KUNGL TEKNISKA HÖGSKOLAN
Numerisk analys och datalogi

Detektion av särdrag i bilder baserad på färginformation

av
Jonas Sjöbergh <d95-jsj@nada.kth.se>

TRITA-NA-E00XY

NADA

Nada (Numerisk analys och datalogi)
KTH
100 44 Stockholm

Department of Numerical Analysis
and Computer Science
Royal Institute of Technology
SE-100 44 Stockholm, SWEDEN

Detektion av särdrag i bilder baserad på färginformation

av
Jonas Sjöbergh <d95-jsj@nada.kth.se>

TRITA-NA-E00XY

Examensarbete i datalogi om 20 poäng
vid Programmet för Datateknik,
Kungliga Tekniska Högskolan år 2000
Handledare på Nada var Tony Lindeberg
Examinator var Tony Lindeberg

Sammanfattning

Syftet med detta examensarbete är att ta fram och utvärdera metoder för särdragsdetektion i bilder som utnyttjar färginformation. Detta har gjorts på främst två olika sätt, genom användande av färgskillnader mellan olika delar i bilden för att öka kontrasten mellan dem och genom användande av detekterade särdrags färg för att bestämma vilka särdrag som är relevanta. Ett antal olika färgmodeller har undersökts för att se hur valet av färgmodell påverkar särdragsdetektion i färgbilder. Dessutom har en realtidsimplementation för användning av färginformation vid särdragsdetektion undersökts och prövats i en konkret tillämpning.

Resultaten visar att färginformation är intressant vid särdragsdetektion och att det inte behöver påverka prestanda alltför mycket. Användning av likheten mellan detekterade särdrags färg och den förväntade färgen hos objektet vars särdrag man vill detektera visade sig vara en kraftfull metod att sålla bort särdrag i bakgrunden.

Abstract

Feature detection in images based on colour information

The purpose of this master's project is to design and evaluate methods for feature detection in images where colour information is used. Two main approaches have been used, using colour information to increase the contrast between different coloured image regions and using the colour of detected image features to help decide which features are interesting. Several different colour models have been studied to determine the influence of the choice of colour model on feature detection in colour images. Also, a real time implementation of feature detection using colour information has been examined and tested in a real application.

The results show that colour information is useful for feature extraction, with only a moderate impact on performance. Using the similarity of the colour of detected features and an expected object colour was shown to be a powerful method to eliminate background features.

Tack

Jag vill här passa på att tacka en del personer som på olika sätt hjälpt mig. De personer jag vill tacka är: Jani Niemenmaa vars hybridpyramidimplementation jag byggt vidare på, Ivan Laptev som bl.a. hjälpt mig med experimenten med handföljning samt en del programmeringsbestyr, Frej Drejhammar som bl.a. har givit mig en del färdiga kodsnuttar så jag slapp implementera de bitarna själv, min handledare Tony Lindeberg samt Lars Bretzner som fungerat som biträdande handledare under en del av examensarbetet.

Experiment med handföljning har gjorts i samarbete med Ivan Laptev, med användning av hans programvara.

Innehåll

Kapitel 1

Inledning

1.1 Introduktion

Detektion av särdrag i bilder innebär letandet efter strukturer av en viss karaktär i bilden. De vanligaste typerna av särdrag är kanter, hörn, regionen och åsar. Särdrag ger intressant information om en bild. Ett exempel på det är en streckteckning av en scen, som man kan betrakta som en kompakt sammanfattning av scenen. Strecken motsvarar kanter i bilden av scenen, vilket innebär att detektion av kanter kan vara intressant i vissa tillämpningar.

Traditionellt har särdragsdetektion vanligen utförts på gråniv̊abilder. Detta beror på flera saker, bl.a. tar färgbilder mer utrymme och det tar längre tid att arbeta med dem än med gråniv̊abilder. I många fall är särdragsdetektion ett svårt problem, särskilt i ”naturliga” bilder som ofta har stor detaljrikedom och då innehåller många särdrag. Färginformation kan i många fall vara till hjälp vid särdragsdetektion.

Detta examensarbete syftar till att ta fram och utvärdera metoder för särdragsdetektion där färginformationen i bilder används. Målet är att ta fram en metod som fungerar bättre än algoritmer som enbart utnyttjar gråniv̊ainformationen i bilder. Den primära tillämpningen är gestigenkänning, men även andra tillämpningar är av stort intresse.

Gestigenkänning är en form av människa-maskininteraktion där man med (i detta fall) handgester samverkar med en dator. Vid gestigenkänning används särdragsdetektion. För att tolka gester måste man kunna avgöra var i bilden handen befinner sig. Särdragsdetektion kan då användas som ett led i denna process, där handens delar detekteras, fingrar detekteras som åsar, fingertoppar och handflatan som regioner.

I huvudsak två olika metoder för användning av färginformation har prövats. I ena fallet används färginformationen för att öka kontrasten mellan olika delar i bilden. I andra fallet används de detekterade särdragens färg för att skilja ut de särdrag som är intressanta för tillämpningen från andra detekterade särdrag. Det går utmärkt att kombinera de båda angreppssätten.

Särdragsdetektion är ofta en tidskrävande operation. I många tillämpningar är det önskvärt att ha realtidsprestanda. Detta är extra viktigt vid interaktiva tillämpningar, t.ex. gestigenkänning. I detta arbete har även möjligheter till realtidsprestanda för särdragsdetektion, med användning av färginformation, undersökts.

Resultaten av examensarbetet visar att båda de angreppssätt som prövats för användande av färginformation i vissa fall kan förbättra särdragsdetektion i bilder avsevärt. När det gäller realtidsprestanda visar det sig att det är möjligt att detektera särdrag, med användning av färginformation, i realtid.

1.2 Rapportens struktur

I kapitel 2 ges bakgrundsinformation om olika områden som har relevans för examensarbetet. I kapitel 3 beskrivs olika metoder för användning av färg vid särdragsdetektion och i kapitel 4 presenteras resultaten av detta. Kapitel 5 behandlar en realtidsimplementation av särdragsdetektion i färgbilder och en tillämpning av detta. Slutligen, i kapitel 6, sammanfattar jag examensarbetet och presenterar de slutsatser jag dragit samt diskuterar möjligheter till fortsatt arbete inom området.

Kapitel 2

Teoretisk bakgrund

Här följer information om olika områden som är relevanta för examensarbetet. Det är inte nödvändigt att läsa alla områden för att förstå mitt examensarbete, men de spelar alla in på ett eller annat sätt.

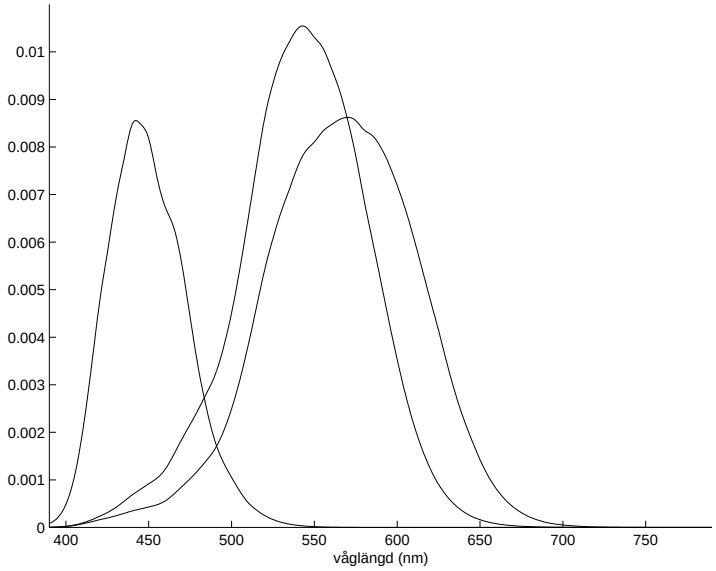
2.1 Färgmodeller

Det mänskliga synsystemet har tre sorters receptorer för färgseende. De har sin största känslighet för rött, grönt respektive blått ljus, se figur 2.1. Detta gör att människor upplever alla färger som kombinationer av färgerna rött, grönt och blått, vilka därför kallas primära färger.

Eftersom alla färger upplevs som en blandning av de primära färgerna kan man tro att man med tre typer av ljus, motsvarande de primära färgerna, skulle kunna få fram samtliga möjliga färger, men så är inte fallet. Man kan inte genom att blanda tre olika typer av ljus framställa alla färger om man inte låter våglängden (färgen) på ljusen variera.

Mänskligt färgseende innebär en inkomplett representation av våglängder. Det innebär att upplevd färg inte är det samma som ljusets våglängd. Mänskligt färgseende kan ge samma upplevda färg för ljus som består av helt olika våglängdskombinationer. Man kan också få olika upplevd färg från ljus med samma våglängd (Sisefsky 1993), beroende på omgivande färger. Hur människor upplever färg påverkas av andra faktorer än det ljus man ser, t.ex. färger i omgivningen, ljusstyrka, minnet m.m.

Eftersom ögat har tre sorters färgreceptorer har också de flesta färgmodeller tre dimensioner. Man kan se en färgmodell som en specifikation av ett koordinatsystem där varje punkt är en färg.



Figur 2.1. Känslighetsspektra för mänskligt seende. Numeriska data är hämtade från webbplatsen `http://cvision.ucsd.edu/index.htm`

2.1.1 Klassiska färgmodeller

RGB

RGB är den överlägset vanligaste färgmodellen för lagring av bilder digitalt. I RGB är varje färgvärde en trippel som talar om hur mycket av varje primär färg (Röd, Grön, Blå) som ingår.

Man kan betrakta RGB som en avbildning från ljusspektra till en tredimensionell sensorrymd:

$$C = \int_{\lambda} e(\lambda) f_C(\lambda) d\lambda \tag{2.1}$$

där $C \in \{R, G, B\}$ är sensorsvaret, $e(\lambda)$ är det ljus som faller in på sensorn och $f_C(\lambda)$ är sensorns känslighetsfunktion. Vanligen normaliserar man färgerna så att värdena ligger mellan 0 och 1.

RGB används i många sorters datorhårdvara för färghantering, t.ex. bildskärmar, scannrar, digitala kameror m.m.

CMY

CMY är i princip samma sak som RGB, om man har RGB normaliserad så att värdena ligger mellan 0 och 1 får man $(C, M, Y) = (1, 1, 1) - (R, G, B)$. CMY används främst för att visa bilder i tryck, t.ex. av färgskrivare. Vid tryck använder

man pigment som ”drar bort” (absorberar) färg från det ljus som faller in och reflekterar resultatet. Cyanfärgat (C) pigment drar bort rött ljus och magenta- (M) och gulfärgat (Y) drar bort grönt respektive blått ljus.

YIQ

YIQ är den kodning av färg som används för TV-sändningar i färg. Y-komponenten är den signal som används av en svart-vit TV, dvs intensiteten. Det mänskliga synsystemet är känsligare för förändringar i intensitet än för förändringar i färgton eller mättnad (den information som finns i de andra komponenterna), därför används större bandbredd till Y-komponenten än till de andra. Omvandling från RGB till YIQ sker genom (Gonzalez och Woods 1992):

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{pmatrix} 0,299 & 0,587 & 0,114 \\ 0,596 & -0,275 & -0,321 \\ 0,212 & -0,523 & 0,311 \end{pmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.2}$$

Opponent

Det finns undersökningar som visar att det mänskliga synsystemet har celler som reagerar på en färg och hämmas av en annan (McIlhagga 1998, Sisefsky 1993, Würtz och Lourens 2000, Westin och Westelius 1988). Dessa celler känner inte av ljus själva i ögat utan arbetar med syninformationen i ett senare led. Cellerna kallas opponentfärgskänsliga. De färgpar som förekommer som opponenter till varandra är röd-grön, blå-gul och vit-svart. En effekt av detta kan man se om man tittar intensivt på t.ex. en röd yta en stund och sedan tittar på en vit yta. Den vita yta kommer då att se grön ut.

Det finns färgmodeller som efterliknar detta beteende genom att ha en färgkanal för varje opponentpar med information om skillnaden mellan opponentfärgerna i paret. Man kan t.ex. ha en röd-grön kanal $RG = R - G$, en blå-gul kanal $BY = 2B - (R + G)$ och en vit-svart kanal $BW = R + G + B$.

HSI/HSV

HSI och HSV är färgmodeller inriktade på mänsklig upplevelse av färg, till skillnad från de tidigare nämnda färgmodellerna som är gjorda för att passa maskiner. H står för Hue = färgton, S står för Saturation = mättnad och I för Intensity (V står för Value = värde och betecknar ungefär detsamma som I).

Färgton betecknar vilken grundfärg vi upplever. Mättnad är hur ”ren” färgen är, i motsats till hur utspädd med vitt färgen är. Röd är en helt mättad färg medan rosa är samma färgton utspädd med vitt, dvs mindre mättad. Intensitet är ljusstyrka.

Att intensitetsinformationen är separat från färginformationen är praktiskt vid vissa typer av bildoperationer, t.ex. är det då lätt att öka kontrasten i en bild utan att störa färginformationen.

HSI ligger nära det sätt på vilket människor upplever färg, vilket gör det lätt att visualisera hur en färg ser ut då man får en beskrivning av färgen i HSI-modellen. Det är mycket svårare för en otränad person att visualisera hur en färg ser ut om man får beskrivningen i RGB. Detta gör att HSI är en bra modell vid tillfällen då människor ska interagera med färg.

Det finns många varianter på HSI/HSV-system. Särskilt är det bestämningen av H som varierar (Palus 1998). Detta beror på att konvertering från t.ex. RGB till HSI ger hög komplexitet i beräkningarna av H om man vill ha en bra översättning. Därför har många kompromissat och fått lättare beräkningar till priset av sämre översättningskvalitet.

En grundmodell för HSI-färgrymden är en kon, där I är höjdaxeln i konen, med svart i spetsen och vitt i mitten av konens botten. S mäter avståndet från höjdaxeln och H anger riktningen från höjdaxeln, se figur 2.2. Konvertering från RGB till detta HSI görs genom följande uttryck (Palus 1998, Gonzalez och Woods 1992):

$$H = \cos^{-1} \frac{0,5((R-G)+(R-B))}{\sqrt{(R-G)^2+(R-B)(G-B)}} \tag{2.3}$$

om $B > G$ tar man $2\pi - H$ istället (H mätt i radianer).

$$S = 1 - \frac{3min(R,G,B)}{R+G+B} \tag{2.4}$$

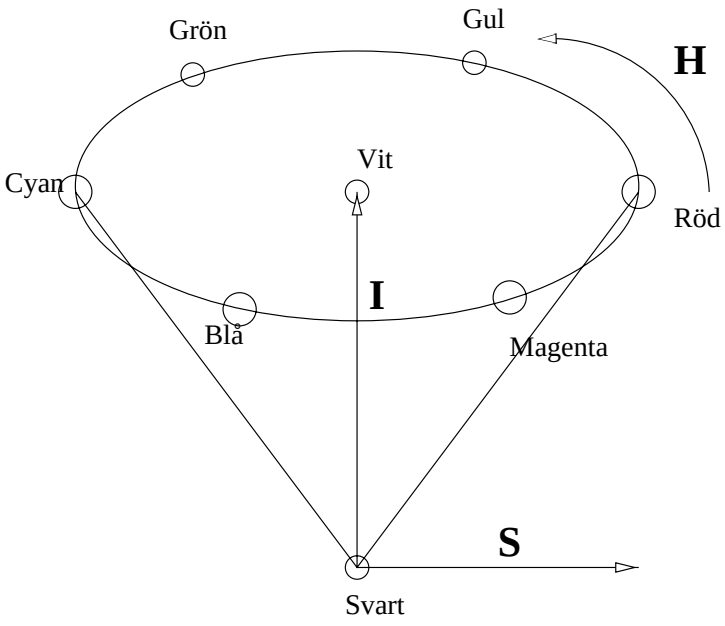
$$I = \frac{R+G+B}{3} \tag{2.5}$$

CIE XYZ

1931 tog *Commission Internationale de l'Eclairage* fram en standardiserad färgmodell baserad på mänskligt seende. Den heter $CIE\ XYZ$ eller bara kort och gott XYZ. Den definierar tre komponenter X, Y och Z som kan kombineras och på så sätt bilda alla färger människor kan se. XYZ är inga verkliga färger. Därför är det möjligt att med dessa tre komponenter framställa samtliga färger, vilket som tidigare nämnts inte är möjligt med tre verkliga färger.

Färgmodellen bygger på experimentstudier där människor fick blanda ljus med våglängderna 700 nm (rött), 546,1 nm (grönt) och 435,8 nm (blått) så att den resulterande färgen upplevdes som likadan som en sökt färg. Resultaten från försök med flera personer sammanställdes till funktioner över färgers sammansättning av de tre primära färgerna, enligt en ”standardobservatör”. Resultatet blev kurvorna i figur 2.3.

Funktionerna hade dock negativa värden för vissa färger, vilket innebar att man var tvungen att lägga till primärfärgen till den sökta färgen istället för den framblandade färgen. För att undvika detta transformerades funktionerna linjärt till en annan uppsättning funktioner, $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ och $\bar{z}(\lambda)$, för att bli av med de negativa värdena. Figur 2.4 visar dessa funktioner.



Figur 2.2. HSI-koordinatsystemet.

Färgen på en yta med reflektans $R(\lambda)$ under belysning av en belysningskälla med spektral fördelning $S(\lambda)$ blir i $CIE\ XYZ$-koordinater:

$$\begin{aligned} X &= k \int R(\lambda)S(\lambda)\bar{x}(\lambda)d\lambda \\ Y &= k \int R(\lambda)S(\lambda)\bar{y}(\lambda)d\lambda \\ Z &= k \int R(\lambda)S(\lambda)\bar{z}(\lambda)d\lambda \end{aligned} \tag{2.6}$$

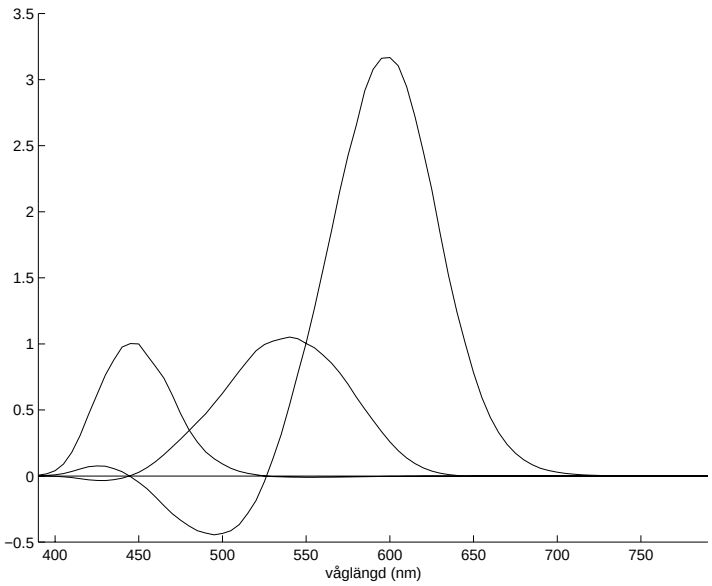
För färger som ej är reflekterade (t.ex. från en monitor) där R och S inte är definierade används:

$$\begin{aligned} X &= k \int P(\lambda)\bar{x}(\lambda)d\lambda \\ Y &= k \int P(\lambda)\bar{y}(\lambda)d\lambda \\ Z &= k \int P(\lambda)\bar{z}(\lambda)d\lambda \end{aligned} \tag{2.7}$$

där P är irradiansen hos det som avger färgen. I båda fallen är k en konstant så att $Y = 100$ för en vit referensfärg. Y är vald så att den motsvarar människors uppfattning av hur ljus en färg är, ljusa färger har högre Y-värde än mörka färger.

Om man normaliserar värdena med summan av de tre komponenterna (normalisering med intensiteten) får man:

$$(x, y, z) = \frac{(X, Y, Z)}{X + Y + Z} \tag{2.8}$$



Figur 2.3. CIE färgmatchningsfunktioner. Numeriska data är hämtade från webbplatsen `http://cvision.ucsd.edu/index.htm`

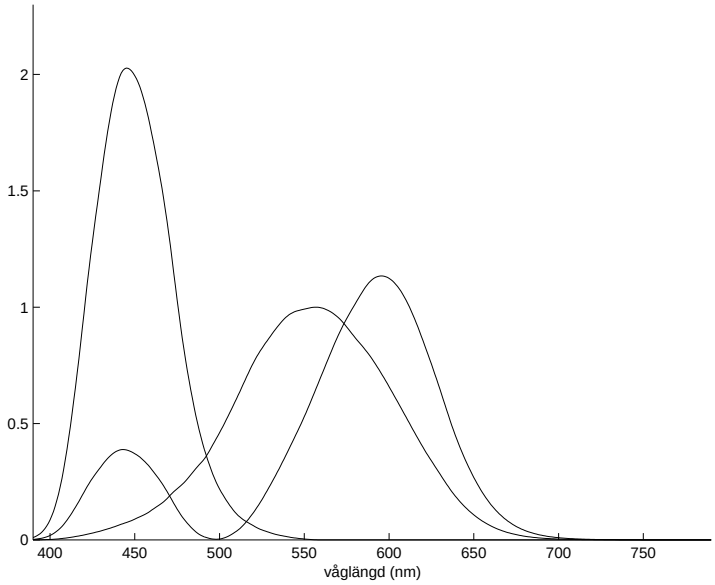
Om man känner x och y kan man lösa ut $z = 1 - x - y$. Figur 2.5 är ett diagram där x och y visas för alla möjliga färger. Det är vanligt att man anger färger med x, y och Y.

Om man tar två punkter i diagrammet, som representerar var sin färg, ligger alla färger som kan fås genom att blanda dessa två färger på linjen mellan punkterna. Detta samband ger ytterligare ett sätt att inse att tre färgkomponenter inte räcker för att framställa samtliga färger. Alla färger som kan framställas genom att kombinera tre färger ligger i triangeln som bildas av de tre färgerna och inga tre färger kan bilda en triangel som täcker hela diagrammet.

I diagrammet ligger de rena (mättade) färgerna på periferin och vit ligger i den punkt där $x = y = z = \frac{1}{3}$. Ett sätt att ange HSI-koordinater är att lägga koordinatsystemets origo i punkten för vit färg och ange H som en vinkel och S som avståndet från vitpunkten (normaliserat så att $S = 1$ ligger på periferin i H-riktningen), (Jähne 1999).

Konvertering från RGB till XYZ sker genom följande linjära transform, (Geusebroek, Smeulders och van den Boomgaard 2000):

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{pmatrix} 0,621 & 0,113 & 0,194 \\ 0,297 & 0,563 & 0,049 \\ -0,009 & 0,027 & 1.105 \end{pmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.9}$$



Figur 2.4. CIE $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ och $\bar{z}(\lambda)$. Numeriska data är hämtade från webbplatsen `http://cvision.ucsd.edu/index.htm`

2.1.2 Gaussisk färgmodell

I Geusebroek, Dev, van den Boomgaard, Smeulders, Cornelissen och Geerts (1999), Geusebroek, Smeulders och van den Boomgaard (2000) och Geusebroek, van den Boomgaard, Smeulders och Dev (2000) tas en färgmodell fram där man kan betrakta de olika färgkanalerna som gaussderivator med avseende på ljusets våglängd. De tre kanalerna blir då 0:te derivatan E, vilket blir intensiteten, förstaderivatan E_λ, en gul-blå komponent och andraderivatan $E_{\lambda\lambda}$ en röd-grön komponent.

Då man i denna färgmodell mäter den spektrala energin $E(\lambda)$ får man:

$$E(\lambda) = E^{\lambda_0,\sigma_\lambda} + \frac{\lambda}{\sigma_\lambda} E_\lambda^{\lambda_0,\sigma_\lambda} + \frac{\lambda^2}{2\sigma_\lambda^2} E_{\lambda\lambda}^{\lambda_0,\sigma_\lambda} + \ldots \tag{2.10}$$

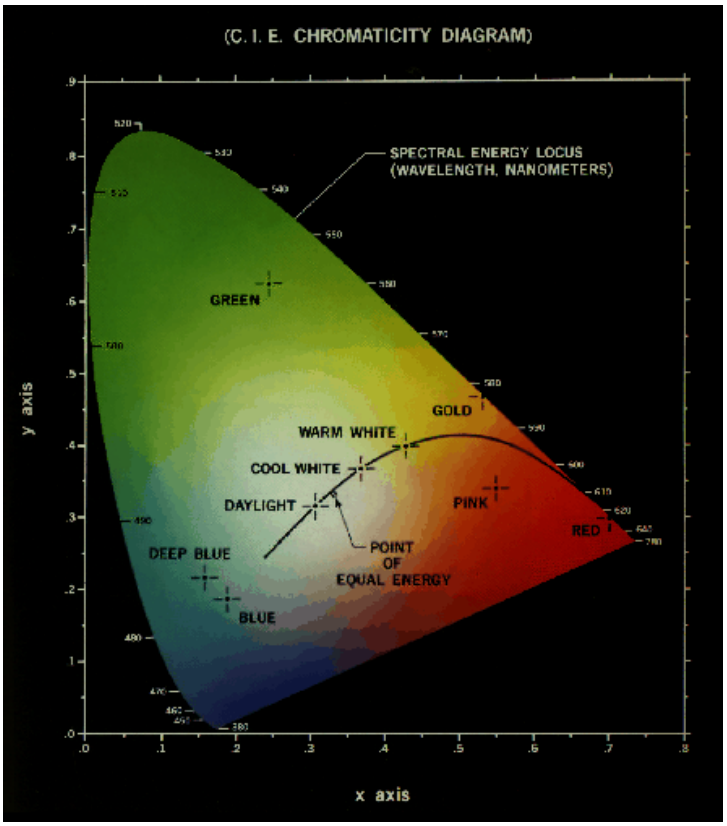
där

$$E^{\lambda_0,\sigma_\lambda} = \int e(\lambda) G(\lambda; \lambda_0, \sigma_\lambda) d\lambda \tag{2.11}$$

mäter den spektrala intensiteten,

$$E_\lambda^{\lambda_0,\sigma_\lambda} = \sigma_\lambda \int e(\lambda) G_\lambda(\lambda; \lambda_0, \sigma_\lambda) d\lambda \tag{2.12}$$

mäter första ordningens spektrala derivata,



Figur 2.5. CIE färgdiagram, bilden är tagen från webbsidan
`http://kiptron.psyc.virginia.edu/steve_boker/ColorVision2/node17.html`

$$E_{\lambda\lambda}^{\lambda_0,\sigma_\lambda} = \sigma_\lambda^2 \int e(\lambda) G_{\lambda\lambda}(\lambda; \lambda_0, \sigma_\lambda) d\lambda \tag{2.13}$$

mäter andra ordningens spektrala derivata, $e(\lambda)$ är energifördelningen i det ljus man mäter och $G(\lambda; \lambda_0, \sigma_\lambda)$

$$G(\lambda; \lambda_0, \sigma_\lambda) = \frac{1}{\sqrt{2\pi\sigma_\lambda}} e^{-\frac{(\lambda-\lambda_0)^2}{2\sigma_\lambda}} \tag{2.14}$$

är en gausskärna placerad i λ_0 på spektral skala σ_λ. G_λ och $G_{\lambda\lambda}$ är gaussderivator med avseende på λ.

Vid bestämning av färgmodellen har man två frihetsgrader, λ_0 och σ_λ. De parametervärden som gör att färgmodellen hamnar så nära det mänskliga synsinnet som möjligt är $\lambda_0 \approx 520\ nm$ och $\sigma_\lambda \approx 55\ nm$ (Geusebroek, van den Boomgaard, Smeulders och Dev 2000). Omvandling från $CIE\ XYZ$ till denna färgmodell

med dessa parametrar sker genom, (Geusebroek, Smeulders och van den Boomgaard 2000):

$$\begin{bmatrix} E \\ E_\lambda \\ E_{\lambda\lambda} \end{bmatrix} = \begin{pmatrix} -0,019 & 0,048 & 0,011 \\ 0,019 & 0 & -0,016 \\ 0,047 & -0,052 & 0 \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.15}$$

En intressant egenskap hos denna modell är att kvoten $E_\lambda / E_{\lambda\lambda}$ ej påverkas av spekulariteter (speglingseffekter) eller intensitetsskillnader (skuggor). Kvoterna E_λ / E och $E_{\lambda\lambda} / E$ påverkas ej av intensitetsskillnader. Detta gör att man genom att detektera särdrag i de olika färgkanalerna och i kvoterna kan se vilka särdrag som beror på skuggor, vilka som beror på spekulariteter osv.

