuable medium for human-computer interaction, provided the applications are carefully chosen and a proper user interface is made.

Introduction

In contrast with human dialogues, human-computer interaction is still predominantly monomedial: generally a keyboard, i.e. a manual medium, is used for computer input and the resulting system output is nearly always presented on screen, i.e. a visual medium (Edwards, 1988). In view of this impoverished communication still obtaining after several decades of computer use, it is understandable that, spurred by technological progress, numerous efforts are now being made to investigate, develop and design multimedia interfaces.

Speech is the first candidate as an alternative or a second medium, both at the input and output side of a computer, because speech is the easiest, fastest and most natural mode of communication between human beings. We therefore welcomed the opportunity provided by the European Community's ESPRIT¹ program to intensify our research on the human factors of speech interfaces, as part of the ESPRIT-HUFIT (Human Factors in Information Technology) 'Office Automation' project. Office tasks are an interesting domain for the application of speech interfaces, in the first place because of their socio-economic importance. The following are the results from this research on the use of speech in three areas:

(1) provision of information on system control to the user, both before and during task execution: spoken instructions and help messages; (2) application of voice commands for system control purposes; (3) addition of content information such as comments or criticisms to other such information visually presented.

Content information is defined here as consisting of the variable messages chosen by the user in the application concerned. In contrast, control information is defined as the invariable control messages that are a prerequisite to enable data to be interchanged between user and computer system (Van Nes, 1987).

To assess its relative value properly, speech was contrasted with an alternative medium in all three areas investigated.

IPO annual progress report 23 1988

¹European Strategic Program for Research and Development in Information Technology

Spoken Instructions

Definition, research motive and experiments

'Spoken instructions' are understood as all the directions for use which should be studied and learned before and, where necessary, consulted during use.

Traditionally, such instructional information has always been in printed form, i.e. as text on paper or on an electronic display. But people generally dislike reading bulky printed manuals, whereas the available space for instructional text on a display may be limited, especially during tasks such as word processing. An investigation of speech as an alternative medium for instructions is held to be justified for these and a number of other reasons (Nakatani et al., 1986).

Three experiments on spoken versus written instructions were carried out in this study; two of them have been published so far (Potosnak & Van Nes, 1984; Van Nes, 1987). 'Written' here and in the rest of this paper means displayed on a CRT.

All three experiments showed in the first place that a variety of tasks, word processing, electronic mail handling and annotating an electronic text, could, in fact, be learned with spoken as well as written instructions. Learning was determined by measuring the knowledge gained about operating the system or by directly monitoring performance with it.

Performance

In the word-processing and electronic mail-handling experiments, subjects had to work first with one, then with the other instruction medium. There was a considerable learning effect in both experiments. This effect, measured in task-execution time and requested number of help messages, was greater when spoken instructions

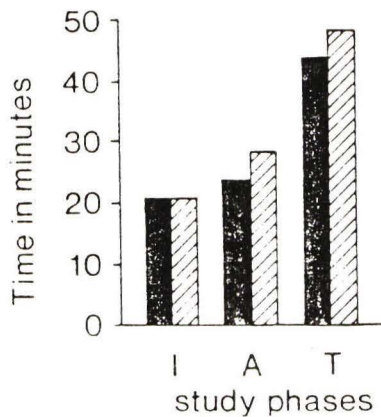


Figure 1: Average initial times needed by subjects to study an instruction set before answering questions about it by heart: I; average additional study times needed to answer the remaining questions: A; and total study times: $T=I+A$. Black bars refer to written and hatched bars to spoken instructions.

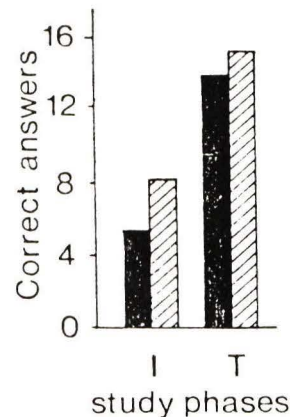


Figure 2: Average number of questions about an instruction set correctly answered by heart after an initial study period: I; and average total number of questions correctly answered: T. Both after written (black bars), or spoken (hatched bars) instructions.

were received first, which possibly implies that speech is a better medium for the initial learning of tasks of this type. The word-processing task, in particular, showed a larger decrement from the first to the second medium experienced by the subjects, both in execution time and number of help messages, when spoken instructions were given first. However, both of these performance measures were themselves also greater when spoken instructions were given first, so there was in any case more scope for improvement.

