KOHTEENSEURANTA KUVASARJASSA ALUEENLAA-JENNUSTEKNIIKALLA

Tero Koistinen, 125276

24.10.2003

Joensuun yliopisto

Tietojenkäsittelytiede

Pro gradu -tutkielma

Tiivistelmä

Alueenlaajennus on kuvankäsittelytekniikka, jossa digitaalinen kuva jaetaan alueisiin yhdistämällä kuvapisteitä yhtenäiseksi alueeksi niiden väriarvon perusteella. Tutkielmassa etsittiin vastausta siihen, onko kolmiulotteinen alueenlaajennustekniikka parempi vaihtoehto kohteen seurannan menetelmänä kuin kaksiulotteinen. Kolmiulotteisessa alueenlaajennuksessa kuvasarjan kuvat muodostavat peräkkäin aseteltuina väriltään homogeenisen kappaleen. Työssä yleistetään 2D alueenlaajennustekniikka 3D tapaukseen kuvasarjojen tapauksessa ja sovelletaan kohteenseurantaan.

Testaamiseen toteutettiin ohjelma, jossa voitiin valita alueenlaajennustekniikka sekä sille erinäisiä parametreja. Ohjelmaa kokeiltiin kolmella eri videoleikkeellä ja saadut tulokset taulukoitiin.

Koska videoleikkeiden määrä oli pieni, ei ehdotonta varmuutta menetelmän paremmuudesta voi saada. Kuitenkin tulosten perusteella voi todeta, että 3D-menetelmä vaikuttaa toimivammalta ratkaisulta, kun kohteen alueenlaajennuksella peittämät alueet ovat päällekkäiset peräkkäisissä kuvakentissä. Muussa tapauksessa 3D-menetelmä hukkaa kohteen.

ACM-luokat (ACM Computing Classification System, 1998 version): I.4.6, I.4.8

Avainsanat: kohteenseuranta, alueenlaajennus

Sisältö

1 JOHDANTO **1**

2 KOHTEEN SEURANTA **2**

2.1 Liikkeen tunnistaminen 3

2.2 Seuranta 6

2.2.1 Tekniikat 6

2.2.2 Kohteen löytäminen 8

2.2.3 Ongelmat 9

2.3 Sovelluksia 11

2.3.1 Valvonta 11

2.3.2 MPEG 13

3 ALUEENLAAJENNUS **17**

3.1 Algoritmi 17

3.2 Aloituspisteen valinta 19

3.3 Laajentumiskriteerit 21

3.3.1 Staattiset kynnysarvot 22

3.3.2 Dynaamiset kynnysarvot 27

3.4 Jälkiprosessointi 28

4 AIKADIMENSIOINEN LAAJENNUS **30**

4.1 Algoritmeja 30

4.2 Laajentumiskriteerit 34

5 ALUEENLAAJENNUSTEKNIIKOIDEN VERTAILUA **35**

5.1 Vertailuohjelma 35

5.2 Tuloksia 37

5.2.1 Taulukoiden tulkinta 38

5.3 Johtopäätöksiä 44

6 YHTEENVETO **46**

Viitteet **47**

1 JOHDANTO

Tutkielma käsittelee videokuvassa esiintyvän kohteen seuraamista alueenlaajennusalgoritmin puitteissa. Alueenlaajennuksesta lähdekirjallisuutta on niukalti, mutta menetelmä onkin varsin yksinkertainen. Lähdekirjallisuudessa mainittiin usein erilaisia videopakkausmenelmiä, koska videokuvan siirto vaatii paljon tiedonsiirtokapasiteettia. Henkilökohtaiset päätelaitteet (mediapuhelin, PDA) vaativat videon toistoon pientä kaistanleveyttä, mutta kohtuullisen hyvää erotuskykyä, jolloin pakkauksen tiiviydellä on suuri merkitys. Kuvasarja pystytään pakkaamaan tehokkaasti, jos peräkkäisien kuvien välillä ei siirretä samaa tietoa moneen kertaan.

Tässä työssä kuvalla tarkoitetaan videoleikkeen yhtä kuvaa. Kuvasarjalla tarkoitetaan kuvien joukkoa, jotka on ajan suhteen peräkkäisessä järjestyksessä. Kuva koostuu kuvapisteistä eli *pikseleistä*. Jokainen pikseli muodostuu yleensä kolmesta *värikomponentista*, joiden muodostamaa joukkoa kutsutaan *väriavaruudeksi*. Esimerkiksi RGB-väriavaruus muodostuu punaisesta (*R*ed), vihreästä (*G*reen) ja sinisestä (*B*lue) kahdeksanbittisestä värikomponentista.

Tässä työssä tutustutaan kohteen seurantamenetelmiin (*object tracking*), alueenlaajennustekniikkaan ja siihen tehtyyn laajennokseen. Alueenlaajennus on kuvan segmentointimenetelmä, jolla kuva jaetaan sisäisesti samankaltaisiin alueisiin. Työssä tutkitaan kahden eri aluelaajennustekniikan soveltuvuutta kohteen seurantaan kuvasarjassa. Koska kirjallisuudesta ei tuloksia aiheesta löytynyt, rakennettiin sovellus, jolla voidaan testata tekniikoiden eroa erilaisilla videoleikkeillä.

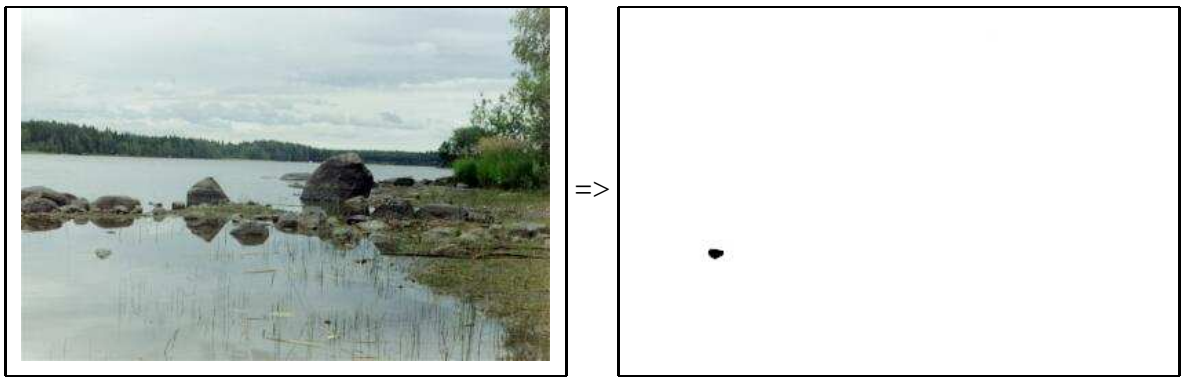
Luvussa kaksi esitellään kohteen seurannan periaatteita ja kaksi aiheeseen liittyvää esimerkkiä. Luku kolme kuvailee alueenlaajennustekniikkaa ja luvussa neljä esitetään 2D-kohteenseuranta-algoritmi ja sille laajennus, joka käsittelee alueen laajentumista kuvasarjan peräkkäisissä kuvissa. Kappaleessa viisi vertaillaan kolmella eri videoleikkeellä kohteen seurannan onnistumista ja kerrotaan tulokset ja niistä vedetyt johtopäätökset. Luvussa kuusi on yhteenveto.

2 KOHTEEN SEURANTA

Kohteen seurannalla pyritään löytämään ja seuraamaan kuvasarjassa liikkuvia kohteita. Kuvasarjalla tarkoitetaan ajallisesti vakiovälein otettuja kuvia, esimerkiksi tavallista videokameraotosta. Kohde on kuvasarjassa oleva olio, joka liikkuu suhteessa kuvan reunoihin tai taustaan. Se voi olla esimerkiksi tiellä liikkuva auto tai vartioidulla alueella käyskentelevä henkilö. Tosiaikainen kohteenseuranta vaatii runsaasti laskentatehoa ja muistia (Che & Kang 1998) johtuen suuresta tietomäärästä ja kuvan jatkuvasta muuttumisesta.

MPEG (Motion Picture Experts Group) on ryhmä eri tahoja, jotka yhdessä kehittelevät standardeja äänen ja videon pakkaamiseen (Martínez 2002). Eri pakkaustavat ovat saaneet nimensä ryhmän lyhenteen mukaan. MPEG koodauksessa pyritään minimoimaan siirrettävän tiedon määrä ja yksi keino on erottaa liikkuvat kohteet taustasta, jolloin yksinkertaisessa tapauksessa riittää siirtää tausta kerran ja muuttuva tieto (kohteet) tarpeen mukaan.

MPEG-4 koodauksessa kuvasarjasta pyritään löytämään liikkuvat kohteet, jotka erotellaan eri kohdetasoille (*object planes*). Kohdetaso on tietorakenne, joka sisältää pelkästään kohteen tiedot. Se voi olla esimerkiksi binäärikuva, josta kaikki kohteen rajaaman alueen ulkopuolella kuuluu taustaan (kuva 1).



Kuva 1: Kohdetaso kuvasarjan kuvassa. Vasemmalla kuvasarjan yksittäinen kuva ja oikealla seurattavan kohteen kohdetaso.

Useissa tapauksissa jokainen kuvasarjan kuva pyritään *segmentoimaan* kohteiden löytämiseksi. Segmentoinnissa kuvasta erotellaan jollain kriteereillä (esim. väri) yhtenäisiä alueita (Mirmehdi & Petrou 2000). Kuvassa 2 alkuperäisen kuvan (kuva 1) värien määrä on pudotettu neljään, jolloin suunnilleen samanväriset alueet on ilmaistu yhdellä värillä.

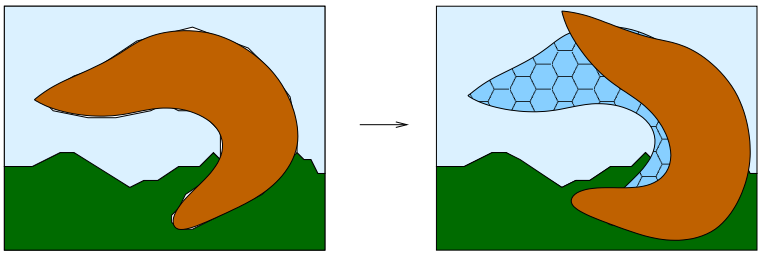
Segmentointi ei kuitenkaan ole suorituskyvyn kannalta tehokasta, koska sen laskeminen vie

Kuva 2: Segmentoitu kuva, kun kuvan värien määrä on vähennetty neljään.

paljon aikaa. Liikkuvien kohteiden löytämiseksi voi käyttää esimerkiksi eroja kuvasarjan kuvien välillä (Wang *et al.* 2000). Jäljempänä kuvataan menetelmiä liikkeen tunnistamiseen, sitten erinäisiä ongelmia liittyen seurantaan ja viimeisenä esitellään muutamia sovelluksia.

2.1 Liikkeen tunnistaminen

Liikkeellä tarkoitetaan kohteen liikkumista taustan suhteen. Liike voi olla kohteen siirtymistä, pyörimistä tai niiden yhdistelmä (kuva 3). Kohde voi myös muuttaa muotoaan.



Kuva 3: Liike siirtymänä ja pyörimisenä.

Liikkeen tunnistaminen tarkoittaa liikkuvan kohteen löytymistä kuvasta tai kuvasarjasta. Tällöin kappale liikkuu ja/tai pyörii suhteessa taustaan.

Liikkeen tunnistuksen tuloksena pyritään löytämään seurattava kohde tai kohteet. Yleisessä tapauksessa liikkeen tunnistamiseen on useita tapoja. Vaikeutta lisää kohteiden liikkumisen aiheuttama taustan peittyminen ja paljastuminen sekä kohteiden keskinäiset päällekkäisyydet. Kuvassa 3 esiin tullut tausta on korostettu.

Jako erilaisiin liikkeen tunnistamistapoihin voidaan tehdä esimerkiksi seuraavilla tavoilla,

joissa lohkolla tarkoitetaan mielivaltaisen muotoista aluetta kuvasta, yksinkertaisimmillaan suorakaidetta (Huang *et al.* 2002):

- Parametriton lohko: kuvasta otetaan lohko, jolle etsitään vastaava lohko seuraavasta kuvasta.
- Parametrinen liikemalli: kuvan lohko kuvataan muutamalla parametrilla (esim. liike, kiertymä ja skaalaus).
- Gradientti-pohjainen malli: muodostetaan liikevektorikenttä, ns. optisen vuon kuvaus, joka perustuu kuvan intensiteetin muutokseen kuvien välisen ajan ollessa vakio.

Liike tunnistetaan edellisillä tavoilla, jos löytyy lohko, jonka sijainti on muuttunut vierekkäisten lohkojen suhteen. Mikäli muutos on sama kaikille kuvan lohkoille, voidaan olettaa, että kameraa liikutetaan liikkeen suunnassa. Lohko täsmätään seuraavassa kuvassa aloittaen liikkeen ennustajan tuottamasta kohdasta ja laajentamalla etsintäaluetta tarpeen mukaan.

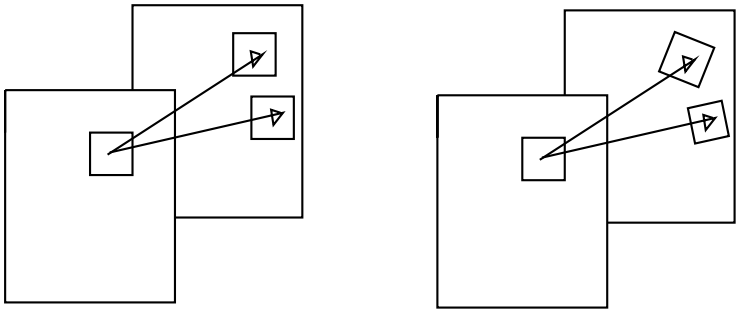
Parametrittomassa täsmäämisessä lohko täsmätään vertaamalla vastaavia pikseleitä toisiinsa. Parametrillisessa täsmäämisessä lohkoa voidaan muuttaa siirtämällä, kiertämällä ja skaalaamalla suhteessa johonkin pisteeseen. Jos kuvat ovat samanlaiset, ei muutoksia ole ja siten ei liikkuvia kappaleita.

Optinen vuo kuvaa pisteen liikettä ajallisesti jatkuvan kuvasarjan kuvien välillä. Ajallinen jatkuvuus tarkoittaa, että kuvasarjan kuvat ovat ajan suhteen peräkkäisessä järjestyksessä.

