

Imaging project

TI2710-D Computer Vision

L.J.P. van der Maaten & J. de Ridder



Groep 5 – De Wijze Wieken

Raoul Harel – 4143205

Mirko Dunnewind – 4097343

Marnix de Graaf – 4172949

Tim Rensen – 4157443

(Gestopt: Daniël Vermeulen – 4100808)

1 Voorwoord

Onze groep, bestaande uit Technische Informatica studenten uit verschillende jaren, is begonnen aan het Imaging Project omdat we allemaal voor het Imaging variantblok gekozen hebben. Vorig jaar ging dit project over de herkenning van verschillende handposities, en daarmee een computer verschillende functies uit laten voeren. Dit jaar is het echter de bedoeling een camerasysteem te implementeren met een toepassing in het verkeer in het Mekelpark.

Iedereen uit het groepje wisselt van taken in overleg. Dit wordt besloten aan het begin van de projectochtend tijdens te vergadering.

We willen graag elkaar in het groepje, onze begeleiders en de gebruikers van de liften van EWI bedanken.

1	VOORWOORD.....	2
2	INLEIDING.....	5
3	PROBLEEMSTELLING.....	5
4	PROGRAMMA VAN EISEN.....	6
4.1	VISUALISATIE PROGRAMMA VAN EISEN	7
5	TESTPLAN	9
6	FUNCTIEBLOKSCHEMA	10
7	PROTOTYPES.....	11
7.1	PROTOTYPE 1.....	11
7.2	PROTOTYPE 2	12
7.3	PROTOTYPE 3	13
7.4	PROTOTYPE 4	14
7.5	PROTOTYPE 5	16
7.6	PROTOTYPE 6	17
8	IMPLEMENTATIE KEUZES.....	20
8.1	NORMALISATIE KEUZES	20
8.1.1	<i>Methodes.....</i>	<i>20</i>
8.1.2	<i>Uiteindelijke keuze.....</i>	<i>21</i>
8.2	LIFTSEGMENTATIE KEUZES	21
8.2.1	<i>Uiteindelijke keuze.....</i>	<i>22</i>
8.3	PERSOONSSEGMENTATIE KEUZES	22
8.3.1	<i>Methodes.....</i>	<i>22</i>
8.3.2	<i>Uiteindelijke keuze.....</i>	<i>24</i>
8.4	KEUZE VOOR HET NIET GEBRUIKEN VAN SKELETONS	25
8.5	LIFTSTATUS KEUZES	25
8.6	PERSOONSDetectie KEUZES.....	27
8.6.1	<i>Filtering op basis van object-size.....</i>	<i>27</i>
8.6.2	<i>Afkeuring van false-positives.....</i>	<i>27</i>
8.6.3	<i>Verdere Benadering.....</i>	<i>28</i>
8.7	PERSOONSTELLING KEUZES	29
8.7.1	<i>Verzamelen van Data</i>	<i>29</i>
8.7.2	<i>Error-detectie en oplossing.....</i>	<i>29</i>
8.7.3	<i>Verkeersberekening.....</i>	<i>30</i>
8.7.4	<i>Alternatieve persoonstelling.....</i>	<i>30</i>
9	EVALUATIE EN TESTEN SYSTEEM.....	32
10	CONCLUSIE EN AANBEVELING.....	33
11	LITERATUURLIJST	34
12	FIGUREN EN TABELLEN	35
12.1	FIGUUR 1 - FUNCTIEBLOKSCHEMA	35

12.2	TABEL 1 - TESTRESULTATEN	36
13	BIJLAGEN	37
13.1	EIGEN EINDBEOORDELINGEN	37
13.1.1	<i>Marnix de Graaf</i>	37
13.1.2	<i>Mirko Dunnewind</i>	38
13.1.3	<i>Raoul Harel</i>	38
13.1.4	<i>Tim Rensen</i>	39
	BEOORDELING GROEP	39
	EIGEN BIJDRAGE.....	40
13.2	LOGBOEK.....	40

2 Inleiding

Beeldherkenning wordt dagelijks en overal gebruikt. Denk niet alleen aan streepjescodes en QR-codes, maar ook bijvoorbeeld aan kentekenherkenning in een flitspaal. In de toekomst zullen er wellicht volledig automatische auto's bestaand die met behulp van camera's over de weg kunnen navigeren. Het verwerken van beelden door computers zal een deel van ons leven worden.

Er wordt op dit moment veel onderzoek gedaan om beeldherkenning beter te maken. Het zal ingezet worden voor taken die voor mensen lastig zijn te doen of die teveel tijd kosten. Hierbij kan je denken aan het identificeren van personen met behulp van enkel een foto. Beeldherkenning gaat in de toekomst een hele belangrijke rol spelen in de industrie – dat is zeker.

3 Probleemstelling

Iedereen die wel eens gebruik maakt van de liften op EWI weet dat ze ongelooflijk snel zijn, maar ook dat het wachten op de lift kan ongelooflijk lang duren. Het is natuurlijk voor te stellen dat niet elke lift even vaak wordt gebruikt. Ook het aantal personen dat gebruik maakt van een lift kan erg verschillen.

Voor het efficiënter maken van de liften is het handig om te weten hoe vaak een lift naar een verdieping gaat en hoeveel personen er daar gebruik van maken. Door middel van het analyseren van camerabeelden is het mogelijk om deze data te verkrijgen.

Door de camera met een live systeem in te zetten op verschillende verdiepingen en liften (niet tegelijkertijd) is het later mogelijk een totaalbeeld te krijgen van het lift-verkeer. Dat kan dan gebruikt worden voor de optimalisatie van het liftsysteem.

De probleemstelling:

Is het mogelijk om te detecteren hoe vaak en door hoeveel personen één personenlift in EWI gebruikt wordt tijdens een bepaald interval op één verdieping?

4 Programma van eisen

Performance-eisen:

1. Het systeem telt de mensen die de lift in- en uitlopen en toont deze gegevens in de GUI
2. In minimaal 75% van de gevallen wordt het juiste aantal personen geteld
3. Het systeem moet real-time werken. Met real-time wordt bedoeld: de gecombineerde reactie- en uitvoertijd van de beeldverwerking is kleiner dan 2 seconden.
4. Het systeem moet de bewegingen van de liftdeuren detecteren van de lift waaraan de metingen gedaan worden
5. Het systeem moet kunnen werken met video- en camerabeeld
6. De gebruiker kan via de GUI het systeem starten en stoppen
7. De GUI toont vier beelden, waaronder het beeld vóór, tijdens en na bewerking door het systeem

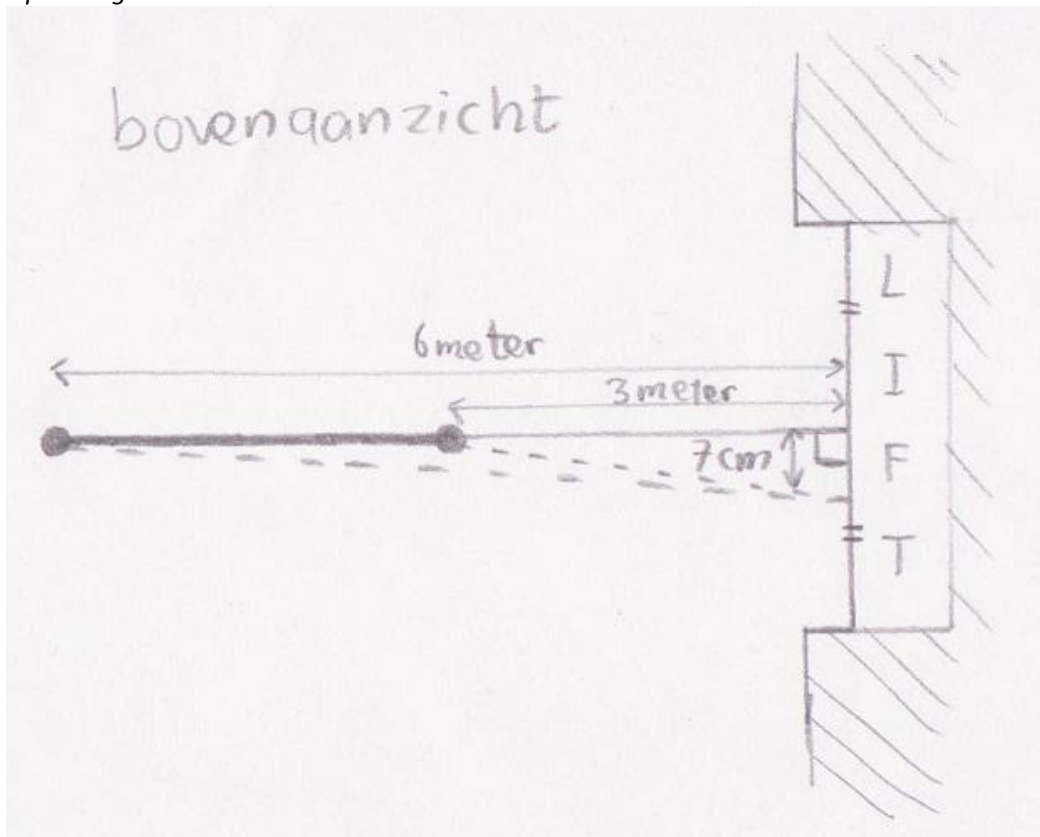
De bovenstaande performance wordt gehaald als er voldaan wordt aan de volgende eisen:

1. Er moet gebruik worden gemaakt van Matlab R2011b
2. Het systeem moet uit kunnen draaien op een TU-computer in zaal 0.010 van Drexelweg, of op een computer met betere specificaties
3. De camera moet precies één volledige persoonslift van EWI in beeld hebben
4. De lift aan de hand waarvan de metingen gedaan worden, moet operationeel zijn gedurende de tijd dat het systeem aan staat
5. Er bevinden zich maximaal 5 mensen tegelijkertijd in beeld om de vereiste performance te halen
6. Bij de start van het systeem mogen er geen voorwerpen tussen de lens en de lift staan
7. Er bevinden zich na initialisatie geen mensen of voorwerpen die het beeld voor 60% of meer in beslag nemen
8. De lichtintensiteit van de omgeving waarin de camera op gericht staat, moet tussen 80 lux (vergelijkbaar met een kantoor of gang) en 10.000 (vergelijkbaar met vol daglicht) lux liggen
9. Er mag geen licht met een lichtintensiteit hoger dan 10.000 lux direct geschenen worden in de lens van de webcam
10. De camera wordt geplaatst zoals aangegeven in de onderstaande tekeningen
11. De camera moet zich op 2.00 meter hoogte ten opzichte van de vloer, van de verdieping waar de camera staat, bevinden buiten de lift (zie onderstaande tekeningen).
12. De camera moet gepositioneerd zijn op de as die loodrecht uit het midden van de lift naar buiten de lift wijst met een maximale afwijking van 10 cm in de horizontale en/of verticale richting (zie onderstaande tekeningen).
13. De lens van de camera moet op het midden van lift gericht staan met een maximale afwijking van 7 cm in horizontale en/of verticale richting ten opzichte van de lens naar de lift over de draaias van de camera (zie onderstaande tekeningen).
14. De camera moet zich op minimaal 3 en maximaal 6 meter afstand bevinden van de liftdeuren, buiten de lift (zie onderstaande tekeningen).
15. De hoek van de camera mag maximaal 0.2 graad afwijken van de originele geplaatste opstelling na plaatsing.
16. De camera mag na de initiële plaatsing niet meer dan 0.1 cm bewegen in elke richting
17. Er mogen zich geen objecten binnen 0.5 meter voor de lens bevinden

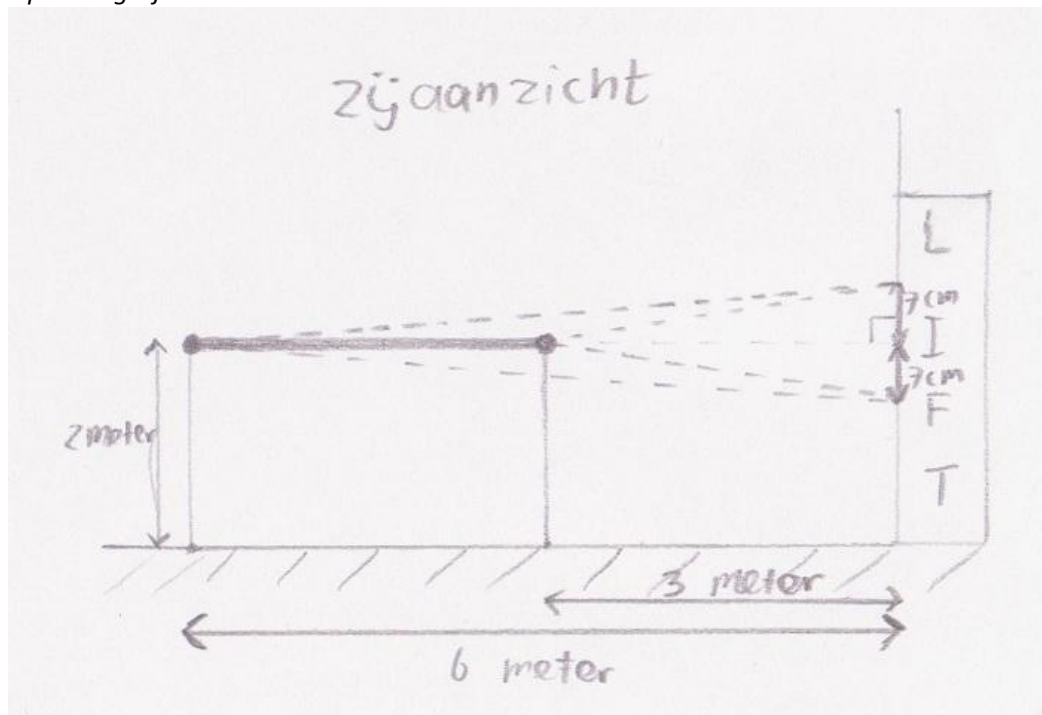
18. De lucht dat de camera waarneemt, moet vrij zijn van rook
19. Er mogen zich geen calamiteiten (bijv. brand, overstromingen, stroomuitval) voordoen in het gebouw
20. De personen lopen op hun voeten
21. De personen lopen niet met hun armen naar buiten of naar boven gericht
22. Een mens moet minimaal 2 seconden volledig in beeld zijn om herkend te worden door het systeem
23. De menselijke vormen (lichaam, hoofd) moeten goed herkenbaar zijn
24. Het beeldscherm van het systeem moet een minimale resolutie van 800x600 pixels hebben
25. De camera moet scherp gesteld zijn op de lift
26. De liftkleur mag niet anders zijn dan de kleur op 6 december 2012
27. Het model van de camera moet van het type Philips SPC700NC zijn
28. De cameraresolutie moet 320x240 bedragen op 5 fps
29. Ingeladen video's moeten een resolutie hebben van 320x240 op maximaal 30 fps

4.1 Visualisatie programma van eisen

Opstelling bovenaanzicht



Opstelling zij aanzicht



5 Testplan

Voor het testen van het uiteindelijke systeem (en de eventuele eerdere prototypes) moet er aan bepaalde eisen voldaan worden. Zo zullen logischerwijs alle tests uitgevoerd worden onder de omstandigheden die in het plan van eisen staan beschreven. Hierbij wordt bij elke test simpelweg bijgehouden hoeveel procent van de ingaande en uitgaande personen in de lift er correct wordt herkend. Daarnaast moet worden opgelet of het systeem niet de mensen gaat tellen die simpelweg voorbij lopen en dus geen aandeel hebben in het op te lossen probleem.