The annotation system instructions were on the average learned equally fast by subjects who received them in either spoken or written form. This involved a complete set of instructions, with corresponding questions afterwards, pertaining to all aspects of dealing with the annotation system. Learning the instructions is defined here as ability to answer the questions correctly. Figure 1 shows that when about 20 minutes was initially spent in studying the hierarchically structured instruction sets, a certain fraction of the questions could be answered by heart. After that, a somewhat longer period of study of the instructions was needed to answer the remaining questions. As regards the duration of both the initial and subsequent learning periods, there were no significant differences between read and heard instructions.

Not all answers given by heart were correct. Figure 2 shows firstly, that on the average, subjects answered about 8 questions correctly by heart after spoken instructions compared with about 5 questions after written ones. This difference was not significant, however. Secondly, Figure 2 shows that the total number of questions that were answered correctly was larger for the subjects who had received spoken instructions than for those who had received written ones. In this case the difference was significant (t-test: $\alpha < .05$).

Preference

The word-processing and electronic mail-handling experiments allowed subjects to compare spoken with written instructions. For the word-processor tasks, five subjects preferred speech, three preferred text. The reasons given for their respective preferences demonstrate that different, inherent aspects of both media, such as volatility versus permanence, are assigned a different importance by the subjects, who therefore express different preferences (Van Nes, 1987). Take for example a word-processing command divided into four steps, i.e. four consecutive key presses. With spoken instructions, subjects could look at the keyboard and press required keys in sequence while listening, without having to look at their screen in the interim, possibly several times, to read the instruction. An interview revealed (Potosnak & Van Nes, 1984) that the subjects of the electronic mail-handling task slightly favoured written instructions. However, those subjects who used the mail system with spoken instructions before they did so with written ones rated it as more interesting, more useful and more fun than subjects who used the mail system with the instruction media in reverse order.

Conclusion

The operation of relatively complicated systems, such as those for electronic text annotation, can be learned from spoken instructions. Learning may proceed just as fast with speech as with text and may possibly be more thorough. Whether speech or text is preferred for learning a task depends on the task as well as on the relative importance that users attach to properties of the instruction media.

Voice commands for system control

Definition, research motive and experiment

Voice commands are an alternative to manual commands. The dominant computer input medium is still the keyboard; but voice input may reduce training requirements and increase input speed compared to typing (Bailey, 1982). Also, typing errors remain a problem for a number of keyboard users (Ogozalek & Van Praag, 1986), whereas it has been shown that subjects may prefer voice to keypress commands, even with a considerable percentage of misrecognitions (Van Nes & Van der Heijden, 1978). With recent progress in speech-recognition technology, voice input now seems feasible, certainly in the case of the limited vocabularies that are commonly employed for control purposes. However, knowledge on the human factors of this input medium is still limited, so that experiments were carried out with voice commands, using real and simulated speech recognition.

Only one experiment, with *simulated* speech recognition, with a large vocabulary of the kind that may be encountered in a speech-to-text conversion system will be reported here. Simulated recognition has been used before, for instance to determine whether a voice-activated typewriter would be useful in composing letters (Gould, Conti & Hovanyecz, 1983). In principle, such conversion systems show an intertwined content- and control- speech input, e.g. for the correct spelling of homonyms; to distinguish punctuation marks from text words with the same spelling; or for shifting from lower to upper case. However, the control input may also be exerted by manual means in such a system, leaving the content input to voice. We investigated both selection of command buttons on a screen with a mouse-actuated cursor and voice commands for control input, in an experiment where the content input, that is messages to be converted to text, were always spoken.

Professional secretaries served as subjects in two experiment sessions, one with voice commands and the other with manual commands. They had to read a prepared text word-by-word from paper, together with a few simple layout commands, e.g. 'centre'; 'new line'. The subjects used commands such as 'text word' if they wanted to escape default interpretation of words such as 'period'. The experimenter, who sat in another room as the subject, pressed a single key for every correct input, thus displaying the already formed words, punctuation marks, etc. to the subject. 'Cognitive errors' made by the subjects, such as omitting a command like 'text word', were recorded. The simulated speech recognition employed included simulated recognition errors that had to be noted and acted upon by the subjects.

Performance

The average time per correct text-unit input was significantly lower for voice control than for manual control, see Figure 3. A 'text unit' is defined here as a word or a punctuation mark; a capitalized word (first letter or entire word) is counted as two text units. However, the rate of errors made by the subjects during their task was significantly higher for the voice-commands part, which led to an error-correction time of 43% of the total text-input time, than for the manual-commands part, where this percentage was 27. Figure 3 shows that this higher error rate led to a longer overall average entry time per text unit for voice commands. That the entry times were high in any case was due to (1) the fact that the entered words were, after he perceived them, produced and displayed by the experimenter (2) subjects having