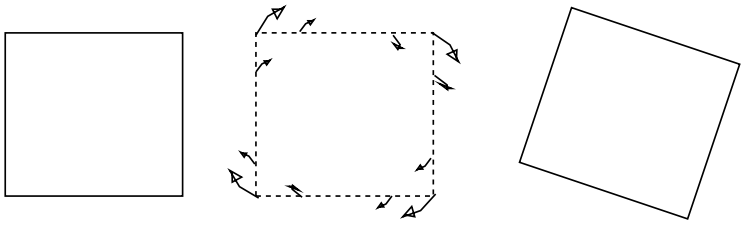
Kuvassa 4 on esitetty parametriton ja parametrillinen lohkon täsmäys. Vasemmanpuoleisen kuvan tapauksessa on esitetty kaksi eri kohtaa, joihin alkuperäisen kuvan aluetta täsmätään (parametriton täsmäys). Oikeanpuoleisessa kuvassa vertailulohkoa on kierretty ja pienennetty alkuperäiseen kuvaan verrattuna (parametrillinen täsmäys).

Optisen vuon tapauksessa muutos kuvataan vektorijoukolla (kuva 5). Vasemmanpuoleinen kuva on alkuperäinen, keskellä optinen vuo kuvattu vektoreilla (alkuperäinen neliö kuvattu katkoviivoilla) ja oikeanpuoleisessa kuvassa kuvasarjan seuraava kuva.

Kuvasarjassa liike voi muodostua seuraavilla tavoilla (Shapiro & Stockman 2001), algoritmisesti helpoimmasta vaikeimpaan:



Kuva 4: Alueen täsmäys lohkoittain.



Kuva 5: Alueen täsmäys optisen vuon kautta.

- Liikkumaton kamera, yksi liikkuva kohde vakiotaustalla.
- Liikkumaton kamera, useita liikkuvia kohteita vakiotaustalla.
- Liikkuva kamera lähes vakiotaustalla (yksi liikkuva kohde).
- Liikkuva kamera, useita liikkuvia kohteita.

Liikkumatonta kameraa voidaan käyttää alueen vartiointiin tai valvontaan. Kyseessä voi olla esimerkiksi rakennuksen sisäinen kulunvalvonta tai tieliikenteen seuranta. Kameran liikkuessa voidaan alueen kappaleita tunnistaa muotojen perusteella ja hahmottaa niiden etäisyyksiä.

Kun kamera on liikkumaton, liikkeen löytämiseksi riittää etsiä kuvien väliset erot (Sifakis & Tziritas 2001), jotka voidaan havaita vertaamalla eroja kynnysarvoon samassa pisteessä tai sen ympäristössä. Mikäli kamera liikkuu, voidaan sen liike ottaa huomioon. Kun kameran liike on poistettu tai muuten huomioitu, voidaan täsmäys suorittaa edellä mainituilla tavoilla.

2.2 Seuranta

Seuranta tuottaa kohteen sijaintia ja liikerataa koskevat tiedot. Kohteen seuraavan aseman laskemiseksi tulee tietää nykyinen asema, nopeus ja suunta:

$$\vec{s}_n = \vec{s}_n + \Delta t \vec{v}_n, \tag{1}$$

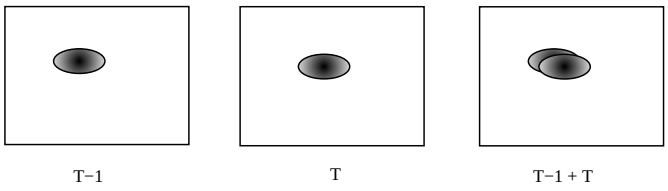
missä $\vec{s}_n$ on kappaleen nykyinen sijainti, t aika ja $\vec{v}_n$ nopeus. Moniulokkeisen kohteen (esim. nisäkäs, jonka raajat liikkuvat kehon suhteen) seuraamisessa on ongelmana ulokkeiden sisällyttäminen liikkeeseen. Tämä voidaan rinnastaa kohteen muodonmuutoksiin. Seurataan esimerkiksi hämähäkin liikettä, kun se tekee verkkoa: hämähäkin levätessä tai vaaran uhatessa se voi käpertyä pallon muotoon.

2.2.1 Tekniikat

Kohteen seurantaan on useita tekniikoita. Suurin osa pyrkii jollakin tavalla ennustamaan löytämänsä kohteen liikettä ja rajaamaan etsintäaluetta sen mukaisesti. Kohteen seurantaa voidaan helpottaa kuvasarjassa ottamalla huomioon edellinen (tai edellisiä) kuvakenttiä. Grinias & Tziritas (2001) esittivät menetelmän, jossa kuvan segmentoinnin jälkeen kukin kohde jaetaan eri tasoille. Menetelmässä käyttäjän tulee määritellä kohteiden suhteelliset syvyyssuuntaiset etäisyydet toisistaan.

Gambotto (1992) esittää menetelmän, jossa yhdistetään alueenlaajennus ja -seuranta. Ensimmäinen kuvakenttä segmentoidaan ja sitä seuraavissa kuvakentissä alueet täsmätään ajallisen vastaavuuden perusteella ja osittaisella alueenlaajennuksella. Alueiden ajallisella vastaavuudella tarkoitetaan alueiden osittaista päällekkäisyyttä peräkkäisissä kuvakentissä.

Kuvassa 6 on kuvattu oliota peräkkäisinä kuvakenttinä (ajat T-1 ja T) ja viimeisessä kaksi edellistä kuvakenttää on yhdistetty, jolloin päällekkäisyys näkyy.



Kuva 6: Alueet osittain päällekkäin peräkkäisissä kuvakentissä.

Gambotton algoritmi koostuu neljästä osasta:

- aluemallin päivitys,
- liikkeen estimointi ja tulkinta,
- lähdealueiden määritys,
- alueenlaajennus ja optimaalisten rajojen etsiminen.

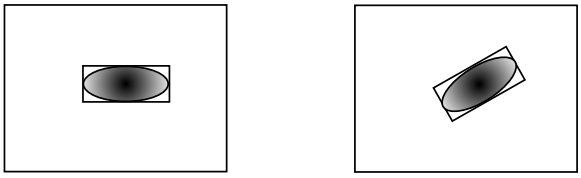
Gambotto määrittelee aluemallin, jossa on kolme komponenttia: *radiometrinen* (radiometric), *geometrinen* (geometric) ja *dynaaminen* (dynamic). Radiometrinen komponentti sisältää tiedon alueen harmaasävyarvon keskiarvosta ja hajonnasta. Geometrinen komponentti sisältää tiedon alueen pinta-alasta, alueen *ikkunasta* (määrittää suorakaiteen, jonka sisällä alue sijaitsee), reunaviivasta ja päällekkäisyydestä. Dynaamisessa komponentissa on tieto massakeskipisteestä, kuvasojen välisestä siirtymästä ja koon muutosvektorista.

Liikkeen estimointi ja tulkinta ennustaa geometriset muutokset ja siirtymät kuvatasossa. Rotaation oletetaan olevan pientä, jolloin sitä ei oteta huomioon. Apuna käytetään binäärisiä aluemaskeja, jotka kuvaavat alueen peittoa kuvatasossa eri ajankohtina, esim. kuvassa 6 maskin voisi kuvitella olevan samainen alue yksivärisenä. Kahden edellisen aluemaskin perusteella estimoidaan alueen koko ja paikka nykyisessä kuvatasossa. Liikkeen laskemiseen on kolme eri tekniikkaa: maskien massakeskipisteiden laskenta, maskien reunaviivojen täsmäys ja binääristen aluemaskien vastaavuus. Gambotton mukaan ensimmäinen tapa tuottaa hyviä tuloksia, kun päällekkäisyyksiä tai suuria segmentointivirheitä ei ole. Toinen tapa toimii hyvin, mikäli alueiden reunat muodostuvat suorista linjoista, yleisessä tapauksessa menetelmä on kuitenkin hankala käyttää. Kolmas tapa ottaa huomioon myös alueiden epäsäännölliset muodot. Tällöin liikevektori löydetään maksimoimalla peräkkäisten maskien leikkauksen koko. Alueen koon muutos lasketaan kahden peräkkäisen kuvatason pienimpien, mahdollisesti tasossa kiertyneiden, suorakaiteen muodostamien alueiden perusteella (kuva 7).

Koon muutos lasketaan suorakaiteelle pituuden ja leveyden osamäärinä peräkkäisissä kuvissa:

$$\begin{aligned} \Delta x &= \frac{x_i}{x_{i-1}} \\ \Delta y &= \frac{y_i}{y_{i-1}} \end{aligned} \tag{2}$$

missä x on suorakaiteen pituus ja y on suorakaiteen leveys. Alaindeksillä kuvataan kuvasarjan indeksiä (aikaa).



Kuva 7: Kahden peräkkäisen kuvakentän alueen oliot suorakaiteen muotoisen alueen sisällä.

Gambotton mukaan menetelmän toimi hyvin vaikka siirtymät olivatkin suuria. Nopeat muutokset, joissa kappale pyörii liian nopeasti, jäävät kuitenkin huomaamatta.

2.2.2 Kohteen löytäminen

Osa menetelmistä käyttää kuvan segmentointia erottelemaan mahdolliset kuvassa esiintyvät liikkuvat kohteet (Wang *et al.* 2000). On myös menetelmiä, jotka etsivät kohteet kuvasarjasta peräkkäisten kuvien erojen perusteella (Sifakis & Tziritas 2001).

Kuvan segmentointi voidaan jakaa neljään eri menetelmään (Fan *et al.* 2001):

- Kynnysarvo
- Reunaviiva
- Alue
- Yhdistelmä

Kynnysarvosegmentoinnissa oletetaan, että vierekkäiset ja samankaltaiset pikselit kuuluvat samaan alueeseen. Samankaltaisuus voi tarkoittaa harmaasävytasoa, väriarvoa tai *pinnoitetta*. Pinnoite on (Mirmehdi & Petrou 2000) samankaltaisena toistuva eri väreistä koostuva pinta. Esimerkiksi tiilipintaa ja ruohoa voidaan pitää pinnoitteena. Tällä menetelmällä saadaan hyviä tuloksia, kun kuva koostuu vain kahdesta voimakkaasti erilaisesta komponentista.

Reunaviivasegmentoinnissa oletetaan, että pikselin arvo muuttuu voimakkaasti kahden alueen välillä. Reunaviivan tulisi olla suljettu ja mahdollisimman ohut. Aina näin ei ole, jolloin on tarvetta käyttää paljon aikaa kuluttavia jälkikäsittelytoimenpiteitä. Näillä toimilla reunaviiva pyritään sulkemaan tai ohentamaan.

Aluesegmentoinnissa oletetaan, että pikseli näyttää samankaltaiselta kyseisessä alueessa. Arvottaminen voi perustua harmaasävy- tai väriarvoon tai pinnoitteeseen.

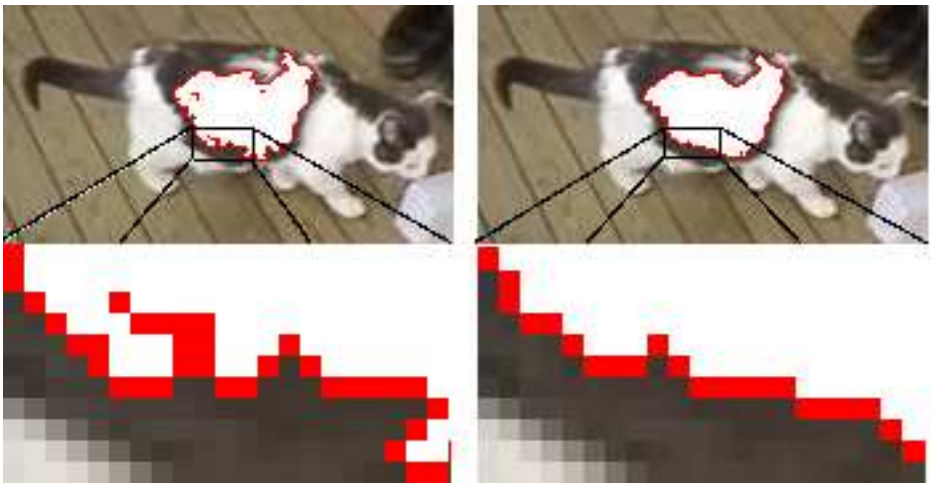
Yhdistelmäsegmentoinnissa reunaviiva- ja alueenlaajennustekniikat yhdistetään, jolloin segmentointitulosten oletetaan olevan tarkempia. Kuvan segmentointi tuottaa mahdolliset seurattavat kohteet saatujen alueiden muodossa.

2.2.3 Ongelmat

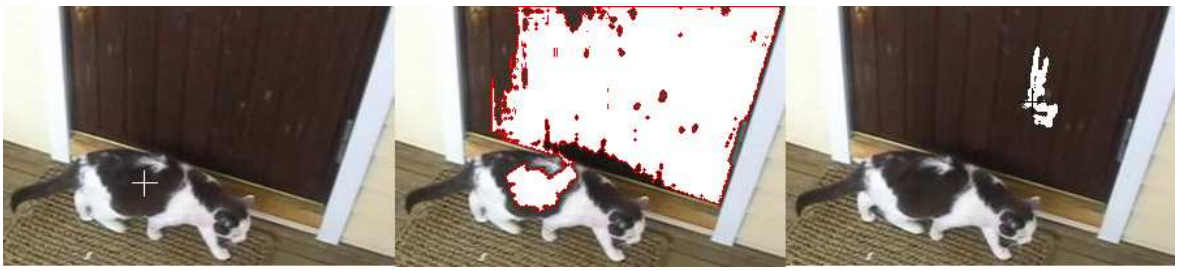
Ongelmia seurannassa (Shapiro & Stockman 2001) aiheuttavat kuvakohina, valoisuuden vaihtelu, kohteen liike (*translaatio* ja *rotaatio*) ja muodonmuutokset (*mutaatio*) sekä kameran liike (translaatio, rotaatio ja skaalaus). Translaatiossa kohde liikkuu kuvakentän yli mielivaltaiseen suuntaan, rotaatiossa kohde pyörii, mutaatiossa sen muoto muuttuu ja skaalauksessa sen koko. Kuvasarjassa etenevät kohteet voivat myös peittää toisensa. Tällöin on käytetty (Grinias & Tziritas 2001) tasoja kohteiden erottelemiseen niiden hallinnan helpottamiseksi.

Kohina vaikeuttaa alueen (kohteen) tarkkojen reunojen löytämistä ja sekoittaa kuvan pieniä yksityiskohtia. Kohinan vähentämiseksi kuva voidaan suodattaa, jolloin kuvaa muokataan suorittamalla kuvapisteille esimerkiksi *mediaanisuodatus*, jossa kuvan jokainen piste saa sitä ympäröivän alueen (esim. 3x3) pisteiden järjestyksessä keskimmäisen arvon (esim. kuvapisteiden arvot: 4,4,3,4,7,7,5,9 ja 5, joista keskimmäinen eli mediaani on 5). Suodatuksen tuloksena kuvainformaatiota lähes aina myös hukataan. Kuvassa 8 kissan kylki aluelaajennettu vasemmalla ilman suodatusta ja oikealla keskiarvoistamalla. Oikeanpuoleisesta kuvasta nähdään, että reunat eivät ole niin rosoiset ja alueen sisälle ei jäänyt yksittäisiä alueeseen kuulumattomia pikseleitä.

