MyP3-optager/-afspiller

Audio–kompression og -dekompression

Indledning

I dette projekt skal der konstrueres en koder/dekoder som kan komprimere og dekomprimere digital audio. Vi vil anvende vores eget format, som minder om det format som bruges i MPEG1.

Vi vil med andre ord lave en meget simpel MP3-optager og -afspiller.

Baggrund

MPEG1 audio-standarden, ISO/IEC 11172-3, definerer forskellige metoder til kompression/dekompression af lyd. De 3 metoder kaldes lag 1, lag 2 og lag 3. Lag 1 anvender den simpleste metode og lag 3 anvender den mest komplicerede (og bedste) metode. MPEG1 lag 3 er også kendt under navnet MP3 (se f.eks. [*Davis Pan*]).

Projektets formål er at konstruere en koder/dekoder som arbejder med et format som minder om MPEG1 lag 1. Vi laver vores eget format: MyP3 © HKA[1], for at undgå visse tekniske detaljer i MPEG-formatet. MyP3-formatet kan ikke leve op MP3's kompressions-ratio. Det betyder imidlertid ikke noget, idet formålet med projektet ikke er at lave den mest effektive koder (selvom vi selvfølgelig også vil se på effektiviteten). Formålet er derimod at gå ind og forstå og arbejde med nogle grundlæggende DSP-discipliner.

MyP3-standarden (skitse)

Vores kodning skal være en såkaldt *frekvens-baseret-lossy-kodning*. Vi kalder metoden frekvens-baseret, idet vi starter med at transformere vores lydsignal over i frekvens-området og vi kalder metoden "lossy", idet vi vil kunne tillade os at "smide" information/lyd væk, hvorved det originale signal ikke fuldstændigt kan gendannes (i modsætning til f.eks. pkzip-kompression som jo er et eksempel på en loss-less kompressions-metode). Ved at smide information væk kan vi komprimere vores signal mere end hvis vi vil insistere på at lave en loss-less kompression.
Den information som vi vælger at smide væk skal selvfølgelig vælges med stor omhu. Vi vil kun smide lyd-information væk som ikke kan høres (undersøgelser af, hvad man kan høre og ikke kan høre, kaldes en psykoakustiske model. I skal selvfølgelig eksperimentere med en sådan). Man kan sige, at vi vil bruge mange bits, og dermed stor nøjagtighed, ved frekvenser, hvor vi øret er mest følsomt, mens vi vil lave en grovere kvantisering ved frekvenser, hvor øret er mindre følsomt. Kort kan man sige:

"mange bits" = "god gengivelse"= "lille kvantiseringsstøj"

Mere præcist vil vi lave en blok-vis frekvensanalyse af vores lydsignal. Det fremkomne frekvenssignal opdeles i forskellige frekvensbånd og hvert frekvensbånd kvantiseres med

[1] Hvad mon KarlHeinz Brandenburg kunne have tjent på MP3, hvis han havde ønsket det? – se Jyllandsposten 14. juni 2001

