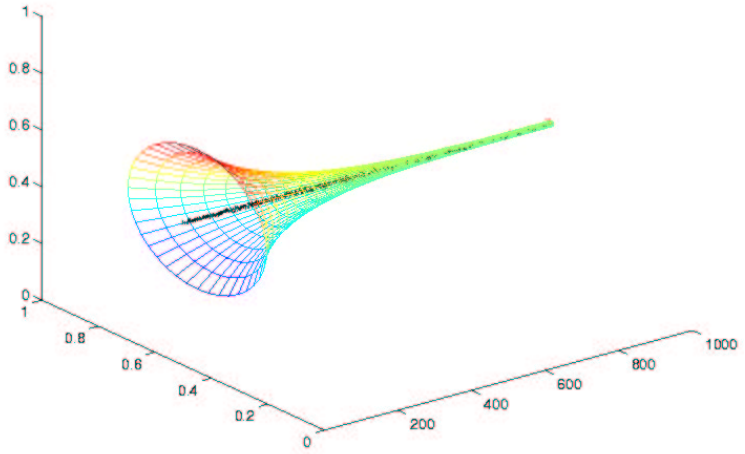
Real-time Object Detectie in Video met gebruik van Mixture Modellen voor achtergrondschatting



Frank Zwarthoed

17 April 2002

Real-time Object Detectie in Video
met gebruik van Mixture Modellen voor achtergrondschatting.

Auteur:	Frank Zwarthoed
Studie:	Informatica
Studierichting:	Technische Informatica
Specialisatie:	Intelligente Autonome Systemen
Faculteit:	FNWI
Universiteit:	Universiteit van Amsterdam
Locatie afstudeerproject	WCW
Datum:	17 April 2002
Begeleiders:	Dr. Ir. B.J.A. Kröse
	Drs. P.J. Withagen

Samenvatting

Bij dit onderzoek hebben we gekeken in hoeverre een background subtraction methode met gebruik van mixture modellen voor het beschrijven van een achtergrondmodel geschikt is voor real-time objectdetectie in video. We hebben hierbij gekeken naar de meest geschikte kleurruimte, representatie van het mixture model, update methode van het mixture model en manier van voorgrond- achtergrond classificatie.

We hebben gekozen voor de genormaliseerde rg kleurruimte. Voor het mixture model wordt er gebruik gemaakt van tweedimensionale kernels met een diagonale covariantiematrix. Voor het aanpassen van de parameters van het mixture model wordt gebruik gemaakt van een incrementele variant van het EM algoritme. Voor de classificatie van de pixels wordt voor iedere pixel de à posteriori waarschijnlijheid van de achtergrondkernel bepaald. De achtergrondkernel is de kernel met de hoogste waarschijnlijkheid. Deze beschrijft de kleur van de achtergrond van het betreffende pixel.

De kwaliteit van het uiteindelijke systeem voldoet niet aan de wensen. Het is met dit systeem niet mogelijk om objecten uniek te identificeren aan de hand van kenmerken die uit de videobeelden worden gehaald. De detectie van objecten is onvolledig, en werkt niet gedurende langere tijd.

Inhoudsopgave

Lijst van figuren

Lijst van tabellen

Hoofdstuk 1

Inleiding

Steeds vaker wordt op plaatsen waar veel mensen bij elkaar komen, zoals stations, openbare gebouwen, en recreatie gebieden, een beveiligingssysteem ingezet voor het detecteren van verdachte handelingen.

Beveiligingssystemen kunen bestaan uit meerdere typen sensoren. Naast bewegingsmelders, drukmelders, radar, microfoons, en lichtsluizen worden ook vaak camera's gebruikt als sensor. Het voordeel van een camera als sensor is dat er behalve detectie van objecten, ook veel informatie over de gedetecteerde objecten beschikbaar is. Zo kan een gedetecteerd object ook direct worden geïdentificeerd. Het nadeel dat hiermee samenhangt is dat er veel informatie wordt aangeboden, en al deze informatie dient binnen een korte tijd verwerkt te worden. Het automatiseren van de extractie van relevante gegevens uit de camerabeelden is wenselijk.

1.1 Context

Deze afstudeeropdracht is een onderdeel van het Autonomous Surveillance (AS) project. Dit is een samenwerkingsverband tussen TNO-FEL en de groep Intelligente Autonome Systemen van de Universiteit van Amsterdam. Het doel van het AS project is het ontwikkelen van een systeem dat in staat is tot het automatisch detecteren van verdachte gebeurtenissen in video.

Het systeem dient te kunnen bestaan uit gedistribueerde niet noodzakelijk overlappende intelligente sensoren van uiteenlopende types. Elke sensor moet aan de hand van de geëxtraheerde gegevens zelf in staat zijn om objecten te detecteren, en zo mogelijk te identificeren. De sensoren dienen in staat te zijn om met elkaar te communiceren. Zo kunnen objecten door het hele systeem, van sensor naar sensor worden gevolgd.

Een belangrijk type sensor bij dit systeem is de camera. Dit omdat er bij camerabeelden veel informatie, als positie, omvang, vorm, snelheid, richting, kleur, enz., van gedetecteerde objecten kan worden verkregen. Voordat deze objecteigenschappen kunnen worden bepaald, dient eerst te worden bepaald welke pixels tot de objecten behoren, en welke tot de achtergrond. Het doel van deze afstudeeropdracht is het automatisch detecteren van objecten in videosequenties, in real-time. Het resultaat hiervan is een segmentatie van het beeld in voorgrondpixels en achtergrondpixels voor ieder frame van de video.

1.2 Objectdetectie in videobeelden

Er zijn verschillende methoden ontwikkeld voor objectdetectie in video. Kanade et al [10] stellen de volgende opdeling voor:

- Optic Flow
- Temporal Differencing
- Background Subtraction

1.2.1 Optic Flow

Bij Optic Flow methoden wordt objectdetectie op basis van beweging gedaan. De snelheid van een gehele scène, dus zowel van de voorgrond als van de achtergrond, wordt geprojecteerd op het beeld. Dit resulteert in een zogenaamd flow field van de scène. Dit flow field geeft voor alle gebieden in het beeld een snelheidsvector. Op basis van verschil in richting en grootte van deze vectoren kunnen gebieden als object of achtergrond worden geclassificeerd.

Het voordeel van optic flow methoden is dat deze goed in staat zijn om objecten te detecteren. Ook kunnen ze zonder problemen worden ingezet bij gebruik van bewegende camera's. Een nadeel is dat optic flow methoden computationeel dermate complex zijn dat ze moeilijk real-time zijn te implementeren.

1.2.2 Temporal Differencing

Temporal differencing is een segmentatie techniek op basis van temporeel verschil in de pixelwaarden. Voor twee achtereenvolgende frames wordt per pixel gekeken of de pixelwaarde, van bijvoorbeeld een kleurkanaal of de intensiteit, duidelijk is veranderd. Als dit het geval is, wordt het betreffende pixel als objectpixel geclassificeerd. Het voordeel van deze methode is dat het niet gevoelig is voor geleidelijke veranderingen in de achtergrond, en er is geen initialisatieperiode nodig. Het belangrijkste nadeel van deze methode is dat alleen veranderingen van twee opeenvolgende pixelwaarden worden gedetecteerd. Zo wordt alleen de rand van een object als object geclassificeerd. Dit zijn de pixels die van twee achtereenvolgende pixelwaarden van kleur zijn veranderd.

1.2.3 Background subtraction

Bij *background subtraction* wordt per pixel de huidige pixelwaarde (bijvoorbeeld kleurwaarde of intensiteit) vergeleken met de waarde van de achtergrond op dezelfde locatie. Hiervoor dient een model voor de achtergrond bekend te zijn. Dit model moet per pixel een waarde voor de achtergrond beschrijven. Wanneer de huidige pixelwaarde in bepaalde mate afwijkt van het achtergrondmodel, wordt de betreffende pixel als voorgrond geclassificeerd. Background subtraction methoden zijn goed in staat tot het detecteren van alle objectpixels. Echter wanneer de achtergrond verandert als gevolg van belichtingsomstandigheden, bestaat de kans dat de hele scène als voorgrond wordt geclassificeerd.

1.3 Beschrijving gekozen aanpak

Voor dit project gelden er enkele eisen voor de object detectie. Zo moet niet alleen de objectdetectie maar ook de identificatie in real-time worden gedaan. Voor een goede identificatie

van objecten na de objectdetectie is het wenselijk dat er geen schaduwen en geen gaten in de objecten worden gedetecteerd.

Gebruik van optic flow voor object detectie in video is moeilijk real-time te implementeren. Temporal differencing methoden detecteren niet altijd alleen de objecten, maar ook gebieden die object zijn geweest. Background subtraction methoden zijn goed in staat objecten te detecteren, maar zijn slecht bestand tegen veranderingen in de omgeving bijvoorbeeld als gevolg van belichtingsomstandigheden.

Bij dit project is gekozen voor een background subtraction methode die gebruik maakt van een dynamisch achtergrondmodel. Het systeem is in staat het achtergrondmodel aan te passen, afhankelijk van de kleurveranderingen in het beeld. Dit heeft een aantal voordelen. Het systeem is zo minder gevoelig voor veranderingen in de belichtingsomstandigheden, er is geen initialisatie nodig met een leeg achtergrondbeeld, en de achtergrond kan worden veranderd zonder dat het systeem opnieuw moet worden gestart.

1.4 Overzicht

In hoofdstuk 2 wordt de theorie van de gekozen aanpak besproken. Hierbij wordt gekeken naar het gebruikte kleursysteem, het gebruikte achtergrondmodel, de methode om het achtergrondmodel aan te passen, en postprocessing om het uiteindelijke resultaat van de classificatie geschikt voor tracking te maken. In Hoofdstuk 3 wordt een aantal experimenten gedaan om een keuze te maken uit de in hoofdstuk 2 geïntroduceerde theoretische mogelijkheden. In Hoofdstuk 4 wordt het complete systeem beschreven zoals dat is ontstaan na de keuzes die zijn gemaakt. Tenslotte worden de conclusies samengevat in Hoofdstuk 5.

Hoofdstuk 2

Object detectie met background subtraction

Het detecteren van objecten in een beeld komt neer op het opdelen van het beeld in voorgrond (de objecten), en achtergrond (alles wat niet object is). Bij dit onderzoek wordt gebruik gemaakt van een background subtraction methode. Dit is een veel toegepaste methode bij object detectie in video, waarbij een model van de achtergrond wordt bijgehouden waarmee elk nieuw frame wordt vergeleken. Dit achtergrondmodel geeft voor elke pixel de bij de achtergrond behorende pixelwaarde (van bijvoorbeeld een kleurkanaal of de intensiteit). Pixels die afwijken van het achtergrondmodel worden als object geclassificeerd. Door Toyama et al. [9] zijn een aantal situaties gegeven die problemen kunnen opleveren voor background subtraction methoden. Een ideale methode zou robuust moeten zijn voor de volgende situaties.

- **Schaduw** Heeft een van de achtergrond afwijkende kleurintensiteit, maar behoort niet tot een object.
- **Camouflage** Objecten kunnen dezelfde kleur hebben als de achtergrond.
- **Geleidelijke belichtings veranderingen** Als gevolg van de veranderende belichtingsintensiteit gedurende de dag verandert de intensiteit van de scène.
- **Abrupte belichtings veranderingen** Als gevolg van het aan- of uitschakelen van de verlichting in een scène verandert niet alleen de intensiteit, maar ook de kleur als gevolg van de kleur van het licht.
- **Periodiek veranderende achtergrond** Per pixel kan de achtergrondkleur oscilleren tussen verschillende kleurwaarden. Bijvoorbeeld een bewegende tak met achtereenvolgens groen van de bladeren en blauw van de lucht.
- **Slapend object** Er kan geen onderscheid worden gemaakt tussen voorgrond objecten die stilstaan of achtergrondobjecten.
- **Verplaatste objecten** Achtergrondobjecten kunnen worden verplaatst of verwijderd.
- **Initialisatie** Het is niet altijd mogelijk op een lege scène te initialiseren.

Een aantal van bovengenoemde problemen kan worden verholpen door de juiste keuze van het het gebruikte kleursysteem, achtergrondmodel, en updatemethode van het achtergrondmodel.

2.1 Kleursysteem

Kleur is het waargenomen effect van licht in het zichtbare spectrum met golflengtes van $400nm$ tot $700nm$. Het fysische vermogen van licht wordt beschreven in een zogenaamde Spectral Power Distribution(SPD). Hierin wordt voor elke golflengte aangegeven wat het vermogen van het waargenomen licht bij die golflengte is.

Een enkele sensor is niet in staat om het gehele zichtbare spectrum te registreren. De sensoren zijn gevoelig in een bepaald gebied van het spectrum. Door sensoren met verschillende gevoeligheden te combineren wordt een zo groot mogelijk gebied van het spectrum beschreven.

Het Commission Internationale de l'Eclairage (CIE) is een organisatie die zich bezig houdt met de standaardisatie voor het beschrijven van kleuren. De kleurwaarden zoals deze door de framegrabber worden geleverd, zijn volgens de RGB standaard van het CIE. De RGB standaard gebruikt drie primaire kleuren om een deel van het spectrum te beschrijven, de zogenaamde tristimuli. Voor RGB zijn dit de kleuren Rood $(700,0nm)$, Groen $(546,1nm)$ en Blauw $(435,8nm)$. Door deze kleuren ieder met een bepaald gewicht te mengen kunnen verschillende kleuren uit het spectrum worden beschreven. Dit wordt de kleurruimte van de RGB standaard of het RGB kleursysteem genoemd. Naast RGB zijn er verschillende andere kleursystemen geïntroduceerd, met verschillende eigenschappen.

Gevers heeft in [4] een aantal veel gebruikte kleursystemen vergeleken. In tabel 2.1 zijn voor verschillende kleursystemen een aantal voor dit onderzoek interessante eigenschappen voor het beschrijven van de kleurruimte samengevat.

Voor object identificatie is het wenselijk dat de kenmerken extractie van de gedetecteerde objecten onafhankelijk is van de orientatie van het object ten opzichte van de camera(Een persoon die vanaf de voorkant wordt bekeken zou hetzelfde moeten worden geïdentificeerd als dezelfde persoon van de zijkant af bekeken). Ook zal de detectie onafhankelijk moeten zijn van de vorm van het object(Een bukkend persoon zou hetzelfde moeten worden geidentificeerd als dezelfde staande persoon). Verder is het belangrijk dat schaduwen niet als object worden geclassificeerd. Bij de kenmerken extractie van de gedetecteerde objecten zijn alleen kenmerken over de objecten gewenst, en niet de kenmerken van de schaduwen bij deze objecten. Het is daarom wenselijk dat het gebruikte kleursysteem onafhankelijk is voor belichtingsomstandigheden.

De kleursystemen die voldoen met betrekking tot de eigenschappen voor robuustheid tegen variabele beeld omstandigheden zijn rgb, xyz, Tint en verzadiging. In de vergelijkingen 2.1 tot en met 2.9 is gegeven hoe deze kleursystemen uit het RGB systeem kunnen worden bepaald.

De data van de framegrabber wordt gegeven in RGB. Vanwege het gebruik van een kostbare *arctan* operatie in de conversie van RGB naar Tint, is de combinatie tint/verzadiging niet geschikt. Het xyz kleursysteem is afgeleid van het XYZ kleursysteem. Het XYZ kleursysteem is vergelijkbaar met het RGB kleursysteem maar maakt gebruik van andere waarden voor de tristimuli. De tristimuli waarden van het XYZ kleursysteem zijn gebaseerd op eigenschappen van het menselijk oog. Alle voor de mens zichtbare kleuren kunnen hiermee worden beschreven. Het xyz kleursysteem vereist dus een extra conversie ten opzichte van rgb, maar heeft voor dit onderzoek verder geen voordelen.

Het blijkt dat het rgb kleursysteem de gewenste eigenschappen heeft. Nadeel van dit kleursysteem is dat de bijbehorende kleurruimte niet lineair is. Bij lage intensiteitwaarden, zie vergelijking 2.7, wordt het kleursysteem instabiel. Er treedt dan een grote variantie in de kleurwaarden op. Het voldoet om van dit kleursysteem alleen de kleurkanalen r en g te gebruiken. Hiermee kunnen alle kleuren uit de rgb kleurruimte worden beschreven.