Valoisuuden vaihtelu aiheuttaa sävyerojen vaihtelua, jolloin segmentoinnin tuloksena voi kahden peräkkäisen kuvan alueet yhtyä tai yksittäinen alue hajota useaan osaan. Kuvassa 9 ristillä merkitystä aloituspisteestä väri on vuotanut ovelle, jolloin seurannan toteuttava ohjelma on tehnyt ovesta alueenlaajennoksen jälkeen seurattavan kohteen, vaikka tarkoituksena oli segmentoida kissan kylki. Lisäksi pistemäisen valolähteen (tai useiden) valaistessa kuva-aluetta, varjojen määrä voi vaihdella. Valoisuuden vaihtelun korjaamiseksi voisi yrittää muuttaa kuvan *kontrastia*, jolloin värien väliset erot muutuvat. Kontrastilla tarkoitetaan kahden vierekkäisen värisävyn eroa. Esimerkiksi mustan ja valkean välinen kontrasti on suuri ja harmaan ja valkoisen (tai mustan) on pieni.



Kuva 8: Alueenlaajennuksen tulos suodattamattomalle kuvalle (vasen) ja suodatetulle (oikea), alla suurennetut osat kuvista.



Kuva 9: Värin vuotaminen “väärälle” alueelle. Vasen kuva tilanne alussa, keskimmäinen alueelaajennuksen lopussa ja oikeanpuoleinen seuraava kuvasarjan kuva, jossa seuranta siirtynyt kissan kyljestä oveen.

Kohteen muodonmuutokset ovat hankalia seurannan kannalta. Mikäli muutos on mutaation lisäksi *kameleonttinen* (värien vaihtelu), seuranta-algoritmin olisi mukauduttava myös sen mukaan. Alueenlaajennustekniikka voisi tässä tapauksessa toimia (koska muodolla ei ole väliä) mikäli se osaa ottaa huomioon myös värin muuttumisen ajan suhteen.

Kameran liikkeen arvioimiseksi on olemassa erilaisia malleja (Wang *et al* 2000), joilla voi yrittää poistaa kameran liikkeen vaikutuksen. Vaikeutena on erottaa kameran liike kohteiden liikkeestä. Resoluution pienentäminen voi nopeuttaa kameran liikkeen löytämistä.

2.3 Sovelluksia

Sovellukset vaihtelevat valvonta- ja seurantatehtävistä videon koodaukseen. Suurin osa lähteistä painottuu videon koodaukseen, joten MPEG-4 esitellään hieman tarkemmin.

2.3.1 Valvonta

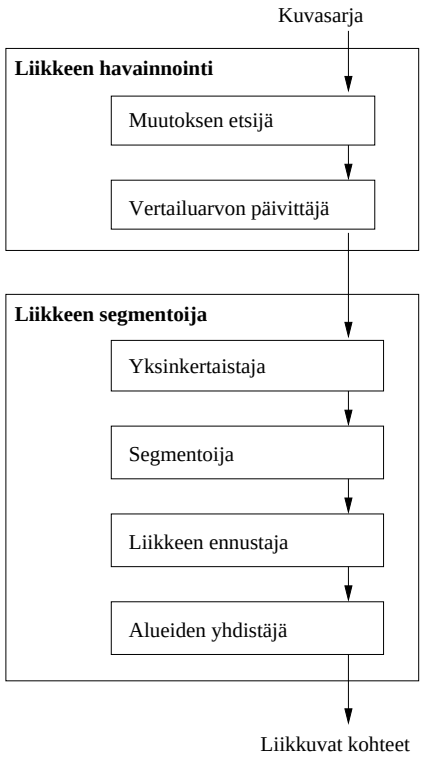
Videovalvontajärjestelmiä on käytössä mm. liikenteen seurannassa, murtosuojauksessa ja lentokenttien valvonnassa (Kim & Kim 2002). Automaattisen valvontajärjestelmän pitää kyetä tulkitsemaan kuvasarja tosiaikaisesti.

Kim & Kim:n esittämä järjestelmä on liikenteen seurantasysteemi, joka koostuu liikkeen havainnointi- ja segmentointikomponentista (kuva 10). Havainnointikomponentti etsii kuvasta liikkuvat kohteet ja segmentointikomponentti yhdistää samankaltaiset pisteet, joilla on samansuuruinen intensiteetti ja liikevektori.

Havainnointikomponentti etsii eroja kahden peräkkäisen kuvan sekä taustakuvan ja tutkittavan kuvan välillä. Komponentti koostuu muutoksen etsijästä (*variation detection*) ja vertailuarvon päivittäjästä (*adaptive thresholding*). Muutoksen suuruutta verrataan vaihtuvaan vertailuarvoon, jota päivitetään kuvan vaihtuessa. Muutoksen etsijä saa syötteenä kaksi peräkkäistä harmaasävykuvaa ja taustakuvan, jossa ei saa olla liikkuvia kohteita. Etsijän tuloksena saadaan voimakkaasti muuttuneet alueet.

Segmentointikomponentti koostuu neljästä alikomponentista:

- Yksinkentaistaja



Kuva 10: Kim & Kim:n seurantajärjestelmän komponentien suhteet.

- Segmentoija
- Liikkeen ennustaja
- Alueiden yhdistäjä

Yksinkertaistaja poistaa epäoleellista tietoa muuttuneista alueista. Se voi esimerkiksi vähentää pinnoitettujen alueiden sisäisiä eroavaisuuksia tai poistaa liian pieniä alueita.

Segmentoija segmentoi kuvan käyttämällä *k-means* algoritmia (Shapiro & Stockman 2001 pp. 282), joka järjestää datapistejoukon siten, että ryhmien sisällä erot ovat pienempia kuin ryhmien välillä. Segmentoija käyttää ominaisuusvektoreita (*feature*), jotka sisältävät tiedon pisteen koordinaateista ja intensiteetistä muodostaessaan arvojoukon.

Liikkeen ennustaja etsii segmentin liikevektorin, joka minimoi liikkuneen segmentin muutokset. Minimointi koskee joukon intensiteettieroja, kun alialuetta verrataan pikselikohtaisesti segmentoijan rajoittamalla alueella eri kohdissa kaavan 3 mukaisesti:

$$f = \frac{1}{N_j} \sum_{(x,y) \in c^j} (I_k(x,y) - I_{k+1}(x + t_x, y + t_y)) \tag{3}$$

missä f on virhefunktio, N_j arvojoukon j pisteiden lukumäärä, c^j arvojoukko j, t_x ja t_y liikevektoreita ja I_k ja I_{k+1} ovat intensiteetit kahdessa peräkkäisessä kuvassa.

Alueiden yhdistäjä yhdistää ylisegmentoinnin mahdollisesti tuottamat liian pienet alueet. Yhdistäminen tapahtuu liikevektoreiden samankaltaisuuden perusteella perustuen kynnysarvoon.

Esitelty menetelmä kykenee kirjoittajien mukaan hyvin löytämään liikkuvat kohteet, vaikka valoisuus muuttuu, ympäristö muuttuu ja kohteet ovat päällekkäin.

2.3.2 MPEG

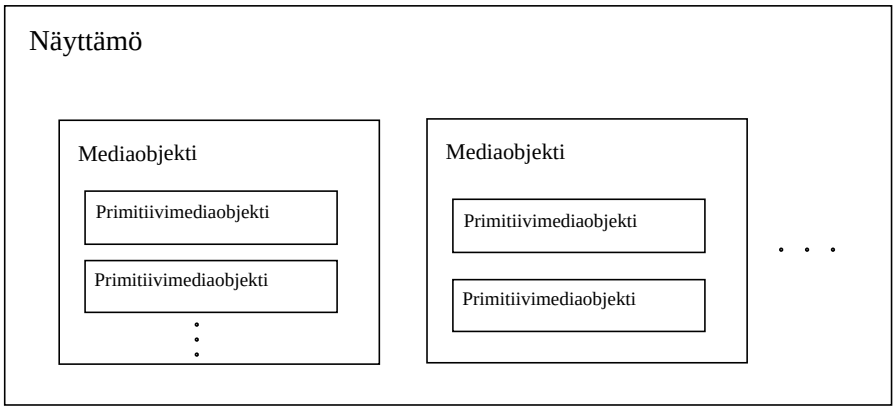
MPEG on kokoelma standardeja audiovisuaalisen informaation esittämiseen (Martínez, 2002). Video, CD ja MP3-musiikin tallennusformaatti perustuvat MPEG-1 standardiin, kun taas digitelevisio ja DVD käyttävät MPEG-2:a. MPEG-4:ää voi käyttää internetissä ja mobiiliverkoissa sekä vastaavissa (PDA laitteissa). MPEG-7 sisältää kuvauksen multimediatiedon sisällöstä. Viimeisin (2003) kehitteillä oleva standardi on MPEG-21, joka laajentaa MPEG-7 sisältämään kuvauksen käyttäjistä, joilla on tiettyjä oikeuksia ja mahdollisuuksia käsitellä multimediatietoa.

Seuraavassa tarkastellaan lähemmin MPEG-4:ää, joka sisältää mm. standardit seuraaville rakenteille ja toiminnoille:

- Ääni-, kuva- ja multimediarakenteet eli ns. mediaobjektit (*media object*).
- Yhdistelmäoliot.
- Siirrettävän tiedon multipleksointi ja synkronointi.
- Käyttäjän mahdollisuus vaikuttaa *näyttämöllä* objektien sijaintiin.

Mediaobjektit voivat olla ääntä, kuvaa, tai niiden yhdistelmiä, jotka voivat olla luonteeltaan luonnollisia (*natural*) tai synteettisiä (*synthetic*). Yhdistelmäoliot voivat sisältää mediaobjekteja, jotka muodostavat ns. audiovisuaalisen näyttämön (*audiovisual scene*). Siirrettävän tiedon multipleksoinnilla ja synkronoinnilla varmistetaan, että mediaobjektille määritelty vähimmäislaatutaso siirron aikana pystytään saavuttamaan.

Näyttämö (kuva 11) koostuu hierarkisesti järjestetyistä mediaobjekteista (MO), jotka voivat koostua useista primitiivimediaobjekteista (PMO, *primitive media object*), joita voivat olla liikkumaton kuva, ääni- tai videojakso. Käyttäjä voi myös muuttaa omaa sijaintiaan näyttämöllä muuttaen siten näkymää.



Kuva 11: MPEG-4 järjestelmän näyttämöhierarkia.

PMO:t sijaitsevat hierarkian lehtitasolla ja voivat rakenteeltaan olla kaksi- tai kolmiulotteisia. Edellisten lisäksi MPEG-4:ssä määritellään myös teksti- ja grafiikkaolio, synteettinen ääni sekä synteettinen puhuva pää, jonka kanssa assosioidun puhutun tekstin perusteella ääni tuotetaan kasvojen ilmeiden lisäksi.

MO sisältää itsessään kaiken tiedon, jolla päätelaite pystyy sen esittämään ympäristöstä tai taustasta riippumatta. Näyttämön hallitsemiseen MPEG-4:ssä on standardoitu edellisten lisäksi mm. seuraavat ominaisuudet:

- Mediaobjektin sijoittaminen minne tahansa näyttämölle.
- MO:n geometristen tai akustisten ominaisuuksien muuttaminen.
- PMO:iden ryhmittäminen yhdeksi yhdistelmämediaobjektiksi (*compound media object*).
- Tietovirran (*streamed data*) liittäminen MO:in.

Tietovirran liittäminen MO:iin liittyy läheisesti kohteenseurantaan. MPEG-4:ssä videoobjekti (VO) voi olla luonnollinen tai synteettinen. Esimerkiksi uutisten juontaja voi olla

video-objekti, joka on erotettu taustasta jollain tekniikalla. Tällöin tausta riittää lähettää vastaanottajalle kerran ja juontaja niin usein kuin asento/ilmeet muuttuvat. Juontajan “siirtämiseksi” riittää siirtää pelkästään tieto hänen peittämästään alueesta, jolloin siirrettävän tiedon ei tarvitse olla suorakaiteen muotoinen video-objekti.

MPEG-4:ssä on standardoidut työkalut ja algoritmit, jotka ratkaisevat seuraavat ongelmat:

- Kuvien ja videon tehokas pakkaus.
- Kaksi- ja kolmiulotteisten verkkojen (*mesh*) pinnoitteiden tehokas pakkaus.
- Kaksiulotteisten verkkojen pakkaus.
- Verkkoja animoivien geometristen tietovirtojen tehokas pakkaus.
- Tehokas suorasaanti (*random access*) kaiken tyyppisiin video-objekteihin.
- Laajennettu kuvien ja videokuvan muokkaaminen (*manipulation*).
- Sisältöperustainen (*content-based*) kuvan ja videon koodaus.
- Sisältöperustainen pinnoitteiden, kuvien ja videokuvan skaalaus.
- Avaruudellinen (*spatial*) ajallinen ja laadullinen skaalautuvuus.
- Virheiden sietokyky (*robustness*) ja joustavuus (*resilience*) virhealttiissa ympäristössä.

Sovelluksen suunnittelijan tehtäväksi jää juontaja-esimerkin tapauksessa juontajan erottaminen taustasta.

Koska MPEG-4 tarjoaa tuen synteettisen tiedon välittämiseen, kanavakohtainen animoitu logo voidaan siirtää päätelaitteelle siten, että sen geometrinen muoto siirretään kerran ja siitä edespäin riittää siirtää logon muutokseen tarvittava tieto. Esimerkiksi kuution päälle kirjailtu tunnus lähetetään siirtämällä kuution tiedot (kulmapisteiden koordinaatit, kuution sijainti näyttämöllä ja pinnoite). Tällöin päätelaite osaa piirtää logon sopivaan paikkaan ruudulla ilman, että kaikki tieto on jatkuvasti tiedonsiirrossa mukana.

Esimerkki kokonaisuudessaan voi sisältää studion taustakuvan, juontajan VO-tiedot, aseman logon ja mahdollisen muuttuvan tekstitiedon, joka näytetään esimerkiksi ruudun alareunassa. Tällöin siirretään ensiksi taustakuva, sitten logon, juontajan ja tekstin tiedot (kirjasin ja

väri). Päätelaite vastaanottaa näiden tietojen jälkeen enää logon ohjaustietoja, lisätekstit ja juontajan muuttuneet piirteet.

Tämän työn kannalta näistä mielenkiintoisin asia on liikkuvan kohteen erottamiseen taustasta, mikä juontaja-esimerkissä tarkoittaa juontajan erottamista taustakuvasta.

