TEKNIIKKA JA LIIKENNE

Tietotekniikka

Tietoliikenne

INSINÖÖRITYÖ

H.264-koodekin soveltuminen IPTV-järjestelmään

Työn tekijä: Joonas Terhivuo
Työn ohjaajat: Jouko Kurki, Jari-Pekka Hela-Aro

Työ hyväksytty: ___. ___. 2009

Jouko Kurki
Yliopettaja

Metropolia

TIIVISTELMÄ

Työn tekijä: Joonas Terhivuo

Työn nimi: H.264-koodekin soveltuminen IPTV-järjestelmään

Päivämäärä: 9.12.2009 **Sivumäärä:** 46

Koulutusohjelma: Tietotekniikka

Suuntautumisvaihtoehto: Tietoliikenne

Työn ohjaaja: Yliopettaja Jouko Kurki

Työn ohjaaja: Kehityspäällikkö Jari-Pekka Hela-Aro

Tämä työ tehtiin Maxisat-yhtiöt Oy:lle. Työssä selvitetään H.264-koodekin: rakenne, toimintaperiaate, koodaustyökalut, suorituskyky ja soveltuvuus eri sovelluksille. H.264 on uusin Joint Video Team-työryhmän työstämä kansainvälinen videonkoodausstandardi. Sen tarkoitus on syrjäyttää MPEG-2-standardi teräpiirtotelevision tallennus-, lähetys- ja jakeluapuvälineenä. H.264 tarjoaa myös tehokkaan koodaustyökaluvalikoiman uusille sovelluksille kuten mobiilivideolle ja IPTV:lle.

Työn teoriaosassa perehdytään ensin H.264-standardiin ja yleisesti lohkopohjaisissa standardeissa oleviin pakkausmenetelmiin. Tämän jälkeen esitellään H.264-koodekin rakenne ja koodekin mahdollistamat eri koodaustyökalut. H.264-koodekista tekee paremman muihin koodeikkeihin nähden sen sisältämät parannukset koodaustyökaluissa. Sen jälkeen käydään läpi H.264-standardin asettamat rajaukset. Rajaukset asetetaan standardissa profiilien ja tasojen avulla. Rajaukset mahdollistavat H.264-koodekin soveltumisen mahdollisimman hyvin monelle eri sovellukselle.

Lopuksi verrataan H.264-koodekin suorituskykyä muihin koodekkeihin, IEEE:n tutkijoiden testituloksien ja oman testintuloksien avulla. SD-videolla tehdyistä tuloksista selviää, että H.264-koodekki pystyy pienillä alle 4 Mbps bittinopeuksilla, jopa puolet pienempään kompressointisuhteeseen kuin MPEG-2:n. H.264-koodekki päihittää myös muilla bittinopeuksilla kilpailevat koodekit. Hyvän kompressointi kykynsä takia H.264-koodekki soveltuu erinomaisesti IPTV-järjestelmään. H.264:n voidaan olettaa valtaavan myös lisää jalansijaa muissa sovelluksissa.

Avainsanat: H.264, Koodekki, MPEG-4 part 10, MPEG-4 AVC

Metropolia

ABSTRACT

Name: Joonas Terhivuo

Title: H.264 Codec Suitability to IPTV Systems

Date: 9.12.2009 **Number of pages:** 46

Department: Information Technology

Study Programme: Telecommunications

Instructor: Principal Lecture Jouko Kurki

Instructor: Development Manager Jari-Pekka Hela-aro

This study was carried out for Maxisat-yhtiöt Oy. In this study, H.264 codec structure, operating principle, coding methods, performance and suitability to various systems was examined. H.264 is the newest international video coding standard produced by Joint Video Team. The main goal of the H.264 standardization is to replace MPEG-2 standard in High-definition television broadcasting and storage services. H.264 also offers efficient coding methods for new systems like mobile video and Internet Protocol Television.

In the theoretical part of this study the focus is first on H.264 standard and coding methods which are in common use in video coding standards. After that, this study presents the H.264 codec structure and different coding methods in H.264. H.264 codec is better than other existing coding methods due to significant improvements in coding methods. Thereafter, this work discusses the H.264 profiles and levels confining the H.264. Using confines in video coding enable the suitability of H.264 codec to various systems.

Finally, the differences in performance between H.264 codec and other existing codecs were examined. The comparison was based on test results obtained by IEEE researchers and the test results of this study. The test results show clearly that H.264 codec can compress video even more that 50% better than the MPEG-2 codec in low bit rates under 4 Mbps. Also for other bit rates H.264 codec outperforms other existing video codecs. Because of effiency video compression H.264 codec fits excellently for IPTV system. Thus, we can assume that H.264 is going to gain a bigger foothold in various systems in the near future.

Keywords: H.264, Codec, MPEG-4 part 10, MPEG-4 AVC

SISÄLLYS

LYHENTEET

AVCHD	Advanced Video Codec High Definition. Sonyn ja Panasonicin kehittämä tallennusformaatti teräväpiirtovideolle.
CACAB	Context-based Adaptive Binary Arithmetic Coding. Aritmeettinen entropiakoodausmuoto, jota käytetään H.264:ssä.
CAVLC	Context-based Adaptive Variable Length Coding. Vaihtelevan pituinen entropiakoodausmuoto, jota käytetään H.264:ssä.
CIF	Common Intermediate Format. Video jonka resoluutioksi on standardoitu 352x288.
DCT	Discrete Cosine Transform; diskreetti kosinimuunnos.
DVB-H	Digital Video Broadcasting - Handhelds. Digitaalisen televisiolähetyksen standardi kännykälle
DVD	Digital Versatile Disc tai Digital Video Disc. Optinen datan tallennuslevyke.
FMO	Flexible Macroblock Ordering. Joustava makrolohkojoukkojen käsittelytapa, joka on käytössä H.264:ssä.
GOP	Group Of Pictures. Määrää videossa missä järjestyksessä kuvat tulevat.
HDTV	High-Defination Television; teräväpiirtotelevisio.
IEC	International Electrotechnical Commission. Kansainvälinen sähköalan standardointiorganisaatio.
IEEE	Institute of Electrical and Electronics Engineers. Kansainvälinen tekniikan alan järjestö.
IPTV	Internet Protocol Television. IP-verkon yli lähetetty televisiokuvan tekniikka.
ISO	International Organization for Standardization. Kansainvälinen standardisoimisjärjestö.
ITU-T	International Telecommunication Union. Kansainvälisen televiestintäliitto ITU:n televiestintäsektori.
JVT	Joint Video Team; standardointiryhmä.
MPEG	Moving Picture Experts Group. Audion- ja videonpakkaamisen standardointi ja asiantuntijatyöryhmä.
MPEG-1	MPEG-työryhmän julkaisema standardi.
MPEG-2	MPEG-työryhmän julkaisema standardi.
MPEG-4	MPEG-työryhmän julkaisema standardi.
NAL	Network Abstraction Layer. NAL-kerros määrittelee H.264:n verkkosovituksen.

PAL	Phase Altermate Line. Euroopassa käytössä oleva videokuvan värijärjestelmä ja koodausmenetelmä.
PSNR	Peak Signal-to-Noise Ratio. Videokuvanlaadun arvioinnissa käytettävä arvo.
ROI	Region Of Interest. Arvo, joka kertoo koodekille ja dekoodekille suorakaiteen vasemman ylänurkan ja oikean alanurkan arvon.
VCEG	Video Coding Experts Group. ITU:n perustama työryhmä.
VCL	Video Coding Layer. H.264-videonkoodausta käsittelevä kerros.

1 JOHDANTO

Digitaaliseen videoon siirtyminen analogisesta videosta 1990-luvun loppupuolella aloitti digitaalisen videonpakkaamisen kehittymisen. Digitaalinen video sisältää paljon ylimääräistä informaatiota, jota siitä poistetaan videonpakkaamisen avulla. Lisäksi eri sovellukset vaativat erilaista videota, joka mahdollistaa videon kompressoinnin eri vaatimuksin.

H.264 on videonpakkausstandardi, joka tunnetaan myös nimellä MPEG-4 AVC (Moving Picture Experts Group - 4 Advanced Video Coding) ja MPEG-4 part 10. Se on tämän hetken kehittynein videonpakkausstandardi ja se on alkanut korvata muita aikaisempia standardeja monilla eri osa-alueilla. Standardi on alkanut yleistyä, koska se on tehokkaampi videonpakkaaja kuin muut tämän hetken standardit. H.264 pystyy pakkaamaan samanlaatuista videokuvaa, jopa puolet pienempään tilaan kuin muut videonpakkausstandardit. [10, s. 159.]

Eri sovellukset vaativat erilaista videota, joten videota voidaan kompressoida eri vaatimuksin. Myös reaaliaikaisen videon siirtäminen langattomasti olisi mahdotonta ilman videonpakkaamista. Esimerkiksi mobiilitelevision reaaliaikainen katsominen kännykkällä ilman videonpakkaamista. Mobiilitelevisiossa eli DVB-H:ssä (Digital Video Broadcasting - Handheld) on käytössä 320x240-resoluutioinen video, jonka päivitysnopeus on 15 kuvaa/s. Jokaisen pikselin väri muodostetaan kolmen päävärin: punaisen, vihreän ja sinisen avulla. Jos jokainen väri kuvattaisiin 8:lla bitillä, tulisi bittinopeudeksi 320x240x15x3x8 eli noin 27,7 Mbps. Tällaisen videon siirtäminen nykyisessä mobiiliverkossa on mahdotonta. Vaikka video saataisiin siirtymään verkon yli, ei kännykän laskentateho riittäisi videon käsittelyyn. H.264:sta käytettäessä videonpakkaamiseen mahdollistetaan siirtonopeus sellaiseksi, että se saadaan siirrettyä kännykkään ja kännykkä pystyy sen käsittelemään.

Toisena esimerkkinä mainittakoon, että H.264:ää käytetään myös HD-videossa (High Defination) eli teräväpiirtovideossa. HD-videoksi luokitellaan video, jonka pystysuuntainen erottelukyky on vähintään 720 pikseliä. Yleisesti käytettyjä resoluutioita HD-videossa on 1280x720, 1440x1080 ja 1920x1080 pikselin resoluutio. Jos kuvan päivitysnopeus olisi 25 kuvaa/s ja värit esitettäisiin 8:lla bitillä, saataisiin bittinopeuksiksi noin 0,55 Gbps, 0,93

Gbps ja 1,24 Gbps. DVD-levylle mahtuu noin 4,7 Gt dataa ja Blu-ray-levylle noin 25 Gt dataa. HD-elokuvan tallentaminen levylle ilman videonpakkaamista olisi mahdotonta, koska muisti ei riitä levyllä.

Teräväpiirtovideon tallentamista varten on luotu tallennusformaatti AVCHD (Advanced Video Coding High Defination). AVCHD on Sonyn ja Panasonicin yhdessä kehittämä tallennusformaatti, joka käyttää H.264-koodekkia videonpakkaamiseen. AVCHD-formaatti mahdollistaa teräväpiirtoelokuvan tallentamisen Bluray-levylle. [4.]

Myös nopean laajakaistan(yli 1Mbps laajakaista) leviäminen on tuonut uusia mahdollisuuksia jakaa videota. Niitä ovat esimerkiksi streaming-videopalvelut ja IPTV.

Aikaisemmin eri sovelluksissa on käytetty paljon MPEG-2-videonpakkausstandardia. Se on kuitenkin jo yli 10 vuotta vanha standardi, eikä se sovellu kovin hyvin nykyaikaisille sovelluksille. H.264-standardi kehitettiin syrjäyttämään MPEG-2:n ja avaamaan samalla uusia mahdollisuuksia sovelluksille, joita ei MPEG-2-videonpakkausstandardia käyttämällä pystytä tarpeeksi hyvin toteuttamaan.

Tässä työssä keskitytään H.264-koodekkiin. Ensin perehdytään miksi videonkoodausta tarvitaan ja miten H.264-standardi on saanut alkunsa. Tämän jälkeen käydään H.264-standardia läpi, jonka jälkeen käsitellään itse H.264-koodekki ja koodaustyökalut. H.264:ssä asetetaan myös erilaisia määrityksiä eri sovelluksia varten, mikä tapahtuu H.264-standardin profiilien ja tasojen avulla. Näiden läpikäynnin jälkeen perehdytään itse H.264-koodekin suorituskykyyn. Koodekin suorituskykyä tutkitaan vertaamalla itse tekemäni testintuloksia IEEE:n tutkijoiden tekemien testien tuloksiin.

H.264-standardissa on tehty jako kuvainformaation pakkaamiseen ja sen erilaisiin sovittamisiin eri verkkojen välillä. Tässä työssä käsitellään vain videonkoodausta käsittelevän VCL-kerroksen (Video Coding Layer) määrityksiä. Verkkosovituksia tukevan NAL-kerroksen (Network Adaption Layer) asiat jätetään kokonaan työn ulkopuolelle. [7, s. 11-12.]

2 H.264-STANDARDI

Kaikki kansainvälisesti merkittävät videonkoodausstandardit on kehitetty kahdessa eri työryhmässä: VCEG- (Video Coding Experts Group) ja MPEG (Moving Picture Experts Group)työryhmässä. VCEG on ITU (International Telecommunication Union) organisaation perustama työryhmä, joka julkaisee suosituksia, jotka käsittelevät videonkoodausta. Videonkoodausta koskevat suositukset ovat nimeltään H.26x. MPEG-työryhmä on ISO:n (International Organization for Standardization) ja IEC:n (International Electrotechnical Commission) perustama. MPEG-työryhmän julkaisemat suositukset videonkoodausta varten kulkevat nimellä MPEG-x. [8, s. 7.]

Nämä kaksi työryhmää ovat 1990-luvun kuluessa julkaisseet mm. seuraavia videonkoodausstandardeja: [16; 17.]

- H.261, julkaistiin 1990 kohdesovelluksena ISDN-video
- MPEG-1, julkaistiin 1993 CDROM-videota varten
- H.262/MPEG-2, julkaistiin 1994 TV-tasoista videota varten
- H.263, julkaistiin 1995 videoneuvotteluja varten
- MPEG-4 part 2, julkaistiin 1999 internet-videota varten
- H.264/MPEG-4 part 10, julkaistiin 2003 ja laajennuksia standardiin on julkaistu 2005 ja 2007.

2000-luvulle tultaessa markkinoille oli syntynyt paljon uusia sovelluksia: kännyköille suunniteltu video 3G-verkon kautta, laajakaistaisen internetin mahdollistama laadukkaiden videoiden katselu internetistä, digitaaliset HDTV-lähetykset. Olemassa olevat koodekit eivät kuitenkaan ominaisuuksiltaan kunnolla riittäneet uusille sovelluksille. Koodauslaitteistojen jatkuvasti kasvava laskenta- ja tallennusresurssien määrä mahdollisti paljon tehokkaampien koodausmenetelmien käyttöönoton. VCEG- ja MPEG-työryhmä päättivät kehittää yhdessä uuden kansainvälisen videonkoodausstandardin uusia sovelluksia palvelemaan. He perustivat kehitystyötä varten uuden yhteisen työryhmän JVT:n (Joint Video Team). JVT-työryhmä alkoi kehittämään H264-standardia ja asetti sille mm. seuraavia tavoitteita:

- yksinkertainen rakenne
- hyvä kompressointikyky: pyrkimys kaksinkertaiseen pakkaussuhteeseen parhaisiin aiempiin videonkoodausstandardeihin verrattuna
- hyvä virheensietokyky
- soveltuvuus erilaisiin verkkoihin
- skaalautuvuus.
- joustava soveltuvuus eri sovelluksien viivevaatimuksiin.

Kaiken kaikkiaan tavoitteena oli päivittää videonkoodausstandardointi nykyiselle vaatimustasolle ja tuoda markkinoille mahdollisimman yleiskäyttöinen ja teknisesti korkeatasoinen koodekki. Työ aloitettiin 2001-vuoden lopussa, ja valmiiksi standardi hyväksyttiin keväällä 2003. Tämän jälkeen H.264-standardiin on tullut laajennuksia vuonna 2005 ja 2007.