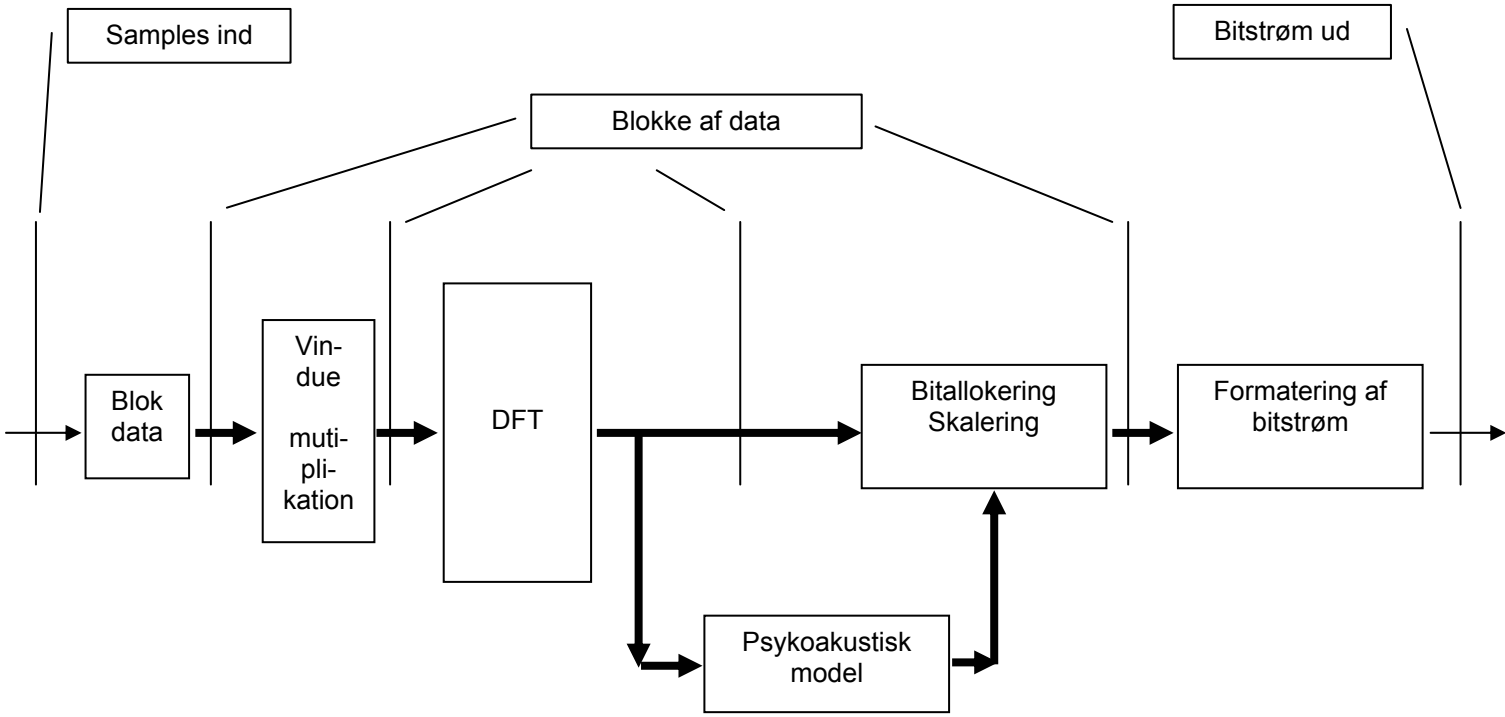
Henrik Karstoft
Version 1.3 august 2003

et passende antal bits (allokationsbits). Desuden kan det være nødvendigt at lave en skalering af værdierne i de enkelte bånd (skaleringsfaktorer). Herefter formateres det kvantiserede frekvenssignal til en bitstrøm (én blok af input-data bliver indkodet i én frame, bitstrømmen bliver nu genereret ved at lægge frames sekventielt efter hinanden).

Dekodningen er den omvendte proces: Bitstrømmen deformateres, således at vi henter de enkelte frames. Hver frame transformeres nu tilbage til tidsområdet og vi har vores lydsignal tilbage igen.

Kvantiseringen i koderen bevirker at det ikke er helt det samme signal som kommer ud af dekompressionen som kom ind i kompressionsrutinen, men har vi gjort tingene rigtigt, er der ingen hørbar ændring.

Koderen kan skitseres som vist nedenfor:



Frameformat

MyP3 formatet af en frame af data, som er indkodningen af alt i alt 512 samples af 16 bit (8192 bits af det ukodede signal), har følgende form:

Header (12 bits)	Allokationsbit (25*4 = 100 bits)	Skaleringsfaktorer (25*4 = 100 bits)	Kvatiserede frekvensdata

Vi lægger os desuden fast på følgende:

1. Vi vil kun kode et mono-signal.

2. Vi vil kun arbejde med én samplingsfrekvens, nemlig 44.1kHz.

3. Vores input samples ligger i intervallet [-1; 1].

4. Koderen/dekoderne arbejder på blokke med længde 512-samples som indkodes til 1 frame.

5. Vi arbejder med 25 frekvensbånd i området 0 til 20kHz (se nedenfor). Hvert bånd kvantiseres med 0 til 16. Vi vil anvende følgende tabel:

Bit-mønster	Antal bits som bruges til indkodning
0000	0
0001	2
0010	3
0011	4
…	…
1110	15
1111	16

Bemærk, springet mellem første og anden række.

6. Vi vil arbejde med 2^6 mulige skaleringsfaktorer, idet vi bruger følgende tabel:

Bit-mønster	Skaleringsfaktor
0000	2^0
0001	2^1
0010	2^2
0011	2^3
…	…
1110	2^{14}
1111	2^{15}

Alle komplekse tal i et bånd skaleres med den samme skaleringsfaktor. Skaleringsfaktoren vælges således at alle real-dele og imaginær-dele ligger mellem -1 og 1, efter multiplikation med skaleringsfaktoren. Skaleringsfaktoren vælges så stor som muligt.