3 ALUEENLAAJENNUS

Alueenlaajennus on tekniikka, jossa kuva jaetaan alueisiin yhdistämällä kuvapisteitä yhtenäiseksi alueeksi niiden väriarvon perusteella (Adams & Bischof 1994). Alkupisteen valinta voi tapahtua käyttäjän toimesta tai jollain algoritmilla kuvan perusteella. Pisteen automaattinen valinta voi perustua esimerkiksi reunanhakualgoritmin löytämien reunojen välissä sijaitsevien pisteiden väriarvoille. Kohteen seurannassa alueet määritellään kohteiksi, joita seurataan. Grinias:n ja Tziritas:n (2001) esittämässä menetelmässä käyttäjä määrää aloitusalueen ja erottelee kohteet eri tasoille (*layer*), jolloin niiden käsittely ja seuranta on helpompaa. Tässä tapauksessa tasot määrittävät kohteiden syvyyssuuntaisen järjestyksen.

3.1 Algoritmi

Alueenlaajennusalgoritmi lähtee aloituspisteestä ja yhdistää naapuripisteitä samaan alueeseen mikäli pisteiden väriarvo on sopivissa rajoissa. Tätä jatketaan naapuripisteille, kunnes kaikki kuvan pisteet on käyty läpi. Toteutuksen eri variaatiot syntyvät sen mukaan valitaanko yksi piste tai monta pistettä, jotka voivat kuulua samaan tai eri alueeseen. Lisäksi eroja syntyy pisteen arvottamisessa. Toteutuksessa voi myös käyttää mahdollisuutta muodostaa uusia alueita, mikäli uusi piste ei sovellu lisättäväksi mihinkään olemassaolevaan alueeseen (Lin *et al* 2000). Alueenlaajennuksessa pienen lisäongelman tuottaa liitettävien pisteiden hakeutuminen kohti jo löydettyjä alueita. Hakeutumisella tarkoitetaan sitä, että piste ei välttämättä kuulu jo löydettyyn alueeseen, johon se arvotetaan, vaan sen kuuluisi sisältyä naapurialueeseen. Yksi ratkaisu on käydä alueet uudelleen läpi käyttäen hyväksi edellisen läpikäynnin tuloksia. Toinen tapa tähän on arvottaa uuden alueen naapureiden reunapisteet, jolloin ne voidaan tarvittaessa siirtää paremmin täsmävään alueeseen. Kolmas menetelmä on toisen kaltainen, mutta arvottaminen tapahtuu sen jälkeen, kun ensimmäinen segmentointikerta on valmis.

Karkeasti ottaen yleisin alueenlaajennusalgoritmi on seuraava:

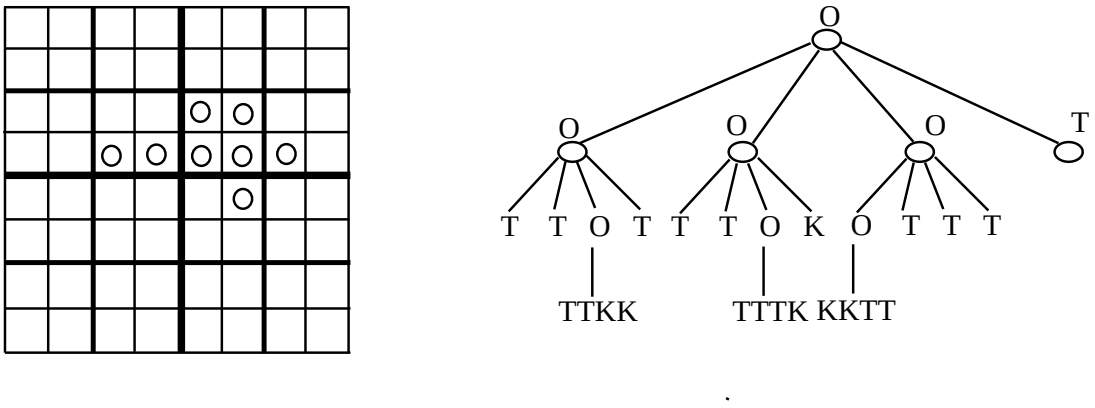
1. Valitse aloituspiste.
2. Vertaa naapuripisteitä aloituspisteeseen ja mikäli aloituspisteen ominaisuudet täsmäävät riittävällä tarkkuudella aloituspisteen ominaisuuksiin, lisää se kyseiseen alueeseen.

3. Laajenna aluetta, kunnes kaikki kuvan pisteet on käyty läpi.

Hajoita ja yhdistä-algoritmissa (*split-and-merge*) (Shapiro & Stockman 2001) kuva pilkotaan ensin *nelipuuhun* (kuva 12), jossa jokainen neliö vastaa yhtä aluetta, joka ei ole enää jaettavissa. Tämän jälkeen alueita yhdistetään, kunnes yhtenäisyyskriteeri ei enää päde. On huomattava, että suorakaiteen muotoisen alueen segmentointiin nelipuuta ei voi käyttää. Algoritmin muodossa tämä tapahtuu seuraavasti:

1. Jaa alue A neljään neliönmuotoiseen alueeseen, mikäli yhtenäisyyskriteeri sen sisältämien pisteiden välillä ei päde.
2. Yhdistä kaksi vierekkäistä aluetta, mikäli yhteinäisyyskriteeri niiden välillä on voimassa.
3. Toista kohdat 1 ja 2, mikäli alueita voidaan vielä pilkkoa tai yhdistää.

Montoya *et al*:n esittämä algoritmi redusoituu yleiseen alueenlaajennusalgoritmiin, jos kuva jaetaan alussa pikselin kokoisiin alueisiin.



Kuva 12: Nelipuussa solmu voi olla tyhjä (T), osittäin täysi (O) tai kokonaan täysi (K). Puun juurisolmu on koko kuva ja solmun lapset etenevät myötäpäivään vasemmasta yläkulmasta lähtien.

Hieman tarkempi versio yleisestä algoritmista (Adams & Bischof 1994) käyttää järjestettyä listaa (SSL, *sequentically sorted list*). Kyseinen algoritmi on käytössä useissa lähteissä pienin muutoksin. Algoritmissa leimaaminen tarkoittaa pisteelle annettua aluetunnusta, joka

on kullekin alueelle ominainen. Reunaleimalla tarkoitetaan aluetunnusta rajapisteelle, joka ei kuulu mihinkään alueeseen. Tällöin algoritmi on seuraava:

1. Leimaa aloituspiste(et) eli anna niille yksilöivä aluetunniste.
2. Laita aloituspisteiden naapurit arvonsa mukaisesti listaan.
3. Toista, kunnes lista on tyhjä:
   (a) Poista ensimmäinen piste x listasta.
   (b) Testaa x naapurit:
      i. Jos kaikilla x:n leimatuilla naapureilla on sama leima.
         A. Leimaa x kyseisellä leimalla.
         B. Päivitä alueen värikeskiarvo (harmaasävy).
         C. Lisää x:n naapurit listaan arvonsa mukaisesti, mikäli eivät ole jossain alueessa tai listassa.
      ii. Muutoin
         A. Leimaa x reunaleimalla (eli piste ei kuulu mihinkään alueeseen).

Kuvassa 13 on piirretty pienen alueen laajentuminen listaa käyttävän algoritmin mukaisesti ja kuvassa 14 on laajentuminen todellisessa kuvassa. Todellisesta esimerkkikuvasta nähdään, että vaikka ihmissilmälle alue näyttää olevan tasavärinen, todellisuudessa se ei sitä ole. Kuvassa laajennus nähdään 200 pisteen hyppäyksin.

3.2 Aloituspisteen valinta

Alueenlaajennusalgoritmit vaativat yleensä käyttäjää asettamaan aloituspisteen (Adams & Bischof 1994), toiset käyttävät kaikkia kuvan pisteitä aloituspisteinä ja yhdistävät samankaltaisia pikseleitä yhtenäisiksi alueiksi (Montoya *et al.* 1999). Yleisesti ottaen aloituspisteen automaattiselle valinnalle ei ole oikeaa tapaa, vaan se riippuu pitkälti sovelluksesta (Stewart *et al.* 2002). Lisäksi joissain tapauksissa aloituspistettä ei valita, vaan algoritmi käy läpi kuvan pisteitä ja muodostaa tarvittaessa uusia alueita (Lin *et al.* 2000). Edelleen käyttäjä voi määrittää alkualueen piirtämällä alueen ääriviivan, jota algoritmi tarvittaessa laajentaa (Grinias & Tziritas 2001 ja Zhong *et al.* 2000). Eräässä menetelmässä (Fan *et al.* 2001)

1	2	2	3	1	2
2	2	2	3	2	2
3	3	6	6	4	2
2	2	2	6	6	2
3	2	6	3	3	6
3	2	3	4	2	4

=>

1	2	2	3	1	2
2	2	2	3	2	2
3	3	6	6	4	2
2	2	2	**6**	6	2
3	2	6	3	3	6
3	2	3	4	2	4

=>

1	2	2	3	1	2
2	2	2	3	2	2
3	3	6	**6**	4	2
2	2	2	**6**	6	2
3	2	6	3	3	6
3	2	3	4	2	4

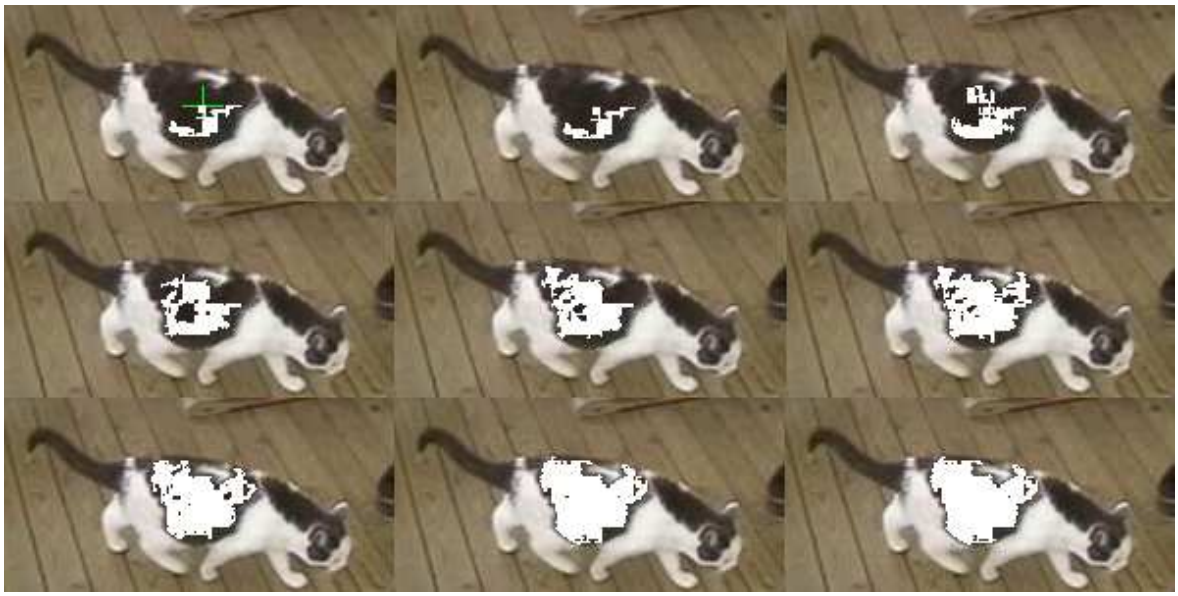
=>

1	2	2	3	1	2
2	2	2	3	2	2
3	3	**6**	**6**	4	2
2	2	2	**6**	6	2
3	2	6	3	3	6
3	2	3	4	2	4

=>

1	2	2	3	1	2
2	2	2	3	2	2
3	3	**6**	**6**	4	2
2	2	2	**6**	**6**	2
3	2	6	3	3	6
3	2	3	4	2	4

Kuva 13: Alueenlaajennusalgoritmin eteneminen järjestettyä listaa käyttäen. Ensimmäinen kuva on alkuperäinen. Toisessa kuvassa näkyy korostettuna lähtöpiste. Viimeisessä kuvassa on segmentoinnin tulos.



Kuva 14: Alueen laajentuminen alueenlaajennusalgoritmin mukaisesti.

etsitään reunaviivoja, joiden perusteella lasketaan aloituspiste. Aloituspisteeksi valitaan reunojen keskeltä löytyvä piste. Adams & Bischof:n menetelmässä (Adams & Bischof 1994) aloituspisteet osoittaa järjestelmän käyttäjä. Valvonta- tai vartiointitehtävissä järjestelmän on kuitenkin itsenäisesti löydettävä liikkuva kohde. Aloituspisteen väriarvon voidaan olettaa kuvaavan yleensä alueen väriarvon keskiarvoa. Valitsemalla useita pisteitä samalta alueelta voidaan myös alueen väriarvojen hajonnan suuruutta arvioida.

Deng ja Manjunath (2001) kvantisoivat kuvan väriarvojakauman eri luokkiin, joita käytetään apuna alkupisteitä muodostettaessa. Luokille lasketaan lukuarvo, joka kuvaa alueen homogeenisuutta. Mikäli kyseinen lukuarvo on tarpeeksi pieni, alueen voidaan katsotaan olevan yhtenäinen, kun taas suuri arvo tarkoittaa suurta hajanaisuutta. Yhtenäinen alue voi olla myös teksturoitu.

Porikli (2002) ja Cramariuc *et al* (1997) käyttävät *gradienttia* (väriarvon muutoksen nopeutta) aloituspisteiden etsinnässä. Pienimmän gradientin omaavat pisteet muodostavat aloituspistejoukon.

Tässä työssä käytetään useita aloituspisteitä myös siten, että niiden keskiarvoa ei lasketa, vaan niitä verrataan ehdokaspikseleiden väriarvoon. Alueeseen katsotaan kuuluvaksi pikselit, joiden ero on sallituissa rajoissa mihin vertailupikseliin tahansa verrattuna. Tällöin täytyy muistaa, että algoritmin käyttämää lajiteltua listaa ei tule käyttää. Tämä johtuu siitä, että useamman pikselin väriarvot voivat olla täysin eri sävyä, jolloin lajittelu toimii väärin. Menetelmällä voi eri värisävyn alueita “yhdistää” samaan alueeseen, mistä saattaa olla hyötyä kohteenseurannassa.

3.3 Laajentumiskriteerit

Laajentumiskriteereitä, joissa suoritetaan pikselikohtaista vertailua, nimitetään lokaaleiksi. Staattiset kynnysarvot ovat lokaaleja, algoritmin edetessä muuttumattomia arvoja. Dynaamiset kynnysarvot voivat muuttua algoritmin eri vaiheissa. Yleisesti kynnysarvona käytetään jotain etukäteen määrättyä tai kuvasta jollain menetelmällä laskettua arvoa. Kyseinen erotustekijä ei kuitenkaan yksin aina riitä, koska kuvan eri alueissa sopiva kynnysarvo vaihtelee (Chang & Li 1994). Tällöin käytetään joko *paikkasidonnaisia* (position-varied) tai *aikasidonnaisia* (time-varied) kynnysarvoja. Paikkasidonnaiset tekijät vaihtelevat kuva-alueittain, jolloin käytetään hyväksi etukäteistietoa kuvasta tai vaihtuvaa kynnysarvoa,

joka on laskettu valmiiksi kuvan eri alueille. Aikasidonnaiset tekijät riippuvat alueenlaajennusprosessin etenemisvaiheesta. Myöhemmin kappaleessa 3.3.2 kerrotaan menetelmästä tarkemmin. Globaalit laajentumiskriteerit tarkoittavat tämän työn puitteissa alueiden yhdistämisen kriteereitä. Kriteerien määrittely vaihtelee, samoin kuin pisteen arvottamisessa, yksinkertaisista dynaamisiin kynnysarvoihin. Vastaavuudelle on käytetty mm. alueiden reunojen ohuutta (Fan *et al.* 2001), värin samankaltaisuutta (mm. Deng & Manjunath 2001) yksinkertaisissa tapauksissa. Dynaamisessa (*adaptive*) yhdistämisessä kynnysarvon arvo vaihtelee alueittain (Chang & Li 1994).