3 H.264-KOODEKKI (ENCODER)

Videokuva koostuu yksittäisistä kuvista, joita näytetään tarpeeksi nopeasti peräkkäin. Ihminen näkee kuvasarjan liikkuvanakuvana eli videona. Kuvasarjan ensimmäinen kuva on avainkuva. Toinen kuva on erotuskuva ensimmäiseen nähden ja kolmantena erotuskuva toiseen nähden jne. Videonsiirrossa voi kuitenkin tapahtua virheitä, joten avainkuva täytyy uusia säännöllisin väliajoin (n. 0,5 - 3 sekuntia). Uuden avainkuvan jälkeen tulee taas erotuskuva siihen nähden. Näin videokuva on jaettu kuvaryhmiin. Kuvaryhmässä on ensimmäisenä avainkuva ja sitten tietty määrä erotuskuvia. Tätä kuvaryhmään jakoa kutsutaan GOP:ksi (Group Of Pictures), joka esim PAL-videossa on normaalisti 12 kuvaa.

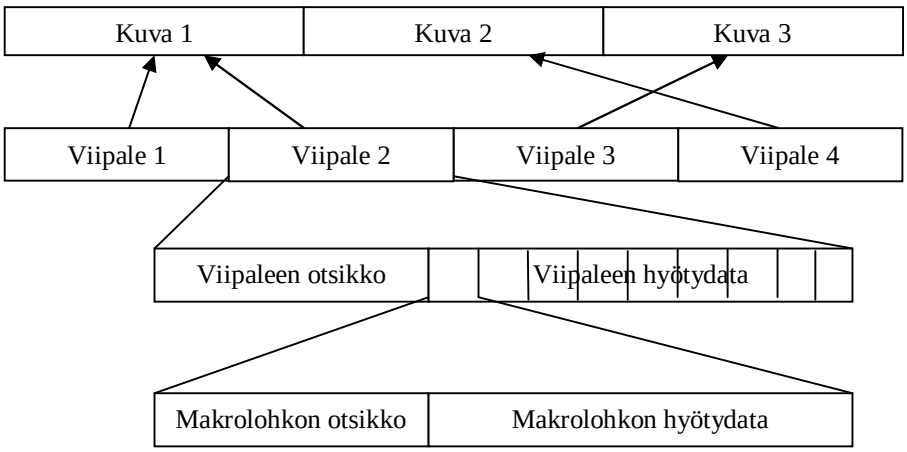
Peräkkäisissä kuvissa on paljon samankaltaisia objekteja hieman eri paikoissa. Esimerkiksi avainkuvaan nähden erotuskuvissa ei ole muuta eroa kuin ihmisen käden liikkuminen. Peräkkäisissä kuvissa ei kokonaisen erotuskuvan lähettäminen olekaan järkevää, vaan kannattaa tutkia mihin suuntaan objektit ovat liikkuneet kuvassa. Peräkkäisistä kuvista pyritään etsimään liikevektoreita, jotka kertovat mihin suuntaan objekti on liikkunut ja kuinka paljon. Tätä etsimistä kutsutaan liike-ennustamiseksi.

Ennustaminen mahdollistaa, että erotuskuvan sijaan lähetetään pelkkä tieto edellisen kuvan objektien liikkumisesta liikevektoreiden avulla sekä ennustekuvan ja todellisen kuvan erotus (residuaalikuva), eikä siis koko erotuskuvaa. Ennustekuva muodostetaan siis siten, että edellisen kuvan makrolohkot siirretään eri paikkoihin siten, että näin saatu liike-ennustettu kuva vastaa mahdollisimman hyvin nykyistä todellista kuvaa. Todellisesta nykyisestä kuvasta vähennetään liike-ennustettu kuva, ja saadaan residuaalikuva, joka lähetetään yhdessä liikevektoreiden kanssa. Vastaanottopäässä muodostetaan edellisestä kuvasta vastaanotettujen liikevektoreiden avulla ennustettu kuva. Tähän lisätään residuaalikuva, jolloin saadaan todellinen seuraava kuva.

Kuvat, jotka muodostetaan menneiden kuvien avulla ovat P-kuvia ja tulevista ja menneistä kuvista muodostetut B-kuvia. Videokuva koostuukiin säännöllisin väliajoin tulevasta avainkuvasta eli I-kuvasta, jota seuraa tietty määrä P- ja B-kuvia.

Kuva koostuu makrolohkoista, jotka jokainen sisältävät luminanssi- ja krominanssinäytearvoja. Nämä näytearvot kertovat jokaisesta pikselistä sen sisältämän väri- ja kirkkausinformaation. Jokaisen kuvan makrolohkot muodostavat viipaleen, joka kertoo miten makrolohkot on koodattu (jos oletetaan kuvan koostuvan aina yhdestä viipaleesta). Esimerkiksi I-viipale sisältää vain I-tyyppisiä makrolohkoja ja P-tyyppinen vain P-tyyppisiä. Näin dekoodekki tietää mitä eri koodaustapoja on käytetty kun kaikki makrolohkot on koodattu samalla tavalla viipaleessa. [10, s. 159-161.]

H.264 on lohkopohjainen videonkoodausstandardi, joten se jakaa kuvan makrolohkoiksi. Makrolohkot ryhmitellään makrolohkojoukoiksi, jotka muodostavat viipaleen. Dekoodekki muodostaa kuvan yhden tai useamman viipaleen avulla (kuva 1).



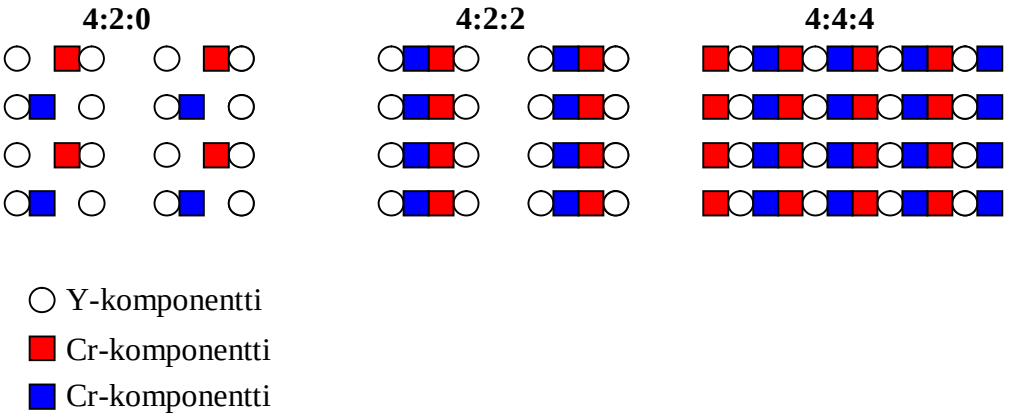
Kuva 1. Hierarkia H.264-standardissa. [9, s. 59-62.]

3.1 Makrolohkot (Makroblocks)

Kaikki nykyaiset videonpakkausstandardit ovat lohkopohjaisia ja niissä kuva jaetaan lohkoihin, joihin itse koodausoperaatiot kohdistetaan. Näitä lohkoja kutsutaan makrolohkoiksi. Makrolohkot sisältävät tietyltä rajatulta alueelta näytearvoja pikseleistä. Näytearvojen avulla muodostetaan pikselin väri. Esimerkiksi RGB-mallia käytettäessä otetaan punaisesta, vihreästä ja sinisestä väriarvo pikselille. Näin saadaan muodostettua oikea väri pikselille kolmen päävärin avulla. RGB-mallin tilalle on MPEG-standardeissa otettu YCrCb-malli. Malli otettiin käyttöön, koska se soveltuu paremmin videonsiirto ja tallennussovelluksiin. RGB-malli ja YCrCb-malli ovat kuitenkin hyvin lähellä toisiaan ja YCrCb-mallin arvot voidaan muuttaa R-, G- ja B-arvoiksi. [21]

YCrCb-mallissa makrolohkot koostuvat kolmesta eri komponentista: luminanssikomponentista ja kahdesta kromikomponentista. Tätä kolmen komponentin yhdistelmää kutsutaan YCrCb-malliksi, joka on yleisesti käytössä MPEG-standardeissa. Komponenttien arvot ovat kokonaislukuja ja niiden avulla kuvataan pikselin arvoa. Y- eli luminanssikomponentti kuvaa kirkkausarvoa ja Cr-komponentti vihreä-punaista vaihtelua ja Cb-komponentti kelta-sinistä vaihtelua. [18.]

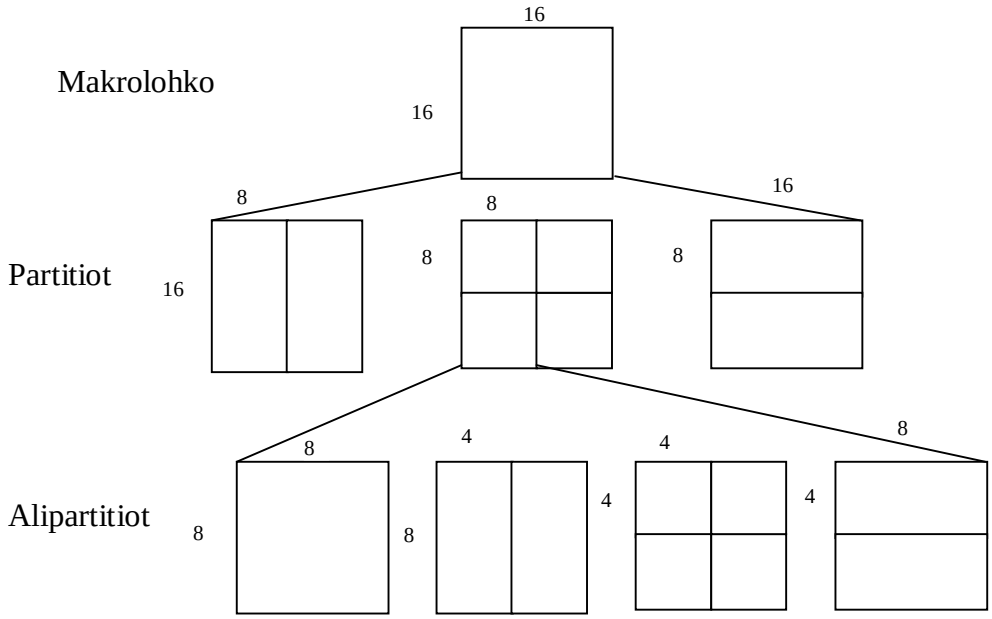
Yleisin koko makrolohkolle on 16x16-pikseliä. Näytteenottotarkkuudella 4:2:0, jokaisesta pikselistä otetaan luminanssiarvo. Eli makrolohko sisältää luminanssikomponentti arvoja 16x16 kappaletta. Krominanssikomponenteista C_r ja C_b otetaan näytteitä vain 8x8 kappaletta eli vain joka toisesta pikselistä. Komponentti arvojen ottomäärä on riippuvainen näytteennottotarkkuudesta. [15, s. 48-49.]



Kuva 2. Eri näytteenottotarkkuuksia 8x8-kokoiselle makrolohkolle. 4:2:0-näytteenotossa kutakin neljää Y-komponenttia varten otetaan yksi Cr- ja Cb-komponentin arvo. 4:2:2-näytteenotossa kutakin neljää Y-komponenttia varten otetaan kaksi Cr- ja Cb-komponentin arvoa. 4:4:4-näytteenotossa jokaista komponenttia otetaan saman verran. [7, s. 15.]

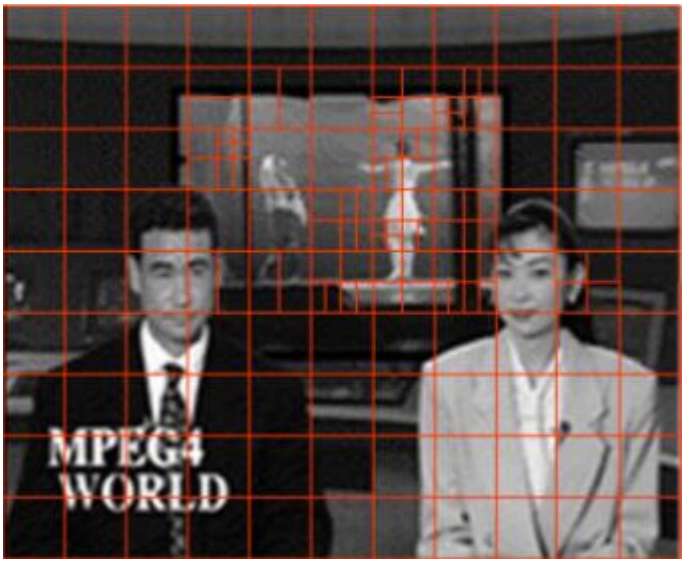
Cr- ja Cb-komponentteja ei kuvata niin tarkasti kuin Y-komponenttia, koska ihmisen silmä on herkempi kirkkausvaihtelulle kuin värivaihtelulle. Värivaihtelun virheet eivät siis ole niin kriittisiä kuin kirkkausvaihtelun (kuva 2). [18.]

Vaikka koodattava kuva paloitellaan 16x16-kokoisiksi makrolohkoiksi, ei makrolohkoja kuitenkaan tarvitse käsitellä kokonaisina. H.264-standardin suurena erona muihin standardeihin on, että se sallii makrolohkojen jaon tarvittaessa pienempiin osiin eli partitioihin (Macroblock partitions). Partitiot voivat olla 16 x 8-, 8 x 16- tai 8x8-pikselin kokoisia. Partitiot voidaan vielä jakaa pienempiin alipartitioihin (Macroblock sub-partitions). Alipartitioiden koko on 8 x 4-,4 x 8-, tai 4 x 4-pikseliä. Makrolohkon jakaminen partitioihin ja alipartitioihin on esitetty alla olevassa kuvassa 3. [5, s. 26.]



Kuva 3. H.264-standardin mahdollistamia tapoja jakaa makrolohko partitioihin (Macroblock partitions) ja alipartitioihin (Macroblock sub-partitions). [5, s. 26.]

Koodausoperaatiot suoritetaan aina yksilöllisesti jokaiseen partitioon. Partitioinnin avulla saadaan lisättyä koodauksen tarkkuutta ja laskutoimitukset yksinkertaistuvat. Esimerkiksi makrolohkot, jotka sisältävät liikettä ja pieniä yksityiskohtia, jaetaan partitioihin ja alipartitioihin. Näin saadaan jokainen yksityiskohta koodattua erikseen, jolloin kompressointisuhde ja kuvanlaatu paranevat. Tasaiseen taustaan, jossa ei ole liikettä, valitaan tavallinen 16x16-makrolohko, koska se ei sisällä yksityiskohtia eikä liikettä. Kuten kuvassa 4, taustalla olevassa televisiossa pienet yksityiskohdat ovat liikkeessä. Niissä kohdissa makrolohko on partitioitu, kun taas tausta koodataan 16x16-makrolohkoina. [7, s. 28]



Kuva 4. Esimerkki kuvasta, joka on jaettu eri kokoisiin lohkoihin. Taustalla näkyvässä televisiossa on käytetty partitioita ja alipartitioita. Muuten on käytetty 16x16-makrolohkon kokoja. [7, s .28.]

Makrolohkojen pakkaamista tehostetaan ennustamalla niitä videonkompressoinnissa. Ennustamisella tarkoitetaan kahden lohkon vertailua keskenään: koodattavan- ja referenssilohkon. Lohkoja voidaan ajatella kokeiltavan päällekkäin ja merkitsemällä eroavaisuudet niissä ylös. Näin dekoodekille voidaan kertoa eroavaisuudet ja mitä on käytetty referenssilohkona, niin se osaa muodostaa koodattavan lohkon.

H.264:ssä makrolohkot ovat joko intra- tai inter-tyyppisiä. Intra-makrolohkoja ennustetaan samassa kuvassa olevien arvojen perusteella. Inter-makrolohkoihin hyödynnetään aikaisempien kuvien ennusteita. Ennustamista ei ole kuitenkaan pakko käyttää makrolohkojen kohdalla. Informaatio voidaan myös siirtää sellaisenaan seuraavaan koodausvaiheeseen.

3.2 Viipaleet (Slices)

H.264:ssä makrolohkot ryhmitellään joukoiksi. Joukkoja kutsutaan viipaleiksi. Yksittäinen kuva voi muodostua yhdestä tai useammasta makrolohkojoukosta eli viipaleesta. Tietyn tyyppinen makrolohkojoukko muodostaa aina tietyn tyyppisen viipaleen. H.264:ssä viipaletyyppejä on määritelty viisi erilaista I-, P-, B-, SI- ja SP-viipale. Jokaisen viipaleen makrolohkot koodataan aina viipaleelle ominaisella tavalla. [5.]