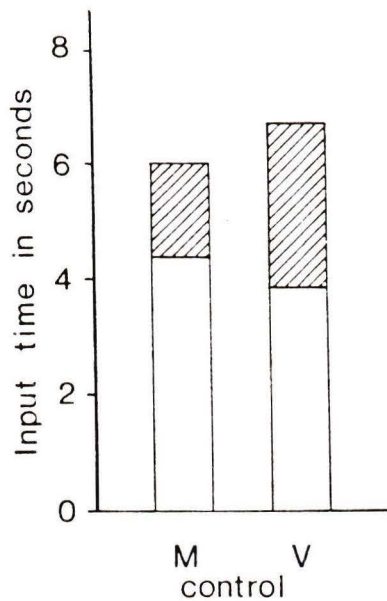


Figure 3: Total input times spent in a speech-to-text conversion task, split up into correct input (open bars) and erroneous input (hatched bars), both averaged over all subjects and text units, for voice (V) and manual (M) control.

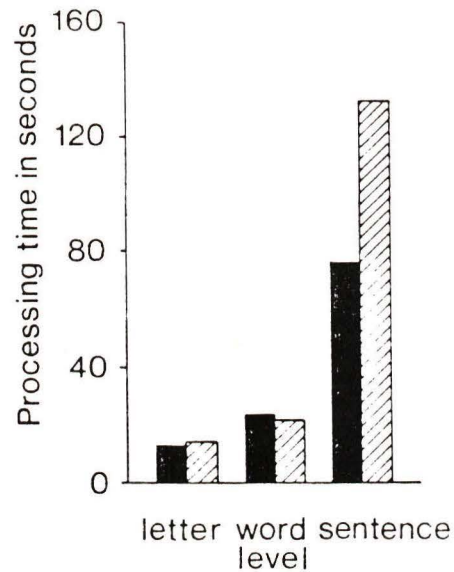


Figure 4: Average processing times for script (black bars) and voice (hatched bars) annotations, at letter, word and sentence level.

not completely mastered their task yet in the preceding training session. The last-named fact can also be concluded from the rather high number of errors.

Preference

Notwithstanding the substantial difference in error rate, 10 of the 12 subjects preferred the voice to the manual mode. One of the reasons given for this preference was: 'it seems to be faster'. In view of the objective results, this judgment is either based on the time needed for correct text-unit input actions alone, or on an underestimation of the time spent in error correction for spoken commands.

Conclusion

The speech-to-text conversion system tested had a lower overall text-input time with manual than with voice commands. However, the reverse is true when only correct input actions are taken into account.

A considerable majority of subjects preferred the voice commands to manual ones. When considering this, two factors have to be taken into account, however; subjects were already using speech for content input in the conversion system in any case, and they were not faced with (simulated) misrecognition of the *same* utterance over and over again, as may happen in real recognition systems. In our system, when a 'misrecognition' occurred, the next text unit entered was 'correctly recognized'.

Voice annotations on texts

Definition, research motive and experiments

The presentation of text on computer-driven displays provides a number of extra options in comparison with print on paper. For instance, notes to the text may be added, by the original writer or others, and stored separately so that they can be displayed together with the text or apart. Furthermore, these notes may be in written or spoken form, as typed or spoken annotations, respectively.

Voice may be a more suitable medium than text for some types of annotation. For example, the author found that voice annotations on scientific manuscripts were an effective means of transferring long and/or subtle comments, even from several annotators who might disagree. This is partly because a spoken message conveys additional information than the written one with the same wording, through its temporal structure and intonation. However, speech messages have their limitations (Bailey, 1982; Van Nes, 1982; Aucella et al., 1987), hence a study on the relative merits of voice and text annotations seemed desirable, both from the point of view of the producer or sender of the annotations and from that of their consumer or receiver.

Three experiments were carried out, one on producing and two on receiving both typed and spoken annotations.

Performance

Production. When subjects were given a free annotation task in which they had to use either text or voice, they made about the same number of text and voice annotations, but in the voice mode about twice as many words were used for conveying approximately the same information. Making text annotations took almost three times as long as voice annotations, the difference being caused by the differing production times of typing and speaking and by specific technical properties of the respective interfaces (Van Nes, 1987).

Reception. In a first experiment on receiving and subsequent processing rather complex, partly conflicting annotations from four different persons, two male and two female, both annotation types had roughly equal processing times. In a second reception experiment designed with all the findings from the first one in mind, the annotation tasks consisted of typical secretarial correction work. A male person made all the annotations; 16 female subjects, all professional secretaries, had to process them. At the level of letter and word corrections, both annotation types were processed equally fast. For corrections involving whole sentences, text annotations led to significantly shorter overall processing times (defined as the period between selection of an annotation and selection of the following annotation), as may be observed in Figure 4. This is probably due to the need to replay long voice annotations in order to be able to process them, i.e. to memory limitations of the subjects.