En motivering till varför dessa kvoter har dessa egenskaper kan fås genom att betrakta följande modell för reflekterat ljus från objekt som uppfattas av en kamera.

$$E(\lambda, x) = i(x)\{\rho_f(x) + (1 - \rho_f(x))^2 R_\infty(\lambda, x)\} \tag{2.16}$$

där i är intensitetsfördelningen, ρ_f är Fresnelreflektansen (spekularitetsreflektans), R_∞ objektets spektrala reflektansfunktion (matt reflektans, färg) och E är den spektrala intensitetsfunktion kameran ser. i och ρ_f beror både på scenens geometri och objektens egenskaper medan R_∞ endast beror på objektens egenskaper.

De partiella derivatorna med avseende på λ blir i denna modell:

$$\begin{aligned} E_\lambda &= i(1 - \rho_f)^2 R_\lambda \\ E_{\lambda\lambda} &= i(1 - \rho_f)^2 R_{\lambda\lambda} \end{aligned} \tag{2.17}$$

Man ser då att $E_\lambda / E_{\lambda\lambda}$ ej beror av intensiteten (i) och spekulariteter (ρ_f) samt att E_λ / E och $E_{\lambda\lambda} / E$ ej beror av intensiteten (i).

2.1.3 Belysningsokänsliga färgrymder

Det kan vara användbart att ha en färgmodell där belysningsförhållanden inte påverkar det uppmätta värdet. Detta kan t.ex. vara praktiskt vid objektigenkänning där man använder ett objekts färg för att identifiera det, se avsnitt 2.3.

En del av de modeller som tagits upp tidigare är belysningsokänsliga på olika sätt. T.ex. är H och S komponenten i HSI helt frikopplade från intensitetsinformationen, så de är oberoende av belysningens intensitet. Detsamma gäller I och Q i YIQ.

rgb, vilket är RGB där komponenterna har normaliserats med avseende på intensiteten är också okänsligt för förändringar i intensitet.

$$(r, g, b) = \frac{(R, G, B)}{(R + G + B)} \tag{2.18}$$

Det räcker nu med två komponenter eftersom den tredje kan fås från sambandet $r + g + b = 1$. Denna färgmodell är mindre känslig för variationer i belysningens

intensitet än RGB men inte nödvändigtvis helt okänslig eftersom även kamerans karakteristik kan påverkas vid förändringar i ljusförhållanden.

I Gevers och Smeulders (1999) och Gevers och Smeulders (1996) föreslås en modell $l_1 l_2 l_3$:

$$\begin{aligned} l_1 &= \frac{(R-G)^2}{(R-G)^2+(R-B)^2+(G-B)^2} \\ l_2 &= \frac{(R-B)^2}{(R-G)^2+(R-B)^2+(G-B)^2} \\ l_3 &= \frac{(G-B)^2}{(R-G)^2+(R-B)^2+(G-B)^2} \end{aligned} \tag{2.19}$$

som är oberoende av intensiteten hos belysningen och spekulariteter i bilden. Där föreslås även en modell $m_1 m_2 m_3$:

$$\begin{aligned} m_1 &= \frac{R_{x_1} G_{x_2}}{R_{x_2} G_{x_1}} \\ m_2 &= \frac{R_{x_1} B_{x_2}}{R_{x_2} B_{x_1}} \\ m_3 &= \frac{G_{x_1} B_{x_2}}{G_{x_2} B_{x_1}} \end{aligned} \tag{2.20}$$

där x_1 och x_2 betecknar att värden mätts i två närliggande punkter. $m_1 m_2 m_3$ är oberoende av belysningens färg.

Dessa färgmodeller bygger på följande modell för hur den färg en kamera registrerar förhåller sig till scenen man avbildar. Kamerasvaret för färgkanalen C vid avbildning av en yta är:

$$C = m_b(n,s)\int_\lambda f_C(\lambda)e(\lambda)c_b(\lambda)d\lambda + m_s(n,s,v)\int_\lambda f_C(\lambda)e(\lambda)c_s(\lambda)d\lambda \tag{2.21}$$

där f_C är känslighetsfunktionen (färgfiltret) för kameran, e är belysningen av ytan, c_s är ytans Fresnelreflektans (spekularitetsreflektans) och c_b är dess matta reflektans, λ betecknar ljusvåglängd, n är ytans normal, s är belysningskällans riktning och v är kamerariktningen. m_b och m_s står för beroendet av scengeometrin för matt reflektans respektive speglingsreflektans.

Då man studerar $l_1 l_2 l_3$ modellen antar man en vit belysningskälla samt att c_s är konstant oberoende av våglängd (alla våglängder speglas lika mycket). $e(\lambda)$ och $c_s(\lambda)$ blir då konstanter, e och c_s. Man antar även att

$$\int_\lambda f_R(\lambda)d\lambda = \int_\lambda f_G(\lambda)d\lambda = \int_\lambda f_B(\lambda)d\lambda = f \tag{2.22}$$

vilket innebär att de olika färgkanalerna ger lika starka svar vid vitt ljus. Detta ger:

$$\begin{aligned} &C = em_b(n,s)k_C + em_s(n,s,v)c_s\textstyle\int_\lambda f_C(\lambda)d\lambda, \\ &k_C = \textstyle\int_\lambda f_C(\lambda)c_b(\lambda)d\lambda \end{aligned} \tag{2.23}$$

där k_C enbart beror av kameran och objektets färg. Differenser mellan de olika färgkanalerna kommer nu att vara oberoende av c_s,

$$C_1 - C_2 =$$

$$em_b(n,s)k_{C_1} + em_s(n,s,v)c_s f - (em_b(n,s)k_{C_2} + em_s(n,s,v)c_s f) = \qquad (2.24)$$

$$em_b(n,s)(k_{C_1} - k_{C_2})$$

Om man studerar komponent l_1 får man:

$$l_1 = \frac{(R-G)^2}{(R-G)^2+(R-B)^2+(G-B)^2} =$$

$$\frac{(em_b(n,s)(k_R-k_G))^2}{(em_b(n,s)(k_R-k_G))^2+(em_b(n,s)(k_R-k_B))^2+(em_b(n,s)(k_G-k_B))^2} = \qquad (2.25)$$

$$\frac{(k_R-k_G)^2}{(k_R-k_G)^2+(k_R-k_B)^2+(k_G-k_B)^2}$$

Uttrycket för l_1 beror nu bara på kamerans egenskaper och objektets färg, under de förutsättningar som antagits ovan, bl.a. vit belysning. Motsvarande resultat får man på samma sätt för l_2 och l_3.

Om man vid färgad belysning betraktar kamerans svar för ytans färg har man:

$$C = m_b(n,s)\int_\lambda f_C(\lambda)e(\lambda)c_b(\lambda)d\lambda \qquad (2.26)$$

Om man antar att kameran har smala filterfunktioner som kan approximeras med deltafunktioner så att $f_C(\lambda) = \delta(\lambda - \lambda_C)$ blir kameravärdena:

$$C = m_b(n,s)e(\lambda_C)c_b(\lambda_C) \qquad (2.27)$$

Då man betraktar m_1 har man:

$$m_1 = \frac{R_{x_1}G_{x_2}}{R_{x_2}G_{x_1}} =$$

$$\frac{(m_{b,x_1}(n,s)e_{x_1}(\lambda_R)c_{b,x_1}(\lambda_R))(m_{b,x_2}(n,s)e_{x_2}(\lambda_G)c_{b,x_2}(\lambda_G))}{(m_{b,x_2}(n,s)e_{x_2}(\lambda_R)c_{b,x_2}(\lambda_R))(m_{b,x_1}(n,s)e_{x_1}(\lambda_G)c_{b,x_1}(\lambda_G))} = \qquad (2.28)$$

$$\frac{c_{b,x_1}(\lambda_R)c_{b,x_2}(\lambda_G)}{c_{b,x_2}(\lambda_R)c_{b,x_1}(\lambda_G)}$$

vilket är oberoende av belysningskällans färg och position.

Dessa två modeller används i Gevers och Smeulders (1999) för objektigenkänning baserad på färg (se avsnitt 2.3) och uppvisar där de egenskaper som förväntats (t.ex. okänslighet för variationer i belysningens färg).

2.1.4 NCS

NCS, Natural Colour System, är en färgmodell som helt baseras på människors förnimmelse av färg. I dagligt tal används ordet röd för många olika färger, t.ex. tegelröd, morotsröd och blodröd. Trots att färgerna är olika har de en gemensam egenskap vilken kallas rödhet. Denna egenskap är olika framträdande hos de olika färgerna, de är mer eller mindre röda. Man kan då tänka sig att det måste finnas en färg som bara uppvisar rödhet, en röd elementarfärg, och att de andra färgerna liknar denna mer eller mindre. Det som skiljer de andra färgerna från den röda elementarfärgen är att de också har likheter med andra elementarfärger, en del färger är gulröda, en del är blåröda en del har drag av svarthet eller vithet.

Det finns sex elementarfärger, rött, grönt, gult, blått, vitt och svart. De fyra första är kulörta färger medan de två sista är okulörta, vilket också kallas neutrala. Alla andra färger kan beskrivas utifrån dessa sex färger. Om man betraktar gulröda färger kan man se att en del färger är mer gula än röda och andra färger är mer röda.

Studier där människor fått avgöra hur stora delar rött och gult en viss färg innehåller visar att alla kommer fram till samma resultat för andelarna av elementarfärgerna i olika färger (Sisefsky 1993). Det är detta NCS bygger på.

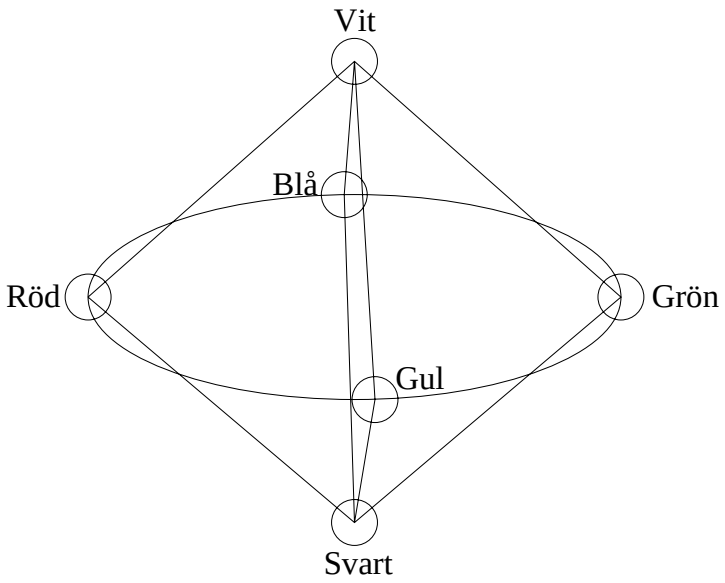
En färg beskrivs genom att ange hur stora delar av de sex elementarfärgerna den består av. En färg kan aldrig bestå av både rött och grönt, inte heller av både gult och blått. Det finns inga färger som samtidigt uppvisar både blåhet och gulhet eller både rödhet och grönhet. Detta gör att en färg kan ha högst två kulörta elementarfärger i sig. I NCS anger man en färgs kulörthet, kulörton och svarthet. Kulörtheten anger hur starka en färgs kulörta egenskaper är tillsammans, kulörtonen anger kulörfärgernas inbördes förhållande och svartheten talar om hur förhållandet mellan svarthet och vithet är i färgen.

NCS kan åskådliggöras med en färgcirkel där de kulörta elementarfärgerna delar cirkeln i fyra lika stora delar i ordningen gult, rött, blått och grönt. Mellan elementarfärgerna finns blandningarna av dessa. Vertikalt genom cirkelns centrum går en linje för gråskalan och man får en symmetrisk dubbelkon, se figur 2.6.

En färg anges t.ex. som 2060-Y60R i NCS. Den första delen, 2060, är svartheten (20) i procent och kulörthet (60) i procent. Den andra delen anger kulörtonen genom förkortningar av elementarfärgerna, här gult (Y för yellow) och rött (R, red) och ett procenttal (60) som syftar på den färg som står efter talet (rött). Denna färg har alltså beståndsdelarna 36% rött (60% rött, 40% gult och en kulörthet på 60% ger $60\% \times 60\% = 36\%$ rött), 24% gult, 20% svart (20% svarthet) och 20% vitt (20% svarthet, 60% kulörthet ger $100\% - 20\% - 60\% = 20\%$ vitt), vilket ger en tegelröd färg.

2.2 Färgkonstans

Ett objekts färg i en bild påverkas av flera faktorer, objektets ytegenskaper, det infallande ljuset och objektets position relativt sensorn (kameran). Ofta är man



Figur 2.6. *NCS* dubbelkon.

intresserad av objektets ytegenskaper (objektets ”riktiga färg”), t.ex. för objektigenkänning baserad på färg (se avsnitt 2.3). Även andra faktorer kan vara av intresse, t.ex. det infallande ljuset, om man kan avgöra vilka delar av bilden som ligger i skugga kan det ge värdefulla ledtrådar för bildens 3D-innehåll.

Begreppet färgkonstans används då man är intresserad av objekts ytegenskaper (färg). Målet med färgkonstans är att för objekt med samma ytegenskaper (samma ”färg”) få samma mätvärden (färg i bilden) oavsett belysningsförhållanden. Detta har visat sig vara ett svårt problem under allmänna förutsättningar. En orsak till detta är att det är ett ett-till-många problem. Många olika kombinationer av belysningskälla och material ger samma sensorsvar. T.ex. ger en röd yta under vitt ljus samma svar som en vit yta under rött ljus och en mörk yta under starkt ljus ger samma svar som en ljus yta under svag belysning. Detta problem kan mildras något om man bestämmer sig för att vissa belysningskällor inte är möjliga i den modell man använder. Man kan t.ex. bestämma att lila belysningskällor är omöjliga, vilket de inte är i verkligheten men i praktiken förekommer lila belysningskällor nästan aldrig.

De flesta metoder för färgkonstans bygger på samma grundtanke. Man beräknar för varje färg som förekommer i bilden vilka kombinationer av belysningskällor och material som kan ge upphov till den aktuella färgen. Därefter bestämmer man sig för att en av de belysningskällor som kan ge upphov till samtliga färger i bilden är den riktiga. För detta steg behövs i allmänhet ytterligare begränsningar och någon sorts urvalsmetod för att bestämma vilken av flera möjliga som är den

”rätta”. När man har separerat ut belysningskällan kan man beräkna objektens ”riktiga färger” från färgerna i bilden.

Ett exempel på en metod för färgkonstans är följande metod, kallad ”Gamut Mapping”. Redogörelsen är hämtad från Finlayson (1996). Denna metod syftar till att uppskatta belysningskällans spektrala egenskaper från en bild.

Man tänker sig en mängd med sensorsvaren från alla möjliga färger under en standardbelysning. En approximation till denna kan skapas genom att ta en bild av n stycken färgade ytor under en standardbelysning c. Man får då mängden C

$$C = \{p_{c,1}, p_{c,2}, \ldots, p_{c,n}\} \tag{2.29}$$

där $p_{c,i}$ är kamerans svar (RGB) för yta i under belysning av c. Då alla färger i C är möjliga färger kommer även alla konvexa kombinationer av dessa vara möjliga färger. Mängden av alla konvexa kombinationer av C kallas $\Gamma(C)$. Det räcker med det konvexa höljet av $\Gamma(C)$ för att representera $\Gamma(C)$, eftersom resten av $\Gamma(C)$ kan uttryckas som konvexa kombinationer av det konvexa höljet.

Då man söker egenskaperna hos belysningskällan i en bild skapar man först $\Gamma(I)$, genom att beräkna det konvexa höljet till de kamerasvar som förekommer i bilden. De avbildningar som avbildar $\Gamma(I)$ så att resultatet ligger helt i $\Gamma(C)$ är möjliga lösningar till färgkonstansproblemet.

En punkt $p_{i,u}$ i $\Gamma(I)$ kan avbildas på punkt $p_{c,v}$ i $\Gamma(C)$ genom multiplikation med en diagonalmatris $D_{v,u}$. Låt den mängd matriser som avbildar $p_{i,u}$ på någon punkt i $\Gamma(C)$ kallas D_u. De diagonalmatriser som avbildar samtliga punkter i det komplexa höljet till $\Gamma(I)$ på punkter i $\Gamma(C)$ är möjliga lösningar. Dessa matriser är snittet av alla D_u.

Snittet är en mängd möjliga diagonalmatriser. För att välja ut en av dessa för att få en uppskattning av belysningskällans egenskaper kan man ta den diagonalmatris som maximerar volymen av avbildningen av $\Gamma(I)$.

Diagonalmatrisen talar om hur belysningskällan i bilden förhåller sig till standardbelysningskällan c. Då en färg (punkt) $p_{i,u}$ i $\Gamma(I)$ motsvarar $p_{c,v}$ under standardbelysning och $p_{c,v} = D p_{i,u}$ kommer belysningskällan i i bilden att förhålla sig till c enligt $c = Di$, om vi tänker oss att belysningskällans färg mäts på samma sätt som ytornas färger.

Denna metod fungerar ganska bra under tämligen strikta krav, att alla ytor ska vara plana, alla ytor ska vara matta och belysningen ska vara lika överallt i bilden. Dessutom behövs ganska många olika ytor i bilden för bra resultat. Det finns flera modifikationer av denna metod för att få bättre resultat eller lätta på kraven, bl.a. i Barnard (2000) och Finlayson (1996).

Fler exempel på metoder för färgkonstans finns i Angelopoulou (2000), Barnard, Martin och Funt (2000), Brainard och Freeman (1997) och Finlayson och Schaefer (2000).

2.3 Igenkänning

De första metoderna för objektigenkänning byggde på att från en bild försöka rekonstruera ett objekts 3D-form. Detta gjordes främst genom detektion av särdrag som bevaras i 2D-representationen av ett objekt, t.ex. kanter och hörn. Från dessa försökte man härleda vad för objekt som gett upphov till särdragen. Tanken att först ta fram ett objekts form och sedan identifiera objektet är rimlig eftersom det ju vanligtvis är ett objekts form som bestämmer vad objektet är för något. Det är ju t.ex. en stols form som gör att det är en stol, medan saker som färg och material kan variera.

Det är dock svårt att identifiera objekt genom att försöka återskapa deras 3D-form. Dels kan man få problem med detektionen av särdragen som metoden bygger på (t.ex. brusiga eller oskarpa bilder kan ställa till problem, se avsnitt 2.4), dels kan många 3D-former ge upphov till samma 2D-bild under olika omständigheter och dels kan en 3D-form ge upphov till många olika 2D-representationer från olika vinklar.

Då man från en bild av ett objekt ska identifiera objektet kan man använda sig av objektets färg som ledtråd. Färg är inte så lämpligt för att känna igen vissa saker, t.ex. är det svårt att genom att betrakta endast en muggs färg inse att det är en mugg, eftersom muggar finns i många olika färger och andra saker än muggar också har dessa färger. Ska man däremot känna igen flaggor kan färg vara en bra ledtråd. Färg är också användbart för att känna igen ett visst objekt, t.ex. en bestämd mugg. Även om inte alla muggar är vita och gröna så kanske just den aktuella muggen är det.

Då man matchar objekt på deras färg blir matchningen i viss mån rotations-okänslig. Rotation i bildplanet påverkar inte ett objekts färg, däremot påverkar andra rotationer färgen om objektet inte har samma färger på alla sidor.

Matchning på färg är mycket känslig för variationer i belysningsförhållanden, t.ex. belysningskällans färg, intensitet och position i förhållande till objektet. Om belysningsförhållandena går att kontrollera, t.ex. vid igenkänning av objekt på ett löpande band i en fabrik, kan matchning på färg ge mycket bra resultat. Om däremot belysningsförhållandena kan variera mycket och utan att man vet hur krävs extra åtgärder (som färgkonstans, se avsnitt 2.2) för att metoden ska vara användbar.

En fördel med att matcha på färg är att metoderna för detta oftast är väldigt snabba, så man kan få realtidsprestanda, vilket inte alltid är möjligt med andra metoder.

2.3.1 Histogrammatchning

En klassisk metod för matchning på färg presenterades i Swain och Ballard (1991). Den användes där dels till att lokalisera ett (känt) objekt i bilden och dels till att identifiera okända objekt. Metoden bygger på att man har en databas med bilder på de objekt man ska identifiera. För att identifiera ett objekt tar man ett

histogram över färgerna i bilden av objektet och ett histogram över färgerna i en bild från objektdatabasen och ser hur lika histogrammen är. Objektet identifieras som det objekt i databasen vars färghistogram är mest likt det okända objektets färghistogram.

För att se hur lika två histogram är används följande metod:

$$\frac{\sum_{i,j,k} min(H_{i,j,k}, I_{i,j,k})}{\sum_{i,j,k} H_{i,j,k}} \tag{2.30}$$

där I är färghistogrammet från den okända bilden, H färghistogrammet från databasbilden och i, j, k färgkanalerna. Man beräknar alltså hur stor del av pixlarna i bilderna som har samma färg.

Författarna rapporterar att man kan få bra objektigenkänning med relativt grov uppdelning av färgerna i histogrammen (totalt ca. 200 färger räcker bra), vilket gör metoden mycket snabb. Metoden fungerar även bra vid måttlig skymning av objektet. Metoden tar ingen som helst hänsyn till färgernas position så den kan inte skilja prickiga saker från randiga om ränderna och prickarna har lika stor yta och samma färg.

2.3.2 Färgkvotshistogram

I Funt och Finlayson (1995) presenteras en variant på histogrammatchning. Istället för histogram över pixlarnas färger använder man histogram över derivatan av logaritmen av pixlarnas värden. Detta gör metoden mindre belysningskänslig.

Motiveringen till detta förfarande är att man för en position x med infallande belysning e^x och reflektans r^x kan se sensorsvaret ρ_k^x för sensor k som:

$$\rho_k^x = e_k^x r_k^x \tag{2.31}$$

vilket inte alltid är sant, men det är en i detta fall användbar modell. I denna modell ger kvoten av sensorsvaren mellan två positioner kvoten mellan deras ytreflektanser, om belysningen är densamma vid båda positionerna.

$$\frac{\rho_k^1}{\rho_k^2} = \frac{e_k^1 r_k^1}{e_k^2 r_k^2} = \frac{r_k^1}{r_k^2}, e_k^1 = e_k^2 \tag{2.32}$$

Kravet på belysningen e_k att den är lika på de båda platserna är rimligt för närliggande punkter. För att bli av med divisionen kan man logaritmera båda leden, vilket ger:

$$ln(\rho_k^1) - ln(\rho_k^2) = ln(r_k^1) - ln(r_k^2) \tag{2.33}$$

Vänsterledet motsvarar derivatan av logaritmen av bilden och högerledet är en belysningsoberoende relation mellan färgen i två punkter. De värden man nu mäter motsvarar kvoten mellan färgerna längs färgkanter.

Författarna rapporterar att denna metod fungerar mycket bra vid variationer i belysningen.

2.3.3 Igenkänning baserad på lokala delområden

Det finns ett flertal metoder för objektigenkänning som baseras på lokal information i små delområden av en bild (Matas, Koubaroulis och Kittler 2000, Slater och Healey 1996, Hall, de Verdière och Crowley 2000). Den lokala informationen kan bestå av färginformation, derivator m.m. En fördel med lokal information är att man får ett visst mått av robusthet mot skymning av delar av objektet.

2.3.4 Färgmodellers inverkan på igenkänning

Vid t.ex. histogrammatchning kan de färgmodeller man använder spela stor roll. Använder man t.ex. rgb blir igenkänningen mindre belysningskänslig. Vanligtvis är det så att en färgmodell som är mindre belysningskänslig är sämre på att särskilja objekt som är olika. Den färgmodell man använder blir en kompromiss mellan särskiljningsförmåga och belysningsinvarians. Vad man prioriterar mest beror på tillämpningen, man tar lämpligen den färgmodell som ger bäst särskiljningsförmåga men fortfarande fungerar under de ljusförhållanden man är intresserad av.

2.4 Särdragsdetektion

2.4.1 Traditionell särdragsdetektion

Särdragsdetektion i bilder innebär att man letar efter strukturer av en viss karaktär i bilderna. Detta gör man för att dessa strukturer förhoppningsvis ger intressant information, t.ex. kan man tänka sig att kanter är viktiga delar i bilder, eftersom en streckteckning kan vara en bra kompakt ”sammanfattning” av en bild och strecken motsvarar kanterna i bilden. De vanligaste särdragen som används är kanter, hörn, regioner och åsar.

Vanligtvis sker särdragsdetektion på gråniv̊abilder. Det finns flera skäl till detta, bl.a. har hårdvara för att läsa in färgbilder, färgscanner, färgkamera m.m., varit dyrt och inte speciellt vanligt. Eftersom färgbilder innehåller mer information än gråniv̊abilder tar de också upp mer lagringsutrymme och det tar längre tid att arbeta med dem. Detta har lett till att man traditionellt bara arbetat med gråniv̊abilder, men på senare tid har man även börjat göra särdragsdetektion på färgbilder.

Notation

I de följande delavsnitten används följande notation:

- L, bildmatris med gråniv̊ainformation.
- L_x, indexbokstaven betecknar partiell derivering av bilden med avseende på indexvariabeln. L_x är derivatan i x-led och L_{yy} är $\frac{\partial^2 L}{\partial y^2}$.
- x, y, bildens koordinatsystem, där x är vågräta dimensionen av bilden och y lodräta.

- u, v, ett lokalt koordinatsystem i en punkt, där v är gradientens riktning i punkten och u är vinkelrät mot v. Basvektorerna e_v och e_u i detta system blir då $e_v = (\cos(\varphi), \sin(\varphi))$ respektive $e_u = (\sin(\varphi), -\cos(\varphi))$ där $\cos(\varphi) = L_x/\sqrt{L_x^2 + L_y^2}$ och $\sin(\varphi) = L_y/\sqrt{L_x^2 + L_y^2}$. Partiella derivator i u och v riktningarna blir $\partial_v = \cos(\varphi)\partial_x + \sin(\varphi)\partial_y$ och $\partial_u = \sin(\varphi)\partial_x - \cos(\varphi)\partial_y$.

Kanter

En kant är gränsen mellan två områden med olika intensitet i bilden. Kanter i en bild motsvarar t.ex. ytnormalskillnad, djupskillnad, färgskillnad eller belysningsskillnad (skuggor) i scenen bilden avbildar.

En kant i bilden innebär att gradientens magnitud är hög i de punkter som ligger på en kant. Ett enkelt sätt att detektera kanter är att tröskla gradientmagnituden med ett lämpligt tröskelvärde. Det kan dock vara svårt att välja ett lämplig tröskelvärde. Man kan också betrakta kanter som de punkter där gradientmagnituden har lokala maxima i gradientriktningen. Det innebär att andraderivatan i gradientriktningen ska vara 0 och att tredjederivatan skall vara negativ. Vi får då:

$$\begin{cases} L_{vv} = 0 \\ L_{vvv} < 0 \end{cases} \tag{2.34}$$

Det finns även andra sätt att detektera kanter.

Hörn

Hörn är intressanta strukturer i bilder. T.ex. har många kantdetektorer problem vid hörn och kan ge konstiga resultat i närheten av hörnpunkter. Hörn ger också intressant information om scenen bilden föreställer, de kan representera skymningspunkter m.m.

Ett sätt att karaktärisera hörn är att betrakta hörn som punkter där beloppet av gradientmagnituden, $|\nabla L|$, är högt och krökningen, κ, hos nivåkurvorna till intensitetsfunktionen också är stor. Då båda dessa ska vara höga kan man ta maxima hos $\kappa|\nabla L|^n$ som hörn, där n är en parameter för att bestämma den relativa vikten mellan κ och $|\nabla L|$.

κ kan uttryckas i partiella derivator av intensitetsfunktionen som:

$$\kappa = \frac{L_y^2 L_{xx} - 2L_x L_y L_{xy} + L_x^2 L_{yy}}{(L_x^2 + L_y^2)^{3/2}} \tag{2.35}$$

Då $|\nabla L| = (L_x^2 + L_y^2)^{1/2}$ är $n = 3$ det lägsta värdet på n för vilket man undviker divisionen, vilket är fördelaktigt eftersom gradientmagnituden (nämnaren) kan vara 0. Om vi väljer $n = 3$ får vi hörn som lokala maxima hos:

$$L_y^2 L_{xx} - 2L_x L_y L_{xy} + L_x^2 L_{yy} \tag{2.36}$$

Regioner

Regioner är områden i bilden som har samma färg. Dessa områden kallas även blobbar. Laplaceoperatorns, ∇^2, lokala extremvärden infaller i centrum av regioner som är ljusare eller mörkare än omgivningen och har ungefär lika stor utsträckning i alla riktningar. Detta gör att den är en ofta använd blobbdetektor. Tecknet på extremvärdet talar om huruvida det är en ljus, < 0, eller en mörk, > 0, region. Om man vill detektera alla regioner, både ljusa och mörka, kan man kvadrera Laplacesvaret och detektera lokala maxima.