Päätarkoitus viipaleiden muodostamiselle on tarjota suojaa lähetyksessä tapahtuvia virheitä vastaan. H.264:ssä virheisiin sopeutumista parannetaan

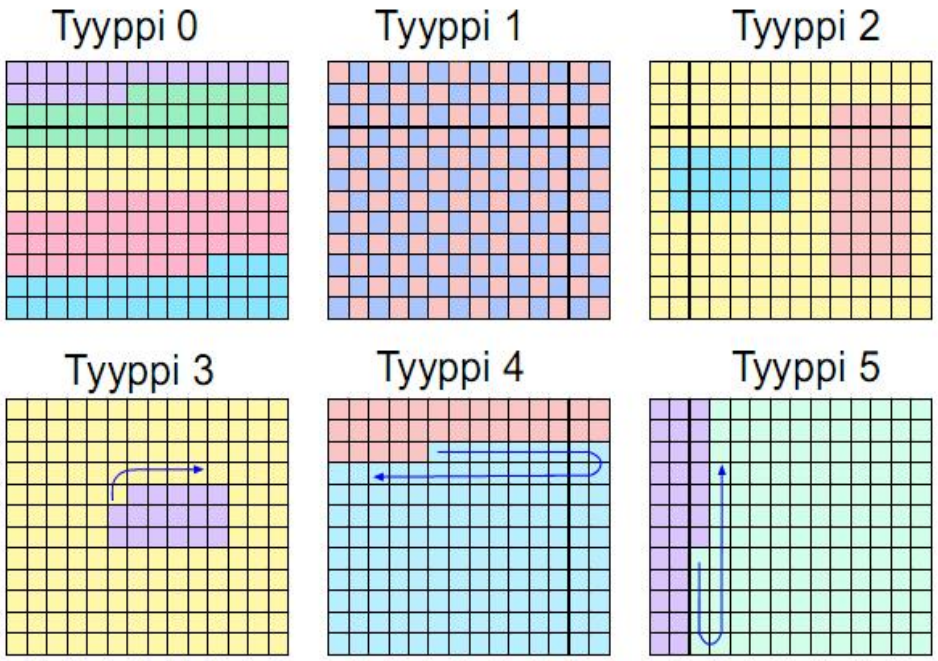
ns. joustavalla makrolohkojen käsittelyllä eli FMO:lla (Flexible Macroblock Ordering). Joustavan makrolohkojoukkojen käsittelyn avulla voidaan vaikuttaa, miten kuva koostuu viipaleista. Standardissa on määritelty 6 eri viipalejakotapaa ja seitsemäs tapa on jätetty käyttäjän itse määrittelemäksi (kuva 5). [7, s. 16-17]

Ensimmäinen tyyppi 0 on sama kuin ei FMO:ta käytettäisi lainkaan. Makrolohkot ovat peräkkäisenä jonona ja ne on luettu rasterijärjestyksessä eli vasemmalta oikealle ja ylhäältä alas. Dekoodekille täytyy näin ilmoittaa viipaleen pituuskoodi, jotta se pystyy dekoodaamaan viipaleista kuvan. Myös koko kuvan muodostuessa yhdestä viipaleestä käytetään tyyppiä 0 (kuva 5).

Toinen tapa on jakaa makrolohkot ns. shakkilauta muotoon. Tämä tyyppi 1 parantaa virheen sietoa. Esimerkiksi virheen sattuessa makrolohkoon se voidaan koodata identtisesti viereisen makrolohkon mukaan. Näin katsoja ei välttämättä edes huomaa virhettä kuvassa. Tyyppi 1 myös parantaa videon salausta. Ulkopuolisen jolle kuvan ei haluta näkyvän on hyvin vaikea saada muodostettua sitä, mikäli ei ole tietoa, että viipaleet on jaettu ns. shakkilautamuodostelmaan. [7, s. 16-17]

Kolmas tapa eli tyyppi 2 on tarkoitettu kuvalle, josta halutaan erikseen koodata jokin mielenkiintoinen asia, esimerkiksi ihmisen kasvot tai muu taustasta erottuva ja liikettä sisältävä objekti. Näin koodekki pystyy kompressoimaan taustaa enemmän ja keskittymään keskeiseen asiaan kuvassa. Suorakaiteen muotoisille alueille on annettu ROI-arvot (Region Of Interest). Näiden ROI-arvojen avulla koodekki ja dekoodekki tietää suorakaiteen vasemman ylänurkan ja oikean alanurkan. Näin molemmissa on oikean kokoinen alue käytössä. [8, s. 31-32.]

Joustavan makrolohkon mahdollistavat tyypit 3-5 ovat periaatteeltaan samanlaisia. Niissä kaikissa tietty alue kasvaa tai pienenee tiettyä nopeutta. Koodekille ja dekoodekille kerrotaan alueen kasvunopeus, suunta ja paikka.



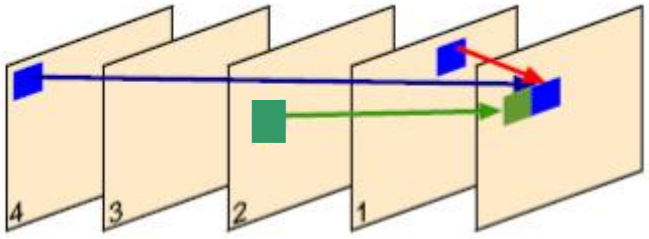
Kuva 5. Eri joustavan makrolohkon käsittelyn mahdollistavat tyypit. Jokainen väri esittää eri viipaletta. Tyyppi 6 on käyttäjän itse määrittämä. [7, s. 17.]

Makrolohkojoukko siis muodostaa viipaleen. Viipaleen voidaankin ajatella H.264:ssä kertovan ainoastaan, miten makrolohkojoukko on koodattu. Itse kuva voi muodostua monesta erilaisesta viipaleesta. H.264:ssä ei ole käytössä enää aikaisemmista standardeista tuttua GOP:ia (Group Of Pictures). GOP on aikaisemmissa standardeissa määrännyt, missä järjestyksessä viipaleet tulevat. Esimerkiksi IBBP-viipalejärjestys on tarkoittanut, että ensin tulee I-viipale sitten kaksi B-viipaletta ja lopuksi P-viipale, jonka jälkeen tulee taas I-viipale. Näin koodekki on pakotettu koodaamaan koko ajan tietyssä järjestyksessä, vaikka niin ei saavutettaisi hyvää kompressointisuhdetta. H.264:ssä viipale kertoo vain, miten makrolohkojoukko on koodattu ja missä ne sijaitsevat kuvassa. GOP:sta luopuminen myös kasvattaa referenssien hyödyntämisen mahdollisuutta. Eri viipaleita H.264:ssä on määritelty viisi erilaista: I-,P-,B-,SI- ja SP-viipaleet. Jokainen viipale on siis makrolohkojoukko ja sille on omat ominaisuudet koodauksen kannalta. [5.]

I-viipale: Tehdään ainoastaan Intra-koodaus ja toimii synkronointikohtana videossa. I-viipaletta voidaan käyttää referenssiviipaleena.

P-viipale: Inter- tai Intra-koodataan ja sitä voidaan käyttää referenssiviipaleena. P-viipaleen muodostuksessa voidaan käyttää myös useampaa eri re-

ferenssiviipaletta. Muodostuksessa käytetään kuitenkin vain jo koodattuja referenssiviipaleita. P-viipaleen muodostaminen usean eri referenssikuvan avulla kuva 6.



Kuva 6. P-viipaleen muodostaminen useita eri referenssikuvia käyttäen. Sininen neliö muodostetaan kahden referenssin avulla ja vihreä yhden. [7, s. 32.]

B-viipale: B-viipale Inter- tai Intra-koodataan. Voidaan aikaisemmista standardeista poiketen käyttää myös referenssiviipaleena. Aikaisemmissa standardeissa B-viipale on muodostettu jäljessä olevasta ja edellä olevasta viipaleesta, eikä sitä ole voinut käyttää referenssinä. H.264:ssä B-viipaleen muodostuksessa voidaan käyttää referenssiviipaleita, miten koodekki haluaa. Esimerkiksi kahta jo mennyttä viipaletta, yhtä mennyttä ja yhtä tulevaa viipaletta tai vain yhtä tulevaa viipaletta. [5.]

SP- ja SI-viipaleet on suunniteltu korvaamaan I-viipale H.264:ssä Extended-profiilissa. I-viipaleita voidaan käyttää esimerkiksi IPTV:ssä kanavan vaihtoa varten. Videon synkronointi voidaan myös hoitaa SP- ja SI-viipaleen avulla I-viipaleen sijaan. I-viipale sisältää huomattavasti enemmän dataa kuin muut viipaleet, joten se nostaa bittivirran suuruutta. SP-viipaleissa käytetään liikekompensoitua ennustamista, joten ne sisältävät huomattavasti vähemmän dataa kuin I-viipale. [13, s. 10.]

Esimerkki: Dekoodekki on koodaamassa kanavan 1 P-viipaletta, kun käyttäjä vaihtaa kanavaa. Kanavan vaihtokohdassa täytyisi olla nyt I-viipale, jotta kanavan vaihto onnistuu. Toinen tapa on sijoittaa I-viipaleen tilalle P-viipale ja sen lisäksi SP-viipale bittivirtaan. SP-viipale on ennustettu niin, että dekoodekki pystyy muodostamaan sen avulla kanavan 2 P-viipaleen käyttämällä referenssinä kanavan 1 P-viipaletta. Jokainen vaihto vaatiikin näin oman ylimääräisen SP-viipaleen. Lisäksi vaaditaan ylimääräinen SP-viipale jos vaihto tehdään kanavasta 2 kanavaan 1. Tästä huolimatta SP-viipaleiden käyttö I-viipaleen tilalla säästää bittivirtaa. [2, s. 17]

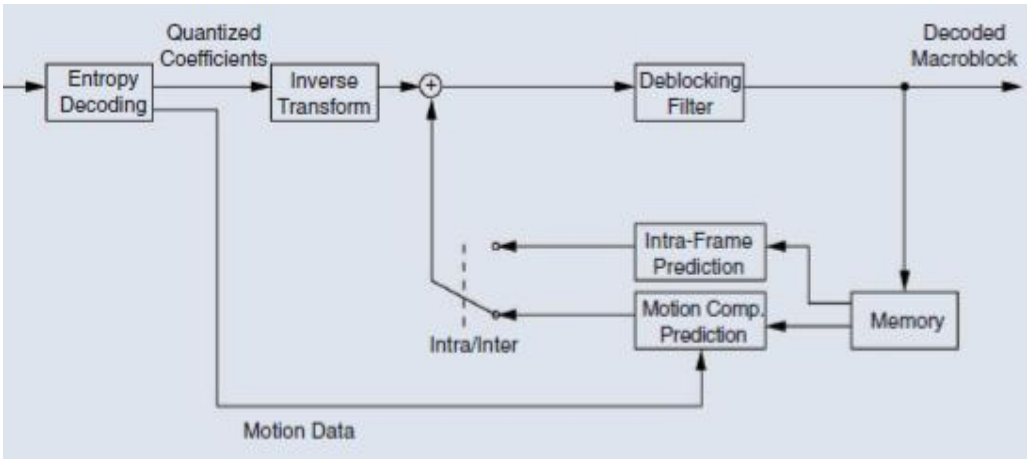
Usein kanavat kuitenkin ovat niin erilaisia, ettei niiden välillä ole mitään yhtäläisyyttä. Näin Inter-ennustaminen menettää merkitystään. Tällöin voidaan käyttää SP-viipaleiden tilalla SI-viipaleita. Tapahtuma on muuten sama kuin SP-viipaleen kohdalla, mutta SI-viipaleissa käytetään ainoastaan ennustamiseen Intra_4x4-ennustamista. SI- ja SP-viipaleet ovat käytössä vain Extended-profiilissa. [6.]

H.264:ssä käytetään paljon ennustamista makrolohkojen-kompressoinnissa ja ennusteista voidaan muodostaa uusia referenssilohkoja, joita voidaan käyttää myöhemmin ennustamisessa. Referenssilohkoon tuleva virhe, jota ei saada korjattua tallentuu dekoodekin muistiin. Tämä voi aiheuttaa suuria virheitä myöhemmin vaikka ei siirtoteitse tapahtuisi virheitä. Jos virhe on jo muistissa olevassa referenssilohkossa, se sotkee koko viipaleen dekoodauksen ja mahdollisesti tallentaa taas virheellisen referenssilohkon muistiin. SP- ja SI-viipaleet on suunniteltu torjumaan tälläisiä virheitä. Dekoodekin huomatessa virheen bittivirrassa, mitä se ei pysty korjaamaan, se voi käyttää tulevaa SP- tai SI-viipaletta palautumiseen ja näin muodostaa oikeanlaisen referenssilohkon. [13, s. 10.]

3.3 H.264-koodekin rakenne

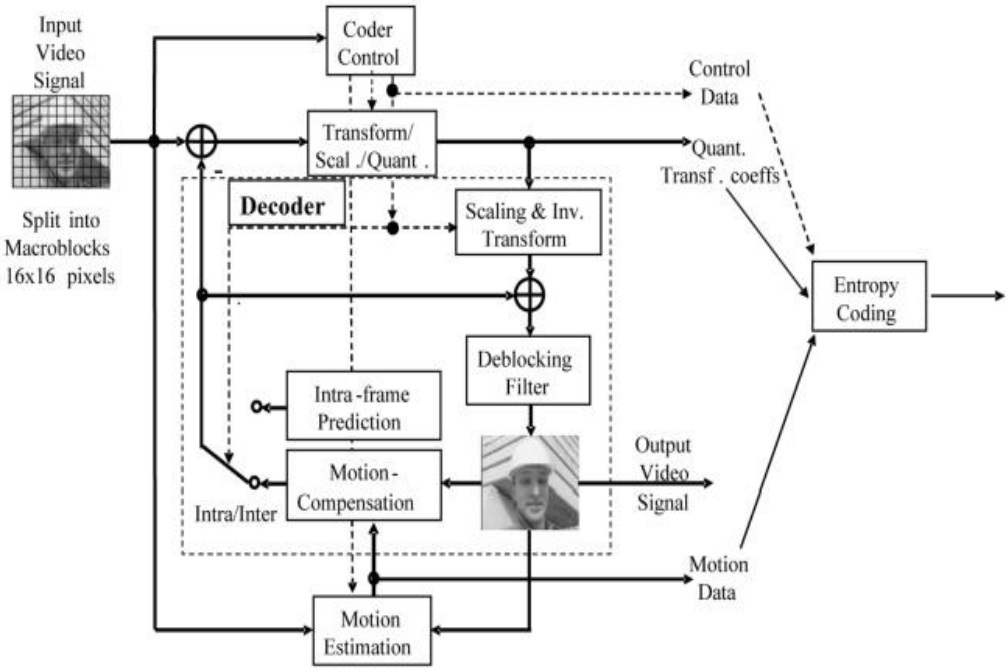
H.264-koodekki on häviöllinen koodaaja, joten kuva ei palaudu dekoodatessa täysin alkuperäiseksi. Pyrkimys on kuitenkin päästä sovelluskohtaisesti mahdollisimman hyvään kuvaan.

Standardi ei sisällä suoranaista spesifikaatiota koodekin rakenteesta ja siksi koodekista ei voi esittää suoraa mallia. Dekoodausprosessi ja dekoodekin rakenne on kuitenkin määritelty standardissa tarkasti (kuva 7).



Kuva 7. Dekoodekin rakenne. [2, s. 9.]

Koodekin voidaan ajatella toimivan samalla tavalla kuin dekoodekki, mutta kaikki tapahtuu päinvastaisessa järjestyksessä. Dekoodekin avulla tehty malli koodekin sisäisestä rakenteesta (kuva 8).



Kuva 8. H.264-koodekin sisäinen rakenne. [1, s. 567.]

Seuraavaksi käydään koodekin toiminta pääpiirteittäin läpi kuvan 8 avulla. Ensimmäiseksi koodekkiin tuleva kuva (Input Video Signal) jaetaan makrolohkoiksi (Split into Macroblocks). Seuraavaksi makrolohko ennustetaan referenssilohkojen avulla. Makrolohkolle etsitään mahdollisimman samanlainen referenssimakrolohko, jonka avulla se ennustetaan. Ennustamisen voi-

daan ajatella olevan kahden päällekkäisen makrolohkon sovittamista toisiinsa ja eroavaisuuksien merkitsemistä ylös.