3.3.1 Staattiset kynnysarvot

Staattisilla kynnysarvoilla tarkoitetaan etukäteen määrättyjä vaihteluvälejä, joille pisteen arvottavan funktion tuloksen on osuttava, jotta piste voidaan hyväksyä kuuluvaksi vertailualueeseen. Harmaasävykuvissa voidaan käyttää esimerkiksi sävyarvoa, jolle sallitaan pieni poikkeama. Vastaavasti värikuvissa esimerkiksi värisävyä (hue), värikylläisyyttä (saturation) ja voimakkuutta (intensity). Yksi mahdollisuus on käyttää *gradienttia*, joka tarkoittaa kuvan väriarvossa tapahtuvaa muutoksen suuruutta (Kumar *et al* 1999). Jos merkitään erotusfunktiota δ:lla, yksinkertaisin vaihtoehto (Adams & Bischof 1994) voisi olla kaavan 4 mukaan laskea pisteen väriarvon poikkeama alueen sen hetkisestä keskiarvosta:

$$\delta(x) = |g(x) - g(A_i)| \tag{4}$$

Kaavassa g tarkoittaa harmaasävyarvoa, $g(A_i)$ alueen i harmaasävyjen keskiarvoa, ja x kuvapistettä.

Keskiarvoon vertailussa pitää aloituspisteen olla huolella valittu, ettei mahdollinen kuvakohina sotke tuloksia (Adams & Bischof 1994), koska kohinainen aloituspiste voi estää alueen laajenemisen. Tämän vuoksi suositellaan, että yhden aloituspisteen sijasta valitaan useita pisteitä, jotka kuvaavat kyseistä aluetta mahdollisimman hyvin. Tällöin sävyarvojen hajonta ei suurene liiaksi.

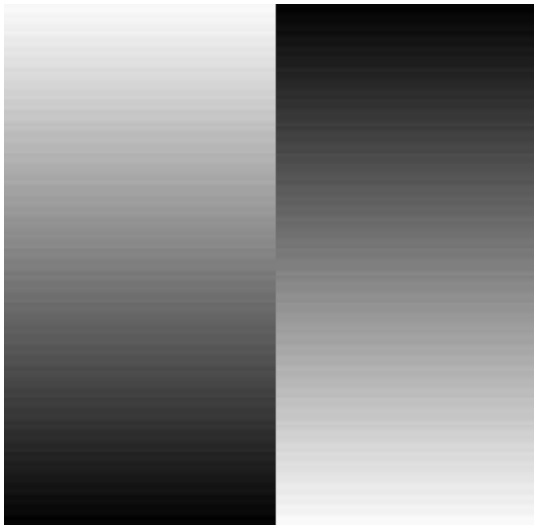
Toinen mahdollisuus on käyttää useaa pistettä ehdokaspistettä arvottamisessa. Tällöin voidaan erotusfunktioksi muotoilla:

$$\delta(x) = min|g(x) - g(x_i)| \tag{5}$$

missä x_i tarkoittaa aloituspistettä i ja muut vastaavasti kuin edellisessä kaavassa.

Kumar *et al* (1999) käyttävät gradienttia pisteen arvottamisessa siten, että kynnysarvona käytetään paikallista eroavaisuutta ϵ ja kuvaan sidottua eroa λ. Molempien kynnysarvojen tulee olla alle annetun rajojen, jotta piste otettaisiin mukaan alueeseen. Hankaluutena on kynnysarvojen asettaminen, joista paikallisen (ϵ) arvioimiseen Kumar *et al* käyttävät useita kuvia ja asettivat sen siten käsin. Kuvaan sidottu kynnysarvo (λ) lasketaan kuitenkin automaattisesti histogrammin minimeistä.

Mikäli alueen väriarvo muuttuu tasaisesti ja käytetään gradienttia laajenemiskriteerinä, voi käydä niin, että segmentoinnin tulos ei vastaa todellista kuva-aluetta. Alue voi vuotaa, mikäli vieressä olevan alueen väri muuttuu sopivassa kohdassa likimain samaksi kuin käsiteltävässä. Kuvassa 15 suorakaiteelta näyttävien tasojen keskellä on lähes samanväriset kaistaleet, jolloin on mahdollista, että alueet yhdistyvät.



Kuva 15: Värin mahdollinen vuotaminen alueiden välillä.

Reynolds & Rolnick (1995) käyttävät myös gradienttia kuvapisteen arvottamiseen. Tällöin käytössä on Sobelin operaattori (matriisi), joka x-suunnassa on annettu (Reynolds & Rolnick 1995) muodossa:

$$M_x = \begin{array}{|c|c|c|}\hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \quad (6)$$

ja y-suunnassa:

$$M_y = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \quad (7)$$

joiden avulla gradientti lasketaan kullekin kuvan pisteelle. gradientin suuruus yhdelle pisteelle lasketaan x- ja y- akseleiden suunnissa ja saa arvon:

$$\delta(x) = \sqrt{{g_x}^2 + {g_y}^2} \tag{8}$$

missä g_x on gradientti x-suunnassa ja g_y vastaavasti y-suunnassa. kaavan 8 lisäksi rajatapauksessa, kun ei tiedetä kuuluuko piste alueeseen vai ei, lasketaan laplacian operaattorin:

$$L = \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \tag{9}$$

avulla tarkempi arvo pisteelle. arvo saadaan laskemalla pisteen ympäristön väriarvot yhteen matriisissa määritellyillä kertoimilla siten, että keskimmäinen matriisin alkio vastaa laskettavaa pikseliä ja ympärillä olevat alkiot pikselin vastaavasti ympärillä olevia pikseleitä. Mikäli operaattorin tuloksena saadaan nolla tai alle, luetaan piste kuuluvaksi taustaan.

Kun käsitellään värillistä kuvaa, yhtenäisyyskriteeri hieman monimutkaistuu, koska pitää ottaa huomioon eri värikomponentit. Fan *et al* (2001) käyttävät YUV väriavaruutta väriarvojen erottelemiseen. Muunnos RGB väriavaruudesta YUV:hen käy seuraavasti:

$$\begin{aligned} Y &= 0.30R + 0.59G + 0.11B \\ U &= 0.493 * (B - Y) \\ V &= 0.877 * (R - Y) \end{aligned} \tag{10}$$

missä Y kuvaa pisteen intensiteettiä, U ja V kuvaavat värikomponentteja. Erotusfunktioksi Fan *et al* (2001) käytettävät seuraavan kaavan mukaista esitystä:

$$\begin{aligned} \delta(x) = {} & |Y(x) - Y(A_i)| + \\ & |U(x) - U(A_i)| + \\ & |V(x) - V(A_i)| \end{aligned} \tag{11}$$

missä Y, U ja V ovat YUV järjestelmän värikomponentteja ja A_i eri värikomponenttien keskiarvot alueessa i. Perusteena YUV:n käytölle on mainittu sen yleinen käyttö videon koodauksessa, jolloin muutosta väriavaruudesta toiseen ei tarvita. Lisäksi kyseinen koodaus erottelee intensiteetin ja värit toisistaan, jolloin rajat saattavat erottua paremmin kuin RGB-koodauksella.

Samansuuntainen idea YUV:n kanssa on HSI-koodaus, joka saadaan RGB:stä seuraavasti (Fränti 1998):

$$\begin{aligned} H &= \frac{1}{360^o}\cos^- 1\left[\frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right], josB > G \\ &\quad 1-\frac{1}{360^o}\cos^- 1\left[\frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right], muutoin \\ S &= 1-\frac{3}{R+G+B}\min\{R,G,B\} \\ I &= \frac{1}{3}(R+G+B) \end{aligned} \tag{12}$$

missä H on pikselin *värisävy* (hue), S on *värin puhtaus* (saturation) ja I on *intensiteeti* (intensity).

Värieron laskemiseen kahden pikselin välillä voidaan käyttää vektorietäisyyttä:

$$\delta(x,y) = \sqrt{\sum_i |x_i - y_i|^2} \tag{13}$$

missä x ja y ilmaisevat pikselit ja alaindeksillä merkitään värikomponenttia.

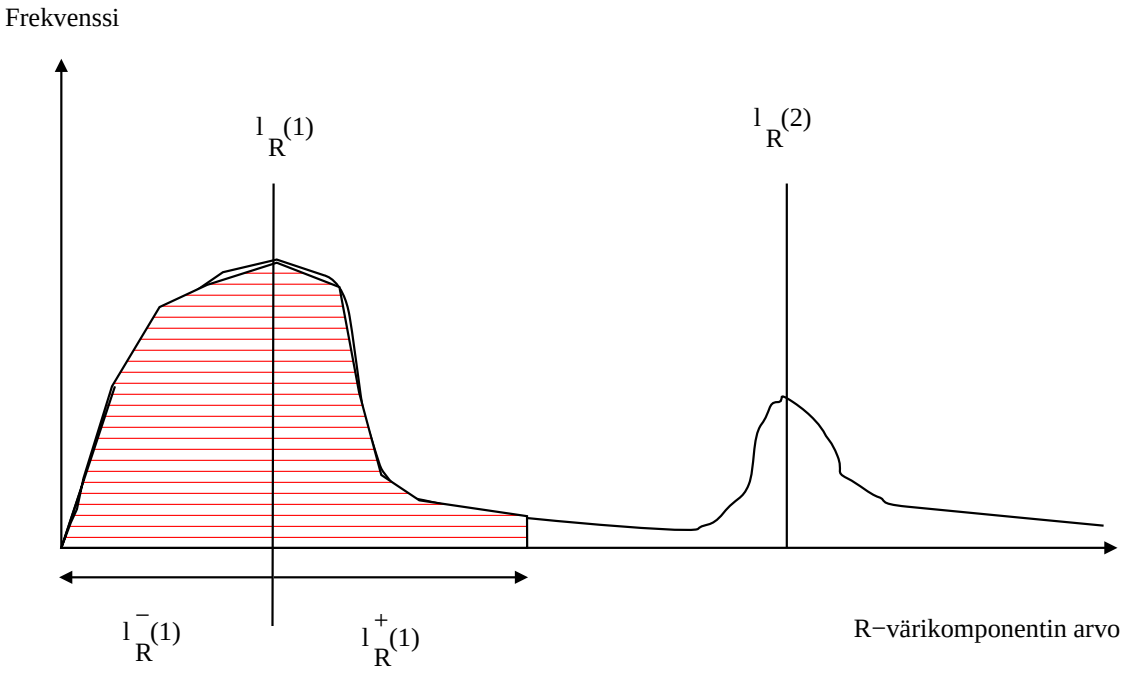
Hieman on epävarmaa, olisivatko tulokset samansuuntaisia, mikäli myös RGB pohjainen kuva jaettaisiin eri värikomponentteihin ja segmentoitaisiin alkuperäinen kuva näiden värikomponenttikuvien perusteella. Tämänkin erotusfunktion ongelmana on alueen tasaisesti muuttuva väri itsessään ja viereisessä alueessa.

Hieman samankaltainen ajatus edellisen kanssa on laajenevan alueen ja pisteen *värietäisyyttä* (Porikli 2002) käyttävä menetelmä. Värietäisyydellä tarkoitetaan alueen ja pisteen värin välistä eroa, kun tarkastellaan alueen ja pisteen värikomponenttien histogrammin maksimien etäisyyksiä toisistaan. Kuvassa 16 on esitetty alueen histogrammi yhden värikomponentin osalta.

Esimerkiksi RGB-väriavaruudessa lasketaan alueen ja värin välille kolme eri värietäisyyttä, yksi kutakin värikomponenttia kohden. Ensiksi lasketaan kaksi etäisyyttä kaikille alueen histogrammin maksimeille ja jokaiselle värikomponentille (yhteensä: $2 \times maksimit \times komponentit$):

$$\begin{aligned} l_k^-(i) &= \tfrac{1}{2}(l_k(i) - l_k(i-1)) \\ l_k^+(i) &= \tfrac{1}{2}(l_k(i+1) - l_k(i)) \end{aligned} \tag{14}$$

missä l^- on pienin etäisyys maksimiin vasemmalta puolelta ja vastaavasti l^+ oikealta, alaindeksillä k tarkoitetaan kutakin värikomponenttia (R, G ja B) ja l_k on värikompo-



Kuva 16: Yhden värikomponentin frekvenssi väriarvon funktiona ja laskettavat suureet. Maksimien lukumäärä $i_R = 2$.

nentin maksimikohta histogrammissa. Seuraavaksi määrätään kullekin värikomponentille ($k \in \{R, G, B\}$) etäisyys l_k aloituspisteestä a:

$$l_k = \begin{cases} l_k^-(i) & jos \quad l_k^- < I_k(a) \leq l_k(i) \\ l_k^+(i) & jos \quad l_k < I_k(a) \leq l_k^+(i) \end{cases} \tag{15}$$

missä $I_k(a)$ on värikomponentin intensiteetti aloituspisteessä a. Näitä käyttämällä saadaan laskettua ero seuraavasti:

$$\delta(x) = \sum_k i_k \log(1 + \frac{|I_k(a) - I_k(x)|}{l_k}) \tag{16}$$

missä i_k on histogrammin perusteella määrätty värimaksimien lukumäärä, l_k on kaavan 15 mukaan määrätty värietäisyys kullekin osavärille ja I_k on värin intensiteetti alueen aloituspisteessä a ja pisteessä x.

Esimerkiksi erään alueen histogrammissa on kaksi maksimia ($i_k = 2$) ja ne ovat taulukon 1 osoittamissa paikoissa. Tällöin saadaan yhteensä kaksitoista l^- ja l^+ arvoa (maksimeita kaksi

Taulukko 1: Alueen historgrammin kaksi maksimikohtaa.