Detektion av ljusa (minima) och mörka (maxima) regioner:

$$\nabla^2 L = L_{xx} + L_{yy} \tag{2.37}$$

Detektion av både ljusa och mörka regioner (maxima):

$$(\nabla^2 L)^2 = (L_{xx} + L_{yy})^2 \tag{2.38}$$

Åsar

En ås är en långsträckt region.

Låt (p, q) vara ett lokalt koordinatsystem i en punkt där $L_{pq} = 0$. p- och q-riktningarna blir då principalkrökningsriktningar och andraderivatorna i dessa riktningar principalkrökningar. Detta system är inte unikt bestämt utan kan roteras godtycklig multipel av 90 grader. Välj en av dessa multipler och kalla den riktning som har störst absolutbelopp av principalkrökningarna den dominerande principalkrökningsriktningen.

En ås kan nu ses som en mängd sammanhängande punkter där intensiteten antar ett maximum eller minimum (för en ljus respektive mörk ås) i den dominerande principalkrökningsriktningen. För ljusa åsar får man:

$$\begin{cases} L_p = 0 \\ L_{pp} < 0 \\ |L_{pp}| \geq |L_{qq}| \end{cases} \quad eller \begin{cases} L_q = 0 \\ L_{qq} < 0 \\ |L_{qq}| \geq |L_{pp}| \end{cases} \tag{2.39}$$

och motsvarande för mörka åsar:

$$\begin{cases} L_p = 0 \\ L_{pp} > 0 \\ |L_{pp}| \geq |L_{qq}| \end{cases} \quad eller \begin{cases} L_q = 0 \\ L_{qq} > 0 \\ |L_{qq}| \geq |L_{pp}| \end{cases} \tag{2.40}$$

där den nedre raden ser till att den riktning extremvärdet antas i ska vara den dominerande principalkrökningsriktningen. Under förutsättning att gradientmagnituden inte är 0 kan detta skrivas i det tidigare använda (u, v) systemet som:

$$\begin{cases} L_{uv} = 0 \\ L_{uu}^2 - L_{vv}^2 > 0 \end{cases} \tag{2.41}$$

där tecknet på L_{uu} avgör vilken typ av ås det är:

$$\begin{cases} L_{uu} < 0 \rightarrow ljus \\ L_{uu} > 0 \rightarrow mörk \end{cases} \tag{2.42}$$

2.4.2 Särdragsdetektion i färgbilder

Ett problem med särdragsdetektion i färgbilder är att en bildpunkt inte har ett skalärt värde som i en gråvåbild utan i stället vanligtvis en trevärd storhet (t.ex. (R, G, B)). Detta gör att de tidigare diskuterade särdragsdetektorerna inte kan användas utan någon sorts modifiering.

Ett enkelt sätt att komma förbi detta problem är att ta var och en av färgkanalerna och göra särdragsdetektion i dem var för sig precis som i gråvåfallet och sedan kombinera resultaten. Kombinera resultaten kan man göra på olika sätt, t.ex. kan man ta alla särdrag som detekterats i någon kanal som särdrag eller så kan man ta bara dem som detekterats i alla kanaler. Hur man kombinerar resultaten beror på tillämpningen.

Om man väljer sin färgmodell väl kan man få intressant information genom särdragsdetektion i de olika färgkanalerna var för sig. Olika kanaler ger då information om olika typer av särdrag. T.ex. kan man i en kanal detektera skuggkanter och i en annan kanter mellan olika material (Geusebroek, van den Boomgaard, Smeulders och Dev 2000).

Ett sätt att få ett värde för gradientmagnituden i färgbilder, för att sedan t.ex. göra kantdetektion, är följande. Låt R, G, och B vara färgkanalerna, vilka kan vara RGB-kanaler men det är inget krav, x och y är bildens vågräta respektive lodräta dimension. Motsvarigheten till gradientmagnitud och gradientriktning i en gråvåbild blir då roten ur det största egenvärdet respektive den tillhörande egenvektorn till matrisen $D^T D$ (Saber, Tekalp och Bozdagi 1997), där

$$D = \begin{pmatrix} (\frac{\partial R}{\partial x}) & (\frac{\partial R}{\partial y}) \\ (\frac{\partial G}{\partial x}) & (\frac{\partial G}{\partial y}) \\ (\frac{\partial B}{\partial x}) & (\frac{\partial B}{\partial y}) \end{pmatrix} \tag{2.43}$$

$$D^T D = \begin{pmatrix} (\frac{\partial R}{\partial x})^2 + (\frac{\partial G}{\partial x})^2 + (\frac{\partial B}{\partial x})^2 & \frac{\partial R}{\partial x}\frac{\partial R}{\partial y} + \frac{\partial G}{\partial x}\frac{\partial G}{\partial y} + \frac{\partial B}{\partial x}\frac{\partial B}{\partial y} \\ \frac{\partial R}{\partial x}\frac{\partial R}{\partial y} + \frac{\partial G}{\partial x}\frac{\partial G}{\partial y} + \frac{\partial B}{\partial x}\frac{\partial B}{\partial y} & (\frac{\partial R}{\partial y})^2 + (\frac{\partial G}{\partial y})^2 + (\frac{\partial B}{\partial y})^2 \end{pmatrix} \tag{2.44}$$

Det finns flera närliggande metoder som använts för att konstruera motsvarigheten till gradientmagnitud och gradientriktning. I Sapiro och Ringach (1996) definieras gradientriktningen på samma sätt som ovan, dvs riktningen hos den egenvektor som tillhör det största egenvärdet. Däremot definieras gradientmagnituden annorlunda. Att bara titta på det största egenvärdet är inte tillräckligt, eftersom även det andra egenvärdet kan variera i storlek. I Sapiro och Ringach (1996) rekommenderas istället differensen mellan största och minsta egenvärdet som värde för gradientmagnituden.

När man väl har definierat en gradientriktning och ett värde för gradientmagnituden kan man göra kantdetektion på det sätt som nämnts tidigare, dvs definiera kanter som de punkter där gradientmagnituden har lokala maxima i gradientriktningen.

Ett klassiskt sätt att använda särdragsdetektion i färgbilder är att kombinera segmentering (uppdelning av bilden i regioner med, i detta fall, samma färg) och kantdetektion. Om man först grovsegmenterar bilden kan man förfina segmenteringen av de områden där det fortfarande finns starka kantsvar inuti en region. Vid förfiningen kan man använda kantsvarens positioner som ledtrådar om lämpliga ställen att dela regioner.

Andra exempel på hur färginformation har används vid särdragsdetektion finns i Androutsos, Androutsos, Plataniotis och Venetsanopoulos (1997), Cumani (1989), Cumani (1991), Geusebroek, Dev, van den Boomgaard, Smeulders, Cornelissen och Geerts (1999), Geusebroek, van den Boomgaard, Smeulders och Dev (2000), Geusebroek, Smeulders och van den Boomgaard (2000), Krishnamoorthi och Bhattacharyya (1997), Pujas och Aldon (1997), Saber, Tekalp och Bozdagi (1997), Stokman och Gevers (1999), Sapiro och Ringach (1996) och Würtz och Lourens (2000).

2.5 Skalval

Vid bildanalys förekommer begreppet skala. Bildstrukturer är bara relevanta på vissa skalor, om man t.ex. tittar på en satellitbild av ett skogsområde är det inte meningsfullt att tala om enskilda blad på träden, vilket det däremot är om man tittar på ett fotografi av ett träd i närbild.

I en bild finns vanligtvis både finskalestrukturer (små detaljer) och grovskalestrukturer (större objekt). Vid särdragsdetektion detekteras oftast bildstrukturer på en viss skala mycket starkare än strukturer på andra skalor. Vanligtvis detekteras finskalestrukturer starkast. För att kunna detektera särdrag på olika skalor med samma särdragsdetektor kan man skapa någon form av bildrepresentation som tar hänsyn till skala. Beskrivning av tre metoder för detta, pyramidrepresentation, skalrumsrepresentation och hybridpyramidrepresentation, följer nedan.

2.5.1 Pyramidrepresentation av bilder

En pyramidrepresentation av en bild är en mängd bilder där bilden på nivå $n + 1$ är en subsamplad version av bilden på nivå n, och den bild man vill representera är nivå 0. Lägger man bilderna ovanför varandra i nummerordning får man en struktur som påminner om en pyramid, därav namnet.

Pyramidrepresentationer av bilder är enkla att skapa. De kräver också ganska lite minnesutrymme, eftersom storleken på nivåerna minskar snabbt. Att nivåernas storlek minskar snabbt gör att det går fort att arbeta med pyramider, t.ex. göra särdragsdetektion i dem, eftersom datamängden reduceras när nivåerna förminskas.

Då en högre nivå är en subsamplad version av originalbilden försvinner små detaljer i bilden och stora strukturer blir mindre. Har man då en särdragsdetektor som detekterar små särdrag i bilder och använder den på alla nivåer i pyramiden kommer man förhoppningsvis detektera särdrag av alla storlekar. Dock brukar särdragsdetektion i pyramider inte ge så bra resultat, vilket beror på att alltför mycket information går förlorad i subsamplingsstegen.

En bra beskrivning av pyramidrepresentationer ges i Burt och Adelson (1983).

2.5.2 Skalrumsrepresentation av bilder

En skalrumsrepresentation av en bild är en samling bilder som representerar originalbilden på olika skalor. Ett sätt att skapa en skalrumsrepresentation är genom att falta originalbilden med gausskärnor med olika varians (Lindeberg 1998b). Skalrumsrepresentationen L, med skalparametern t, av f blir då:

$$\begin{aligned} L(;t) &= g(\cdot;t) * f(\cdot) \\ g(x;t) &= \frac{1}{(2\pi t)^{\frac{N}{2}}} e^{-\frac{(x_1^2+\ldots+x_N^2)}{2t}} \end{aligned} \tag{2.45}$$

Egenskaper som är lämpliga hos en skalrumsrepresentation är linearitet, translationsinvarians och att den inte förstärker eller skapar nya strukturer då man går från en fin skala till en grövre. Det visar sig att dessa egenskaper unikt ger gausskärnor som skalrumsrepresentation (Lindeberg 1998a).

Skalrumsrepresentation av bilder ger bra resultat vid särdragsdetektion och de har analyserats teoretiskt ur skalnormeringssynpunkt (se avsnitt 2.5.4), vilket visat sig vara svårt att göra med pyramidrepresentationer. Däremot kräver skalrumsrepresentationer mycket minnesutrymme och det går åt mycket datorkraft, både till att skapa skalrumsrepresentationen och till att göra särdragsdetektion i den.

Skalrumsrepresentation kan också användas till annat än särdragsdetektion, ett exempel på användning av skalrumsrepresentationer för färgsegmentering finns i Lindeberg (1994).

2.5.3 Hybridpyramidrepresentation av bilder

Både pyramidrepresentationer och skalrumsrepresentationer har önskvärda egenskaper men lider också av vissa nackdelar. De områden där pyramidrepresentationer har sin styrka, låga resurskrav och möjlighet till snabb särdragsdetektion är de områden där skalrumsrepresentationer har sin svaghet. De områden där skalrumsrepresentationer är bra, teoretiskt välgrundad skalnormering och hög kvalitet på detekterade särdrag, är de områden där pyramidrepresentationer är dåliga. Ett försök att behålla de båda metodernas fördelar och samtidigt bli av med nackdelarna är hybridpyramidrepresentation, vilket är en blandning av pyramidrepresentation och skalrumsrepresentation.

I en hybridpyramid skapas en nivå ibland genom subsampling av föregående nivå och i andra fall genom faltning av föregående nivå med en filterfunktion som

liknar en gausskärna. Hur ofta subsampling sker i förhållande till hur ofta faltning sker ger en avvägning mellan hur nära en ren pyramid och ett rent skalrum man vill ligga. Om man subsamplar ofta ger det snabbare särdragsdetektion på bekostnad av kvaliteten på detektionen.

Vid faltning beräknas den nya nivån L' som:

$$L'(x, y) = \sum_{n=-N}^{N} \sum_{m=-M}^{M} c(n, m) L(x - n, y - m) \tag{2.46}$$

där L är föregående nivå, c är filterfunktionen man faltar med, $2N + 1$ och $2M + 1$ är dimensionerna på filtret (vanligtvis gäller $N = M$).

Vid subsampling beräknas den nya nivån L' som:

$$L'(x, y) = \sum_{n=-N}^{N} \sum_{m=-M}^{M} c(n, m) L(2x - n, 2y - m) \tag{2.47}$$

där L är föregående nivå, c är ett utjämningsfilter, $2N + 1$ och $2M + 1$ är dimensionerna på filtret (vanligtvis gäller $N = M$). Vid subsampling behöver inte nödvändigtvis utjämning ske. Ett rent subsamplingssteg, utan utjämning, har $N = M = 0$ och $c(0, 0) = 1$.

När man konstruerar en hybridpyramid gör man vanligtvis inte så att subsampling sker vid godtyckligt valda nivåer, utan man gör subsampling med regelbundna mellanrum. Genereringen av hybridpyramiden sker alltså i cykler där ett subsamplingssteg och ett antal utjämningssteg, då man faltar en nivå med en filterfunktion, ingår:

$$Hybridpyramidcykel = \begin{matrix} Subsampling \\ Faltning+ \end{matrix} \tag{2.48}$$

$Faltning+$ betyder att flera utjämningssteg kan förekomma. En intressant egenskap hos hybridpyramiden är hur många utjämningssteg som sker i varje cykel. Detta kan anges genom att helt enkelt ange antalet faltningar i varje cykel, men man brukar ange det med en parameter ρ, subsamplingstakten.

Om varje cykel innehåller J utjämningssteg som vart och ett ger en skaländring Δt_j får man en total skaländring för en cykel på:

$$\Delta t_{cykel} = \sum_{j=1}^{J} \Delta t_j \tag{2.49}$$

där Δt_j beror av den filterfunktion man faltar med i utjämningsstegen. Då får vi ρ som:

$$\rho = \frac{2}{\sqrt{\Delta t_{cykel}}} \tag{2.50}$$

För utförligare beskrivningar av hybridpyramidrepresentation av bilder rekommenderas Grostabussiat (1997), Lindeberg (1995) och Niemenmaa (2000).

2.5.4 Automatiskt skalval i skalrumsrepresentationer

Vid särdragsdetektion är det finskalestrukturer som ger starkast svar då skalparametern t är liten och grovskalestrukturer som ger starkast svar då t är stor. Att välja en lämplig skala/skalor för särdragsdetektionen kan ibland göras i förväg, men så är inte alltid fallet. T.ex. kan de strukturer man söker variera mycket i storlek från bild till bild och även inom bilden kan olika skalor vara lämpliga för olika strukturer och även för samma struktur, t.ex. längs en kant som är olika skarp på olika ställen. Då kan det vara lämpligt att ha en metod för automatiskt skalval.

De särdragsdetektorer som tagits upp tidigare består alla av olika typer av derivator. Om man betraktar t.ex. ∂_x gör skalrumsrepresentationens egenskap att inga strukturer förstärks då man går till en grövre skala att ∂_x ger svagare svar vid grövre skalor än vid fina skalor, dvs ∂_x avtar hela tiden med t. Om man betraktar

$$t^{\frac{\gamma}{2}}\partial_x \tag{2.51}$$

där γ är en normaliseringsparameter, får man en funktion som först växer och sedan avtar med t. Man kan nu definiera automatisk skalval som detektion av de skalor där denna γ-normaliserade derivata har lokala maxima.

Att detta är en vettig definition av automatiskt skalval kan man se t.ex. genom att betrakta funktionen

$$f(x) = \sin(\omega_0 x) \tag{2.52}$$

och dess skalrumsrepresentation

$$L(x;t) = e^{-\frac{\omega_0^2 t}{2}} \sin(\omega_0 x) \tag{2.53}$$

Amplituden (motsvarande operatorsvaret hos en särdragsdetektor) hos n:te ordningens γ-normaliserade derivata blir

$$t^{\frac{n\gamma}{2}} \omega_0^n e^{-\frac{\omega_0^2 t}{2}} \tag{2.54}$$

vars maximum över t blir

$$t = \frac{\gamma n}{\omega_0^2} \tag{2.55}$$

Om man betraktar $\sqrt{t}$ som skalparameter ser man att skalan med maximal amplitud är proportionell mot våglängden $\lambda = \frac{2\pi}{\omega_0}$, dvs man kan detektera signalens våglängd genom att detektera maximum över skalor för den γ-normaliserade derivatan. Skalparametern talar alltså om vilken storlek ett detekterat objekt har, vilket är precis det man vill att automatisk skaldetektion ska tala om.

En annan trevlig egenskap hos de γ-normaliserade derivatorna är att de beter sig trevligt då man skalar om bilder. Om man har ett skalrumsmaximum för $f(x)$ vid $x = x_0, t = t_0$ kommer man för f' sådan att

$$f(x) = f'(sx) \tag{2.56}$$

ha ett skalrumsmaximum för

$$(x'_0, t'_0) = (sx_0, s^2 t_0) \tag{2.57}$$

Det visar sig att om man vill ha detektion av maxima över någon form av normaliserade derivator och vill ha det nämnda beteendet vid omskalningar av bilden täcker γ-normaliserade derivator alla möjliga normaliseringar (Lindeberg 1999).

2.6 Detektion av hud

Att detektera hud i bilder är ett område med flera möjliga tillämpningar. Några exempel där detektion av hud är användbar är vid ansiktsigenkänning (för att skilja ansiktet från bakgrunden), människa-maskininteraktion (gränssnitt där man pekar med handen, gestigenkänning m.m.), bild- och videokompression (man kan komprimera ointressant bakgrund mycket och behålla detaljrikedom i de ”viktiga” delarna, t.ex. ansikten) och innehållsbaserad sökning i bilddatabaser och videosekvenser (människor är vanliga motiv).

En metod för detektion av hud är att använda histogrammatchning (avsnitt 2.3.1). Man delar då in bilden i många små delområden, t.ex. 5x5 pixlar, och jämför deras färghistogram med ett kontrollhistogram med ”hudfärg”. De områden vars histogram ligger tillräckligt nära kontrollhistogrammet betraktas som hud.

Denna metod kan sedan förfinas på många olika sätt, t.ex. kan man välja ett område som ligger precis vid toleransgränsen för ”tillräckligt lika” och använda dess histogram som kontrollhistogram i ett ytterligare steg. Detta gör att man kan ha hårdare krav i första steget, för att minimera falska svar, och ändå få med de hudfärgade områden som skiljer sig en del från kontrollområdet.

En annan liknande metod är att man lättar på kraven för områden som har många grannområden som är klassade som hud och tar bort områden klassade som hud som inte har några grannar klassade som hud. Det kan ge bättre resultat eftersom man oftast är intresserad av större sammanhängande områden med hud.

Att segmentera ut hud enbart genom dess färg kan fungera ganska bra om man kan kontrollera bakgrunden och belysningen. Det visar sig att olika hudfärger ligger ganska samlade i färgrymden om man väljer en lämplig färgmodell (Terrillon, David och Akamatsu 1998). Många andra material har dock liknande färgegenskaper som hud. Exempel på sådana material är sand, vissa typer av djurpäls, olika typer av trä m.m. (Forsyth och Fleck 1996), vilket kan ge många falska träffar t.ex. vid sökning i bilddatabaser. Ett annat problem med att bara använda färg är att färgen i bilden på hud kan variera kraftigt om belysningens färg varierar, vilket kan vara ett problem med t.ex. indirekt belysning från kläder (Saxe och Foulds 1996).

Om man söker en viss typ av områden med hud, t.ex. ansikten, kan man få mycket bättre resultat om man utöver färgsegmenteringen också har en modell av

hur området ser ut. Ska man leta efter ansikten kan man först detektera områden med hud med ganska stor tolerans, vilket ger många falska träffar men förhoppningsvis får med alla riktiga områden med hud. Sedan tar man de områden som är potentiella ansikten och ser om de uppfyller ytterligare krav, som t.ex. att de är ungefär ovala med ”hål” för ögon och mun.

Exempel på människa-maskininteraktion där detektion av hudfärgade områden är nödvändig finns i Ahmad (1995), Kjeldsen och Kender (1996) och Bretzner och Lindeberg (2000).

Kapitel 3

Metoder för användande av färginformation vid särdragsdetektion

3.1 Särdragsdetektorer som använder färginformation

3.1.1 En regiondetektor för färgbilder

I en intensitetsbild har man i varje bildpunkt ett skalärt värde, vilket de klassiska särdragsdetektorerna är utvecklade för. I en färgbild har man ett trevärt (vektor-) värde i varje punkt. Det gör att man inte kan använda de klassiska särdragsdetektorerna rakt av.

Blobbdetektorn för gråvåbilder som beskrivs i avsnitt 2.4.1,

$$\nabla^2 = L_{xx} + L_{yy} \tag{3.1}$$

är spåret i Hessianmatrisen:

$$\begin{pmatrix} L_{xx} & L_{xy} \\ L_{xy} & L_{yy} \end{pmatrix} \tag{3.2}$$

Detta gör att den har vissa invariansegenskaper, t.ex. påverkas den inte av rotationer av bilden.

Motsvarigheten till Hessianmatrisen blir för färgbilder en tredimensionell tensor där man har tre matriser som den i ekvation 3.2 i lager på varandra. Om man gör tensorkontraktion, vilket är motsvarigheten till spåret, på denna tensor får man en vektor

$$\begin{bmatrix} A_{xx} + A_{yy} \\ B_{xx} + B_{yy} \\ C_{xx} + C_{yy} \end{bmatrix} \tag{3.3}$$

där A, B och C är de tre färgkanalerna (t.ex. RGB).

I gråvåfallet tar man ofta beloppet eller kvadraten av operatorn i ekvation 3.1 för att få med både ljusa och mörka regioner. Beloppet av en vektor är normen av

vektorn och kvadraten av en vektor är normen i kvadrat. Då man i detta fall söker lokala maxima påverkas detta inte av om man har normen i kvadrat eller om man drar roten ur detta. Normen i kvadrat av vektorn i ekvation 3.3:

$$(A_{xx} + A_{yy})^2 + (B_{xx} + B_{yy})^2 + (C_{xx} + C_{yy})^2 \tag{3.4}$$

blir en rimlig utvidgning av gråvådetektorn i ekvation 3.1. Detta uttryck har jag använt som en ny kombinerad operator som tar hänsyn till färginformation vid särdragsdetektion. Man kan också motivera valet av denna med att då gråniva-operatorns värde är högt i en viss punkt i en kanal är även normen av vektorn (där elementen är gråvådetektorns operatorsvar) hög i den punkten.

Denna operator fungerar bra som blobbdetektor i färgbilder. I avsnitt 4.2 visas resultat av användning av denna operator för särdragsdetektion i färgbilder.

Då det gäller tidsåtgång tar det något längre tid att beräkna derivateuttryck med denna detektor jämfört med gråvådetektion i de olika färgkanalerna separat (man måste göra en extra summation), men man får en stor tidsvinst vid detektion av maxpunkter i skalrummet. Detta beror på att man bara har ett skalrum att gå igenom, till skillnad från det fall då man arbetar med de olika färgkanalerna separat, då man har ett skalrum att gå igenom för varje kanal. I realtidsimplementationen med hybridpyramid istället för skalrum, se avsnitt 5.1 för information om denna och avsnitt 2.5.3 för information om hybridpyramider, är det steg som tar längst tid just detektionen av maxima. Tidsvinsten märks särskilt om det finns många särdrag, vilket det gör i ”naturliga” bilder.

Detta gör att man i en trekanalers färgbild (t.ex. en RGB-bild) kan detektera särdrag med den nya detektorn utan att det kostar tre gånger så mycket tid som för vanlig gråvåbaserad detektion.

En annan fördel med den kombinerade operatorn framför de separata kanalerna är att man inte får dubbla svar, dvs varje särdrag ger endast ett operatorsvar, medan man i de separata kanalerna kan få svar från samma särdrag i flera av kanalerna och då måste detektera att så skett och ta bort alla utom ett av dessa svar. Detta kanske inte alltid är nödvändigt, men oftast vill man bara ha ett operatorsvar från varje särdrag.

Ofta är man intresserad av att ta bort särdrag som överlappar varandra väldigt mycket och då förmodligen svarar mot samma bildstruktur, t.ex. kan en bildstruktur ge svar vid två närliggande skalor på lite olika positioner, vilket ger två detekterade särdrag. Även i detta fall är det en fördel att slippa dubbletterna man kan få i de separata kanalerna, eftersom det tar längre tid om man har många svar att undersöka.

Experiment för utvärdering av denna detektor har skett på två sätt. Först prövades den i Matlab, som är ett programverktyg med stöd för många typer av matematiska beräkningar. Där gjordes särdragsdetektion med automatiskt skalval i skalrum, på det sätt som gås igenom i avsnitt 2.5.4. I Matlab undersöktes främst detektionsförmågan hos den nya operatorn, dels jämfört med särdragsdetektion i gråvåbilder och dels jämfört med gråvådetektion separat i de olika färgka-

nalerna. Dessutom undersöktes hur valet av färgmodell påverkade operatorn. De färgmodeller som prövats finns uppräknade i avsnitt 3.2.1.

Den nya regiondetektorn har också prövats i en realtidsimplementation som beskrivs närmare i avsnitt 5.1. Där har förutom detektionsförmåga också tidsåtgång hos de olika stegen i särdragsdetektion samt särdragsdetektionens stabilitet över tiden i videosekvenser (om särdrag ”försvinner” från en bild till en annan) prövats.

3.1.2 Åsdetektorer för färgbilder

Heuristisk metod

Motiveringen att då operatorsvaret är högt i någon färgkanal är summan av operatorsvaren också hög kan även användas för åsdetektorer. Vid gråvådetektion kan man använda operatorn:

$$(L_{xx} - L_{yy})^2 + 4L_{xy}^2 \tag{3.5}$$

som ger starkt svar i punkter som ligger på en ås, för åsdetektion. Denna metod är snabbare än den metod som beskrivs i avsnitt 2.4.1. Vid åsdetektion i färgbilder kan alltså följande uttryck användas:

$$(A_{xx} - A_{yy})^2 + 4A_{xy}^2 + (B_{xx} - B_{yy})^2 + 4B_{xy}^2 + (C_{xx} - C_{yy})^2 + 4C_{xy}^2 \tag{3.6}$$

där A, B och C är de tre färgkanalerna. Denna operator har jag undersökt. Detta sätt att utvidga åsdetektion till färgbilder är mycket lik utvidgningen av blobbdetektion beskriven i förra avsnittet. Denna åsdetektor har samma egenskaper som den nya regiondetektorn, både när det gäller hur väl den fungerar och vilka för- och nackdelar den har.

Detta är ett heuristiskt sätt att motivera utvidgningen av åsdetektorn, därför kallas denna metod från och med nu den heuristiska metoden (för åsdetektion) i denna rapport. Två andra mer teoretiskt motiverade utvidgningar har också undersökts.

Dubbla vinkeln

I avsnitt 2.4.1 definieras ett lokalt koordinatsystem i en punkt med koordinataxlarna p och q sådant att $\frac{\partial^2 L}{\partial p \partial q} = 0$, vilket används vid åsdetektion. I en färgbild är man intresserad av att bilda sådana koordinatsystem i varje kanal och sedan kombinera denna information.

Ett sätt att kombinera riktningar är att addera dubbla vinkeln för riktningarna. Detta beskrivs i t.ex. Mardia (1972). Utifrån riktningen $(\cos(\beta), \sin(\beta))$ bildar man

$$(\cos(2\beta), \sin(2\beta)) = (\cos^2(\beta) - \sin^2(\beta), 2\cos(\beta)\sin(\beta)) \tag{3.7}$$

Med β bestämd av riktningen hos det lokala (p, q)-systemet blir de ingående komponenterna:

$$\cos^2(\beta) = \frac{1}{2}\Big(1 + \frac{L_{xx} - L_{yy}}{\sqrt{(L_{xx} - L_{yy})^2 + 4L_{xy}^2}}\Big) \tag{3.8}$$

$$\sin^2(\beta) = \frac{1}{2}\big(1 - \frac{L_{xx} - L_{yy}}{\sqrt{(L_{xx} - L_{yy})^2 + 4L_{xy}^2}}\big) \tag{3.9}$$

$$2\cos(\beta)\sin(\beta) = \sqrt{1 - \frac{(L_{xx} - L_{yy})^2}{(L_{xx} - L_{yy})^2 + 4L_{xy}^2}} = \frac{2|L_{xy}|}{\sqrt{(L_{xx} - L_{yy})^2 + 4L_{xy}^2}} \tag{3.10}$$

$$\cos^2(\beta) - \sin^2(\beta) = \frac{L_{xx} - L_{yy}}{\sqrt{(L_{xx} - L_{yy})^2 + 4L_{xy}^2}} \tag{3.11}$$

För att kombinera riktningarna för åssvar i de olika färgkanalerna kan man välja att summera de dubbla vinklarna över färgkanalerna C:

$$\sum_C (\cos^2(\beta) - \sin^2(\beta), 2\cos(\beta)\sin(\beta)) = \sum_C \frac{(C_{xx} - C_{yy}, 2C_{xy})}{\sqrt{(C_{xx} - C_{yy})^2 + 4C_{xy}^2}} \tag{3.12}$$

som definierar en ny riktning där det kombinerade åssvaret är starkt. Ett sätt att ta hänsyn till att de olika kanalerna kan ha olika starkt åssvar från början är att vikta riktningen med åssvaret $\sqrt{(L_{xx} - L_{yy})^2 + 4L_{xy}^2}$, vilket ger:

$$\sum_C (C_{xx} - C_{yy}, 2C_{xy}) \tag{3.13}$$

Detta blir en vektor med två komponenter:

$$\begin{bmatrix} \sum_C (C_{xx} - C_{yy}) \\ \\ \sum_C 2C_{xy} \end{bmatrix} \tag{3.14}$$

En metod för åsdetektion i färgbilder är att betrakta lokala maxima hos normen för denna vektor som punkter på en ås. Då man söker maxima påverkas inte detta av om man betraktar normen eller normen i kvadrat. Normen i kvadrat blir:

$$\Big(\sum_C (C_{xx} - C_{yy})\Big)^2 + \Big(\sum_C 2C_{xy}\Big)^2 \tag{3.15}$$

Detta uttryck för åsdetektion har också undersökts. Jag kallar denna metod ”dubbla vinkeln” i resten av denna rapport. Metoden fungerar ungefär lika bra som den heuristiska metoden. I avsnitt 4.3 jämförs de olika metoderna för åsdetektion.