Jokainen makrolohko koodataan intra- tai inter-ennustamalla. Intra-ennustamisessa käytetään vain saman viipaleen makrolohkoja hyväksi. Inter-ennustamisessa voidaan käyttää myös aikaisemmin koodattujen viipaleiden makrolohkoja hyväksi. Ennustaminen tuottaa ennuste-informaation (Motion Data) ja ennustemakrolohkon. Ennustemakrolohkosta vähennetään alkuperäinen makrolohko ja lopputulos on residuaalimakrolohko. Mikäli sopivaa referenssilohkoa ei löydetä voidaan ennustaminen jättää tekemättä ja siirtyä seuraavaan koodausvaiheeseen (kuva 8).

Residuaalimakrolohko siirretään muunnoskoodattavaksi (Transform Coding). Siellä koodekki poistaa residuaalimakrolohkosta sen sisältämää jäännösinformaatiota, kuitenkin säilyttämällä kuvainformaation. Seuraavaksi residuaalimakrolohko kvantisoidaan eli siitä poistetaan vähän merkityksellisiä bittejä. Kvantisointi on ensimmäinen häviöllinen koodausvaihe. Tähän asti koodattu residuaalimakrolohko voitaisiin siis palauttaa alkuperäiseksi makrolohkoksi tekemällä käänteismuunnos ja lisäämällä siihen ennustemakrolohko (kuva 8).

Kvantisoinnin jälkeen residuaalimakrolohko siirretään entropiakoodattavaksi (Entropy Coding). Ennen siirtämistä entropiakoodattavaksi koodekki kuitenkin tekee residuaalimakrolohkosta identtisen kopion. Kopio siirretään dekoodattavaksi. H.264-koodekki sisältääkin myös dekoodekin (Decoder). Dekoodauksen jälkeen kopio voidaan pistää lohkosuodattimen (Deblocking filter) läpi mikäli koodekki katsoo sen tarpeelliseksi. Kopiolle tehdyn dekoodauksen ja mahdollisen suodatuksen jälkeen makrolohko tallennetaan koodekin muistiin. Näin on saatu uusi referenssilohko, jota voidaan käyttää muiden makrolohkojen ennustamisessa (kuva 8).

Lopuksi koodekinohjaimen (Coder Control) merkinnät (Control Data) kaikki koodausvaiheet läpi käynyt residuaalimakrolohko ja ennusteinformaatio (Motion Data) entropiakoodataan (Entropy Coding). Entropiakoodauksessa pyritään poistamaan lähetettävien bittien määrää hyödyntämällä bittijonon sisältämää tilastollista jakaumaa. Usein esiintyvät bittisymbolit korvataan lyhyillä symboleilla ja harvemmat pitemmillä.

4 H.264-KOODAUSTÖKALUT

Seuraavaksi käydään läpi tarkemmin eri koodaustapahtumat, jotka tekevät H.264-koodekista tehokkaan videonkompressoijan.

4.1 Ennustaminen (Prediction)

Kompressiotehokkuutta huomattavasti parantava ominaisuus H.264-koodekissa muihin nähden on ennustaminen. Ennustamista käytetään hyvin paljon H.264:ssä ja se on viety pitemmälle kuin aikaisemmissa standardeissa, joissa makrolohkon arvoja on ennustettu vain makrolohkon sisällä. H.264 mahdollistaa makrolohkojen ennustamisen muutenkin kuin makrolohkon sisällä, mikä on parantanut kompressointikykyä huomattavasti.

Ennustamisessa hyödynnetään makrolohkojen samankaltaisuutta. Esimerkiksi jos löydetään koodekin muistista paljon koodattavaa makrolohkoa muistuttava vertailumakrolohko eli referenssimakrolohko, voidaan niistä laskea vain eroavaisuudet. Tämän jälkeen ilmotetaan dekoodekille mitä makrolohko käytettiin referenssinä ja mitkä oli eroavaisuudet.

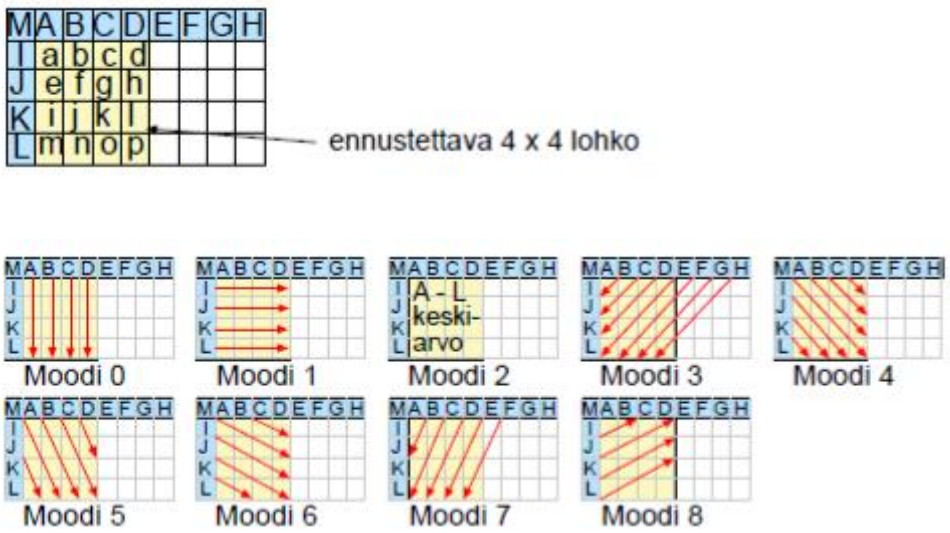
H.264 mahdollistaa kaksi tapaa ennustaa. Ennustaminen voidaan tehdä saman viipaleen makrolohkojen avulla eli Intra-ennustaminen tai jo aikaisempien koodattujen viipaleiden makrolohkojen avulla eli Inter-ennustaminen. Intra- ja Inter-ennustaminen sisältää eri tapoja, miten ennustaminen voidaan tehdä. Koodekin täytyy osata päättää, mitä ennustusmenetelmää sen tulisi käyttää, jotta saavutettaisiin mahdollisimman hyvä laatu kuvalle bittivirran sallimissa rajoissa. Eri bittivirroille ei ole olemassa yhtä oikeaa ennustusmenetelmää, vaan paras lopputulos saadaan eri ennustusmenetelmällä eri tilanteissa. Paras lopputulos saavutettaisiin, kun käytäisiin kaikki menetelmät läpi joka kerta ja niistä valittaisiin parhaan tuloksen antava. Tämä kuitenkin vaatisi laitteelta suuren laskentatehon, eikä se toimisi reaaliaikaisessa videokuvassa. Käytännössä joudutaankin rajaamaan ennustamisen etsintäaluetta referenssilohkoissa. Tämän avulla saadaan laskentamäärää pienennettyä ja suurimmassa osassa lohkoja saadaan valittua paras ennustusmenetelmä. Mikäli makrolohkolle/partitiolle ei löydy toimivaa referenssilohkoa, voidaan ennustaminen jättää tekemättä.

4.1.1 Intra-ennustaminen (Intra Prediction)

Kompressiotehokkuutta saadaan kasvatettua huomattavasti, kun otetaan huomioon makrolohkojen sisältämä keskinäinen samankaltaisuus. Samankaltaisuuden huomioimista eli ennustamista yksittäisen viipaleen sisällä, kutsutaan intra-ennustamiseksi.

Intra-ennusteet hyödyntävät vain yksittäisen viipaleen aikaisemmin koodattujen makrolohkojen tietoa koodattavaan makrolohkoon. Aikaisemmat MPEG-koodekit hyödyntävät ennustamista vain koodattavan makrolohkon sisällä, kun taas H.264-koodekki hyödyntää ennustamista vierekkäisiin makrolohkoihin. Verrattavat naapurivertailulohkot sijaitsevat vasemmalla ja/tai yläpuolella koodattavaan makrolohkoon nähden. Viitteet tehdään jo lähetettyihin makrolohkoihin. Intra-ennustamiseen H.264:ssä on kaksi eri menetelmää: Intra_4x4- ja Intra_16x16-ennustaminen. Lisäksi ennustamiset voidaan tehdä eri tavalla eri makrolohkoille. Näitä tapoja kutsutaan moodeiksi.

Intra_4x4-ennustamisessa makrolohko, joka on 16x16-kokoinen jaetaan kuuteentoista 4x4-kokoiseen alueeseen. Jokainen 4x4-kokoinen alue ennustetaan yksilöllisesti. Jokaiseen alueeseen siis voidaan käyttää eri ennustustapaa. Intra_4x4-ennustaminen tukee yhdeksää eri ennustamistapaa luminanssilohkolle. Ennustamistavat määräävät missä suunnassa ennustaminen tehdään. Luminanssilohkolle mahdolliset ennustussuunnat on esitetty kuvassa 9. [2, s. 11.]



Kuva 9. Intra_4x4-ennustamisen eri tavat luminanssilohkolle. Keltainen kuvaa ennustettavaa lohkoa ja sininen on viereisten lohkojen arvot. [7, s. 20.]

Esimerkiksi jos on valittuna ennustusmoodi 0 kaikki A-sarakkeen arvot ennustetaan A:n arvon avulla ja B-sarakkeen arvot B:n arvolla. Moodi 1:ssä I-rivit arvot ennustetaan I-arvon avulla jne. Moodi 2:ssa lasketaan keskiarvo kaikista naapuriarvoista ja 4x4-alue ennustetaan keskiarvon avulla. Tälläinen tarkka ennustaminen sopii hyvin kuvan kohtaan, joka sisältää paljon yksityiskohtia.

Intra_16x16-ennustamisessa ennustetaan koko makrolohko samaa tapaa käyttämällä. Tapoja miten makrolohko ennustetaan on neljä erilaista Intra_16x16: vaakasuora-, pystysuora-, DC- ja taso-ennustaminen. Intra_16x16-ennustaminen tapahtuu muuten samalla tavalla kuin Intra_4x4-ennustaminen ainoana erona on, että Intra_16x16-ennustaminen tehdään koko 16x16-alueelle saman moodin mukaan. Tämä tehdään sen takia, että makrolohko, jota ei ole jaettu partitioihin tai alipartitioihin, sisältää yleensä hyvin vähän väri- ja kirkkausmuutoksia. Ennustamisen ei tarvitsekaan näin olla niin tarkkaa. [1, s. 569.]

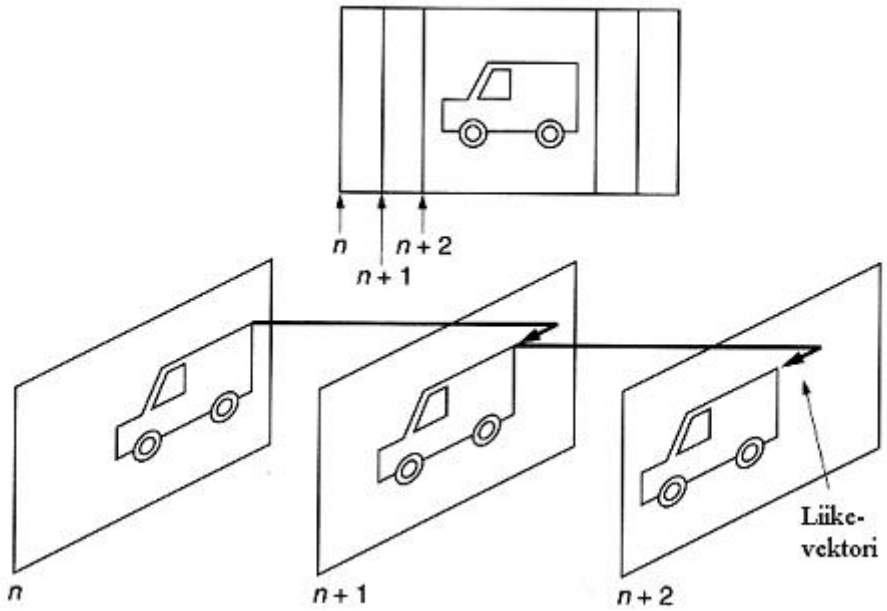
Krominanssikomponentit Cb- ja Cr-ennustus on samanlainen kuin Intra_16x16-ennustaminen. Krominanssikomponentit jaetaan vain 8x8-alueisiin, ja niiden ennustamiseen on käytössä samat moodit kuin Intra_16x16-ennustamisessa luminanssikomponentille. Krominanssikomponentin ennustus ei tarvitse olla yhtä tarkka kuin luminanssikomponentin, koska ne muuttuvat hyvin vähän pienellä alueella eikä ihmisen silmä ole niin herkkä krominanssikomponentin virheille kuin luminanssikomponentin. [1, s. 12.]

4.1.2 Inter-ennustaminen (Inter Prediction)

Merkittävin kompressiotehokkuutta lisäävä menetelmä on aikaisempiin standardeihin nähden on H.264:ssä saavutettu liike-estimoinnin ja -kompensoinnin avulla. Tehokkuutta liikkeen-estimointiin on saatu ottamalla H.264:ssä käyttöön: lohkorakenteiden vaihtuva koko, eri tarkkuuksien käyttö liikevektoreiden vertaamisessa referenssikuvaan, monien eri referenssikuvien hyödyntäminen ja uudenlainen tapa ennustaa B-viipaleita. Useiden eri tapojen ja referenssilohkojen käyttö on myös suurin syy videokoodekkien jatkuvasti kasvavaan laskenta- ja muistiresurssivaatimuksiin.

4.1.3 Liikekompensoitu ennustaminen (Motion Compensated Prediction)

Koodattavan makrolohkon/partition ja referenssimakrolohkon luminanssikomponentin välillä on yleensä sijaintieroa (kuva 10). Sijaintieroa kuvataan liikevektorin (motion vector) avulla. Liikevektori on kaksiulotteinen vektori, joka ilmaisee koodattavan alueen ja referenssialueen matkan ja suunnan eron. Näin dekoodekille ei tarvitse lähettää muuta tietoa kuin, mihin suuntaan ja paljonko makrolohko on liikkunut. Itse makrolohkoa ei välttämättä tarvitse lähettää enää uudelleen dekoodekille, vaan lähetetään ennustettavan makrolohkon ja referenssinä käytetyn makrolohkon ero. Näin säästetään paljon bittivirran suuruudessa.



Kuva 10. Liikevektorin muodostaminen liikkuneelle lohkolle. Kuvassa n-frame toimii referenssinä n+1-framelle ja n+1-frame referenssinä n+2-framelle. [11, s. 10.]

Jokaiselle makrolohkolle etsitään oma liikevektori. Mikäli makrolohko on jaettu partitioihin muodostuu useita liikevektoreita. Jos makrolohko on jaettu 4x4-pikselin kokoisiin alipartitioihin, voi yhdestä makrolohkosta muodostua 16 eri liikevektoria. Kuvassa usein viereiset objektit liikkuvat samassa suhteessa ja suunnassa, mikä mahdollistaa ennustamisen liikevektoreille. Kunkin liikevektorin ohelle täytyy tallentaa tieto, mitä referenssilohkoa on milloinkin käytetty. Tätä tietoa käytetään hyödyksi dekoodausvaiheessa. [3.]

H.264-standardissa ei määritellä, miten ennusteinformaatioon osoittavat liikevektorit tarkalleen ottaen löydetään. Liikevektorien etsiminen eli liikkeen estimointi on jätetty kokonaan koodekin suunnittelijan vastuulle. Standardi kuitenkin määrittelee tarkkuuden, millä luminanssikomponentti voi osoittaa referenssikuvaan. Liikevektori voi osoittaa ½-näytteen tarkkuudella tai ¼-näytteen tarkkuudella referenssikuvaan nähden. Väliarvot sopivat yleensä paremmin liikevektorien muodostamiseen kuin varsinaiset näytearvot. Väliarvot koodekki joutuu interpoloimaan eli arvioimaan viereisten arvojen perusteella. Krominanssikomponentin liikevektoreita muodostettaessa joudutaan ottamaan makrolohkon oikealta yksi sarake ja alhaalta yksi rivi. Näin ylimääräisten sarakkeen ja rivin avulla krominanssikomponentille muodostetaan bilineaarisen interpoloinnin avulla liikevektori. [3, s. 162-165; 1, s. 569-570.]