$i \backslash k$	R	G	B
1	$l_R(1) = 15$	$l_G(1) = 30$	$l_B(1) = 50$
2	$l_R(2) = 25$	$l_G(2) = 35$	$l_B(2) = 75$

ja värikomponentteja kolme), kullekin värille neljä, joista punaisen arvot:

$$\begin{aligned}
l_R^-(1) &= \tfrac{1}{2}(l_r(1) - l_r(0)) = \tfrac{1}{2}(15 - 0) = 7.5 \\
l_R^+(1) &= \tfrac{1}{2}(l_r(2) - l_r(1)) = \tfrac{1}{2}(25 - 15) = 5.0 \\
l_R^-(2) &= \tfrac{1}{2}(l_r(2) - l_r(1)) = \tfrac{1}{2}(25 - 15) = 5.0 \\
l_R^+(2) &= \tfrac{1}{2}(l_r(3) - l_r(2)) = \tfrac{1}{2}(255 - 25) = 115.0
\end{aligned}$$

Lasketaan etäisyydet $l_k, k \in \{R, G, B\}$, kun aloituspisteellä arvot R=22,G=34,B=55 saadaan:

$$l_R = \begin{cases} 7.5 & 7.5 < 22 \leq 15 \\ 5 & 15 < 22 \leq 5 \\ \hline 5 & 5 < 22 \leq 25 \\ \hline 115 & 25 < 22 \leq 115 \end{cases}$$

Muut värit lasketaan vastaavasti. Tämän jälkeen voidaan laskea pisteen väriarvon ero lasketun etäisyyden avulla. Näin saadaan pisteen $\{R = 19, G = 25, B = 45\}$ punaiselle värille:

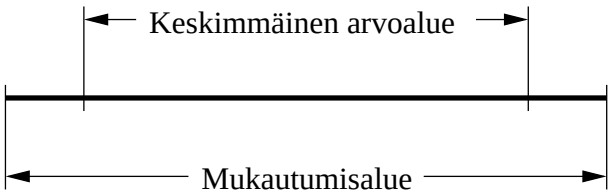
$$\delta(19)_R = 2\log(1 + \tfrac{|22-19|}{5}) = 0.408$$

Kaavan 16 mukaan tulee laskea summa kaikille kolmelle eri osavärille, jolloin saadaan lopullinen ero. Poriklin mukaan menetelmä toimii useissa tapauksissa hyvin.

3.3.2 Dynaamiset kynnysarvot

Pisteen erottaminen naapureista vaihtelee kuvan eri osissa johtuen värisävyjen erilaisista painotuksista kuvan alueella. Kynnysarvo voidaan laskea myös algoritmin edetessä (Chang & Li 1994). Aluksi kuva jaetaan pieniin samankokoisiin alueisiin, joiden oletetaan olevan homogeenisia. Nämä alueet yhdistetään toisiinsa, kunnes kynnysarvo ylittyy. Pisteen väriarvoa voidaan verrata myös alueen väriarvon varianssilla laajennettuun väriarvojoukkoon. Tällöin varianssi päivittyy jatkuvasti, kun uusia pisteitä lisätään alueeseen. Varianssilla on jokin yläraja, joka estää liian poikkeavien pisteiden lisäämisen alueeseen.

Dynaamisuus tulee siitä, että kynnysarvo päivitetään aina kun uusi alue yhdistetään vanhaan (joka mahdollisesti sisältää useita pikkualueita). Mainitut kynnysarvot lasketaan automaattisesti prosessin edetessä. Alueita yhdistetään mikäli niille lasketut paikkasidonnaiset kynnysarvot ovat sallituissa rajoissa. Rajat määritellään siten, että kummallekin alueelle lasketaan alueen väriarvojen hajonnan perusteella *mukautumisalue* (adaptive range), jonka sisällä toisen keskimääräinen alueen arvo (X) tulee olla. Käyttäjä määrittelee *mukautumiskertoimen* λ, joka kertoo kuinka suuri vaihteluväli keskimääräisellä arvolla saa olla. Mukautumiskertoimen arvo vaihtelee välillä $\lambda \in [0..1]$. Jos alueet ovat yhdistettävissä, toisen alueen keskimääräinen arvo tulee olla mukautumiskertoimen avulla korjatulla välillä toisella alueella. Alueiden yhdistämistä jatketaan, kunnes ei löydy enää yhdistettävissä olevia vierekkäisiä alueita. Kuvassa 17 esitetty mukautumisalue ja mukautumiskertoimella supistettu alue. Mukautumisalue lasketaan prosessin edetessä ja kertoimella supistettu alue määrää naapurialueen keskiarvon sallitut rajat.



Kuva 17: Mukautumisalue ja kertoimella supistettu toisen alueen arvoalue.

Oletetaan esimerkiksi, että alueet A_1 ja A_2 ovat vierekkäin ja alueen A_1 mukautumisalue on [4..7] ja mukautumiskertoimen λ arvo on 0.50. Tällöin A_2 keskiarvon X_2 tulee sijaita 50% sisällä A_1:n mukautumisaluetta laskettuna sen keskiarvon suhteen, jolloin X_2 voi vaihdella välillä [5,6]. Samoin lasketaan alueen A_2 keskiarvon vaihteluväli suhteessa A_1:een. Mikäli molemmat keskiarvot sopivat naapurialueen mukautumisalueelle, alueet voidaan yhdistää. Chang & Li käyttivät arvoa $\lambda = 0.80$, koska heidän mukaansa se tuotti riittävän hyviä tuloksia. Arvolla $\lambda = 0.80$ esimerkkitapauksessa vaihteluväliksi alueelle A_1 saadaan $X_2 \in [4.4, 6.6]$.

3.4 Jälkiprosessointi

Jälkiprosessoinnissa alueista pyritään muodostamaan järkeviä kokonaisuuksia. Kohteiden käsitteellinen mallintaminen tulee yleensä mukaan tässä vaiheessa. Esimerkiksi ihmisen seuranta kuvasta (Fan *et al.* 2001) voi perustua ihmisen jakamiseen eri kehonosiin kuten pää,

torso ja raajat, ja määrittämällä niille joitain yhteenkuuluvuuskriteereitä (asentoja). Tässä on sellainen ongelma, että nisäkkäillä yleensä on neljä raajaa, torso ja pää, jolloin lajin tunnistettavuus voi olla vaikeaa (jos sillä on merkitystä sovelluksen kannalta).

Kun tiedetään millainen olio kuvassa on, sen seuraaminen voi helpottua liikkeen ennustettavuuden parantuessa. Hälytysjärjestelmät voivat toimia tämän perusteella, joko hylkäämällä tapahtuman (tunnistetaan kissa) tai hälyttämällä apua (tunnistetaan ihminen).

4 AIKADIMENSIOINEN LAAJENNUS

Alueenlaajennusta on käytetty pääasiassa yksittäisen kuvan segmentointiin. Kohteenseurannassa 2D-alueenlaajennusta voidaan käyttää kohteen tai sen osan tunnistamisen apuna seurannan edetessä. Tällöin kohde määritellään alueeksi, jonka väri ja sen mahdollinen hajonta ovat seurannan kriteerinä. Kohteenseuranta alkaa aloituspisteen tai -pisteiden valinnalla, jolloin tulee määrätyksi alueen väriarvo ja hajonta. Kun seuranta etenee kuvasarjan seuraavaan kuvaan, täytyy etsiä oikean värisävyn omaava aloituspiste ja aloittaa alueenlaajennus tämän pisteen ympärille. Koska 2D-algoritmissa tunnistus tapahtuu pelkän alueen värin tai muun edellä mainitun pikselin arvottamiseen liittyvän kriteerin perusteella, on mahdollista, että seuranta-algoritmi eksyy seuraamaan jotain muuta aluetta. Tällöin tuntuisi luontevalta käyttää edellisen alueen pisteitä seuraavan kuvan aloituspisteinä (*pikselilokaatioina*), jolloin saadaan 3D-alueenlaajennusalgoritmi. Seuraavaksi esitellään kohteenseuranta-algoritmit ja niiden jälkeen alueen laajentumiskriteerit.

4.1 Algoritmeja

Kohteen seuranta-algoritmina 2D-tapauksessa on käytetty seuraavaa:

1. Hae kuvasarjan ensimmäinen kuva ja kysy käyttäjältä aloituspiste tai -pisteet.
2. Toista järjestyksessä kaikille kuvasarjan kuville.
   (a) Laajenna alue alueenlaajennusalgoritmin mukaisesti aloittaen aloituspisteestä.
   (b) Etsi aloituspiste seuraavalle kuvalle käyttäen saadun alueen koko- ja sijaintitietoja.

Aloituspisteen laskenta seuraavalle kuvalle tapahtuu yksinkertaisesti laskemalla laajennetun alueen peittämän suorakaiteen keskipiste. Koska aloituspiste voi sijaita seuraavassa kuvassa haetun alueen ulkopuolella, pistettä haetaan 8-naapuristoa käyttäen laajentamalla naapuriston peittämää aluetta, kunnes löytyy piste, joka täyttää arvottamiskriteerit tai jokin ennalta asetettu raja etsinnän laajentumiselle tulee vastaan. Parempi menetelmä olisi laskea uusi aloituspiste liiketiedon perusteella, mutta tässä ongelmana on tarpeeksi stabiilin keskipisteen

löytäminen. Lisäksi olisi oletettava, että kohde ei paljasta sopivanväristä pintaa yllättäen suuria määriä. Ongelmia tälläinen alueen etsintä tuottaa, mikäli alue sattuu vuotamaan muualle kuin on tarkoitus, kuten kävi kuvassa 15. Tällöin alueen keskipiste ei enää vastaa seurantaan valitun kohteen keskipistettä.

Kun 2D-algoritmi laajennetaan käyttämään aikaa kolmantena ulottuvuutena, saadaan seuraavanlainen 3D-algoritmi:

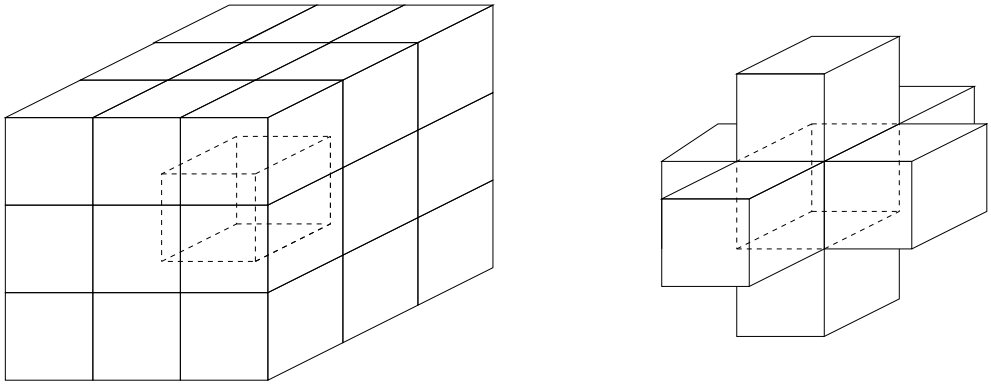
1. Hae kuvasarjan ensimmäinen kuva ja kysy käyttäjältä aloituspiste tai -pisteet.
2. Laajenna alue alueenlaajennusalgoritmin mukaisesti ja ota alueen pikselilokaatiot talteen.
3. Toista järjestyksessä kaikille kuvasarjan kuville, niin kauan kuin laajentumiskriteeri pätee.
    (a) Laajenna alue alueenlaajennusalgoritmin mukaisesti käyttäen edellisen alueen pikselilokaatioita siemenpisteinä ja käyttäen naapureina nykyisen kuvan pikseleitä.

Algoritmi on samankaltainen kuin tavanomaisessa alueenlaajennuksessa. Ensimmäisen kuvan alue tulee laajentaa 2D algoritmilla. Pisteen arvottamisfunktiossa tulee eroa, johtuen siitä, että naapuripisteet muodostavat 26-naapuriston (kuution) tai 6-naapuriston (pisteen naapurit ovat sen jokaisella sivulla), eivätkä 8- tai 4-naapuristoa kuten tason tapauksessa. Eroa algoritmissa tulee myös siitä, että mahdollisia siemenpisteitä voi tulla hyvinkin paljon samalle alueelle. Siemenpisteiksi ei kuitenkaan kannata valita sellaisia pisteitä, joiden kaikki naapuripisteet on jo sisällytetty alueeseen. 3D-menetelmässä alue voi laajeta “yli rajojen”, mutta kohdetta ei välttämättä hukata (kuva 18).

Pisteen arvottaminen tapahtuu käytännössä *xy*-tasossa ja ajan suhteen yhteen suuntaan. Tällöin naapuripisteitä on 17 (8+9) tai 5 (4+1). Edellisessä tapauksessa käytetään 8-naapuristoa vastaavaa ja jälkimmäisessä 4-naapuristoa vastaavaa laajentumissuuntaa. Kuvassa 19 vasemmalla kuvattu 27-naapuristo ja oikealla 6-naapuristo. Aloituspiste on merkitty katkoviivalla.

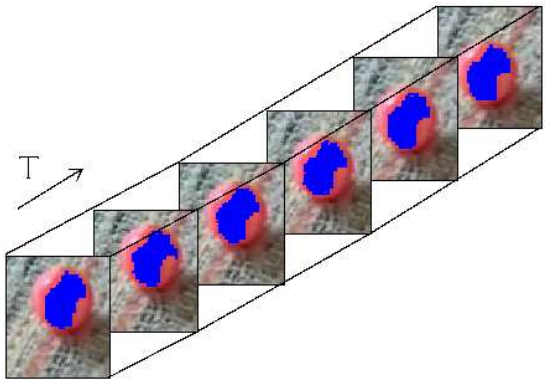
3D-algorimi käy sellaisenaan kohteen seurantaan. Kuvasarja voi tietenkin olla sellainen, että kohdetta ei tällä tavoin voi seurata, jolloin menetelmästä ei ole hyötyä. Tämä tapahtuu silloin, kun kappale liikkuu niin paljon, että seuraavassa kuvassa edelliset pikselilokaatiot

Kuva 18: Alueen laajentuminen 3D tapauksessa yli rajojen. Ensimmäinen kuva ennen laajentumista, toisessa alue laajenee oveen ja kolmannessa alueet eronneet.



Kuva 19: Naapuriston muodostuminen.

eivät osu päällekäin seuratun alueen kanssa. Kuvassa 20 nähdään sarja kuvia, jotka peräjälkeen asettamalla havainnollistavat 3D-algoritmin etenemistä muodostamalla kolmiulotteiden kappaleen.



Kuva 20: Kappaleen muodostuminen kuvakenttiä peräjälkeen asettamalla.

Tarkentamalla 3D-algoritmia saadaan:

1. Laajenna ensimmäinen kuva 2D-algoritmin mukaisesti.
2. Leimaa nykyisessä kuvakentässä pisteet, joiden sijainti on sama kuin edellisen kuvakentän alueella, alueensa mukaisesti.
3. Lisää listaan pisteet, jotka on leimattu ja joilla on vähintään yksi leimaamaton naapuripiste.
4. Toista, kunnes lista on tyhjä:
   (a) Poista ensimmäinen piste x listasta.
   (b) Testaa x naapurit:
      i. Jos kaikilla x:n leimatuilla naapureilla on sama leima.
         A. Leimaa x kyseisellä leimalla.
         B. Päivitä alueen värikeskiarvo (harmaasävy).
         C. Lisää x:n naapurit listaan arvonsa mukaisesti, mikäli eivät ole jossain alueessa tai listassa.
      ii. Muutoin
         A. Leimaa x reunaleimalla (eli piste ei kuulu mihinkään alueeseen).

