2015•2016
FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN
master in de industriële wetenschappen: elektronica-ICT

Masterproef
Real-Time Motion Compensated Video Camera

Promotor :
Prof. dr. ir. Luc CLAESEN

Promotor :
Dhr. BART STUKKEN

Leendert Wilms

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

Gezamenlijke opleiding Universiteit Hasselt en KU Leuven

2015•2016

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: elektronica-ICT

Masterproef

Real-Time Motion Compensated Video Camera

Promotor :
Prof. dr. ir. Luc CLAESEN

Promotor :
Dhr. BART STUKKEN

Leendert Wilms

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

universiteit hasselt | KU LEUVEN

Woord vooraf

Voor het maken van deze masterproef heb ik kunnen rekenen op de steun van een aantal personen. Graag wil ik mijn interne promotor en begeleider professor Luc Claesen bedanken voor de zeer interessante en uitdagende masterproef en de morale steun die ik af en toe nodig had voor het voltooien van deze proef. Het was niet altijd even gemakkelijk waar verscheidene pogingen steeds een negatief resultaat gaven.

Graag zou ik ook nog Martijn Rymen willen bedanken die mij elke dag een plaats gaf om te werken en vaak handige tips gaf telkens ik op dezelfde problemen bleef botsen.

Tot slot wil ik nog mijn familie en vrienden bedanken voor de aanmoedigende woorden en steun de afgelopen maanden.

Leendert Wilms

Januari 2016

Inhoudsopgave

Lijst van tabellen

Lijst van figuren

Abstract

Huidige beeldstabilisatietechnieken berusten op het stabiliseren van een beeld tijdens het opnemen van de foto. Deconvolutietechnieken zijn softwarematige beeldrestoratietechnieken door het bewegingspad van de camera te achterhalen in een al opgenomen blurry foto.

In deze thesis wordt er onderzocht naar een alternatieve beeldstabilisatietechniek die gebruikmaakt van een hardwarematig bekomen bewegingspad, achterhaald uit sensoren om softwarematige deconvolutie toe te passen. Aan de hand van een camera wordt uitgelegd hoe beeldstabilisatie werkt. Het onderzoek naar een alternatieve beeldstabilisatie wordt door middel van een foto omschreven.

Indien real-time gebaseerde beeldstabilisatie wordt toegepast, is snelheid van de rekenkracht en een perfecte timing tussen camera en sensoren van enorm groot belang. Er wordt experimenteel een manier gezocht om een correcte timing te verkrijgen tussen een Hercules 1.3MP camera module en de MPU-6050 sensoren. Met het bewegingspad worden de algoritmes Richardson-Lucy en Wiener Filter toegepast met behulp van het programma Matlab.

Het toepassen van beide deconvolutie algoritmes op een blurry beeld met een bewegingspad verkregen uit de sensordata geeft zeer goede resultaten, buiten een soort van ringing effect. Verder onderzoek naar dit effect is nog vereist. Voor real-time compensatie zijn de technieken echter niet snel genoeg voor de huidige standaard beeldresoluties. Timing tussen de cameramodule en de sensordata is niet accuraat genoeg indien er gewerkt wordt met twee aparte systemen.

Summary

Current techniques for image stabilization are based on stabilizing an image during recording. Deconvolution techniques are software-based image restoration techniques using an already recorded blurry image for retrieving a blur kernel (motion of camera).

This thesis investigates an alternative stabilization technique using a software deconvolution algorithm on hardware obtained sensor data which represents a blur kernel. Via a camera the different types of image stabilization are explained. The alternative stabilization technique is described using a single input image.

Timing is very crucial for real-time video compensation and access to a lot of computing power is recommended. Via experiments, a correctly time based protocol is sought using the Hercules 1.3MP webcam and the MPU-6050 sensors. The Richardson-Lucy and Wiener Filter algorithms are applied to the blur kernel using a program called Matlab.

Results using both deconvolution algorithms on a blurry input image are very promising with the exception of some kind of ringing effect. Further research on this effect is required. However, these techniques are not fast enough for current standards in image resolutions. A time based protocol using two separate systems is not accurate enough.

1. Inleiding

Een camera is niet meer weg te denken in de huidige maatschappij. Bijna iedereen heeft er zelfs één op zak, namelijk zijn smartphone. De smartphone is de dag van vandaag één van de meest gebruikte camera's om een foto mee te nemen. Ze zijn ontworpen om snel en efficiënt een beeld vast te leggen. Doordat de smartphonecamera zo licht is, is deze vrij onstabiel en meer onderhevig aan blur. Blur is een ander woord voor een wazig uitziende foto of onscherpe foto. Professionele digitale spiegelreflexcamera's zijn daarentegen veel zwaarder. De reden hierachter is dat een zwaardere camera minder onderhevig is aan kleine bewegingen door zijn massatraagheidsmoment. Dit zorgt voor een stabiele camera die op zich weer scherpere foto's produceert. De veel lichtere smartphonecamera is wel onderhevig aan allerlei soorten bewegingen.