4.2 Muunnoskoodaus (Transform Coding)

Muunnoskoodauksen tarkoitus on vähentää residuaalimakrolohkon sisältämän datan määrää mahdollisimman paljon, kuitenkin kuvainformaatiota hävittämättä. Aiemmissa videonkoodausstandardeissa käytetään muunnoskoodaukseen diskreetti kosinimuunnosta eli DCT-muunnosta (Discrete Cosine Transform). DCT-muunnoksessa joudutaan käyttämään liukulukuja laskuissa. Liukulukujen käyttö aiheuttaa paljon pyöristysvirheitä, jotka saavat aikaan eri tuloksia pyöristyksissä koodekin sisäisen dekoodaussilmukan ja ulkoisten dekoodekkien välillä. Tätä koodausvirhettä kutsutaan liukumaksi. Aikaisemmissa videonkoodausstandardeissa ei ole käytetty niin paljon Inter-ennustamista kuin H.264:ssa, mikä on hyvin altis liukumavirheille. [2, s.13.]

H.264:ssä on otettu DCT-muunnoksen tilalle käyttöön yksinkertaisempi kokonaislukumuunnosmenetelmä. Kokonaislukumuunnoksessa residuaalimakrolohkon arvoja kerrotaan DCT:stä johdetulla muunnosmatriisilla. Koska kyseessä on vain kokonaislukumuunnoksia, vältytään pyöristykseltä ja näin liukumavirheiltä. Kokonaislukumuunnoskoodaus on myös laskennallisesti huomattavasti DCT-muunnosta yksinkertaisempi, jolloin säästetään laitteen laskentatehoa. [2, s. 13.]

H.264:ssä käytetään kolmea eri muunnoskoodausvaihetta. Kaikille residuaalimakrolohkoille ei suoriteta kaikkia vaiheita vaan vaiheiden määrä on riippuvainen miten residuaalimakrolohko on koodattu. Ensimmäinen tapa kuitenkin

tehdään kaikille residuaalimakrolohkoille riippumatta, miten se on koodattu. Ensimmäinen tapa tehdään niin luminanssikomponentille ja molemmille krominanssikomponenteille erikseen. Ensimmäisessä muunnoskoodaustavassa muunnosmatriisin kooksi on valittu 4x4. Tämä koko on valittu sen takia, että se on pienin mahdollinen koko, mihin makrolohko on voitu jakaa. Muunnosmatriisi H_1 esitetty alla kuvassa 11. Muunnosmatriisilla kerrotaan 4x4-luminanssikomponentti ja molemmat 4x4-krominanssikomponentit erikseen.

$$H_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

Kuva 11. Muunnosmatriisi H_1. [2, s. 13.]

Jos makrolohko on 16x16-kokoinen ja sen ennustamiseen on käytetty Intra-ennustamista tehdään sille muunnoskoodauksen toinen vaihe myös. Toisessa vaiheessa makrolohkon luminanssikomponentit kerrotaan Hadamard-matriisilla. Hadamard-matriisi sisältää vain +1- tai -1-arvoja, ja matriisin kaikki rivit ovat erilaisia (kuva 12). Makrolohkossa joka on 16x16-kokoinen ja jossa on käytetty Intra-ennustustamista, on huomattu sen luminanssi- ja krominanssikomponenttien muistuttavan paljon toisiaan. Tämän takia makrolohkon luminanssikomponentille suoritetaan myös muunnoskoodauksen toinen vaihe, jossa käytetään muunnosmatriisia H_2. Makrolohkon krominanssikomponentit kootaan omaksi 2x2-lohkoiksi ja ne kerrotaan myös Hadamard-matriisilla. Krominanssikomponenttien kohdalla kuitenkin käytetään 2x2-kokoista Hadamard-matriisia H_3 (kuva 12). Muunnoskoodaus tuottaa matriisin, jossa eniten merkitsevät arvot ovat keskittyneet matriisin vasempaan yläkulmaan. Tämä tehostaa kompressointia, kun matriisi luetaan zig-zag-skannauksen avulla ja sitten entropiakoodataan. [2, s. 13.]

$$H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \qquad H_3 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Kuva 12. Muunnosmatriisit H_2 ja H_3. [2, s. 13.]

4.3 Kvantisointi (Quantization)

Kvantisointi on koodauksen ensimmäinen häviöllinen vaihe H.264:ssä. Tätä ennen informaatio voidaan palauttaa täysin alkuperäiseen muotoon tekemällä residuaalimakrolohkolle käänteismuunnos ja lisäämällä siihen ennustemakrolohko.

Kvantisoinnissa poistetaan vähämerkityksellisiä arvoja muunnoskoodauksen tuloksena saadusta matriisista. Kvantisointi tapahtuu kvantisointifunktion avulla kaavan 1 tapaan. Kaavan 1 funktio sisältää kvantisointiaskeleen pituuden Q, jota voidaan säätää kvantisointiparametrin avulla. Parametrille on määritelty standardissa 52 eri arvoa joiden avulla kvantisointiaskeleen pituutta säädetään. Parametrin kasvattaminen kuudella kaksinkertaistaa askeleen pituuden. Askelpituuden kaksinkertaistaminen pienentää bittivirtaa noin puoleen. Kvantisoinnilla saadaankin tehokkaasti säädeltyä koodatun bittivirran kokoa ja samalla kuvanlaatua. [7, s. 33-35.]

$$X_q(i,j) = sign\{\mathbf{Y}(i,j)\}\frac{|\mathbf{Y}(i,j)| + f(Q_s)}{Q_s} \qquad (1)$$

missä:

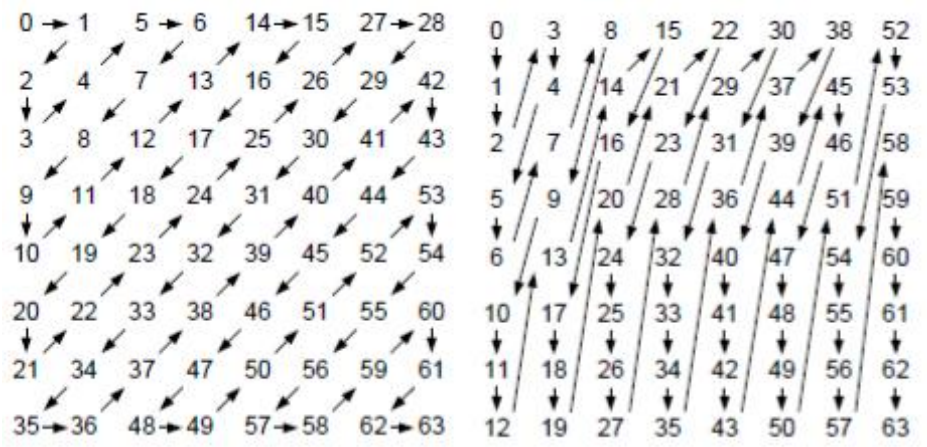
- i = lohkon rivi-indeksin arvo
- j = lohkon sarake-indeksin arvo
- Q = kvantisointiaskeleen pituus
- $f(Q_s)$ = ohjaa kvantisointia lähellä nollaa

Kvantisoinnin jälkeen makrolohkosta tehdään identtinen kopio. Alkuperäinen makrolohko siirretään entropiakoodattavaksi ja kopio siirretään dekoodattavaksi. Dekoodauksen jälkeen kopio makrolohko tallennetaan koodekin muistiin, jolloin sitä voidaan käyttää seuraavien makrolohkojen ennustamisessa (kuva 8).

4.4 Entropiakoodaus (Entropy Coding)

Ennen entropiakoodausta makrolohkon arvot luetaan lukujonoksi jommankumman H.264:n sallimista zig-zag-skannaustapojen avulla (kuva 13). Zig-zag-skannauksessa makrolohkon arvojen lukeminen aloitetaan vasemmasta

yläkulmasta ja lopetetaan oikeaan alakulmaan. Makrolohkon eniten merkitsevät arvot sijaitsevat yleensä lohkon vasemmassa yläkulmassa. Zig-zag-skannauksen avulla ne saadaan lukujonon alkuun. Lukujonon loppuarvot ovatkin yleensä vähämerkityksellisiä tai nollia. Skannauksen avulla saatu lukujono sopii hyvin näin entropiakoodattavaksi.



Kuva 13. H.264:n mahdollistamat kaksi eri 8x8-kokoisen makrolohkon lukemisesta zig-zag-skannauksen avulla. Muun kokoisille makrolohkoille skannaus on samanlainen. [5, s. 175.]

Entropiakoodauksen tarkoituksena on esittää aiempien vaiheiden tuottama videodata mahdollisimman pienellä bittimäärällä. Videodatasta ei saa kuitenkaan hävittää mitään, vaan se täytyy saada täysin alkuperäiseen muotoon dekoodattaessa. Entropiakoodauksen ideana on esittää bittijono vain vähemmän bittejä sisältävässä muodossa. Tähän pakkaamiseen H.264:ssä voi käyttää kahta eri vaihtoehtoa: ns. aritmeettista koodausta tai vaihtelevanpituista koodausta. Näitä koodauksia sovelletaan bittivirran ns. syntaksisiin elementteihin. Näitä syntaksisia elementteja ovat mm.

- kuvan, viipaleen ja makrolohkon otsakkeet
- viipaleiden hyötydata
- makrolohkon tyyppi ja koodattu lohkorakenne
- referenssikuvat
- liikevektorit
- kvantisoidut muunnoskertoimet.

4.4.1 Vaihtelevan pituinen koodaus (Variable Length Coding)

H.264:ssä vaihtelevan pituinen koodaus suoritetaan menetelmällä, jossa residuaalimakrolohkot koodataan käyttämällä CAVLC-entropiakoodausta (Context-based Adaptive Variable Length Coding) ja kaikki muut syntaksiset elementit Exp-Golomb-kooditaulukon avulla.

Exp-Golomb-kooditaulukko on kaksisarakkainen taulukko, jossa kutakin kokonaislukua vastaa säännöllisellä laskentatavalla muodostettu binäärikoodi. Binäärikoodit on muodostettu datan tilastollisen jakauman avulla, ja ne kuvaavat syntaksisen elementin arvoa. Kullekin syntaksiselle elementille valitaan elementin tyypin mukaan yksi neljästä valintatyypistä. Tyyppi kuvaa miten taulukosta valitaan oikea rivi eli arvo elementille. Esimerkiksi makrolohkolle, joka voi saada vain positiivisia arvoja, on oma tulkintataulukkonsa kuin liikevektorille, joka voi saada myös negatiivisia arvoja. Eri tyypit on määritelty niin, että usein esiintyvät elementin arvot on kuvattu lyhyillä binäärikoodeilla ja harvemmin esiintyvät pitemmillä.

Residuaalimakrolohkot koodataan vaihtelevan pituisessa entropiakoodauksessa CAVLC-entropiakoodauksen avulla. Se on muuten samanlainen kuin Exp-Golomb-koodaus, mutta CAVLC-koodauksessa taulukot muuttuvat.

Muunnoskoodatut ja kvantisoidut residuaalimakrolohkot sisältävät runsaasti nollia. Nollasta poikkeavat arvot sijaitsevat lisäksi aina bittijonon alussa johtuen zig-zag-skannauksesta. Vierekkäisten residuaalimakrolohkojen arvot myös korreloivat keskenään. Näitä tietoja hyväksi käyttäen CAVLC-koodauksessa muodostetut koodaustaulukot muuttuvat koko ajan ja pysyvät näin ajan tasalla. Näin saadaan entropiakoodattua kullekin bittivirralle ominaisella tilastollista todennäköisyyttä sisältävän taulukon avulla eikä käytetä valmiiksi määriteltyjä taulukoita.

4.4.2 Aritmeettinen koodaus (Arithmetic Coding)

H.264-standardissa aritmeettinen koodaus kulkee nimellä CABAC-koodaus (Context-based Adaptive Binary Arithmetic Coding). CABAC-koodaus on entropiakoodausmenetelmistä tehokkain H.264:ssä. Se on suunniteltu ainoastaan binääriesitysten entropiakoodaaamiseen ja se suoritetaan aina kolmessa osavaiheessa. [14.]

Ensimmäiseksi koodattavan syntaksisen elementin arvo kaksiarvoistetaan. Näin saadaan luotua uniikki binääriesitys elementille. Mikäli syntaksinen elementti on jo valmiiksi kaksiarvoinen, ei sitä muuteta. Toisessa vaiheessa valitaan koodattavalle elementille aiemmin koodattujen elementtien avulla sille sopiva malli. Mallin valinta on CABAC-koodauksen tärkein vaihe, sillä se määrittelee todennäköisyysjakauman kullekin koodattavalle symbolille. Standardissa on määritelty noin 400 erilaista mallia, joista valitaan oikea eri elementille. Malli sisältää tiedon eri symbolien bittien todennäköisyysjakauman. Tämän todennäköisyysjakauman perusteella tehdään itse aritmeettinen koodaus. [14.]

Kolmannessa vaiheessa syntaksisen elementin binääriesitys muunnetaan mallia käyttämällä aritmeettiseksi koodiksi. Koodauksen jälkeen mallia päivitetään saadun tuloksen avulla. Näin aritmeettinen koodaus pysyy adaptiivisena, eli mallit muuttuvat koodauksen mukaan. CABAC-koodaus pitääkin yllä tilastoa kunkin viipaleen osalta. [14]

4.5 Lohkosuodatin (Deblocking Filter)

Lohkopohjainen koodaus aiheuttaa epäjatkuvuutta viereisten makrolohkojen ja partitioiden reunoille. H.264:n suuri ero aikaisempiin koodausstandardeihin nähden on, että siihen on sisällytetty lohkoisuutta poistava suodatin koodausprosessiin. Aikaisemmin lohkoisuutta on poistettu dekoodausvaiheessa ulkoisen suodattimen avulla.

Lohkoisuutta poistava suodatin tuottaa paremman laadun esitettävään videokuvaan ja saa aikaan parempia ennustustuloksia. Mikäli suodattimen voimakkuudeksi valitaan nollasta poikkeava arvo eli suodatus tehdään, käy se makrolohkot läpi yksi kerrallaan. Esimerkikki kuvasta ilman lohkosuodatinta kuva 14 ja sama kuva, jossa käytetty lohkosuodatinta kuva 15. [19.]

Suodatin tarkastelee makrolohkon reunojen eri puolilla olevia luminanssikomponentteja ja vertaa niitä toisiinsa. Mikäli tietyt standardin ja suodattimen voimakkuuden määräämät reunaehdot toteutuvat, suodatus tehdään. Suodatuksessa saadut arvot vaihdetaan makrolohkon arvojen tilalle. [19.]

Suodatuksen voimakkuutta voidaan säätää viipale-, reuna- tai pikselitasolla. Näin voidaan säilyttää jo valmiiksi hyvälaatuisen kuvan terävyys tai tehostaa suodatusta, jolloin heikkolaatuiseen kuvaan saadaan lisää pehmeyttä. [19.]

Kuva 14. Kuva ilman lohkosuodatinta. [19.]

Kuva 15. Kuva, jossa käytetty lohkosuodatinta. [19.]

5 H.264-MÄÄRITYKSET

Standardissa on määritelty suuri määrä erilaisia koodaustyökaluja erilaisia bittivirtoja varten esim. videopuhelut ja HD-elokuvat. Tarkoitus ei kuitenkaan ole, että koodekit tukevat kaikkia standardin määrittelemiä bittivirtoja, koska ne poikkeavat paljon toisistaan. Koodausprosessit onkin rajattu sovellusten tarpeiden mukaan. Rajaus tapahtuu standardissa määriteltyjen profiilien ja tasojen avulla. Näin laitteet ja ohjelmistot tiettyjä sovelluksia varten ovat yksinkertaisempia ja halvempia valmistaa.

5.1 Profiilit ja tasot (Profiles and Levels)

Jokainen sovellus asettaa omat vaatimuksensa ja rajansa mm. bittivirralle ja kuvan laadulle. Myös sovelluksen käyttämä laite asettaa rajoja kuten käytet-

tävissä oleva laskentakapasiteetti ja näytön resoluutio, esim kännykässä. H.264-standardi on suunniteltu toimimaan mahdollisimman monelle jo nyt olevalle sovellukselle ja tulevaisuudessa tuleville uusille sovelluksille. H.264-standardi tarjoaa valtavan määrän eri koodaustyökaluja ja määrittää, millaista dekoodekista ulos tulevan kuvan täytyy olla. Tällaisen koodekin rakentaminen, joka tukisi kaikkia H.264-standardin määrittelemiä koodaustyökaluja ja kuvan laatuja, olisi hyvin kallista. Lisäksi tietyssä sovelluksessa käytetty koodekki tuottaisi vain sovellukselle sopivaa bittivirtaa, jolloin suuri osa sen koodaustyökaluista olisi tarpeettomia.