Voor de testsituaties die zullen worden gegeven, definiëren we het volgende:

Herkennen/herkenning: dit houdt in dat het systeem denkt dat er 1 persoon in- of uit de lift loopt. Dit zal de respectievelijke teller ophogen met 1. Het tegenovergestelde hiervan is dat de tellers niet veranderen en dat het systeem dus denkt dat er niemand in- of uit de lift loopt.

De voorgedefinieerde situaties spelen simpelweg in op de enkelvoudige basisfuncties van het systeem. Deze tests zullen dan ook worden uitgevoerd door ons zelf als "figurant". De volgende tests zullen op zijn minst worden uitgevoerd:

- Lift is dicht en er zijn geen personen in beeld (er vindt geen herkenning plaats).
- Lift gaat open en dicht zonder dat er een personen in beeld zijn (er vindt geen herkenning plaats).
- Lift gaat open en dicht terwijl er geen personen in beeld zijn, behalve 1 enkel persoon in de lift (er vindt geen herkenning plaats).
- De lift gaat open en er stapt 1 enkel persoon uit de lift, verder zijn er geen andere personen in beeld (het systeem herkent 1 persoon die uit de lift loopt).
- Er loopt 1 enkel persoon de lift in vanaf buiten het beeld, verder zijn er geen andere personen in beeld (het systeem herkent 1 enkel persoon die de lift in loopt).
- Er staat 1 enkel persoon stil (niet lopend) voor de lift, terwijl die open en dicht gaat, verder zijn er geen andere personen in beeld (er vindt geen herkenning plaats).
- De lift is dicht en er loopt 1 enkel persoon voor de lift langs, parallel met de muur op 1 meter afstand van de muur, verder zijn er geen personen in beeld (er vindt geen herkenning plaats).
- De lift is open en er loopt 1 enkel persoon voor de lift langs, parallel met de muur op 1 meter afstand van de muur, verder zijn er geen personen in beeld (er vindt geen herkenning plaats).
- Er loopt 1 enkel persoon, met een T-shirt met een klein contrast tot de muur, de lift in vanaf buiten het beeld, verder zijn er geen personen in beeld (het systeem herkent 1 persoon die de lift in loopt).

Wanneer aan deze minimalistische tests is voldaan en het systeem voor al deze situaties werkt, is het van belang een grootschaligere test te doen. Er worden dan combinaties van de verschillende tests gedaan.

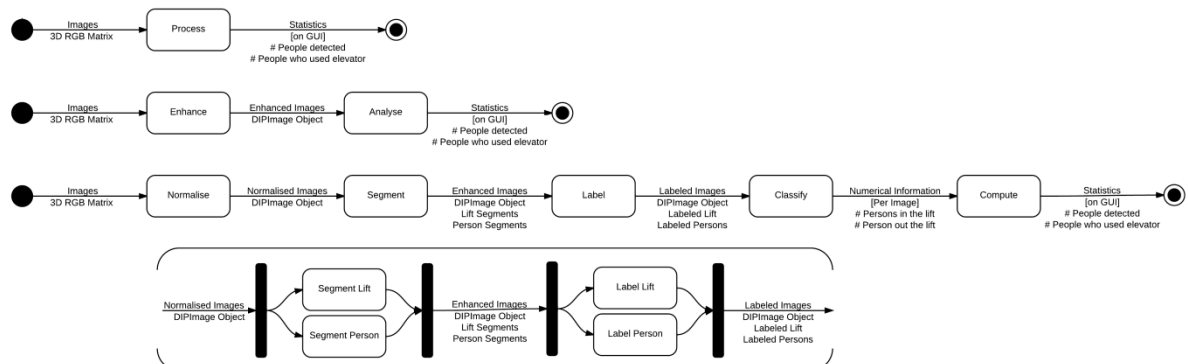
De volgende combinaties moeten in ieder geval getest worden:

- De lift gaat open en er stappen 2 personen uit de lift, verder zijn er geen andere personen in beeld (het systeem herkent 2 personen die uit de lift lopen).
- Er stappen 2 personen in de lift en de lift gaat dicht, verder zijn er geen andere personen in beeld (het systeem herkent 2 personen die in de lift lopen).

- De lift gaat open, er stapt 1 persoon uit en 1 persoon in, dan gaat de lift dicht, verder zijn er geen andere personen in beeld (het systeem herkent 1 persoon die de lift uit loopt en 1 persoon die er in loopt).
- De lift gaat open en er staat 1 persoon in de lift, 1 ander persoon loopt uit de lift, verder zijn er geen andere personen in beeld (het systeem herkent 1 persoon die uit de lift loopt).
- De lift gaat open en dicht en er staan 2 personen in de lift, verder zijn er geen andere personen in beeld (er vindt geen herkenning plaats).
- De lift gaat open en er lopen 2 personen uit, 1 persoon is 2 meter lang, de ander 1,5 meter lang, verder zijn er geen andere personen in beeld (het systeem herkent 2 personen die uit de lift lopen).
- Er lopen 2 personen met een redelijk gelijk shirt de lift in, de lift gaat dicht, verder zijn er geen andere personen in beeld (het systeem herkent 2 personen die de lift in lopen).
- De lift gaat open en er lopen 2 personen uit met een shirt met een klein contrast met de muur, verder zijn er geen personen in beeld (het systeem herkent 2 personen die de lift uit lopen).

Ook aan deze eisen moet worden voldaan.

6 Functieblokschema



In de figuren (12.1 Figuur 1) staat een grote versie van het functie blokschema.

7 Prototypes

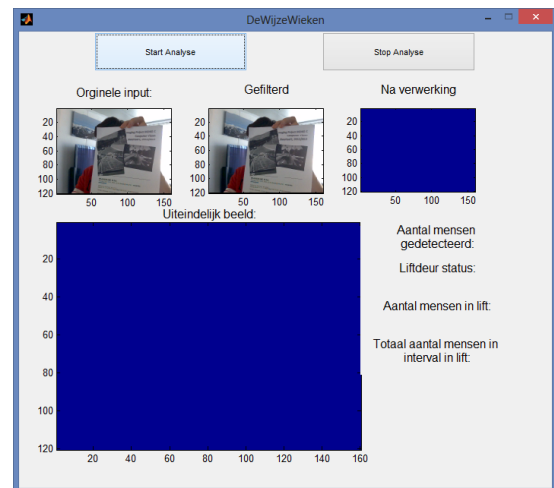
Door het project heen is er elke week een prototype afgeleverd. Elk prototype is apart beschreven met zijn werking en features. Hierbij worden ook een aantal van de keuzes belicht die er zijn gemaakt. De rest van de keuzes zijn per onderdeel van de beeldverwerking te vinden in het volgende hoofdstuk “Implementatie keuzes”.

7.1 Prototype 1

Donderdag 22 november is prototype 1 afgeleverd. In dit hoofdstuk is te lezen wat er in dit prototype verwerkt is.

Prototype 1 bevat onder andere een werkende GUI, die vier beelden toont: Originele beeld, gefilterd beeld, verwerkt beeld en het uiteindelijke beeld. Dit is te zien in *Figuur 1*. De ‘originele input’ laat de beelden zien die rechtstreeks van de webcam naar de computer worden gestuurd. Het ‘Gefilterd’ laat het beeld na de verwerking *normalise* zien. ‘Na verwerking’ bevat het beeld na *segmentation*. Wat deze functies precies doen, wordt hieronder beschreven.

Prototype 1 bevat vier dummy-functies: *normalise*, *property*, *classification* en *count*. Voor deze operaties geldt: *input = output*. Met het beeld dat binnen komt, wordt dus niets gedaan en wordt gewoon teruggestuurd door deze functies. Later sturen deze functies verwerkte beelden door naar de volgende functie, die er ook een bewerking mee doen.



Figuur 1 - De werkende GUI van Prototype 1

In dit prototype zitten verder nog twee functies die wel iets met het beeld doen: *labeling* en *segmentation*. Zoals te zien in *Figuur 1* maakt de functie *segmentation* het beeld zo goed als blauw. *Segmentation* zou de lift moeten herkennen qua kleur. Op dit moment worden alleen de kleuren van de lift soms met iets lichter blauw getoond. *Segmentation* werkt door eerst het beeld op te splitsen in 3 nieuwe beelden die de losse RGB kanalen zijn van het invoer beeld. Hier wordt vervolgens afzonderlijk een threshold op toegepast en de beelden worden weer samengevoegd. Hier zou een gesegmenteerd beeld uit moeten komen. Doordat *segmentation* niet de juiste informatie retourneert, kan de functie *labeling* (te zien bij ‘Uiteindelijk beeld’) ook haar werk (vaststellen wat de lift is) niet uitvoeren. De functie *labeling* werkt door een label functie uit te voeren over het beeld dat binnenkomt.

Evaluatie

Momenteel werkt er nog niet erg veel in het prototype. We hebben een werkende GUI die in ieder geval het originele beeld en enkele verwerkte beelden kan tonen. Ook zijn er in de code al dummy functies gemaakt om later code in te plaatsen, waardoor de structuur van het programma vanaf nu al duidelijk is. De segmentatie- en labelingsfuncties zullen flink verbeterd moeten worden, willen andere functies hiervan gebruik kunnen maken, die wel nodig zijn voor het uiteindelijke doel van dit programma. De basis is nu gelegd, en nu kunnen er nieuwe functies gemaakt en verbeterd worden.

7.2 Prototype 2

Donderdag 29 november is prototype 2 ingeleverd. In dit hoofdstuk wordt het verschil met prototype 1 en de toevoegingen beschreven.

De user interface is in dit prototype iets aangepast. Er is een knop bijgekomen waarin 'Capture Background' staat. Bij het drukken op deze knop wordt één nieuw beeld opgeslagen dat van de webcam komt. Het is idee achter deze functie is: het opslaan van de achtergrond zonder mensen erop. Deze knop is tijdelijk, aangezien het opslaan later geautomatiseerd zal gaan worden.

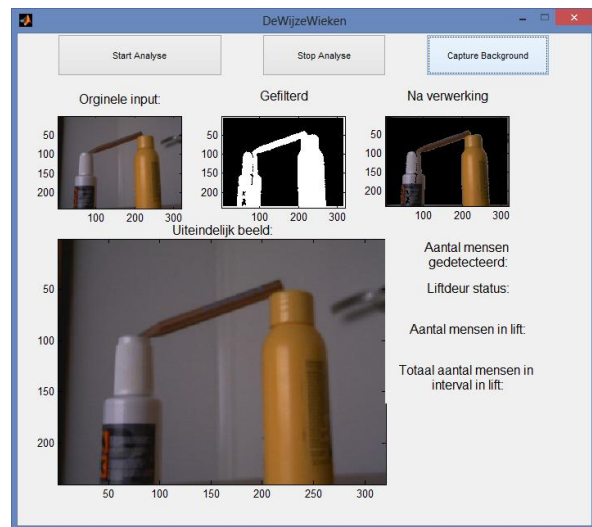
Het beeld bij 'Gefilterd' is een binaire representatie van het verschil tussen een eerder opgeslagen beeld en de huidige invoer. Het filtert dus wezenlijk de achtergrond weg waardoor interessante objecten(mensen) overblijven. Dit wordt gedaan door van de huidige genormaliseerde invoer twee nieuwe beelden te maken: één beeld is de opgeslagen achtergrond minus het huidige beeld, en de ander het huidige beeld minus de opgeslagen achtergrond. Door op de losse RGB kanalen van deze beelden een threshold toe te passen hou je alleen nieuwe objecten in het beeld over. Het idee hiervoor komt uit 'Tracking Groups of People' door Stephan J. McKenna in het jaar 2000. Ideeën uit dit artikel zullen later meer gebruikt worden, bijvoorbeeld bij het daadwerkelijk kunnen tellen van mensen in groepen, omdat het doel van dit artikel redelijk overeenkomt met wat wij proberen te doen.

Bij 'Na verwerking' wordt het beeld weergegeven dat gemaakt is door het beeld bij 'Gefilterd' als mask op de huidige genormaliseerde invoer toe te passen. Hierdoor houd je dus een beeld over wat alleen nieuwe objecten laat zien, maar dan ook in kleur.

Door te normaliseren houd je een beeld over wat altijd dezelfde lichtintensiteit heeft. Dit verbetert het herkenningsproces aanzienlijk. Als er bijvoorbeeld een liftdeur opengaat is ineens veel meer licht. De camera past zich hier automatisch op aan door de intensiteit over het gehele beeld te verlagen. Hierdoor is echter de rest van de achtergrond ook donkerder, wat door de segmentatiecode wordt opgepakt als een nieuw object. Dit is uiteraard niet de bedoeling.

Het normaliseren gebeurt door op dit moment door de gemiddelde intensiteit van alle pixels en alle kleurlagen te nemen en dit dan door een lineaire verhogen of verlagen bij te stellen naar de vastgestelde standaard intensiteit van 120. Dit werkt niet helemaal correct met de zojuist geïmplementeerde methode om het beeld van de achtergrond te ontzien. Wat er dus voor het volgende prototype gedaan moet worden is het live beeld normaliseren aan de hand het gemaakte achtergrondbeeld. Verdere uitleg over gebruikte normalisatie van het begin tot het eind van het project is te vinden in het hoofdstuk over implementatiekeuzes.

Om goed te kunnen tellen hoeveel mensen er nu precies de lift in stappen is het handig om te weten of de liftdeuren open zijn. Hiervoor is het functie LiftDetect geschreven. LiftDetect gebruikt thresholding op een bepaalde kleur (namelijk de kleur van de liftdeuren) om de lift-segments uit de



frame te kunnen brengen. Deze segments worden dan geanalyseerd door het functie liftVisible. LiftVisible meet een aantal eigenschappen (bijv. oppervlakte en omtrek) van deze segments om daaruit te kunnen concluderen of de liftdeuren open of dicht zijn. Deze functionaliteit is echter nog niet in het prototype gebracht omdat dit momenteel fouten oplevert.

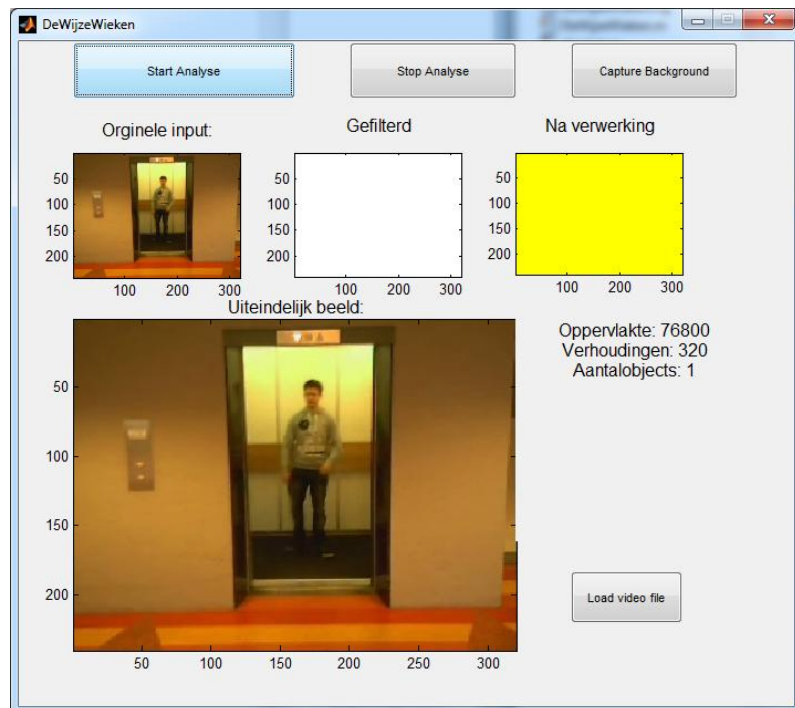
Hoewel het wegfilteren van de achtergrond nu goed werkt moet het statistieken verzamelen nog geïmplementeerd worden. Ook moet er nog een closing toegepast worden op de verwerkte beelden om dit goed te kunnen verwerken. Verder mist er ook nog een markering wat het systeem als mens herkent, wat handig is om de telling te kunnen controleren. Kortom, nog niet alles is al geïmplementeerd, maar wat er in zit werkt al wel goed.

