Määrittelydokumentti

Pekka Väänänen

13. Toukokuuta 2013

Aiheen kuvaus

Ongelma

Pakkaamaton kuvadata vie suuremmalla kuvakoolla runsaasti tilaa, joten sen pakkaaminen on usein tarpeen.

Harjoitustyön aihe on JPEG-pakkauksen kaltainen kuvanpakkausalgoritmi. Tarkemmin ilmaistuna kyseessä on JFIF-pakkauksen muunnos.

Algoritmi

1. Kuva muutetaan RGB-väriavaruudesta YCbCr-väriavaruuteen, koska tässä väriavaruudessa värin kirkkaus ja sävy on eritelty omiin värikanaviinsa. Pikselien kirkkaus voidaan täten tallentaa korkeammalla tarkkuudella kuin niiden värisävy.
2. Kuva jaetaan 8x8 pikselin kokoisiin ruutuihin, ja jokaiselle värikanavalle suoritetaan diskreetti kosinimuunnos (DCT). Jokaisen pikselin värisävy (chroma-komponentit) tallenetaan matalammalla tarkkuudella kuin kirkkaus (luma-komponentti).
3. Ruutujen taajuusinformaatio kvantisoidaan siten, että korkeiden taajuuksien erottelutarkkuus laskee eniten.

4. Aikaisempien askeleiden lopputulos pakataan häviöttömällä algoritmilla, eli tässä tapauksessa Huffman-koodauksella.

Käytettävät algoritmit

- DCT (DCT-II) ja IDCT (DCT-III)
- Huffman-koodaus

DCT-algoritmi suorittaa lineaarisen muunnoksen diskreetistä syötedatasta taajuusesitysmuotoon, jossa signaali esitetään eri taajuksilla värähtelevien kosinien summana. Algoritmista on olemassa erilaisia muunnoksia, mutta tässä harjoitustyössä käytän DCT-II:sta, joka muuttaa datan taajuusmuotoon. Käytän myös vastaavaa käänteismuunnosta DCT-III:sta, joka muuttaa DCT-II:n laskemat taajuuskertoimet takaisin diskreetiksi signaaliksi.

Huffman-koodauksessa syötedatan symboleista muodostetaan binääripuu, jossa harvoin esiintyvät symbolit päätyvät puun lehtiin. Symbolien binääriesitys

muodostetaan niiden sijainnin mukaan, jolloin useimmin esiintyville tapauksille valitaan lyhyempi esitysmuoto.

Tavoitteet

Aikavaativuus

Tavoitteenani on päästä algoritmien standardiratkaisujen aikavaativuuksiin. En siis käytä esim. DCT:n laskemiseen Fast Fourier Transformista johdettua $O(N \log N)$-versiota.

DCT-II:n aikavaativuus on $O(N^2)$, sillä jokaisen pakattavan näytteen kohdalla on käytävä läpi myös jokainen muukin näyte. Sama pätee DCT-III:lle. Huffman-koodauksen aikavaativuus on $O(N \log N)$, sillä aluksi koko syötedata pitää käydä kerran lävitse eri symboleiden esiintymistiheyksien selvittämiseksi, mutta varsinaisessa koodauksessa haetaan arvoja binääripuusta. Näytteille suoritetaan myös muita operaatioita, mutta niissä ei tarkastella muita näytteitä, joten tämän prosessoinnin aikavaativuus on $O(N)$.

Tilavaativuus

Algoritmin suorituksessa on luonnollisesti pidettävä muistissa pakattava kuva kokonaisuudessaan, mutta tätä ei lasketa mukaan tilavaativuuden O-analyysiin.

Syöte muutetaan toiseen väriavaruuteen, ja erilliseksi taajuusesitykseksi, joten tilavaativuus on näiden osalta $O(2N)$. Myös Huffman-puu on pidettävä muistissa, mutta se vie tilaa vain vakiomäärän. Lopullinen pakattu kuva säilötään muistiin myös, ja pahimmillaan se voi viedä muistia yhtä paljon (tai hieman enemmän) kuin alkuperäinen syöte. Algoritmin tilavaativuus on siis $O(N)$.

Lähteet

1. Hamilton (1992), JPEG File Interchange Format http://www.w3.org/Graphics/JPEG/jfif3.pdf, s. 3-4

2. (1992) JPEG Standard (JPEG ISO/IEC 10918-1 ITU-T Recommendation T.81) http://www.w3.org/Graphics/JPEG/itu-t81.pdf, s. 15, 27-28, 87-89

3. Haberdar (2012), Discrete Cosine Transform Tutorial http://www.haberdar.org/Discrete-Cosine-Transform-Tutorial.htm

4. Abelson, G. Sussman, J. Sussman (1990) Structure and Interpretation of Computer Programs http://mitpress.mit.edu/sicp/full-text/sicp/book/node41.html#fig:huffman - Example: Huffman Encoding Trees