Koodekki on järkevämpää rakentaa sovelluskohtaisesti, jolloin ne tukevat vain tietyn sovelluksen tarvitsemia koodaustyökaluja ja kuvan laatua. Standardissa tämä rajaus eri sovellusten välille tehdään profiilien ja tasojen avulla. Näin määritellään, mitä koodaustyökaluja koodekki voi käyttää ja minkälaista bittivirtaa dekoodekki antaa ulos. Lisäksi profiilien ja tasojen avulla koodekki ja dekoodekki saadaan toimimaan keskenään. Profiilit määrittävät koodekille, mitä koodaustyökaluja se voi käyttää, ja tasot määrittelet dekoodekille, minkälaista kuvan laadun täytyy olla. [3, s.7.]

5.1.1 Profiilit (Profiles)

H.264-standardissa määritellään useita eri profiileita, jotta se saadaan sopivaksi mahdollisimman monelle eri sovellukselle. Profiileita on kolme tavalliseen videon koodaukseen ja neljä korkeatasoiselle videolle. Lisäksi löytyy neljä profiilia pelkästään liikkumattomien kuvien kompressointiin ja kolme profiilia skaalautuvaan koodaukseen. Profiileja on tullut lisää standardiin, kun H.264:stä on tehty uusia julkaisuja. Seuraavaksi käydään läpi tämän hetken, vuoden 2009 profiilit.

Baseline-profiili on suunniteltu vähäisen suorituskyvyn, pienien bittivirtojen ja huonon viiveensietokyvyn sovelluksille. Tällaisiä sovelluksia ovat mm. heikko laatuinen internetkonferenssi ja videopuhelut. Baseline-profiili tukee vain I- ja P-viipaleita, virheensietoa parantavia työkaluja ja joustavaa makroblokkien järjestystä eli FMO:ta (taulukko 1). [5.]

Main-profiili on suunniteltu lähetysympäristön sovelluksille kuten SD-laatuinen digitaalitelevisio. Se tukee I-,P- ja B-viipaleita ja CABAC-entropiakoodausta (taulukko 1). [5.]

Extended-profiili on suunniteltu internetin kautta lähettäville sovelluksille kuten elokuvien tai television lähettämiseen IP-verkon yli eli IPTV (Internet Protocol Television). Se tukee I-, P-, B-, SI- ja SP-viipaleita. Koska profiili on suunniteltu virhealttiiseen lähetysympäristöön, on siihen lisätty virheiden käsittelyä parantavia työkaluja (taulukko 1). [5.]

Taulukko 1. Baseline-, Main- ja Extended-profiilien tukemia koodaustyökaluja. [5.]

Työkalu	Baseline	Main	Extended
I-viipaleet	X	X	X
P-viipaleet	X	X	X
B-viipaleet		X	
SI-viipaleet			X
SP-viipaleet			X
CABAC		X	
CAVLC	X	X	X
Lomitettu koodaus		X	X
FMO	X		X

High-profiili määriteltiin hyvän kuvanlaadun vaativille sovelluksille kuten videon editoinnille, HD-DVD:lle ja Blu-ray-levylle sekä HDTV-käyttöön. High 10-profiili mahdollistaa 10-bittisen näytteenottotarkkuuden. High 4:2:2-profiili on määritelty ammattimaiseen käyttöön, ja se tukee 4:2:2-näytteenottoa. High 4:4:4-profiili on myös suunniteltu ammattimaiseen käyttöön, ja se tukee 4:4:4-näytteenottoa (taulukko 2). [5.]

Taulukko 2. High-profiilien sallimia koodaustyökaluja. [5.]

Työkalu	*High*	*High 10*	*High 4:2:2*	*High 4:4:4*
Kaikki samat kuin Main-profiilissa	*X*	*X*	*X*	*X*
8 bitin resoluutio	*X*	*X*	*X*	*X*
9-10 bitin resoluutio		*X*	*X*	*X*

11-12 bitin resoluutio				*X*
4:2:0 näytteenottotarkkuus	*X*	*X*	*X*	*X*
4:2:2 näytteenottotarkkuus			*X*	*X*
4:4:4 näytteenottotarkkuus				*X*
Ennustava häviötönkoodaus				*X*

Standardiin on määritelty myös ainoastaan ammattimaiseen käyttöön tarkoitettuja profiileja. Ammattimaiseen käyttöön suunnitellut profiilit: High 10 Intra-profiili, High 4:2:2 Intra-profiilit, High 4:4:4 Intra-profiili ja CAVLC 4:4:4 Intra-profiili. High Intra-profiileissa ei käytetä kuin Intra-ennustamista. Näin kamerat voivat käyttää näitä profiileja, kun ennustus on vain kuvan sisäistä. Profiilit myös soveltuvat hyvin videoneditointiin. Koodaustyökalut ovat samanlaisia kuin High-profiileissa. Ainoastaan CAVLC 4:4:4 Intra-profiiliin on lisätty High 4:4:4-profiiliin nähden vaihtelevan pituinen entropiakoodaus. [5.]

Skaalautuvan koodauksen kehittämisen aloittaminen on myös vaikuttanut H.264-standardiin. Skaalautuvuus tarkoittaa, että samasta täyden kuvalaadun bittivirrasta on erotettavissa huonompilaatuisia osabittivirtoja hylkäämällä osa lähetystä bittivirrasta hallitusti. Tästä on se hyöty, että eri nopeuksiset yhteydet pystyvät hyödyntämään samaa koodattua lähtöbittivirtaa. Huononnus voidaan tehdä resoluution, kuvapäivitysnopeuden, kuvanlaadun tai näiden yhdistelmien suhteen. [20.]

Skaalautuvaa koodausta varten H.264-standardiin on määritelty kolme eri profiilia: Skaalautuva Baseline-profiili, Skaalautuva High-profiili ja Skaalautuva High Intra-profiili.

Skaalautuva Baseline-profiili on suunniteltu kännykän vastaanottamaan videoon, keskustelu- ja valvontasovelluksiin tai muihin yksinkertaisen dekoodauksen omaaviin sovelluksiin. Skaalautuva High-profiili on suunniteltu monimutkaisempiin lähetysympäristöihin kuin skaalautuva Baseline-profiili. Skaalautuva High Intra-profiili on suunniteltu pelkästään liikkumattomien kuvien kompressointiin. [5.]

5.1.2 Tasot (Levels)

Profiilit määrittävät koodauksessa käytettävät koodaustyökalut niin tasot määrittävät rajat itse videokuvalle. Jokaiselle profiilille on määritelty useita eri tasoja. Tasojen avulla määritetään videokuvalle resoluutio, päivitysnopeus ja bittinopeus. Tasot määrittävät myös dekoodekille sen tarvitseman muistin määrän ja makrolohkojen prosessointinopeuden. Tasojen avulla määriteltyjä rajoja taulukko 3. [3.]

Taulukko 3. Tasojen määrittäviä rajoja. [12;5, s. 283.]

Taso	*Makrolohkoa/s*	*Makrolohkoa/kuva*	*Baseline-, Main- ja Extended-profiilin maksimi bittinopeus*	*High-profiilin maksimi bittinopeus*	*High 4:2:2- ja High 4:4:4-profiilin maksimi bittinopeus*
1	*1 485*	*99*	*0,064 Mbps*	*0,080 Mbps*	*0,256 Mbps*
1b	*1 485*	*99*	*0,128 Mbps*	*0,160 Mbps*	*0,512 Mbps*
1.1	*3 000*	*396*	*0,192 Mbps*	*0,240 Mbps*	*0,768 Mbps*
1.2	*6 000*	*396*	*0,384 Mbps*	*0,480 Mbps*	*1,536 Mbps*
1.3	*11 880*	*396*	*0,768 Mbps*	*0,960 Mbps*	*3,072 Mbps*
2	*11 880*	*396*	*2 Mbps*	*2,5 Mbps*	*8 Mbps*
2.1	*19 800*	*792*	*4 Mbps*	*5 Mbps*	*16 Mbps*
2.2	*20 250*	*1 620*	*4 Mbps*	*5 Mbps*	*16 Mbps*
3	*40 500*	*1 620*	*10 Mbps*	*12,5 Mbps*	*40 Mbps*
3.1	*108 000*	*3 600*	*14 Mbps*	*17,5 Mbps*	*56 Mbps*
3.2	*216 000*	*5 120*	*20 Mbps*	*25 Mbps*	*80 Mbps*
4	*245 760*	*8 192*	*20 Mbps*	*25 Mbps*	*80 Mbps*
4.1	*245 760*	*8 192*	*50 Mbps*	*62,5 Mbps*	*200 Mbps*
4.2	*522 240*	*8 704*	*50 Mbps*	*62,5 Mbps*	*200 Mbps*
5	*589 824*	*22 080*	*135 Mbps*	*168,75Mbps*	*540 Mbps*
5.1	*983 040*	*36 864*	*240 Mbps*	*300 Mbps*	*960 Mbps*

Kun bittivirralle on määritelty tietty profiili ja taso, täytyy dekoodekin myös osata käsitellä kaikkia määrittelytason alapuolella olevia tasoja.

6 H.264-KOODEKIN SUORITUSKYKY

H.264 on tämän hetken kansainvälisen videonkoodausstandardoinnin viimeisin tuotos. Sen kehitti ITU-T ja ISO/IEC yhdessä ja sen tarkoitus on korvata MPEG-2-standardi ja tarjota mahdollisuus korkearesoluutioiseen HD-videoon. H.264:ssä on tätä varten paljon laajempi tehokkaiden koodaustyökalujen valikoima kuin muissa videonkoodausstandardeissa. Näiden työkalujen avulla se avaa ovia uusille sovelluksille kuten mobiilivideolle ja IPTV:lle. Seuraavaksi käymme läpi, miten suorityskykyä ja kuvan laatua voidaan verrata eri koodausstandardien kesken. Verrataan myös itse tekemäni testin tuloksia IEEE:n tutkijoiden saamiin tuloksiin H.264-koodekin tehokkuudesta. Lisäksi nähdään miten eri koodaustyökalut vaikuttavat H.264:n kompressointisuhteeseen.

6.1 Suorituskyvyn mittaaminen

Videokuvan laatua voidaan arvioida katsojan näkökulmasta tai mittaustulosten avulla. Kumpikaan arviointimenetelmä ei anna oikeaa tai väärää vastausta. Katsoja voi pitää videokuvan laatua parempana, vaikka mittaus osoittaisikin sen huonommaksi. Mittausten tulosten kanssa täytyykin tehdä kompromisseja.

Suorituskyvyn arvioinnissa tarkastellaan kolmea osatekijää: videokuvan laatua, bittivirran suuruutta ja koodausprosessin vaatimaa laskentakapasiteettia. Nämä kaikki tekijät ovat suoraan tekemisissä toistensa kanssa. Parempaa videokuvan laatua saadaan nostamalla bittivirtaa. Pitämällä bittivirta samana, parempaa kuvanlaatua saadaan tehostamalla koodausta ja näin kuormitetaan laitteen laskentakapasiteettia. Päätökset laadun, bittivirran ja laskentakapasiteetin välillä tehdään sovelluskohtaisesti. Esimerkiksi kännykän videopuheluissa joudutaan tulemaan toimeen pienellä bittivirralla ja laskentakapasiteetilla. Tämä heikentää videokuvan laatua, mutta mahdollistaa itse palvelun ja parantaa kännykän akun kestoa. Teräväpiirto- eli HD-elokuvissa taas panostetaan videokuvan laatuun ja jätetään vähemmälle huomiolle bittivirran suuruus ja tarvittava laskentakapasiteetti.

Suorituskykyä arvioitaessa osatekijöistä laitteen vaatima laskentakapasiteetti ja bittivirta on helppo mitata ja verrata tuloksia keskenään. Videokuvan laadussa asia on päinvastoin. Toinen katsoja saattaa pitää kirkkaammasta ja terävästä videokuvasta, kun taas toinen pitää pehmeästä ja tummemmasta videokuvasta. Myös ulkopuoliset tekijät saattavat vaikuttaa hyvin paljon arviointiin.

Videokuvan laadun arviointia ja vertaamista varten on kehitetty myös täysin objektiivinen menetelmä Peak Signal-to-Noise Ratio eli PSNR. PSNR-arvo saadaan laskemalla alkuperäisen ja kompressoidun kuvan pikseleiden poikkeamien neliöiden keskiarvo. Keskiarvo suhteutetaan korkeimpaan pikseliarvoon ja muunnetaan logaritmiasteikolle. Alla esitetty kaava, jolla PSNR-arvo lasketaan (kaava 2). [3, s. 25.]

$$PSNR_{dB} = 10\log_{10}\frac{(2^n - 1)^2}{MSE} \qquad (2)$$

missä:

- PSNR = arvo desibeleinä
- MSE = pikseleiden poikkeamien neliöiden keskiarvo

PSNR-arvo lasketaan kuva kuvalta koko videolle ja siitä otetaan keskiarvolukema. Näin saadaan arvo, kuinka hyvin kompressoitu video vastaa alkuperäistä videota. Mitä suurempi arvo saadaan, sitä paremmin video vastaa alkuperäistä. PSNR-arvo on kuitenkin täysin matemaattinen, ja se saattaa ilmoittaa, että videon kuvanlaatu on erittäin hyvä, vaikka se katsojan mielestä ei sitä olisikaan. [3, s. 25.]

Myös hyvin yleinen käytetty laskentamenetelmä videon kuvanlaadulle on PSNR-Y-laskenta. Tässä menetelmässä tehdään sama PSNR-arvon laskenta, mutta vain kuvan luminanssikomponenteille. Menetelmä on huomattavasti nopeampi kuin tavallinen PSNR-laskenta, koska laskemista ja vertaamista on huomattavasti vähemmän. Tämä mahdollistaa eri standardien vertaamisen nopeasti ja luotettavasti.

6.2 H.264-koodekin suorituskyky

IEEE (Institute of Electrical and Electronics Engineers) on kansainvälinen tekniikan alan järjestö. Sen piiriin kuuluu laaja julkaisutoiminta, koulutus ja standardien määrityksissä mukanaolo. IEEE:tä voidaan pitää yhtenä maailman suurimpana ja merkittävimpänä teknillisenä järjestönä. IEEE:n jäsenet Thomas Wiegand, Gary J.Sullivan, Gisle Bjontegaard ja Ajay Luthra ovat tehneet testejä videokompressoinnista. Heidän saamiinsa testituloksiin perehdytään seuraavaksi. [1.]

Testissä testattiin neljää eri kansainvälisesti standardoitua koodekkia: MPEG-2, H.263, MPEG-4 part 2 ja H.264. Testit tehtiin kolmella eri sovellusalueella: streaming-videolla, videopuhelulla ja viihdevideolla.

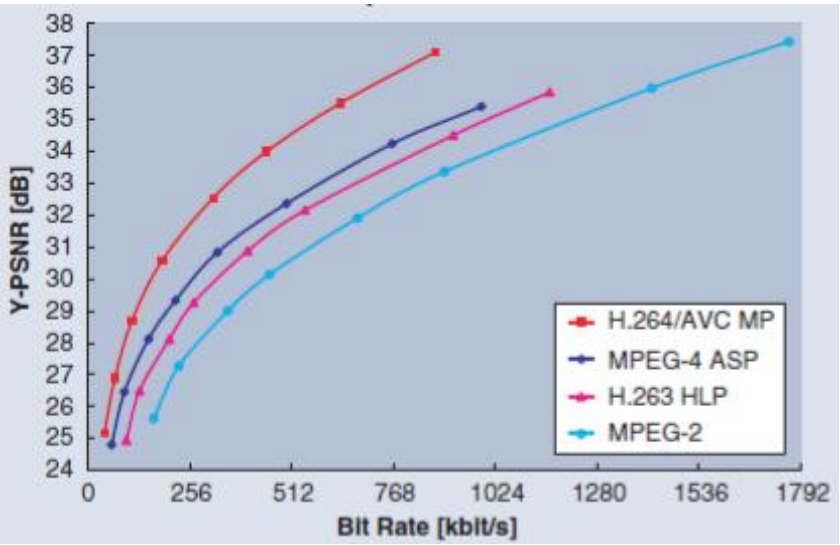
Streaming-videossa kaikki koodekit säädettiin käyttämään perusproliifiaan. H.264:ssä käytettiin Main-profiilia ja CABAC-entropiakoodausta. H.264 oli odotusten mukaisesti muita huomattavasti tehokkaampi kompressoimaan. Se saavutti 64 %:n bittivirran säästön MPEG-2:n verrattuna, 48 % säästön H.263:n ja 37 % MPEG-4 part 2:een verrattuna. [1.]