7.3 Prototype 3

Donderdag 6 november is prototype 3 ingeleverd. In dit hoofdstuk wordt het verschil met prototype 2 en de toevoegingen beschreven.

Allereerst is de user interface bijgewerkt. Deze bevat nu een nieuwe “Load video” knop die mogelijk maakt video’s in het systeem te kunnen laden en te analyseren. Ten tijde van het inleveren van prototype 3 werkte de functionaliteit achter deze knop echter nog niet zonder de GUI uiteindelijk te laten crashen.

De beelden “Gefilterd” en “Na verwerking” in de GUI geven andere gegevens weer. Na gedrukt te hebben op de knop “Capture Background”,



waardoor de voorgrond analyse gestart wordt, toont beeld “Gefilterd” het opgeslagen beeld nadat die verwerkt was door *liftSegment*. *LiftSegment* neemt een beeld op en segmenteert een lift daaruit op basis van kleur (veel rood, weinig blauw en groen). Het beeld “Na verwerking” toont de persoon-segmenten uit de huidige beeld. Persoon-segmentatie wordt gedaan door de functie *segmentPerson*.

SegmentPerson werkt nog redelijk hetzelfde, en is alleen aangepast om met andere type images om te gaan. Bovendien wordt ruis weggehaald door eerst enkele keren een erosie uit te voeren en vervolgens weer een zelfde aantal dilaties.

Naast deze toevoegingen toont de applicatie nu ook een paar statistieken. Deze zijn de oppervlakte van de persoon-segmenten en ook de verhouding (ratio's) van deze.

Andere implementaties die niet zichtbaar zijn in de GUI zijn de functies *analyze*, *classify* en *compute*. De functie *classify* bepaalt het aantal mensen die binnen- of buiten de lift staan op een bepaalde

moment. Dit wordt gedaan door de centrum van elke persoon te vergelijken met de grens van de liftsegment. Als die binnen valt wordt aangenomen dat de hele mens ook binnen ligt.

De functie compute neemt de verzamelde gegevens uit classify en probeert het werkelijke aantal mensen die de lift gebruikt hebben te schatten. De data uit classify wordt opgeslagen in een lijst, waar de gegevens in elke index gelijk zijn aan de output van de functie voor één frame uit de laatste 30 frames. De functie haalt het meest voorkomende aantal mensen binnen de lift per frame. Dit is een schatting van de werkelijke aantal mensen die in de lift staan. Als het aantal mensen binnen de lift in het huidige frame nul is, is de conclusie dat de liftdeuren dicht zijn en dat de lift nu dat aantal mensen bevat.

Tot slot neemt de functie analyse de gegevens uit compute, voegt ze toe aan de huidige statistieken en maakt ze zichtbaar op de GUI. Analyse is een containerklasse die verschillende andere klassen aanroept.

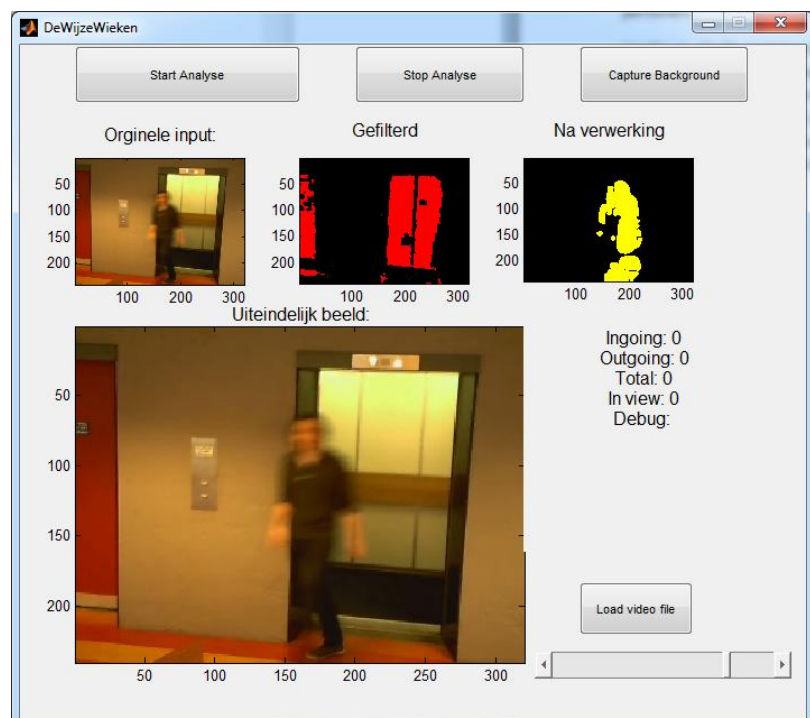
Doordat de code flink onoverzichtelijk begon te worden, en er problemen optraden met veel aan hetzelfde bestand werken is alle code geherstructureerd. Zo zijn er meer losse klassen en vallen kleinere klassen onder grote containerklassen. Dit maakt het ook makkelijker om bepaalde functionaliteit specifieker testen, en om bepaalde gedeeltes van het programma aan of uit te zetten.

Het wegfilteren van de achtergrond levert nog veel ruis op, en is redelijk traag. Hier moet nog flink aan gewerkt worden. Verder is het niet heel nuttig om twee keer hetzelfde beeld te laten zien, maar de plaatsen zijn wel nuttig voor later. Er kan momenteel nog niet bepaald worden waar een persoon zich bevindt, alleen of, en hoeveel personen er zijn. Daarvan worden nu dus ook nog geen statistieken getoond, die er wel in moeten komen te zitten.

7.4 Prototype 4

Donderdag 13 november is prototype 4 ingeleverd. In dit hoofdstuk wordt het verschil met prototype 3 en de toevoegingen beschreven.

Nieuw in prototype 4 is de classify die kan herkennen of iemand binnen of buiten de lift staat. Hier zijn drie methoden voor ontwikkeld. De eerste, en tevens ook de simpelste methode gebruikt het zwaartepunt van een object dat herkend is als persoon om te bepalen of de persoon in de lift staat of er buiten. Hierbij wordt simpelweg gekeken of dit zwaartepunt overlap heeft met de liftdeuren of niet. Deze methode wordt echter niet gebruikt voor prototype 4. In prototype 4 wordt de tweede



ontwikkelde methode gebruikt die niet alleen de locatie van het zwaartepunt ten opzichte van de lift gebruikt, maar ook de Cartesian box van een object dat herkend is als persoon om te bepalen wat de afstand en locatie is van een persoon ten opzichte van de lift. Deze box maakt het mogelijk voor het systeem om te bepalen of het object voor of in de lift staat in het geval dat het zwaartepunt van het object overlapt met het lift-object. De derde methode die ontwikkeld is bepaalt hoeveel mensen er in of uit de lift zijn gegaan door te kijken of het middelpunt van het persoon-object in het vlak van de lift gepositioneerd is en of de oppervlakte van dat object krimpt of als er een nieuw object bij komt of deze groeit. Dit is namelijk het effect van de liftdeuren; de objecten die in de lift staan worden kleiner of groter als de lift sluit of open gaat. Het idee is om bij deze methode ook nog de Cartesian box van de persoon-objecten te gebruiken om te kijken of ze binnen of buiten de lift staan.

Met de gegevens van de classify-methode is er een compute-methode opgesteld die aan de GUI de statistieken “Ingaand verkeer”, “Uitgaand verkeer”, “Totaal verkeer”, “Aantal mensen in beeld” en “Debug” (die laat zien hoeveel gegevens er nog in het geheugen geplaatst zijn) doorgeeft, waarna de GUI deze aan de rechterkant presenteert.

Er is besloten dat er geen autocalibratie meer nodig is zoals eerder bedacht. In het PvE is vastgesteld dat de camera zeer weinig mag bewegen, en autocalibratie die eerder gebouwd was leverde meer problemen op dan het oploste. Bovendien is er overgestapt op beweging voor segmentatie(hierover later meer) waardoor autocalibratie sowieso overbodig werd.

Voor prototype 4 is er een nieuwe methode voor de lift-detectie bedacht, echter is deze nog niet ingebouwd in prototype 4. Het idee achter deze nieuwe methode is om niet alleen de lift simpelweg zo goed mogelijk te segmenteren en te labelen, maar deze te voorzien van een kader om de lift heen. Hiervoor wordt eerst door een threshold bepaald welk deel van het beeld de lift is, waarna eventuele ruis met een opening weggewerkt wordt. Vervolgens worden de hoekpunten van het lift-object bepaald aan de hand van de minimum en maximum waarden voor het objecten en wordt er tussen deze hoekpunten een kader getekend op het scherm. Dit kader maakt duidelijk voor de gebruiker wat het systeem herkend heeft als lift en geeft een rechthoekig vlak met duidelijk gedefiniëerde hoekpunten waar eventueel andere delen van het systeem van kunnen profiteren. Bijvoorbeeld het onderdeel dat detecteert welke mensen er gebruikmaken van de lift is hierbij gebaat, aangezien er nu een vlak beschikbaar is in de systeemgegevens waarin de persoon-objecten moeten liggen, willen deze in aanmerking komen voor de bepaling of er gebruik gemaakt wordt van de lift.

Net als voorgaande prototypes, bevat ook prototype 4 weer een groot deel segmentatie-onderdelen. De segmentatie is een van de belangrijke delen van het gehele proces. Wanneer de segmentatie niet correct werkt, kan het al snel verkeerde data voor de vervolg processen opleveren. Het is daarom van belang dat altijd de juiste data voor de vervolg processen wordt geleverd. Tot aan dit prototype zijn 4 verschillende methodes gebruikt.

Als eerste is er geprobeerd het inkomende frame van het gekalibreerde frame af te trekken en het resultaat absoluut te maken, en hierbij de ruis weg te filteren. Dit heeft als nadeel dat als de lift opengaat dit herkend wordt als een groot object, en dat mensen met bijvoorbeeld een grijs shirt niet herkend worden voor een grijze achtergrond omdat er te weinig contrast is. Om dit op te lossen is bij de tweede methode ook bewegingsherkenning toegevoegd, en zo worden beide methodes samen gevoegd. Het nadeel hiervan is dat mensen niet herkend worden als ze stil staan. Als alternatief is er een methode ontwikkeld die kijkt naar het verschil tussen 2 beelden, waarvan de een verticaal

verschoven was. Maar met deze methode was niet rekening gehouden met egale oppervlakten als de muur en de liftdeur die dan ook als bewegend zouden zien omdat een groot deel van dit oppervlakte weinig verschilt als het wordt bewogen. Deze methode is dus niet nuttig. Die laatste methode is uitgebreid met een dilatie met een lang verticaal structurerend object om zo alleen mensen te herkennen. Dit geeft al een stuk beter resultaat dan eerst, al zitten er nog wel gaten in de objecten, voornamelijk wanneer er sprake is van een egaal oppervlakte. In dit prototype is de laatste methode geïmplementeerd omdat deze het beste werkt tot nu toe. Een uitgebreidere uitleg over de segmentatiemethodes is te lezen in het hoofdstuk over implementatiekeuzes.

De segmentatie is nog steeds niet optimaal en kan verbeterd worden. Hier zal dan ook nog naar gekeken worden. De resultaten van de statistieken zijn veelbelovend, maar ook deze zullen nog flink geoptimaliseerd en verbeterd moeten worden. In de lift is de herkenning heel slecht. Daarbuiten bevat deze prima, hoewel de schaduw nog steeds lastig is. Verder is het handig voor toekomstig gebruik als we weten of de lift open of dicht is.

7.5 Prototype 5

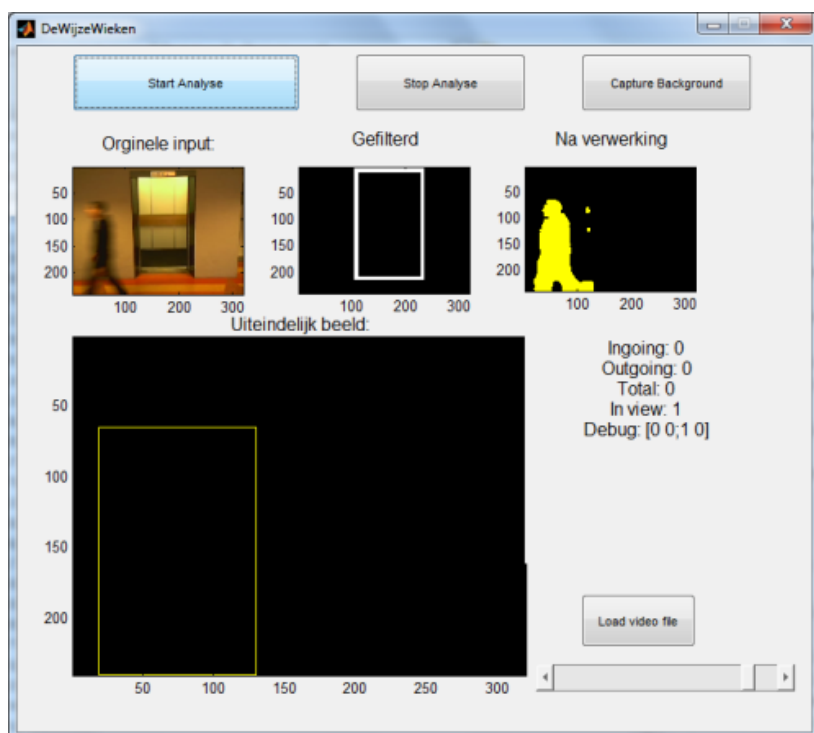
Op donderdag 20-12-12 is prototype 5 ingeleverd. In dit hoofdstuk wordt het verschil met prototype 4 en de toevoegingen beschreven.

In de gui is in prototype 5 iets aangepast voor testdoel-einden, maar er is al besloten dit er niet tot in de laatste versie in te laten zitten. Zo word nu in het grote beeld aangegeven waar een persoon herkent wordt door een vierkant (een verkleinde cartesian box) rondom een herkent persoon aan te geven.

De box wordt, zoals beschreven bij prototype 4, gebruikt om te tellen of iemand binnen of buiten de lift staat, en het bleek dat de herkenning verbeterde als de box iets

kleiner gemaakt wordt, daarmee rekening houdende met schaduw. Mocht iemand zich niet naast de lift bevinden, kan door te kijken of de box groter is dan de lift (de persoon staat buiten) of kleiner (de persoon staat in de lift) bepaalt worden of deze persoon wel of niet de lift in is gegaan. Het kleiner maken van de box is in dit prototype toegevoegd.

Code die in prototype 4 wel grotendeels af was maar nog niet in gebruik is in dit prototype wel geïmplementeerd. Het gaat hierbij om Analyse.m. Daardoor toont de gui nu wel statistieken. Analyse roept functies aan die mensen en de lift labelt en analyseert hoeveel mensen in en uit de lift gaan.



De code uit `normalise.m` is grotendeels commentaar gemaakt omdat ook deze code weinig nut had. Er wordt in principe niet meer genormaliseerd omdat dit in de instellingen van de camera zelf te regelen is, en dat veel betere resultaten oplevert.

De functie `compute.m` is sneller gemaakt. In het vorige prototype werd het aantal mensen dat in en uit de lift gingen geteld door informatie uit alle frames te gebruiken. Nu wordt daar een bepaald aantal laatste frames voor gebruikt.

