Ukeoppgaver til uke 18 INF2310, våren 2010
Kompresjon og koding – del II

1. En fax-oppgave

a. Et ark med tekst og enkle strektegninger skal sendes pr digital fax over en modemlinje med kapasitet på 4 800 biter/sekund. Vi bruker en standard fax med 1728 fotosensorer per linje og 1075 linjer per side. Faxmaskinen gjør en terskling av bildet av siden (slik at du sitter igjen med et binært bilde kun med pikselverier 0 og 1). Hvor lang tid tar det å overføre en side uten kompresjon?

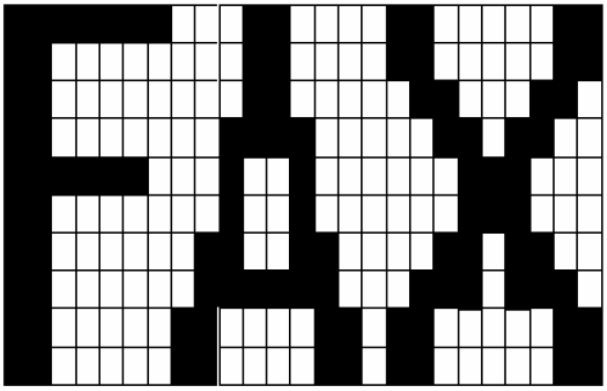
b. Anta at vi hadde kunnet gjøre tekstgjenkjenning på den delen av arket som inneholder tekst, og representert symboler og mellomrom med 7 bits ASCII. Anta at det maksimalt er 60 tegn pr linje og 50 linjer pr side. Anta også at vi kunne beskrevet strektegningene som maksimalt 500 rektangler per side, og at sidene på rektanglene er parallelle med kantene på siden. Gi et "worst case" estimat av hvor mange biter du vil trenge for å beskrive innholdet på siden med en oppløsning som svarer til faxens oppløsning, og hvor lang tid det vil ta å overføre dette over modemlinjen.

c. Vi vil gjerne undersøke hvor mye det er å spare på å separere ASCII-tegn fra alt annet i en fax, og sende 7 bits ASCII kode for hvert tegn, mens resten sendes ukomprimert – uansett hva det er. Hvis halvparten av hver side igjennomsnitt er ASCII-tegn, hvor mye sparer vi da i forhold til ordinær fax?

2. Huffman-koding av løpelengder i binært bilde.

Utsnittet på 25 * 10 piksler av et binært bilde nedenfor kan representeres med 250 biter. Ser vi på runlength-representasjonen av det samme utsnittet, finner vi at det består av 82 "runs" med lengder mellom 1 og 8 piksler. Hvis vi bruker 3 biter på hver, blir dette 246 biter. Imidlertid er det mulig å gjøre dette litt mer kompakt ved å Huffman-kode de 82 løpelengdene. Ved løpelengdetransformasjon av binære bilder trenger vi ikke å lagre tallpar (gråtone, løpelengde) slik som for gråtonebilder. Vi trenger bare løpelengdene, for det er bare to mulige intensitetsverdier.

Løpelengdene finnes i tabellen til høyre.



7 3 2 4 2 5 2
2 8 2 4 2 5 2
2 8 2 5 2 3 2 1
2 7 4 5 2 1 2 2
6 3 1 2 1 6 3 3
2 7 1 2 1 6 3 3
2 6 2 2 2 4 2 1 2 2
2 6 6 3 3 1 3 1
2 5 2 4 2 1 2 5 2
2 5 2 4 2 1 2 5 2

Finn Huffmann-koden til løpelengdene i tabellen til høyre over, og finn det totale antall biter etter koding av løpelengdene.

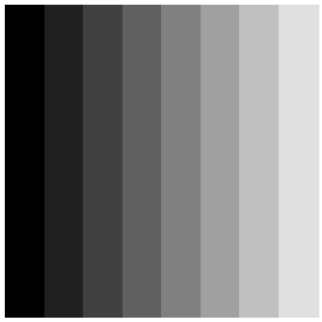
3. Teorioppgave: Løpelengdekoding i binært bilde med naturlig binærkode

Du skal gjøre en løpelengde (”run-length”) transform på et $2^n \cdot 2^n$ piksels binært bilde. Anta at du gjør dette linje for linje i bildet, ved å angi første pikselverdi, deretter løpelengdene, og verdien 0 to ganger etter hverandre som EOL-markør. Anta også at du bruker en felles naturlig binærkode for både pikselverdiene og løpelengdene.

a. **Finn et uttrykk for det høyeste antall løpelengder, N, som du med disse forutsetningene kan ha i en linje i bildet hvis løpelengde-transformen skal gi noen kompresjon i forhold til det binære bildet?**
b. **Hva blir den høyeste verdien av N for hhv. n = 4, n = 8 og n = 10? Er forholdet mellom det maksimale antall løpelengder vi kan ha og fortsatt oppnå kompresjon, og bredden av bildet konstant etter hvert som vi øker størrelsen på bildet?**

4. Transformasjoner og kompresjon av et gråtonebilde.

a. **Anta at vi har et 512×512 piksels gråtonebilde med 8 bitplan. Pikselverdien er 0 langs venstre kant av bildet, og øker med 32 i jevne ”trappetrinn” mot høyre, slik at det dannes 8 vertikale striper som vist i figuren nedenfor.**



Hvor mange biter vil vi måtte bruke per linje hvis vi løpelengde-transformerer dette gråtonebildet og bruker en felles naturlig binærkode for både pikselverdier og løpelengder, og bruker verdien 0 to ganger etter hverandre til å indikere slutten av en linje (EOL) ?

b. Vis kodetreet og finn kodeboken for en Huffman-koding av resultatet av løpelengde-transformen ovenfor. Anta fortsatt at vi bruker (0 0) til å indikere EOL.

c. Finn en omtrentlig verdi for det gjennomsnittlige antall biter per piksel (i det opprinnelige bildet) når du bruker denne Huffman-koden.
Angi også den omtrentlige kompresjonsfaktoren.

5. Praktisk sammenligning av JPEG og JPEG2000

Programpakken ENVI er en pakke for bildeanalyse av satellittbilder som vi har Linux-lisenser på. Den startet fra linux ved å skrive ”envi”.

a. Les inn bildet ¨¨¨¨~inf2310/bilder/mona.png:
Start envi, velg File->Open External File->Generic Formats->PNG
b. Vis fram bildet ved å åpne display:
Fra Available Bands List, velg filen, så New Display og Load Band
Nå får du opp bildet i 2-3 zoom-vinduer, velg det med File-meny, fra File-menyen velg Save Image As ->Image File ->…. Output file type JPEG eller JPEG 2000. Dette gir deg ulike valg for kompresjonsgrad og kompresjon med eller uten tap.

c. Prøv å lagre bildet med JPEG med ulike kompresjonsgrad. Les så bildet inn igjen og studer bildekvalitet.
d. Lag et bilde av rekonstruksjonsfeilen mellom original og komprimert bilde. I ENVI kan du gjøre dette ved å velge fra hovedmenyen : Basic Tools -> Band Math.
I feltet for band math expression kan du for eksempel skrive
sqrt((float(b1)-float(b2))^2)
Du vil bli bedt om å velge b1 og b2 fra listen over filer (eller evt. fra fil).
Studer bildet av rekonstruksjonsfeilen for ulik grad av kompresjon.
e. Prøv å lagre bildet som JPEG 2000 med ulike valg av kompresjonsgrad. Studer det komprimerte bildet og rekonstruksjonsfeilen for å sammenligne JPEG mot JPEG 2000.