Preference

Production. Of the 12 subjects who had to produce text as well as voice annotations, 8 preferred the voice version for a variety of reasons, for example: 'because it is faster' or 'complete sentences are used more easily'. The 4 remaining subjects preferred the text version for a variety of other reasons, for example: 'when typing after reading (the text to be annotated) one stays in the same mental framework', or: 'it is easier to make changes'.

Reception. In the first experiment of this type, where complex annotations from four persons had to be processed, 3 subjects preferred voice, 6 preferred text and 5 had a mixed preference or none at all. A rather striking result was that the subjects were clearly influenced by the perceived authority of the respective annotators; they did not really know what to do with annotations that were formulated as well as spoken in a doubtful manner, for instance: 'In my opinion this may be viewed as...'. On the other hand, some subjects explicitly objected to being ordered, so to speak, to change the text in a certain way. This was especially true when they did not know the annotator concerned.

In the second experiment, preference for voice or text tended to depend on the level of the annotations to be processed. At letter level, 7 subjects preferred voice, 3 preferred script and 8 had no preference. This picture gradually changed at the higher text levels; at word level, 5 subjects still preferred voice against 6 who preferred script, 7 having no preference. But at sentence level, only 3 subjects (conditionally) preferred voice ('provided that the speaking rate is slowed down'), whereas 12 preferred script, 3 subjects having no preference. Thus, overall preference tended to be for text when annotations from somebody else had to be processed.

Conclusion

From the point of view of the annotator, voice appears to be the more efficient medium, because longer voice annotations took less time to make than text annotations. Two-thirds of the annotators also preferred the voice version. For receiving and processing annotations the picture is more or less reversed. With regard to performance, the advantage of voice has disappeared, as total processing times are the same or, for corrections involving sentences, even longer in the voice mode. This is reflected in the preference scores; taking all results from both reception experiments together, roughly three quarters of the subjects preferred text to voice. Making as well as processing voice annotations in more complex situations, such as writing or refereeing a scientific manuscript need to be investigated systematically, because some evidence suggests voice to be especially useful then.

Discussion

In general, the foregoing data are not unfavourable to the application of speech in human-computer interaction. An interesting aspect of the voice-preference data is that people vary in their positive or negative evaluation of inherent properties of speech, such as accentuation and volatility, and therefore judge its value differently with respect to other input and output media. In view of the fact that speech interfaces, although available for many years, are only very slowly gaining ground (Aucella et al., 1987), prudence seems justified when generalizing our research results to practice. Moreover, in practical environments other factors that were not investigated hitherto may be important, for instance disturbing others with audible speech to or from a machine. However, it seems clear that speech can be a valuable medium for computer input as well as output in appropriate applications with a proper user interface. If speech is used to *complement* manual input and visual output while making a distinction between different types of information, full advantage may be taken of the favourable properties of such a speech channel.

Acknowledgements

The work reported in this paper was carried out by a team in which all members contributed to the results. Many thanks are extended to Leo Beuk, Jan Douma, Joe Hary, Theo de Jong, all the subjects, and especially to Wessel Kraaij, Luisella Kraak, Piet van Lingen, Martijn de Loor, Henk Sprenkels and Elly van Veghel, who carried out the experiments from which results were quoted.

References

- Aucella, A., Kinkead, R., Schmandt, C. & Wichansky, A. (1987) Voice: technology searching for communication needs. *Proceedings CHI'87 Human Factors in Computing Systems and Graphics Interface (Toronto, April 5-9, 1987)*. New York: ACM, 41-44.
- Bailey, R.W. (1982) *Human Performance Engineering*. Englewood Cliffs, New Jersey: Prentice-Hall Inc., 304.
- Edwards, A.D.N. (1988) The design of auditory interfaces for visually disabled users. *Proceedings CHI'88 Human Factors in Computing Systems (Washington D.C., May 15-19, 1988)*. New York: ACM, 83-88.
- Gould, J.D., Conti, J. & Hovanyecz, T. (1983) Composing letters with a simulated listening typewriter. *Communications of the ACM*, 26(4), 295-308.
- Nakatani, L.H., Egan, D.E., Ruedisueli, L.W., Hawley, P.M. & Lewart, D.K. (1986) TNT: a talking tutor 'N' trainer for teaching the use of interactive computer systems. *Proceedings CHI'86 Human Factors in Computing Systems (Boston, April 13-17, 1986)*. New York: ACM, 29-34.
- Nes, F.L. van & Heijden, J. van der (1978) The use of computers by ordinary people. *IPO Annual Progress Report*, 13, 102-107.
- Nes, F.L. van (1982) Perceptive, cognitive and communicative aspects of data processing equipment. *Proceedings 1982 International Zürich Seminar on Digital Communications - Man-Machine interaction (Zürich, March 9-11, 1982)*, 259-262.
- Nes, F.L. van (1987) Human factors engineering of interfaces for speech and text in an office environment. *Proceedings 4th Annual ESPRIT Conference (Brussels, September 28-29, 1987)*, 1452-1457.
- Ogozalek, V.Z. & Praag, J.C. van (1986) Comparison of elderly and younger users on keyboard and voice-input computer-based composition tasks. *Proceedings CHI'86 Human Factors in Computing Systems (Boston, April 13-17, 1986)*. New York: ACM, 205-211.
- Potosnak, K.M. & Nes, F.L. van (1984) Effects of replacing text with speech output in an electronic mail application. *IPO Annual Progress Report*, 19, 123-129.

