Toteutusdokumentti Pihla Toivanen

Ohjelmani koostuu kolmesta isommasta osa-alueesta:

1. RGB-tiedoston luku ja erottelu 8*8 -blokkeihin sekä muu alustus.

- Separate(): Luetaan .rgb -tiedosto ja erotellaan se blocks[][][][] -taulukkoon. Kehitin indeksointiin systeemin, jossa 1. ulottuvuus on 8*8-blokin indeksi, 2 seuraavaa ulottuvuutta 8*8 -blokin indeksit ja neljäs ulottuvuus on tarkoitettu kolmea RGB -arvoa varten.
- DecreaseBlocks() ja increaseBlocks() ovat algoritmiin kuuluvia lisäys -ja vähennysfunktioita.
- ConvertToRgb ja convertToYCbCr kääntävät rgb-arvoja YcbCr -arvoiksi ja toisin päin.

2. Diskreetti kosinitransformaatio ja käänteistransformaatio
   - doForBlocks(): yleinen pohja, jota Main kutsuu
   - DCT tekee diskreetin kosinitransformaation. Tein ensin transformaation ilman dynaamista ohjelmointia eli kosiniarvot laskettiin aina uudelleen. Sitten tein dynaamisen ohjelmoinnin taulukon (giveDpvalue -metodi), joka tallentaa ja ottaa aiemmin laskettuja arvoja käyttöön.
   - ApplyIDCT, käänteinen transformaatio
   - kvantisointi ja dekvantisointi: kerrotaan/jaetaan speksissä määritetyllä kvantisointimatriisilla.

3. Huffman-koodaus

Tein muutoin "perus-Huffmanin", paitsi että en tallenna nollia ollenkaan (transformoidusta kuvadatastani parhaimmillaan 92% nollia esim. kokonaan mustassa kuvassa). Tallennan siis vain nollasta eroavat arvot ja jokaista ennen, kuinka monta nollaa datassa on ennen sitä.

- Frekvenssien laskeminen
- Trie-rakenne
- Tiedostoonkirjoituksessa käytän kirjastotiedostoa bittien bufferointiin.
- Implementoin minimikeon (PQ.java) itse käyttäen valmista ArrayListiä.

Lisäksi ohjelmaan kuuluu: Swing-moduuli, joka näyttää kuvadata-arrayn kuvana, sekä (tulee kuulumaan...) terminaalikäyttöliittymä.