In dit prototype is veranderd hoe de lift gedetecteerd wordt. In plaats van herkenning op basis van kleur (namelijk de herkenning van veel rood en weinig blauw en groen) is er nu herkenning op kleurovergangen ingebouwd. Deze aanpak heeft als voordeel dat de lijnen van de lift erg goed zichtbaar zijn, hoewel deze methode wel meer rekenkracht kost. Het verschil in resultaat is niet erg groot; bij deze methode is de lift wat groter en de lijnen iets nauwkeuriger. Dit leverde net iets beter resultaat op en daarom is dit veranderd in dit prototype. Meer over deze keuze is te lezen in het hoofdstuk over implementatiekeuzes.

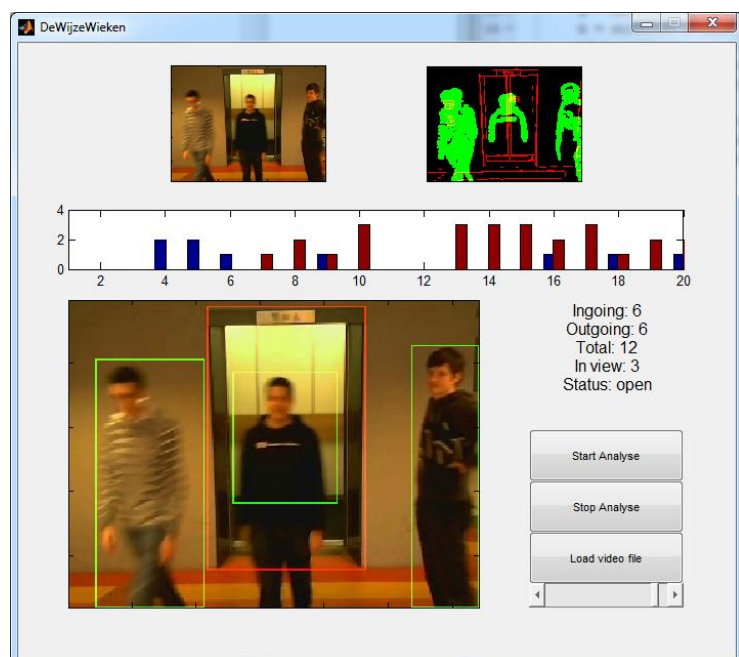
In `segmentPerson.m` is het structurerend element aangepast naar een veel groter element dan in het vorige prototype, en is de erosie verwijderd. Dit omdat er anders teveel blur kwam tussen personen.

Verder is uit dit prototype niet meer gebruikte, verouderde en overbodige code weg gehaald. Hoewel het prototype steeds meer naar wens wordt, moeten er nog wel een aantal dingen verbeteren. De detectie van de lift zelf werkt nu naar behoren, alleen de detectie van mensen in de lift gaat niet tot heel slecht omdat de lift zelf ook als object herkent wordt. Dit moet in het volgende prototype dus verbeterd worden. Verder zou het tellen van mensen nog geoptimaliseerd kunnen worden. Dit kan mede gedaan worden door de status van de liftdeuren goed te bepalen.

7.6 Prototype 6

Op zondag 13-1-13 is prototype 6 ingeleverd. In dit hoofdstuk wordt het verschil met prototype 1 en de toevoegingen beschreven.

In dit laatste prototype is de GUI naar de uiteindelijke layout veranderd. Hierbij is linksboven het originele beeld te zien, rechts de liftherkenning met daarboven op de bewegingsherkenning en linksonder het originele beeld met daar overheen de door het programma als mensen herkende personen, en het gebied dat door het programma als lift herkent is. Ook is boven het grote beeld een balk met daarin informatie over wat het programma weet over de hoeveelheid mensen per frame, namelijk of deze zich binnen of buiten de lift bevinden. Rechts zijn de statistieken te zien en is het programma te bedienen.



In vorige versies werd gekeken of er een lift was door de functie liftvisible. Dit werkte niet bepaald precies: de functie keek of er mensen in beeld waren. De nieuwe vervangen functie, liftDetect, werkt ook daadwerkelijk aan de hand van de lift zelf. In de bovenste 30% van de liftbounding box wordt gemeten hoe groot de herkende objecten zijn, waarbij de objecten de liftdeuren zijn. Zijn deze erg groot, dan zijn de liftdeuren dicht, worden deze kleiner dan gaan de liftdeuren open, worden ze niet herkend zijn de liftdeuren open en als ze weer groter worden gaan de liftdeuren dicht. Hierover is meer te lezen in het hoofdstuk over implementatiekeuzes.

Er is ook een andere segmentatie doorgevoerd. Er is heel veel werk gestoken in de segmentatie vanaf het begin, alleen hiervan wordt weinig meer gebruikt. Tijdens het ontwikkelen in de laatste week bleek de nieuwe segmentatie, ondanks veelbelovende resultaten, niet goed te werken binnen de lift. Er was voor andere testdoeleinden een segmentatie geschreven die puur op bewegingsdetectie werkt, en deze bleek beter te werken dan de huidige. Deze is dan ook geïmplementeerd in het prototype, ook omdat er geen tijd meer was om te kijken waarom de eerder gebruikte versie niet meer goed werkte. De veranderingen in de –nu dus niet meer gebruikte- segmentatie was onder andere het gebruik van een andere segmentatie op de plek van de lift. Wat wel gebruikt werd bij de oude segmentatie en dat nog wel wordt gebruikt, is posterization(functie posterize) waarbij er een stuk minder kleuren gebruikt worden(lagere kleurendiepte). Hierdoor ging veel ruis weg en werd de herkenning een stuk beter.

De compute functie is erg uitgebreid. In eerdere prototypes werd gebruik gemaakt van een geschiedenis met een constante grootte. Zo werden bijvoorbeeld alleen de laatste 100 frames geanalyseerd. Nu wordt gebruik gemaakt van ‘rondes’ (een ronde is de tijd sinds de lift open is totdat hij weer dicht gaat). Een andere nieuwe functionaliteiten zijn foutcorrectie (het programma herkent wanneer er verkeerde data opgenomen wordt) en het vaststellen van verkeer op basis van een afgeleide.

Nu worden eerst eigenschappen van objecten bepaald (om te kijken of ze mensen zijn), vervolgens bepaalt het programma of deze zich binnen of buiten de lift bevinden (door hun boundingbox met die van de lift te vergelijken), en fouten hierin (over verschillende frames) worden gecorrigeerd door de functie patchpat. Aan de hand van deze data kan met behulp van een afgeleide bepaald worden hoeveel mensen er in of uit de lift lopen. Een uitgebreidere uitleg hierover is te lezen in het hoofdstuk over implementatiekeuzes.

Soms levert de segmentatie bij 2 of meer dicht bij elkaar staande mensen 1 groot object op (ze zijn allebei ‘connected’ als het ware). Dit zou eerst herkend worden als een enkel persoon, maar in dit prototype lost de functie headHunter dit op. Als van een object de verhoudingen niet lijken te kloppen voor een enkel persoon wordt deze functie geactiveerd. Die kijkt in het gebied van het object of er losse hoofden te vinden zijn. Er wordt door middel van scanlines een patroon zoals bijvoorbeeld 0001110001110 als twee hoofden herkent (elk groepje van 1 is een hoofd, achtergrond is 0). Het is dan aannemelijk dat er in dit geval twee hoofden zijn, en dat dit object dus niet 1 maar 2 personen inhoudt. Hierop wordt de telling aangepast. Als de functie headHunter gebruikt wordt, is in het beeld rechtsboven dit object in het blauw gezien.

De in het PvE opgestelde eis dat in minimaal 75% van de gevallen het juiste aantal personen wordt herkend wordt gehaald, dus wat dat betreft is het programma goed gelukt. Toch zou het programma nog verbeterd kunnen worden op sommige punten. Zo zijn er uit de testvideo’s opvallende dingen

naar voor gekomen, zoals bijvoorbeeld het geval dat in een enorme chaos het programma het goede aantal mensen in en uitgaan telt, alleen als er bij een andere video simpel 1 persoon rustig de lift uit komt wordt deze niet goed verwerkt. Al met al is er waarschijnlijk door veel optimaliseren en veel testen een iets beter slagingspercentage uit te halen, maar het programma voldoet aan de eisen en werkt snel, en dat was de bedoeling.

8 Implementatie keuzes

De hele beeldverwerking waarop de herkenning van het systeem is gebaseerd bestaat uit verschillende stappen. Deze stappen zijn voornamelijk de onderdelen die in het functie blok schema staan beschreven. Per stap in de gebruikte “pipeline” zijn de keuzes voor verschillende oplossingen beschreven.

8.1 Normalisatie keuzes

Voor het normaliseren zijn er verschillende methodes geprobeerd. De eerste methodes bleken al snel niet correct te werken en zorgden niet voor verbeteringen in de kwaliteit van het programma of de frames.

8.1.1 Methodes

Methode 1

De eerste methode die geprobeerd is toe te passen, was het intensiteitsniveau van de frames normaliseren rond een bepaald gemiddelde. Voor het gemiddelde lag de waarde tussen de uiterste intensiteitswaarden voor de hand. Deze is voor het gemak vastgesteld op 125. Door een factor te berekenen uit het verschil van de twee gemiddelden, kunnen alle waarden in het frame wat verlaagd, of juist verhoogd worden. Het probleem met deze methode is dat wanneer er een object in beeld komt met bijvoorbeeld een relatief hoge intensiteit, de gemiddelde intensiteit van het frame ook omhooggaat. Dit zorgt er dan weer voor dat deze methode alle intensiteitswaarden naar beneden haalt, dus ook die van de achtergrond. Dit levert totaal verschillende frames op en is dus geen normalisatie, vooral omdat de achtergrond juist hetzelfde moet blijven om van het huidige frame af te kunnen trekken.

Methode 2

Het DIPImage archief heeft een methode om een beeld gelijk te maken op basis van het histogram. Dit leek goed te werken in de single frame tests. Maar het toepassen hiervan bleek een tijdrovende manier te zijn. Zo erg dat het systeem niet meer real-time zou kunnen werken. Daarnaast bleek het ook niet goed te werken door een constante verandering van lichtintensiteit wanneer er een object in beeld komt. Dit is hetzelfde probleem als methode 1 waarvan de oorzaak bij methode 3 naar voren zal komen.

Methode 3

Daarna werd besloten een frame van de achtergrond op te slaan in de eerste kalibratie stap. Dit gaf de mogelijkheid om zowel een genormaliseerd frame als een origineel frame op te slaan. Door elk binnenkomend frame te vergelijken met het originele kalibratie frame kan de achtergrond worden bepaald, wat wordt gebruikt als een mask op het inkomende frame en het genormaliseerde kalibratie frame. Van beide frames is dan dus alleen de waarschijnlijke achtergrond gekozen en hierop wordt methode 1 toegepast.

Dit zou dus een veranderingsfactor van de gemiddelde intensiteit van de twee achtergrond frames maken en deze toepassen op het inkomende frame. Deze methode werkte niet goed, omdat bleek dat de webcam de witbalans en lichtintensiteit van het beeld steeds aanpast. Hierdoor wordt er dus nog steeds geen correcte normalisatie uitgevoerd omdat de inkomende beelden grote verschillen

kunnen tonen wanneer er een object in beeld komt. Het uitzetten van de automatische witbalans en lichtintensiteit leverde een duidelijk beter resultaat.

8.1.2 Uiteindelijke keuze

Uit de bovenstaande methodes bleek vooral dat de webcam zijn lichtintensiteit en witbalans niet automatisch mag veranderen. Het is uitermate lastig om hier steeds op te compenseren. Door te stellen dat deze instellingen niet mogen veranderen tijdens het draaien van het systeem is er al zeer weinig verandering van lichtintensiteit tussen de frames te zien.

Met de genoemde instellingen van de webcam blijken alle beschreven normalisatie methodes weinig toe te voegen aan de werking van het systeem. Hierdoor is gekozen om de normalisatie weg te laten aangezien er toch geen significante meerwaarde is. Er is dus uiteindelijk gekozen geen normalisatie te doen omdat het niets toevoegt en het niet nuttig is om daaraan dan processortijd te besteden.

Omdat normalisatie niet meer noodzakelijk is door de bovenstaande argumenten is gekozen geen nieuwe methodes hiervoor te gaan zoeken en te focussen op de verdere processing van de frames.

Herziende methode

Omdat de segmentatie de gehele tijd nog zeer lastig bleek te zijn, is er een manier gevonden om dit iets beter te laten werken. Deze manier is ondergebracht onder normalisatie omdat het de frames in z'n geheel voorbewerkt. De methode die er nu wordt gebruikt en dus wel nut heeft is posterization. Dit verdeelt de kleuren in de frames in minder verschillende kleuren. Per kleur laag kan apart worden bepaald hoe groot de stappen tussen de waardes moeten worden. Er ontstaan grotere overgangen en ook grotere egale vlakken. Een bijkomend voordeel hiervan is dat de ruis al direct verminderd word.

8.2 Liftsegmentatie keuzes

Voor het programma is het van belang om te weten waar de lift zich bevindt, om vast te kunnen stellen of er mensen in- of uitgelopen zijn.

Er zijn meerdere manieren om de plaats van de lift vast te stellen. Eén manier is op basis van kleur en een andere manier is op basis van kleurovergangen. Er zal naar beide manieren gekeken worden en uiteindelijk wordt hier een keuze tussen gemaakt.



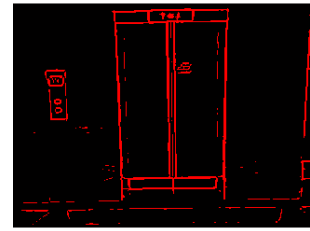
Figuur 1

In prototype 4 is de lift puur op basis van kleuren herkend. Dit is te zien in *Figuur 2*. Het idee achter deze segmentatie is dat de rood-, groen- en blauwwaarde zich elk op een bepaald interval moeten bevinden. De roodwaarde zal hoog zijn, en de groen- en blauwwaarde een stuk lager. Aan de hand hiervan kan bepaald worden waar de liftdeuren zich bevinden. Deze aanpak heeft als voordeel dat het niet veel rekentijd kost, maar waarschijnlijk zal deze aanpak minder goed werken als het beeld donkerder of juist lichter wordt.



Figuur 2

Een andere manier is zoals eerder gezegd op basis kleurovergangen de lift zoeken. Deze aanpak kijkt naar de overgangen van de drie kleurlagen (edge detection). Dit zal goed werken bij het beeld in *Figuur 1*, want er zijn grote sprongen in kleuren (rood, zwart, wit, grijs). In *Figuur 3* zijn deze sprongen te zien na drempeling.



Figuur 3

Deze aanpak heeft als voordeel dat de lijnen van de lift erg goed zichtbaar zijn en bij iets minder kleurverschil zal dit ook nog werken, maar voor deze edge-detection is meer rekenwerk nodig dan bij de eerste aanpak.

Als er van deze verschillende resultaten de boundingbox (min, max) bepalen, komen we uit op de volgende resultaten:



8.2.1 Uiteindelijke keuze

De eerste aanpak is snel en effectief, maar bij deze aanpak zullen de deuren iets te klein worden herkend door de metalen randen onderaan de deuren. De tweede manier kost meer rekenkracht, maar geeft betere en nauwkeurigere resultaten.

Hoewel er niet veel verschil in resultaat zit tussen beide aanpakken, is gekozen voor de tweede manier. Dit omdat uit de praktijk bleek dat een iets te ruime marges beter werkt. Bij lichte beweging van de camera zal de boundingbox met de tweede aanpak nog steeds om de lift zitten; bij de eerste aanpak ligt de box dan snel naast de deuren. De iets langere rekentijd maakt relatief gezien geen verschil, omdat alleen bij het opstarten liftSegment uitgevoerd wordt.