Yoshida & Shimosato (2001) käyttävät aikaa kolmantena ulottuvuutena käyttäessään *vesiputous*-algoritmia (watershed) alueen seurantaan.

4.2 Laajentumiskriteerit

Laajentumiskriteerit ovat samankaltaiset kuin kuvatasolle suoritetulla arvottamisella. Ainoastaan mahdollinen spatiaalinen komponentti täytyy laskea laajentaen kolmanteen ulottuvuuteen, mikä tarkoittaa ainoastaan kaavan 8 mukaisen kriteerin sovittamista aika-akselille. Tällöin voidaan käyttää samoja matriiseja, mutta lisätään t-termit, jolloin lasketaan gradientti seuraavasti:

$$\delta(x) = \sqrt{{G_x}^2 + {G_y}^2 + {G_{t_x}}^2 + {G_{t_y}}^2} \tag{17}$$

missä G_{t_x} on gradientti tx-tasossa ja G_{t_y} ty-tasossa.

Useamman pisteen tapauksessa pikselin valinta on tehty sellaiseksi, että niiden värieron pitää olla suurempi kuin kaavojen 5 ja 13 mukaan laskettu väriero. Tällä varmistetaan, että pisteiden hajonta on mahdollisimman suuri. Pisteitä käytetään siten, että jokaiselle SSL:ään lisättävälle pisteelle suoritetaan vertailu kaikkiin valittujen pisteiden väriarvoihin ja piste hyväksytään mukaan, mikäli lisättävän pisteen väriarvo on sallituissa rajoissa. Hajontaa ei siis käyteta, kuten alkuperäisessä menetelmässä (Adams & Bischof 1994) ehdotetaan.

Alueen laajentumisen rajoitusfunktiona on mahdollisuus käyttää alueen koon muuttumista kahden edellisen kuvakentän mukaisesti siten, että edellisen alueen leveys ja korkeus arvoista vähennetään jälkimmäisen alueen $leveys \times 1.05$ ja $korkeus \times 1.05$, ja sallitaan laajennuksen etenevän kyseisen määrän verran. Esimerkiksi jos vanha leveys olisi 100 ja nykyinen 110, niin saadaan $|105 - 110| = 5$ ja alueen vasen reuna saisi olla 5 pistettä enemmän vasemmalla ja vastaavasti oikeanpuoleinen reuna 5 pistettä oikealla entisen alueen rajoista. Samalla tavalla lasketaan sallittu koon muutos korkeussuunnassa.

5 ALUEENLAAJENNUSTEKNIIKOIDEN VERTAILUA

Alueenlaajennustekniikoita vertaillaan seuraamalla kuvasarjasta osoitettua aluetta niin kauan, kunnes seuranta epäonnistuu. Laajennusalgoritmi pysyy samana, mutta algoritmin parametrit muuttuvat hieman eri ajokerroilla. Tarkoituksena ei ole saavuttaa mahdollisimman täydellistä liikkuvan kuvan aluetta, vaan seurata aluetta niin monessa kuvakentässä kuin mahdollista.

Testiaineistona on käytetty videokuvaa lähes yksivärisestä voimakkaan punaisesta pallosta, joka pysyy paikallaan (testivideo 1), mutta kamera liikkuu (kuva 21). Toisena testivideona on käytetty kissan kävelyä talon kuistilla, jolloin sekä seurattava kohde että kamera liikkuvat (testivideo 2). Kolmas testivideo on kissan tepastelua nurmikolla (testivideo 3). Viimeisessä videossa kissan seuranta lopetettiin ennalta asetettuun kuvakenttään, koska kissa peittyy puun ja pensaikon taakse kyseisen kuvakentän jälkeen.

Kuva 21: Testivideokuvat: punainen pallo (vasen), kissa kuistilla (keskellä) ja kissa nurmikolla.

Seuraavassa kappaleessa esitellään vertailualgoritmin toteutusta, sen jälkeen saatuja tuloksia ja lopuksi tulosten perusteella tehtyjä johtopäätöksiä.

5.1 Vertailuohjelma

Vertailut suoritettiin ajamalla testiohjelmaa eri parametrein, joilla pystyy valitsemaan kynnysarvon, 2D/3D-menetelmän, värin koodauksen, suodatuksen, naapuruuden, SSL-lajittelun ja algoritmin. Kaikkia mahdollisia yhdistelmiä ei testattu, koska tarkoituksena oli löytää jokin yhdistelmä, jolla seuranta onnistuu silmämääräisesti täydellisenä. Tämä tarkoittaa sitä,

että algoritmi ei hukkaa asetettua kohdetta missään vaiheessa. Algoritmisesti hukkaamisen kriteerinä on siemenpisteen löytymisen epäonnistuminen. Alueenlaajennuksen aloituspiste tai -pisteet ja aloittava kuvakenttä ovat kaikilla ajokerroilla samat. Tulokset ovat monen pisteen tapauksessa samaan sävyyn perustuvia. Kuvassa 22 nähdään kuinka alue laajenee, kun käytetään useampaa pistettä ja useampaa värisävyä. Apua monivärisyyden käytöstä ei kuitenkaan lopputuloksen kannalta havaittu, mikä johtunee siitä, että kuvasarjan edetessä valoisuus muuttuu hieman ja tiukemmat rajoitukset (pieni kynnysarvo) estävät laajentumisen. Testivideon 2 tapauksessa 2D-menetelmällä ei löydetty sellaista yhdistelmää, että kissaa olisi pystytty seuraamaan kuvasarjan loppuun asti.

Kuva 22: Kissa aluelaajennettu usean värisävyn avulla.

Ohjelmalle voi antaa seuraavia asetuksia:

- Kynnysarvo
- 2D/3D algoritmin valinta
- Värin koodaus (RGB tai HSI)
- Suodatus päälle/pois
- Naapuruus (4 tai 8)
- Siemenpisteen hakemisalueen koko (säde)
- SSL järjestys päälle/pois
- Alueenlaajennuksen rajaus päälle/pois

Molemmille väriavaruuksille väriero on laskettu kaavan 13 mukaisesti. Gradientin laskentaan on käytetty kaavaa 8. Mikäli gradientti on valittu, käytettiin siemenpisteen hakualgoritmissa oletusarvoa 0.04, koska gradienttia ei voi käyttää siemenpisteen valinnan kriteerinä. Tällöin voitaisiin valita piste, jonka ympäristö on tasainen, mutta väriarvo on jotain muuta kuin seuratun kohteen.

5.2 Tuloksia

Taulukossa 2 on tuloksia pallon seurannasta (testivideo 1). Algoritmien nopeus muuttuu sen mukaan kuinka eri ominaisuuksia on otettu mukaan laskentaan. Käytettäessä arvottamisfunktiona gradienttia, tulokset eivät olleet lupaavia, joten niitä ei montaa kirjattu. Syynä gradientin avulla saatuihin heikkoihin tuloksiin on alueen liiallinen laajentuminen, joka johtuu jyrkkien reunojen puutteesta. Samoin alueen laajentumisen rajoittaminen ei tuonut apua alueen liialliseen laajentumiseen, koska alue ei laajentunut tarpeeksi nopeasti, jolloin kohde hukattiin helposti.

Tulokset on taulukoitu siten, että näkyvissä on:

- Kynnysarvo
- 2D/3D algoritmi
- Värin koodaus
- Suodatus
- SSL (katso kappale 3 Alueenlaajennus)
- Kuvakentät
- Epäonnistumisprosentti
- Aika

Kynnysarvo on kaavan 13 tapauksessa normalisoitu välille [0..1], missä pieni lukuarvo tarkoittaa värien samankaltaisuutta (eli värien ero on pieni). Gradientin laskennassa sille on käytetty omaa kynnysarvoaluetta. Suodattimena on käytetty keskiarvosuodatusta (mean) alueen ehdokaspisteelle ja vertailupisteelle laskettuna. SSL tarkoittaa, onko

alueenlaajennusalgoritmin käyttämä listan järjestäminen käytössä. Kuvakentät-sarakkeessa olevat lukuarvot tarkoittavat epäonnistuneiden kenttien suhdetta kaikkiin kuvakenttiin ja epäonnistumisprosentti-sarakkeessa näkyy vastaava suhde prosentteina. Epäonnistuminen tarkoittaa sitä, että algoritmi ei löydä sopivaa aloituspistettä (2D) tai -aluetta (3D) seuraavasta kuvakentästä. Aika-sarakkeessa on ilmoitettu suhteellinen suoritusaika kyseiselle esimerkille. Taulukoiden väliset ajat eivät ole vertailukelpoisia. Algoritmien nopeus PentiumII 466MHz suorittimella ja Linux:lla varustetussa tietokoneessa seurannan aikana oli noin 15 kuvakenttää sekunnissa (*fps*), jota voi pitää vertailuaikana (1.00). Jos siis aika sarakkeessa on merkintä 2.15, niin fps putoaa n. 7:ään. Taukoissa on käytetty myös seuraavia merkintöjä eri parametrien asetuksista.

1. Käytetty monen pisteen arvottamisfunktiota.
2. Käytetty gradientti-arvottamisfunktiota.
3. Rajoitettu alueenlaajennus käytössä.
4. Käytetty laajempaa siemenpisteen etsintäaluetta (säde 25 => 50).

Taulukot ovat koottu lohkoista, jotka eroavat tuloksen, kynnysarvon tai menetelmän mukaan. Lohkot on järjestetty seuraavan listan mukaisesti sekä sisäisesti että keskenään:

1. Pieni epäonnistumisprosentti
2. Pieni kynnysarvo
3. Nopeus

Kynnysarvon pienuudella on merkitystä siinä mielessä, että pienellä kynnysarvolla onnistunut seuranta ei levitä aluetta liian laajalle. Jos alue leviää liian laajalle on mahdollista hukata kohde. Pienimmän kynnysarvon omaavat onnistuneet seurannat ovat taulukon alussa ja nopeusjärjestyksessä.

5.2.1 Taulukoiden tulkinta

Taulukoihin on koottu testivideoiden ajokerrat eri parametrein. Tärkein kenttä taulukoissa on epäonnistumisprosentti (E-%), joka arvolla nolla tarkoittaa, että algoritmi ei hukannut aluet-

Taulukko 2: Testivideo 1 / Pallo.

Kynnysarvo	2D/3D	Koodaus	Suodatus	SSL	Kuvakentät	E-%	Aika
0.01 [1]	2D	HSI	Ei	Ei	0 / 670	0	1.00
0.01 [1]	3D	HSI	Ei	Ei	0 / 670	0	1.06
0.01 [1]	2D	HSI	Ei	Kyllä	0 / 670	0	2.15
0.01 [1]	3D	HSI	Ei	Kyllä	0 / 670	0	3.34
0.02 [1]	3D	HSI	Ei	Ei	0 / 670	0	1.00
0.02	3D	HSI	Ei	Kyllä	0 / 670	0	2.36
0.02	2D	HSI	Kyllä	Kyllä	2 / 670	0.3	2.98
0.02	2D	HSI	Kyllä	Ei	3 / 670	0.5	0.79
0.02	2D	HSI	Ei	Kyllä	3 / 670	0.5	1.39
0.02	2D	HSI	Ei	Ei	4 / 670	0.6	0.75
0.02	2D	RGB	Kyllä	Kyllä	5 / 670	0.8	0.77
0.02	2D	RGB	Ei	Kyllä	20 / 670	3.0	0.70
0.02	2D	RGB	Kyllä	Ei	113 / 670	16.8	0.70
0.02	2D	RGB	Ei	Ei	232 / 670	34.5	0.68
0.02	3D	HSI	Ei	Ei	547 / 670	81.6	0.70

Taulukko 3: Testivideo 2 / Kissa kuistilla.

Kynnysarvo	2D/3D	Koodaus	Suodatus	SSL	Kuvakentät	E-%	Aika
0.02	3D	RGB	Ei	Ei	0 / 1178	0	1.08
0.02	3D	RGB	Ei	Kyllä	0 / 1178	0	2.02
0.02 [1,3,4]	2D	RGB	Kyllä	Ei	22 / 1178	1.9	1.06
0.02 [1,3,4]	2D	RGB	Ei	Kyllä	74 / 1178	6.3	1.43
0.02 [1,3,4]	2D	RGB	Kyllä	Kyllä	30 / 1178	2.6	1.96
0.02 [1]	2D	RGB	Ei	Kyllä	168 / 1178	14.3	1.26
0.01 [4]	2D	RGB	Ei	Kyllä	256 / 1178	21.8	1.00
80 [1,2,3,4]	2D	RGB	Kyllä	Ei	280 / 1178	23.8	2.08
0.03	2D	RGB	Ei	Kyllä	371 / 1178	31.5	1.96
0.03	2D	RGB	Kyllä	Kyllä	317 / 1178	26.9	7.22
0.01	3D	RGB	Ei	Ei	985 / 1178	83.6	1.00

Taulukko 4: Testivideo 3 / Kissa nurmella.

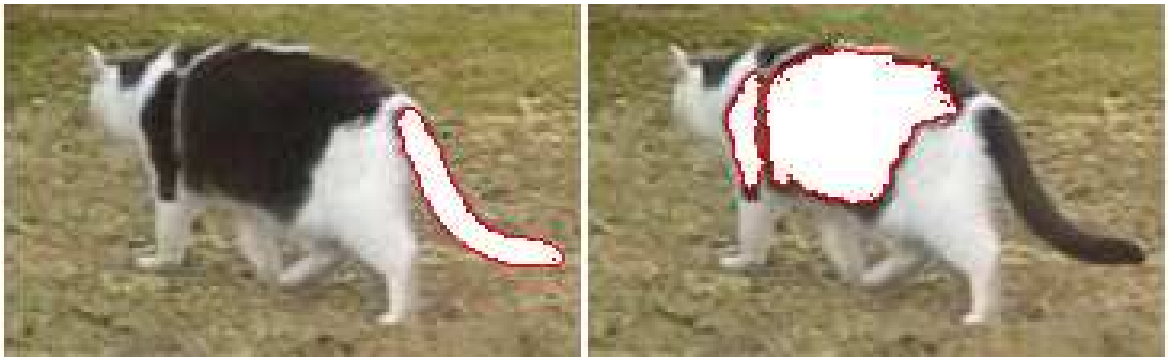
Kynnysarvo	2D/3D	Koodaus	Suodatus	SSL	Kuvakentät	E-%	Aika
0.03 [1]	3D	RGB	Ei	Ei	0 / 800	0	1.00
0.03 [1]	3D	RGB	Ei	Kyllä	0 / 800	0	1.88
0.04	3D	RGB	Ei	Ei	0 / 800	0	1.00
0.04	3D	RGB	Ei	Kyllä	0 / 800	0	2.00
0.05	2D	RGB	Kyllä	Ei	0 / 800	0	1.00
0.05	2D	RGB	Kyllä	Kyllä	0 / 800	0	3.65
0.04	2D	RGB	Kyllä	Kyllä	1 / 800	0.1	1.67
0.04	2D	RGB	Kyllä	Ei	1 / 800	0.1	1.02
0.05	2D	RGB	Ei	Kyllä	4 / 800	0.5	1.50
0.05	2D	RGB	Ei	Ei	5 / 800	0.6	1.00
150 [2]	2D	RGB	Ei	Kyllä	46 / 800	5.6	1.93
150 [2,3]	2D	RGB	Ei	Ei	60 / 800	7.5	1.00

ta kertaakaan koko videoleikkeen toiston aikana. Taulukossa ovat molemmat algoritmit tulosten mukaan paremmuusjärjestyksessä, jotta niiden vertailu samankaltaisilla parametreilla olisi helpompaa. Mikäli epäonnistuprosentti on nolla, määrää paremmuuden kynnysarvon pienuus ja algoritmin suorituksen nopeus. Taulukot on myös jaettu sisäisesti lohkoihin, jotta samankaltaiset tulokset erottuvat helpommin. Testivideon 1 tapauksessa (pallo) taulukko on jaettu viiteen eri lohkoon, jotka ovat määräytyneet E-%:n ja kynnysarvon mukaan.