SPICOS: a cooperative natural-language dialogue system

D.G. Bouwhuis and R.P.G. Collier

Human beings are quite substantially more complex than even the most advanced technical systems. Still, humans often have much less difficulty in communicating with each other - even without any physical contact - than in communicating with technical systems. Their advantage is the use of spoken natural language. Therefore, interaction between humans and future technical systems should also be made possible in the form of human language. SPICOS is a venture to explore this possibility by integrating all the state-of-the-art expertise in language and speech technology into a demonstration system.

Ease of use

Advanced technology allows systems manufacturers to make products that can do almost anything imaginable. However, the user often finds himself unable to fully exploit the functionality of a new system, because the use of it is too complicated. This is one of the reasons that the purchase of complex systems nowadays becomes more and more dependent on elements like ease of use, employee training, cost of application development and continuity of development, rather than on system performance and price alone. Indeed, easy and effective communication with a complex system is an important aspect in the assessment of the system by a potential user. Interaction by means of spoken natural language can become a very decisive factor in such an assessment.

Philips Research has now developed the natural-language dialogue system SPICOS in a precompetitive joint research project^[1] with Siemens.

The SPICOS system

In natural-language communication between humans and machines a number of basic processing steps are required. We can recognize these from the quite typical example of a user who wants to retrieve information from a database by means of the SPICOS system

First of all, *speech recognition* is needed for automatic identification of the individual words of an utterance. This operation is in fact the conversion from spoken information to words in text and has nothing to do with interpreting or understanding, although sometimes there is a crude analysis of the syntactic structure. Next, a *linguistic-analysis* module must compute the meaning of the utterance as a whole. Frequently, this is not possible on the basis of its semantic properties only, and dialogue properties have to be taken into account as well; this is done by *dialogue handling*. As a result a logical expression is derived that can be used as a

This article has been produced in cooperation with the Philips Research Publicity Dept., Eindhoven, The Netherlands.

database query, so that an answer can be derived from the stored data. This answer is again a logical expression, which is converted into a 'written form' (i.e. ASCII format) of natural language by the *answer generation* module. Finally, the answer is converted into a spoken response by *speech synthesis* (fig.2).

If for any reason an appropriate database query cannot be derived directly, the system still has to come up with an adequate reaction to the user. This is another function of the dialogue handling module: *counterquestion generation*. On hearing the answer or the counterquestion, the user can initiate a new request and in this way effectively set up a dialogue with the system.

The different parts of the SPICOS system have been worked out by different participating groups: PFH and PRLB (speech recognition), Siemens Research (speech recognition and linguistic analysis) and IPO (answer generation, dialogue handling and speech synthesis). At the moment the working language for SPICOS is German. In the following sections we will describe the SPICOS system in some more detail, with emphasis on the contributions made by IPO.

Speech recognition

Fig.3 shows two graphical representations of a German sentence as a function of time. Upon reception, not only the actual words but also their boundaries are unknown to the system; so all of these have to be estimated. Two procedures, which are based on Hidden Markov Modelling (HMM) and an elementary language model, are used for this. They require large amounts of computing time. In order to speed up recognition, SPICOS employs a special hardware front-end and a fast look-ahead search (of some 500 ms); this approach is particularly successful in long words. But despite all measures taken, SPICOS - like all other systems in this domain - is still too slow to operate in real time.

Currently SPICOS is able to identify 1200 different words in continuous speech, which compares favourably with alternative systems.

Linguistic analysis

The input to the linguistic-analysis module is a string of words with no meaning attached; if the string represents a grammatically completely correct sentence, a logical formula is constructed in a language called^[1] EL/F, representing the syntactic structure only. This is translated into another language EL/R, in which meanings are attached to the units of what is now called the proposition. Ideally, it should be possible to construct a database query directly from this formula; in actual practice this is far from true. Many utterances in a dialogue are not information requests, while others can only be answered after preprocessing.