Kleur systeem	device onafh.	lineair	gezichtspunt onafh.	object vorm onafh.	highlights	bel. intens.
RGB	-	+	-	-	-	-
XYZ	+	+	-	-	-	-
rgb	-	-	+	+	-	+
xyz	+	-	+	+	-	+
L*a*b*	+	-	-	-	-	-
U*V*W*	+	-	-	-	-	-
I1I2I3	-	+	-	-	-	-
YIQ	-	+	-	-	-	-
YUV	-	+	-	-	-	-
Intensiteit	-	+	-	-	-	-
Tint	-	-	+	+	+	+
Verzadiging	-	-	+	+	-	+

Tabel 2.1: Enkele kleursystemen en voor objectdetectie interessante eigenschappen. Conclusie uit Gevers:Color in Image Search Engines [4]

$$[htb]r = \frac{R}{(R+G+B)} \tag{2.1}$$

$$g = \frac{G}{(R+G+B)} \tag{2.2}$$

$$b = \frac{B}{(R+G+B)} \tag{2.3}$$

$$x = \frac{X}{(X+Y+Z)} \tag{2.4}$$

$$y = \frac{Y}{(X+Y+Z)} \tag{2.5}$$

$$z = \frac{Z}{(X+Y+Z)} \tag{2.6}$$

$$Intensiteit = R+G+B \tag{2.7}$$

$$Tint = arctan\left(\frac{\sqrt{3}\,(G-B)}{(R-G)+(R-B)}\right) \tag{2.8}$$

$$Verzadiging = 1-3\cdot min\,(r,g,b) \tag{2.9}$$

2.2 Het Achtergrondmodel

De objectdetectie wordt gedaan op basis van verschil in kleur tussen het object en de achtergrond. De achtergrondkleur wordt gerepresenteerd door een dynamisch model. De parameters van het achtergrondmodel kunnen door het systeem worden veranderd, en de background subtraction methode kan zo robuust worden gemaakt voor veranderingen in de achtergrond. Dit dynamische achtergrondmodel wordt bepaald op basis van de in het verleden opgetreden kleurwaarden. Het kleurverleden van een pixel wordt beschreven door de verdeling van

kleurwaarden die in het verleden zijn opgetreden. Deze verdeling wordt hier benaderd door een zogenaamd *mixture model*. Dit is een sommatie of mixture van Gaussische verdelingen. Het achtergrondmodel wordt gevormd door een deel van het mixture model. Een soortgelijke aanpak, is eerder al toegepast door onder andere Stauffer en Grimson [8] in de RGB kleurruimte, en Stijnman en van den Boomgaard [3] in de rg kleurruimte.

2.2.1 Het Mixture Model

Het achtergrondmodel dient de kleurwaarde van de achtergrond te beschrijven. Omdat niet bekend is of de huidige kleurwaarde van een pixel een object of de achtergrond beschrijft, wordt iedere geregistreerde kleurwaarde bijgehouden. De verdeling van alle in het verleden geregistreerde kleurwaarden van een pixel, wordt het kleurverleden van het pixel genoemd.

Wanneer een pixel een eenvoudig achtergrond beschrijft, dus een constante kleurwaarde heeft afgezien van de ruis, wordt aangenomen dat de kleurwaarden normaal verdeeld zijn. Het kleurverleden van dit pixel kan worden benaderd met een Gaussische verdeling. Als dit pixel ergens in het verleden een object heeft beschreven, wordt aangenomen dat de kleur van het object ook kan worden benaderd door een Gaussische verdeling. Ook de schaduw van dit object kan worden beschreven met een Gaussische verdeling. Het totale kleurverleden kan worden benaderd door een gewogen som van de drie Gaussische verdelingen[1]. De weegfactor per verdeling is een maat voor hoe vaak de betreffende kleur is geregistreerd. Deze wordt gegeven door $\pi_k = p(k)$, de kans op kernel k.

Één verdeling uit deze som wordt een *kernel* genoemd. Een kernel k geeft de kans op een kleurwaarde $\vec{x}$. In d dimensies wordt deze kans gegeven door de volgende Gaussische verdeling:

$$p(\vec{x}|k) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_k)^T \Sigma_k^{-1} (\vec{x}-\vec{\mu}_k)}. \tag{2.10}$$

Hierin is $\vec{\mu}_k$ het gemiddelde, en Σ_k de covariantiematrix behorende bij kernel k.

Het kleurverleden kan worden benaderd door een som van gewogen kernels. Iedere kernel wordt vermenigvuldigd met een weegfactor $\pi_k = p(k)$. Dit wordt ook wel het *mixture model* genoemd:

$$p(\vec{x}) = \sum_{k=1}^{K} \pi_k p(\vec{x}|k). \tag{2.11}$$

Het is wenselijk dat het mixture model zo snel mogelijk een goede benadering van het kleurverleden geeft, ongeacht de geregistreerde kleurwaarden. Om dit te bereiken moet initieel aan alle mogelijke kleurwaarden dezelfde waarschijnlijkheid worden toegekend. Deze situatie kan worden benaderd door de kernels homogeen over de kleurruimte te initialiseren.

Wij hebben ervoor gekozen om de kernels over de diagonaal van de kleurruimte te verdelen. Dit is niet de meest homogene initialisatie maar wel een die ongeacht het aantal kernels en ongeacht de gebruikte vorm van het mixture model (zie sectie 2.3) kan worden gerealiseerd. De parameters μ_k^i voor het gemiddelde van een kernel k en kleurkanaal i bij gebruik van $nKer$ kernels in het mixture model, zijn gelijk aan $\frac{k}{nKer+1}$. De paramters van de covariantiematrix worden allemaal gelijk aan $\frac{1}{100 \cdot nKer}$ genomen. De weegfactoren behorende bij een kernel zijn initieel voor alle kernels gelijk: $\frac{1}{nKer}$. Hierbij wordt aangenomen dat de maximale kleurwaarde gelijk is aan 1.

[1] Wanneer in een scène veel verschillend gekleurde objecten verschijnen, kan het nodig zijn om meer dan drie Gaussische verdelingen te gebruiken voor de benadering van het kleurverleden. Er is dan minder kans dat de achtergrondverdeling wordt 'verstoord' door het aanpassen van de parameters van het mixture model.

2.2.2 Achtergrond en Voorgrond

Op basis van het mixture model moet worden besloten of een pixel met de huidige kleurwaarde tot de voorgrond of tot de achtergrond behoort. De aanname wordt gedaan dat de achtergrondkleur de kleur is die in het verleden het meest is voorgekomen. Deze kleur wordt beschreven door de kernel met de hoogste weegfactor π_k.[2] Het achtergrondmodel bestaat uit de kernel met de hoogste weegfactor, de *achtergrondkernel*[3].

De classificatie van een pixel gebeurd op basis van de huidige kleurwaarde, en het huidige achtergrondmodel. Eén van de manieren waarop dit kan worden gedaan is de afstand in de kleurruimte tussen de huidige kleurwaarde en het gemiddelde van de achtergrondkernel uit te drukken in het aantal standaarddeviaties van de achtergrondkernel. Wannneer deze afstand groter is dan 3 keer de standaarddeviatie, wordt het betreffende pixel als voorgrond geclassificeerd.

Een andere mogelijkheid is het bepalen van de à posteriori waarschijnlijkheid dat de kleur van een pixel $\vec{x}$ tot het achtergrondmodel (de kernel k met de hoogste weegfactor π_k) behoort. Deze waarschijnlijheid wordt gegeven door:

$$P(k|\vec{x}_t) = \frac{\pi_{k,t-1} p(\vec{x}_t|k)}{p(\vec{x}_t)}. \tag{2.12}$$

Wanneer deze kans groter is dan 50% wordt een pixel als achtergrond geclassificeerd, anders als voorgrond[4].

2.3 Het EM algoritme

Om de objectdetectie robuust voor veranderingen in de achtergrond te maken moet het systeem in staat zijn het achtergrondmodel aan te passen. Dit wordt gedaan door de parameters van het mixture model(zie vergelijking 2.11) aan te passen aan de huidige kleurwaarde van het pixel. Een veel gebruikte aanpak hiervoor is het Expectation-Maximization (EM) algoritme. Dit algoritme is in 1977 geïntroduceerd door Dempster, et. al. [2].

Het EM algoritme is een algemene methode voor het vinden van de *maximum-likelihood* schatting van de parameters van een verdeling van een gegeven dataset, wanneer de dataset incompleet is. De dataset bestaat hier uit een reeks onafhankelijke meetwaarden van de kleurwaarde van een pixel. De verdeling die deze dataset beschrijft is het mixture model. Er wordt gebruik gemaakt van een incrementele variant van het EM algoritme voor het aanpassen van de parameters van het mixture model. Zo wordt elke keer na het meten van een kleurwaarde van een pixel het mixture model aangepast aan de gemeten kleurwaarde. Hierbij is gebruik gemaakt van *exponentieel vergeten*. Zo wordt niet het complete kleurverleden van een pixel beschreven, maar alleen het recente kleurverleden.

[2]Een andere optie is de kernel met de hoogste weegfactor/variantie ratio $\frac{\pi}{\sigma^2}$ als achtergrondmodel te nemen [8]. In dit geval wordt ook de variantie van de betreffende kernel meegenomen. Een kernel die een achtergrondkleur beschrijft zal over het algemeen een lagere variantie hebben dan een kernel die een objectkleur beschrijft. Een achtergrond beweegt over het algemeen niet, terwijl voorgrond over het algemeen wel in beweging is. De kleur van een bewegend oppervlak zal een grotere variantie vertonen als gevolg van een veranderende lichtweerkaatsing op het oppervlak. Zodoende geeft de variantie dus extra informatie over een kernel al dan niet aan de achtergrond toe te kennen.

[3]Er kunnen ook meerdere kernels in het achtergrondmodel worden opgenomen. Bijvoorbeeld de belangrijkste kernels waarvoor de som van de weegfactoren boven een bepaalde grens ligt. Op deze manier kan een achtergrond worden beschreven door meerdere kleuren. Dit komt voor bij bijvoorbeeld bewegende bladeren, golvend water, of bij pixels die een rand met een hoog contrast beschrijven. Deze pixels kunnen afwisselend twee sterk verschillende kleuren beschrijven.

[4]Uit het experimenten in sectie 3.4 blijkt dat de keuze voor deze grenswaarde niet van grote invloed is op de prestaties van het systeem.

De updateformules van het EM algoritme voor een mixture model met parameters π_k, $\vec{\mu}_k$, en $\sigma_k^{2,ij}$ de elementen van de covariantiematrix Σ_k zijn gegeven in de vergelijkingen 2.13, 2.14, en 2.15. De afleiding van deze vergeijkingen is terug te vinden in bijlage A.

$$\pi_{k,t+1} = \pi_{k,t} + \gamma\left(P(k|\vec{x}_{t+1}) - \pi_{k,t}\right) \tag{2.13}$$

$$\vec{\mu}_{k,t+1} = \vec{\mu}_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(\vec{x}_{t+1} - \vec{\mu}_{k,t}) \tag{2.14}$$

$$\sigma^{2,ij}_{k,t+1} = \sigma^{2,ij}_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left(\left(x^i_{t+1} - \mu^i_{k,t+1}\right) \left(x^j_{t+1} - \mu^j_{k,t+1}\right) - \sigma^{2,ij}_{k,t} \right) \tag{2.15}$$

Hierin is $P(k|\vec{x})$ de à posteriori waarschijnlijkheid dat kleurwaarde $\vec{x}$ tot kernel k behoort(zie vergelijking 2.12).

2.3.1 Update formules voor kernels met een diagonale covariantiematrix

Voor het bepalen van de à posteriori waarschijnlijkheid moet een kostbare matrix inversie van de covariantiematrix worden gedaan. Door de benadering te doen dat de covariantie matrix alleen op de diagonaal van 0 afwijkende elementen heeft, wordt de inverse van de covariantiematrix het bepalen van de inverse van de componenten van de covariantiematrixe. De update formules worden dan:

$$\pi_{k,t+1} = \pi_{k,t} + \gamma\left(P(k|\vec{x}_{t+1}) - \pi_{k,t}\right) \tag{2.16}$$

$$\vec{\mu}_{k,t+1} = \vec{\mu}_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(\vec{x}_{t+1} - \vec{\mu}_{k,t}) \tag{2.17}$$

$$\sigma^{2,ii}_{k,t+1} = \sigma^{2,ii}_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left(\left(x^i_{t+1} - \mu^i_{k,t+1}\right)^2 - \sigma^{2,ii}_{k,t} \right) \tag{2.18}$$

2.3.2 Update formules voor sferische kernels

Een andere benadering die kan worden gedaan is de covariantie matrix van de vorm $\Sigma = \sigma^2\mathbf{I}$ te kiezen. De keuze voor dezelfde elementen op de diagonaal van de covariantie matrix resulteert in sferische kernels. Dit geeft een minder nauwkeurige benadering, maar bespaart tijd omdat er minder elementen moeten te worden uitgerekend. De update formules voor d dimensionale kernels zijn:

$$\pi_{k,t+1} = \pi_{k,t} + \gamma\left(P(k|\vec{x}_{t+1}) - \pi_{k,t}\right) \tag{2.19}$$

$$\vec{\mu}_{k,t+1} = \vec{\mu}_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(\vec{x}_{t+1} - \vec{\mu}_{k,t}) \tag{2.20}$$

$$\sigma^2_{k,t+1} = \sigma^2_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left(\frac{\|\vec{x}_{t+1} - \vec{\mu}_{k,t+1}\|^2}{d} - \sigma^2_{k,t} \right) \tag{2.21}$$

2.3.3 Update formules voor ééndimensionale kernels

Door de banadering te doen dat de kleurwaarden per kleurkanaal onafhankelijk van elkaar zijn, kan het kleurverleden van elk kleurkanaal door een ééndimensionaal mixture model worden beschreven. Voor alle kleurkanalen worden aparte kernels gebruikt. De update

formules van het EM algoritme voor kernel k worden dan:

$$\pi_{k,t+1} = \pi_{k,t} + \gamma\left(P(k|x_{t+1}) - \pi_{k,t}\right) \tag{2.22}$$

$$\mu_{k,t+1} = \mu_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|x_{t+1})(x_{t+1} - \mu_{k,t}) \tag{2.23}$$

$$\sigma^2_{k,t+1} = \sigma^2_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left((x_{t+1} - \mu_{k,t+1})^2 - \sigma^2_{k,t}\right) \tag{2.24}$$

2.3.4 Update formules volgens de methode van Stauffer en Grimson

Stauffer en Grimson hebben een alternatieve update methode geintroduceert [8]. Dit is een aanpassing van het EM algoritme die is voorgesteld om de implementatie van het update algoritme sneller te maken. De tijdwinst wordt behaald doordat het gemiddelde en de variantie alleen kernels die kernels worden aangepast waarvan de afstand tussen het gemiddelde van de kernel en de huidige kleurwaarde niet te groot is (bijvoorbeeld tot de grens $3 \cdot \sigma$). Wel wordt de weegfactor van alle kernels aangepast, maar hiervoor wordt een benadering voor de *à posteriori waarschijnlijkheid* gedaan. Voor het EM algoritme is deze gegeven in vergelijking 2.12. De benadering hiervoor is gegeven in vergelijking 2.28. Wanneer de afstand tussen de huidige kleurwaarde en het gemiddelde van een kernel voor alle kernels te groot is, wordt de kernel met de laagste weegfactor π geherinitialiseerd. Deze kernel krijgt een relatief hoge variantie, en een gemiddelde dat gelijk is aan de geregistreerde kleurwaarde. De updateformules van de alternatieve update methode bij gebruik van d kleurkanalen, en dus d-dimensionale kernels, zijn gegeven in de vergelijkingen 2.25 tot en met 2.27.

$$\pi_{k,t+1} = \pi_{k,t} + \gamma(M(k, \vec{x}_{t+1}) - \pi_{k,t}) \tag{2.25}$$

$$\mu^i_{k,t+1} = \mu_{k,t} + \frac{\gamma'}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t}) \tag{2.26}$$

$$\sigma^2_{k,t+1} = \sigma^2_{k,t} + \frac{\gamma'}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left(\|\vec{x}_{t+1} - \vec{\mu}_{k,t+1}\|^2 - \sigma^2_{k,t}\right) \tag{2.27}$$

Hierin is

$$M(k, \vec{x}_{t+1}) = \begin{cases} 1 & als \quad \vec{x}_{t+1} \in k \\ 0 & anders \end{cases} \tag{2.28}$$

en

$$\gamma' = \gamma p(\vec{x}_{t+1}). \tag{2.29}$$

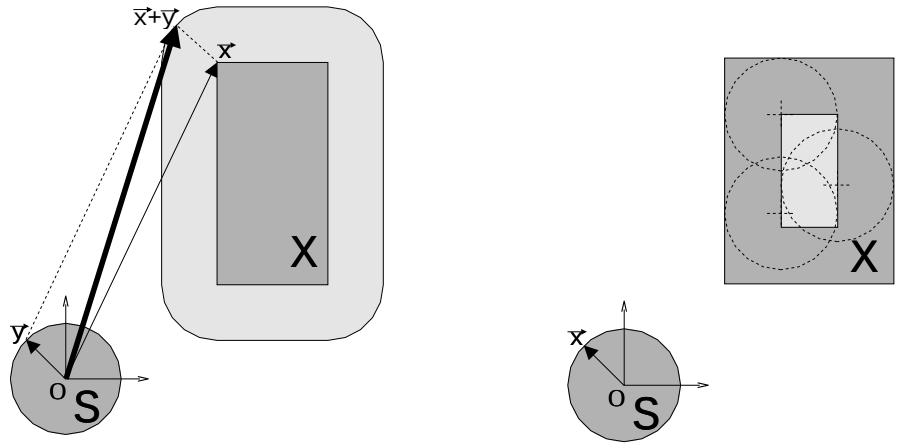
Stauffer en Grimson maken gebruik van sferische kernels voor de implementatie van het mixture model. De update formule voor het aanpassen van de variantie elementen van het mixture model wijkt echter een factor $1/d$ af van het originele EM algoritme met gebruik van sferische kernels. Dit zorgt voor een sterkere verandering van de variantie parameters van het mixture model dan bij gebruik van een EM algoritme.

De benadering die wordt gedaan voor de à posteriori waarschijnlijkheid op kleurwaarde $\vec{x}$ laat de kleurwaarden die het verst buiten de kernels vallen buiten beschouwing. Dit zal resulteren in een kleinere waarde voor de variantie, en dus 'gepiektere' kernels.