8.3 Persoonssegmentatie keuzes

De segmentatie is een van de belangrijke delen van het gehele proces. Wanneer de segmentatie niet correct werkt, kan het al snel verkeerde data voor de vervolg processen opleveren. Het is daarom van belang dat altijd de juiste data voor de vervolg processen wordt geleverd. Aan het eind zal blijken dat door de gekozen manier van tellen van personen, de segmentatie veel minder belangrijk blijkt te zijn.

8.3.1 Methodes

Methode 1

Als eerste is er geprobeerd het inkomende frame van het gekalibreerde frame af te trekken en het resultaat absoluut te maken. Dit geeft de verandering van intensiteit op de verschillende kleur lagen. Wanneer er niets in beeld is levert dit een nagenoeg zwart beeld op omdat er geen verschil is tussen de frames. De kleine verschillen die er te zien zijn is dan ook ruis en moet eruit gefilterd worden. Hiervoor wordt er een threshold toegepast, die rond de 25 is vastgesteld. Dit filtert de ruis eruit, maar laat nog wel de grotere veranderingen zien, de personen die in beeld komen dus.

Deze methode heeft twee grote nadelen. Ten eerste worden (delen van) kledingstukken met een laag contrast met de achtergrond (hier vooral grijs en rood) niet gezien als verandering omdat het contrast te laag is. Deze delen worden dan gezien als geen verandering of ruis. Als tweede geeft deze methode een grote verandering weer als de lift opengaat. Hierdoor is het niet meer mogelijk om veranderingen, die wordt veroorzaakt door personen die voor de lift staan of erin/uit lopen, waar te nemen.

Methode 2

Door de nadelen van methode 1 moest er iets toegevoegd worden. Deze methode maakt gebruik van het resultaat van methode 1 en combineert dit met de verandering tussen het huidige en vorige frame. Dit combineert dus de verandering met de achtergrond met de beweging in de video stream. Door beide binaire beelden met elkaar te vermenigvuldigen, komt er een beeld uit dat sowieso alleen maar veranderingen bevat.

Uiteraard worden er nog gepaste dilaties en erosies uitgevoerd voor en na het combineren van beide beelden. Dit omdat onder andere het binaire beeld van de bewegingen kleiner is dan die van de veranderingen met de achtergrond. De binaire objecten van de bewegingen worden groter gemaakt zodat er uiteindelijk een resultaat uitkomt dat meer de contouren van een persoon heeft.

Nadeel van de methode is dat wanneer een persoon in beeld niet beweegt, hij ook niet wordt herkend. Aangezien het systeem personen die de lift in of uit lopen moet herkennen en niet personen die stilstaan, is het dus nog een onderwerp van discussie of dit nadeel een probleem is. Wat er al wel gesteld kan worden is dat wanneer een deel van de persoon niet beweegt, er ook maar een deel van de persoon wordt herkend. Het herkennen van halve personen doet de vervolg stappen in het proces geen goed, daarom is deze methode niet geheel geschikt.

Methode 3

Een vlugge methode die tussendoor een mogelijkheid zou kunnen zijn was kijken naar het verschil tussen 2 beelden, waarvan de een verticaal verschoven was. Het huidige beeld wordt dan een aantal pixels naar beneden geschoven, waarna wordt gekeken naar kleine verschillen tussen het verschoven en niet verschoven beeld.

Bij deze specifieke segmentatie zou een specifieke herkenning en telling van personen moeten zitten. Deze methode zou de objecten/mensen moeten herkennen die verticaal in het beeld lopen. Maar met deze methode was niet rekenen gehouden met egale oppervlakten als de muur en de liftdeur die dan ook als bewegend zouden zien omdat een groot deel van dit oppervlakte weinig verschilt als het wordt bewogen. Deze methode is dus niet nuttig.

Methode 4

Methode 3 is uitgebreid met een dilatie met een lang verticaal structurerend object. We willen personen herkennen, ook die naast elkaar staan. Een dilatie teveel in de horizontale richting zal personen met elkaar combineren, wat natuurlijk niet de bedoeling is. Daarom hebben we er voor gekozen het voornamelijk verticaal te dileren. Dit geeft al een stuk beter resultaat dan eerst, al zitten er nog wel gaten in de objecten, voornamelijk wanneer er sprake is van een egaal oppervlakte.

Op het binaire beeld van de beweging worden nu een groot aantal dilaties losgelaten. Omdat dit wordt gecombineerd met het binaire beeld van waar de achtergrond veranderd is, levert het een redelijk grote vlek op in de lift. Dit komt omdat bij een open lift de gehele lift wordt gezien als veranderd en is dus opgenomen in het binaire beeld. Buiten de lift werkt het wel een stuk beter.

De vraag is nu wat er precies nodig is om te zien of er een persoon in of uit de lift loopt, want een betere segmentatie is lastig te bereiken.

Methode 5

Doordat een ander deel van het systeem nu een waarde uitgeeft over de status van de lift, kan de segmentatie in de lift worden aangepast wanneer de lift open is. Als de lift open is zal het binaire beeld in de lift alleen bestaan uit het binaire beeld van de beweging en niet de combinatie met het verschil met de achtergrond.

Daarnaast zijn door een vernieuwde normalisatie de thresholds in de segmentatie sterk aangepast. Dit omdat door de zogenaamde posterization methode de kleur waardes in grotere stappen dan 1 worden opgedeeld. Er zijn in totaal minder verschillende kleuren aanwezig in de frames. Omdat er grotere stappen worden genomen bij de kleurwaardes, moeten de thresholds ook flink omhoog om verschil te kunnen vinden.

Doordat de opvlakten nu egaler worden en de ruis er uit gefilterd is, levert dit nu zowel buiten als binnen de lift een beter resultaat op. De randen tussen de verschillende egale oppervlakten zijn weliswaar groter, maar dat wordt ook goed gefilterd door de vergrote thresholds.

Probleem - schaduw

Schaduw wordt bij de segmentatie ook mee genomen. Dit omdat een schaduw ook een verschil met de achtergrond is en net zoals een persoon beweegt. De overweging is om dit weg te filteren. Methodes op het internet laten zien dat zogenaamde “shadow removal” best lastig is en niet vlekkeloos werkt. Het lijkt dus niet rendabel en nuttig om schaduwen te detecteren/verwijderen aangezien dit hoogstwaarschijnlijk alsnog een frame oplevert met een groot verschil op de plek van de schaduw in vergelijking met het kalibratie plaatje¹.

8.3.2 Uiteindelijke keuze

In/na de vakantie is de code voor het daadwerkelijk tellen van personen pas echt gemaakt. Deze methode is gemaakt door een segmentatie te gebruiken die puur gebaseerd is op beweging in het beeld en dus geen gebruik meer maakt van de achtergrond. We hebben deze herkenningmethode getest met de segmentatie uit methode 5 en dit leek in eerste instantie even goed te werken. Na iets verder kijken bleek het opeens met de nieuwere versie van het systeem een gekke segmentatie op te leveren binnen de lift.

Aangezien het eind van het project in zicht is en de code wel een beetje af moest zijn, was er geen tijd om nog uit te vinden waarom de segmentatie uit methode 5 opeens niet meer goed werkt, wat hij in eerdere versies nog wel goed deed.

¹ Gebruikte documentatie shadow removal: <http://ijest.info/docs/IJEST10-02-09-93.pdf>

Doordat de zeer simpele segmentatie door middel van beweging alleen een goed resultaat oplevert bij het tellen van personen, is ervoor gekozen deze segmentatie te gebruiken en de andere segmentaties (methode 5) opzij te zetten.

8.4 Keuze voor het niet gebruiken van skeletons

Voor het herkennen van losse personen is gekeken naar het gebruik van de skeleton functie, om precies te zijn de functie 'skel' uit de bwmorph functie van matlab. Dit zou toegepast kunnen worden om personen te kunnen tellen. Dit is echter niet gedaan vanwege de enorme rekenkracht, oftewel tijd, die dit kostte. Op het moment van maken lag de gemiddelde verwerkingstijd in het programma rond de 0.40 seconden per frame. Met de skeleton functie geïmplementeerd steeg dit tot 4 seconden met niets in beeld(er werd ruis verwerkt) en tot boven de 5 seconden voor personen of andere objecten in beeld. Een andere methode die tegelijkertijd door een ander teamlid werd ontwikkeld, het aan de hand van bepaalde measurements identificeren en tellen of iets een persoon was, kostte echter maar 0.10 tot 0.20 seconden.

Aan de andere kant werkte de skeleton functie voor herkenning wel erg goed. Er kon namelijk door branchpoints, uiteindes en groottes van skeleton's heel goed bepaald worden of het object in beeld een persoon was, maar ook veel beter hoe veel personen dan in beeld waren. Daarom is er toch nog geprobeerd de skeleton functie te versnellen. Dit is onder andere gedaan door te spelen met het laatste argument uit de bwmorph functie: n. n is hierbij het aantal keren dat de functie wordt uitgevoerd. In eerste instantie stond deze op 'Inf', wat inhoudt dat de functie wordt uitgevoerd totdat het beeld niet meer veranderd. Door n een lage waarde te geven wordt de functie significant sneller, maar het resultaat ook slechter. Door wat te spelen met de waarde van n kon de rekentijd verkort worden naar rond de 1,5 seconden, maar het resultaat was eigenlijk te slecht, en het duurde nog steeds veel te lang.

Ook is er gekeken naar vergelijkbare functies uit bwmorph zoals 'thin', 'branchpoints' en 'shrink'. De laatste bleek het snelste en beste te werken en leverde ook goed resultaat, maar ook deze functie deed er rond de 2 seconden over waardoor uiteindelijk andere methodes veel beter bleken te zijn.

8.5 Liftstatus keuzes







Voor het programma is het handig om te weten in welke staat de lift zich bevindt. Er hoeft bijvoorbeeld niet gekeken te worden of iemand de lift in- of uitloopt als de lift dicht is. Kijken of er mensen de lift in zijn gegaan als de lift net open gaat, is ook niet van belang.

De lift zal in vier verschillende stadia verdeeld worden: 'open', 'closing', 'closed' en 'opening'.

Het vaststellen in welke staat de lift zich bevindt, zal gebeuren met een stukje van de bovenkant van de lift. Aan de hand van de boundingbox (zie *segmentLift*) zal dit stukje bepaald worden, zoals in de volgende figuren te zien is. De hoogte van dit stukje is zo'n 25% van de gedetecteerde liftbreedte.



Zoals te zien in het linker beeld: als de deur dicht is, is er veel rood in beeld (roodwaarde hoog, groen- en blauwwaarde laag). Als de deur open is, is het beeld geel/wit (rood-, groen- en blauwwaarde hoog). Het grote verschil tussen beide beelden zijn dus de groen- en blauwwaarde. Dat is ook uit de volgende beelden te halen:

	Dicht	Open
Rood		
Groen		
Blauw		

De grootste sprong is te zien bij de groene waarden, er is dan ook voor gekozen om deze kleurwaarden te gebruiken bij het indelen van de stadia. Bij dichte deuren ligt de groenwaarde tussen de 20 en 45; bij open deuren ligt deze waarde tussen 240 en 255. Een drempeling van <50 zal bijvoorbeeld goed werken, net zoals >230 . Er is voor de eerste optie gekozen, omdat er dan objecten in beeld zijn als ook de deuren in beeld zijn.

Met de gekozen drempeling en een erosie komen er bij een gesloten- en open deur de volgende beelden uit:



Door oppervlakte-metingen kunnen hier al twee stadia uit afgeleid worden:

- 'closed' als er twee grote vlakken gemeten worden
- 'open' als er geen twee vlakken gemeten worden

'Opening' en 'closing'

Het wordt iets lastiger om vast te stellen of de deuren aan het openen of aan het sluiten zijn. Aan de hand van alleen het onderstaande beeld kan dit niet worden bepaald.



Om dit wel vast te kunnen stellen, ligt het voor de hand om de grootte van de vlakken van het vorige frame erbij te pakken.



De deur moet wel open gaan ('opening') als de oppervlakte steeds kleiner wordt (de deuren zijn steeds minder in beeld). Omgekeerd geldt dan ook: de deuren moeten wel sluiten ('closing') als de oppervlakte steeds groter wordt (de deuren komen steeds meer in beeld).

Uiteraard moet het verschil in oppervlakte ook weer gedrempeld worden, aangezien het verschil tussen 850 en 851 niet direct betekent dat de deuren dicht gaan (ruis). Pas wanneer de verschillen tussen twee frames groot genoeg zijn, kan er met zekerheid worden gezegd dat de deuren openen of sluiten.

8.6 Persoonsdetectie keuzes

Voordat het programma werkelijke ingaand en/of uitgaand verkeer detecteert, het is noodzakelijk om het aantal personen in beeld te tellen. Dit wordt gedaan in een aantal stappen:

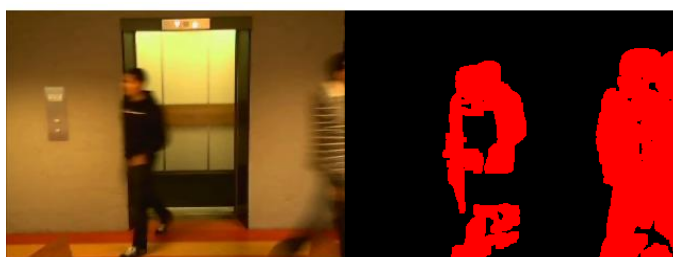
1. Filtering op basis van object-size.
2. Afkeuring van false-positives.
3. Verdere benadering.

Elke van de drie stappen wordt in dit hoofdstuk uitgelegd respectievelijk in paragraaf 8.6.1, 8.6.2 en 8.6.3.

8.6.1 Filtering op basis van object-size

Gegeven is een segmenteerd beeld, met vermoedelijk alle segmenten van de personen die zich in het oorspronkelijke frame bevinden (figuur 8.6.1 en 8.6.2).

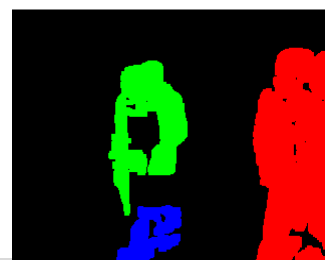
Het is mogelijk dat sommige segmenten geen personen zijn. Die moeten weggefilterd worden op basis van meerdere condities en tests. De eerste filtratie is ook het simpelst – op basis van object-grootte. Elk 8-connected object wordt gelabeld alleen als het aan een bepaalde grootte-threshold voldoet (figuur 8.6.3). Er is te zien dat in bepaalde gevallen er meer objecten gelabeld worden dan het aantal personen dat er werkelijk zijn. Dit probleem wordt via de volgende stappen opgelost.



Figuur 8.6.1 & 8.6.2 – Een voorbeeld frame en de persoon-segmenten daarvan.

8.6.2 Afkeuring van false-positives

Het blauwe object in figuur 8.6.3 is een voorbeeld van een false-positive. Er is al een object die de persoon het best beschrijft (groen). Blauw is ook gelabeld omdat het toevallig op de size-constraint voldoet.



Figuur 8.6.3 - Labeling

Er zijn meerdere methoden om false-positives te detecteren. Hier zijn een aantal condities die het programma in rekening neemt:

1. Object breedte/hoogte ratio is raar (bijvoorbeeld te dun om een persoon te zijn).
2. Het boundingbox is te klein.
3. De positie van het boundingbox is te hoog of laag.

In figuur 8.6.3 zal blauw aan conditie 3 niet voldoen, en dus zal ook geen deel nemen in opvolgende verwerkingen.