Videopuhelussa on tärkeää koodauksen reaaliaikaisuus ja mahdollisimman pieni viive. Koodekit optimoitiin sopimaan videopuhelun vaatimusten mukaan. H.264:ssä otettiin käyttöön Baseline-profiili, joka on profiileista yksinkertaisin. MPEG-2:sta ei ole ollenkaan suunniteltu videopuhelun alhaisille bittivirroille, joten se jätettiin kokonaan pois testin tästä vaiheesta. H.264 oli keskimäärin 29 % MPEG-4 part 2:sta parempi ja 28 % H.263:sta parempi. [1.]

Viihdevideotestissä testattiin 720 x 576 resoluution DVD-videota ja 1280 x 720 resoluution HD-videota. MPEG-4 part 2:ta ja H.263:ta ei otettu mukaan ollenkaan tähän testin osaan, sillä niitä ei yleisesti käytetä ollenkaan viihdesovelluksissa. H.264 saavutti keskimäärin 45 % säästön bittivirrassa verrattuna MPEG-2:n DVD-resoluutiolla ja HD-resoluutiolla noin 25 – 45 % säästöä. Parhaimmillaan käyttämällä H.264:ssä High-profiilia päästiin jopa noin 60 %:in bittivirran säästöön. [1.]

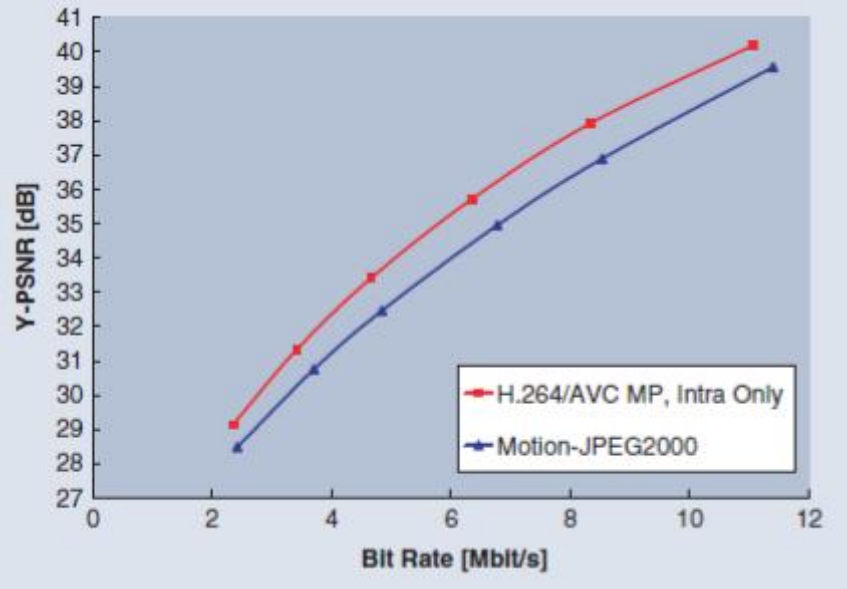
Myös IEEE:n jäsenet Jörn Ostermann, Jan Bromans, Peter List, Detlev Marpe, Matthias Narroschke, Fernando Pereira, Thomas Stockhammer ja Thomas Wedi ovat tehneet testejä H.264:n suorituskyvystä. Jäsenet vertasivat

CIF-videota H.264:n, MPEG-4 Visual ASP:n, H.263 HLP:n ja MPEG-2:n kesken. CIF-video on 352 x 288-resoluutioista ja käytettävä framerate oli 15. Testissä laskettiin vain luminanssikomponentin erot eli PSNR-Y-arvo. Kaikissa standardeissa valittiin videostreamaukseen sopiva profiili ja taso. Jäsenet saivat testissään seuraavia tuloksia. H.264 saavutti noin 63 %:n säästön bittivirrassa MPEG-2:n verrattuna ja noin 35 % säästön H.263 HLP:hen ja MPEG-4 ASP:in verrattuna (kuva 16). [2]



Kuva 16. IEEE:n jäsenten tuloksista piirretty kuvaaja H.264, MPEG-4 ASP, H.263 HLP ja MPEG-2:n välillä. Vertailuvideona käytetty CIF-videota. [2.]

Ostermann ja kumppanit tekivät myös testin H.264:llä vain intrakoodausta käytettäessä. Intrakoodaus on yleinen videokameroissa, koska ennustamista ei voi tehdä toisiin kuviin nähden. He vertasivat H.264:n kompressiotehokkuutta vain intrakoodattujen kuvien kompressointiin suunniteltuun Motion-JPEG2000-standardiin. Hämmästykseksi huomattiin H.264:n pystyvän parempaan kompressointiin intrakoodatussa videossa kuin niihin suunniteltu Motion-JPEG2000-standardi (kuva 17). [2.]



Kuva 17. H.264:n suorituskyky ja Motion-JPEG2000 suorituskyky intrakoodatussa videokuvassa. [2.]

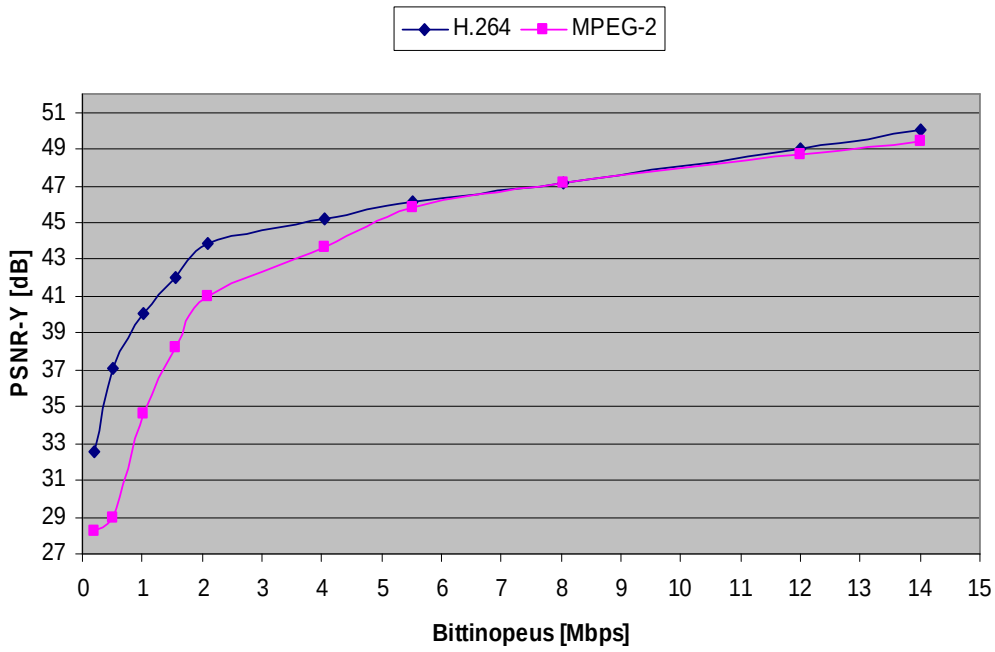
6.2.1 Testi

Jotta saataisiin vertailevia tuloksia IEEE:n jäsenten testeihin, suoritin myös itse testin H.264:n ja MPEG-2:n välillä. Testissä käytin 768 x 576-resoluution SD-videota 25:n fps:llä (Frames Per Second). Videoksi valitsin paljon liikkuvia yksityiskohtia sisältävän videon. Tälläinen video on koodekille hyvin vaikeasti kompressoitava. Testeissä käytin Adobe Premiere 9:ää videon kompressoinnissa ja MSU Video Quality Measurement Tool-ohjelmaa PSNR-Y-arvon laskemisessa. H.264:ssä oli käytössä Main-profiili ja 3.1-taso. H.264-videon lisäksi kompressoitiin normaaleilla koodaustyökalujen avulla eli en siis käyttänyt CABAC-entropiakoodausta. MPEG-2-videossa oli käytössä main-profiili ja main-taso. Kaikki muut asetukset annettiin olla oletusarvoina. Testissä kompressoitiin video molemmilla standardeilla käyttämällä eri maksimibittinopeuksia.

6.2.2 Tulokset

Saaduista tuloksista kuvasta 18 näkyy hyvin kuinka paljon parempaan kompressointikuvanlaatu suhteeseen objektiivisesti H.264 pystyy. Erityisesti pieniä bittinopeuksia käytettäessä tuloksissa on suuria eroja. Suurempiin bittinopeukseen mentäessä tulokset tasaantuvat ja MPEG-2 pystyy melkein yhtä hyvään tulokseen kuin H.264.

Huomasin myös H.264:n pakkaavan videon pienempään tilaan kuin MPEG-2:n, pienillä alle 5 Mbps:n bittinopeuksilla. Säästöä 30 sekunnin pituisessa videon tallentamisessa levylle tuli noin 30%. Molemmissa kuitenkin säädin maksimibittinopeudet aina samaan arvoon. H.264 käytti siis vähemmän maksimikaistaa kuin MPEG-2:n. H.264 soveltuukin hyvin lähetyssovelluksiin, joissa yleensä vaaditaan mahdollisimman pienellä kaistalla mahdollisimman hyvää videokuvan laatua.

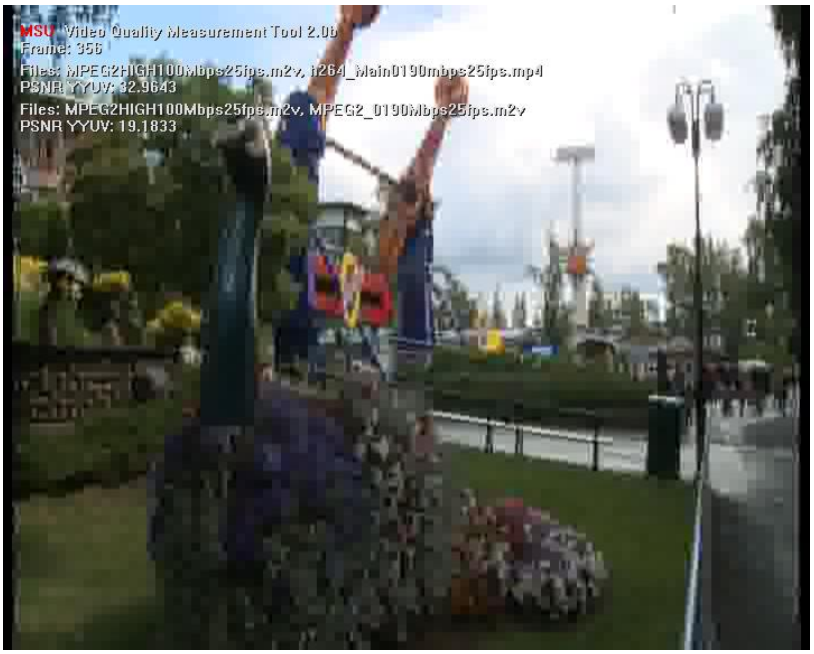


Kuva 18. H.264:n ja MPEG-2:n suorituskyky SD-videossa. Itse suorittamassani testissä.

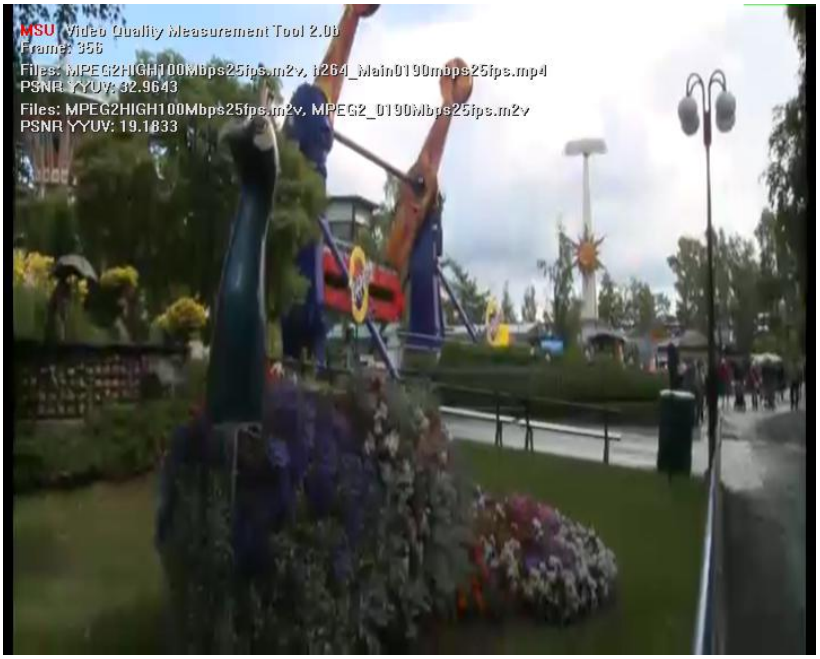
Tulokset vastaavat paljon IEEE:n jäsenten saamia tuloksia. H.264 pystyi alle 4 Mbps bittinopeuksilla noin 50% parempaan kompressointisuhteeseen kuin MPEG-2:n. Vielä parempaan tulokseen H.264 pystyy, kun mennään vielä pienempiin bittinopeuksiin.

Otin myös subjektiivista vertailua varten esimerkki framejä vertailtavaksi. Subjektiivisesti arvioituna H.264 teki videokuvasta paljon pehmeämpää kuin MPEG-2:n. MPEG-2:n jätti makrolohkojen välille selviä eroja ja kuvan objektit näytti hyvin neliömäisiltä. Pienillä alle 1 mbps nopeuksilla ei MPEG-2:n videosta tahtonut saada mitään selvää. Kuitenkin jo 0,190 Mbps nopeuksisesta H.264 videosta erottui hyvin video kokonaisuutena. Yksityiskohdat kuiten-

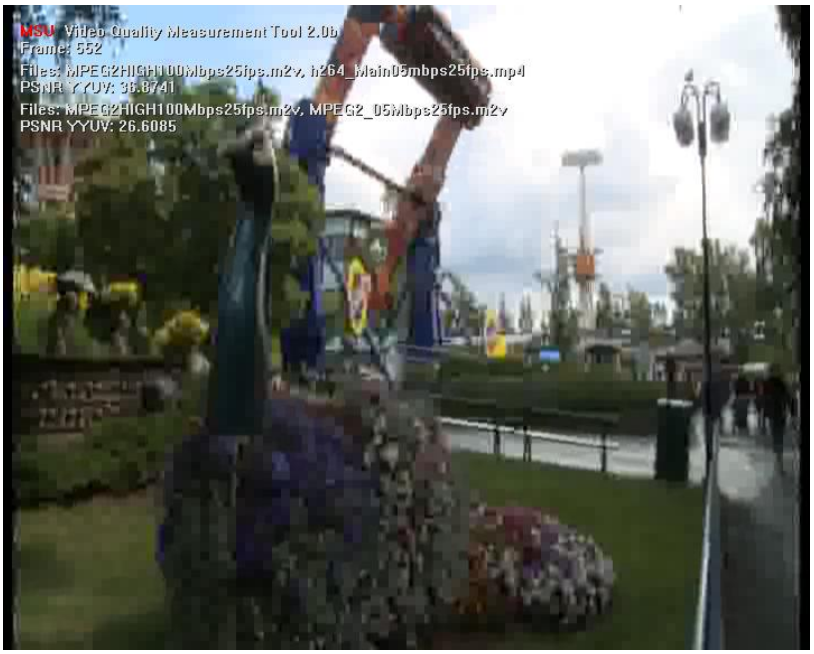
kin siinäkin sotkeutuivat. MPEG-2:ssa vasta mentäessä yli 1 Mbps videoon kuva alkoi muuttua subjektiivisesti mielestäni katsottavaksi, yksityiskohdat sotkeentuivat kuitenkin vieläkin liikaa. Kun taas 0,5 Mbps nopeuksisesta H.264-videosta pystyi tunnistamaan ihmiset ja se vastasi subjektiivisesti mielestäni parempaa kuin 2 Mbps MPEG-2 video. Bittinopeuksien kasvaessa yli 5 Mbps subjektiivisesti kuvissa oli mielestäni hyvin vaikea huomata eroja. Alla esimerkkiotoksia MPEG-2- ja H.264-videosta 0,190 Mbps, 0,5 Mbps, 1,01 Mbps ja 2,01 Mbps bittinopeuksilla (kuvat 19-26).