2.4 Postprocessing

2.4.1 Binaire morfologie

Het resultaat van de object detectie bevat nog ongewenste eigenschappen. De segmentatie vertoont ruis in de achtergrond, en gaten in de objecten. Er wordt gebruik gemaakt van



(a) MinkowskiAdditie

(b) MinkowskiVerschil

Figuur 2.1: Minkowski additie en Minkowski verschil. Het licht gekleurde gebied is het resultaat van respectievelijk de minkowski additie van gebied X met structurerend element S(figuur 2.1(a)), en het minkowski verschil van gebied X met structurerend element S(figuur 2.1(b)).

enkele simpele binaire morfologische operaties om het resultaat van de objectclassificatie te verbeteren. Deze zijn onder andere beschreven door Smeulders en van den Boomgaard [7].

Binaire mathematische morfologie beschrijft de morfologische operaties op binaire beelden. De basisoperaties van de morfologie zijn dilatie en erosie. Deze operaties beschrijven een interactie tussen de verzameling van alle objectpixels X in het oorspronkelijke beeld met de verzameling pixels van het *structurerend element* S. S is gedefinieerd met een oorsprong, de vorm van S is bepalend voor het resultaat van de morfologische operaties. Dilatie en erosie zijn gedefinieerd in termen van *Minkowski additie* en *Minkowski verschil.*

Minkowski additie is gedefinieerd als

$$X \oplus S = \{\vec{x} + \vec{y} | \vec{x} \in X, \vec{y} \in S\} \tag{2.30}$$

$$= \bigcup_{\vec{x} \in S} X_{\vec{x}} \tag{2.31}$$

Dit is de vereniging van alle getransleerde verzamelingen $X_{\vec{x}}$, waarbij $\vec{x}$ een willekeurige vector uit S is(zie figuur 2.1(a)).

Minkowski verschil is gedefinieerd als

$$X \ominus S = \{\vec{x} | S_{\vec{x}} \subseteq X\} \tag{2.32}$$

$$= \bigcap_{\vec{x} \in S} X_{\vec{x}} \tag{2.33}$$

Het Minkowski verschil bestaat uit alle punten $\vec{x}$ waarvoor S gecentreerd in dit punt geheel binnen X ligt(zie figuur 2.1(b)).

Dilatie is gedefinieerd als:

$$\delta_S(X) = X \oplus \check{S}. \tag{2.34}$$

Erosie is gedefinieerd als:

$$\epsilon_S(X) = X \ominus \check{S}. \tag{2.35}$$

Waarbij $\check{S}$ de gespiegelde is van S in de oorsprong.

Het effect van het toepassen van een dilatie op een binaire afbeelding is het 'groeien' van de objecten in de afbeelding. Omgekeerd is het effect van een erosie op een binaire afbeelding het 'krimpen' van de objecten. De mate waarin de objecten groeien en krimpen is afhankelijk van de keuze voor het structurerend element. De vorm van dit structurerend element is afhankelijk van het doel voor ogen. Elementen met dezelfde vorm als het structurerend element, zullen door de morfologische operatie worden benadrukt. Bij dit onderzoek worden de morfologische operaties gebruikt voor het verwijderen van de ruis uit het resultaat van de voorgrond- achtergrondsegmentatie. Deze ruis bestaat uit losse pixels. Er is gekozen voor een simpel 3×3 structurerend element met op alle 9 posities de waarde 1.

Door het toepassen van een erosie wordt ruis bestaande uit één pixel verwijderd, maar ook objecten worden vanaf de randen een pixel kleiner gemaakt. Om dit te herstellen wordt hierna een dilatie toegepast om het oorspronkelijke formaat van de objecten te herstellen. Dit wordt ook wel een opening genoemd, zie vergelijking 2.36.

$$\begin{aligned} X \circ S &= \epsilon_S(X) \oplus S \\ &= \left(X \ominus \check{S}\right) \oplus S \end{aligned} \tag{2.36}$$

Om gaten in objecten te verwijderen kan een dilatie worden toegepast. Dit heeft echter weer het effect dat objecten worden vervormd, namelijk langs de randen worden deze iets groter. De correctie hiervoor wordt gedaan aan de hand van een erosie. Deze volgorde wordt ook wel sluiting genoemd, zie vergelijking 2.37.

$$\begin{aligned} X \bullet S &= \delta_S(X) \ominus S \\ &= \left(X \oplus \check{S}\right) \ominus S \end{aligned} \tag{2.37}$$

Achtereenvolgens wordt dus eerst een opening toegepast om ruis in de achtergrond te verwijderen, en vervolgens een sluiting om gaten in de objecten te verwijderen.

Bij het gekozen structurerend element (een 3×3 met op alle 9 posities de waarde 1) is een efficiente implementatie van dilatie en erosie mogelijk. Hierbij wordt voor elk pixel bekeken of deze als object is geclassificeerd. Als dit het geval is, wordt gekeken of de buren van het betreffende pixel moeten worden aangepast. Alleen de buren van het pixel vallen binnen het structurerend element. De buren van een pixel worden volgens een 8-geconnecteerde omgeving bepaald, overeenkomstig met de vorm van het structurerend element. Hierbij worden alle acht aangrenzende pixels als buur van een pixel beschouwd (dus ook die pixels die schuin boven, en schuin onder liggen).

2.4.2 Labelling

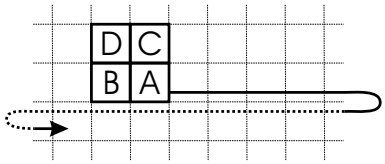
Een geconnecteerde groep voorgrondpixels wordt een *blob* genoemd. Een blob geeft aan welke pixels in het beeld een object beschrijven. Het is de bedoeling dat verschillende objecten ook als verschillende objecten worden gedetecteerd. Afzonderlijke blobs worden daarom met verschillende waarden gelabeld.

Voor de labelling van de objecten is gebruik gemaakt van een labelalgoritme zoals dat is beschreven door Horn [5]. Bij dit algoritme wordt het beeld rij voor rij van links naar rechts,

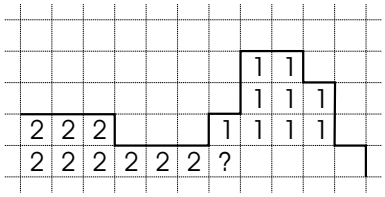
en van boven naar beneden doorlopen. Als wordt gekeken naar een pixel op locatie A, dan is informatie nodig over de labelwaardes van de pixels op locatie B, C, en D(zie figuur 2.2. De volgende regels worden nu toegepast:

1. Als pixel A achtergrond is, doe niets.
2. Als pixel A voorgrond is en D is gelabeld, copieer het label van D.
3. Als pixel A voorgrond is en B is gelabeld en C is niet gelabeld, copieer dan het label van B.
4. Als pixel A voorgrond is en C is gelabeld en B is niet gelabeld, copieer dan het label van C.
5. Als pixel A voorgrond is en B, C, en D zijn niet gelabeld, geef A een nieuw label.
6. Als pixel A voorgrond is en zowel B als C zijn hetzelfde gelabeld, copieer dan het label van B.
7. Als pixel A voorgrond is en B en C zijn verschillend gelabeld (zie figuur 2.3), stel label B gelijk aan label C en copieer label B.

Alle labels die bij bovenstaande regels aan elkaar gelijk zijn gesteld, beschrijven hetzelfde object.



Figuur 2.2: De labelwaarde van het pixel op positie A is gebaseerd op de waarden die al zijn toegekend aan de pixels op posities B, C, en D.



Figuur 2.3: Bij het labellen kan het gebeuren dat twee labels aan elkaar gelijk moeten worden gesteld(zie stap 7 van het label algoritme)

Hoofdstuk 3

Experimenten

In dit hoofdstuk worden een aantal experimenten gedaan om de methoden van Stijnman en Stauffer met onze methode te vergelijken, en de juiste keuze te maken voor update methode en implementatie van het mixture model. Er wordt gekeken of de methoden in staat zijn om de parameters van het mixture model van een pixel juist aan te passen, zodat dit mixture model een goede benadering van de kleurgeschiedenis van het betreffende pixel geeft. Er wordt gebruik gemaakt van een kunstmatig gegenereerde sampleset. Het voordeel van het gebruik van kunstmatig gegenereerde samples boven de werkelijke kleurwaarden uit videobeelden, is dat het resultaat van de updatemethoden beter kan worden vergeleken met de input. Bij het genereren van kunstmatig gegenereerde samples zijn de waarden voor het gemiddelde en de variantie van de sets bekend. Van het kleurverleden van een pixel afkomstig uit een videoregistratie, is het niet mogelijk een preciese waarde voor gemiddelde en variantie van alle voorgekomen kleuren te geven. Er worden verschillende samplesets gebruikt om de kleurgeschiedenis van een pixel in video te simuleren. Zo kan er bijvoorbeeld één sampleset worden gebruikt om een achtergrond te beschrijven, en een tweede sampleset om een objectkleur te beschrijven. Om de beide methoden juist te testen, moeten de samplesets zo goed mogelijk de kleurwaarden zoals deze in echte videobeelden voorkomen benaderen. Er wordt daarom eerst bepaald wat de ruis in de kleurwaarde van werkelijk video sequenties is.

3.1 Bepaling van de ruis in de kleurwaarde bij gebruik van videosequenties

De kunstmatig gegenereerde samplesets zullen een benadering van een werkelijke situatie zijn. Om deze benadering zo realistisch mogelijk te maken wordt hier een testvideo van een werkelijke situatie geanaliseerd. Daartoe wordt een surveillance video van de vertrekhal van Schiphol gebruikt. De beelden van deze video hebben een licht beige achtergrond van de tegelvloer van de hal. De objecten in deze video zijn vooral donker gekleurd.

3.1.1 Opzet

Van twee pixels van de testvideo worden de kleurwaarden per kleurkanaal in de tijd geregistreerd. Om het overzicht te behouden zijn alleen van de eerste 600 frames van de video gebruikt. Eén van de pixels ligt in een gebied waarin regelmatig objecten passeren. In het gebied van het tweede pixel passeren geen objecten. Van alle twee pixels worden de kleurwaarden van de R, G, en B kleurkanalen en van de genormaliseerde r en g kanalen geregistreerd,

en uitgezet tegen de tijd. Ook worden er kleurhistogrammen geplot van de geregistreerde kleurwaarden.

3.1.2 Resultaten

Figuur 3.1: Frame uit de video met de posities van de te analiseren pixels

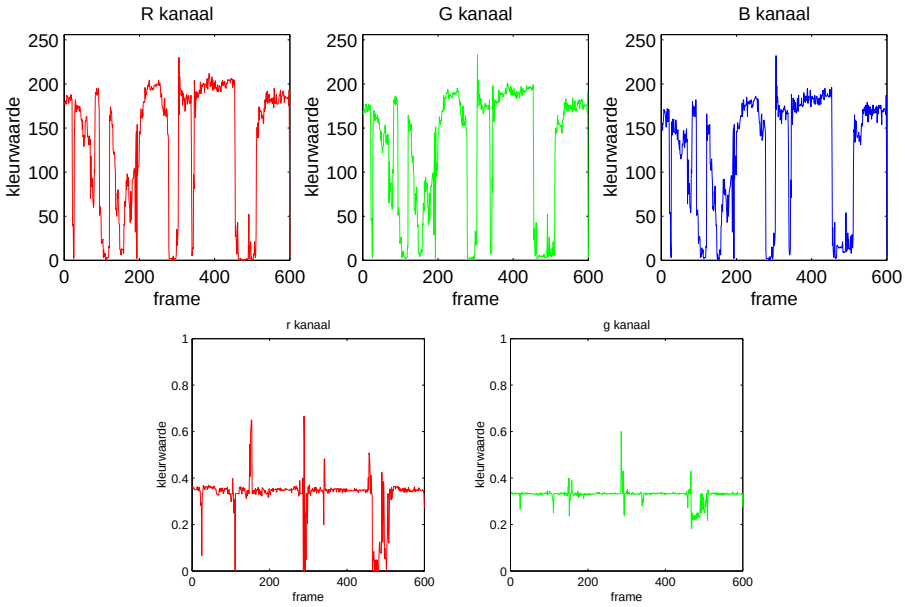
In figuur 3.1 is één frame van een van de testvideos te zien. Van deze video is op twee plaatsen de kleurgeschiedenis van de betreffende pixels vastgelegd. Het eerste pixel bevindt zich op locatie $x = 100$, $y = 100$, hier passeren geregeld objecten. Het tweede pixel is die op locatie $x = 46$, $y = 8$. Deze positie laat gedurende de hele video achtergrond zien.

In de figuur 3.2 is het kleurverleden van pixel 1 weergegeven voor de kleurkanalen R, G, B, r, en g. Hetzelfde is gegeven voor pixel 2 in de figuur 3.4. In de figuren 3.3 en 3.5 zijn de kleurhistogrammen van pixel 1 en pixel 2 gegeven. In figuur 3.6 is een gedeelte van de kleurwaarden voor pixel 1 gegeven. Deze kleurwaarden zijn genomen in een periode waarin een object gepasseerd is.

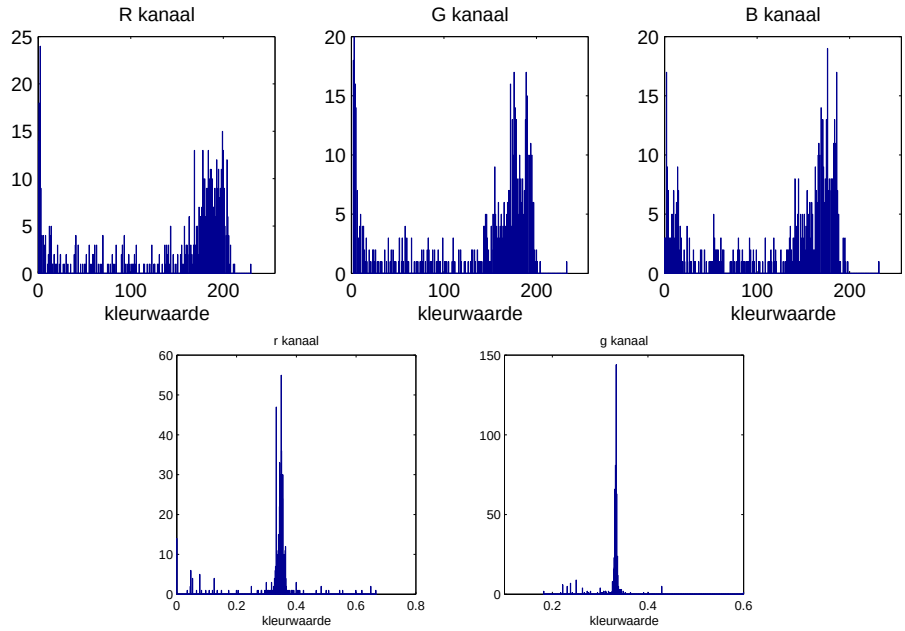
In de figuren 3.7 en 3.8 zijn de genormaliseerde r en g kanalen tegen elkaar uitgezet. In figuur 3.7 zijn de kleurwaarden gegeven tijdens het passeren van één object. In figuur 3.8 zijn de kleurwaarden van de achtergrond gegeven.

3.1.3 Discussie

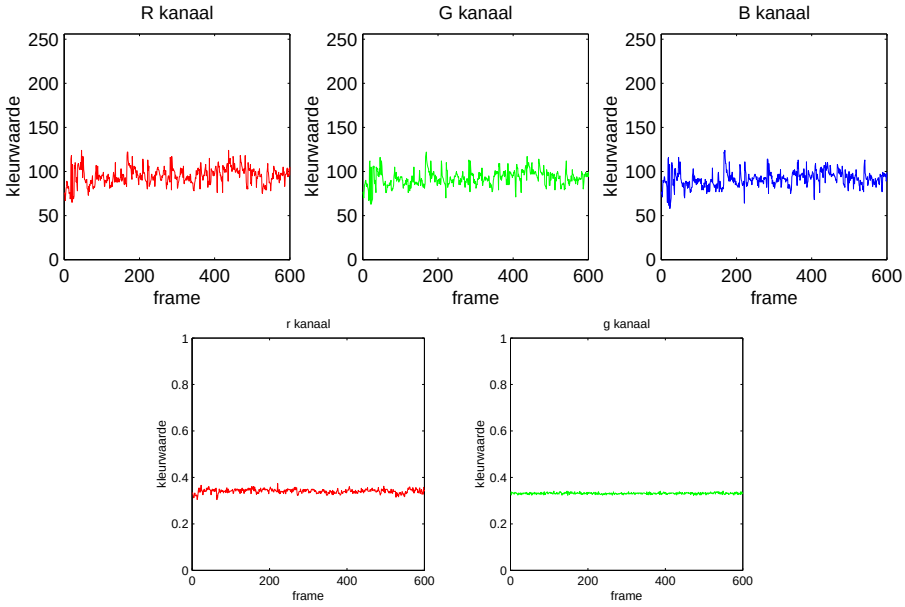
De originele RGB kleurwaarden die door de framegrabber worden geleverd zijn omgezet naar het genormaliseerde rgb kleurmodel, waarvan alleen de r en g kleurkanalen worden gebruikt.