8.6.3 Verdere Benadering

Soms wordt een object als één persoon beschouwd, terwijl het meer dan één persoon is (figuur 8.6.4). Om dit op te lossen zijn er twee methoden bedacht, waarvan één niet geschikt bleek te zijn. De twee methoden zijn:

1. Custom opening en body-telling.
2. Hoofden tellen.



Figuur 8.6.4 – Twee personen in hetzelfde object.

Custom Opening en Body-Telling

Deze methode gebruikt een custom structurerend-element die heel dun en heel hoog is. Hiermee wordt een opening operatie gevoerd die verschillende personen uit elkaar trekt. Als laatste wordt het beeld opnieuw gelabeld en het aantal objecten geteld.

Deze methode werkt redelijk goed maar bleek heel traag te zijn, omdat er zo veel operaties (opening, labeling, measurements) nodig zijn. Het was niet praktisch om die te gebruiken.

Hoofden Tellen

Deze methode neemt het bovenste deel van het object, en probeert daar hoofden te tellen. Dit wordt gedaan door horizontale scanlines te comprimeren tot een matrix die een soort samenvatting van elk scanline representeert. Bijvoorbeeld gegeven het volgende scanline:

[0 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0]

Zo'n samenvatting zou zijn:

[0 1 0 1 0 1 0]

In het kort, het algoritme verkort lange patterns tot één concreet getal. De som van deze “samenvatting” is een goede schatting van het aantal hoofden aanwezig op die scanline.

Als we dit op elke scanline toepassen en een gemiddelde van nemen een goede benadering van het aantal mensen is bereikt. Deze methode bleek heel goed te werk is ook tenminste 6x zo snel dan het alternatief (custom opening). Daarom wordt deze methode ook gebruikt in de definitieve versie van het programma.

8.7 Persoonstelling keuzes

Het vaststellen van ingaand en uitgaand verkeer lijkt vrij simpel. Dat is echter helemaal niet zo. Dat komt doordat functies zoals segmentatie en mens-detectie soms foute resultaten opleveren (en dat kan geen systeem vermijden). Het ontstaat wanneer er naar de verzamelde data gekeken wordt en het is niet duidelijk welke metingen goed zijn en welke foutief.

Het is dus noodzakelijk om eerst die fouten proberen te vinden en te verbeteren, en pas daarna de data verder te analyseren. In paragraaf 8.7.1 wordt uitgelegd hoe data verzameld wordt, in 8.7.2 hoe fouten geïsoleerd worden en verbeterd. Tot slot wordt in paragraaf 8.7.3 de berekening van ingaand en uitgaand verkeer besproken.

8.7.1 Verzamelen van Data

Data wordt verzameld in 'rounds'. Een round begint wanneer de liftdeuren open zijn en eindigt wanneer ze beginnen dicht te gaan. Het heeft geen zin om buiten die interval te werken omdat mensen toch de lift niet in- of uitgaan wanneer de deuren gesloten zijn.

Verder wordt data per frame verzameld. Dat houdt in dat per frame de volgende operaties worden gevoerd:

1. Functies die in de persoonsdetectie keuzes besproken zijn worden geroepen om objecten te associëren met het juiste aantal mensen die ze bevatten.
2. Objecten ondergaan een paar tests. Die tests checken of ze binnen of buiten de lift liggen.
3. Het totaal aantal objecten binnen en buiten de lift wordt geteld en opgeslagen.

Het is vrij simpel om te check of een object binnen of buiten de lift ligt: Als het boundingbox van het object helemaal binnen die van de lift past, dan wordt het meegeteld als 'in', zo nee dan als 'uit'. Het aantal mensen die het object bevat worden opgeteld bij de juiste teller (de in- of uit-teller). De twee tellers worden dan opgeslagen in een variable genaamd *history*.

8.7.2 Error-detectie en oplossing

Het variable *history* bevat nu (aan het eind van een ronde) een matrix van per-frame-data. Laten we de lijst van waarden die per-frame de 'in'-teller representeren de *inStream* noemen. En de lijst van waarden die per-frame de 'uit'-teller representeren de *outStream*.

De *inStream* en *outStream* kunnen fouten bevatten. Er volgt een voorbeeld: Neem aan dat het volgende de 'ideale', foutloze beschrijving is van *history*:

Frame #	1	2	3	4	5	6	7	8	9	10
In	2	2	2	2	2	2	1	1	1	1
out	0	0	0	0	0	0	1	1	1	1

Helaas in het praktijk ziet het meer zo uit:

Frame #	1	2	3	4	5	6	7	8	9	10
In	2	2	0	2	2	2	1	1	1	1
out	0	0	0	0	0	0	1	1	1	1

De error ligt in frame 3, waar de segmentatie en/of labeling per ongeluk 0 opleverden. Deze fout is makkelijk te herkennen omdat het een onlogisch gat in de reeks vormt. Het is vanzelfsprekend wat de juiste waarde moest zijn.

Dit is hoe het algoritme werkt: hij scant de *history* en probeert de bovengenoemde gaten te vinden en te verbeteren. Gaten worden gedetecteerd door kettingen van de eenzelfde getal te vinden die omringd zijn door langere kettingen van een andere getal(len).

Het gat wordt vervolgens vervangen door een van zijn burens, of een gemiddelde daarvan.

8.7.3 Verkeersberekening

De *inStream* en *outStream* zijn de fout-detectie ondergaan en geven nu een mooie benadering van de werkelijkheid. Het is nu tijd om daaruit de ingaand en uitgaand verkeer af te leiden.

Dit wordt gedaan door afgeleiden. Laten we de afgeleide van de *inStream* 'din' noemen en de afgeleide van de *outStream* 'dout'. We gaan door met het voorbeeld van net:

Frame #	1	2	3	4	5	6	7	8	9	10
In	2	2	2	2	2	2	1	1	1	1
Out	0	0	0	0	0	0	1	1	1	1
Din	0	0	0	0	0	0	-1	0	0	0
Dout	0	0	0	0	0	0	1	0	0	0

Opmerkelijk is het feit dat wanneer een persoon uit de lift stapt (frame 6-7) vinden we een complement-duo in *din* en *dout*.

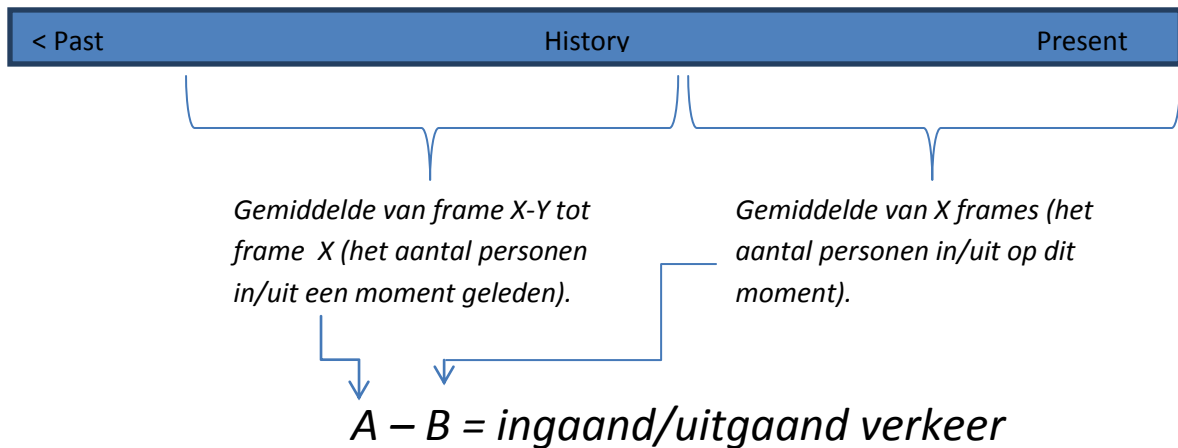
Dus op die manier worden transities gedetecteerd. *Din* wordt doorgelopen totdat een non-zero getal opgenomen wordt. Vervolgens wordt de respectievelijke omgeving in *dout* gecheckt. Als daar een complement te vinden is, dan vond een transitie plaats (uitgaand of ingaand).

Alle gedetecteerde transities worden bij elkaar opgeteld en naar de GUI gestuurd. Daar worden ze zichtbaar voor de gebruiker.

8.7.4 Alternatieve persoonstelling

In prototype 4-5 was een andere manier gebruikt voor het vaststellen van het verkeer. In plaats van data op basis van rondes te verzamelen, is er een poging gedaan om dat constant te doen, en ook constant daaruit het verkeer af te leiden.

Dat houdt in dat de variabele *history* een constante grootte had, bijvoorbeeld honderd frames. Na elke iteratie van de main-loop werden de X laatste frames genomen uit de *history*. Vervolgens was het gemiddelde van die X frames (van *inStream* en *outStream*) gebruikt als benadering voor het werkelijk aantal mensen die in of uit de lift staan. Deze getallen worden dan vergeleken met de gemiddelden van X-Y tot X frames 'geleden'. Uit het verschil kan ingaand of uitgaand verkeer gevonden zijn.



Deze methode werd uiteindelijk niet gebruikt, omdat het onnauwkeurig bleek te zijn wanneer er grote groepen van mensen in beeld waren. Hoe meer mensen in beeld staan – hoe groter de kans dat segmentatie en mens-detectie foutieve data opleveren. Let op dat het detecteren van deze ‘gaten’ (zie 8.6) in de reeks van *inStream* en *outStream* werkt alleen wanneer alle frames uit die ronde beschikbaar zijn. Bijvoorbeeld in het volgende reeks:

Frame #	1	2	3	4	5	6	7	8	9	10
In	2	2	0	2	2	2	1	1	1	1

Het is duidelijk dat frame 3 een gat vormt alleen wanneer frames 4-6 ook zichtbaar zijn. Stel nou voor dat het bovengenoemde algoritme net bij frame 3 is gekomen:

Frame #	1	2	3
In	2	2	0

Je zou kunnen denken dat er gewoon twee mensen uit de lift gestapt zijn , terwijl dat niet het geval is.

9 Evaluatie en testen systeem

Het systeem is met het laatste prototype (prototype 6) getest op de werking. Per filmpje zijn de resultaten te vinden in hoofdstuk 12.2 in Tabel 1. Elk filmpje bevat 1 of een paar van de genoemde tests in het testplan. Alle punten in het testplan zijn aan bod gekomen.

Er is op de volgende manier gemeten: Per filmpje zijn er 4 getallen, in en out, en daarvan de waarden die het systeem opgaf en wat het werkelijk was. Mochten in en out overeenkomen tussen de werkelijke en de gemeten waarde is er een score gegeven van 100%. Komt één van de twee getallen overeen, dan is er een score van 50%. Als ze allebei niet overeenkomen is er een score van 0%. Er zijn ook rijen zonder resultaat. Hierbij voldeed het filmpje niet aan het programma van eisen.

Uit deze tests blijkt dat in 77% van de gevallen het systeem het correcte resultaat geeft. Dit aantal voldoet aan de gestelde eisen. Waarschijnlijk ligt deze waarde nog iets hoger, omdat er na het uitvoeren van de tests nog een aantal kleine verbeteringen is toegevoegd aan het prototype.

De herkenning van mensen kan het programma erg goed. In de meeste gevallen wordt een persoon daadwerkelijk als een persoon herkent, en meerdere personen die tegen elkaar aan staan of elkaar overlappen worden ook als meerdere personen herkent. De verwerking van die personen gaat vervolgens redelijk. Helaas wordt niet altijd iedereen op de goede plek herkent. Dat wil zeggen, als een persoon voor de lift staat heeft het systeem soms moeite te bepalen of deze nu binnen of buiten de lift is. Over het algemeen gaat dit echter wel goed.

Het programma toont de werking en de statistieken duidelijk, crasht niet, en geeft ook geen foutmeldingen. Het programma werkt real time en voldoet aan alle eisen. De conclusie is dus dat het programma goed werkt.

10 Conclusie en aanbeveling

In dit hoofdstuk zal er antwoord op de probleemstelling gegeven worden en worden aanbevelingen gedaan.

De probleemstelling was als volgt: “Is het mogelijk om te detecteren hoe vaak en door hoeveel personen één personenlift in EWI gebruikt wordt tijdens een bepaald interval op één verdieping?”

Het antwoord hier op is duidelijk: ja, dat is zeker mogelijk. Het programma telt het aantal mensen vrij nauwkeurig. Het programma heeft, zoals verwacht, de meeste moeite met grote aantallen mensen. Dit komt echter meer doordat het beeld dan zo vol is, dat mensen elkaar veel overlappen. Het grootste probleem met een programma als dit is momenteel verwerkingstijd. Zelfs met een erg snelle computer is het moeilijk het programma realtime te laten werken als alle door ons gewenste functies erin gezeten zouden hebben. De herkenning zou beter kunnen door bijvoorbeeld tracking toe te passen, en andere segmentatiemethodes te gebruiken, alleen dan kan je pas ver achteraf na analyse van de beelden weten wat de statistieken zijn en dat was niet de bedoeling.

Uit het programma zouden ook nog opvallende resultaten verbeterd kunnen worden. Zo kan het bij de huidige versie voorkomen dat als 6 mensen een lift uit komen en in een complete chaos rondlopen het programma het juiste aantal personen herkent, maar als later een enkel persoon in alle rust uit de lift komt deze niet herkend wordt.

Ook is er momenteel geen mogelijkheid (meer) om door te spoelen als er van de video-input functionaliteit gebruik gemaakt wordt. Voor snelle analyse zou dat wel handig zijn.

Al met al voldoet het product aan de gestelde eisen en doet wat het moet doen. Kleine verbeteringen zijn mogelijk, maar binnen de huidige specificaties is het een erg functioneel programma.