Kuva 19. Esimerkkiotos MPEG-2-videosta, jossa bittinopeus 0,190 Mbps.



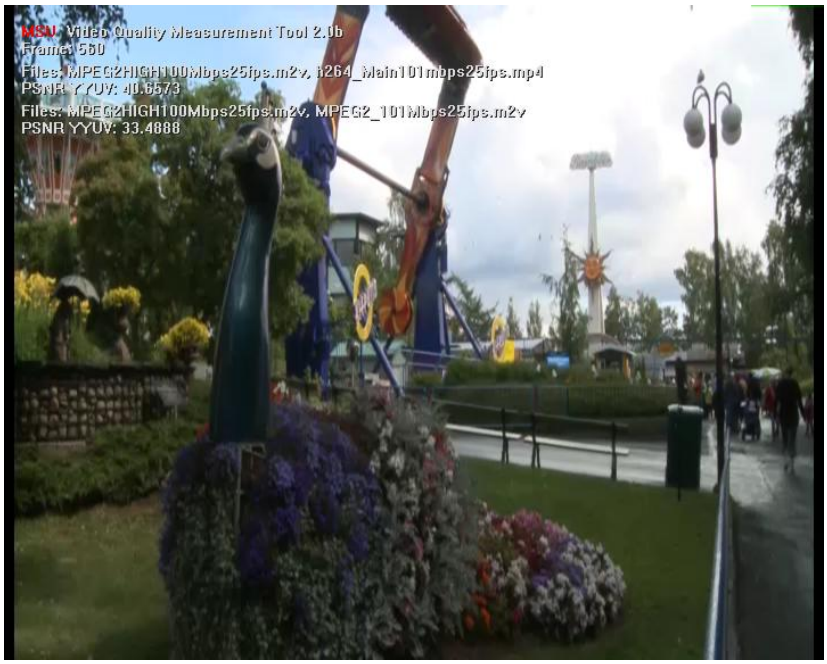
Kuva 20. Esimerkkiotos H.264-videosta, jossa bittinopeus 0,190 Mbps.



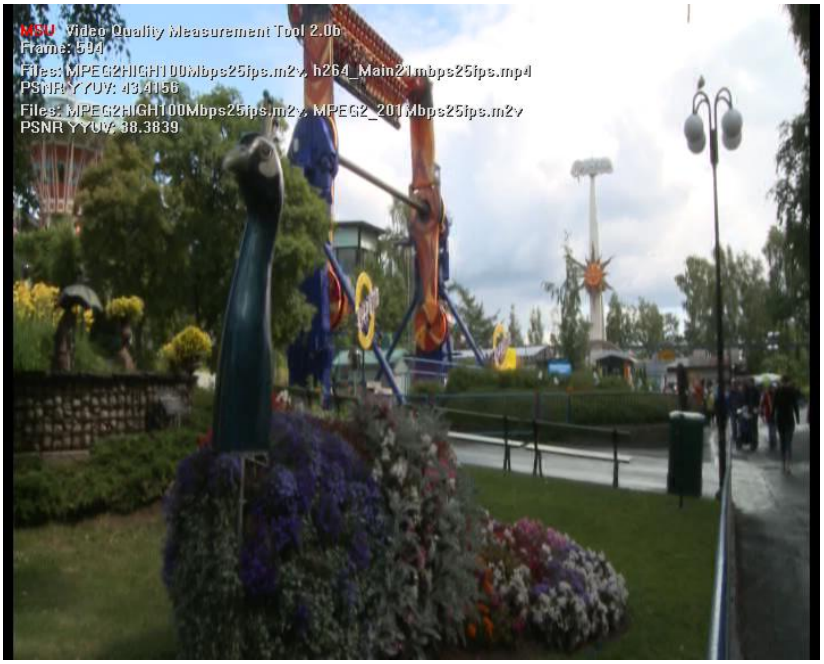
Kuva 21. Esimerkkiotos MPEG-2-videosta, jossa bittinopeus 0,5 Mbps.

Kuva 22. Esimerkkiotos H.264-videosta, jossa bittinopeus 0,5 Mbps.

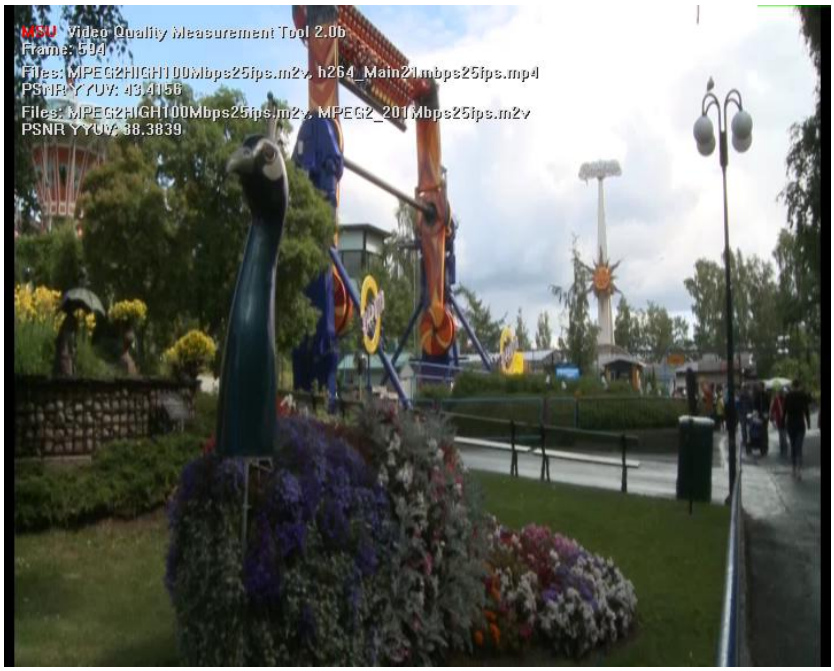
Kuva 23. Esimerkkiotos MPEG-2-videosta, jossa bittinopeus 1,01 Mbps.



Kuva 24. Esimerkkiotos H.264-videosta, jossa bittinopeus 1,01 Mbps.



Kuva 25. Esimerkkiotos MPEG-2-videosta, jossa bittinopeus 2,01 Mbps.



Kuva 26. Esimerkkiotos H.264-videosta, jossa bittinopeus 2,01 Mbps.

6.3 Koodaustyökalut ja suorituskyky

H.264-standardissa on määritelty paljon eri koodaustyökaluja. Profiilit määrittävät, mitä työkaluja milloinkin voidaan käyttää. Kuitenkaan kaikkia ei kannata aina ottaa käyttöön, vaan valinta pitää tehdä sovelluskohtaisesti. Yleisesti ottaen koodaustyökalut parantavat kompressointisuhdetta ja näin pienentä-

vät bittivirtaa. Väärin valittuina työkalut voivat kuitenkin lisätä bittivirtaa. Seuraavaksi perehdymme IEEE:n tutkijoiden Jörn Ostermannin, Jan Bormansin, Peter Listin, Detlev Marpen, Matthias Narroschken, Fernando Pereiran, Thomas Stockhammerin ja Thomas Wedin tekemään testiin, jossa tutkittiin eri koodaustyökalujen vaikutusta bittivirran suuruuteen. Testissä oli käytössä koko ajan H.264-koodekki ja sama video, jossa pidettiin sama laatutaso. Näin saadaan selville miten eri koodaustyökalut vaikuttavat videon pakkaamiseen H.264:ää käytettäessä. He päätyivät testissään seuraaviin lopputuloksiin: [2.]

- Makrolohkojen jakaminen partitioihin ja alipartitioihin pienenpää bittivirtaa tyypillisesti 4–20 %.
- B-viipaleiden käyttöönotolla pienennettiin bittivirtaa parhaimmillaan 10 %.
- CAVLC pienentää bittivirtaa noin 30 % ja CABAC-entropiakoodauksen käyttö vähensi bittivirtaa parhaimmilaan 10 % CAVLC:n verrattuna.
- Lohkoisuutta poistava suodatin vähensi bittivirtaa noin 9 %.

Yhteensä testin lopputuloksia tarkastellessa täytyy myös muistaa eri koodaustyökalujen tuoman monimutkaisuuden ja laskennan lisääntymisen koodaukseen. Laskentamäärä kasvoi monen työkalun kohdalla yli 10 %, jolloin laskenta-aika myös pitenee. Lisäksi laskentakapasiteetti on monessa sovelluksessa rajoitettu laitteen mukaan, joten kaikkia koodaustyökaluja ei pystytä käyttämään, vaikka profiili sen sallisikin. [2.]

7 YHTEENVETO

H.264 on kansainvälisen videonkoodausstandardoinnin viimeisin tuotos, JVT-työryhmän työstämä tehokas koodekki. Se on kehitetty korvaamaan MPEG-2-standardi teräväpiirtovideon lähetys- ja jakeluapuvälineenä. H.264 tarjoaa myös tehokkaan työkaluvalikoiman uusille sovelluksille, mm. kännykkävideolle ja IPTV:lle.

H.264 on lohkopohjainen videonkoodausstandardi, kuten aikaisemmatkin ITU-T:n ja ISO/IEC:n työstämät videonkoodausstandardit. Lohkopohjaisessa koodauksessa koodattava kuva jaetaan 16x16-pikselin kokoisiin makrolohkoihin ja koodataan lohko kerrallaan. H.264-standardi mahdollistaan muista videonkoodausstandardeista poiketen makrolohkon jakamisen partitioihin ja alipartitioihin. Itse makrolohkot ryhmitellään makrolohkojoukoiksi, joita kutsutaan viipaleiksi. Viipale kertoo mitä koodausmenetelmiä makrolohkojoukon koodauksessa on käytetty. Yhden tai useamman viipaleen avulla dekoodekki muodostaa näkyvän kuvan.

Koodausprosessin ensimmäisessä osavaiheessa makrolohkolle luodaan ennustemakrolohko ennustamalla makrolohkon informaatiota, joko Intra- tai Inter-ennustamalla. Toisessa osavaiheessa makrolohkosta vähennetään ennustamisen avulla muodostettu ennustemakrolohko ja saadaan residuaalimakrolohko. Seuraavaksi residuaalimakrolohko muunnoskoodataan ja kvantisoidaan. Viimeisessä osavaiheessa koodattu residuaalimakrolohko ja muu koodausprosessissa syntyvä dekoodauksessa tarvittava informaatio entropiakoodataan.

Koodattu bittivirta voidaan dekoodata standardin määrittelyt täyttävällä dekoodekilla. Koodekin tuottaman bittivirran muoto on määritelty yksiselitteisesti standardissa yhdessä dekoodausprosessin kanssa. Näin varmistetaan laite- ja ohjelmistovalmistajien koodekkien ja dekoodekkien yhteensopivuus. Sovellusten vaatimat tietyt rajaukset koodauksessa ja kuvanlaadussa määritellään H.264-standardin profiilien ja tasojen avulla.

Lopuksi tarkasteltiin H.264-koodekin suorituskykyä muihin tunnettuihin koodekkeihin verrattuna. Tarkastelu tehtiin IEEE:n tutkijoiden tekemien testien tuloksien ja itse tekemäni testin tuloksien avulla. JVT-työryhmä oli asettanut H.264-standardisoinnin alussa yhdeksi vaatimukseksi, että H.264:n täytyy

pystyä puolet tehokkaampaan videopakkaamiseen kuin sen hetkiset aikaisemmat videonpakkausstandardit, kuitenkin kuvanlaadun säilyessä. Tuloksia tutkimalla huomattiin, että JVT:n asettama vaatimus toteutuukin varsin hyvin. Varsinkin pienemmillä bittivirroilla H.264-koodekki pystyy huomattavasti parempaan kuvanlaatuun puolet pienemmillä bittinopeuksilla. Suuremmilla yli 4 Mbps bittinopeuksilla paremmuus muihin koodekkeihin nähden pienenee. Kuitenkin käyttämällä H.264:ssä High-profiilia ja pakkaamalla HD-videota voidaan saavuttaa jopa 60% säästö bittivirrassa MPEG-2:n nähden.

H.264 on tämän hetken tehokkain koodekki ja se tulee syrjäyttämään muut kilpailijansa kokonaan. Sen tarjoaman valtavan koodaustyökalujen määrän avulla se soveltuu erinomaisesti niin pienen bittinopeuksien sovelluksiin kuin HD-tasoisiin sovelluksiin. H.264 tulee tulevaisuudessa avaamaan täysin uusia mahdollisuuksia eri sovelluksille, niin tulevaisuuden televisiossa eli IPTV:ssä kuin muissakin videonjakelun ja -pakkaamisensovelluksissa.

VIITELUETTELO

[1] Wiegand, T. - Sullivan, G.J. - Bjontegaard, G. - Luthra, A. 2003. Overview of the H.264/AVC Video Coding Standard. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 7, July 2003, 560-576.

[2] Ostermann, J. - Bormans, J. - List, P. - Marpe, D. - Narroschke, M. - Pereira, F. - Stockhammer, T. - Wedi, T. 2004. Video Coding with H.264/AVC: Tools, Performance and Complexity. IEEE Circuits and Systems Magazine, first quarter 2004, 7-28.

[3] Kahilainen, Juha. Turun Yliopisto. H.264/AVC-videonkoodaja: rakenne, toimintaperiaate, menetelmät ja suorituskyky. LuK-tutkielma. 2006.

[4] Wikipedia, AVCHD [verkkodokumentti, viitattu 18.11.2009]. Saatavissa http://en.wikipedia.org/wiki/AVCHD.

[5] ITU-T 2007. ITU-T Recommendation H.264. Advanced video coding for generic audiovisual services, Series H: Audiovisual and Multimedia Systems, Infrastructure of audiovisual services - Coding of moving video 11/2007.

[6] Vcodex, H.264 [verkkodokumentti, viitattu 18.11.2009]. Saatavissa http://www.mcuol.com/download/upfile/20071012093928_H%5B1%5D.264_MPEG-4%20Part%2010%20White%20Paper.pdf.

[7] Hanhijärvi, Jussi. H.264 'MPEG4/10 AVC Tutkielma eräästä videonkoodausstandardista, 2009.

[8] Hannuksela, Miska M. Tampereen teknillinen yliopisto. Error-Resilient Communication Using H.264/AVC Video Coding Standard, 2009.

[9] Lee, Jae-Beom - Kalva, Hari. Springer. The VC-1 and H.264 Video Compression Standards for Broadband Video Services, 2008.

[10] Richardson, I.E.G. 2003. H.264 and MPEG-4 video compression. Video coding for next-generation multimedia. West Sussex: John Wiley & Sons.

[11][Kovanen, Mika. MPEG-4 ja H.264 videonkoodausmenetelmät teräväpiistovideossa. Insinöörityö 2007.

[12] Wikipedia, H.264 [verkkodokumentti, viitattu 18.11.2009]. Saatavissa http://en.wikipedia.org/wiki/H.264.

[13] Stockhammer, T. – Hannuksela, M. H.264/AVC Video for Wireless Transmission, 2005. IEEE Wireless Communications.

[14] Erik can Bilsen AudioVisuality, CABAC [verkkodokumentti viitattu 18.11.2009].Saatavissa http://bilsen.com/index.htm?http://bilsen.com/aic/CABAC.htm.

[15] Benoit, H. Focal Press. Digital Television Third edition.

[16] Wikipedia, VCEG [verkkodokumentti, viitattu 16.11.2009]. Saatavissa http://en.wikipedia.org/wiki/VCEG.

[17] Wikipedia, MPEG [verkkodokumentti, viitattu 16.11.2009]. Saatavissa http://en.wikipedia.org/wiki/MPEG.

[18] Afterdawn, YCrCb [verkkodokumentti, viitattu 16.11.2009]. Saatavissa http://fin.afterdawn.com/sanasto/selitys.cfm/ycbcr.

[19] Vcodex, Deblocking filter [verkkodokumentti, viitattu 25.11.2009]. Saatavissa http://www.vcodex.com/files/h264_loopfilter.pdf.

[20] Fraunhofer, The Scalable Video Coding Amendment of the H.264/AVC Standard [verkkodokumentti, viitattu 25.11.2009]. Saatavissa http://ip.hhi.de/imagecom_G1/savce/index.htm.

[21] Wikipedia, YCrCb [verkkodokumentti, viitattu 17.11.2009]. Saatavissa http://en.wikipedia.org/wiki/Ycrcb.