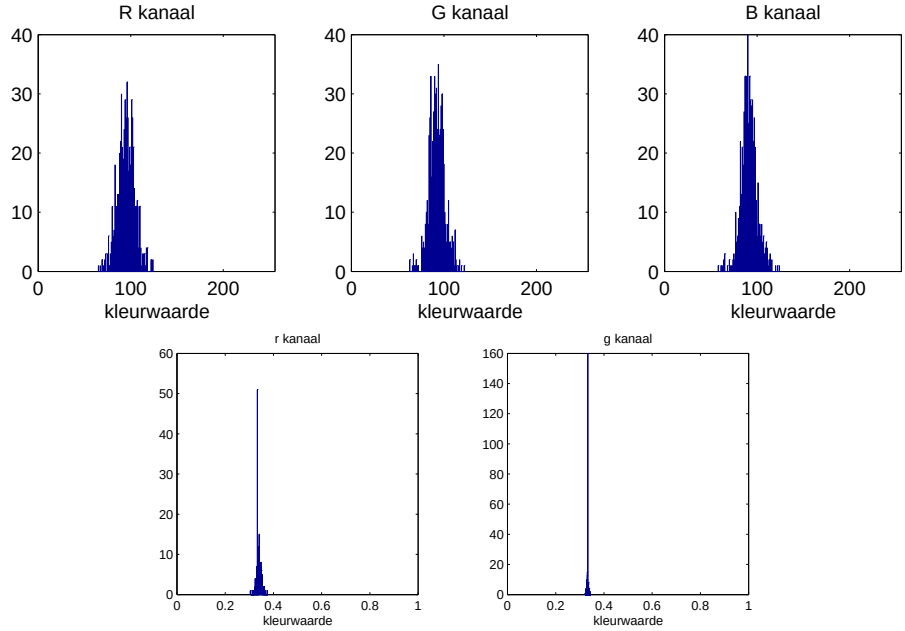
Figuur 3.2: Kleurverleden van pixel 1 voor verschillende kleurkanalen.



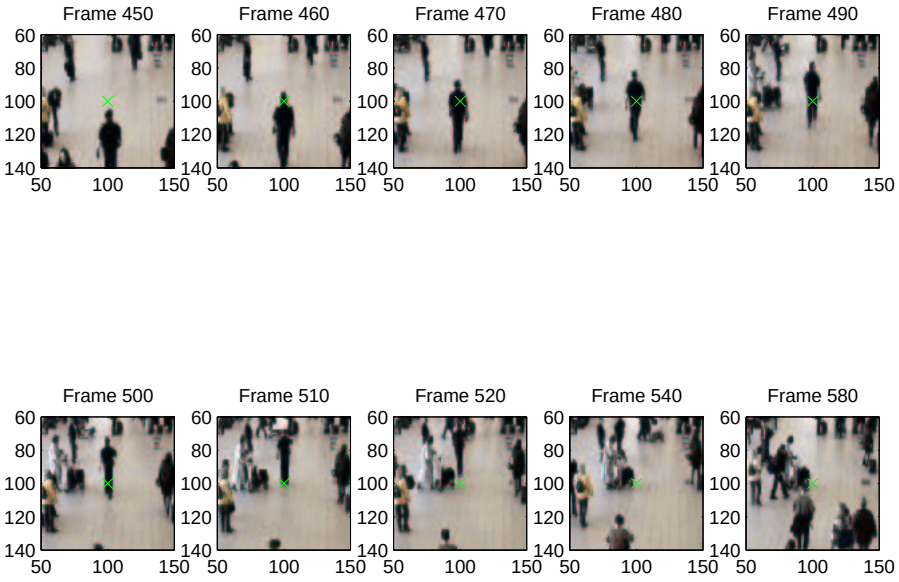
Figuur 3.3: Kleurhistogrammen van pixel 1.



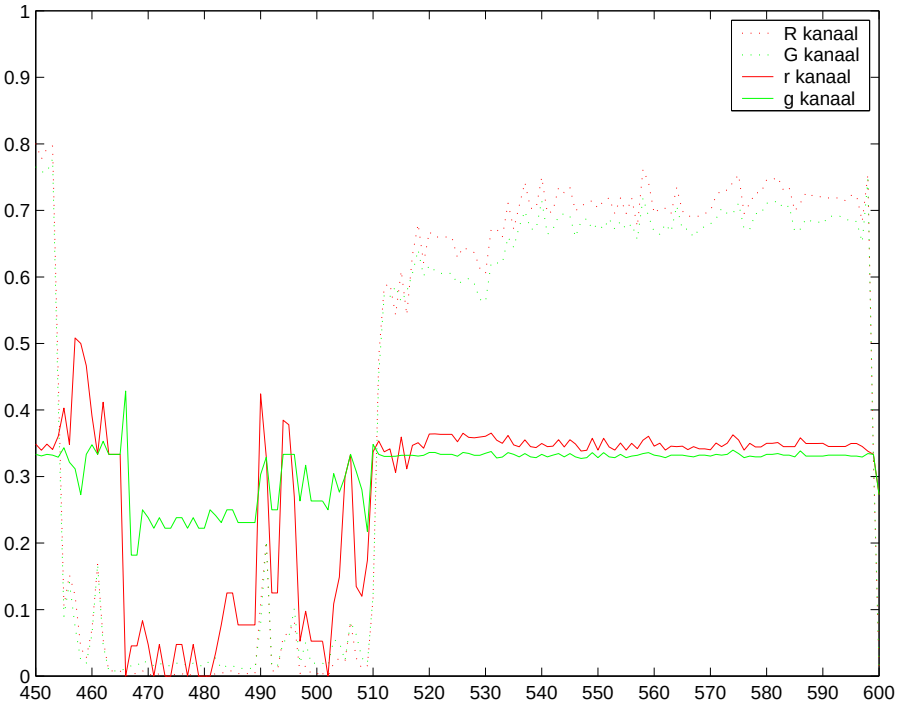
Figuur 3.4: Kleurverleden van pixel 2 voor de verschillende kleurkanalen.



Figuur 3.5: Kleurhistogrammen van pixel 2.

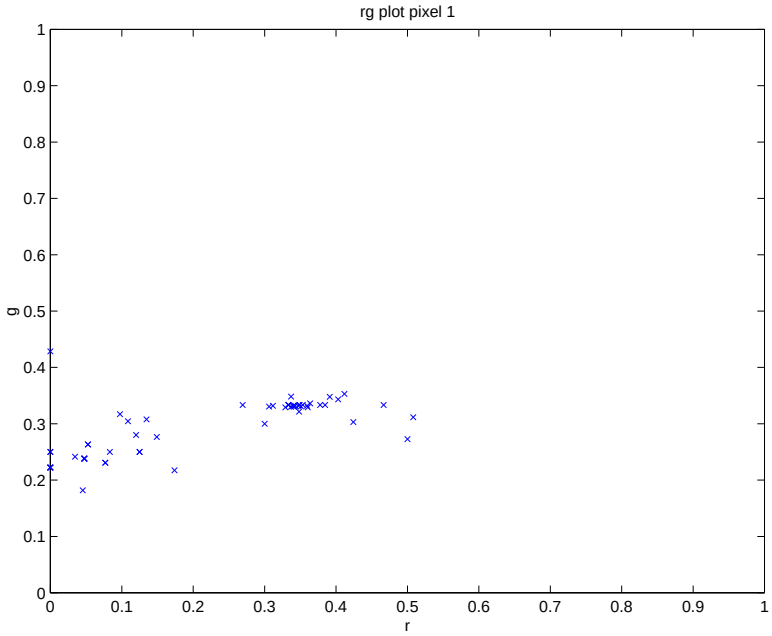


(a) Beelden van een passerend object.



(b) Kleurwaarden van de R, G, r, en g kleurkanalen bij een passerend object

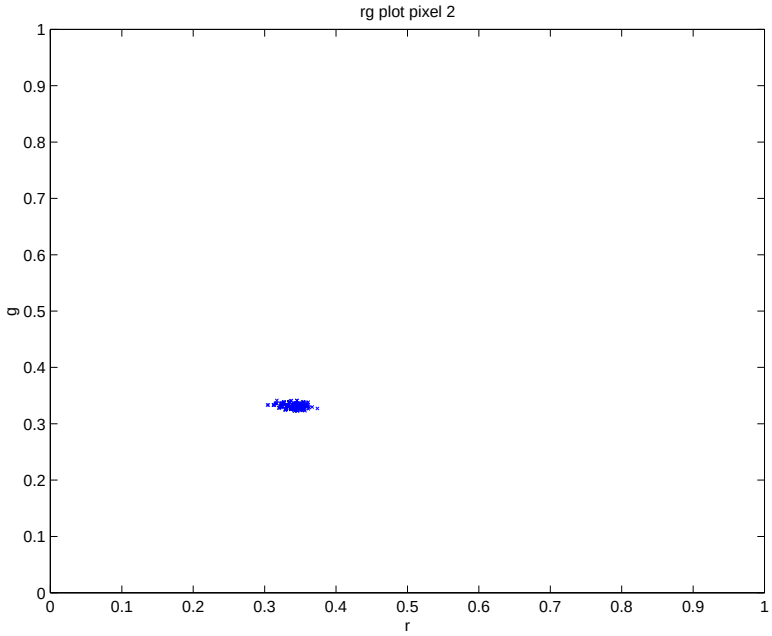
Figuur 3.6: Vergelijking van RG en rg kleurkanalen bij een passerend object. De waarden van de R en G kleurkanalen zijn hier ter vergelijking op 1 genormaliseerd.



Figuur 3.7: rg plot voor pixel 1. Dit zijn de kleurwaarden van pixel 1 van frame 450 t/m 520. In deze periode is een donkergekleurd object pixel 1 gepaseerd. Het resultaat is een kleurverdeling met een grote variantie.

In de figuren 3.2 en 3.4 zijn de verschillende kleurkanalen uitgezet tegen de tijd. In deze figuren is te zien dat de kleurkanalen van R, G, en B vrijwel hetzelfde verloop hebben. In figuur 3.4 wordt de achtergrondkleur beschreven. Voor het R,G, en B kanaal worden den kleurwaarden van de achtergrond beschreven door een verdeling met een fluctuerende gemiddelde waarde voorzien van ruis. Voor de r, en g kanalen is de ruis duidelijk minder. Ook de fluctuatie van de gemiddelde achtergrondkleur is bij de r en g kleurkanalen niet terug te vinden. In figuur 3.2 is het kleurverleden weergegeven bij het passeren van objecten. Wanneer een (in dit geval donkerkleurig) object wordt beschreven, wordt de kleurwaarde van zowel het R, G, als het B kanaal duidelijk lager. Het valt op dat de randen van deze 'dalen' in de grafieken van het kleurverleden niet altijd 'schrp' zijn. In het kleurverloop van de r en g kanalen is een duidelijke, niet variërende waarde voor de achtergrondkleur te zien. Op plaatsen waar in de R, G, en B kanalen een objectkleur werd geregistreerd als een sterke uitwijking naar beneden, wordt in de r, en g kanalen een smallere, minder sterke uitwijking naar boven of beneden gevonden. Wel hebben deze uitwijkingen in de kleurwaarde scherpe randen.

Eén van de passages van een object is weergegeven in figuur 3.6. In deze figuur zijn de kleurwaarden gegeven van de R, G, r en g kanalen. Hieruit blijkt dat alle kanalen ongeveer gelijk reageren op een passerend object. De verandering van de R en G kanalen is echter een stuk sterker dan die in de r en g kanalen. Het pixel waarbij het object passeert wordt bekeken gedurende frame 460 tot frame 510. De kleurwaarden van de r en g kanalen worden gedurende het passeren van het object een paar keer gelijk aan de kleurwaarde van het pixel tijdens het beschrijven van de achtergrond. Dit is ondermeer het geval rond frame 460 en 490. Het lijkt precies de positie te zijn die op dat moment op de rand van het object valt.



Figuur 3.8: rg plot voor pixel 2. Dit pixel beschrijft de achtergrond. In de rg kleurruimte vormen de kleurwaarden een compact cluster. De variantie van de kleurwaarden van het r kanaal is $\sigma_r^2 = 8,3 \cdot 10^{-5}$. De variantie van het g kanaal is $\sigma_g^2 = 1,2 \cdot 10^{-5}$

Bij frame 460 is dat in de nek van de passerende persoon, bij frame 490 tussen de benen van de persoon.

Verder nemen de kleurwaarden van het r en g kanaal na het passeren van het object direct weer de waarde van de achtergrondkleur aan. De kleurwaarden van het R en G kanaal blijven nog zo'n 30 frames afwijken van de waarde van de achtergrondkleur. Dit is ongeveer van frame 510 tot 540 het geval. Uit de bijbehorende beelden van de video blijkt dat dit de schaduw van het gepasseerde object is die wordt geregistreerd.

Van de gegevens van de R, G, B en de genormaliseerde r en g kanalen zijn de kleurhistogrammen bepaald, zie figuur 3.3 en figuur 3.5. In de kleurhistogrammen behorende bij de genormaliseerde r en g kanalen van pixel 1 is een scherpe, hoge piek te zien. Naast deze piek zijn er ook een aantal kleinere pieken te zien. Deze kleinere pieken behoren tot de objectkleuren. De kleurhistogrammen behorende bij de genormaliseerde r en g kanalen van pixel 2 bestaan uit één scherpe piek (zie figuur3.5). De kleurhistogrammen van de genormaliseerde r en g kanalen van pixel 2 bestaan uit een enkele scherpe piek, het kleurverleden van deze kanalen van pixel 2 beschrijft dus één kleur: de achtergrondkleur. Van de kleurwaarden van de r en g kleurkanalen van pixel 2 zijn de varianties bepaald. Voor het r kanaal is $\sigma_r^2 = 8,3 \cdot 10^{-5}$. Voor het g kanaal $\sigma_g^2 = 1,2 \cdot 10^{-5}$.

In de rg plot van pixel 1 in figuur 3.7 is een grote variantie in de waargenomen kleurwaarden te zien. In de periode dat deze kleurwaarden zijn geregistreerd, is één donkergekleurd object het pixel gepaseerd. In figuur 3.8 is te zien dat alleen achtergrond één compacte cluster in de rg ruimte geeft. De variantie van de waargenomen kleurwaarden verschilt wel per kleurkanaal.

3.1.4 Conclusie

Na bestudering van de kleurhistogrammen van een achtergrondpixel maken we de aanname dat de kleurwaarden van een kleurkanaal normaal verdeeld zijn wanneer het pixel achtergrond representeert. De variantie van de verdeling is in de orde van 10^{-5} voor een genormaliseerd r of g kanaal. De verdeling voor het beschrijven van een objectkleur heeft een hogere variantie.

Er is gekozen voor het rg kleurmodel omdat is gebleken dat dit model ongevoelig is voor schaduw en andere belichtingsomstandigheden. Ook blijkt dit model minder variantie te vertonen in de geregistreerde kleurwaarden. Dit maakt dit kleurmodel geschikter voor het beschrijven van de achtergrondkleur. Nadeel van dit kleurmodel is dat het instabiel wordt bij lage intensiteiten. Dit is besproken in sectie 2.1, en is te zien aan de grote variantie in de kleurwaarden in figuur 3.7. Dit kan resulteren in een slechte detectie van objecten voor een donkere achtergrond, of in 'gaten' bij het detecteren van donkere objecten.

3.2 Update methoden voor het mixture model

Bij dit experiment wordt bekeken op welke manier de updatemethoden van Stijnman en van den Boomgaard [3] met het EM algoritme, en van Stauffer en Grimson [8] met een alternatief update algoritme, de parameters van een mixture model aanpassen. Dit wordt gedaan door kleurwaarden aan te bieden die zijn genomen uit een random gegenereerde sampleset.

3.2.1 Opzet

Het experiment dat hier gedaan wordt, is een simulatie van een video van 1000 frames waarvan één pixel is bekeken met een constante achtergrondkleur. De kleurwaarde van één van de kleurkanalen van dit pixel ligt op een kwart van de maximale kleurwaarde. De variantie van de ruis van dit kleurkanaal is $1 \cdot 10^{-5}$ genomen. Dit is in de orde van de ruis van de geregistreerde achtergrondkleurwaarden van video, zoals gevonden in het experiment in sectie 3.1.

Bij dit experiment wordt één kleurkanaal gesimuleerd. Er wordt een sampleset van 1000 samples aangeboden. De samples uit deze set zijn normaal verdeeld met een gemiddelde van $\mu = 0.25$ en een variantie van $\sigma^2 = 1.0 \cdot 10^{-5}$. Beide methoden krijgen dezelfde samples te verwerken. Elke keer na het aanbieden van één sample passen beide methoden de parameters van het bijbehorende mixture model aan met een learnrate van $\gamma = 0.01$. Het mixture model bestaat hier uit één kernel. Na het aanbieden van elk sample moeten twee parameters aangepast worden, het gemiddelde μ, de variantie σ^2. De parameter voor de weegfactor π is niet interesant omdat deze altijd gelijk aan 1 zal zijn bij het gebruik van 1 kernel. Na het aanbieden van de complete sampleset wordt het eindresultaat van beide methoden vastgelegd. Dit experiment wordt 100 keer herhaald zodat het gemiddelde en de spreiding van de resultaten van de beide methoden kunnen worden vergeleken.