11 Literatuurlijst

Tracking groups of people – Stephan J. McKenna – 2000:

http://www.computing.dundee.ac.uk/staff/stephen/cviu2000_www.pdf

Get wavelength from image:

<http://www.mathworks.nl/matlabcentral/answers/14587>

“Toward Face Detection, Pose Estimation and Human Recognition from Hyperspectral Imagery”:

<http://isda.ncsa.illinois.edu/peter/publications/techreports/2004/TR-20041108-1.pdf>

Background removal and human detection methods, based on video changes, color combination and selection and more:

<http://www.montefiore.ulg.ac.be/~barnich/files/barnich10thesis.pdf>

Body part recognition:

<http://www.cs.dartmouth.edu/~cs104/BodyPartRecognition.pdf>

Model based recognition:

<http://www.cs.dartmouth.edu/~cs104/BodyPartRecognition.pdf>

“Shadow detection and removal in colour images using matlab”:

<http://ijest.info/docs/IJEST10-02-09-93.pdf>

“Tracking groups of people”:

http://www.computing.dundee.ac.uk/staff/stephen/cviu2000_www.pdf

“Counting people in crowds”:

http://nguyendangbinh.org/Proceedings/ICCV/2003/iccv03/0122_yang.pdf

“Histograms of oriented gradients for human detection”:

http://hal.archives-ouvertes.fr/docs/00/54/85/12/PDF/hog_cvpr2005.pdf

“Fast and Effective Features for Recognizing Recurring Video Clips in Very Large Databases”:

<http://mmc36.informatik.uni-augsburg.de/mediawiki/images/f/f3/VMDL97.pdf>

“Bilayer segmentation of live video”:

http://research.microsoft.com/en-us/um/people/ablake/papers/ablake/criminisi_cvpr06.pdf

“Tracking and counting human in visual surveillance system”:

<http://www.slideshare.net/iaeme/tracking-and-counting-human-in-visual-surveillance-system-15665149>

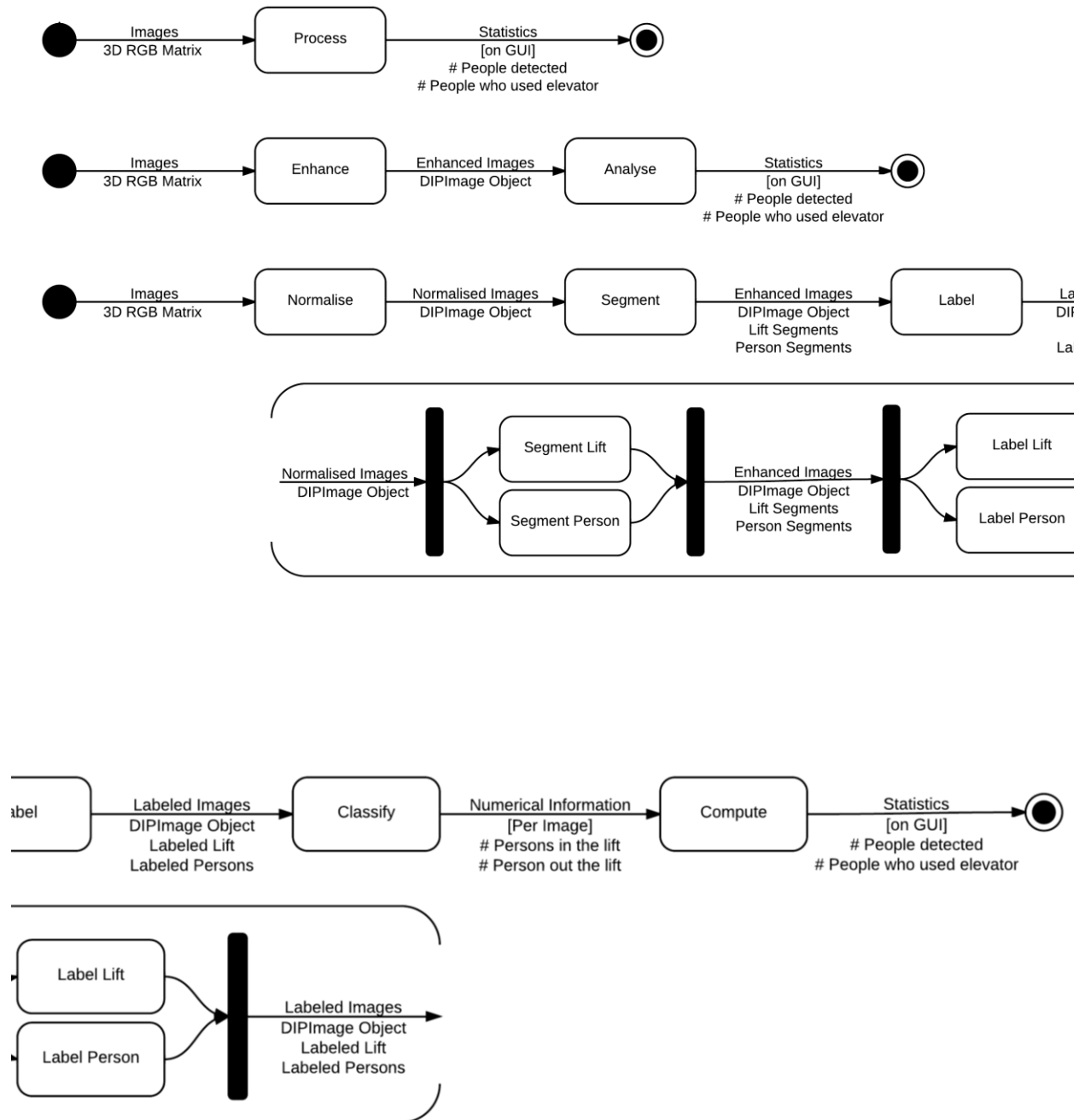
“Moving object detection using matlab”:

<http://www.ijert.org/browse/august-2012-edition?download=715%3Amoving-object-detection-using-matlab&start=100>

12 Figuren en tabellen

12.1 Figuur 1 - Functieblokschema

Dit figuur is gesplitst in twee delen aangezien het anders niet leesbaar op de pagina kon.



12.2 Tabel 1 - Testresultaten

Tabel met de resultaten van de tests. Gebruikt in hoofdstuk 9. De rijen zonder test resultaat zijn video's die niet aan het programma van eisen voldoen.

Filmpje	Programma		Werkelijk		Score	Opmerkingen
	in	uit	in	uit		
1	0	0	0	0	100	
2	1	1	1	1	100	
3	1	2	1	4	50	
4	0	0	0	0	100	
5						
6	2	1	2	1	100	
7						
8	2	1	2	1	100	
9	0	1	0	2	50	
10	2	0	2	0	100	
11	0	1	0	1	100	
12	2	2	2	1	50	
13	2	1	2	1	100	
14	2	2	2	1	50	
15						
16	1	1	2	1	50	
17	2	2	4	1	0	
18	4	2	3	1	0	
19	1	1	1	1	100	
20						
21	1	2	1	2	100	
22	2	0	2	1	50	
23	4	2	2	1	0	
24						
25	1	2	1	1	50	
26	2	1	2	1	100	
27						
28	0	0	0	0	100	
29	1	0	1	0	100	
30						
31						
32	0	0	0	0	100	
33	0	1	0	1	100	
34	1	1	1	1	100	
35	0	2	0	2	100	
36						
37	4	0	4	0	100	

76,78571

13 Bijlagen

In de bijlagen kunnen de extra stukken gevonden worden.

13.1 Eigen eindbeoordelingen

Elk groepslid heeft een beoordeling geschreven over de groep en over zichzelf. Hierbij is ook inbegrepen welke bijdrage hij heeft gehad binnen de groep en het project.

13.1.1 Marnix de Graaf

Verslag

Ik heb me voornamelijk bezig gehouden met de prototypes, van begin tot eind. Veel ervan heb ik gemaakt, en andere aangepast en verbeterd. Daarbij heb ik meegewerkt met het verslag in het algemeen, het PvE, het functieblokschema en ook documentatie geschreven over het niet gebruiken van skeleton. Ik heb de inleiding geschreven, Evaluatie en testen van het systeem en een groot gedeelte van de conclusie.

Implementatie

Ik heb me in het begin meer bezig gehouden met code dan aan het eind. Wellicht aan het eind wat te weinig, omdat ik toen veel aan het verslag deed. Ik heb onder andere de segmentatie in het begin geschreven, de GUI gemaakt, veel toegevoegd aan DeWijzeWieken.m, auto-calibratie geschreven(hoewel dit er niet meer zin zit) en een deel van de toenmalige labeling. Ook heb ik tussendoor code opgeruimd en geoptimaliseerd.

Ik heb voornamelijk veel geholpen en uitgezocht voor andere teamleden. Zo heb ik veel met Tim samengewerkt voor bijvoorbeeld liftdetect en de segmentatie. Ik heb implementaties uitgeprobeerd voor Face recognition, tracking, het gebruik van skeleton voor goede herkenning, andere BWmorph functies en ik heb veel gespeeld met de Vision toolkit. Bij die laatste heb ik gekeken of dit beter werkte, maar dit bleek allemaal veel te traag te zijn. Ook heb ik nog veel getest en daaruit bugfixes gedaan.

Natuurlijk heb ik ook geholpen met brainstormen voor nieuwe ideeën, oplossingen en keuzes maken.

Beoordeling groep

Al met al ben ik blij met de groep. In het begin van het project gingen we niet zo snel, deden we bijna niets thuis en hebben we niet snel genoeg dingen bereikt. Na enkele weken sloeg deze houding vrij snel om en heeft iedereen er veel harder aan gewerkt. Met iedereen viel gelukkig goed samen te werken.

Daniel heeft al met al niet zoveel toegevoegd aan dit project. Hij was vaker afwezig en zorgde vervolgens niet dat hij bij kwam met de groep. Dat is erg jammer. Raoul vond ik in het begin erg stil, maar dat kwam later helemaal goed en verbaasde me met enorme stukken heel goed werkende code. Daar ben ik erg positief over. Tim is erg gezellig en een fijn persoon om mee samen te werken, en hij doet ook goed zijn best. Mirko heeft ons gelukkig vanaf het begin weerhouden om sommige keuzes te maken en heeft heel constructief meegedaan met vergaderingen.

Mijn eigen inzet had af en toe wat hoger gekund, maar ik heb wel altijd met iedereen meegedacht, veel toegevoegd aan vergaderingen en ook de sfeer goed omhoog gehouden.

13.1.2 Mirko Dunnewind

Verslag

Voor het verslag heb ik uiteraard de documentatie en keuze begeleiding geschreven bij de delen waarvan ik de implementatie heb gedaan. Daarnaast ben ik bezig geweest met het in elkaar zetten van het tussenverslag en het uiteindelijke verslag. Dit hield in (vooral voor het eindverslag) dat ik de opmaak over gelijk heb gesteld, fouten verbeterd, structuur heb aangebracht en kleine teksten heb geschreven. Ook heb ik het testplan en de probleemstelling geschreven

Implementatie

Voor de implementatie ben ik zelf voornamelijk bezig geweest met de normalisatie en segmentatie van personen. Uiteindelijk is van beide niet veel meer in de code terug te vinden. Dit aangezien een normalisatie niet essentieel nodig bleek te zijn met de juiste webcam instellingen en dat een beetje goede normalisatie te tijdrovend zou zijn. Daarnaast leek tot het einde van het project de segmentatie essentieel voor de rest van het systeem. Daarom ben ik (ook op aandringen van mijn groep) hiermee steeds verder gegaan. Zo zijn er nieuwe methodes uitgedacht en geprobeerd. Uiteindelijk viel de segmentatie niet veel meer te verbeteren, maar toen bleek pas dat de rest van het systeem helemaal niet zo'n goede segmentatie nodig had. Naast deze twee delen heb ik vooral in het begin de rest geholpen met problemen in de code en zelf een paar kleine delen geschreven, zoals een deel van het video laden systeem.

Beoordeling groep

In het begin van het project ging het nog wat traag binnen de groep. Iedereen leek zijn eigen dingen te hebben en zo kwam er niet zoveel van het project terecht in eerste instantie. Al snel bleek ook dat het herkennen van personen die in en uit de lift lopen best lastig is. Hierdoor zaten we zelf ook wel even met de handen in het haar, maar het kwam er uit eindelijk natuurlijk op neer dat we wel een werkend systeem neer moesten zetten met het huidige onderwerp.

In het begin moest iedereen nog een beetje aan elkaar wennen, maar al snel begreep iedereen elkaar beter en hebben we goede vergaderingen gehad. Als groep konden we zowel serieus als speels zijn. Zo zijn er ook veel grappen over tafel gegaan, waarvan ik persoonlijk denk dat die juist hebben bijgedragen aan de groep.

Van alle groepsleden was Tim de meest steady persoon. Hij had bijna altijd alles gedaan wat hij moest doen en leverde ook een beter dan gemiddelde inzet op alle gebieden. Daniel had altijd mooie ideeën maar het ontbrak hem bij de uitvoering, maar hij is dan uiteindelijk ook gestopt met het project. Marnix was misschien de meest vrolijke en wilde uit de groep, desalniettemin leverde hij altijd wel goede producten af, misschien een beetje laat. Daarnaast heeft Marnix wel de groep goed bij elkaar gehouden. Als laatste is Raoul de wat stillere. Aan het begin moest hij volgens mij een beetje wennen aan zijn groepje, maar na een paar weken was dat al weer helemaal weg en leverde hij perfecte stukken code aan, om maar wat te noemen.

13.1.3 Raoul Harel

Verslag

Aan het begin van het project probeerde ik (samen met anderen) in het algemeen brainstormen over het onderwerp. Ik heb meegedacht aan onze oorspronkelijke PvE en functieblokschema en daarop mijn feedback gegeven.

Hierna heb ik in elke prototype mijn deel ingevuld (wat ik voor die prototype geprogrammeerd heb). Bij prototype 3 was het ook mijn beurt om het in elkaar te zetten.

Na de vakantie heb ik voor het eindverslag uitgebreid documentatie geschreven van functies classify, compute en alle hulp-functies die daarbij horen. Ook heb ik een paar implementatie-keuzes toegelicht (in verband met onze mens-detectie methoden en een toelichting van alternatieven voor het functie compute).

Implementatie

Het eerst wat ik geïmplementeerd heb was een soort template van alle files en functies die we in de komende weken gaan invullen. Het eerste functie die ik helemaal zelf programmeerde was de eerste versie van liftDetect. Vervolgens moest ik de videoLoader van Daniel in het programma integreren opdat we met videos zouden kunnen testen.

Daarna kreeg ik de taak om functies classify en compute te implementeren. Elke prototype heb ik wat aan verbeterd, of een andere methode bedacht. Testen van mijn code was nog een beetje moeilijk omdat functies waarvan ik afhankelijk was nog niet klaar waren.

Tijdens de kerst vakantie is dat opgelost en testen was mogelijk. Voor het eerst kon ik precies zien welke methoden goed en welke slecht werkten. Aan het einde van de vakantie heb ik een redelijk werkende versie bereikt. Functies die ik hiervoor bewerkte waren: classify, headHunter, compute, patchpat (en meer die in het eindversie niet gebruikt worden).

In de laatste week heb ik alles wat we gedaan hebben in elkaar gezet en compute/classify verbeterd tot zover ik kon.

Anders

Af en toe heb ik anderen geholpen met hun implementaties en ik was ook altijd aanwezig wanneer we het system live testten.

Beoordeling Groepsleden

Aan het begin waren we als groep wat traag. We hebben heel veel discussies gehad maar deden weinig aan de implementatie. Het heel moeilijk te zijn een paar groepsleden te 'bereiken'. Hiermee bedoel ik dat sommigen de deadlines niet serieus genoeg namen of te weinig aan hun taak gedaan.

Net voor de vakantie zijn die problemen min of meer opgelost (op Daniel na natuurlijk). Gesprekken met de docenten leken het incentive te zijn om meer te gaan doen.

Al met al, ben ik wel blij met dit groepje. We hadden een moeilijke onderwerp (en waren 1 man te kort) en toch hebben we een werkend programma opgeleverd. Dat vind ik heel goed!

13.1.4 Tim Rensen

Beoordeling groep

De groep is na verloop van tijd naar elkaar gegroeid vind ik. De eerste paar bijeenkomsten moesten we erg wennen aan elkaar. Het schoot allemaal niet erg op, met deadlines werd erg laks omgegaan. Ik had het gevoel dat er een aantal mensen bij zaten die erg gemotiveerd waren en ook sommigen iets minder.

De samenwerking ging goed, hier was wel wat opstarttijd voor nodig: de eerste weken ging het minder soepel dan de laatste weken. Iedereen wist toen wat hij moest doen, dit kwam mede door de twee vergaderingen die we op een projectdag deden.

We hadden een redelijk gevarieerde groep, iedereen met zijn sterke en minder sterke kanten. Gelukkig sloot dit wel goed op elkaar aan, op elk gebied was er wel iemand goed. We hebben elkaar geholpen en uitleg gegeven bij de punten waar iemand niet goed in was.

De andere groepen hadden bijna allemaal zes leden, wij begonnen er al met ééntje minder. Daniël is deze periode met het project gestopt, dus we bleven maar met z'n vieren over. We hebben dit op moeten vangen, en dat is volgens mij aardig gelukt. We moesten hele duidelijke deadlines stellen en iedereen moest precies weten wat hij moest doen. Gelukkig hebben we met deze kleine groep nog een redelijk goed werkend programma kunnen afleveren.

Eigen bijdrage

Ik heb een grote bijdrage geleverd aan het project volgens mij, ik heb er erg veel tijd ingestoken. Ik hoorde dit ook van de andere groepsleden, daar was ik blij om. In het uiteindelijk programma zitten er twee duidelijke onderdelen van mij in: liftsegmentatie en liftStatus met bijbehorende documentaties. Het programma van eisen hebben we met zijn alle opgesteld; Daniël en ik hebben samen de definitieve PvE samengesteld en opgestuurd.