Elementen i vektorn i ekvation 3.14 är summan över färgkanalerna av de diskret beräknade derivateuttrycken. Då summeringen kommuterar med derivateberäkningarna är dessa uttryck detsamma som beräkning av derivatorna i medelvärdesbilden (en gråniåvbild som är summan av färgkanalerna), vilket är den bild man vanligtvis använder vid gråniådetektion. Detta motsvarar alltså gråniådetektion av åsar, vilket gör att denna metod inte är en ”genuin” färgmetod, men den är en ny metod att beräkna åsar vars härledning baseras på innehållet i färgkanalerna.

Starkaste riktning

Låt α vara en riktning. I denna riktning kan man mäta åsstyrka som:

$$(L_{\alpha\alpha} - L_{\bar{\alpha}\bar{\alpha}})^2 \tag{3.16}$$

där L är en gråvåbild och $L_{\alpha\alpha}$ innebär partiell derivering av L två gånger i riktning α (motsvarande för $L_{\bar{\alpha}\bar{\alpha}}$), $\bar{\alpha}$ är vinkelrät mot α. Då:

$$\frac{\partial L}{\partial \alpha} = \cos(\alpha)\frac{\partial L}{\partial x} + \sin(\alpha)\frac{\partial L}{\partial y} \tag{3.17}$$

fås:

$$(L_{\alpha\alpha} - L_{\bar{\alpha}\bar{\alpha}})^2 =$$

$$(\cos(\alpha)\partial_x + \sin(\alpha)\partial_y)^2 L - (\sin(\alpha)\partial_x - \cos(\alpha)\partial_y)^2 L =$$

$$(\cos^2(\alpha) - \sin^2(\alpha))(L_{xx} - L_{yy}) + 4\cos(\alpha)\sin(\alpha)L_{xy} =$$

$$\big(\cos(2\alpha)(L_{xx} - L_{yy}) + \sin(2\alpha)2L_{xy}\big)^2 =$$

$$\cos^2(2\alpha)(L_{xx} - L_{yy})^2 + \sin^2(2\alpha)(2L_{xy})^2 + 2\cos(2\alpha)\sin(2\alpha)(2L_{xy})(L_{xx} - L_{yy}) =$$

$$\tfrac{1+\cos(4\alpha)}{2}(L_{xx} - L_{yy})^2 + \tfrac{1-\cos(4\alpha)}{2}(2L_{xy})^2 + \sin(4\alpha)(2L_{xy})(L_{xx} - L_{yy}) =$$

$$\tfrac{1}{2}((L_{xx} - L_{yy})^2 + (2L_{xy})^2) + \tfrac{\cos(4\alpha)}{2}((L_{xx} - L_{yy})^2 - (2L_{xy})^2) + \sin(4\alpha)(2L_{xy})(L_{xx} - L_{yy}) \tag{3.18}$$

Om man summerar detta uttryck över de olika färgkanalerna, C, får man ett mått på den totala åsstyrkan i färgkanalerna:

$$\sum_C \big(\frac{1}{2}((L_{xx}-L_{yy})^2+(2L_{xy})^2)+\frac{\cos(4\alpha)}{2}((L_{xx}-L_{yy})^2-(2L_{xy})^2)+\sin(4\alpha)(2L_{xy})(L_{xx}-L_{yy})\big) \tag{3.19}$$

Definiera R, S, och T som:

$$R = \sum_C (L_{xx} - L_{yy})^2 \tag{3.20}$$

$$S = \sum_C (2L_{xy})^2 \tag{3.21}$$

$$T = \sum_C L_{xy}(L_{xx} - L_{yy}) \tag{3.22}$$

och sätt in i ekvation 3.19:

$$\frac{1}{2}(R + S) + \frac{\cos(4\alpha)}{2}(R - S) + \sin(4\alpha)2T \tag{3.23}$$

För att få α till den riktning i vilken åsstyrkan är maximal deriveras uttrycket m.a.p. α och derivatan sätts lika med 0:

$$-2\sin(4\alpha)(R-S) + 4\cos(4\alpha)2T = 0 \tag{3.24}$$

och α kan bestämmas genom:

$$\tan(4\alpha) = \frac{4T}{R-S} \tag{3.25}$$

Då $\tan(\varphi) = \frac{a}{b}$ blir $\cos(\varphi) = \frac{b}{\sqrt{a^2+b^2}}$ och $\sin(\varphi) = \frac{a}{\sqrt{a^2+b^2}}$. Detta ger med ovanstående uttryck för $\tan(4\alpha)$:

$$\cos(4\alpha) = \frac{R-S}{\sqrt{(4T)^2+(R-S)^2}} \tag{3.26}$$

$$\sin(4\alpha) = \frac{4T}{\sqrt{(4T)^2+(R-S)^2}} \tag{3.27}$$

Sätts detta in i uttrycket för åsstyrkan i ekvation 3.23 fås:

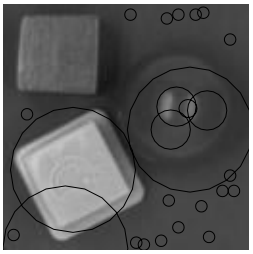
$$\frac{1}{2}(R+S+\frac{(R-S)^2+(4T)^2}{\sqrt{(4T)^2+(R-S)^2}}) = \frac{1}{2}(R+S+\sqrt{(4T)^2+(R-S)^2}) \tag{3.28}$$

$R+S$ är det uttryck den heuristiska motiveringen gav, här tas ytterligare en term med. Denna metod för åsdetektion, som jag kallar ”starkaste riktning”, har också undersökts. Skillnaden mellan denna metod och den heuristiska är liten när det gäller detekterade särdrag. Beräkningsarbetet är något större för denna metod. I avsnitt 4.3 jämförs de olika metoderna för åsdetektion i färgbilder.

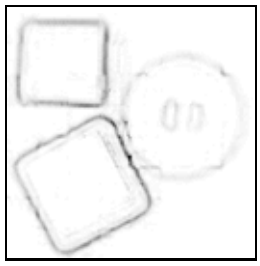
3.1.3 Andra särdragsdetektorer för färgbilder

En annan variant av blobbdetektion, då all färginformation specifikt tas med i operatorn, är att detektera regioner som de punkter där skalärprodukten av närliggande färgvärdesvektorerna i bilden har lokala maxima. Dessa maxima inträffar när vinkeln mellan vektorerna har lokala minima, dvs vektorerna är så likriktade som möjligt. Det blir en blobbdetektor av detta, men den blir inte speciellt bra. Man behöver ta hänsyn till att regionen måste kontrastera mot omgivningen också (vilket inte görs med denna metod), inte bara att den är homogent färgad. I figur 3.1 visas ett exempel på resultatet av blobdetektion med denna operator.

En närliggande metod är att ta de punkter där skalärprodukten är minimal (dvs vinkeln mellan färgvektorerna stor) som kantpunkter. Denna typ av kantdetektor fungerade bra när jag prövade den. En sådan kantdetektor finns beskriven i Gauch (1998). I figur 3.2 visas skalärproduktens värde för en bild. Man ser att skalärprodukten är låg vid kanterna.



Figur 3.1. Blobbdetektion genom detektion av lokala maxima hos skalärprodukten mellan närliggande bildelement. De 25 starkaste operatorsvaren visas.



Figur 3.2. Skalärprodukten av färgvärdena i bilden i figur 3.1.

3.2 Särdragsdetektion i olika färgrymder

Ett sätt att använda färginformation i bilder för att få bättre särdragsdetektion är att utnyttja skillnader i färg för att få bättre kontrast i indata. Då man har en gråniv̊abild kan regioner se lika ut trots att de har olika färg. Ser de lika ut blir det svårt att detektera särdrag som bygger på skillnader mellan dessa regioner, t.ex. kanter mellan regionerna. Om regionerna då skiljer sig i färg kan man utnyttja denna information för att få bättre kontrast.

Det finns många sätt att representera färg, vilket tidigare nämnts i avsnitt 2.1. Jag gjorde särdragsdetektion i ett antal olika färgmodeller för att se om någon färgmodell lämpar sig bättre än andra för särdragsdetektion i färg. För att undersöka det gjorde jag blobbdetektion och åsdetektion i ett antal bilder, främst bilder av händer mot olika bakgrunder.

Detektionen skedde på två sätt. Först gjordes särdragsdetektion separat i de olika färgkanalerna där en särdragsdetektor för gråniv̊abilder användes. De detekterade särdragen från de olika kanalerna sammanställdes sedan och dubbletter (samma särdrag detekterat i flera kanaler) togs bort så att varje särdrag bara fanns med en gång. Slutligen presenterades de N mest signifikanta särdragen, för olika värden på N.

Vid särdragsdetektion på det andra sättet användes en ny särdragsdetektor där summan av operatorerna i de separata kanalerna togs som ny operator, mer information om denna finns i avsnitt 3.1.1. Även här presenterades de N starkaste operatorsvaren för olika N.

Detektionen skedde i samtliga fall med automatiskt skalval på det sätt som beskrivs i avsnitt 2.5.4. I de fall då åsdetektion gjordes genomfördes också ett hoplänkningssteg, se avsnitt 3.4.3.

För blobbar användes gråniv̊adetektorn

$$t^2(L_{xx} + L_{yy})^2 \tag{3.29}$$

(se avsnitt 2.4.1) och den nya operatorn

$$t^2((A_{xx} + A_{yy})^2 + (B_{xx} + B_{yy})^2 + (C_{xx} + C_{yy})^2) \tag{3.30}$$

där A, B och C är de olika färgkanalerna och t är skalparametern.

För åsar användes operatorn

$$\sqrt{t}^3((L_{xx} - L_{yy})^2 + 4L_{xy}^2) \tag{3.31}$$

respektive

$$\sqrt{t}^3((A_{xx} - A_{yy})^2 + 4A_{xy}^2 + (B_{xx} - B_{yy})^2 + 4B_{xy}^2 + (C_{xx} - C_{yy})^2 + 4C_{xy}^2) \tag{3.32}$$

som ger starkt svar i punkter som motsvarar åsar. Denna metod för åsdetektion är snabbare än den metod som tas upp i avsnittet 2.4.1.

För att få balans mellan färgkanalerna normaliserades varje kanal så att värdena i kanalerna låg i intervallet [0,1]. Normaliseringen gjordes genom:

$$v'_c(x, y) = \frac{\left(v_c(x, y) - min_c\right)}{max_c - min_c} \tag{3.33}$$

där $v_c(x, y)$ är värdet i punkten (x, y) i kanal c, max_c och min_c är största respektive minsta värdet i kanal c och v' är det normaliserade värdet. Detta är en mycket enkel metod som kan ge problem om alla värden i en kanal ligger inom ett begränsat intervall. Då sträcks intervallet ut till hela det nya intervallet ($[0, 1]$) vilket istället för att ge balans mellan färgkanalerna har motsatt effekt samt förstärker brus. De bilder som undersökts har inte haft detta problem.

Om kanalerna ej normaliseras kan man få problem med att en kanal ger mycket starkare operatorsvar för särdrag som egentligen skulle betraktas som svaga, än operatorsvar i andra kanaler som ska betraktas som starka.

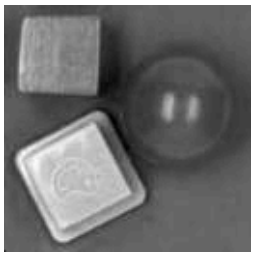
3.2.1 Undersökta färgmodeller

De färgmodeller som undersökts är följande (de flesta beskrivs mer utförligt i avsnitt 2.1):

- RGB, den vanliga RGB-modellen.
- HSI,
  - $I = \frac{R+G+B}{3}$
  - $S = 1 - \frac{min(R,G,B)}{R+G+B}$
  - $H = acos(\frac{2R-G-B}{2\sqrt{D}})$, där $D = (R-G)^2 + (R-B)(G-B)$, och H sätts till $2\pi - H$ om $B > G$
- YIQ, färgmodellen som används i färg-tv, främst i USA.
- YUV, färgmodellen som används i färg-tv i Europa. YIQ och YUV påminner mycket om varandra. Y-kanalen är precis densamma, medan de andra två kanalerna skiljer sig lite åt.
- Gaussisk färgmodell, i Geusebroek, van den Boomgaard, Smeulders och Dev (2000) beskrivs en färgmodell där man betraktar de olika färgkanalerna som gaussderivator med avseende på våglängd. Beskrivs kort i avsnitt 2.1.2.
- Sfäriska koordinater ruv, man betraktar (R,G,B) som en vektor och byter till sfäriska koordinater där $r = \sqrt{R^2 + G^2 + B^2}$, $tan(u) = G/R$, $tan(v) = B/\sqrt{R^2 + G^2}$.

3.2.2 Innehållet i färgkanalerna hos olika färgmodeller

I figur 3.3 visas gråvårepresentationen av en bild av några plastleksaker. I figur 3.4 visas innehållet i de olika färgkanalerna i de färgmodeller som använts för samma bild. I gråvåbilden är kontrasten mellan den röda bollen till höger och den lila bakgrunden väldigt dålig, medan samtliga färgmodeller har minst en kanal med god kontrast mellan bollen och bakgrunden. Här skulle alltså färginformationen vara värdefull för att detektera bollen.



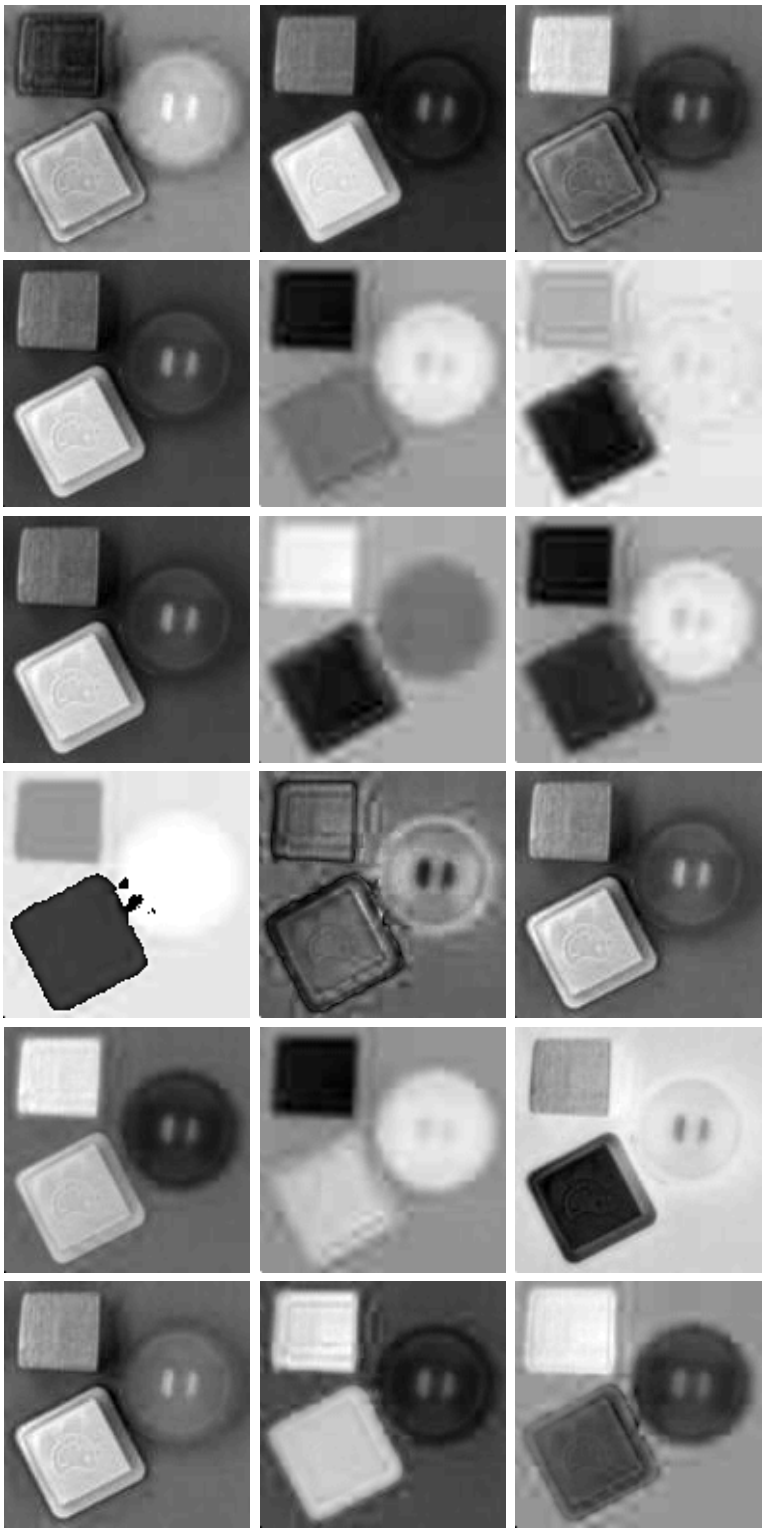
Figur 3.3. Gråvårepresentation av en bild av några leksaker. Bilden är tagen från webbsidan `http://zomax.wins.uva.nl:5345/vis_user/`

En annan sak man kan notera är att bilden ursprungligen var komprimerad med jpeg-komprimering. Denna delar in bilden i block och arbetar på varje block för sig, vilket vanligtvis inte märks (algoritmen är anpassad så att människor inte ska se blockkanterna) och i RGB-kanalerna syns inte blocken. Däremot ser man kantigheter i bilden i andra färgmodeller, t.ex. YIQ. Detta kan leda till vissa problem vid särdragsdetektion, t.ex. kan kantdetektion i jpeg-komprimerade bilder ge oväntade och oönskade resultat. De flesta bilder jag gjort särdragsdetektion på har, för att undvika sådana problem, inte varit komprimerade. Denna bild är bra för att illustrera vissa egenskaper hos färgmodeller, därför använder jag den som exempel trots att den varit komprimerad.

3.3 Viktning av särdrags signifikans med avseende på sökt färg

Ett annat sätt att använda färginformationen i bilder vid särdragsdetektion är att kontrollera hur lika till färgen särdragen är en viss sökt färg. I de tillämpningar jag främst tittat på (gestigenkänning) söker man särdrag som svarar mot delar av en hand, t.ex. fingertoppar (blobbar), fingrar (åsar) och handflatan (blobb). Om man har en naturlig bakgrund, t.ex. en kontorsmiljö, ger särdragsdetektorer många starka svar i bakgrunden. Att på ett intelligent sätt sålla bort dessa kan vara svårt.

En metod man då kan använda är att vikta särdragens signifikans efter hur ”hudfärgade” de områden i bilden särdragen svarar mot är. Jag har prövat detta på ett enkelt sätt. Först tog jag en bild av en hand och segmenterade manuellt ut



Figur 3.4. Färgkanalernas informationsinnehåll i de färgmodeller som använts. Varje rad är en färgmodell, de tre kanalerna ligger från vänster till höger i den ordning de kommer i namnet (RGB ger först R sedan G och sist B). De modeller som visas är uppifrån och ned: RGB, YIQ, YUV, HSI, Gaussisk färgmodell och sfäriska koordinater.

handen och gjorde hela bakgrunden svart. Sedan gjorde jag ett 3D-histogram där varje cell svarade mot en (R, G, B)-koordinat och räknade för varje cell hur många punkter i bilden som svarade mot den färgen. Värdet för cellen för bakgrundsfärgen (svart, $(0, 0, 0)$) sattes sedan till 0. Detta förfarande kan upprepas för flera bilder för att få bättre robusthet.

Vid särdragsdetektion tog jag för varje särdrag den region av bilden som motsvarade särdraget (en cirkel med radien proportionell mot detektionsskalan för blobbar, ellipser för åsar) och beräknade ett värde baserat på färgerna i dessa punkter i bilden, där varje färgs värde är värdet i färgens cell i histogrammet. De färger som förekom ofta i ”hudfärgsbilden” får då höga värden medan färger som inte förekommer i ”hudfärgsbilden” har värdet 0. Särdragens signifikans viktades med deras likhet med hudfärg genom att signifikansvärdet multiplicerades med medelvärdet av histogramvärdena för färgerna i de punkter som ingick i särdragets region.

Vikten w_r för en region r blir alltså:

$$w_r = \frac{\sum_{i=1}^{n} H_{R(r_i),G(r_i),B(r_i)}}{n} \tag{3.34}$$

där n är antalet punkter i r, r_i är punkt i i regionen, $R(r_i)$ är RGB-värdet för R-kanalen i punkten r_i (motsvarande för $G(r_i)$ och $B(r_i)$) och slutligen är $H_{R,G,B}$ värdet i histogrammet i cellen för RGB-värdet (R, G, B). Den nya signifikansen s'_r för regionen r blir då:

$$s'_r = w_r s_r \tag{3.35}$$

där s_r är signifikansen för r före viktningen.

Detta fungerade bra, de hudfärgade särdragen blev då de starkaste särdragen (se avsnitt 4.4 och avsnitt 5.3.2). Metoden kan naturligtvis användas till att vikta särdragen med vilken sökt färg som helst, bara man skapar ett histogram för den sökta färgen. Det går också bra att ha flera önskade färger genom att t.ex. addera värdena i flera histogram.

Den här metoden är tämligen tidskrävande. Det som främst tar tid är att gå igenom alla punkter i regionen och kontrollera deras färg. Man kan för att snabba upp detta kontrollera endast en delmängd av dessa punkter, t.ex. de närmast centrum av regionen eller några slumpvis utvalda punkter. Jag har provat att istället ta ett antal slumpvis valda punkter ur regionen, vilket fungerar bra och är lite snabbare än att behandla alla punkter. Dessutom har jag provat att bara titta på centrumpunkten i regionen, vilket går mycket fortare och fungerar ungefär lika bra. I avsnitt 4.4 och avsnitt 5.3.2 har viktning av särdrag på dessa sätt undersökts.

Man kan få problem med denna metod om man har mycket spekulariteter i bilden, eftersom man då kan få en vit (egentligen belysningskällans färg) fläck på objektet och om den ligger mitt i regionen objektet upptar får man fel uppfattning om regionens färg då man bara tittar på centrumpunkten (eller de närmast centrum). Slumpvis valda punkter eller att titta på alla punkter är då mer robust, men det löser inte problemet helt, eftersom regionen då inte har lika mycket av den färg man söker om stora delar är spekulariteter (har ”fel” färg).

Viktning av signifikans med avseende på sökt färg har vissa problem. Ett problem är att färgen inte alltid är densamma utan varierar, t.ex. beroende på belysnings- och kameraförhållanden. Detta kan åtgärdas på olika sätt, se avsnitt 2.2 och 2.6 för beskrivning av problem och möjliga åtgärder.

Man kan även använda andra sätt att mäta hur lik en sökt färg är en viss färg i bilden. Man kan använda sig av andra färgmodeller och inte nödvändigtvis använda alla färgkanaler, t.ex. kan man ta bara H-kanalen i HSI-modellen. Man kan också byta histogrammet mot en sannolikhetsfunktion (som t.ex. kan beräknas från ett färghistogram) som talar om hur stor sannolikheten är att en viss färg motsvarar den sökta färgen. Ett histogram kan ses som en uppskattning av frekvensfunktionen för en sannolikhetsfördelning.

Jag har i realtidsimplementationen av särdragsdetektion (avsnitt 5.1) testat histogrammetoden dels med hela regionen, dels med slumpvis valda punkter i regioner och dels med endast centrumpunkten i regionen. Jag har främst testat att vikta efter likhet med hudfärg.

3.4 Metoder för grafisk presentation av detekterade särdrag

3.4.1 Blobbar

I denna rapport presenteras detekterade blobbar grafiskt genom att en cirkel som motsvarar blobben ritas ut ovanpå bilden blobben detekterats i. Denna cirkel har centrum i den punkt i bilden där blobben detekterats och cirkelns storlek beror på den skala blobben detekterats på. En blobb som detekteras på skalnivå t illustreras med en cirkel med radie $\sqrt{t}$.

3.4.2 Åsar

Då åsar är långsträckta regioner och regioner illustreras med cirklar är det naturligt att illustrera åsar med ellipser. Då en ås detekteras i en punkt beräknas den ellips som representerar åsen genom att betrakta den skala åsen detekteras på och andramomentsmatrisen:

$$\mu = \int \begin{pmatrix} L_x^2 & L_x L_y \\ L_x L_y & L_y^2 \end{pmatrix} \omega(x, y) dx dy \tag{3.36}$$

där ω är en fönsterfunktion kring den punkt man är intresserad av.

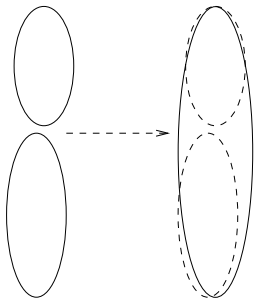
Ellipsens korta halvaxel, b, (åsens bredd) sätts till $\sqrt{t}$, där t är den skala åsen detekterats på. Den långa halvaxeln, a, sätts till:

$$a = b\frac{\lambda_1}{\lambda_2} \tag{3.37}$$

där λ_1 och λ_2 är andramomentsmatrisens största respektive minsta egenvärde. Ellipsens orientering bestäms genom att låta den långa halvaxeln a få samma riktning som den egenvektor som hör till det största egenvärdet hos μ.

3.4.3 Hoplänkning av närliggande åsar

Vid åsdetektion i skalrumsrepresentationer genomfördes ett hoplänkningssteg i detektionsprocessen. Detta steg innebär att detekterade åsar som har ungefär samma bredd, ungefär samma riktning och ligger ”på rad” slås ihop till en ås som sträcker sig från starten av den första åsen till slutet av den andra åsen, se figur 3.5.



Figur 3.5. Ihoplänkning av två åsar till en längre ås.

Mer formellt krävdes följande för att två åsar, i och j, ska länkas ihop:

- att $|b_i - b_j| < 4$, där b_i är den korta halvaxelns längd (bredden) hos ellipsen som representerar ås i (detektionsskala ganska lika)
- att $\sqrt{(\alpha_c - \alpha_i)^2 + (\alpha_c - \alpha_j)^2} < 0.15$, där α_c är riktningen mellan centrumpunkterna för ellipserna, α_i är riktningen för den långa halvaxeln för ellips i och vinklarna mäts i radianer (ellipserna ligger på rad)
- att $a_i + a_j - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} > 0$, där (x_i, y_i) är centrumpunkten för ellips i (ellipserna överlappar)

Om dessa krav är uppfyllda slås de två åsarna ihop till en ny ås där:

- signifikansen är den högsta av de hopslagna åsarnas signifikanser
- längden på den långa halvaxeln $a' = \frac{a_i + a_j + \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{2}$
- längden på den korta halvaxeln är samma som hos den ås som hade den längsta långa halvaxeln
- $\alpha' = \frac{\alpha_i a_i + \alpha_j a_j}{a_i + a_j}$
- $(x', y') = \frac{(x_i, y_i)_i a_i + (x_j, y_j) a_j}{a_i + a_j}$

Kapitel 4

Resultat vid användande av färginformation vid särdragsdetektion

4.1 Valet av färgmodells inverkan på särdragsdetektion i färgbilder

De färgmodeller som fungerade bra vid detektion i de olika färgkanalerna separat var också de modeller som fungerade bra för de kombinerade särdragsdetektorerna. Likaså påverkades blobbdetektion och åsdetektion likadant av de olika färgmodellerna. De följande diskussionerna om de olika modellernas egenskaper gäller för samtliga dessa typer av särdragsdetektion i färgbilder.