3.2.2 Resultaten

In figuur 3.9(a) is het resultaat te zien van één van de experimenten met het EM algoritme. Hierin zijn de samples gegeven, en het verloop van de parameters van de kernel. De doorgetrokken lijn geeft het gemiddelde van de kernel na het aanbieden van een sample. De stippellijnen geven de '3σ-grens', van de kernel aan. In figuur 3.9(b) zijn de resultaten van de waarden voor het gemiddelde en de variantie van de kernel in histogramvorm weergegeven. Het gemiddelde en de varianties van deze histogrammen zijn gegeven in tabel 3.1. Dezelfde

	μ	σ^2
1 kernel		
Sample set	$2.5 \cdot 10^{-1}$	$1.0 \cdot 10^{-5}$
Resultaat EM algoritme	$2.501 \cdot 10^{-1}(3 \cdot 10^{-4})$	$9.7 \cdot 10^{-6}(1 \cdot 10^{-6})$
Resultaat methode Stauffer	$2.50 \cdot 10^{-1}(3 \cdot 10^{-3})$	$3.4 \cdot 10^{-3}(2 \cdot 10^{-3})$

Tabel 3.1: Resultaten van simulatie van het leren van de achtergrond. Gegeven zijn de gemiddelde waarden, en tussen haakjes de bijbehorende variantie

resultaten met gebruik van de update methode van Stauffer en Grimson zijn gegeven in de figuren 3.10(a) en 3.10(b) en tabel 3.1.

3.2.3 Discussie

In de figuren 3.9(a) en 3.10(a) is te zien hoe de beide methoden de paramters van het mixture model aanpassen aan de gegeven samples voor één iteratie van het experiment. De methode van Stijnman met het EM algoritme, past de parameters van het mixturemodel aan zodat deze met een vloeiend verloop richting het gewenste evenwicht convergeren. Namelijk naar de waarden die de sampleset heeft voor gemiddelde en variantie. De methode van Stauffer met het alternatieve update algoritme convergeert de parameters van de kernel sneller naar het gemiddelde van de sampleset. Verder is de uiteindelijke waarde van de variantie te klein. Als gevolg hiervan wordt de kernel opnieuw geïnitialiseerd wanneer een sample buiten de kernel valt. Dit is te zien aan de sprongen in de waarden van de parameters van het mixturemodel wanneer deze in de buurt van de waarden van de sampleset komen.

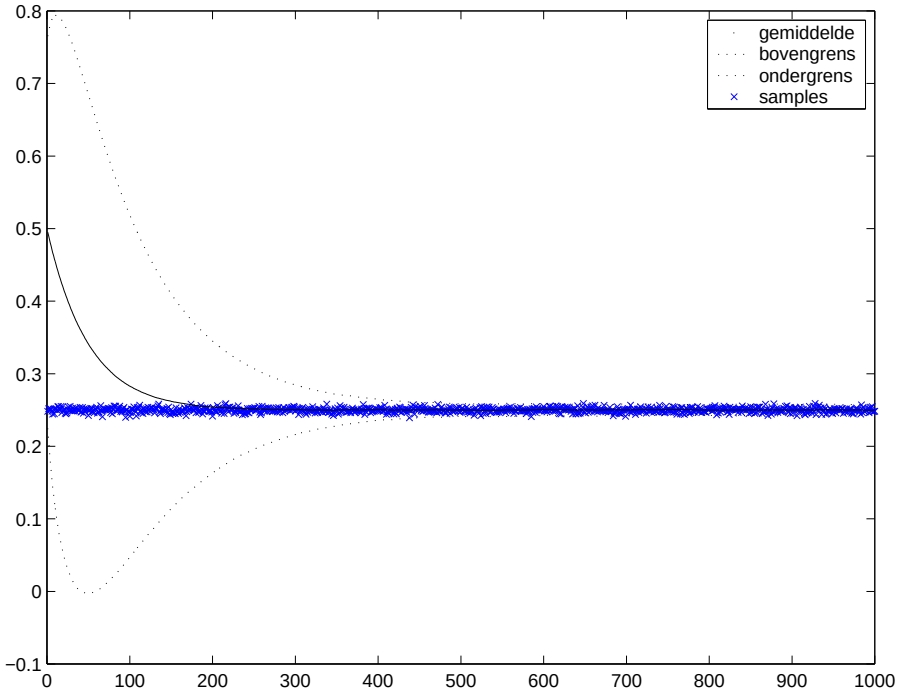
In tabel 3.1 zijn de waarden van de parameters van het mixture model na het aanbieden van alle samples gegeven. Hieruit blijkt dat het mixture model met gebruik van het EM algoritme dat gebruikt is door Stijnman en van den Boomgaard de waarden voor gemiddelde en variantie van de sampleset goed benaderd. Bij gebruik van de updatemethode van Stauffer en Grimson wordt alleen de waarde voor het gemiddelde goed benaderd, niet de waarde voor de variantie.

Dat de variantie van de kernel te klein wordt is een gevolg van de updatemethode die Stauffer en Grimson hebben toegepast. De samples die het verst van het gemiddelde van de kernel liggen worden niet aan de kernel toegekend, en zorgen niet voor een aanpassing van de kernelparameters. Omdat niet bij alle samples het mixture model wordt aangepast, wordt een verkeerde schatting van de variantie van de complete set gemaakt.

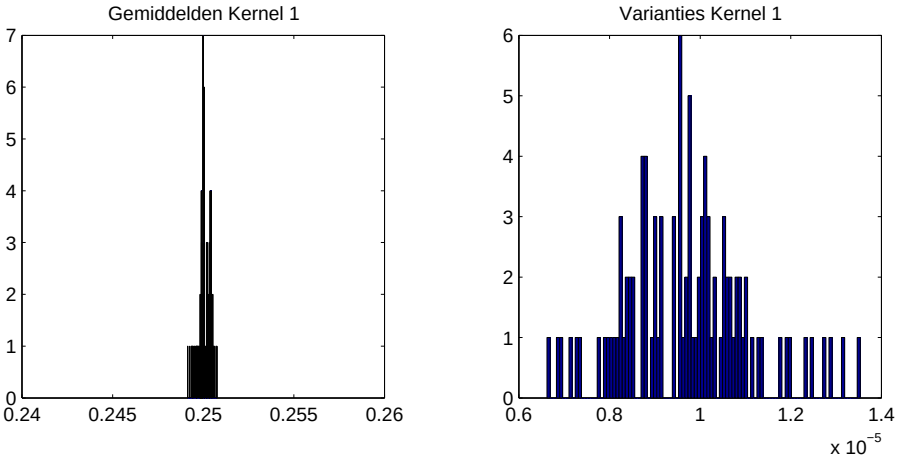
3.2.4 Conclusie

De methode van Stijnman en van den Boomgaard [3] met het originele EM algoritme [2] past de parameters van het mixturemodel op een nette manier aan. De uiteindelijke resultaten geven een goede benadering van de waarden van de sampleset.

De update methode van Stauffer en Grimson [8] geeft niet de juiste resultaten. Het gemiddelde van de sampleset wordt wel goed geschat, maar de variantie van de sampleset wordt niet goed benaderd. Deze updatemethode zal niet goed in staat zijn om een model van de achtergrond te benaderen. Over het algemeen zullen te veel kleurwaarden als achtergrond worden geclassificeerd. We besluiten deze methode niet verder te gebruiken.

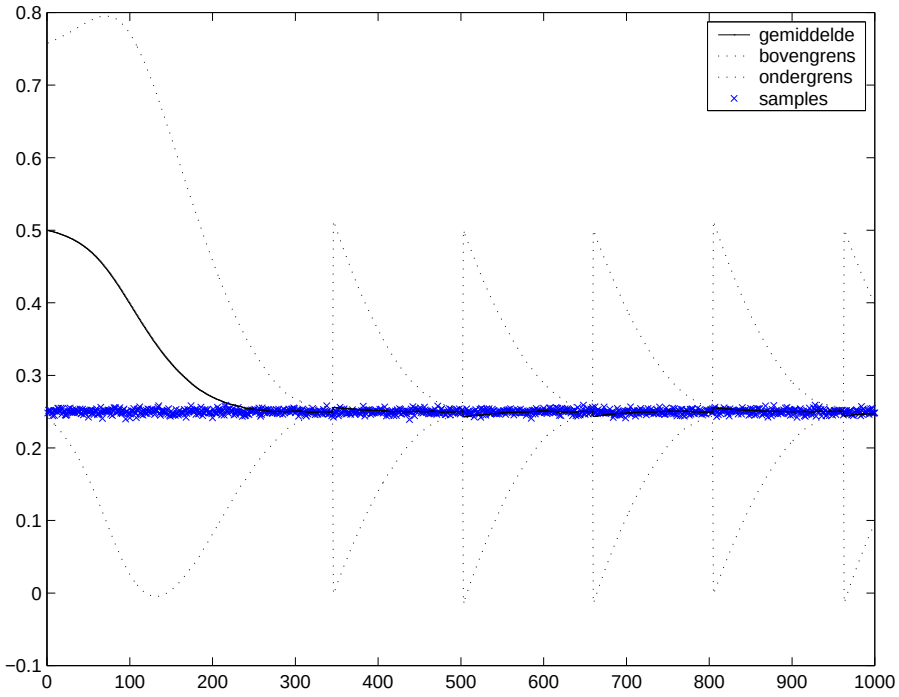


(a) Het verloop van de kernelparameters bij toepassing van het EM algoritme.

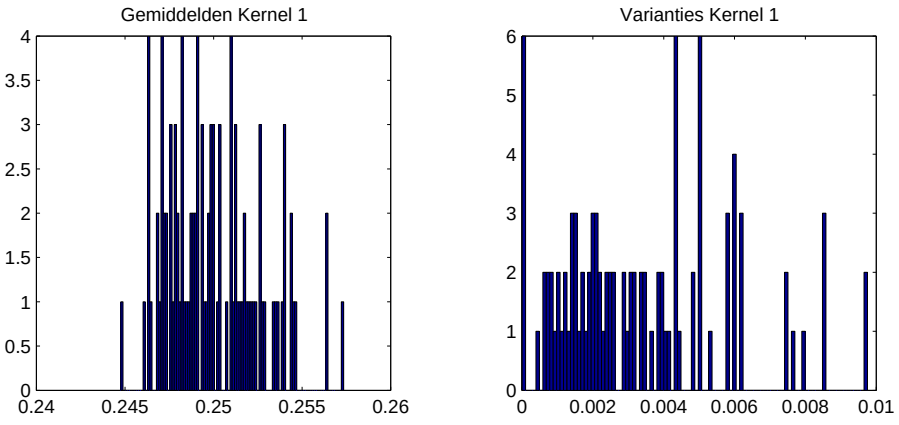


(b) Histogrammen van de uiteindelijke waarden van de kernelparamters met gebruik van het EM algoritme.

Figuur 3.9: Verloop en resultaat van de kernelparameters bij gebruik van het EM algoritme



(a) Het verloop van de kernel parameters bij toepassing van het update alogoritme van Stauffer en Grimson.



(b) Histogrammen van de uiteindelijke waarden van de kernelparameters met gebruik van de update methode van Stauffer en Grimson, na 100 herhalingen.

Figuur 3.10: Verloop en resultaat van de kernelparameters bij gebruik van de alternatieve update methode van Stauffer en Grimson.

3.3 Representaties van de verdeling van kleurwaarden

Er zijn verschillende manieren om de verdeling van kleurwaarden te representeren. Dit experiment wordt gedaan om te kijken hoe de verschillende representaties in staat zijn om de kleurverdeling van een pixel te benaderen. De volgende representaties worden hierbij vergeleken.

- **Representatie 1** Gebruik van ééndimensionale kernels voor het beschrijven van het kleurverleden van een pixel. De aanname wordt gedaan dat de kleurkanalen onafhankelijk zijn van elkaar. Voor elk kleurkanaal wordt zo een ééndimensionaal mixture model gebruikt. De update vergelijkingen van het EM algoritme voor het aanpassen van de parameters van de mixture models zijn gegeven in sectie 2.3.3.
- **Representatie 2** Het benaderen het kleurverleden met een mixture van sferische D-dimensionale Gaussische verdelingen. Hierbij is D het aantal dimensies van de gebruikte kleurruimte. De covariantiematrix van de Gaussische verdelingen is van de vorm $\Sigma = \sigma^2\mathbf{I}$, waarbij $\mathbf{I}$ de eenheidsmatrix is. De update vergelijkingen van het EM algoritme voor deze representatie is gegeven in sectie 2.3.2.
- **Representatie 3** Het benaderen van het kleurverleden met een mixture van D-dimensionaleGaussische verdelingen. Hierbij is D het aantal dimensies van de gebruikte kleurruimte. De covariantie matrix van een Gaussische verdeling bestaat uit een diagonaal matrix waarbij afwijkende elementen op de diagonaal kunnen staan. De update vergelijkingen van het EM algoritme voor deze representatie zijn gegeven in sectie 2.3.1.

3.3.1 Opzet

De parameters van de bovengenoemde representaties worden aangepast met het EM algoritme zoals dit is beschreven in sectie 2.3. Hierbij krijgen alle representaties dezelfde samples aangeboden. De samples zijn genomen uit een aantal verschillende samplesets. Deze samplesets simuleren de kleur van een object, of de kleur van de achtergrond. De samples uit deze sets zijn twee dimensionaal, en normaal verdeeld met een gemiddelde van $\vec{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$ en een variantie van $\Sigma = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}$. Er is gebruik gemaakt van één samplset voor het simuleren van een achtergrondkleur, en drie samplesets voor het simuleren van de objectkleuren. De eigenschappen van deze samplesets zijn samengevat in tabel 3.2. Het totaal aantal aangeboden samples is opgedeeld in de volgende 7 intervallen:

- **Interval 1** 6000 samples uit de achtergrondverdeling.
- **Interval 2** 500 samples uit objectverdeling1.
- **Interval 3** 500 samples uit de achtergrondverdeling.
- **Interval 4** 500 samples uit objectverdeling2.
- **Interval 5** 500 samples uit de achtergrondverdeling.
- **Interval 6** 500 samples uit objectverdeling3.
- **Interval 7** 1500 samples uit de achtergrondverdeling.

De parameters van elke representatie worden met het EM algoritme aangepast. Hierbij is de learnrate 0.001 genomen. Een samle wordt als voorgrond geclassificeerd wanneer de afstand tot het gemiddelde van de kernel k met de hoogste weegfactor meer dan $3 \cdot \sigma_k$ is. Voor elk interval wordt geteld hoeveel samples als object worden geclassificeerd. Hierbij worden samples alleen meegeteld wanneer er meer dan 3 samples achtereen als object worden geclassificeerd. Zo wordt getracht samples die ontstaan als gevolg van ruis in de sampleset niet mee te rekenen. Dit wordt gedaan om een goed beeld te krijgen van hoe lang het duurt voor een gedetecteerd object weer als achtergrond wordt geclassificeerd. Dit experiment wordt voor elke representatie herhaald met gebruik van 1, 2, 3, 4 en 5 kernels.

	μ_1	μ_2	σ_1^2	σ_2^2
Achtergrondverdeling	0.5	0.5	$1.0 \cdot 10^{-5}$	$3.0 \cdot 10^{-5}$
Objectverdeling1	0.5	0.7	$1.0 \cdot 10^{-4}$	$1.0 \cdot 10^{-4}$
Objectverdeling2	0.3	0.5	$1.0 \cdot 10^{-4}$	$1.0 \cdot 10^{-4}$
Objectverdeling3	0.3	0.7	$1.0 \cdot 10^{-4}$	$1.0 \cdot 10^{-4}$

Tabel 3.2: Gemiddelde en variantie van de gebruikte verdelingen

3.3.2 Resultaten

De resultaten van dit experiment zijn samengevat in tabel 3.3. Per interval is aangegeven hoeveel samples er vanaf het begin van het interval achtereenvolgend als object worden geclassificeerd. Als object geclassificeerde samples die niet tot een reeks van achtereenvolgende als object geclassificeerde samples behoren zijn hierbij niet meegeteld.

	AG	OBJ1	AG	OBJ2	AG	OBJ3	AG
Representatie 1							
1 Kernel	0	100	0	98	0	98	0
2 Kernels	31	46	0	56	0	500	0
3 Kernels	0	500	0	500	0	413	79
4 Kernels	0	500	0	500	139	500	421
5 Kernels	0	500	0	500	0	500	80
Representatie 2							
1 Kernel	0	181	0	142	0	60	0
2 Kernels	31	379	0	276	0	238	0
3 Kernels	0	500	0	500	0	360	122
4 Kernels	0	500	0	439	75	227	270
5 Kernels	0	500	0	500	0	500	0
Representatie 3							
1 Kernel	0	100	0	98	0	98	0
2 Kernels	31	376	0	253	0	228	24
3 Kernels	0	500	0	500	0	360	122
4 Kernels	0	500	0	443	68	236	261
5 Kernels	0	500	0	500	0	500	0

Tabel 3.3: Aantal samples dat per interval als object wordt geclassificeerd. AG: samples uit de achtergrondverdeling. OBJX: samples uit objectverdeling X.

3.3.3 Discussie

Bij gebruik van 1 kernel wordt bij geen van de drie representaties een groot deel van de objectsamples juist geclassificeerd. Ook bij gebruik van 2 kernels wordt een groot deel van de samples niet juist geclassificeerd. Bij alle representaties zijn bij het gebruik van 3, 4 of 5 kernels de resultaten vergelijkbaar. In sommige gevallen is het resultaat bij gebruik van 3 kernels zelfs beter dan bij het gebruik van 4 kernels. Het lijkt dat dit met de initialisatie van de kernels te maken heeft.

Het resultaat van de classificatie van de derde objectsampleset is bij alle representaties minder dan de classificatie van de eerste twee objectsamplesets. Ook de achtergrondsampleset wordt dan niet meer goed geclassificeerd. Als te detecteren objecten vaak dezelfde kleur hebben, zullen deze dus steeds korter worden gedetecteerd.