Taulukoista nähdään, että seuranta ei 2D:llä onnistunut testivideo 2:ssä (kissa kuistilla). Tässä tapauksessa seurattiin hyvän alun jälkeen muuta kuin kissaa (kuva 23). Kuitenkin, kun käytettiin alueenlaajennoksen rajoitusta, ei kissaa hukattu alussa niin nopeasti. Vaikeuksia 2D-menetelmässä testivideossa 2 tuottivat videon samanväsyiset värit kissan ja ympäristön kohteiden välillä.

Testivideon 2 tapauksessa kissa hukataan 2D-menetelmällä kesken seurannan ja seurataan ovea kissan sijaan, kunnes se oven hävittyä kuvasta saadaan taas kiinni. 3D-algoritmilla tässä tapauksessa seurataan sekä ovea että kissaa (kuva 18). Huomionarvoista jälkimmäisessä tapauksessa on, että kissaa ei kadoteta seurannan aikana. Testivideossa 3 huomionarvoista parhaasta 2D-tuloksesta on mainita, että seuranta siirtyi kissan kyljestä kissan häntään, mutta seuranta silti onnistui (kuva 24).

Kuva 23: Testivideossa 2 kissa hukattiin aina 2D-algoritmilla. Tässä kuva tapauksesta, jossa seuranta siirtynyt viereiseen kenkäpariin.



Kuva 24: Kissa testivideossa 3 samassa kuvakentässä eri algoritmeilla. Vasemmalla 2D ja oikealla 3D.

Mikäli vertailualue ei ole väriltään puhdas, RGB-koodaus ei laajenna aluetta niin helposti. Kuvassa 25 nähdään tilanne, jossa likainen väri laajenee helpommin HSI-koodauksella kuin vastaava tilanne RGB:tä käytettäessä (kuva 26). Molemmissa käytetty samaa kynnysarvoa.

Kuva 25: HSI-koodauksen tuottama alueenlaajennuksen tulos. Ensimmäisessä kuvassa seuranta on kohdallaan, toisessa kuvassa alue on laajentunut kenkäpariin. Viimeisessä kuvassa on tilanne laajennuksen loputtua.

Kuvassa 27 nähdään kuinka gradienttia (kaava 8) käyttämällä alue leviää liian laajalle (kynnysarvona käytetty 150). Viimeinen kuva on saatu käyttämällä keskiarvoistavaa arvottamisfunktiota (kaava 4 kynnysarvolla 0.04). Samantapainen tilanne kissaesimerkissä (testivideo 2) on kuvassa 28. Kuvasarjan edetessä pallo välillä hukataan, kun käytetään gradienttia arvottamisfunktiona.

Kuva 26: RGB-koodauksen tuottama alueenlaajennuksen tulos samassa kohden kuin kuvan 25 tapauksessa. Ensimmäinen kuva vastaa HSI kuvan ensimmäistä kuvakenttää ja toinen jälkimmäistä.

Kuva 27: Alueen laajentuminen gradienttia käytettäessä testivideossa 1. Ensimmäinen kuva ennen laajennusta, toinen gradienttia käyttäen ja kolmas kuva laajennettu ilman gradienttia.

Kuva 28: Alueen laajentuminen gradienttia käytettäessä testivideossa 2. Ensimmäinen kuva ennen laajennusta, toinen laajennuksen jälkeen.

Syy siihen, miksi gradientti tuottaa epäkelpoja tuloksia, on pisteen arvottaminen sen ympäristön perusteella, eikä verrattuna johonkin alkuarvomäärättyyn vertailuarvoon. Esimerkeistä nähdään, että ellei jyrkkää reunaa tietystä kohden löydy, piste sisällytetään alueeseen riippumatta sen väriarvosta. Kokeileminen eri jyrkkyyksillä (kynnysarvoilla) ei juurikaan parantanut seurannan tulosta. Alue hukattiin tai leviämisen johdosta seurattiin "väärää" aluetta. Koska gradientti 2D-menetelmällä tuotti heikkoja tuloksia (ja 3D-menetelmä käyttää 2D-menetelmää pisteiden lisäämisessä alueeseen), ei sitä toteutettu 3D-algoritmille.

5.3 Johtopäätöksiä

Aineiston perusteella voi todeta, että 3D-menetelmä seuraa kohdetta paremmin kuin 2D-menetelmä. Siihen ei oteta kantaa, onko alueenlaajennusalgoritmien tulos visuaalisesti tyydyttävä eli kuinka hyvin algoritmin löytämä alue vastaa havainnoijan näkemää aluetta.

Ajetuilla videoleikkeillä 3D-algoritmi seurasi kohdetta loppuun asti virheittä, kunhan toteutuksen parametrit oli asetettu sopivasti. 2D-toteutuksen puolesta puhuu sen soveltuvuus nopeasti liikkuvien kohteiden seurantaan, koska 3D-algoritmi hukkaa varmuudella kohteen, joka siirtyy enemmän kuin sen edellisen kuvakentän peittämän alueen verran. Suurin syy 2D-algoritmin heikkoudelle testivideossa 2 (kissa kuistilla) lienee alueen siemenpisteen etsintäfunktio, koska siemenpisteen sijainti hukkui muutaman kerran ja algoritmi seurasi siten

jotain muuta.

Tuloksista nähdään myös, että RGB-koodausta ei kannata käyttää, mikäli seurattava kohde (alue) on väriltään puhdas. Samoin SSL-listan käyttöä voisi tehostaa poistamalla siitä järjestyksen käytön. Tosin testivideossa 1 (pallo) nähdään, että 3D-menetelmällä käyttäen kynnysarvoa 0.02 listan käyttö tuottaa onnistuneen seurannan, mutta ilman listaa seuranta epäonnistuu yli 80%:sti. Toisaalta testivideossa 2 (kissa kuistilla) nähdään, että listan käyttö tuplaa suoritusajan lopputuloksen ollessa sama. Jos tingitään hieman kynnyarvossa (tai jopa käyttämällä samaa kynnysarvoa), saadaan järjestämättömällä listalla suoritusaikaa vähennettyä huomattavasti. HSI-koodauksella 3D-algoritmi pystyi seuraamaan palloa kuvasarjan loppuun asti. Kynnysarvon ollessa pienempi ja käytettäessä RGB:tä, 3D-algoritmi hukkaa pallon nopeasti riippumatta suodatuksesta tai naapuruuden asteesta.

Naapuruuden tai suodatuksen valinnalla ei näyttäisi suurta merkitystä olevan seurannan tuloksen kannalta. Laajemman (8-naapuruuden) valinta lisää algoritmin suoritusaikaa hiukan. 3D-algoritmin hyvänä puolena hitaasti liikkuvissa kohteissa on, että niitä ei hukata niin helposti kuin 2D-algoritmilla. Haittapuolena taas on, että jos alue sattuu leviämään ympäristöön, niin ympäristöäkin seurataan, niin kauan kuin se on näkyvillä, kuten kuvasta 18 nähdään.

6 YHTEENVETO

Alueenlaajennus 3D-menetelmällä voi tuoda lisätarkkuutta seurannan onnistumiseen verrattuna vastaavaan tilanteeseen 2D-menetelmään verrattuna. Menetelmän heikkoutena on kohteen hukkaaminen, mikäli kohde liikkuu enemmän kuin se peittää verrattuna edelliseen kuvakenttään. 3D-alueenlaajennos mainitulla aineistolla osoittaa sen olevan käyttökelpoinen tapauksissa, joissa liike on rauhallista. Itse alueenlaajennusalgoritmia voisi suhteellisen huoletta käyttää ilman listan järjestystoteutusta, koska sillä näyttää olevan suhteellisen vähäinen merkitys lopputuloksen kannalta, mutta nopeusetu olisi suuri (esimerkeissä 1.5-7 kertainen). Nopeusetu saattaisi pienentyä toisenlaisella tietorakenteella (esim. järjestetty puu), mutta tätä ei testattu.

Tekniikoista sen verran, että olisi mielenkiintoista vertailla pisteiden arvottamisfunktioita laajemmminkin kuin tässä työssä on ollut mahdollista. Erityisesti pinnoitteet vaikuttavat olevan tehokkaita segmentoinnin apuvälineitä (Mirmehdi & Petrou 2000). Mikäli alueen voisi määrätä täydellisesti, molemmat alueenlaajennustekniikat tuottaisivat todennäköisesti saman tuloksen seurannan onnistumisen suhteen, mikäli kappale ei liikkuisi liian nopeasti.

Mahdollisesti menetelmät kannattaisi yhdistää siten, että 3D-menetelmää käytetään niin kauan kuin mahdollista ja jos kohde hukataan eli seurattavan alueen pinta-ala menee nollaksi, etsitään se 2D-menetelmällä ja jatketaan siitä 3D:llä.

VIITTEET

Adams R., Bischof L. (1994): Seeded Region Growing, *IEEE Transactions on patter analysis and machine intelligence* **16** (6), pp. 641-647

Chang Y-L., Li X. (1994): Adaptive Image Region-Growing, *IEEE Transactions on Image Processing* **3** (6), pp. 868-872

Che E.Y., Kang M.H. (1998): Multiple Target Tracking in Clutter Backgrounds Using Self-Organizing Feature Map, *IEEE International Joint Conference on Neural Networks Proceedings*, **2**, pp. 1162-1166

Cramariuc B., Gabbouj M, Astola J. (1997): Clustering based region growing algorithm for color image segmentation *13th International Conference on Digital Signal Processing Proceedings* **2** pp. 857-860

Deng, Y.; Manjunath, B.S. (2001): Unsupervised segmentation of color-texture regions in images and video, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16**, pp. 800-810

Fan J., Yau D.K.Y., Elmagarmid A.K., Aref W.G. (2001): Automatic image segmentation by integrating color-edge extraction and seeded region growing, *IEEE Transactions on Image Processing* **10**, pp. 1454 -1466

Fränti P. (1998): Digital Image Processing *Lecture Notes* Joensuun yliopisto, Tietojenkäsittelytieteen laitos

Gambotto J.-P. (1992): A region-based spatio-temporal segmentation algorithm *Image, Speech and Signal Analysis, 11th IAPR International Conference on Pattern Recognition Proceedings*, **3**, pp. 189-192

Grinias I., Tziritas G. (2001): A semi-automatic seeded region growing algorithm for video object localization and tracking, *Signal Processing: Image Communications*, **16**, pp. 977-986

Huang C., Choo Y-R., Chung P-C. (2002): Combining region-based differential and matching algorithms to obtain accurate motion vectors for moving object in a video sequence, *22nd International Conference on Distributed Computing Systems Workshops Proceedings* pp. 202-207

Kim J.B., Kim H.J. (2002): Efficient region-based motion segmentation for a video monitoring system, *Pattern Recognition Letters 24 (2002)* pp. 113-128

Kumar S., Asari K.V., Radhakrishnan K.V. (1999): A new technique for the segmentation of lumen from endoscopic images by differential region growing *42nd Midwest Symposium on Circuits and Systems*, **1**, pp. 414-417

Lin Z., Jin J., Talbot H. (2000): Unseeded region growing for 3D image segmentation, *Visual Information Processing:Conferences in Research and Practice in Information Technology* **2**

Martínez J.M.(2002): MPEG-7 Overview (version 8), *ISO/IEC JTC1/SC29/WG11, Coding of Moving Pictures and Audio*
`http://mpeg.telecomitalialab.com/standards/mpeg-4/mpeg-4.htm`

Mirmehdi, M.; Petrou, M. (2000): Segmentation of color textures , *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2**, pp. 142-159

Montoya G., Gil C., Garcia I. (1999): Load Balancing for a Class of Irregular and Dynamic Problems: Region Growing Image Segmentation Algorithms, *IEEE Seventh Euromicro Workshop on Parallel and Distributed Processing* **16**, pp. 163-169

Porikli F. (2002): Automatic Threshold Determination of Centroid-Linkage Region Growing by MPEG-7 Dominant Color Descriptors, *IEEE International Conference on Image Processing (ICIP)*

Reynolds R.G., Rolnick, S.R. (1995): Learning the parameters for a gradient-based approach to image segmentation from the results of a region growing approach using cultural algorithms, *IEEE International Conference on Evolutionary Computation*, **2**, pp. 819-824

Shapiro L. G., Stockman G.C. (2001): Computer Vision, *Prentice Hall* New Jersey, ISBN 0-13-030796-3

Sifakis E., Tziritas G. (2001): Moving object localisation using a multi-label fast marching algorithm, *Signal Processing: Image Communications* **16**, pp. 963-976

Stewart R.D., Fermin I., Opper M. (2002): Region Growing With Pulse-Coupled Neural Networks: An Alternative to Seeded Region Growing, *IEEE Transactions on neural networks* **13** (6), pp. 1557-1562

Wang Y., Doherty J.F., Van Dyck R.E. (2000): Moving Object Tracking in Video, *IEEE 29th Applied Imagery Pattern Recognition Workshop, Image Compression and Coding* pp. 95-101

Yoshida T., Shimosato T. (2001): Motion image segmentation using 3-D watershed algorithm, *International Conference on Image Processing Proceedings*, **2**, pp. 773-776

Zhong H., Wenyin L., Li S. (2000): Interactive Tracker - A Semi-automatic Video Object Tracking and Segmentation System, *Microsoft Research China*
`http://research.microsoft.com/asia/dload_files/group/mediasearching/icme_tracking-4th.pdf`