Exempel på resultaten vid särdragsdetektion finns i figur 4.3 och 4.6 (blobbar med den kombinerade operatorn), figur 4.2 och 4.5 (blobbar separat i de olika färgkanalerna), figur 4.9 (åsar med den kombinerade åsdetektorn) och figur 4.8 (åsar separat i de olika färgkanalerna).

Bildstorleken är i samtliga fall 128x128 pixlar. Särdragsdetektion har skett med automatiskt skalval i skalrum, skalrummet hade 13 nivåer, $8 \leq t \leq 4000$. De operatoruttryck som använts är

$$t^2(L_{xx} + L_{yy})^2 \tag{4.1}$$

(blobbar en kanal i taget),

$$t^2((A_{xx} + A_{yy})^2 + (B_{xx} + B_{yy})^2 + (C_{xx} + C_{yy})^2) \tag{4.2}$$

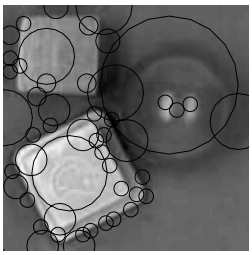
(blobbar tre kanaler),

$$\sqrt{t}^3((L_{xx} - L_{yy})^2 + 4L_{xy}^2) \tag{4.3}$$

(åsar en kanal i taget) samt

$$\sqrt{t}^3((A_{xx} - A_{yy})^2 + 4A_{xy}^2 + (B_{xx} - B_{yy})^2 + 4B_{xy}^2 + (C_{xx} - C_{yy})^2 + 4C_{xy}^2) \tag{4.4}$$

(åsar tre kanaler). Dubbletter (samma särdrag detekterat i olika kanaler) har eliminerats.



Figur 4.1. De 50 mest signifikanta regionerna i bilden av leksaker, detekterade med gråvådetektion.

Det visade sig att HSI gav sämre resultat än förväntat, vilket till viss del förmodligen beror på att H-kanalen, som jag trodde skulle ge intressant information, inte gjorde det. Ett problem med H-kanalen är att den är periodisk, H beräknas som:

$$H = acos(\frac{2 * R - G - B}{2\sqrt{D}}) \tag{4.5}$$

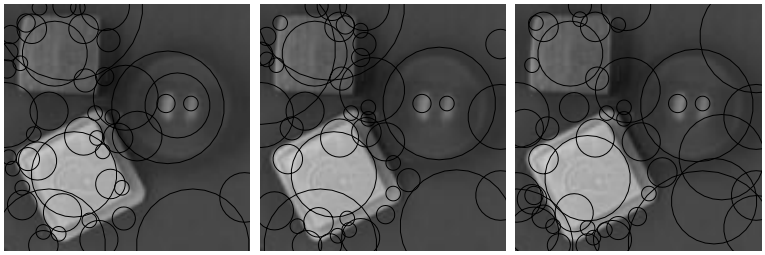
där $D = (R - G)^2 + (R - B) * (G - B)$ och H sätts till $2\pi - H$ om $B > G$. Ett värde nära H-kanalens maximum ligger alltså egentligen nära minimum också, vilket inte tas i beaktande vid det nuvarande testförfarandet. Detta gör att man i de regioner som har värden vid maximum eller minimum kan få många små regioner med väldigt hög kontrast (vilket ger hög signifikans för det detekterade särdraget) när vissa delar av regionen ligger på ena sidan gränsen mellan minimum och maximum och andra delar på andra sidan. Man kan också få instabilitet då nämnaren i divisionen vid beräkningen av H ligger nära 0.

En annan mindre nackdel med HSI-modellen är att det beräkningsmässigt är tungt att byta till HSI-modellen, jämfört med många av de andra modellerna. Dock är inte detta ett så stort problem, eftersom många andra steg i särdragsdetektionen tar mycket längre tid.

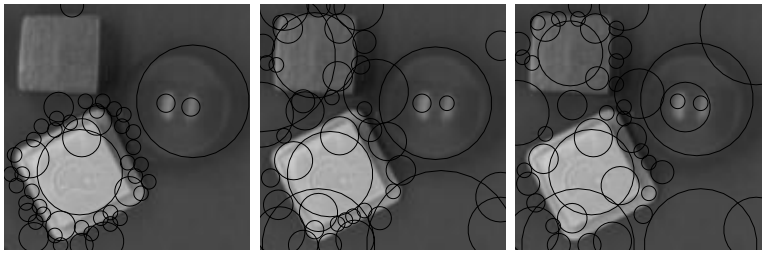
Den gaussiska färgmodellen gav bra resultat och är i övrigt också tilltalande eftersom den har flera teoretiska fördelar. Det är ganska enkelt (beräkningsmässigt) att byta till gaussisk färgmodell.

YIQ och YUV hade likartade resultat och var båda bra. Det krävs samma mängd beräkningsarbete som för att konvertera till den gaussiska färgmodellen.

Sfäriska koordinater gav bra resultat. Det krävs, precis som för HSI, en del tunga beräkningar vid byte från RGB till denna modell. Värt att notera är att trots att två kanaler i denna färgmodell motsvarar vinklar lider inte modellen av det problem som drabbar H-kanalen i HSI-modellen, dvs att kanalerna är periodiska. Detta beror på att alla RGB-värden ligger i samma kvadrant, så de vinklar som är aktuella ligger i ett begränsat område.

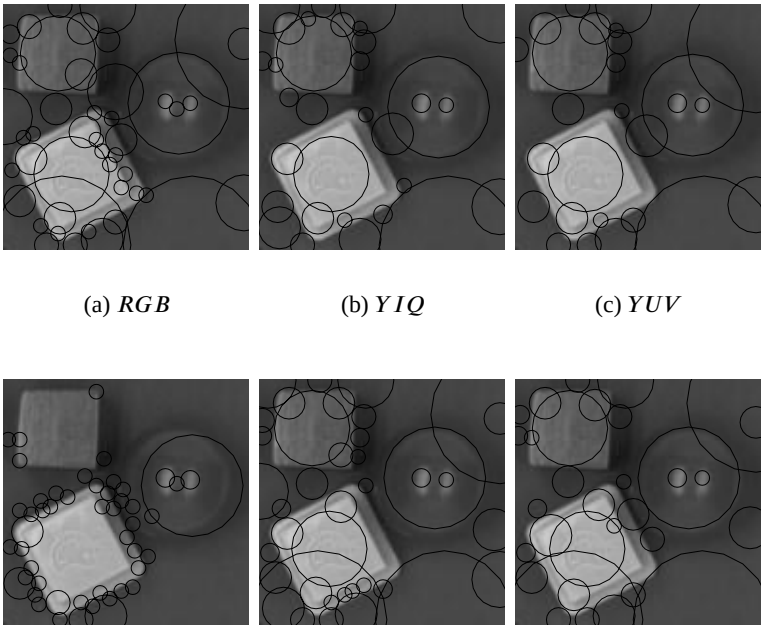


(a) RGB (b) YIQ (c) YUV



(d) HSI (e) Gaussisk (f) Sfäriska koordinater

Figur 4.2. De 50 mest signifikanta regionerna i bilden av leksaker, detekterade genom detektion med gråvåoperatorn separat i varje kanal, varefter resultaten sammanställts och dubbletter eliminerats. Resultatet visas för olika färgmodeller.



(a) RGB (b) YIQ (c) YUV

(d) HSI (e) Gaussisk (f) Sfäriska koordinater

Figur 4.3. De 50 mest signifikanta regionerna i bilden av leksaker, detekterade med den kombinerade operatorn. Resultatet visas för olika färgmodeller.

r-kanalen (magnituden av RGB-vektorn) blir en intensitetsbild, detsamma gäller I-kanalen från HSI, Y-kanalerna i både YIQ och YUV samt den första kanalen i den gaussiska färgmodellen. En kanal som motsvarar intensiteten är lämpligt ur den synvinkeln att man då vid särdragsdetektion i den färgmodellen åtminstone inte missar särdrag man skulle hittat vid traditionell särdragsdetektion (i enbart en intensitetsbild). Dock kan dessa särdrag ”dränkas” i mängden av särdrag detekterade i andra färgkanaler, som t.ex. diskuterats ovan för HSI-modellen kan man, t.ex. på grund av brus i bilden, få särdrag som inte är intressanta men ändå har höga signifikansvärden.

RGB gav inte bäst resultat, men bra resultat och har den fördelen att indata vanligen är i RGB så inga extra beräkningssteg krävs för att konvertera till denna modell.

4.2 Jämförelse av de olika metoderna för särdragsdetektion

Resultaten vid särdragsdetektion med de olika metoder som tagits upp i denna rapport finns i figur 4.3 och 4.6 (blobbar med den kombinerade operatorn), figur 4.2 och 4.5 (blobbar separat i de olika färgkanalerna), figur 4.9 (åsar med den kombinerade åsdetektorn) och figur 4.8 (åsar separat i de olika färgkanalerna).

Bildstorleken är i samtliga fall 128x128 pixlar. Särdragsdetektion har skett med automatiskt skalval i skalrum, skalrummet hade 13 nivåer, $8 \leq t \leq 4000$. De operatoruttryck som använts är

$$t^2(L_{xx} + L_{yy})^2 \tag{4.6}$$

(blobbar en kanal i taget),

$$t^2((A_{xx} + A_{yy})^2 + (B_{xx} + B_{yy})^2 + (C_{xx} + C_{yy})^2) \tag{4.7}$$

(blobbar tre kanaler),

$$\sqrt{t}^3((L_{xx} - L_{yy})^2 + 4L_{xy}^2) \tag{4.8}$$

(åsar en kanal i taget) samt

$$\sqrt{t}^3((A_{xx} - A_{yy})^2 + 4A_{xy}^2 + (B_{xx} - B_{yy})^2 + 4B_{xy}^2 + (C_{xx} - C_{yy})^2 + 4C_{xy}^2) \tag{4.9}$$

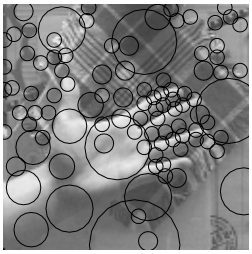
(åsar tre kanaler). Dubbletter (samma särdrag detekterat i olika kanaler) har eliminerats.

I figur 4.1 visas resultatet av vanlig blobbdetektion med automatiskt skalval på gråvåbilden av några leksaker. Det ser ut som om den röda bollen till höger i bilden har detekterats, men det är inte riktigt så. Det som detekterats som en region är bollen och bakgrunden tillsammans uppe i högra hörnet, som kontrasterar mot de båda andra leksakerna. Man ser detta bl.a. genom att lokaliseringen av särdraget

inte stämmer överens med bollens position, i figur 4.2 och figur 4.3 där bollen detekterats är särdragets positionering betydligt bättre.

I figur 4.2 har särdragsdetektion gjorts separat i de olika färgkanalerna med gråvåoperatorn. Sedan har alla detekterade särdrag från de olika kanalerna slagits samman och dubbletter har eliminerats. Här har bollen detekterats ordentligt. Man ser också att HSI-modellen inte har detekterat en av de andra leksakerna bland de 50 mest signifikanta regionerna. Denna har detekterats, t.ex. i I-kanalen, som är densamma som gråvåbilden i figur 4.1, men många små ointressanta regioner detekteras med höga signifikansvärden och döljer på så sätt regioner som svarar mot intressanta särdrag (de andra leksakerna).

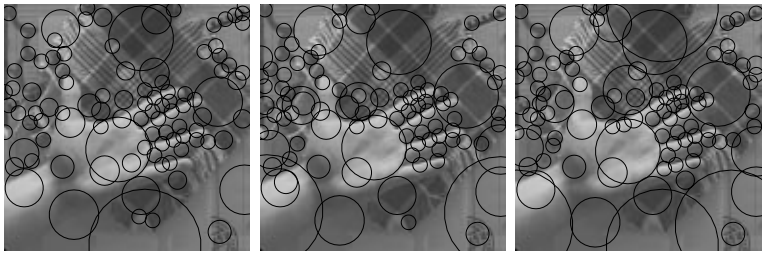
I figur 4.3 har särdragsdetektion gjorts med den kombinerade operatorn från avsnitt 3.1.1, i olika färgmodeller. Även här har bollen detekterats, men återigen har man i HSI-modellen missat intressanta särdrag, här båda de andra leksakerna, bland de 50 mest signifikanta regionerna. Annars har samtliga modeller hittat de intressanta särdragen.



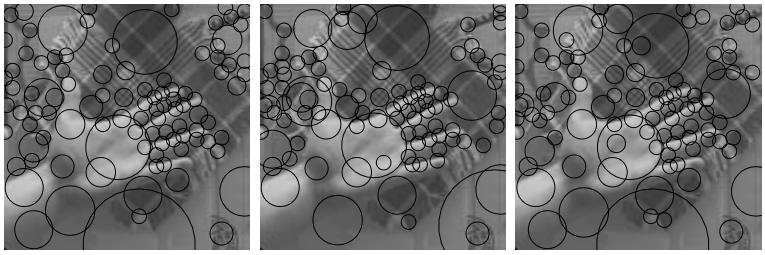
Figur 4.4. De 100 mest signifikanta regionerna, detekterade genom gråvådetektion.

I figur 4.4, figur 4.5 och figur 4.6 har blobbdetektion gjorts på olika sätt i en bild av en hand mot en komplex bakgrund. Här ser man att gråvådetektion inte får med lillfingerspetsen bland de 100 mest signifikanta regionerna. Vissa färgmodeller har inte heller detekterat lillfingerspetsen, medan andra har detekterat den. De färgmodeller som detekterat den har gjort det både då detektion skett separat i de olika färgkanalerna och då detektion skett med den kombinerade operatorn.

I figur 4.7, figur 4.8 och figur 4.9 har åsdetektion gjorts på olika sätt i en bild av en hand mot en enkel bakgrund. Här ser man att samtliga färgmodeller har fått med de intressanta särdragen, men att i just detta exempel gav färginformationen inte så mycket, eftersom man med enbart gråvådetektion också detekterade de intressanta särdragen. Detta beror på att gråvåkontrasten är hög mellan handen och bakgrunden. Hade däremot gråvåkontrasten i bilden varit sämre men kontrasten i färg märkbar hade de metoder som utnyttjar färg fungerat bättre än gråvåbaserad detektion.

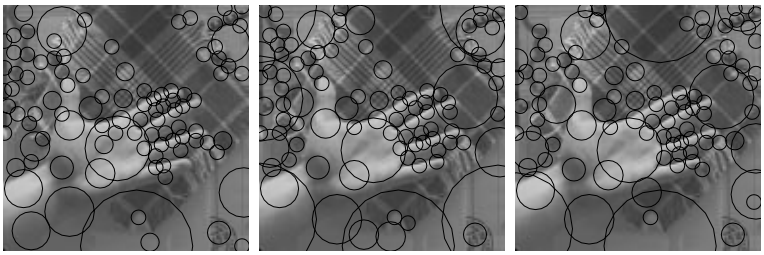


(a) RGB (b) YIQ (c) YUV

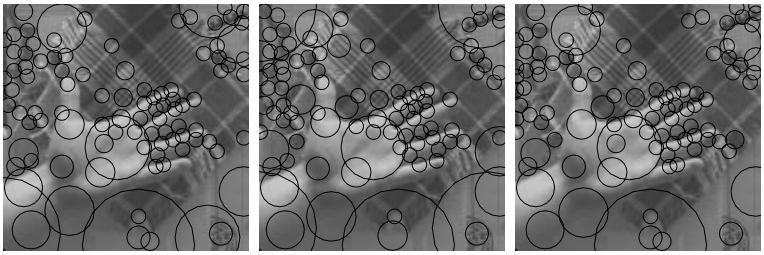


(d) HSI (e) Gaussisk (f) Sfäriska koordinater

Figur 4.5. De 100 mest signifikanta regionerna, detekterade genom detektion med gråvåoperatorn separat i varje kanal, varefter resultaten sammanställts och dubbletter eliminerats. Resultatet visas för olika färgmodeller.

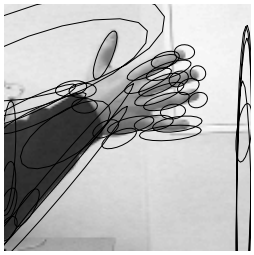


(a) RGB (b) YIQ (c) YUV

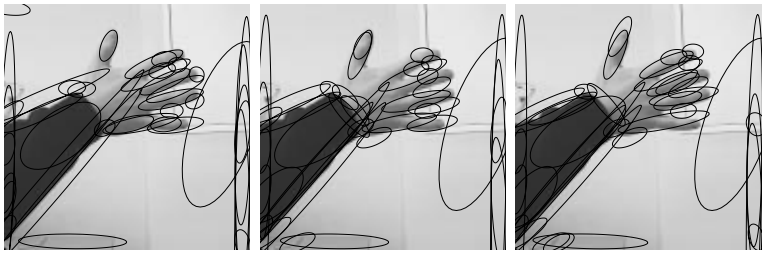


(d) HSI (e) Gaussisk (f) Sfäriska koordinater

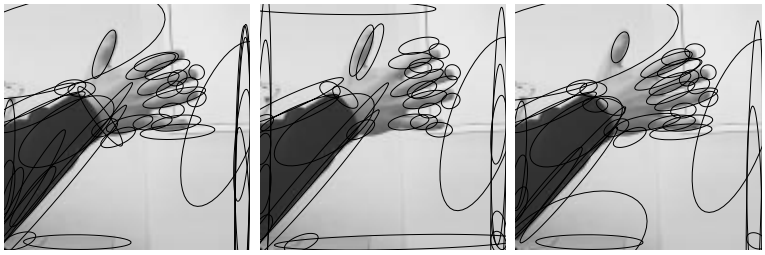
Figur 4.6. De 100 mest signifikanta regionerna, detekterade genom detektion med den kombinerade regiondetektorn. Resultatet visas för olika färgmodeller.



Figur 4.7. De 40 mest signifikanta åsarna, detekterade genom gråvådetektion, varefter sammanlänkning av åsar skett och dubbletter eliminerats.

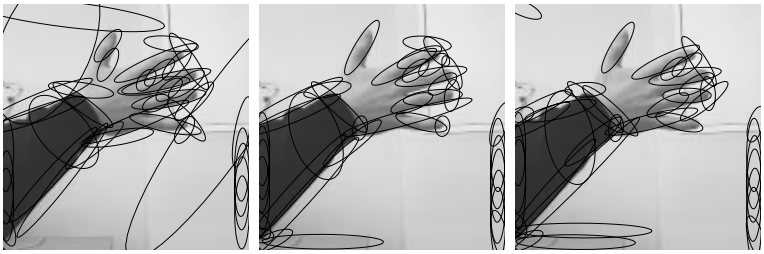


(a) RGB (b) YIQ (c) YUV

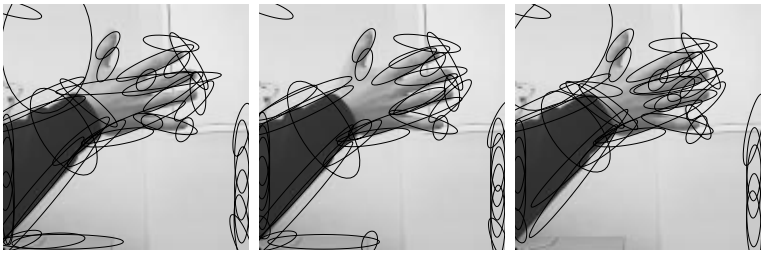


(d) HSI (e) Gaussisk (f) Sfäriska koordinater

Figur 4.8. De 40 mest signifikanta åsarna, detekterade genom detektion med gråni-våoperatorn separat i varje kanal, varefter resultaten sammanställts, sammanlänkning av åsar skett och dubbletter eliminerats. Resultatet visas för olika färgmodeller.



(a) RGB (b) YIQ (c) YUV

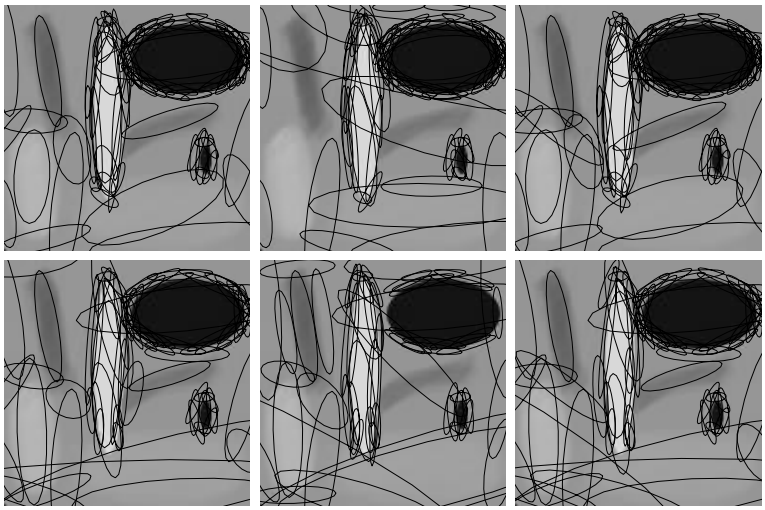


(d) HSI (e) Gaussisk (f) Sfäriska koordinater

Figur 4.9. De 40 mest signifikanta åsarna, detekterade genom detektion med den kombinerade åsdetektorn, varefter sammanlänkning av åsar skett och dubbletter eliminerats. Resultatet visas för olika färgmodeller.

4.3 Jämförelse av de kombinerade uttrycken för åsdetektion

I figur 4.10 och figur 4.11 finns två exempel på åsdetektion med de kombinerade uttrycken från avsnitt 3.1.2. Åsdetektion har skett på två olika bilder, båda med 128x128 pixlar. Åsdetektionen gjordes med automatiskt skalval i skalrum, skalparametern t låg i intervallet $2 \leq t \leq 1000$, skalrummet hade 15 skalnivåer. Utrensning av dubbletter och hoplänkning av åsar har skett, sedan har de 100 mest signifikanta åsarna visats. Två olika färgmodeller har använts, dels RGB och dels den gaussiska färgmodellen. I figur 4.12 och figur 4.13 visas de 10 mest signifikanta åsarna av de 100 åsarna i figur 4.10 respektive 4.11.

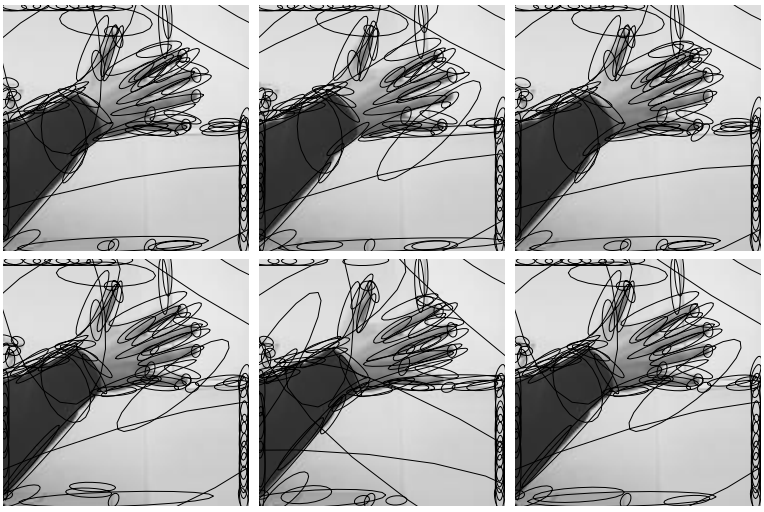


Figur 4.10. 100 mest signifikanta åsarna. I övre raden har färgmodellen RGB använts, i undre raden den gaussiska färgmodellen. Till vänster ses resultatet av särdragsdetektion med den heuristiska metoden, i mitten ”dubbla vinkeln” och till höger ”starkaste riktning”. I mittenkolumnen har ett par åsar missats, främst den böjda åsen under den stora svarta, som inte kommit med i någon av färgmodellerna.

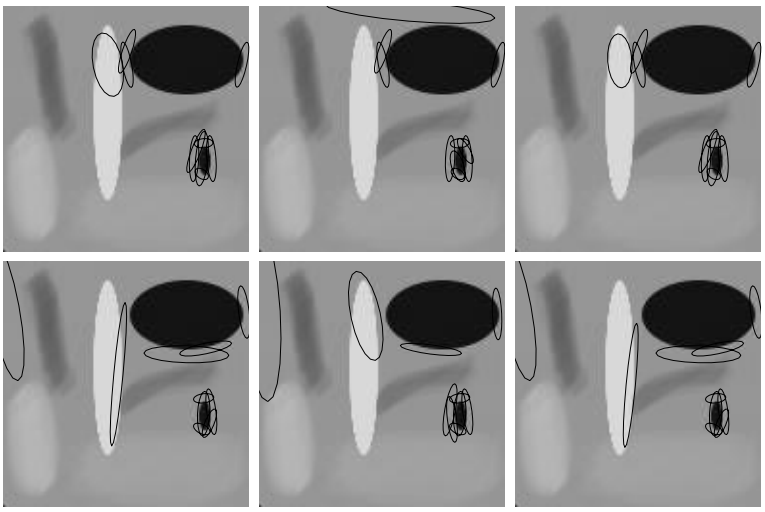
Bilden i figur 4.10 är en syntetisk bild med åsar av varierande längd, bredd och färg. Den heuristiska metoden och starkaste riktning-metoden är mycket lika när det gäller detekterade särdrag. Båda metoderna har detekterat åsarna med god precision i både position och skala. Dubbla vinkeln-metoden däremot är något sämre och har missat ett par åsar, främst den böjda åsen under den stora svarta åsen.

I bilden i figur 4.11 är resultatet mycket lika för alla de kombinerade uttrycken. Samtliga har fått med de intressanta åsarna (fingrarna) med god precision.

I figur 4.12 syns att samtliga metoder främst har detekterat färgskillnaden mellan de svarta delarna och bakgrunden. Detta beror på att kontrasten i samtliga

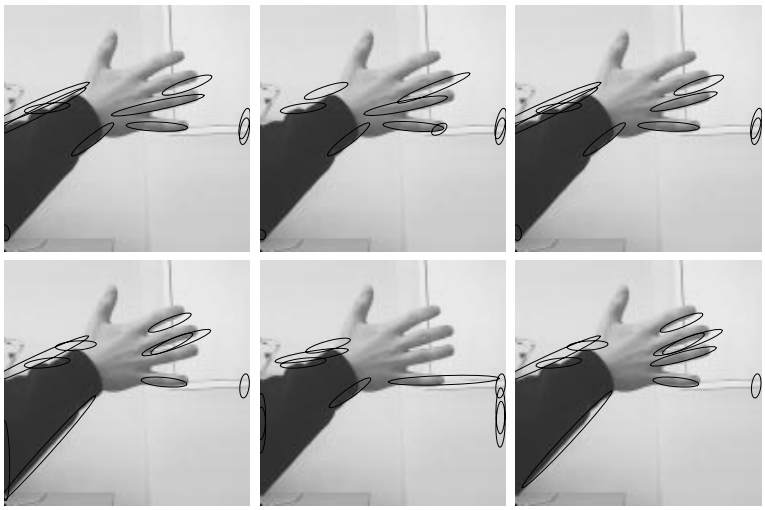


Figur 4.11. 100 mest signifikanta åsarna. I övre raden har färgmodellen RGB använts, i undre raden den gaussiska färgmodellen. Till vänster ses resultatet av särdragsdetektion med den heuristiska metoden, i mitten ”dubbla vinkeln” och till höger ”starkaste riktning”. I samtliga resultat har de intressantaste åsarna (fingrarna) kommit med.



Figur 4.12. 10 mest signifikanta åsarna av de i figur 4.10. I övre raden har färgmodellen RGB använts, i undre raden den gaussiska färgmodellen. Till vänster ses resultatet av särdragsdetektion med den heuristiska metoden, i mitten ”dubbla vinkeln” och till höger ”starkaste riktning”.

färgkanaler där är mycket hög, vilket gör att de åsar som detekteras får mycket hög signifikans.



Figur 4.13. 10 mest signifikanta åsarna av de i figur 4.11. I övre raden har färgmodellen RGB använts, i undre raden den gaussiska färgmodellen. Till vänster ses resultatet av särdragsdetektion med den heuristiska metoden, i mitten ”dubbla vinkeln” och till höger ”starkaste riktning”.

I figur 4.13 ser man att även för de 10 mest signifikanta åsarna är den heuristiska metoden och starkaste riktning-metoden mycket lika. Däremot skiljer sig dubbla vinkeln-metoden från de andra genom att färre intressanta särdrag (fingrarna) har hamnat bland de 10 starkaste åssvaren.

Sammanfattningsvis kan sägas att dubbla vinkeln-metoden är något sämre än de båda andra metoderna, som är mycket lika. Då den heuristiska metoden kräver något mindre beräkningsarbete kan den vara att föredra, medan den teoretiska motiveringen för starkaste riktning-metoden är bättre än den heuristiska motiveringen.