3.3.4 Conclusie

Het lijkt niet nodig meer dan drie kernels te gebruiken. Bij alle gebruikte representaties zijn de resultaten bij gebruik van 3, 4 en 5 kernels vergelijkbaar.

Er volgt uit dit experiment geen duidelijke voorkeur voor één van de representaties.

3.4 Data van videobeelden

In sectie 2.2.1 zijn verschillende implementaties van het mixture model besproken. In sectie 2.2.2 zijn verschillende manieren van classificatie besproken. Door deze implementaties van het mixture model en van de classificatie te combineren ontstaan verschillende manieren voor het bepalen van de voorgrond. Bij dit experiment wordt gekeken welke van de implementaties de beste is. Hierbij wordt gekeken naar het resultaat van de object classificatie. Bij gebruik van de genormaliseerde r en g kleurkanalen (zie sectie 2.1) vergelijken we de volgende 6 methoden voor het bepalen van de voorgrond.

1. ééndimensionale kernels, classificatie op basis van afstand tussen geregistreerde kleurwaarde en gemiddelde van de achtergrondkernel.
2. ééndimensionale kernels, classificatie op basis van à posteriori waarschijnlijkheid $P(k|x)$.
3. tweedimensionale sferische kernels, classificatie op basis van afstand tussen geregistreerde kleurwaarde en gemiddelde van de achtergrondkernel.
4. tweedimensionale sferische kernels, classificatie op basis van à posteriori waarschijnlijkheid $P(k|x)$.
5. tweedimensionale kernels met diagonale covariantiematrix, classificatie op basis van afstand tussen geregistreerde kleurwaarde en gemiddelde van de achtergrondkernel.
6. tweedimensionale kernels met diagonale covariantiematrix, classificatie op basis van à posteriori waarschijnlijkheid $P(k|x)$.

3.4.1 Opzet

Alle methoden worden tegelijkertijd uitgevoerd, zodat ze dezelfde beelden te verwerken krijgen. Alle methoden gebruiken dezelfde learnrate van 0.01. Na 2000 frames worden de resultaten van de 6 methoden opgeslagen. Op dat moment bevindt zich een object in de geregistreerde scène. Gedurende de voorafgaande periode van 2000 frames zijn er ook objecten in de scène geplaatst, verwijderd en van plaats veranderd.

Op de resultaten van de methoden waarbij de posteriori waarschijnlijkheid is bepaald (methode 2, 4, en 6), wordt een threshold operatie toegepast. Alle pixels waarvan de pixelwaarde groter dan 50% van de maximale pixelwaarde is krijgen de waarde 1, alle overige pixels de waarde 0. De gekozen drempelwaarde is niet van grote invloed op het uiteindelijke resultaat (zie de histogrammen in sectie 3.4.2). Het uiteindelijke resultaat van alle methoden is een binaire classificatie in voor- en achtergrondpixels.

3.4.2 Resultaten

In figuur 3.11 zijn de grijswaardenbeelden te zien zoals deze door de methoden 2, 4, en 6 worden geproduceerd. Deze grijswaarden geven de kans dat het pixel een object beschrijft. Deze wordt elke update voor elk pixel is bepaald, en is hier gegeven na update 2000. Van de grijswaardenbeelden is het histogram bepaald. Verder is in deze figuur het binaire resultaat na het toepassen van een drempelwaarde te zien.

De classificatie resultaten van alle hier gebruikte methoden zijn gegeven in figuur 3.12. Hierin is het originele beeld te zien, en de classificatie van de verschillende methoden. In de scène die wordt bekeken bevindt zich op dat moment één object. Dit is een persoon met donkere kleding achter een lichtgekleurde tafel.

3.4.3 Discussie

De histogrammen behorende bij de grijswaardenbeelden van de methoden 2, 4 en 6, hebben allemaal ongeveer dezelfd vorm. Ze bestaan uit twee pieken, één voor de lage intensiteitswaarden, en één voor de hoge intensiteitswaarden. In het gebied ertussen zijn alle histogrammen vlak, en hebben daar lage waarden. Voor iedere pixel geldt dus dat deze of een hoge of een lage waarschijnlijkheid voor het beschrijven van de voorgrond heeft. Een waarde ertussenin komt weinig voor. De drempelwaarde die wordt gekozen waarboven een pixel als object wordt geclassificeerd zal vanwege de vorm van de histogrammen niet van grote invloed zijn op het uiteindelijke resultaat van de classificatie

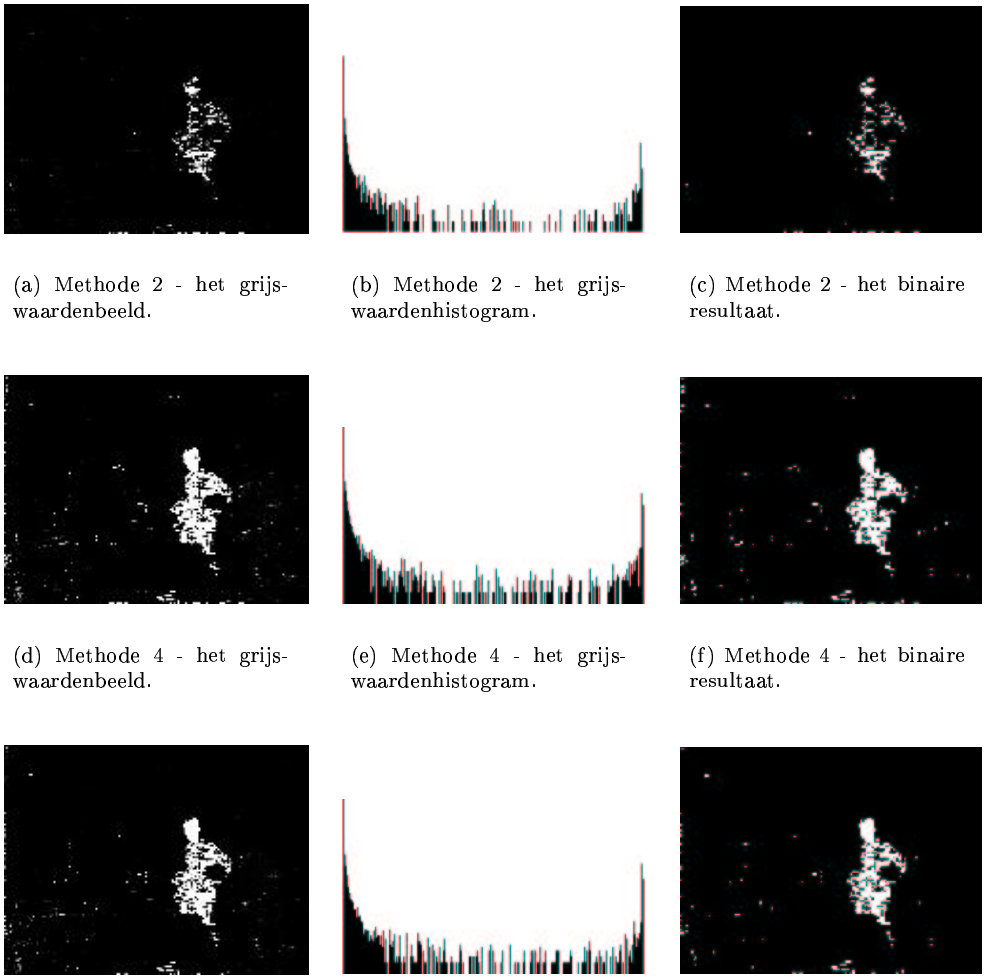
Het object zoals dat origineel is te zien in figuur 3.12(a), wordt door alle methoden gedetecteerd. De kwaliteit van de detectie verschilt echter per methode. Het resultaat van de methoden 1 (figuur 3.12(b)), 3 (figuur 3.12(d)) en 5 (figuur 3.12(f)) met de binaire classificatie, bevat meer ruis in de achtergrond dan de methoden 2 (figuur 3.12(c)), 4 (figuur 3.12(e)) en 6 (figuur 3.12(g)) waarbij $P(k|x)$ wordt bepaald. Bij methode 2 wordt slechts een klein deel van het object correct geclassificeerd. De resultaten van methode 4 en methode 6 zijn gelijkwaardig.

3.4.4 Conclusie

De drempelwaarde voor de à posteriori waarschijnlijkheid waarboven een pixel als object wordt geclassificeerd is 0,5 gekozen.

De resultaten van methode 4 en methode 6 vertonen grote gelijkenis. Uit de theorie blijkt echter dat de implementatie van methode 6 een nauwkeurigere benadering van de achtergrondkleur kan maken (vanwege de diagonale covariantiematrix met verschillende elementen op de diagonaal van de kernels). De resultaten van de objectdetectie kan in bepaalde gevallen met methode 6 dus beter zijn dan met methode 4.

Gaten in het object worden waarschijnlijk mede veroorzaakt door de lage kleurintensiteit. Dit zorgt voor een grote variantie in de gedetecteerde kleurwaarden in de gebruikte rg kleurruimte. De donker gekleurde monitor in de achtergrond zorgt mede voor een slechte de-



(a) Methode 2 - het grijswaardenbeeld.

(b) Methode 2 - het grijswaardenhistogram.

(c) Methode 2 - het binaire resultaat.

(d) Methode 4 - het grijswaardenbeeld.

(e) Methode 4 - het grijswaardenhistogram.

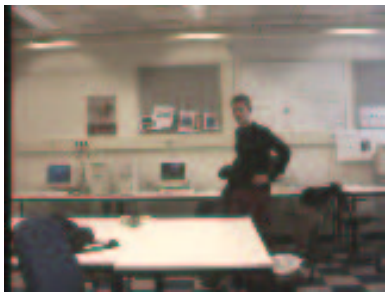
(f) Methode 4 - het binaire resultaat.

(g) Methode 6 - het grijswaardenbeeld.

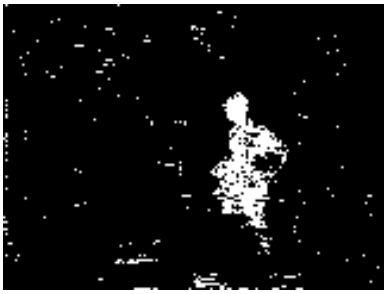
(h) Methode 6 - het grijswaardenhistogram.

(i) Methode 6 - het binaire resultaat.

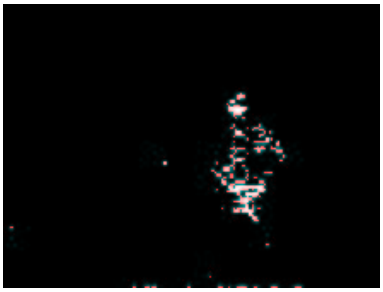
Figuur 3.11: Bepaling van de drempelwaarde van de grijswaardenbeelden. Vanwege het vlakke middenstuk van de histogrammen is de drempelwaarde op 50% gesteld.



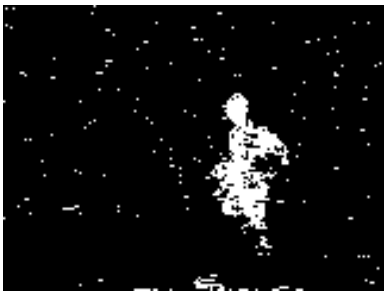
(a) Origineel beeld



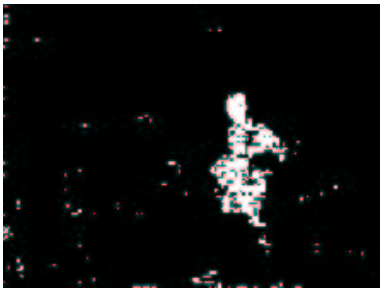
(b) Methode 1 - 1D kernels, binaire classificatie voorgrond.



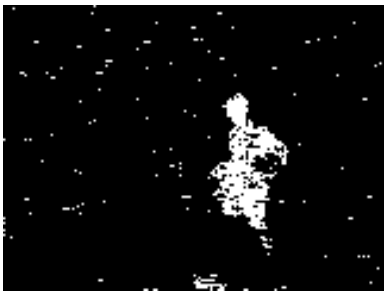
(c) Methode 2 - 1D kernels, posteriori kans op voorgrond.



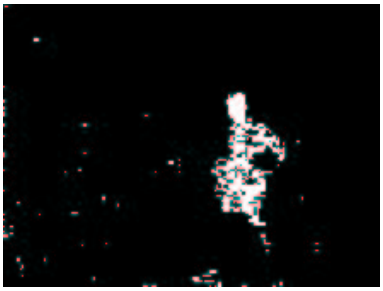
(d) Methode 3 - ND sferische kernels, binaire classificatie voorgrond.



(e) Methode 4 - ND sferische kernels, posteriori kans op voorgrond.



(f) Methode 5 - ND kernels met diagonale covariantiematrix, binaire classificatie voorgrond.



(g) Methode 6 - ND kernels met diagonale covariantiematrix, posteriori kans op voorgrond.

Figuur 3.12: Resultaten van objectdetectie met de verschillende methoden (met gebruik van de r and g kleurkanalen).

tectie van het object. Ook is het object zelf donker gekleurd waardoor er gaten in het object ontstaan. Het gezicht, en de handen die lichter zijn gekleurd worden beter gedetecteerd.

Behalve de gaten in de objecten is er in het uiteindelijke resultaat bij alle methoden nog ruis in de achtergrond te zien. Deze ruis bestaat uit gebieden van één of twee pixels per object, en dient niet als object te worden geclassificeerd. Er is dus nog een postprocessing operatie nodig om deze ruis te verwijderen.

Hoofdstuk 4

Implementatie

In hoofdstuk 2 zijn een aantal mogelijkheden geintroduceert voor de representatie van het mixture model, en de manier waarop een pixel wordt geclassificeerd. Dit leverde een aantal methoden voor objectdetectie. De kwaliteit van de objectdetectie met gebruik van de verschillende methoden is bekeken in hoofdstuk 3. In dit hoofdstuk beschrijven we het systeem waarop de tests zijn gedaan. Vervolgens worden de timing resultaten van de verschillende methoden bij gebruik van dit systeem behandeld.

4.1 Opzet van het systeem

De opzet van het totale systeem valt in drieën op te delen. We behandelen hier de hardware, het besturingssysteem en de software.

4.1.1 Hardware

- CPU: Intel Pentium III $1GHz$
- framegrabber: Hauppauge WinTV go, met BT878 video digitizer.

  In combinatie met de gebruikte videokaart wordt 24 bits RGB kleuren geleverd. De maximale resolutie is 320×240 pixels. De schaling naar de gewenste resolutie wordt hardwarematig door de framegrabber gedaan. Voor de horizontale schaling wordt een 6-tap interpolatie filter gebruikt, voor de vertikale schaling een 5-tap interpolatie filter met 'line-store'.
- videokaart: Matrox G400
- camera: Sony EVI-D31

  Pan/tilt/zoom kleuren camera met auto focus en auto whitebalancing. Nadeel: Whitebalancing zorgt voor kleurveranderingen in de achtergrond wanneer er een relatief groot donker object in beeld is. De auto whitebalancing kan alleen via een interface met de PC worden uitgeschakeld.