A further, frequently occurring difficulty is *ambiguity*, by which one sentence can lead to different database queries and possibly to different answers. Therefore, neither a pure database query language nor a linguistic-analysis system *per se* leads to a robust interactive information system.

[1] SPICOS = Siemens-Philips-IPO COntinuous Speech recognition and understanding system; SPICOS-I was completed in 1987 and SPICOS-II, described here, in 1990.

[2] EL/F = Ensemble Language / Formal; EL/R = Ensemble Language / Referential.

Dialogue handler

When the SPICOS system is unable to cope successfully with the current information, the *dialogue handling* module tries to obtain additional, mostly disambiguating information from the user. Ambiguities can have many forms; e.g. in the process of speech recognition the system may be unable to decide between 'cine' or 'keine' ('one' or 'none'). This would lead to a counterquestion like 'Did you say "cine" or "keine"?' When the user responds with either single word, recognition will become quite accurate.

Syntactic and semantic ambiguity is handled in a similar way: the system asks specifically for disambiguation that can be provided in single words by the user. Still, as the language coverage of the grammar can never be complete, questions occasionally occur that cannot be treated because the required grammar rules are not implemented.

References

In actual human dialogues one particular phenomenon is that of *implicit reference*. This happens for example in the question 'Did D write a note and did E receive it?' The word 'it' refers back to the word 'note'. This kind of backward reference is called an *anaphora* and SPICOS contains algorithms to resolve a range of basic anaphoras. Meanwhile, it has proved possible to extend the theoretical description of these references considerably. Also *cataphoras*, or forward references, can now be resolved.

Another feature of the dialogue handler is a *dialogue history*. Most linguistic-analysis tools can only provide a representation for a single sentence, dependent as they are on local syntactic structure. In a dialogue, however, previous sentences provide a context in which incomplete references can be resolved on a logical basis.

The final feature that deserves mentioning, is the *user certification* that precedes the actual interaction. In this way all references indicated by personal pronouns and possessives can be easily handled. For instance, the user can ask 'Did F get a copy of my report?', where the word 'my' will be correctly interpreted.

Presuppositions

Many questions reflect some partial knowledge of the answer, i.e. they contain some presuppositions, valid or not. For example, the question 'Did G get the leaflets?' logically leads to a search for leaflets in the database. If it turns out that G received only one leaflet, the answer could be 'No', leaving the user under the impression that no leaflet was received at all. In cases like this SPICOS interprets 'leaflets' as 'at least one leaflet'. This kind of presupposition is termed *nonfatal*, as the query can be adapted relatively easily to produce a satisfactory answer. However, wrong presuppositions are not always harmless. The question 'Was action taken after receipt of the letter?' cannot reasonably be answered if there was no letter. This is called a *fatal* presupposition, and the user is appropriately informed concerning the absence of such a letter, after which an alternative question can be formulated.

Answer generation

In the *answer generation* module, the answer from the database is converted from an abstract logical form into words. The main properties of this module are *sincerity* and *cooperation*, which means that the

information asked for is stated in full, along with a formulation of how the question has been interpreted. Consequently, an answer will not be just 'Yes' or 'No', but e.g. 'Yes, all letters were written by G'. If the user judges that this information is not what he intended to learn, he can choose a more appropriate question to proceed with.

Speech synthesis

The text string from the answer generation module is first translated into a phonetic form, because one letter may sound very differently in various contexts. Next, the speech sounds are generated by *diphone* synthesis; diphones run from the middle of one speech sound (a 'phoneme') to the middle of the next one. The word Philips e.g. would be composed as: #f, fi, il, ll, lp, ps, s#, where 'l' represents the short vowel like in 'bit', and # represents the silent period preceding and following the word. The main advantage of such a procedure is that the numerous transitions between consecutive speech sounds are automatically covered and need not be computed by an extensive rule system. This type of synthesis is fast and allows arbitrary text messages to be made audible.

Of itself diphone speech sounds monotonous. Therefore, the spoken output has to be enriched with variations in speech melody. Appropriate intonation rules specify which pitch movements may follow each other, and where they should occur. Many rises and falls of about half an octave coincide with accented words, some others occur at syntactic boundaries (*fig.4*). Usually, automatic accent lending is not explicitly related to word or sentence meaning. The handling of presuppositions in SPICOS, however, makes it possible to implement accent rendering in a more principled way. If a presupposition is violated, it is most natural to accentuate the corrected information like in 'No, G wrote two reports', in which 'two' would have been the negation of, say, 'one' and would be clearly accented. These and similar features notably increase the naturalness of the speech output, and also indicate how a question was interpreted, and how the user could proceed.