4.4 Effekten av viktning av särdrags signifikans

I figur 4.14 och figur 4.15 visas särdragsdetektion med respektive utan viktning av särdragens signifikans med dessas likhet med hudfärg. Viktningen har baserats enbart på centrumpunkten. I figur 5.5 har viktning gjorts på flera olika sätt för samma bild. Se avsnitt 3.3 och ekvationerna 3.34 och 3.34 för en mer detaljerad beskrivning av hur viktningen har gått till.

Man ser att viktningen kraftigt styr in de detekterade regionerna så att de som ligger i områden med hudfärg prioriteras. I figur 4.14 finns det även regioner som inte är områden med hud men som har snarlik färg, vilket gör att även dessa rankas högt. Ett par regioner som inte liknar hudfärg alls har också kommit med, vilket

beror på att de har hög signifikans före viktningen och att det inte finns fler starka hudfärgade regioner.

Figur 4.14. 100 mest signifikanta regioner, med (vänster) respektive utan (höger) viktning av särdragens signifikans med deras likhet med hudfärg.

Figur 4.15. 100 mest signifikanta regioner, med (vänster) respektive utan (höger) viktning av särdragens signifikans med deras likhet med hudfärg.

4.5 Särdragsdetektionens stabilitet över tiden

I realtidsimplementationen, se avsnitt 5.1, finns möjlighet att göra särdragsdetektion i videosekvenser. I dessa sekvenser har jag studerat detektionens stabilitet över tiden, dvs om särdrag som detekteras i en bildruta detekteras i de följande bildrutorna med ungefär samma värden för position, skala och signifikans.

Då det objekt i bilden särdraget motsvarar försvinner ur bilden på ett eller annat sätt eller rör sig så att objektets storlek i bilden eller skillnaden mellan objektet och bakgrunden ändras mycket kommer särdraget naturligtvis inte detekteras med liknande egenskaper som innan förändringarna. Därför har jag endast tittat på de

särdrag som representerar objekt i bilden som finns kvar över en längre tid och rör sig ganska långsamt eller inte alls.

De sekvenser jag undersökt är sekvenser där en hand rörs framför en bakgrund, som oftast är en ”lätt” bakgrund i den mening att den är till stora delar homogen och då ger upphov till få särdrag. Handens tillstånd växlar under sekvenserna mellan olika lägen, t.ex. knuten hand, pekande med ett finger eller alla fem fingrar utsträckta.

I de sekvenser jag tittat på varierar detektionsskalan och signifikansen väldigt lite från bild till bild. Den detekterade skalan varierar vanligtvis med mindre än 1%. Signifikansen varierar ibland med något över 10% i sekvenser på 10 bilder och vanligtvis under 6% från bild till bild. Då ofta flera särdrag har ungefär lika hög signifikans skiftar rankningsordningen mellan särdragen då deras signifikans fluktuerar. I de sekvenser jag tittat på brukar ett specifikt särdrags rankning hålla sig inom ett interval på 5 placeringar.

Även i x- och y-led var särdragen stabila, de låg hela tiden väl lokaliserade till den punkt i bilden där det objekt de representerade befann sig. Sammanfattningsvis kan man säga att särdragsdetektionen är tämligen stabil över tiden.

Kapitel 5

Realtidsprestanda vid särdragsdetektion i färgbilder

5.1 Realtidsimplementation

I ett tidigare examensarbete på CVAP utfört av Jani Niemenmaa (Niemenmaa 2000), som bygger på tidigare arbeten Grostabussiat (1997) och Lindeberg (1995), implementerades en hybridpyramidstruktur för att kunna göra särdragsdetektion med automatiskt skalval i realtid. Denna implementation har jag byggt vidare på och då främst lagt till möjligheter att använda färginformation. I avsnitt 2.5.3 finns en introduktion till hybridpyramider, mer utförlig beskrivning finns i ovan angivna referenser.

5.1.1 En tidigare implementation

Niemenmaa har skrivit en implementation av en hybridpyramid i C++. Den kan detektera blobbar, med automatiskt skalval, i stillbilder och presenterar sedan resultatet grafiskt genom att rita ut cirklar, representerande detekterade regioner, överlagrade på originalbilden. Viss tidtagningsinformation samt mer precis information om de detekterade särdragen ges också av programmet. Programmet kan använda pyramider med olika struktur när det gäller hur ofta subsamplingar ska ske i pyramiden. Detta styrs via indatafiler som genereras i förväg av Matlabscript. Blobbdetektionen sker endast i gråniväbilder.

5.1.2 Utförda utvidgningar av implementationen

De utvidgningar av Niemenmaas implementation jag har gjort när det gäller färginformation är följande:

- särdragsdetektion i flera färgkanaler
- den kombinerade blobbdetektorn $(A_{xx}+A_{yy})^2+(B_{xx}+B_{yy})^2+(C_{xx}+C_{yy})^2$, beskriven i avsnitt 3.1.1

- viktning av särdrag efter sökt färg genom att titta på färgen hos centrumpunkten, alla punkter i regionen eller x antal slumpvis valda punkter i regionen, se avsnitt 3.3 för utförligare beskrivning
- byte av färgrymd för en eller flera kanaler

Jag har också lagt till följande förbättringar:

- möjlighet till särdragsdetektion i videosekvenser
- möjlighet till särdragsdetektion i en realtidssituation, där bilder kontinuerligt tas från en kamera
- möjlighet till åsdetektion
- bättre normalisering av pyramiden, vilket möjliggör snabbare exekvering
- optimerat vissa delar av koden för bättre prestanda
- maximahantering, begränsning av utmatning till endast de N starkaste, utrensning av dubbletter, m.m.

Slutligen har jag skrivit ett litet program som utifrån en bild skapar ett histogram över färgvärdena i bilden. Histogrammet sparas på det format realtidsimplementationen använder för viktning av de detekterade särdragens signifikans.

5.2 Utvärdering av realtidsimplementationen

I figur 5.1 visas resultatet av blobbdetektion i bilden med leksakerna som förekom i figurerna 4.1, 4.2 och 4.3. Den hybridpyramid som har använts hade 30 nivåer, $\rho =$ 0,4 (subsampling sker var femte nivå), $1 \leq t \leq 3760$, bilden hade en upplösning på 128x128 pixlar. De operatoruttryck som användes var

$$t^2(L_{xx} + L_{yy})^2 \tag{5.1}$$

(blobbar en kanal i taget) och

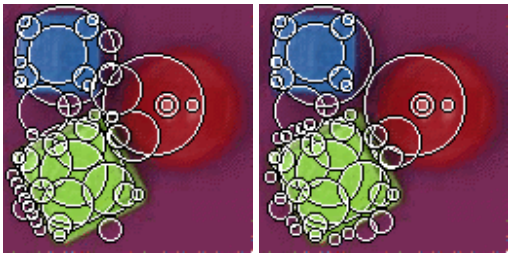
$$t^2((A_{xx} + A_{yy})^2 + (B_{xx} + B_{yy})^2 + (C_{xx} + C_{yy})^2) \tag{5.2}$$

(blobbar tre kanaler).

Precis som i skalrumsfallet detekterades inte bollen vid gråvådetektion utan bara vid detektion i färg. Man kan också notera att lokalisering, både i bilden och i skala som väntat var sämre i denna implementation än vid detektion i skalrum.

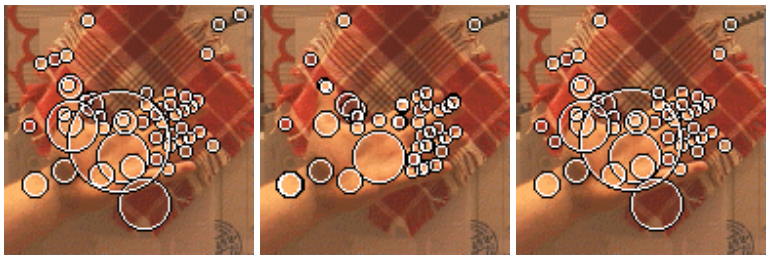
I figur 5.2 visas resultatet av blobbdetektion i samma bild som förekom i figurerna 4.4, 4.5 och 4.6. Detektionen har skett med samma parametrar som i exemplet ovan.

(a) (b) (c)

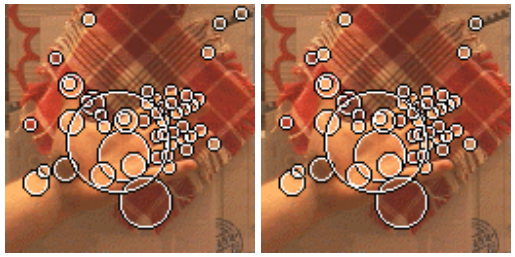


(d) (e)

Figur 5.1. De 50 mest signifikanta regionerna i bilden av leksaker, detekterade i realtidsimplementationen. Detektion har skett med olika metoder, (a) gråvådetektion, (b) separat i RGB-kanalerna, (c) separat i YUV-kanalerna, (d) den kombinerade operatorn i RGB samt (e) den kombinerade operatorn i YUV.



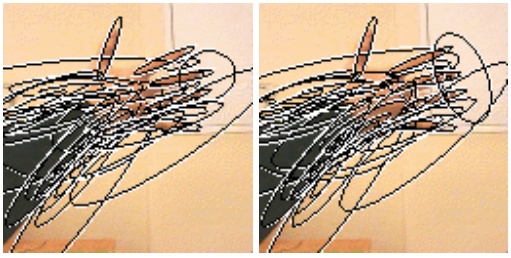
(a) (b) (c)



(d) (e)

Figur 5.2. De 50 mest signifikanta regionerna i bilden av en hand, detekterade i realtidsimplementationen. Detektion har skett med olika metoder, (a) gråvådetektion, (b) separat i RGB-kanalerna, (c) separat i YUV-kanalerna, (d) den kombinerade operatorn i RGB samt (e) den kombinerade operatorn i YUV.

(a) (b) (c)



(d) (e)

Figur 5.3. De 40 mest signifikanta åsarna i bilden av en hand, detekterade i realtidsimplementationen. Detektion har skett med olika metoder, (a) gråvådetektion, (b) separat i RGB-kanalerna, (c) separat i YUV-kanalerna, (d) den kombinerade heuristiska åsdetektorn i RGB samt (e) den kombinerade heuristiska åsdetektorn i YUV.

Här ser man att hybridpyramidimplementationen inte är riktigt lika bra på att detektera särdrag som detektion i skalrum, vilket leder till att lillfingertoppen inte detekteras.

I figur 5.3 visas resultatet av åsdetektion i bilden som förekom i 4.7, 4.8 och 4.9. Den hybridpyramid som använts är återigen 30 nivåer, $\rho = 0{,}4$ (subsampling var femte nivå), $1 \leq t \leq 3760$, bilden hade en upplösning på 128x128 pixlar. De operatoruttryck som användes var

$$\sqrt{t}^3((L_{xx} - L_{yy})^2 + 4L_{xy}^2) \tag{5.3}$$

(åsar en kanal i taget) och

$$\sqrt{t}^3((A_{xx} - A_{yy})^2 + 4A_{xy}^2 + (B_{xx} - B_{yy})^2 + 4B_{xy}^2 + (C_{xx} - C_{yy})^2 + 4C_{xy}^2) \tag{5.4}$$

(åsar tre kanaler). Vid åsdetektion i realtidsimplementationen har ingen hoplänkning av åsar skett.

Man kan här se att handens fingrar i samtliga fall detekterats väl, precis som i skalrumsfallet. Den svarta tröjärmen har hög kontrast mot omgivningen och ger därför upphov till många detekterade åsar på olika skalor. Man får svar för dels hela åsen och dels på båda sidor om kanterna mellan ärmen och bakgrunden.

5.3 Mätningar av tidsåtgången vid särdragsdetektion

5.3.1 Tidsåtgång vid olika metoder för särdragsdetektion

Vid undersökning av tidsåtgången vid de olika metoderna att detektera särdrag gjordes blobbdetektion på bilden i figur 5.4. Bildens dimensioner är 384x288 pixlar. Ingen viktning med avseende på sökt färg användes i dessa test. Detektion och utrensning av dubbletter, dvs maxima som överlappar väldigt mycket, gjordes inte, eftersom den delen är ganska ineffektivt implementerat. Detta skulle påverka detektion i flera kanaler orättvist mycket, eftersom man då hittar mycket fler maxima att undersöka. Detektionen skedde med en hybridpyramid med 30 nivåer, vilket är en hög pyramid. Subsampling skedde var femte nivå, $\rho = 0{,}4$. Skalparametern t ligger för denna pyramid i intervallet $1 \leq t \leq 3760$. Den särdragsdetektion som har använts är blobbdetektion.

Två olika typer av initialt tröskelvärde för signifikansen användes, dels ett lågt värde (10) där många maxima detekteras och dels ett högt värde (900) där ca. 50 maxima detekterades i gråvivå, ca 70 för den kombinerade operatorn och ca. 300 (varav många dubbletter) detekterades i de separata kanalerna. Med en hög tröskel går maximadetektionen mycket fortare än med en låg tröskel. I tabell 5.1 finns resultaten av tidtagningen sammanställda.

Detta exempel är medvetet valt för att ta lång tid, för att på så sätt tydliggöra skillnaderna mellan de olika metoderna. I tabell 5.2 finns ett exempel då snabbare särdragsdetektion gjorts. Där har en bild med dimensionerna 128x128 pixlar, en

Figur 5.4. De 50 mest signifikanta regionerna.

Moment	Tid gråvå (s)	Tid tre kanaler (s)	Tid kombinerad operator (s)
Initialisering	0,03	0,08	0,08
Skapa skalnivåerna	0,21	0,61	0,61
Derivateberäkningar	0,17	0,53	0,64
Maximadetektion	0,31 (0,1)	0,96 (0,3)	0,42 (0,1)
Maximafiltrering	0,01	0,02	0,01
Totalt	0,73 (0,52)	2,2 (1,5)	1,8 (1,5)

Tabell 5.1. Tidsåtgång för särdragsdetektion i hybridpyramid. Pyramiden har 30 nivåer, $\rho = 0{,}4$, skalparametern t ligger i intervallet $1 \leq t \leq 3760$, bilden har en upplösning på 384x288 pixlar. Två olika trösklar har använts, dels värdet 10 (lågt) och dels värdet 900 (högt). Tider inom parentes gäller det högre tröskelvärdet. Exemplet är medvetet valt för att ta lång tid för att tydliggöra skillnaderna mellan metoderna. Tar man en mindre bild, högre tröskel eller en pyramid med färre nivåer går det mycket fortare. Se även tabell 5.2 för ett exempel på snabbare detektion.

Moment	Tid gråvå (s)	Tid tre kanaler (s)	Tid kombinerad operator (s)
Initialisering	0,01	0,02	0,02
Skapa skalnivåerna	0,005	0,03	0,03
Derivateberäkningar	0,01	0,03	0,03
Maximadetektion	0,005	0,01	0,005
Maximafiltrering	0,0	0,0	0,0
Totalt	0,03	0,09	0,09

Tabell 5.2. Tidsåtgång för särdragsdetektion i hybridpyramid. Pyramiden har 11 nivåer, $\rho = 0{,}8$, skalparametern t ligger i intervallet $1 \leq t \leq 5470$, bilden har en upplösning på 128x128 pixlar, en tröskel på värdet 1000 (högt) har använts.

pyramid med 11 nivåer där subsampling sker vid två nivåer av tre, $\rho = 0{,}8$, $1 \leq t \leq 5470$ och en tröskel på 1000 (högt) använts.

De olika momenten jag delat in förloppet i är följande: initialisering av pyramidstrukturen, skapa skalnivåerna, derivateberäkningar, maximadetektion och maximafiltrering.

- Initialiseringsfasen är det som sker först och innefattar allokering av minne för pyramiden, konvertering av bilden till pyramidens format, initialisering av diverse hjälpfunktioner, t.ex. maximahanteringen. Det som sker i denna fas görs bara en gång, oavsett hur många bilder man ska detektera särdrag i. Ska man arbeta med långa videosekvenser påverkas alltså inte det antal bilder man kan behandla per sekund av tiden denna fas tar.

- Skapandet av skalnivåerna är den fas då man vid varje nivå antingen faltar bilden med en filterfunktion eller subsamplar bilden för att skapa nästa nivås bild.

- Derivateberäkningar är då man på varje nivå i pyramiden beräknar $(\nabla^2)^2$.

- Maximadetektion sker genom att man hittar punkter i pyramiden som är maxima i en 3D-omgivning.

- Maximafiltrering består i detta fall av att detekterade maxima sorteras med avseende på signifikans.

I tabell 5.1 syns att detektion i tre kanaler separat som väntat tar ungefär tre gånger mer tid än detektion i gråvåfallet. Detektion med den kombinerade operatorn tar något mindre tid, vilket främst beror på att maximadetektion bara sker i en pyramid, till skillnad mot i tre separata kanaler då det sker i tre pyramider. Att det inte går lika fort som i gråvåfallet beror på att det finns fler maxima att detektera eftersom även kontrast i färg ger upphov till särdrag (maxima), inte bara kontrast i intensitet som i gråvåfallet.

Man ser också att man vinner mycket tid på att sätta en hög tröskel för maximadetektionen, är tröskeln tillräckligt hög går det lika fort att göra särdragsdetektion i tre separata kanaler som med den kombinerade operatorn, vars främsta fördel är att maximadetektionen går snabbare än för de separata kanalerna. Risken med en hög tröskel är att man sätter den för högt och då missar intressanta särdrag. Signifikansen hos särdragen varierar mellan olika bilder, beroende på belysningsförhållanden och färgförhållanden mellan olika delar i bilden. Detta gör att det kan vara svårt att sätta en lagom hög tröskel. Man kan byta tröskel under detektionens gång och på så sätt få en adaptiv tröskling. Om man t.ex. bara ska ta fram de 200 mest signifikanta särdragen kan man efter att ha detekterat 200 särdrag med en ganska låg tröskel höja tröskeln, eftersom alla särdrag bland de 200 mest signifikanta måste ha en signifikans högre en det nuvarande särdrag nummer 200. På så sätt kan man hela tiden öka sin tröskel.

Maximafiltreringen tar längre tid i det fall då man detekterar särdrag separat i tre kanaler, vilket beror på att man då får mycket fler särdrag. Detta beror dels på att man har bättre kontrast och på så sätt kan upptäcka fler särdrag och framför allt på att de flesta särdrag detekteras i mer än en kanal.

5.3.2 Tidsåtgång vid viktning av särdrags signifikans

De tre metoder som undersökts är att ta centrumpunkten, ett antal slumpvis valda punkter i regionen och samtliga punkter i regionen och sedan kontrollera dessas värden i ett histogram, se avsnitt 3.3. Tiderna i tabell 5.3 är uppmätta vid särdragsdetektion på bilden i figur 5.4, som har en upplösning på 384x288 pixlar. Det är totalt 1280 detekterade regioner som ska viktas med sin likhet till hudfärg. Särdragsdetektion utan viktning av särdragens signifikans tar ca. 0,7 sekunder med de inställningar som användes, vilket var endast gråvådetektion i en hybridpyramid med 30 nivåer, $\rho = 0{,}4$, $1 \leq t \leq 3760$.

Metod	Tid (s)
Alla punkter	0,09
Centrumpunkten	0,005
5 slumpvis valda	0,02
10 slumpvis valda	0,03
50 slumpvis valda	0,1

Tabell 5.3. Tidsåtgång för viktning av signifikans m.a.p. likhet med hudfärg. Total 1280 regioner, de flesta små, har viktats. Då större delen av regionerna är små, en typisk region har ca. 80 punkter, tar det längre tid att undersöka 50 slumpvis valda punkter än att undersöka alla punkter. Detta beror på att det tar längre tid att slumpvis välja en punkt än att gå igenom punkterna i en förutbestämd ordning.

Metoden att ta ett slumpvis urval av punkterna i regionen tar längre tid ju fler punkter man vill undersöka, men blir också mer robust ju fler punkter man undersöker. Tar man tillräckligt många slumpvis valda punkter kommer det ta längre tid än att undersöka samtliga punkter i regionerna. Vid hur många punkter gränsen går beror på bl.a. hur stora regionerna är och hur lång tid det tar att slumpvis välja en punkt. I figur 5.5 visas vilken inverkan de olika metoderna har på rankningen av särdragen efter deras signifikans.

Då man använder alla punkter eller många slumpvis valda punkter blir den extra tidsåtgången för viktningen märkbar, men om man använder endast ett fåtal slumpvis valda punkter eller enbart centrumpunkten blir tidsskillnaden mellan särdragsdetektion med viktning och särdragsdetektion utan viktning försumbar.

Detta varierar till viss del med under vilka andra förhållanden särdragsdetektionen sker. Har man en pyramid där subsampling sker ofta går särdragsdetektionen fortare, likaså om man har lägre upplösning på bilden man detekterar i. Om man t.ex. har en pyramid där subsampling sker vid ungefär två nivåer av tre i pyramiden

(a) (b)

(c) (d)

(e) (f)

Figur 5.5. De 50 mest signifikanta regionerna efter viktning med avseende på likhet med färg. Viktning har skett med olika metoder, (a) ingen viktning, (b) alla punkter i regionen, (c) endast centrumpunkten, (d) 5 slumpvis valda punkter, (e) 10 slumpvis valda punkter och (f) 50 slumpvis valda punkter.

och en bild med upplösning på 128x128 pixlar så tar motsvarande särdragsdetektion bara 0,03 sekunder. I det fallet skulle ju även 0,005 sekunder vara en märkbar förändring, men prestandaskillnaden blir ändå inte så stor. Då man gör snabbare särdragsdetektion får man i regel mycket färre särdrag (detektionsförmågan blir sämre), vilket gör att viktning också går fortare. Även vid det snabbare detektionsförfarandet var tidsskillnaden med och utan viktning försumbar för de snabba viktningsmetoderna.

Det bör tilläggas att det tar rätt lång tid att läsa in och packa upp histogrammet i den form jag använder det. Totalt tar minnesallokering, inläsning och uppackning ca. 0,7 sekunder. Detta behöver bara göras en gång vid initialiseringen, så det påverkar inte det antal bilder per sekund man kan behandla. Ska man bara behandla en bild blir tidsskillnaden märkbar, men en extra väntetid på 0,7 sekunder är inte så farligt.

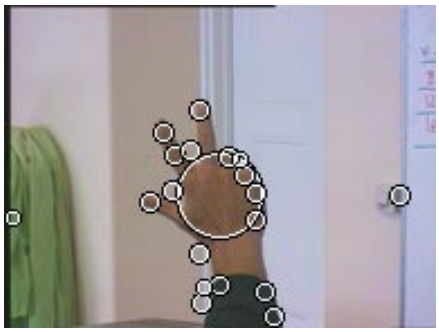
5.3.3 Möjliga prestanda

Då realtidsimplementationen kan använda hybridpyramider med olika egenskaper kan snabbheten hos detektionen variera beroende på vilken kvalitet på särdragsdetektionen som eftersträvas. Låg kvalitet möjliggör snabb särdragsdetektion medan hög kvalitet tar längre tid.

För att undersöka hur snabb detektion man kan uppnå och ändå behålla ”tillräckligt bra” kvalitet på särdragsdetektionen provkördes realtidsimplementationen på en videosekvens av en hand. Jag prövade endast blobbdetektion, med användning av färg. Färginformationen användes till viktning av signifikansen hos särdragen med hur lika hudfärg de regioner i bilden var som representerade särdragen. Endast centrumpunkten i regionen undersökters. Färginformationen användes också till att förbättra kontrasten i bilden genom att den nya särdragsdetektorn från avsnitt 3.1.1 användes.

Mitt kriterium för ”tillräckligt bra” var att handflatan och fingertoppar på utsträckta fingrar skulle detekteras med skala och position motsvarande deras storlek och position i bilden. Värdet på den använda tröskeln och hur ”snabb” hybridpyramid (många subsamplingssteg ger snabb pyramid) som användes var de parametrar som varierades. De bilder som ingick i videosekvensen hade en upplösning på 192x144 pixlar. Vid denna upplösning är handens delar tydligt urskiljbara och särdragsdetektion ger bra resultat. Ett exempel på en bildruta i sekvensen och detekterade särdrag i denna finns i figur 5.6. Videosekvensen bestod av 256 bilder.

Den snabbaste detektionen jag tyckte var tillräckligt bra använde en pyramid där $\rho = 0{,}8$, vilket innebär att subsampling sker vid två nivåer av tre. Pyramiden hade 10 nivåer, $1 \leq t \leq 1639$. En ganska hög tröskel (600) för signifikansen användes. Då behandlades i snitt 8,3 bilder per sekund i sekvensen. Den dator som användes var en PC med dubbla Pentium III 550 MHz processorer (dock körs implementationen endast på en processor) och 512 megabyte internminne, vilket i dagsläget är en ganska snabb, men inte en otroligt snabb, persondator.



Figur 5.6. De 20 mest signifikanta särdragen vid snabb särdragsdetektion i en videosekvens. Upplösningen i bilden är 192x144 pixlar.

Samma inställningar ger med gråvådetektion 14 bilder per sekund. Viktning av signifikansen ger ingen märkbar skillnad i tidsåtgång.

8,3 bilder per sekund är ganska bra prestanda, men något högre vore bra för att interaktion (t.ex. genom gester detekterade på detta sätt) inte ska upplevas som störande långsam. Det går lätt att parallellisera beräkningsarbetet till flera processorer för att få bättre prestanda, flera processorer kan t.ex. arbeta på var sin färgkanal (skapande av skalnivåer och beräkning av derivator) eller ta var sin bild i en sekvens. Behöver man bättre prestanda finns det mycket snabbare maskiner att få tag på redan i dagsläget och utveckling på datorområdet går fort framåt. Realtidsimplementationen är inte heller väldigt effektivt implementerad, utan det går förmodligen att göra ganska stora optimeringar av koden för att få bättre prestanda.

Ytterligare ett sätt att snabba upp särdragsdetektionen är att arbeta med mindre bildstorlek. Halverar man upplösningen i x- och y-led går särdragsdetektionen ungefär fyra gånger snabbare. Dock försvinner små detaljer i bilder då man förminskar dem. Särdragsdetektion med samma inställningar som ovan prövades också på sekvenser med lägre upplösning på bilderna. Vid halverad upplösning (96x72 pixlar) behandlades 30 bilder per sekund vid särdragsdetektion med den nya särdragsdetektorn i färg och 51 bilder per sekund med gråvådetektion. Fingrar och hand gick fortfarande att detektera, men lokaliseringen, både i position och i skala var sämre. Ett exempel på resultat finns i figur 5.7.

Vid särdragsdetektion då både blobbar och åsar detekteras tar detektionen ca. 50% längre tid än vid endast blobbdetektion. Detta innebär att antalet bilder man kan behandla per sekund minskar med ungefär en tredjedel om man även vill göra åsdetektion.

Sammanfattningsvis kan man säga att realtidsimplementationen inte riktigt når upp till önskvärda realtidsprestanda vid detektion i flera färgkanaler, dock finns möjligheter att uppnå detta genom snabbare hårdvara eller optimeringar av koden. Realtidsimplementationen klarar att göra gråvådetektion, med viktning av detekterade särdrags signifikans med deras likhet med hudfärg, i realtid.

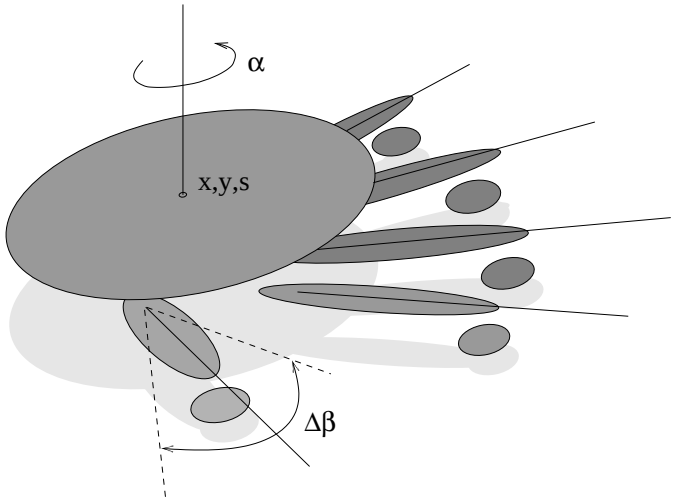
Figur 5.7. De 20 mest signifikanta särdragen vid snabb särdragsdetektion i en videosekvens. Upplösningen i bilden är 96x72 pixlar.