4.1.2 Besturingssysteem

Redhat Linux 7.1

- kernel: 2.4.17

- compiler: gcc 2.96
- glibc 2.1+
- desktop environment: KDE 2.2.1

4.1.3 Software

De implementatie van de objectdetectie is gedaan in C++. Voor de opzet van de implementatie is gebruik gemaakt van een *main loop* waarin de verschillende stappen van de objectdetectie worden gedaan. Voordat de *main loop* wordt uitgevoerd wordt het systeem eerst geïnitialiseerd. Dit bestaat uit het kiezen van initiele waarden voor de parameters van het mixture model en de waarde van de learnrate. In de *main loop* worden achtereenvolgens de volgende stappen uitgevoerd:

1. Het grabben van een beeld
2. Het converteren van de beelddata naar de gewenste kleurruimte
3. Het updaten van de parameterset van het mixture model
4. Het segmenteren van het originele beeld
5. Het toepassen van morfologische operatoren op het gesegmenteerde beeld
6. Het labellen van het gesegmenteerde beeld
7. Het weergeven van het resultaatbeeld

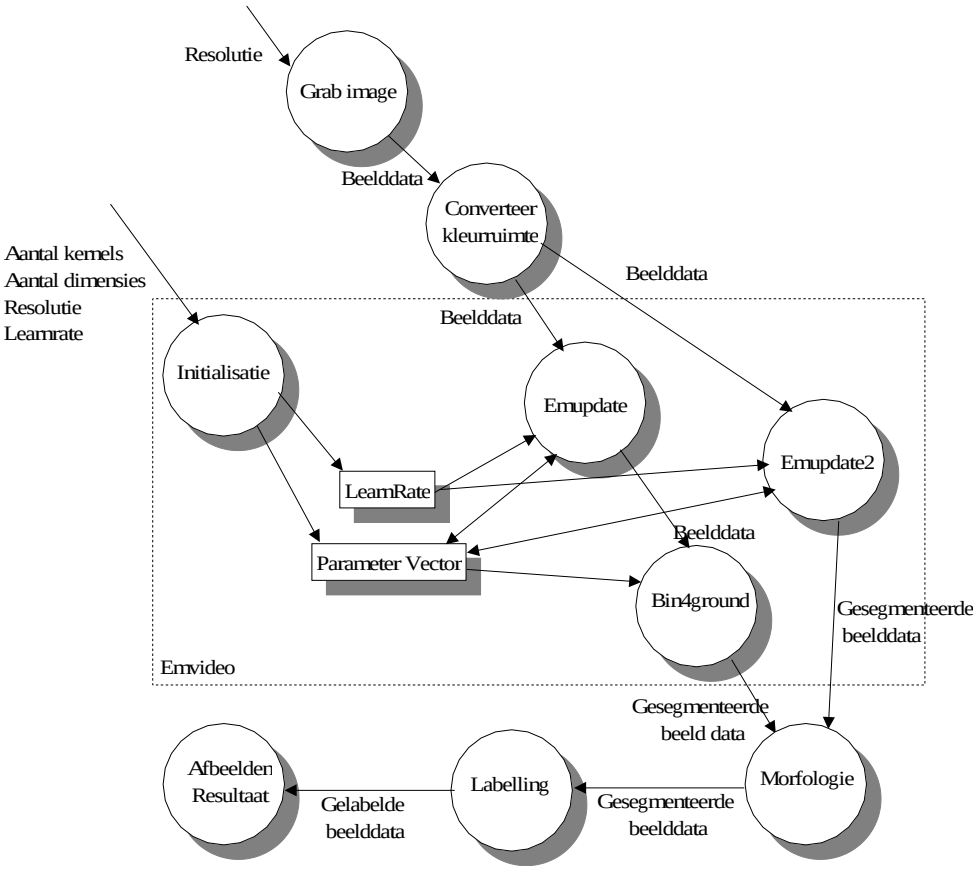
Het dataflow diagram van de software is gegeven in figuur 4.1.

class EMvideo

public:	
EMvideo	initialisatie van het mixture model
EMupdate	update van het mixture model met het EM algoritme
Bin4ground	binaire classificatie
EMupdate2	update van het mixture model met het EM algoritme waarbij gelijk de à posteriori waarshijnlijkheid wordt bepaald voor classificatie
private:	
nPix	aantal pixels per beeld
nKer	aantal kernels in het mixture model
nPar	aantal kernel paramters per pixel
nDim	aantal dimensies van de gebruikte kleurruimte
learnrate	learnrate van het EM algoritme
*parvec	pointer naar de paramterers van het mixture model

Tabel 4.1: Class EMvideo: een EMvideo object en de bijbehorende attributen

Implementatie van het mixture model, de bijbehorende update vergelijkingen en de verschillende segmentatie methoden is gedaan in de hierna beschreven classes. IN deze classes is een EMvideo object geimplementeerd. De attributen van een EMvideo object zijn gegeven in tabel 4.1. Een van de attributen van een EMvideo object is *parvec*. Dit is de parameter 'vector' met alle parameters van de mixture modellen van alle pixels. Als datastructuur



Figuur 4.1: Data Flow Diagram van de software. Elk EMvideo object kan op twee manieren update van de paramterset, en classificatie van het originele beeld doen.

voor de parameterset van het mixturemodel is gekozen voor een C-array. De opbouw van deze parameterset verschilt per keuze voor het mixture model[1]. De volgende classes zijn geïmplementeerd:

- **EMvideo1D** Implementatie van het mixture model met ééndimensionale kernels, en de bijbehorende update formules (zie sectie 2.3.3). De opbouw van de parameter vector voor één pixel bij gebruik van K kernels en D kleurkanalen:

$$\begin{aligned} parvec \quad = \{ \quad & \pi_{1,1}, \pi_{2,1}, ..., \pi_{K,1}, \mu_{1,1}, \mu_{2,1}, ..., \mu_{K,1}, \sigma_{1,1}, \sigma_{2,1}, ..., \sigma_{K,1}, \\ & \pi_{1,2}, \pi_{2,2}, ..., \pi_{K,2}, \mu_{1,2}, \mu_{2,2}, ..., \mu_{K,2}, \sigma_{1,2}, \sigma_{2,2}, ..., \sigma_{K,2}, \\ & ..... \\ & \pi_{1,D}, \pi_{2,D}, ..., \pi_{K,D}, \mu_{1,D}, \mu_{2,D}, ..., \mu_{K,D}, \sigma_{1,D}, \sigma_{2,D}, ..., \sigma_{K,D} \} \end{aligned}$$

[1] De opbouw van de parameterset bij gebruik van 1D kernels en onofhankelijke kleurkanalen is zo gekozen dat de paramterset bij een update in sequentiele volgorde wordt doorlopen. Een C-array leek daarom de meest efficiente datastructuur voor de parameterset. Bij een andere keuze voor het mixture model wordt de parameterset bij het updaten echter niet meer in sequentiele volgorde doorlopen. Het kan daarom zijn dat een andere keuze voor de datastructuur van de parameterset kan resulteren in een snellere update methode (bijvoorbeeld een c++ vector).

Hierin is $\pi_{k,d}$ de weegfactor van kernel k voro kleurkanaal d, $\mu_{k,d}$ het gemiddelde van kernel k voor kleurkanaal d en $\sigma_{k,d}$ de waarde van de standaarddeviatie van kernel k voor kleurkanaal d. Het mixture model bevat $3 \cdot D \cdot K$ parameters per pixel.

- **EMvideoND** Implementatie van het mixture model met multi dimensionale sferische kernels, en de bijbehorende update formules (zie sectie 2.3.2). De opbouw van de parameter vector voor één pixel met K kernels en D kleurkanalen:

$$\begin{aligned} parvec \quad = \{ \quad & \pi_1, \pi_2, ..., \pi_K, \\ & \mu_{1,1}, \mu_{2,1}, ..., \mu_{K,1}, \\ & \mu_{1,2}, \mu_{2,2}, ..., \mu_{K,2}, \\ & ... \\ & \mu_{1,D}, \mu_{2,D}, ..., \mu_{K,D}, \\ & \sigma_1, \sigma_2, ..., \sigma_K \} \end{aligned}$$

  Hierin is π_k de weegfactor van kernel k, $\mu_{k,d}$ het gemiddelde van kernel k voor dimensie d en σ_k de waarde van de elementen op de diagonaal van de covariantiematrix van kernel k. Totaal heeft het mixture model $K \cdot (D + 2)$ parameters per pixel.

- **EMvideoNDmv** Implementatie van het mixture model met multi dimensionale kernels met als covariantie matrix een diagonaal matrix, en de bijbehorende update formules (zie sectie 2.3.1). De opbouw van de parameter vector voor één pixel met K kernels en D kleurkanalen:

$$\begin{aligned} parvec \quad = \{ \quad & \pi_1, \pi_2, ..., \pi_K, \\ & \mu_{1,1}, \mu_{2,1}, ..., \mu_{K,1}, \\ & \mu_{1,2}, \mu_{2,2}, ..., \mu_{K,2}, \\ & ... \\ & \mu_{1,D}, \mu_{2,D}, ..., \mu_{K,D}, \\ & \sigma_{1,1}, \sigma_{2,1}, ..., \sigma_{K,1}, \\ & \sigma_{1,2}, \sigma_{2,2}, ..., \sigma_{K,2}, \\ & ... \\ & \sigma_{1,D}, \sigma_{2,D}, ..., \sigma_{K,D} \} \end{aligned}$$

  Hierin is π_k de weegfactor van kernel k, $\mu_{k,d}$ het gemiddelde van kernel k voor dimensie d en $\sigma_{k,d}$ de waarde van de elementen op de diagonaal van de covariantiematrix van kernel k. Het mixture model heeft $K \cdot (2D + 1)$ parameters per pixel.

Voor het grabben van de beelden van de camera, de weergave van de beelden en het verkrijgen van timinginformatie is gebruik gemaakt van de volgende classes van de Ilab neuromorphic vision toolkit (http://ilab.usc.edu/sdoc/html/main.html) van de University of Southern California:

- **V4Lgrabber** Definitie en functies van de Video4linux framegrabber.
- **PixRGB<byte>** Representatie voor de pixels van het beeld in RGB kleuren.
- **Image<PixRGB<byte>>** Representatie van een beeld.
- **Xwindow** Simpel Xwindow voor het weergeven van een beeld.
- **Timer** Timer met representatie in milliseconden.

4.2 Timing resultaten van het systeem

4.2.1 Opzet

Op het systeem zoals dat in sectie 4.1 is beschreven zijn de in sectie 3.4 beschreven experimenten gedaan. Bij deze experimenten is een resolutie van 160×120 pixels gebruikt. Bij de methoden 1, 3 en 5 is de binaire classificatie bepaald. Zoals ook blijkt uit figuur 4.1 wordt update en classificatie hierbij apart gedaan. Zodoende kan deze bij deze methoden de timing van de afzonderlijke onderdelen ook apart worden bepaald. Bij de methoden 2, 4 en 6 wordt de à posteriori waarschijnlijkheid bepaald. Dit gebeurt in dezelfde stap als de update van de parameterset. Het is dus niet mogelijk om bij deze methoden de timing van de update en de 'classificatie' apart te bepalen.

4.2.2 Resultaten

De timing resultaten van de methoden zijn gegeven in tabel 4.2. Dit zijn de gemiddelde tijden in ms per onderdeel. Ook de variantie in de gemeten tijden is gegeven (tussen haakjes).

	Methode	update (ms)	segmentatie (ms)	update en segmentatie (ms)
1	1D kernels, Binaire classificatie	$79,0(0,7)$	$10,9(0,2)$	$89,9(0,9)$
2	1D kernels, Grijswaarden classificatie			$80,5(0,6)$
3	ND sferische kernels, Binaire classificatie	$53,6(0,4)$	$20,5(0,5)$	$74,1(0,9)$
4	ND sferische kernels, Grijswaarden classificatie			$54,3(0,5)$
5	ND kernels met diagonale covariantiematrix, Binaire classificatie	$50,5(0,4)$	$17,9(0,4)$	$68,4(0,8)$
6	ND kernels met diagonale covariantiematrix, Grijswaarden classificatie			$51,7(0,4)$

Tabel 4.2: De timingresultaten van de updatemethoden zoals deze in sectie 3.4 zijn beschreven. De resultaten zijn in ms, tussen haakjes de variantie van de resultaten.

4.2.3 Discussie

Van de geteste implementaties is methode 6 de snelste, gevolgd door methode 4. De overige methoden zijn duidelijk langzamer. Wanneer de update en classificatie van methode 3 en methode 5 echter ook 'samen' worden geïmplementeerd zou dit voor deze methoden een beter resultaat kunnen opleveren. Omdat de updatestap alleen bij deze methoden al ongeveer evenveel tijd nodig heeft dan het resultaat van methode 4 en methode 6, zal het uiteindelijke resultaat echter niet sneller zijn dan methode 4 en 6. Methode 1 en 2 met ééndimensionale kernels zijn duidelijk langzamer dan de overige methoden.

4.2.4 Conclusie

In sectie 3.4.4 is al geconcludeerd dat methode 6 de voorkeur heeft boven methode 4. De resultaten van dit experiment geven geen aanleiding om de keuze voor methode 6 te weerleggen.

Hoofdstuk 5

Discussie en Conclusies

Bij dit onderzoek hebben we gekeken in hoeverre een background subtraction methode die gebruik maakt van mixture modellen om de achtergrondkleur te modelleren, geschikt is voor objectdetectie in video. Er is gekeken welke kleurruimte het best kan worden gebruikt, welke update methode kan worden gebruikt, welke vorm voor het mixture model het meest geschikt is en welke manier van classificatie het beste resultaat geeft.

5.1 Discussie

In sectie 2.1 zijn een aantal veelgebruikte kleursystemen bekeken. Er is voor gekozen om voor de kleurruimte gebruik te maken van de genormaliseerde r en g kleurkanalen. Deze ruimte is gekozen omdat de geregistreerde kleurwaarden hierbij onveranderd blijven onder verandering van de belichtingsintensiteit. Het gevolg hiervan is dat er geen schaduw wordt gedetecteerd, en dat de belichtingsintensiteit kan veranderen zonder dat dit van invloed is op de uiteindelijke objectdetectie. Een nadeel van het gebruik van de r en g kleurkanalen is dat de geregistreerde kleurwaarden 'instabiel' worden bij lage kleurintensiteit, zoals blijkt in sectie 3.1. De verdeling van kleurwaarden voor deze kanalen krijgt dan een grote variantie.

Voor het aanpassen van de parameters van het mixture model is een veel gebruikte incrementele variant van het EM algoritme met *exponentieel vergeten* toegepast. Het blijkt dat deze methode bij gebruik van één kleurkanaal een goede benadering van het recente kleurverleden geeft (zie sectie 3.2). Om de snelheid van het aanpassen van het mixturemodel te verhogen, is ook een alternatieve update methode bekeken. Deze methode die door Stauffer en Grimson is voorgesteld [8], bleek niet in staat om een goede benadering van de variantie van een verdeling van kleurwaarden te geven. Een nadeel van het gebruikte EM algoritme is te zien wanneer objecten zich gedurende langere tijd op één locatie bevinden. Het achtergrondmodel wordt veranderd door de gedetecteerde objectkleuren. Wanneer op deze locatie weer achtergrond zichtbaar wordt, moet hiervoor weer een goed model worden geleerd. Op deze locatie werkt de objectdetectie gedurende het leren van het achtergrondmodel niet volledig. De tijd die nodig is om een achtergrondmodel te leren is afhankelijk van de gebruikte learnrate. Ook de mate waarin het achtergrondmodel wordt verstoord door objectkleuren is afhankelijk van de gebruikte learnrate. Hoe lager de learnrate hoe langer het duurt voor een goed model is geleerd, en des te minder wordt het achtergrondmodel verstoord door een passerend object.

De vorm van de kernels in het mixture model is van invloed op hoe goed het kleurverleden kan worden benaderd. In sectie 2.3 zijn drie manieren geïntroduceerd om de verdeling

van kleurwaarden van een pixel te benaderen. De eerste manier maakt gebruik van een mixture model met ééndimensionale kernels waarbij elk kleurkanaal door een apart mixture model wordt beschreven. De tweede manier maakt gebruik van één mixture model met N-dimensionale sferische kernels om de verdeling van kleurwaarden in een N-dimensionale kleurruimte te beschrijven. De laatste manier gebruikt ook N-dimensionale kernels maar deze hebben een diagonale covariantiematrix. Zo kan een verdeling van kleurwaarden met verschillende varianties per kleurkanaal beter worden benaderd. Uit het experiment in sectie 3.4 blijkt dat de twee laatste manieren een betere classificatie opleveren dan het gebruik van ééndimensionale kernels. Ook qua snelheid verschillen deze methoden elkaar niet veel, maar zijn ze wel sneller dan het gebruik van ééndimensionale kernels.

Er zijn twee methoden geïntroduceerd voor het classificeren van het beeld in voor- en achtergrond. Bij de eerste 'binaire' methode wordt de afstand in de kleurruimte tussen de huidige kleurwaarde en het gemiddelde van de achtergrondkernel bepaald in termen van de standaarddeviatie van de achtergrondkernel. Bij de tweede methode wordt de à posteriori waarschijnlijkheid van de huidige kleurwaarde per pixel bepaald. Dit resulteert in een grijswaardenbeeld met de waarschijnlijkheden dat een pixel de achtergrond beschrijft. Deze laatste methode geeft minder ruis in de achtergrond (Zie resultaten van sectie 3.4).

Het systeem kan worden versneld door niet elke update het achtergrondmodel van alle pixels aan te passen. Wel wordt elke pixel elke update geclassificeerd. Omdat het bepalen van de à posteriori waarscijnlijkheid een van de meest tijdrovende stappen is, en deze tijdens de updatestap ook moet worden bepaald, zal deze oplossing met de classificatie methode op basis van à posteriori waarschijnlijkheid weinig snelheidswinst behalen.

5.2 Conclusie

Er is gekozen voor object detectie op basis van background subtraction in de rg kleurruimte. Hierbij wordt per pixel een achtergrondmodel bepaald naar aanleiding van een benadering van het kleurverleden. Voor de benadering van het kleurverleden is gekozen voor een mixture model van Gaussische kernels met diagonale covariantiematrix. De elementen van dit mixture model worden aangepast volgens een incrementeel EM algoritme met exponentieel vergeten. Classificatie van een pixel gebeurt door de à posteriori waarschijnlijkheid van de kernel met de hoogste weegfactor te bepalen. Wanneer deze waarschijnlijkheid hoger dan 0.5 is, wordt het betreffende pixel als achtergrond geclassificeerd.

De kwaliteit van de objectdetectie is onder meer afhankelijk van de gebruikt learnrate. De waarde van de learnrate is sterk afhankelijk van de situatie waarin het systeem gaat worden gebruikt.

Het uiteindelijke resultaat van de objectdetectie met het beschreven systeem vertoont vaak gaten in de gedetecteerde objecten. Ook komt het voor dat objecten gesegmenteerd worden, dus als meerdere objecten worden geclassificeerd. De kwaliteit van de detectie is niet goed genoeg voor unieke identificatie van objecten in video. Tracking op basis van de resultaten van het beschreven systeem zal daarom geen goede resultaten geven.

5.3 Toekomstig onderzoek

De keuze voor de gebruikte rg kleurruimte blijkt een probleem vanwege de waarden bij lage kleurintensiteit. De oplossing hiervoor kan worden gezocht in het aanpassen van de huidige kleurruimte. Kleurwaarden met een kleurintensiteit lager dan een bepaalde drempelwaarde kunnen bijvoorbeeld als 'donker' worden beschouwd. Het probleem is dan hoe hoog deze dre-

melwaarde moet worden genomen. Ook kan er voor een andere kleurruimte worden gekozen. Behalve de eigenschappen van een kleurruimte met betrekking tot de kwaliteit van objectdetectie, moet ook de rekentijd die samenhangt met de gebruikte kleurruimte worden bekeken. Bij de keuze voor de kleurruimte moet onder andere worden gelet op de complexiteit van de conversie vanuit de RGB kleurruimte en het aantal dimensies van de kleurruimte (hoe hoger het aantal dimensies, hoe meer paramters van het mixture model moeten worden aangepast).