It is interesting to note that with the combination of processing modules as described above, a spoken interaction can be maintained for which, despite the enormous complexity of the system, no user manual is needed at all. In fact, there is not even one.

Implementation

Currently the SPICOS system has been implemented on three computers in a network configuration (*fig.5*). The time elapsing between the end of a spoken input and the onset of the spoken answer, is about 20 times the duration of the input. Most of this time is taken up by speech recognition, the duration of which can vary considerably, depending on acoustic ambiguities in the speech signal. Despite the fact that speech recognition in SPICOS is speaker-dependent, as it has been trained on a single speaker, the system is sometimes quite liberal and recognizes other speakers benevolently. However, there is also a speaker-adaptive mode, in which a new speaker has to say only a few words, after which he will also be able to work with SPICOS (at a slight cost in recognition rate). Future research will especially be directed at human factor aspects; these will be of overriding importance when response time has been brought down to nearly real-time levels.

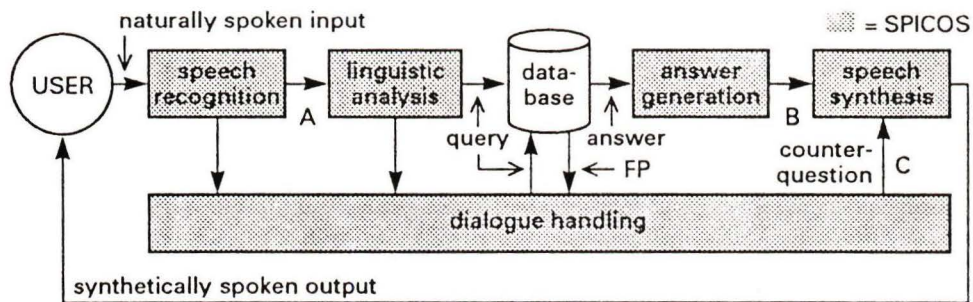


Fig.2 The information flow during interactive use of the SPICOS system. At the points labelled A, B and C communication takes place in the well-known ASCII format, which implies that the information can also directly be displayed as text. FP = fatal presupposition (see main text).

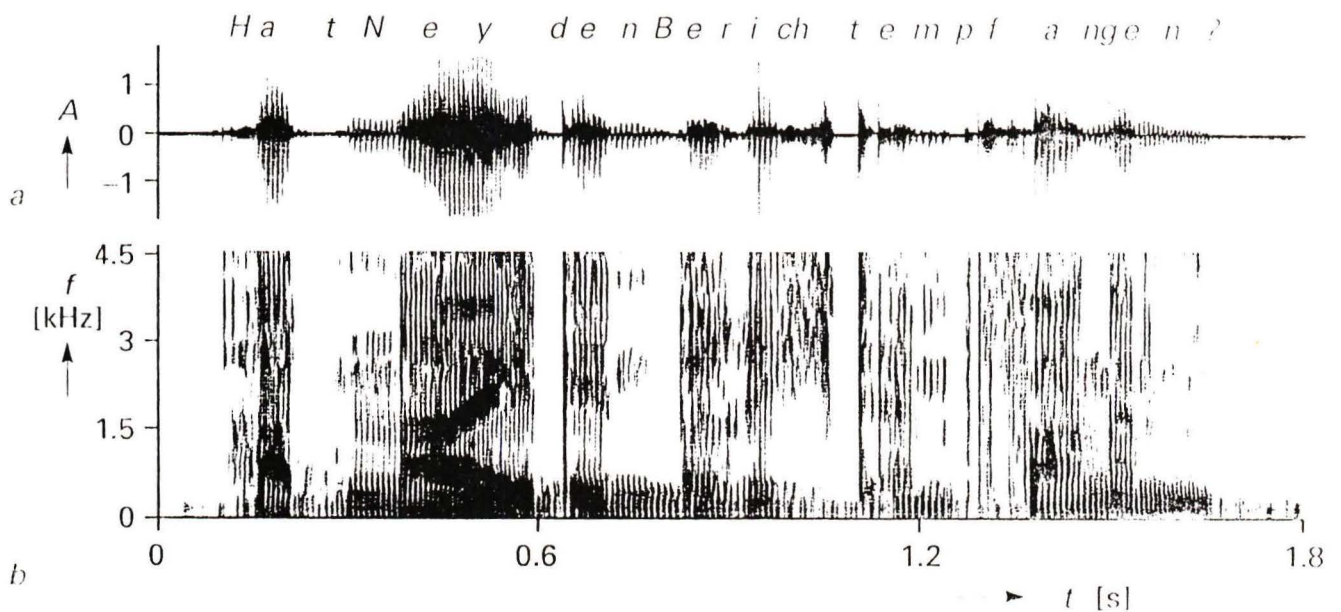


Fig.3 Two graphical representations of the naturally spoken German sentence 'Hat Ney den Bericht empfangen?', which are both a function of time. a) Instantaneous amplitude of the speech signal; b) 'spectrogram' with frequency along the vertical axis; the degree of shading indicates the intensity of the various frequency components.