5.4 Användning av detekterade särdrag för handföljning

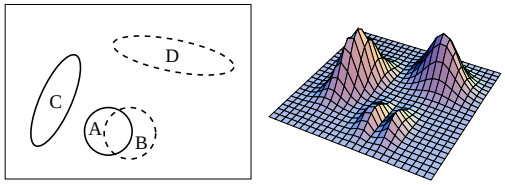
Ett exempel på tillämpning av särdragsdetektion är handföljning. I Laptev och Lindeberg (2000) beskrivs en metod för handföljning där handens position, riktning och tillstånd detekteras. Med tillstånd menas här hur man håller fingrarna. Exempel på tillstånd handen kan befinna sig i är: handen är knuten, handen pekar med ett finger, handen spretar med alla fem fingrar.

Metoden går i korthet ut på att särdrag detekteras i en bild av en hand. Sedan försöker man anpassa en modell av en hand till de detekterade särdragen. Handmodellen är en hierarkisk modell där man på första nivån har en blobb som representerar handflatan. På nästa nivå har man åsar som representerar fingrarna. Sist i modellen kommer blobbar, som representerar fingertopparna. I modellen ingår samband som ska gälla mellan de olika delarna. Fingeråsarna och fingertoppsblobbarna ska befinna sig på ungefär samma skala och handflatan ska befinna sig på en större skala än dessa. Vissa rumsliga samband ingår också i modellen, fingeråsarna ska i ena änden ansluta till handflatan och fingertoppsblobbarna ska ligga i andra änden av åsen. I figur 5.8 finns en illustration av handmodellen.

Då man har detekterat särdragen i bilden beräknar man vilken position, orientering och tillstånd hos handmodellen som bäst stämmer överens med dessa. Detta görs genom att beräkna olikheten mellan detekterade särdrag och de särdrag en viss modell av handen ger upphov till. Den modell som är minst olik detekterade data är då den modell man vill ha. Olikheten mellan två uppsättningar särdrag (de detekterade och de som skulle genereras av den modell av handen man undersöker) beräknas genom att ta kvadraten av differensen mellan gaussfunktioner som motsvarar särdragen, se figur 5.9. Särdrag som överlappar särdrag i den andra mängden mycket ger då upphov till låg olikhet medan särdrag som inte överlappar något särdrag i den andra mängden ger upphov till stor olikhet. För en utförligare genomgång av metoden hänvisas till Laptev och Lindeberg (2000).



Figur 5.8. Handmodell för handföljning. Figuren är tagen ur Laptev och Lindeberg (2000).



(a) De regioner särdragen motsvarar

(b) Beräknad olikhet

Figur 5.9. Beräkning av olikheten mellan två uppsättningar särdrag, t.ex. detekterade särdrag (A och C) och särdrag genererade av modellen (B och D). Figurerna är tagna ur Laptev och Lindeberg (2000).

Ivan Laptev på CVAP har implementerat denna metod för handföljning. Implementationen arbetar med videosekvenser där en användare rör sin hand framför kameran. Programmet detekterar handens position i bilden, åt vilket håll handen är riktad och handens tillstånd.

För att avgöra handens och fingrarnas positioner använder programmet särdragsdetektion. Särdragsdetektionen sker genom grånivådetektion med automatiskt skalval i skalrumsrepresentation, vilket är tidskrävande. Särdragsdetektionen sker ej i realtid, men själva handföljningen går i realtid om man använder i förväg detekterade särdrag. Hur hybridpyramidimplementationen kan användas dels för att snabba upp processen och dels för att lägga till användandet av färginformationen i videosekvenserna har undersökts och resultaten presenteras i avsnitt 5.5.

Handföljning kan användas för människa-maskininteraktion. Handföljningsimplementationen har t.ex. använts för att styra ett enkelt ritprogram. Handens tillstånd talar om vilken operation som ska utföras, t.ex. kan ”peka med ett finger” innebära att man vill rita med fingertoppen och ”knuten hand” innebära att man vill flytta ett ritat objekt.

5.5 Resultat vid användning av detekterade särdrag för handföljning

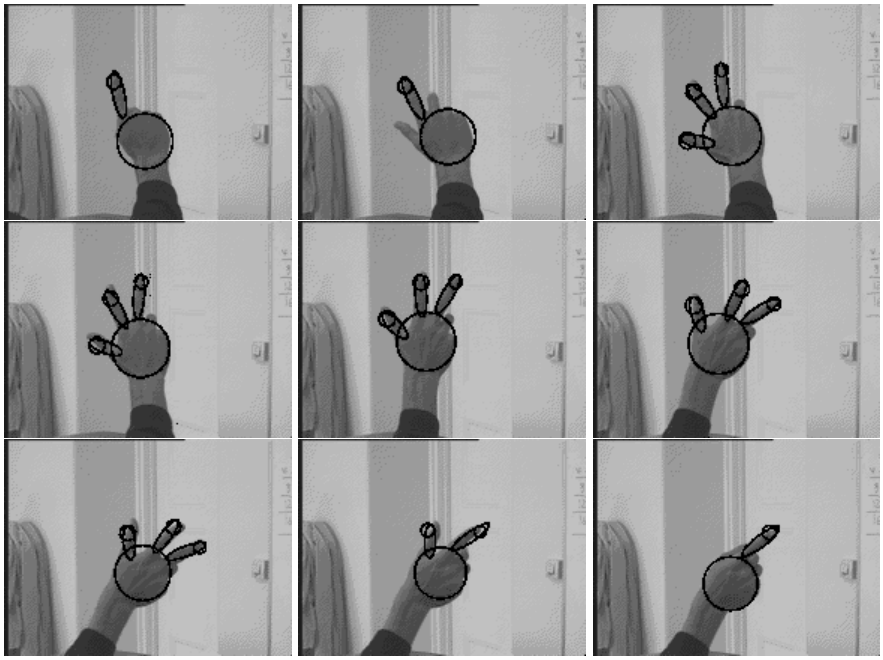
För att utvärdera hur väl realtidsimplementationen i avsnitt 5.1 fungerar i en konkret tillämpning kördes handföljningsimplementationen med särdrag detekterade av realtidsimplementationen som indata.

Först kördes handföljning med endast blobbar som indata, eftersom realtidsimplementationen då endast detekterade blobbar. Det visade sig att handföljningen då gick dåligt, trots att de detekterade särdragen såg bra ut. Detta berodde på att enbart blobbar inte är tillräckligt diskriminativa för att kunna utföra handföljning. Därför utvidgades realtidsimplementationen med åsdetektion.

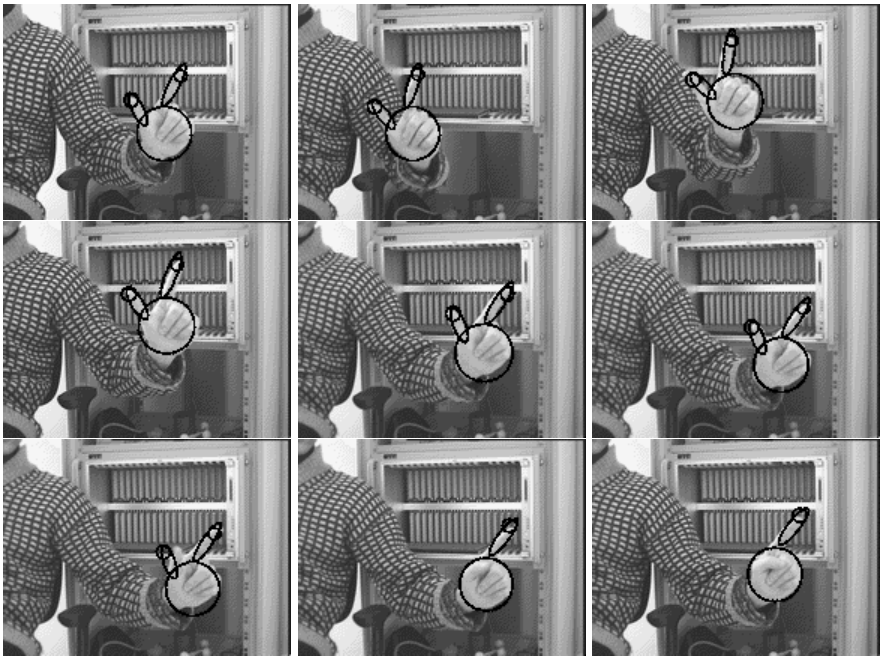
Två olika videosekvenser har prövats. Den ena sekvensen är en ”lätt” sekvens, där en hand rör sig framför en bakgrund som skiljer sig från handen i intensitet och som har få detaljer (dvs ger upphov till få särdrag i bakgrunden). Ett antal bildrutor från sekvensen finns i figur 5.10, där även resultatet av handföljning i sekvensen ritats ut. I denna sekvens fungerar gråvådetektionen som utförs i vanliga fall mycket bra.

Den andra sekvensen som prövades var en ”svår” sekvens, där en hand rörs framför en detaljrik bakgrund (många detekterade särdrag i bakgrunden) där gråvåkontrasten tidvis är dålig mellan delar av hand och bakgrund. Ett exempel på hur denna sekvens ser ut finns i figur 5.11, med resultat av handföljning på särdrag detekterade i färg, och i figur 5.12, med särdrag detekterade i gråvå. I denna sekvens fungerar den gråvådetektion som sker i vanliga fall dåligt.

Vid särdragsdetektionen användes först den kombinerade blobbdetektorn beskriven i avsnitt 3.1.1 i vanliga RGB-färgmodellen. Detta fungerade bra på sekvensen med enkel bakgrund, men mindre bra på sekvensen med detaljrik bak-

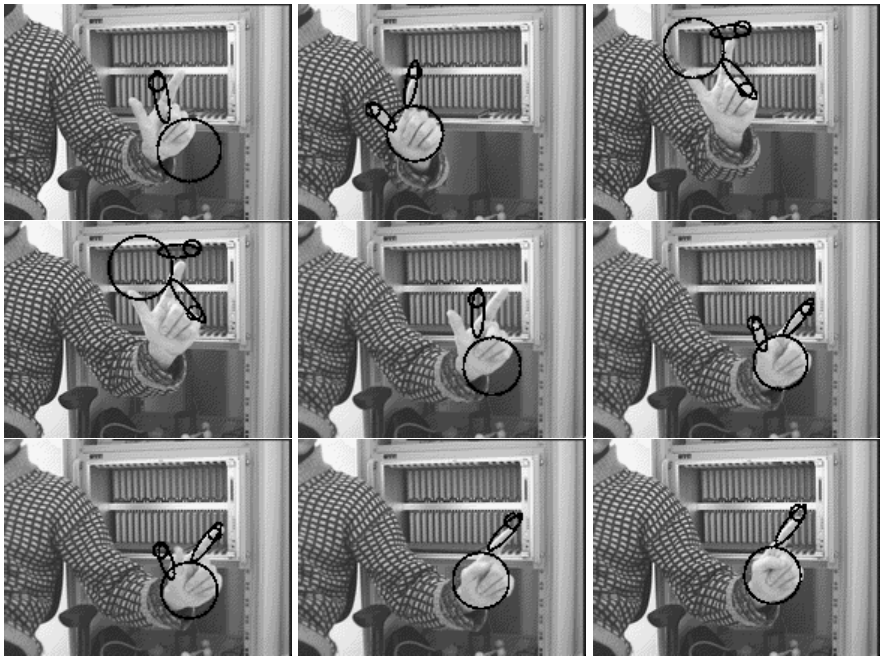


Figur 5.10. Några bildrutor från en videosekvens med en hand mot en enkel bakgrund. Resultatet av handföljningsalgoritmen har ritats ut ovanpå en grånivåversion av bildrutor från videosekvensen.



Figur 5.11. Några bildrutor från en videosekvens med en hand mot en komplicerad bakgrund. Resultatet av handföljningsalgoritmen med särdragsdetektion i färg har ritats ut ovanpå en grånivåversion av bildrutor från videosekvensen.

grund. Därefter prövades blobbdetektion separat i de olika färgkanalerna i YIQ-färgmodellen på den senare sekvensen, vilket fungerade bättre. Åsdetektionen skedde separat i de tre YIQ-kanalerna. Ingen hoplänkning av detekterade åsar skedde.



Figur 5.12. Några bildrutor från en videosekvens med en hand mot en komplicerad bakgrund. Resultatet av handföljningsalgoritmen med gråniväsärdragsdetektion har ritats ut ovanpå en gråniväversion av bildrutor från videosekvensen.

I sekvensen med enkel bakgrund fungerade handföljningen mycket bra med de detekterade särdragen, handföljningen detekterade handens och fingrarnas position och riktning samt handens tillstånd i hela sekvensen. Denna sekvens fungerar bra även med grånivådetektion, men försöket visar att det är möjligt att med realtidsimplementationen detektera särdrag som är tillräckligt bra för handföljning.

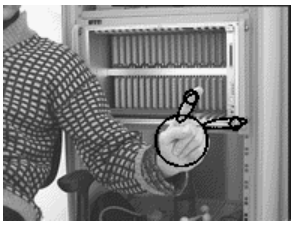
I sekvensen med detaljrik bakgrund gick det bra i vissa delar av sekvensen och dåligt i andra. Det största problemet i denna sekvens var att det fanns så mycket särdrag i bakgrunden att det var svårt att skilja ut handens särdrag från de andra. Realtidsimplementationen lyckades dock detektera handens särdrag, vilket inte den tidigare använda särdragsdetektionen lyckats med, så färginformationen gav intressant information.

Då problemet låg i att skilja ut intressanta särdrag från särdrag i bakgrunden prövades även särdragsdetektion med viktning av de detekterade särdragens signifikans efter hur lika hudfärg de områden i bilden särdragen svarade mot var. Detta gjorde att handens särdrag, vilka är de särdrag man är intresserad av, oftast var de mest signifikanta särdragen. Dock var kontrasten tidvis mycket dålig mellan delar

av handen och bakgrunden, t.ex. när pekfingertoppen befann sig i den position den övre vänstra bilden i figur 5.11. Detta gjorde att signifikansen var låg även efter viktningen i de fallen och de särdragen hamnade då under några bildrutor långt ned i rankningen.

När handföljning gjordes på särdrag detekterade i färg och viktade med likhet med hudfärg fungerade även sekvensen med detaljrik bakgrund bra nästan hela tiden. Vid tre tillfällen detekterades inte handen korrekt, ett tillfälle varade endast en bildruta (handens riktning felaktig), ett tillfälle varade två bildrutor (fel handtillstånd) och det värsta tillfället varade 10 bildrutor (fel handtillstånd). Då sekvensen totalt är 512 bilrutor lång är detta förhållandevis liten del av sekvensen. Ett exempel på hur det kan se ut när handföljningen misslyckas finns i figur 5.13. Där har särdrag i bakgrunden felaktigt antagits tillhöra handen.

I figur 5.12 finns exempel på resultatet av handföljning, i sekvensen med detaljrik bakgrund, då särdragsdetektionen har skett endast i gråvå. Då fungerar sekvensen betydligt sämre, handen detekteras korrekt i totalt ca. 70 av de 512 bildrutorna. Detta innebär att färginnehållet i bilderna gav intressant information och att särdragsdetektion i färg innebar en rejäl förbättring.



Figur 5.13. Exempel på en bildruta då handföljningen misslyckats och en del av bakgrunden har antagits tillhöra handen. Handföljning har gjorts med särdrag detekterade i färg och med viktning av särdragens signifikans med deras likhet med hudfärg.

De problem som förekommer vid användande av färginformation är fortfarande att det ibland är svårt att skilja ut vilka särdrag som tillhör bakgrunden och vilka som tillhör handen. Detta skulle förmodligen kunna åtgärdas genom att istället för att vikta särdragens signifikans med deras likhet med hudfärg ange hur sannolikt det är att ett visst särdrag är hudfärgat, men detta har vi inte undersökt.

Ytterligare tre, lite kortare, videosekvenser har prövats. I dessa var bakgrunden något mer komplicerad än i sekvensen med enkel bakgrund, men kontrasten mellan hand och bakgrund var hög. Både blobbar och åsar detekterades separat i de olika färgkanalerna i YIQ-färgmodellen. Resultatet var bra i alla tre sekvenserna. I appendix A ges exempel på resultatet av handföljning i dessa sekvenser.

Sammanfattningsvis kan man säga att särdrag detekterade av realtidsimplementationen mycket väl kan användas för handföljning och att det i vissa fall kan ge bättre resultat än den tidigare metoden tack vare att även färginformationen

används. De fall då färginformationen är viktigt för att förbättra prestanda är då bakgrunden är komplicerad, dels om det finns mycket särdrag i bakgrunden och dels om kontrasten mot bakgrunden är dålig. Detta är vanligt i ”naturliga” miljöer, t.ex. kontorsmiljöer.

Kapitel 6

Avslutande ord

6.1 Sammanfattning

Olika möjligheter att använda färginformation vid särdragsdetektion i bilder har undersökts. Två olika sätt att använda färginformationen har prövats, dels användning av färg för att få bättre kontrast mellan olika regioner i bilder och dels användning av detekterade särdrags färg för att enkelt filtrera ut de relevanta särdragen. En realtidsimplementation av särdragsdetektion (blobb- och åsdetektion) med användning av färginformation har tagits fram. Denna implementation har prövats i en konkret tillämpning.

I avsnitt 4 visades valet av färgmodells inverkan vid särdragsdetektion. Det visade sig att vissa färgmodeller var bättre än andra, främst YIQ, YUV och den gaussiska färgmodellen var bra. Där visades också att en ny särdragsdetektor, beskriven i avsnitt 3.1.1, för färgbilder fungerar bättre än traditionell gråvådetektion i fall där gråvåkontrasten är dålig. Detektion i de olika färgkanalerna separat fungerade också bra, men är något mer tidskrävande.

Viktning av särdragens signifikans med avseende på deras likhet med en sökt färg visades vara en enkel och kraftfull metod att sålla bort särdrag detekterade i bakgrunden.

Undersökningar av särdragsdetektion i videosekvenser visade att särdragsdetektion är tämligen stabilt över tiden, dvs särdragen detekteras med liknande egenskaper i närliggande bilder i sekvenserna.

Utvärderingen av realtidsimplementationen, som använder hybridpyramidrepresentation, visade att denna, som väntat, inte var lika bra som särdragsdetektion i skalrum. Särdrag detekterade av realtidsimplementationen visades dock vara tillräckligt bra för att kunna användas i en konkret tillämpning (handföljning). Användandet av färginformationen i bilder gav i vissa fall bättre resultat än särdrag detekterade med den tidigare använda metoden (detektion i skalrum på gråvåinformation, ej realtid).

Undersökningar av tidsåtgången vid olika typer av särdragsdetektion visade att det är möjligt att göra särdragsdetektion i realtid. Användande av färginformation

för att filtrera ut intressanta särdrag har mycket liten inverkan på tidsåtgången. Särdragsdetektion i flera färgkanaler tar betydligt mer tid än detektion endast i gråniv̊a, men detektion i tre kanaler kan ske på mindre än tre gånger den tid det tar i gråniv̊afallet.

Realtidsimplementationen klarar att göra gråniv̊adetektion i bilder tillräckligt snabbt för att klara realtidskrav. Även särdragsdetektion i flera färgkanaler går i realtid om upplösningen på bilderna som ska behandlas är tillräckligt låg och hybridpyramidens parameter ρ (hur ofta subsampling ska ske) är tillräckligt hög. $\rho = 0.8$ och en bildstorlek på ca. 100x100 pixlar gav möjlighet att detektera särdrag i många (ca. 20 vid både blobb- och åsdetektion i färg) bilder per sekund, men kvaliteten på detekterade särdrag var då på gränsen till för dålig. Sänks värdet på ρ till 0,5 blir kvaliteten på särdragen bättre.

Vid körning på den Pentium III 550 MHz processor som använts vid experimenten kan den nuvarande implementationen hantera 6 bilder, av storlek 192x144 pixlar, per sekund, med ganska bra kvalitet på detekterade särdrag, vid detektion av både åsar och blobbar. Med snabbare processor eller en effektivare implementation verkar det inte orimligt att effektiviteten ska kunna lyftas till genuina realtidsprestanda inom en snar framtid.

6.2 Slutsatser

Användande av färginformation i bilder för att få bättre kontrast vid särdragsdetektion kan ge mycket bättre resultat än särdragsdetektion baserad enbart på gråniv̊ainformation. Hur mycket bättre beror på hur bilden ser ut. Är det många regioner i bilden som har olika färg men snarlik intensitet får man stora förbättringar. Har man däremot en bild med färger som skiljer sig mycket i intensitet ger färginformationen förmodligen inte så mycket, eftersom resultatet då redan är bra vid gråniv̊adetektion.

Då man använder färginformationen för att få bättre kontrast tar det vanligtvis längre tid att göra särdragsdetektion än om man bara använder gråniv̊ainformationen. Använder man tre kanaler tar det ungefär tre gånger så mycket datorkraft, eller tre gånger så lång tid om man använder samma utrustning i båda fallen. Det går ju också som tidigare nämnts att under vissa förutsättningar använda informationen i alla tre färgkanalerna med mindre än tre gångers prestandaskillnad (avsnitt 5.3.1).

Man behöver inte använda alla tre färgkanalerna, man kan tänka sig att använda bara två kanaler, t.ex. intensitet och färgton, för att få bättre resultat än med enbart gråniv̊ainformation, men förmodligen något sämre än om man använder all färginformation. I vissa tillämpningar kanske det är lämpligt att istället för gråniv̊ainformationen använda en färgkanal, t.ex. om skillnaderna i gråniv̊a är små men skillnaderna i röd-grön nyans är stora. Vanligtvis är det dock intensiteten i bilden som ger mest information (och det är den man använder som gråniv̊abild).

Slutligen behöver en prestandaförlust på en faktor tre inte nödvändigtvis vara så farligt, väntar man några år har datorhårdvara, med nuvarande utvecklingstakt,

blivit mer än tre gånger snabbare. Behöver man bättre kontrast vid särdragsdetektionen kan färginformationen mycket väl vara värd att använda till detta.

Olika färgmodeller ger olika bra resultat vid särdragsdetektion i färg. Den gaussiska färgmodellen gav mycket bra resultat och har dessutom andra önskvärda egenskaper, som t.ex. att den har en intensitetskanal, att den har en teoretisk grund som tyder på att den ligger nära det mänskliga synsinnets funktion och att det är enkelt att översätta från RGB till denna. Detta gör att denna färgmodell är ett bra val vid särdragsdetektion i färgbilder.

Även YIQ och YUV fungerade bra vid särdragsdetektion. Båda modellerna har en intensitetskanal och det är enkelt att översätta från RGB till dessa modeller. Detta gör att även dessa modeller kan vara ett bra val av färgmodell.

Har man mycket hårda prestandakrav kan man använda RGB (vilket förmodligen är det format indata har) för att undvika extra beräkningar. RGB fungerar ganska bra, om än inte lika bra som de tidigare nämnda färgmodellerna.

När det gäller viktning av särdrags signifikans med deras likhet med en sökt färg är det en kraftfull metod att på ett enkelt sätt få fram vilka särdrag som är intressanta. Då prestandaförlusten vid viktning av särdrag är mycket liten är detta en bra metod i de fall då man vet att de särdrag man söker har en viss färg. Då man söker särdrag som kan ha många olika färger eller en färg som inte skiljer sig speciellt mycket från bakgrundsobjektens färg har man naturligt nog inte någon nytta av denna metod.

6.3 Framtida möjliga utvidgningar

Inom detta område finns det många möjligheter till utvidgningar. När det gäller användande av färginformationen för att få bättre kontrast finns det många andra typer av särdrag att undersöka när det gäller en operator som utnyttjar färginformationen. Det är dock enkelt att utvidga nuvarande metoder genom särdragsdetektion i de olika färgkanalerna separat. Även för de särdragstyper jag tittat på kan det vara givande att försöka skapa nya operatorer som utnyttjar färginformationen.

När det gäller viktning av särdrags signifikans efter likhet med sökt färg, främst då hudfärg, vore det intressant att titta på andra färgmodeller. Det man kan undersöka är t.ex. om vissa färgmodeller är mer diskriminativa och om man kan använda färre färgkanaler.

Litteraturförteckning

Ahmad, S. (1995). A useable real-time 3d hand tracker, *Proceedings of the 28th IEEE Asilomar Conference on Signals, Systems and Computers, Pacific Grove*, IEEE Computer Society Press, sid. 1257–1261.

Androutsos, P., Androutsos, D., Plataniotis, K. N. och Venetsanopoulos, A. N. (1997). Subjective analysis of edge detectors in color image processing, *Proceedings of ICIAP'97, 9th International Conference On Image Analysis And Processing, Florence*, Vol. 1310 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 119–126.

Angelopoulou, E. (2000). Objective colour from multispectral imaging, *Proceedings of the Sixth European Conference on Computer Vision, Dublin*, Vol. 1842 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 359–374.

Barnard, K. (2000). Improvements to gamut masiding colour constancy algorithms, *Proceedings of the Sixth European Conference on Computer Vision, Dublin*, Vol. 1842 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 390–403.

Barnard, K., Martin, L. och Funt, B. (2000). Colour by correlation in a three-dimensional colour space, *Proceedings of the Sixth European Conference on Computer Vision, Dublin*, Vol. 1842 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 375–389.

Brainard, D. H. och Freeman, W. T. (1997). Bayesian color constancy, *Journal of the Optical Society of America* **14**(7): 1393–1411.

Bretzner, L. och Lindeberg, T. (2000). Qualitative multi-scale feature hierarchies for object tracking, *Journal of Visual Communication and Image Representation* **11**: 115–129.

Burt, P. J. och Adelson, E. H. (1983). The Laplacian pyramid as a compact image code, *IEEE Trans. Communications* **9:4**: 532–540.

Cumani, A. (1989). Edge detection in multispectral images, *Technical report*, Istituto Elettrotecnico Nazionale "Galileo Ferraris", Torino, Italy.

Cumani, A. (1991). Efficient contour extraction in color images, *Technical report*, Istituto Elettrotecnico Nazionale "Galileo Ferraris", Torino, Italy.

Finlayson, G. D. (1996). Color in perspective, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **18**(10): 1034–1038.

Finlayson, G. D. och Schaefer, G. (2000). Constrained dichromatic colour constancy, *Proceedings of the Sixth European Conference on Computer Vision, Dublin*, Vol. 1842 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 342–358.

Forsyth, D. A. och Fleck, M. (1996). Finding naked people, *European Conference on Computer Vision, Cambridge*, Vol. 1065 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 592–602.

Funt, B. V. och Finlayson, G. D. (1995). Color constant color indexing, *IEEE Transacions on Pattern Analysis and Machine Intelligence* **17**(5): 522–529.

Gauch, J. M. (1998). Segmentation and edge detection, *i* Sangwine och Horne (1998), kapitel 9, sid. 163–187.

Geusebroek, J.-M., Dev, A., van den Boomgaard, R., Smeulders, A. W. M., Cornelissen, F. och Geerts, H. (1999). Color invariant edge detection, *Second International Conference on Scale-Space Theories in Computer Vision, Corfu*, Vol. 1682 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 459–464.

Geusebroek, J.-M., Smeulders, A. W. M. och van den Boomgaard, R. (2000). Measurement of color invariants, *Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head Island*, IEEE Computer Society Press, sid. 50–57.

Geusebroek, J.-M., van den Boomgaard, R., Smeulders, A. W. M. och Dev, A. (2000). Color and scale: The spatial structure of color images, *Proceedings of the Sixth European Conference on Computer Vision, Dublin*, Vol. 1842 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 331–341.

Gevers, T. och Smeulders, A. W. M. (1996). A comparative study of several color models for color image invariant retrieval, *Proceedings of the First International Workshop on Image Databases & Multimedia Search, Amsterdam*, Amsterdam University Press, sid. 17–26.

Gevers, T. och Smeulders, A. W. M. (1999). Color based object recognition, *Pattern Recognition* **32**: 453–464.

Gonzalez, R. C. och Woods, R. E. (1992). *Digital Image Processing*, Addison-Wesley.

Grostabussiat, P. (1997). *On Hybrid Multi-Scale Representations*, Licentiate dissertation, Dept. of Numerical Analysis and Computing Science, KTH (Royal Institute of Technology), Stockholm, Sweden.

Hall, D., de Verdière, V. C. och Crowley, J. L. (2000). Object recognition using coloured receptive fields, *Proceedings of the Sixth European Conference on Computer Vision, Dublin*, Vol. 1842 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 164–177.

Jähne, B. (1999). Hyperspectral and color imaging, *i* B. Jähne, H. Haußecker och P. Geißler (eds), *Handbook of Computer Vision and Asidlications*, Academic Press, kapitel 11, pp. 309–320.

Kjeldsen, R. och Kender, J. (1996). Finding skin in color images, *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, Killington*, IEEE Computer Society Press, sid. 312–317.

Krishnamoorthi, R. och Bhattacharyya, P. (1997). Color edge detection using orthogonal polynomials, *Proceedings of the Third Asian Conference on Computer Vision, Hong Kong*, Vol. 1351 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 590–597.