Eksempel
Hvis den maximale værdi af real-delene og imaginær-delene i et frekvensbånd er 0.0558, anvender vi skaleringsfaktoren 2^4 idet vi herved opnår at:
$0.0558*2^4 = 0.8933 < 1$ (Vi kan ikke vælge 2^5 , thi $0.0558*2^5 = 1.7866$)

7. Headeren er #HFFF (sync-word).

8. En frame skal altid bestå at et helt multiplum af 16 bit (eventuelt 0-padding indtil dette er opfyldt, => højst tilføjelse af 15 bit).

9. De skalerede komplekse tal lægges med real-delen først og herefter imaginær-delen. Tallene lægges således at de tal som hører til de mindste frekvenser kommer først. Tallene lægges i 1.N-format hvor 1+N er antallet af bit som bruges til indkodningem jvf. tabel. Hvis allokationsbittene er 0000, gemmes *ingen* værdier fra det pågældende frekvensbånd.

10. I indkoderen vil vi anvende en skaleret DFT, dvs. vi vil beregne DFT vha. formlen:

$$X(k) = \frac{1}{512}\sum_{n=0}^{511} x[n]e^{-j(\frac{2\pi}{512}k)n}$$

Andre krav til koderen:

1. Blokkene har et 50% overlap og vi vil bruge et Hanningvindue.

2. De kodede data lægges i en fil, vi vil hermed ikke arbejde med egentlige bitstrømme.

Projektbeskrivelse

1. Færdiggørelsen af MyP3-standarden, uklarheder eller udeladelser i standarden klarlægges (en afklaring-af/tilføjelse-til standarden kræver selvfølgelig en afdækning af mulige alternativer og herefter et kvalificeret valg).

2. Koderen realiseres i et højniveau programmeringssprog, f.eks. C++ eller MatLab. I kan arbejde mere eller mindre med optimeringen af den psykoakustiske model som bruges ved indkodningen, herved fremkommer der flere ”niveauer” for en løsning (overvej, hvad jeres 80%, 100% og 120% løsning skal være).

3. Dekoderen realiseres først i et højniveau programmeringssprog, f.eks. C++ eller MatLab.
   Eventuelt laves en realisering på EZ-kit-lite boardet. Det er en selvfølge, at dekoderen testes ordentligt (f.eks. lyttetest i lytterum osv.).

4. Jeres gruppe kan prøve at bytte koder/ dekoder med en anden gruppe. Prøv også at sammenligne jeres koder/dekoder, med en ”rigtig” MP3-koder/ -dekoder.

5. Prøv at anvende jeres koder/decoder på vores reference CD (udleveres af jeres vejleder). Mål koder/decoder ”preformance”.

6. Kom med kvalificerede forslag til forbedringer af MyP3-standarden (hvis der er tid).

Generelt

- Projektet er et undersøgelsesprojekt . I skal således ikke koncentrere jer om fine brugergrænseflader og smarte betjeningsdetaljer. I stedet skal i arbejde med at udtænke og realisere en prototype på det ovenfor beskrevne system.

- I forbindelse med opgaverne ovenfor vil det være en god ide at gøre jer klart, hvad I vil acceptere som 80%, 100% og 120% løsninger af (del-)opgaver.

- Det er ikke meningen at I skal lave en rigtig MP3-koder/-dekoder, derfor skal I passe på med at læse alt for meget om netop denne standard, det vil muligvis kunne give anledning til nogen forvirring.

Litteratur

Der er en sand velsignelse af litteratur, bøger, artikler, web-sider osv. om emnet. Jeg har valgt nogle enkelte ud:

- Davis Pan: "A Tutorial on MPEG/Audio Compression". IEEE Multimedia Vol. 2, No. 7, 1995, pp. 60-74. (Lidt svær, men alligevel en god intro til MPEG1-kompression)

- Torben Poulsen: ”Lydopfattelse”,Note nr. 2108 fra DTU

- http://www.tnt.uni-hannover.de/project/mpeg/audio/faq/mpeg1.html#1-7

Frekvensbånd

Vi vil anvende nedenstående frekvensbånd (bemærk, at tabellen er baseret på et 512-DFT og en samplingsfrekvens på 44.100Hz):

Bånd nr	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Første bin	0	2	3	4	5	6	7	9	11	13	15	17	20	23	27	32	37	43	52	62	75	90	111	140	180
Sidste bin	1	2	3	4	5	6	8	10	12	14	16	19	22	26	31	36	42	51	61	74	89	110	139	179	256
Båndbredde [bins]	2	1	1	1	1	1	2	2	2	2	2	3	3	4	5	5	6	9	10	13	15	21	29	40	77

Grænse frekvens	0	129	215	301	388	474	560	732	904	1077	1249	1421	1680	1938	2283	2713	3144	3661	4436	5297	6417	7709	9518	12016	15461	22050

Henrik Karstoft, Århus, august 2003