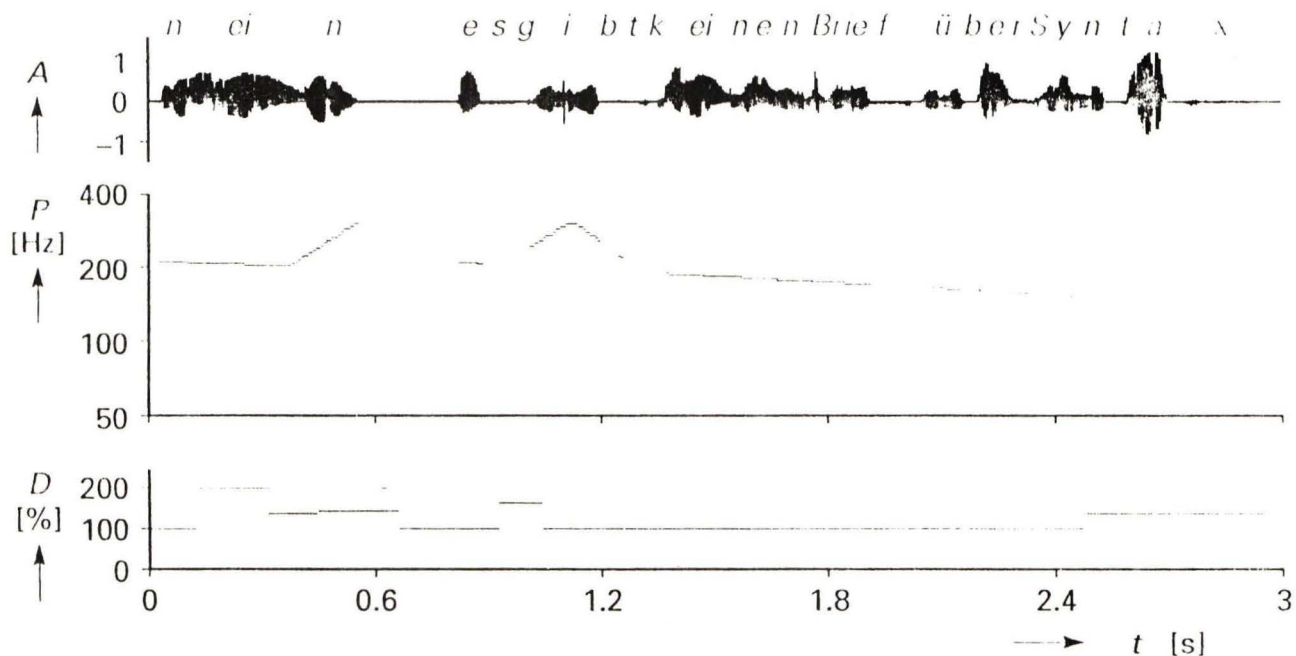


Fig.4 Synthetically spoken sentence with automatically generated pitch rise in 'nein' and contrastive accent on 'gibt', which can be recognized as local excursions of the pitch P . All speech segments have standard durations indicated by 100%. The actual duration D used for synthesis has often been chosen differently.

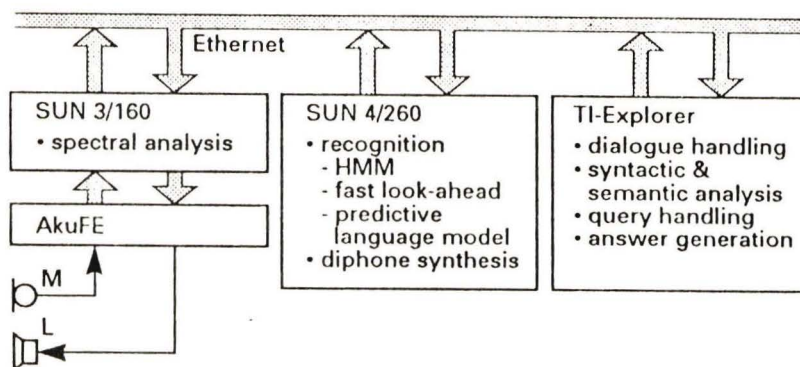


Fig.5 Overview of the present implementation of the SPICOS system. The hardware comprises three computers (SUN, TI) and a specially designed Acoustical Front-End (AkuFE). The computers communicate via an Ethernet-connection. M = microphone; L = loudspeaker; HMM = Hidden Markov Modelling.