Laptev, I. och Lindeberg, T. (2000). Tracking of multi-state hand models using particle filtering and a hierarchy of multi-scale image features, *Technical Report ISRN KTH/NA/P--00/12--SE*, Dept. of Numerical Analysis and Computing Science, KTH (Royal Institute of Technology), Stockholm, Sweden.

Lindeberg, T. (1994). *Scale-Space Theory in Computer Vision*, Kluwer, kapitel 11.2.

Lindeberg, T. (1995). On hybrid multi-scale image representation, *Technical report*, Dept. of Numerical Analysis and Computing Science, KTH (Royal Institute of Technology), Stockholm, Sweden. Opublicerat manuskript.

Lindeberg, T. (1998a). Feature detection with automatic scale selection, *International Journal of Computer Vision* **30**(2): 77–116.

Lindeberg, T. (1998b). Principles for automatic scale selection, *Technical Report ISRN KTH/NA/P--98/14--SE*, Dept. of Numerical Analysis and Computing Science, KTH (Royal Institute of Technology), Stockholm, Sweden. I: B. Jähne (et al., eds), Handbook on Computer Vision and Asidlications, volume 2, pp 239-274, Academic Press, Boston, USA.

Lindeberg, T. (1999). Automatic scale selection as a pre-processing stage for interpreting the visual world, *Technical report*, Dept. of Numerical Analysis and Computing Science, KTH (Royal Institute of Technology).

Mardia, K. (1972). *Statistics of Directional Data*, Academic Press, London.

Matas, J., Koubaroulis, D. och Kittler, J. (2000). Colour image retreival and object recognition using the multi modal neighbourhood signature, *Proceedings of the Sixth European Conference on Computer Vision, Dublin*, Vol. 1842 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 48–64.

McIlhagga, W. (1998). Colour vision, *i* Sangwine och Horne (1998), kapitel 2, sid. 7–25.

Niemenmaa, J. (2000). *Feature detection in images with the hybrid-pyramid representation: System design, theoretical analysis and experimental evaluation*, Master's thesis, Dept. of Numerical Analysis and Computing Science, KTH (Royal Institute of Technology). Under utarbetande.

Palus, H. (1998). Representations of colour images in different colour spaces, *i* Sangwine och Horne (1998), kapitel 4, sid. 67–90.

Pujas, P. och Aldon, M.-J. (1997). Estimation of the color image gradient with perceptual attributes, *Proceedings of ICIAP'97, 9th International Conference On Image Analysis And Processing, Florence*, Vol. 1310 of *Lecture Notes in Computer Science*, Springer-Verlag, sid. 103–107.

Saber, E., Tekalp, A. M. och Bozdagi, G. (1997). Fusion of color and edge information for improved segmentation and edge linking, *Image and Vision Computing* **15**(10): 769–780.

Sangwine, S. J. och Horne, R. E. N. (eds) (1998). *The Colour Image Processing Handbook*, Optoelectronics Imaging and Sensing, Chapman and Hall, London.

Sapiro, G. och Ringach, D. L. (1996). Anisotropic diffusion of multivalued images with asidlication to color filtering, *IEEE Transactions on Image Processing* **5**(11): 1582–1586.

Saxe, D. och Foulds, R. (1996). Toward robust skin identification in video images, *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, Killington*, IEEE Computer Society Press, sid. 379–384.

Sisefsky, J. (1993). Hur ser färgerna ut? Ögat och fysiken oense, *Forskning och Framsteg* **1**: 16–22.

Slater, D. och Healey, G. (1996). The illumnation-invariant recognition of 3d objects using local color invariants, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **18**(2): 206–210.

Stokman, H. och Gevers, T. (1999). Detection and classification of hyper-spectral edges, *Proceedings of the Tenth British Machine Vision Conference, Nottingham*, Vol. 2, BMVA Press, sid. 643–651.

Swain, M. J. och Ballard, D. H. (1991). Color indexing, *International Journal of Computer Vision* **7**(1): 11–32.

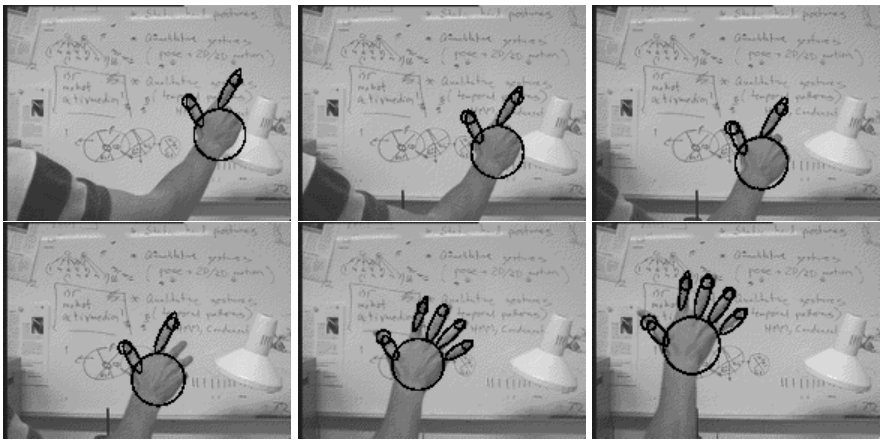
Terrillon, J.-C., David, M. och Akamatsu, S. (1998). Automatic detection of human faces in natural scene images by use of a skin color model and of invariant moments, *Proceedings of the third International Conference on Automatic Face and Gesture Recognition, Nara*, IEEE Computer Society Press, sid. 112–117.

Westin, C.-F. och Westelius, C.-J. (1988). *A colour model for hierarchical image processing*, Master's thesis, Linköping University, Sweden, SE-581 83 Linköping, Sweden. LiTH–ISY–EX–0857.

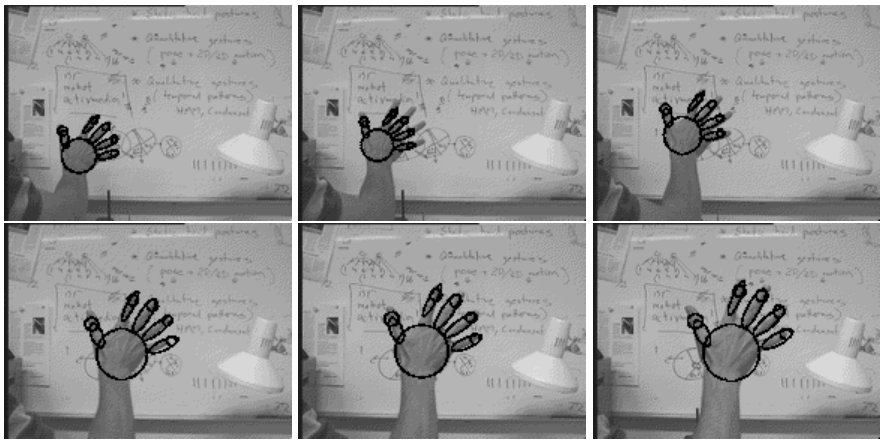
Würtz, R. och Lourens, T. (2000). Corner detection in color images through a multiscale combination of end-stosided cortical cells, *Image and Vision Computing* **18**: 531–541.

Bilaga A

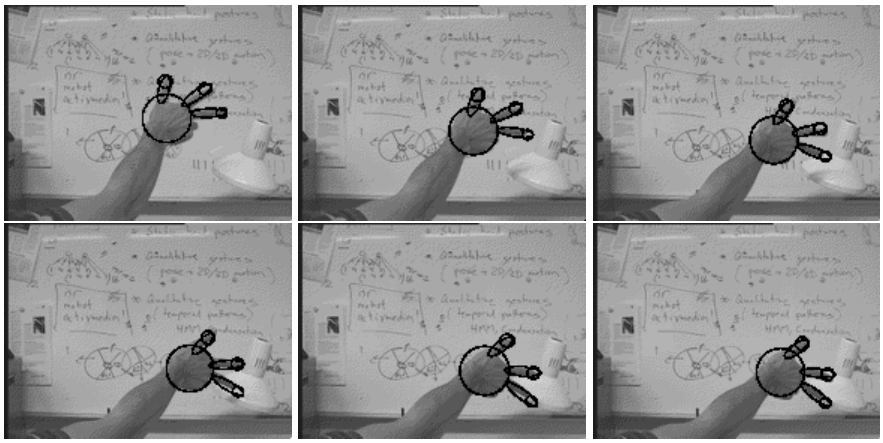
Ytterligare videosekvenser



Figur A.1. Exempel ur en videosekvens i vilken särdragsdetektion och handföljning har prövats. Resultatet av handföljningen är utritat i gråskaleversioner av bilder ur videosekvensen.



Figur A.2. Exempel ur en videosekvens i vilken särdragsdetektion och handföljning har prövats. Resultatet av handföljningen är utritat i gråskaleversioner av bilder ur videosekvensen.



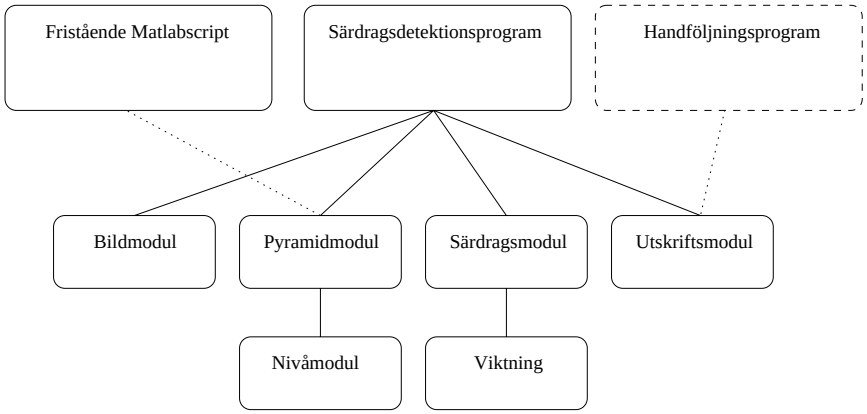
Figur A.3. Exempel ur en videosekvens i vilken särdragsdetektion och handföljning har prövats. Resultatet av handföljningen är utritat i gråskaleversioner av bilder ur videosekvensen.

Bilaga B

Beskrivning av programvara

B.1 Övergripande struktur

I figur B.1 visas strukturen hos realtidsimplementationen. Strukturdata för hybridpyramider, antal nivåer, vid vilka nivåer subsampling ska ske och normeringskoefficienter för särdragsdetektion, genereras före körning av ett Matlabscript. Detta tar lång tid, men då datafiler väl genererats kan de användas till särdragsdetektion i realtid. Vid viktning av detekterade särdrags signifikans används histogram. Ett program för generering av histogram från bilder har också skapats.



Figur B.1. Övergripande struktur för realtidsimplementationen.

Särdragsdetektionsprogrammet gör blobb- och åsdetektion, med stöd för användning av färginformationen i bilder, och presenterar resultatet grafisk samt skriver ut detekterade särdrag. Programmet består av ett antal olika moduler. Ett antal inparametrar kan anges för att bestämma hur särdragsdetektionen ska ske.

Följande parametrar kan anges:

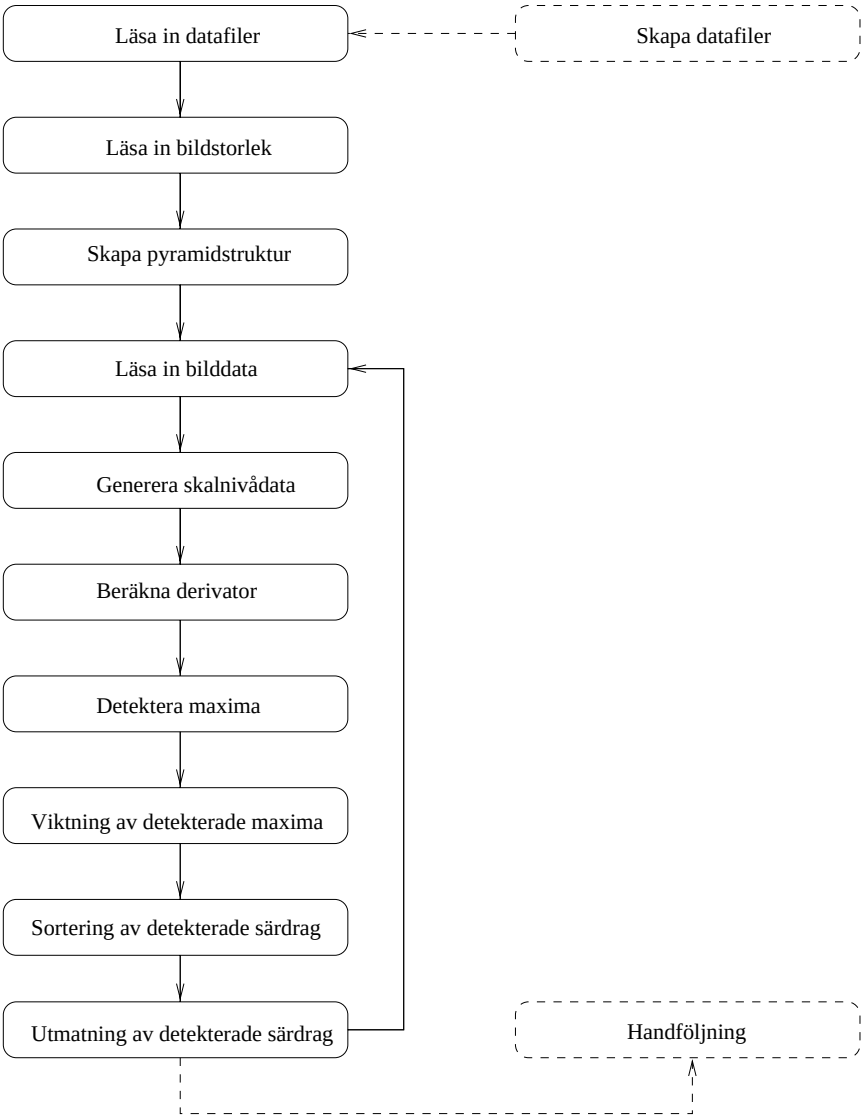
- Datafil med strukturdata för hybridpyramid.
- Datafil med normeringskoefficienter.
- Tröskelvärde vid särdragsdetektion.
- Histogramfil för viktning av särdrags signifikans.
- Maximalt antal detekterade särdrag för utskrift.
- Begränsning av maximadetektion till ett visst skalintervall.

Det finns dessutom olika versioner av särdragsdetektionsprogrammet som har olika egenskaper. En del versioner arbetar på en statisk bild, andra på en videosekvens med många bilder. Det finns också en skillnad i om särdragsdetektion ska ske i gråvå, separat i de olika färgkanalerna eller med en kombinerad operator. Dessutom varierar den färgmodell som används vid särdragsdetektion i färg.

De moduler som ingår i programmet är:

- Bildmodul, hanterar inläsning av bilder och byte av färgrymd.
- Pyramidmodul, skapar hybridpyramider med hjälp av datafiler genererade av ett Matlabscript. Använder i sin tur en nivåmodul som hanterar nivåerna i pyramiden. Där görs derivateberäkningar och maximadetektion.
- Särdragsmodul, hanterar detekterade särdrag. Modulen rensar ut dubbletter, sorterar särdrag, beräknar ellipser för åsar m.m. Denna modul använder en viktningsmodul som viktar särdrags signifikans med deras likhet med en sökt färg.
- Utskriftsmodul, presenterar detekterade särdrag grafiskt, skriver ut data på disk. Denna modul är den som kommunicerar med handföljningsprogrammet, som tar utdatafilerna från särdragsdetektionsprogrammet och gör handföljning baserad på dessa särdrag.

Hur särdragsdetektion sker i programmet beskrivs i figur B.2. De moment som inte ingår i särdragsdetektionsprogrammet är streckade.



Figur B.2. Särdragsdetektionens förlopp i realtidsimplementationen.

B.2 Kommandoradsparametrar

B.2.1 Särdragsdetektionsprogram

Särdragsdetektionsprogrammet har tre obligatoriska parametrar:

```
colourFeature <strukturdatafil> <normeringsfil> <bildfil>
```

Följande parametrar måste anges till särdragsdetektionsprogrammet:

- `strukturdatafil`, namnet på den fil med strukturdata som ska användas.
- `normeringsfil`, namnet på den tillhörande normaliseringskoefficientfilen.
- `bildfil`, namnet på den bild i vilken särdragsdetektion ska ske (för de versioner som arbetar med sekvenser anges namnet på sekvensen).

Dessutom kan ett antal ej obligatoriska parametrar anges, dessa anges då efter de tre obligatoriska parametrarna. De övriga parametrarna är:

- `-n` N, begränsa utmatningen av detekterade maxima till de N mest signifikanta.
- `-t` T, använd T som tröskelvärde vid särdragsdetektion.
- `-h histogramfil`, använd viktning av detekterade särdrags signifikans, det histogram som ska användas finns i fil `histogramfil`. Anges inte denna parameter sker ingen viktning.
- `-tmin` t_{min}, detektera endast särdrag på skalnivåer med $t \geq t_{min}$.
- `-tmax` t_{max}, detektera endast särdrag på skalnivåer med $t \leq t_{max}$.

B.2.2 Histogramgenereringsprogram

Till histogramgenereringsprogrammet anges två parametrar:

```
make3Dhistogram <bildfilnamn> <histogramfilnamn>
```

Dessa har följande innebörd:

- `bildfilnamn`, namnet på den fil som histogrammet ska skapas ifrån. Alla punkter i bilden som är svarta, $(R, G, B) = (0, 0, 0)$ antags tillhöra bakgrunden. I nuvarande implementation används endast en bild per histogram, men det är rättframt att utvidga histogramgenereringen till att använda flera bilder.
- `histogramfilnamn`, namnet på den fil i vilken histogrammet ska lagras.

B.2.3 Normaliseringskoefficientscript

Till normaliseringskoefficientscriptet anges fyra parametrar:

```
makeDatafiles <filnamnsprefix> <c> <n> <p>
```

Dessa har följande innebörd:

- `filnamnsprefix`, de datafiler som skapas döps av programmet till `filnamnsprefix.descriptor` (strukturdata) och `filnamnsprefix.norms` (normaliseringskoefficienter).
- `c`, anger hur ofta subsampling ska ske, subsampling sker en gång varje c nivåer. c behöver inte vara ett heltal.
- `n`, antal nivåer i pyramiden.
- `p`, anger det p som används i L_p-normaliseringen.

B.3 Funktionsanrop

Realtidsimplementationen är inte tänkt att köras som ett självständigt program, utan den är tänkt att användas som ett stödbibliotek med funktioner för särdragsdetektion. Detta kan sedan användas av program där särdragsdetektion behövs. De viktigaste funktionerna i varje modul redovisas här.

B.3.1 Bildmodulen

- `unsigned char *readImage(char *filename, int *width, int height)`, läser in en bild lagrad i en fil med namn `filename` på `PPM(raw)` format. `height` och `width` sätts till dimensionerna hos bilden och funktionens returvärde är en pekare till bilddata. Bilddata ligger lagrat som $R_1G_1B_1R_2G_2B_2\ldots$, där (R_i, G_i, B_i) är RGB-värdet i bildpunkt i. Punkterna är lagrade radvis, från vänster till höger, uppifrån och ned.
- `int readImage2(char *filename, unsigned char *image, int width, int height)`, som `readImage` med två skillnader. I `readImage2` skickas ett minnesområde där bilddata ska lagras med (i readImage allokeras nytt minne). Returvärdet för `readImage2` är 1 om det gick att läsa in bilden, 0 annars.
- `void convertImageToPyramidBase(unsigned char *image, Pyr pyramid, int width, int height)`, sätter basnivån i pyramidobjektet `pyramid` till gråvårepresentationen av bilden `image`. `width` och `height` är bildens dimensioner.

- `void convertImageToYIQ(int noofPyrs, int *channelToPyrMap, Pyr **pyrVec, unsigned char *image, int width, int height)`, sätter basnivåerna i ett antal pyramidobjekt till innehållet i färgkanalerna i bilden `image` konverterad till YIQ-färgmodellen. `noofPyrs` talar om hur många pyramidobjekt som ska användas. `channelToPyrMap` är en vektor som talar om vilken färgkanal som ska lagras i vilken pyramid, värdet i position i talar om vilken färgkanal pyramid i ska innehålla. `pyrVec` är en vektor med pekare till pyramidobjekt. `width` och `height` är bildens dimensioner. Motsvarande funktion finns även för färgmodellerna RGB, YUV och den gaussiska färgmodellen. Då byts `YIQ` i funktionsnamnet till `RGB`, `YUV` respektive `Gauss`.

B.3.2 Pyramidmodulen

Pyramider är C++ objekt. En utförlig beskrivning av implementationen av pyramider finns i Niemenmaa (2000), vars arbete utvidgats här. Följande metoder finns i pyramidklassen:

- `Pyr(const char *pyramidfile, const char *normfile, int width, int height, real maxThreshold)`, skapar ett pyramidobjekt där `pyramidfile` är namnet på en datafil med strukturdata för pyramiden, `normfile` är namnet på en datafil med normaliseringskoefficienter, `width` och `height` är dimensionerna på basnivån och `maxThreshold` är tröskelvärde vid särdragsdetektion.

- `void reducePyramid()`, skapar skalnivåerna i pyramiden.

- `void derivPyramid()`, beräknar derivateuttryck (blobb- och åsstyrka) på samtliga skalnivåer.

- `void derivPyramid(real tmin, real tmax)`, beräknar derivateuttryck (blobb- och åsstyrka) på de skalnivåer som behövs om detektion bara ska ske i skalintervallet `[tmin, tmax]`.

- `void derivSum(Pyr *p1, Pyr *p2)`, beräknar summan av derivateuttrycken i tre pyramider.

- `void derivSum(Pyr *p1, Pyr *p2, real tmin, real tmax)`, som ovan, enbart i skalintervallet `[tmin, tmax]`.

- `void findmaxima()`, detekterar lokala maxima i derivateuttrycken i pyramiden.

- `void findmaxima(real tmin, real tmax)`, som ovan, enbart i skalintervallet `[tmin, tmax]`.

B.3.3 Särdragsmodulen

- `void initMax(char *histogramFilename)`, initialiserar särdragshanteraren. `histogramFilename` är namnet på en fil där det histogram som ska användas vid viktning av särdragens signifikans är lagrat. Om `histogramFilename` är `NULL` görs ingen viktning.
- `void clearMax()`, återställer särdragshanteraren till utgångsläget.
- `int getNoofMax()`, returnerar antalet detekterade särdrag.
- `void sortMax()`, sorterar särdragen på deras signifikans, starkaste särdraget först.
- `void removeDuplicateMax()`, eliminerar dubbletter (särdrag som överlappar mycket).
- `void clipToNMax(int N)`, begränsar antalet särdrag till de `N` första.
- `void colourClosenessStr(unsigned char *image, int w, int h)`, viktar särdragens signifikans med deras likhet med en sökt färg.

B.3.4 Utskriftsmodulen

- `void displayBlobs(char *imageFilename)`, visar detekterade särdrag grafiskt, överlagrade på bilden i filen `imageFilename`.

B.4 Format på datafiler

B.4.1 Strukturdata för hybridpyramiden

Strukturdata för hybridpyramider lagras i textfiler. Ett exempel på innehållet i en strukturdatafil är:

```
Rho 0.800000
tmax 40.000000
Smoothfilter File smooth.filter
Subsamplingfilter File  ss.filter
Level 0, Estimated Variance 0.0, Subsampled No, Grid Spacing 1
Level 1, Estimated Variance 1.0, Subsampled No, Grid Spacing 1
Level 2, Estimated Variance 2.0, Subsampled No, Grid Spacing 1
Level 3, Estimated Variance 7.0, Subsampled Yes, Grid Spacing 2
Level 4, Estimated Variance 24.0, Subsampled Yes, Grid Spacing 4
Level 5, Estimated Variance 40.0, Subsampled No, Grid Spacing 4
```

De fyra första raderna är till för att en läsare ska få en sammanfattning av innehållet i filen, där anges ρ-värde, vilket skalintervall och vilken typ av filter

som använts vid genereringen av strukturdata. Dessa rader ignoreras av pyramidimplementationen.

Sedan följer själva strukturdata, på formen:
`Level` X`, Estimated Variance` V`, Subsampled` S`, Grid Spacing` H
X är ett heltal som anger vilken nivå i pyramiden raden behandlar. V är ett flyttal som är en uppskattning av skalvärdet för nivån (värdet ignoreras, det värde som anges i normaliseringskoefficientfilen används istället). S anger om nivån är en subsamplad version (värdet `Yes`) eller en utjämnad version (värdet `No`) av föregående nivå. Heltalet H anger avståndet mellan två bildpunkter i denna nivå (avståndet är 1 på basnivån och dubblas sedan vid varje subsampling).

B.4.2 Normaliseringskoefficienter

Normaliseringskoefficientfiler är också textfiler. Ett exempel på innehållet i en koefficientfil är:

```
level = 1 variance = 3.000000 dx = 1 dy = 2
C=0.468750 N=0.772259 norm = 1.647487
level = 2 variance = 5.071429 dx = 1 dy = 2
C=0.215332 N=0.772259 norm = 3.586365
level = 3 variance = 7.451178 dx = 1 dy = 2
C=0.121511 N=0.772259 norm = 6.355494
level = 1 variance = 3.000000 dx = 2 dy = 1
C=0.468750 N=0.772259 norm = 1.647487
level = 2 variance = 5.000000 dx = 2 dy = 1
C=0.215332 N=0.772259 norm = 3.586365
level = 3 variance = 7.000000 dx = 2 dy = 1
C=0.121511 N=0.772259 norm = 6.355494
level = 1 variance = 3.000000 dx = 2 dy = 2
C=0.562500 N=0.936798 norm = 1.665418
level = 2 variance = 5.071429 dx = 2 dy = 2
C=0.191406 N=0.936798 norm = 4.894289
level = 3 variance = 7.451178 dx = 2 dy = 2
C=0.084123 N=0.936798 norm = 11.136090
```

Koefficientfiler består av rader på följande format:
`level =` X `variance =` V `dx =` dx `dy =` dy `C=`C `N=`N `norm =` n
I exemplet ovan har varje rad brutits före `C=` för att få plats på sidan. X anger vilken nivå som raden gäller. V anger vilket skalvärde nivån motsvarar. dx och dy anger vilken deriveringsordning normaliseringskoefficienten på denna rad gäller för (derivering dx gånger i x-led och dy gånger i y-led). n är normaliseringskoefficienten. C och N ignoreras av programmet.

B.4.3 Viktningshistogram

Histogramfilerna är textfiler. Ett exempel ur en histogramfil är:

```
histogram, format is 256*256*256 int
pos of (R,G,B) is R+256*G+256*256*B
hist stored in form x,y -> x next values
in hist is y, first pos is 0
size of picture was 10, 10 (w,h)
16777210,0
3,10
2,15
1,30
```

De fem första raderna talar om hur histogrammet är lagrat. Sedan kommer ett antal rader på formen:
`n, v`
vilket betyder att de `n` följande cellerna i histogrammet har värdet `v`. Cellerna är lagrade i en vektor med 16777216 (256x256x256) element. Ordning i vektorn är sådan att cellen för (R, G, B) har position $R + 256G + 65536B$.

B.4.4 Detekterade särdrag

I de fall då utskrift av detekterade särdrag sker lagras särdragen i textfiler. Det skapas en fil för blobbar och en för åsar. Om detektionen skett på en sekvens bilder skapas ett sådant par för varje bildruta. På första raden i en särdragsfil anges antalet särdrag i filen. Sedan anges särdragen ett per rad.

Ett exempel ur en fil med detekterade blobbar:

```
3
44 135 47.418377 0.034637 61 63 53 0.000016
49 143 11.628028 0.033964 73 69 64 0.000018
48 127 11.772715 0.029541 71 70 61 0.000017
```

Blobbar skrivs ut som $x\ y\ t\ s\ R\ G\ B\ w$, där (x, y) är den bildpunkt och t är den skala blobben detekterats på. s är signifikansvärdet för blobben och (R, G, B) är färgen i den punkt blobben detekterats. w är den vikt som använts vid viktning av detta särdrags signifikans (ett mått på dess likhet med den sökta färgen). x, y, R, G och B är heltal, de andra flyttal.

Ett exempel ur en fil med detekterade åsar:

```
3
17 246 179.343597 0.154095 4.685362 1.167769 175 131 105 0.000019
97 243 189.208267 0.113987 4.493360 2.440766 186 147 108 0.000021
10 183 45.878040 0.078210 4.774290 1.553784 116 106 96 0.000014
```

Åsar skriv ut som *x y t s q v R G B w*. De nya värdena *q* och *v* beskriver förhållandet mellan ellipsens halvaxlar ($\frac{a}{b}$, där *a* är den långa halvaxeln och *b* den korta) respektive riktningen för den långa halvaxeln, angiven som vinkeln, i radianer, från x-axeln. Båda dessa är flyttal. De andra parametrarna är desamma som för blobbar.