Ik heb veel samengewerkt met Marnix, wat erg goed liep. We hebben veel opgezocht over segmentatie en hier mee getest.

De laatste weken heb ik erg uitgebreid getest, waarbij ik bij de definitieve versie van het programma een tabel gemaakt heb met de scores. Verder heb ik geholpen met de prototypen werkend krijgen bij een deadline.

13.2 Logboek

Week 1

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	SVN instellen	<i>Version control with Subversion</i> door Ben Collins-Sussman, Brian W. Fitzpatrick & C. Michael Pilato. Lux(eenheid) op Wikipedia	Helpen met probleemstelling bepalen en PvE opstellen

Marnix	-	Geen	Helpen met probleemstelling, PvE, Functieblokschema, genotuleerd
Mirko	-		Helpen PvE en functieblokschema, standaard logboek
Raoul	-	Eenheid van licht opgezocht	Meegedacht aan probleemstelling, PvE, functieblokschema en testplan
Tim	-	Eenheid van licht opgezocht	Groot deel PvE opgeschreven

Week 2 (maandag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Kleurbepaling geïmplementeerd	howto google code hosting subversion tortoissvn Image Acquisition Toolbox Documentation DIPimage user manual	Feedback op PvE en probleemstelling geleverd.
Marnix	Gui gemaakt	Projecthandleiding	Voorwoord, probleemstelling aangepast, bijgedragen aan PvE
Mirko	-	<ul style="list-style-type: none">- Wavelength from image (misschien handig voor omzetten in matlab)- Face Detection and Human Recognition from Hyperspectral Imagery (gebruik maken van wavelength 700nm wat mensen normaal zijn)- Background removal and human detection methods, based on video changes, color combination and selection and more (ViBe en ground-truth zijn interessant)- Body part recognition (kan helpen bij algemene herkenning van personen)- Model based recognition (vooral gebruik makend van bewegingen in beeld)	Probleemstelling en verhaal er omheen.
Raoul	- Lift herkenning	An Effective Method for	Nog een bijdrage aan het

	geïmplementeerd	People Detection in Grayscale Image Sequences DIPlib Funtion Reference DIPimage User Manual	PvE
Tim	-	Projecthandleiding om te kijken wat de eisen voor het verslag waren.	Basisverslag opgesteld en alles dat al af was hier ook gelijk ingezet.

Week 2 (donderdag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Segmentatie, Labeling en weergave van de twee in de GUI implementeren	DIPimage user manual	Programma van eisen verbeterd.
Marnix	Gui, dummy functies, beeld weergeven, Eigenschappen bepalen	DIPImage http://svn.assembla.com/svn/et3660in_2/MATLAB/Tools/dip_image/dipimage/msr2obj.i.m msr2obj, http://www.computing.dundee.ac.uk/staff/stephen/cviu2000_www.pdf , http://nguyendangbinh.org/Proceedings/ICCV/2003/iccv03/0122_yang.pdf , http://www.dtic.mil/dtic/tr/fulltext/u2/a459706.pdf	PvE verbeterd
Mirko	De pipeline met simpele functie, het normaliseren en enkele andere algemene problemen in de hoofd-applicatie oplossen	DIPImage	Test plan
Raoul	Segmentatie, lift-detection in still-images, mergen van lift-detection en de GUI	DIPimage user manual	PvE
Tim	GUI, dummy functies, 4x beeld weergeven, begonnen met eigenschappen bepalen	DIPImage	PvE verbeterd en opgestuurd

Week 3 (maandag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Optimal thresholding prototype getest (geeft tot nu toe nog teveel andere gelabelde objecten), liftdetectie (=segmentatie+labeling) verbeterd	Helpbrowser Matlab voor hist en histc, A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems , Digital Image Processing (Chapter 10) by R.C. Gonzalez and R.E. Woods	-
Marnix	Achtergrond wegfilteren, object herkenning	http://www.computing.dundee.ac.uk/staff/stephen/cviu2000_www.pdf (mensen herkennen, achtergrond wegfilteren) Matlab documentatie over verschillende beeldbewerkingsfuncties Recovering human body configurations: combining segmentation and recognition (lichaamsdelen herkennen)	-
Mirko	Geholpen met het herkennen van de lift door de gemaakte plaatjes te analyseren. Eindelijk weer uit elkaar getrokken beeld terug kunnen combineren en een beetje normalisatie dus.	Enkele simpele matlab documentaties over de werking van bepaalde functie. Verder niet relevant.	-
Raoul	lift-detectie, concept van het lift-visibility function geïmplementeerd en het blauwe scherm probleem opgelost.	Matlab Function Reference	

Tim	Proberen om achtergrond weg te filteren, en objecten over te houden.		
-----	--	--	--

Week 3 (donderdag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Liftdetectie verbeterd, zodanig dat de liftdeuren los van elkaar onderscheiden kunnen worden en het duidelijk is wanneer de deuren dicht zijn, deels opens staan of uit het beeld zijn. Segmentatie en labeling in programma ingevoerd.	Gauss Blur Practicum gemaakt	Meegedacht over de verbetering van het programma van eisen.
Marnix	Segmentatie verbeterd, delen samengevoegd, verbeteren algemeen, proberen te laten werken op eigen laptop	- Practicum gemaakt	PvE
Mirko	Normalisatie uitwerken, helpen met de nieuwe koppeling tussen de verschillende delen van de pipeline. Matrix conversie problemen oplossen.	http://www.mathworks.nl/matlabcentral/answers/10716-how-to-normalise-image-intensity Practicum 2 (gemaakt)	Groot deel van het tussenverslag samengevoegd en gechecked.
Raoul	lift-detectie en het lift-visibility function geïmplementeerd en verbeterd. toMatrix verbeterd	Matlab Function Reference Practicum 2	
Tim	Een de overgebleven objecten als mask gemaakt, zodat alleen de objecten in kleur overblijven.	Practicum 2 gemaakt	

Week 4 (maandag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Code liftDetect van comments voorzien, onderdeel om video in te laden maken	MathWorks Documentation Center - VideoReader class , MathWorks Documentation Center - uigetfile MathWorks Documentation Center - VideoReader read function	
Marnix	Segmentatie closing en erosie toegevoegd en code verplaatst. Labeling werkend gemaakt.	http://hal.archives-ouvertes.fr/docs/00/54/85/12/PDF/hog_cvpr2005.pdf (vormdetectie mensen)	Conclusie, prototype 2, verbetering prototype 1, meedenen PvE
Mirko	Normalisatie en wat algemene kleine onderdelen		Verslag wat betreft opmaak en consistentie verbeterd alvast voor de volgende keer
Raoul	De hele code refactored als een deel van het clean-up operatie, 1e versie van classificatie geïmplementeerd		Iets meer aan PvE gedaan, verslag nagekeken en feedback
Tim	Closing voor de objecten toegevoegd. Webcam + matlab geeft grote problemen.	http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.1771&rep=rep1&type=pdf	

Week 4 (donderdag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël			
Marnix	Skeleton functie maken.	http://www.mathworks.nl/help/images/ref/bwmorph.html	
Mirko		<p>Normalise color layers based on all the other layers:</p> <p>http://www.mathworks.se/matlabcentral/newsreader/view_thread/171190</p> <p>Normalise in different N x M subarea:</p> <p>http://mmc36.informatik.uni-augsburg.de/mediawiki/images/f/f3/VMDL97.pdf</p> <p>Normalization and segmentation based on motion and change likelihood of the color:</p> <p>http://research.microsoft.com/en-us/um/people/ablake/papers/ablake/criminisi_cvpr06.pdf</p>	Verhaal normalisatie, testplan aanpassen, helpen verbetering van de PvE.
Raoul	functies classificatie, compute verbeterd, comments en bugfixes, alternatieve functie liftDetect2 geschreven		Testplan nakijken, PvE verbeteren aan de hand van feedback
Tim	basis statistieken tonen in de GUI om uit te zoeken hoe het precies werkt.	Practicum 1 voor measurements	

Week 5 (maandag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Code liftDetect van comments voorzien, onderdeel om video in te laden maken	MathWorks Documentation Center - VideoReader class , MathWorks Documentation Center - uigetfile ,MathWorks Documentation Center - VideoReader read function	
Marnix	Skeleton functie gemaakt voor classificatie en telling van personen, en variaties hierop gemaakt.	http://www.mathworks.nl/help/images/ref/bwmorph.html	
Mirko	Het inladen van de video verbeteren en het maken van een slider om door de video te bladeren. Begonnen met andere manier van segmenteren door het toevoegen van beweging		Testplan nog even aangepast, op basis van het commentaar voor het verslag.
Raoul	functies classify en compute geïmplementeerd, bugfixes en verdere clean-up van de main file		
Tim	Closing voor de objecten toegevoegd. Webcam + matlab geeft grote problemen.	http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.1771&rep=rep1&type=pdf	

Week 5 (donderdag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Videolader gemaakt en classify voor in- en outgoing personen		Begonnen met prototype 4 beschrijving
Marnix	Auto Calibratie, optimalisatie controleren, bugfixes, Code opschonen		
Mirko	Verder met segmentatie op basis van verticale gestructureerde elementen. Belangrijke webcam problemen in prototype fixen	-	-
Raoul	alternatieve implementaties geschreven van classify en compute. Prototype 4 in elkaar gezet.		
Tim	Alternatieve manier voor liftDetect bedacht en begin mee gemaakt	Slides Beeldverwerking voor edge-detection	Opzet gemaakt met documentatie van de alternatieve manier

Week 6 (maandag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël	Classify verder uitgewerkt, enkele foutsituaties erin afgevangen en correcte integrate in GUI en systeem gemaakt.	Mathworks Documentation Center - Disp	Prototype 4 zo ver mogelijk afgemaakt, nog input van de rest nodig
Marnix	Auto calibratie begin gemaakt, skeleton geprobeerd. Mensen die dicht bij elkaar staan scheiden in 2 losse personen.	http://www.mathworks.nl/help/images/ref/bwmorph.html	Argumentatie over waarom skeleton niet goed is om te gebruiken. Prototype 3 en 4 bijgevoegd
Mirko	Geprobeerd segmentatie weer beter te maken, door onder andere literatuur te lezen. Dit door onder andere edge detection en een fill daarop toe te passen. Daarnaast moeten de frames voor de thresholding simpeler worden gemaakt, nu gebeurt dat door een gaussian filter met lage sigma die het beeld scherper maakt.	<p>Shadow removal (lastig en werkt niet super goed, dus implementatie is waarschijnlijk niet echt nuttig)</p> <p>http://ijest.info/docs/IJEST10-02-09-93.pdf</p> <p>Moving object detection (legt alleen de basis uit, niet de precieze methodes)</p> <p>http://www.ijert.org/browse/august-2012-edition?download=715%3Amoving-object-detection-using-matlab&start=100</p> <p>Human counting (korte uitleg over combineren van verschillende objecten dat bij 1 persoon horen)</p> <p>http://www.slideshare.net/iaeme/tracking-and-counting-human-in-visual-surveillance-system-15665149</p>	Verhaal over keuzes normalisatie getypt, verhaal over keuzes segmentatie voor een deel getypt

Raoul	De main file aangepast. Met Daniel aan zijn classify versie gewerkt.		
Tim	liftDetect verbeterd, het gaat nu steeds beter werken	Slides Beeldverwerking	Documentatie over liftDetect geupdate

Week 6 (donderdag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël			
Marnix	Tim met liftstatus geholpen. Kijken of tracking nuttig is, en uitgetprobeerd.	Computer Vision System Toolbox: http://www.mathworks.nl/help/vision/index.html	Prototype 5
Mirko	Smoothing en sharpening van frames bekeken en dit werkt totaal niet beter. Daarnaast gezocht naar andere manieren om het beeld te versimpelen maar kom steeds op dezelfde methodes en die zijn te smooth.		
Raoul	Compute en classify verbeterd. Outgoing detectie werkt nu. Ingoing gedeeltelijk. Prototype 5 in elkaar gezet. Anderen geholpen met hun implementaties.		
Tim	Aan liftStatus gewerkt, bleek op basis van beweging niet te werken...	-	Met documentatie van liftStatus begonnen

Vakantie

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël			
Marnix	Face Recognition uitgeprobeerd. Uiteindelijk te traag bevonden. Tim geholpen met liftdetect	Computer Vision System Toolbox: http://www.mathworks.nl/help/vision/index.html	Uitzoeken wat er mist aan verslag; begonnen met prototypes corrigeren
Mirko	Weer verder met segmentatie, aangezien dit volgens ons een heel belangrijk deel is voor de rest van het systeem en het wel goed moet werken. Andere methodes proberen/vinden. Posterization toegevoegd in de normalisatie die met grotere segmentatie thresholds een beter resultaat geeft, ook in de lift. Enkele kleine bugs verholpen, onder andere in het laden van een video.	Gebruikte posterization functie: http://www.mathworks.nl/matlabcentral/fileexchange/1362-posterize/content/posterize.m	Keuzes segmentatie en normalisatie bijgewerkt
Raoul	Alternatieve manieren bedacht en getest voor compute en classify. Ze werken nu heel goed. De nieuwe algoritme gebruikt de volgende nieuwe functies: headHunter - telt mens hoofden in een gesegmenteerde mens-blob. patchpat - Smoothd' een data-stream om het meer consistent te maken. Ook is een nieuwe segmentatie gemaakt op basis van grayscale images		

	ipv rgb images. Main file is ook aangepast. Ook 2 Test videos gecompileerd uit onze test-opnames om verschillende situaties makkelijk te kunnen testen.		
Tim	liftStatus gemaakt op basis van kleurlagen. Helemaal getweakt zodat het op alle 37 filmpjes werkt. De versie deelt de lift in vier stadia in: 'closed', 'opening', 'open' en 'closing'.		Documentatie van liftStatus en liftDetect (met afweging) afgemaakt.

Week 7 (maandag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël			
Marnix			Prototype 4, 5 en overige kleine dingen
Mirko	Tweaken van normalisatie en segmentatie.		Verhaal normalisatie keuzes afgemaakt.
Raoul	Aan iedereen uitgelegd wat ik in de vakantie gedaan heb. Implementaties van iedereen gemerged tot 1 versie. Compute en classify verbeterd, nu ~80% succesrate ipv 70%		Eindrapport: Hoofdstuk over mens-detectie. Hoofdstuk over het vaststellen van verkeer. Deel van hoofdstuk over verschillende implementatie keuzes.
Tim	Samen met Raoul liftStatus ingebouwd in de meest recente versie van het programma. Verder aan iedereen uitgelegd hoe liftDetect en liftStatus werken.		

Week 7 (donderdag)

Naam	Bijdrage implementatie	Literatuur gelezen	Bijdrage verslag
Daniël			
Marnix	Uitgebreide testvideo gemaakt en ermee getest		Alle prototypes afgemaakt en bijgewerkt.
Mirko	Segmentatie bekeken in de nieuwste versie van de rest van het systeem waar het niet meer goed bleek te werken, keuze gemaakt om er niet meer mee verder te gaan omdat simpele segmentatie beter werkt en het de laatste dag is.		Volledige segmentatie keuzes afgemaakt, inclusief keuze argumentatie voor de simpele segmentatie. Verslag in zijn geheel samengevoegd en gechecked.
Raoul	Final versie in elkaar gezet. headHunter performance verbeterd (om dat het soms traag werkte). Performance is nu ~x10 beter (nieuwe algoritme voor nested functie pattern bedacht + een aanpassing in de main scan-line loop).		Documentatie/keuzes persoonsdetectie en telling geschreven.
Tim	Uitgebreid getest.		Testverslag geschreven in tabelvorm.