Soms ontstaan fouten in de detectie doordat de achtergrond dezelfde kleur heeft als de objecten. Dit kan worden verholpen door gebruik te maken van diepte informatie. Wanneer een object dezelfde kleur heeft als de achtergrond, kan het van de achtergrond worden onderscheiden omdat het dichter bij de camera staat dan de achtergrond. Dit is al eerder toegepast door Harville et. al. [6].

De huidige update methode verstoort het achtergrondmodel wanneer er objecten worden gedetecteerd. Niet alleen de parameters van de kernels die de objectkleur beschrijven worden gewijzigd, maar ook de parameters van de kernels die de achtergrondkleur beschrijven. Een oplossing kan zijn om alleen die kernels te veranderen die waarbij de waarschijnlijkheid van de huidige kleurwaarde boven een bepaalde drempelwaarde komt. Zoiets werd ook gedaan door Stauffer en Grimson. Het probleem is het vinden van een methode om de variantie van de kernel correct aan te passen. Ook moet er een goede waarde voor de drempelwaarde worden gevonden.

Wanneer het achtergrondmodel aan het systeem is geleerd, hoeft dit niet voor elk pixel elk frame aangepast te worden. Door het achtergrond model voor een deel van alle pixels aan te passen is er per frame minder tijd nodig voor de update stap. Wel moet elk frame alle pixels worden geclassificeerd. Bij de huidige implementatie wordt daarvoor de à posteriori waarschijnlijheid bepaald. Dit is de meest tijdrovende stap van de update van de parameterset. In de huidige vorm wordt daarom weinig voordeel behaald door niet voor alle pixels elke update de parameterset aan te passen. Hiervoor zou een efficiëntere classificatiemethode moeten worden geïntroduceerd.

Een probleem bij de huidige implementatie van mixture model en update methode ontstaat wanneer gedurende lange tijd achtergrond wordt gedetecteerd. De paramters van de verschillende kernels 'groeien dan naar elkaar toe'. De waarden van de parameters van de verschillende kernels worden gelijk aan elkaar. Het gevolg hiervan is dat de à posteriori waarschijnlijkheid van de achtergrondkernel gelijk wordt aan die van de overige kernels. De waarde van de à posteriori waarschijnlijkheid van de achtergrondkernel wordt uiteindelijk gelijk aan $\frac{1}{k}$ met k het aantal kernels van het mixture model. De gekozen drempelwaarde van 0.5 voor de à posteriori waarschijnlijkheid waarboven een pixel als achtergrond wordt geclassificeerd voldoet niet. Een ander classificatie criterium kan zijn om de à posteriori waarschijnlijkheid van de achtergrondkernel te vergelijken met die van de kernel met de op één na hoogste weegfactor. Wanneer die van de achtergrondkernel hoger is, wordt het pixel als achtergrond geclassificeerd.

Een ander nadeel is dat er effectief minder kernels overblijven om het kleurverleden te beschrijven wanneer de kernels precies dezelfde vorm hebben. De parameters van de kernels worden op dezelfde manier aangepast. De kernels blijven dus qua vorm gelijk aan elkaar, ook als er van de achtergrondkleur afwijkende kleurwaarden worden geregistreerd.

Een verbetering die kan worden geintroduceerd is het gebruik van een variabele learnrate. Wanneer het achtergrondmodel goed is geleerd kan een lagere learnrate gebruikt worden zodat gedetecteerde objecten niet voor een te grote wijziging van het achtergrondmodel zorgen. De learnrate moet echter ook niet te laag worden gekozen omdat dan veranderingen in de achtergrond als gevolg van bijvoorbeeld belichting dan niet snel genoeg worden verwerkt in het achtergrondmodel. Het is misschien verstandig de learnrate afhankelijk van de kleurverandering van het totale beeld te nemen.

Bijlage A

Afleiding van de update formules van het EM algoritme

Het mixture model (vergelijking A.2) bestaat uit een sommatie van gewogen Gaussische verdelingen (vergelijking A.1). De à posteriori waarschijnlijkheid dat een kleurwaarde $\vec{x}_t$ aan kernel k kan worden toegekend, is gegeven in vergelijking A.3

$$p(\vec{x}|k) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_k)^T \Sigma_k^{-1} (\vec{x}-\vec{\mu}_k)} \tag{A.1}$$

$$p(\vec{x}) = \sum_{k=1}^{K} \pi_k p(\vec{x}|k) \tag{A.2}$$

$$P(k|\vec{x}) = \frac{\pi_k p(\vec{x}|k)}{p(\vec{x})} \tag{A.3}$$

De parameters van het mixture model worden aangepast met het EM algoritme. De standaard variant van het EM algoritme is onder andere behandeld door Bilmes [1]. De vergelijkingen voor het bepalen van de parameters met het standaard EM algoritme zijn gegeven in de vergelijkingen A.4, A.6 en A.8. Deze vergelijkingen worden gebruikt voor het afleiden van een incrementele variant van het EM algoritme. Het resultaat hiervan is gegeven in de vergelijkingen A.5, A.7 en A.9.

$$\pi_{k,t} = \frac{1}{t} \sum_{t'=1}^{t} P(k|\vec{x}_{t'}) \tag{A.4}$$

$$\begin{aligned} \pi_{k,t+1} &= \frac{1}{t+1} \sum_{t'=1}^{t+1} P(k|\vec{x}_{t'}) \\ &= \frac{1}{t+1} \left(\sum_{t'=1}^{t} P(k|\vec{x}_{t'}) + P(k|\vec{x}_{t+1}) \right) \\ &\simeq \frac{1}{t+1} \left(t\pi_{k,t} + P(k|\vec{x}_{t+1}) \right) \end{aligned}$$

$$\pi_{k,t+1} \simeq \pi_{k,t} + \frac{1}{t+1}(P(k|\vec{x}_{t+1}) - \pi_{k,t}) \tag{A.5}$$

Het gemiddelde $\vec{\mu}_{k,t+1}$ van kernel k op tijdstip $t+1$:

$$\vec{\mu}_{k,t} = \begin{pmatrix} \mu^1_{k,t} \\ \mu^2_{k,t} \\ \vdots \\ \mu^i_{k,t} \end{pmatrix} \tag{A.6}$$

$$\begin{aligned}
\mu^i_{k,t} &= \frac{\sum_{t'=1}^{t} P(k|\vec{x}_t) x^i_t}{\sum_{t'=1}^{t} P(k|\vec{x}_t)} \\
\mu^i_{k,t} &= \frac{1}{t\pi_{k,t}} \sum_{t'=1}^{t} P\{k|\vec{x}_{t'}\} x^i_{t'} \\
\mu^i_{k,t+1} &= \frac{1}{(t+1)\pi_{k,t+1}} \sum_{t'=1}^{t+1} P(k|\vec{x}_{t'}) x^i_{t'} \\
&= \frac{1}{(t+1)\pi_{k,t+1}} \left(\sum_{t'=1}^{t} P(k|\vec{x}_{t'}) x^i_{t'} + P(k|\vec{x}_{t+1}) x^i_{t+1} \right) \\
&= \frac{1}{(t+1)\pi_{k,t+1}} \left(t\pi_{k,t}\mu^i_{k,t} + P(k|\vec{x}_{t+1}) x^i_{t+1} \right) \\
&= \frac{1}{(t+1)\pi_{k,t+1}} \left(t \left(\frac{t+1}{t}\pi_{k,t+1} - \frac{1}{t} P(k|\vec{x}_{t+1}) \right) \mu^i_{k,t} + P(k|\vec{x}_{t+1}) x^i_{t+1} \right) \\
\mu^i_{k,t+1} &= \mu_{k,t} + \frac{1}{(t+1)\pi_{k,t+1}} P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t})
\end{aligned} \tag{A.7}$$

De variantie $Sigma_{k,t+1}$ van kernel k op tijdstip $t+1$

$$\Sigma_{k,t} = \begin{pmatrix} \sigma^{2,11}_{k,t} & \sigma^{2,12}_{k,t} & \cdots & \sigma^{2,1j}_{k,t} \\ \sigma^{2,21}_{k,t} & \sigma^{2,22}_{k,t} & \cdots & \sigma^{2,2j}_{k,t} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma^{2,i1}_{k,t} & \sigma^{2,i2}_{k,t} & \cdots & \sigma^{2,ij}_{k,t} \end{pmatrix}$$

$$\sigma^{2,ij}_{k,t} = \frac{\sum_{t'=1}^{t} P(k|\vec{x}_t)(x^i_{t'} - \mu^i_{k,t})(x^j_{t'} - \mu^j_{k,t})}{\sum_{t'=1}^{t} P(k|\vec{x}_t)} \tag{A.8}$$

$$\begin{aligned}
\sigma^{2,ij}_{k,t} &= \frac{1}{t\pi_{k,t}} \sum_{t'=1}^{t} P(k|\vec{x}_{t'})(x^i_{t'} - \mu^i_{k,t})(x^j_{t'} - \mu^j_{k,t}) \\
\sigma^{2,ij}_{k,t+1} &= \frac{1}{(t+1)\pi_{k,t+1}} \sum_{t'=1}^{t+1} P(k|\vec{x}_{t'})(x^i_{t'} - \mu^i_{k,t+1})(x^j_{t'} - \mu^j_{k,t+1}) \\
&= \frac{1}{(t+1)\pi_{k,t+1}} \left(\sum_{t'=1}^{t} P(k|\vec{x}_{t'})(x^i_{t'} - \mu^i_{k,t+1})(x^j_{t'}\mu^j_{k,t+1}) + P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t+1})(x^j_{t+1} - \mu^j_{k,t+1}) \right) \\
&\simeq \frac{1}{(t+1)\pi_{k,t+1}} \left(t\pi_{k,t}\sigma^{2,ij}_{k,t} + P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t+1})(x^j_{t+1} - \mu^j_{k,t+1}) \right) \\
&\simeq \frac{1}{(t+1)\pi_{k,t+1}} \left(t \left(\frac{t+1}{t}\pi_{k,t+1} - \frac{1}{t} P(k|\vec{x}_{t+1}) \right) \sigma^{2,ij}_{k,t+1} + P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t+1})(x^j_{t+1} - \mu^j_{k,t+1}) \right) \\
\sigma^{2,ij}_{k,t+1} &\simeq \sigma^{2,ij}_{k,t} + \frac{1}{(t+1)\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left((x^i_{t+1} - \mu^i_{k,t+1})(x^j_{t+1} - \mu^j_{k,t+1}) - \sigma^{2,ij}_{k,t} \right)
\end{aligned} \tag{A.9}$$

A.1 Exponentieel Vergeten

We willen niet het gehele verleden meenemen bij het aanpassen van de parameters. We maken gebruik van de laatste $\frac{1}{\gamma}$ samples om de parameters aan te passen(exponentieel vergeten). De update formules die volgen uit de vergelijkingen A.5, A.7, en A.9, worden dan:

$$\pi_{k,t+1} = \pi_{k,t} + \gamma(P(k|\vec{x}_{t+1}) - \pi_{k,t}) \tag{A.10}$$

$$\mu^i_{k,t+1} = \mu_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t}) \tag{A.11}$$

$$\sigma^{2,ij}_{k,t+1} = \sigma^{2,ij}_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left((x^i_{t+1} - \mu^i_{k,t+1})(x^j_{t+1} - \mu^j_{k,t+1}) - \sigma^{2,ij}_{k,t}\right) \tag{A.12}$$

A.2 Gaussische distributies met diagonale covariantiematrix

Om een kostbare matrix inversie te besparen bij het bepalen van de kans op de huitdig kleurwaarde gegeven de parameters van kernel k in vergelijking A.1, wordt de benadering gedaan dat de covariantie matrix alleen op de diagonaal een van 0 afwijkende waarde heeft. De covariantie matrix is van de vorm

$$\Sigma_{k,t} = \begin{pmatrix} \sigma^{2,11}_{k,t} & 0 & \cdots & 0 \\ 0 & \sigma^{2,22}_{k,t} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^{2,ii}_{k,t} \end{pmatrix}$$

De update formules worden dan

$$\pi_{k,t+1} = \pi_{k,t} + \gamma(P(k|\vec{x}_{t+1}) - \pi_{k,t}) \tag{A.13}$$

$$\mu^i_{k,t+1} = \mu_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t}) \tag{A.14}$$

$$\sigma^{2,ii}_{k,t+1} = \sigma^{2,ii}_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left((x^i_{t+1} - \mu^i_{k,t+1})^2 - \sigma^{2,ii}_{k,t}\right) \tag{A.15}$$

A.3 Sferische Gaussische distributies

Door te kiezen voor een d-dimensionale sferische Gaussische distributie hoeft slechts één van de d parameters van de covariantie matrix te worden aangepast. De covariantiematrix is van de vorm $\Sigma_{k,t} = \sigma^2_{k,t} \cdot \mathbf{I}$, en de waarde van de parameters op de diagonaal zijn gelijk. Voor het bepalen van deze waarde nemen we het gemiddelde van de waarden voor de variantie parameters die zouden volgen uit A.15. De update formules worden door het bepalen van het gemiddelde voor de variantieparameter

$$\pi_{k,t+1} = \pi_{k,t} + \gamma(P(k|\vec{x}_{t+1}) - \pi_{k,t}) \tag{A.16}$$

$$\mu^i_{k,t+1} = \mu_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t}) \tag{A.17}$$

$$\sigma^2_{k,t+1} = \sigma^2_{k,t} + \frac{\gamma}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left(\frac{\|\vec{x}_{t+1} - \vec{\mu}_{k,t+1}\|^2}{d} - \sigma^2_{k,t}\right) \tag{A.18}$$

A.4 Alternatieve update methode

Door Stauffer en Grimson [8] is een alternatieve methode voor het aanpassen van de parameters geïntroduceerd. Bij deze methode worden alleen die kernels aangepast waarvoor de huidige kleurwaarde aan de betreffende kernel kan worden toegekend. Wanneer een kleurwaarde aan geen van de kernels kan worden toegekend, wordt de kernel met de laagste weegfactor geherinitialiseerd. Deze kernel krijgt voor het gemiddelde de waarde van de huidige kleurwaarde, voor de variantie een relatief hoge waarde en voor de weegfactor een relatief lage waarde.

Bij de kernels die wel aangepast worden, wordt een benadering gedaan voor de à posteriori waarschijnlijkheid. Deze benadering is gegeven in vergelijking A.19. Ook hier wordt gebruik gemakt sferische Gaussische distributies. Opvallend is het verschil in de update van de variantie parameters. In vergelijking A.23 is te zien dat de update een factor $\frac{1}{d}$ verschilt met de update van het EM algoritme met sferische kernels.

De update formules volgens Stauffer en Grimson zijn:

$$M(k, \vec{x}_{t+1}) = \begin{cases} 1 & als \quad \vec{x}_{t+1} \in k \\ 0 & anders \end{cases} \tag{A.19}$$

$$\gamma' = \gamma p(\vec{x}_{t+1}) \tag{A.20}$$

$$\pi_{k,t+1} = \pi_{k,t} + \gamma(M(k, \vec{x}_{t+1}) - \pi_{k,t}) \tag{A.21}$$

$$\mu^i_{k,t+1} = \mu_{k,t} + \frac{\gamma'}{\pi_{k,t+1}} P(k|\vec{x}_{t+1})(x^i_{t+1} - \mu^i_{k,t}) \tag{A.22}$$

$$\sigma^2_{k,t+1} = \sigma^2_{k,t} + \frac{\gamma'}{\pi_{k,t+1}} P(k|\vec{x}_{t+1}) \left(\|\vec{x}_{t+1} - \vec{\mu}_{k,t+1}\|^2 - \sigma^2_{k,t} \right) \tag{A.23}$$

Bibliografie

[1] Jeff Bilmes. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical report, International Computer Science Institute, Berkeley CA and Computer Science Devision, Department of Electrical Engineering and Computer Science, U.C. Berkeley, April 1998.

[2] N. M. Dempster,A.P. Laird and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. B*, 39:185–197, 1977.

[3] G. Stijnman en R. van den Boomgaard. Background estimation in video sequences. Technical Report 10, Intelligent Sensory Information Systems Group, University of Amsterdam, 1999.

[4] Theo Gevers. Color in image search engines, 1999. Reader multimedia informatie B.O., Intelligent Sensory Information Systems Group, University of Amsterdam.

[5] Berthold Klaus Paul Horn. *Robot Vision.* MIT electrical engineering and computer science series. The MIT Press, 1986.

[6] M.Harville G.Gordon J.Woodfill. Adaptive background subtraction using color and depth. *Proceedings of IEEE Conference on image processing*, October 1999. Thessaloniki, Greece.

[7] A.W.M. Smeulders and R. van den Boomgaard. An introduction to image processing and computer vision, 1992. Syllabus Beeldinformatieverwerking, University of Amsterdam.

[8] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2:246–252, 1999.

[9] Kentaro Toyama, John Krumm, Barry Brumitt, and Brian Meyers. Wallflower: Principles and practice of background maintenance. In *ICCV (1)*, pages 255–261, 1999.

[10] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.