Huidige technieken voor het stabiliseren van het beeld bestaan uit twee groepen. De eerste groep van technieken gaan het beeld tijdens het opnemen stabiliseren. Een voorbeeld van deze techniek is Optical Image Stabilization waarbij het invallend beeld op de sensor door extra toegevoegde hardware zo stabiel als mogelijk wordt gehouden. De andere technieken gaan een wazig genomen foto verscherpen en verbeteren door *'image processing'*. Huidige technieken gebruikt in smartphones bestaan uit de eerste groep. Hiervoor zijn echter dedicated hardware cameramodules voor nodig waarin het stabiel houdend systeem is ingebouwd die enkel de 'flagship' smartphones hebben. Bestaat er geen stabilisatietechniek die in huidige cameramodules ook kan worden toegepast?

Blind deconvolution zijn technieken die van een blurry foto een verscherpt beeld maken. Deze technieken berusten op het achterhalen van een bewegingspad of de Point Spread Function (PSF). Blind staat dus voor een onbekende PSF. Dit bewegingspad stelt de beweging voor die de camera heeft gemaakt tijdens het nemen van de foto. Deze blind deconvolutiontechnieken vereisen echter veel rekenkracht en nemen hierdoor veel tijd in beslag.

In deze paper wordt een manier onderzocht die, met behulp van sensorwaarde van accelerometers, een bewegingspad opstellen. Immers een smartphone heeft ook accelerometersensoren aan boord. Met dit bewegingspad worden er deconvolutie algoritmes toegepast, namelijk het Richardson-Lucy algortime en het Wiener Filter algortime. Beide technieken vereisen een bewegingspad dat gekend is. De snelheid van het deconvolutie algortime zal uitmaken of ze geschikt zijn voor het stabiliseren van videobeelden. Aan de hand van een foto in plaats van videobeelden worden resultaten van deconvolutie getoond, immers video is niets meer dan snel opeenvolgende foto's.

2. Onderdelen van een digitale spiegelreflexcamera

Een camera bestaat uit een aantal onderdelen die onderling samenwerken om een digitaal beeld vast te leggen in de vorm van een foto. Deze onderdelen zijn een diafragma, een mechanische of elektronische shutter, een beeldsensor en een lens. Al deze onderdelen, buiten de lens, zitten in een behuizing die afgeschermd is van het licht. De lens is het enige onderdeel dat zorgt voor invallend licht op de beeldsensor.

Voor het nemen van een foto zijn er ook nog een aantal instellingen die al dan niet kunen worden aangepast. Deze zijn de ISO-waarden oftewel de lichtgevoeligheid van de beeldsensor, de sluitertijd oftewel hoelang het licht invalt op de beeldsensor en het diafragma oftewel de grootte van de opening.

2.1. Het diafragma – bepaling van scherptediepte

Van al de verschillende instellingen die je handmatig kan aanpassen, is het aanpassen van het diafragma het snelst merkbaar. Het diafragma is een meestal ronde of veelhoekige opening die op de optische as tussen de lens en de sensor staat. Deze opening wordt groter of kleiner gemaakt om een dieptezicht te creëren [4].

Bij een kleine opening wordt de hoeveelheid doorgelaten licht ook kleiner en neemt de scherptediepte toe. Dit laat toe om de achtergrond van een foto even scherp waar te nemen als het object of de persoon in de voorgrond. Nadelen van een kleiner diafragma is het toenemen van brekings- en buigingsverschijnselen die op de zijkanten van een foto merkbaar worden in de vorm van een vermindering in helderheid. Dit verschijnsel wordt ook wel vignettering genoemd.

Bij een grote opening wordt de hoeveelheid doorgelaten licht ook groter en neemt de scherptediepte af. De achtergrond van een foto zal hierdoor veel waziger worden dan het object of de persoon in de voorgrond. Let op: een klein getal betekent een grote opening van het diafragma en een groot getal, een kleine opening.

Figuur 1: Effect scherptediepte bij het verkleinen van het diafragma (Bron: http://www.nandoonline.com/)

2.2. De shutter – bepaling van belichtingstijd

Eén van de belangrijkste onderdelen van een camera is de '*shutter'* of sluiter. Dit is het systeem dat de hoeveelheid licht dat invalt op de beeldsensor gaat bepalen. De '*shutterspeed'* is met andere woorden de tijdspanne dat de sensor belicht wordt. Een belangrijke factor in de fotografie is om de waarde van de sluitertijd aan te passen. Een "rolling shutter" is de meest gebruikte techniek in spiegelreflexcamera's. De manier waarop deze sluiter werkt kan vergeleken worden met een rolluik dat opengaat en dichtgaat. De shutterspeed is gelijk aan de tijd tussen het opengaan en het dichtgaan van de rolluiken.

Een shutterspeedwaarde wordt uitgedrukt in een breuk die de belichtingstijd voorstelt. Indien er tijdens het nemen van een foto een voorwerp een beweging maakt, wordt deze naarmate de belichtingstijd groter is, meer uitgesmeerd. Dit merk je door op figuur 2 de onderkant van de waterval te bekijken. Een grotere breuk in shutterspeed betekent een scherper beeld bij het fotograferen van een beweging. [12]

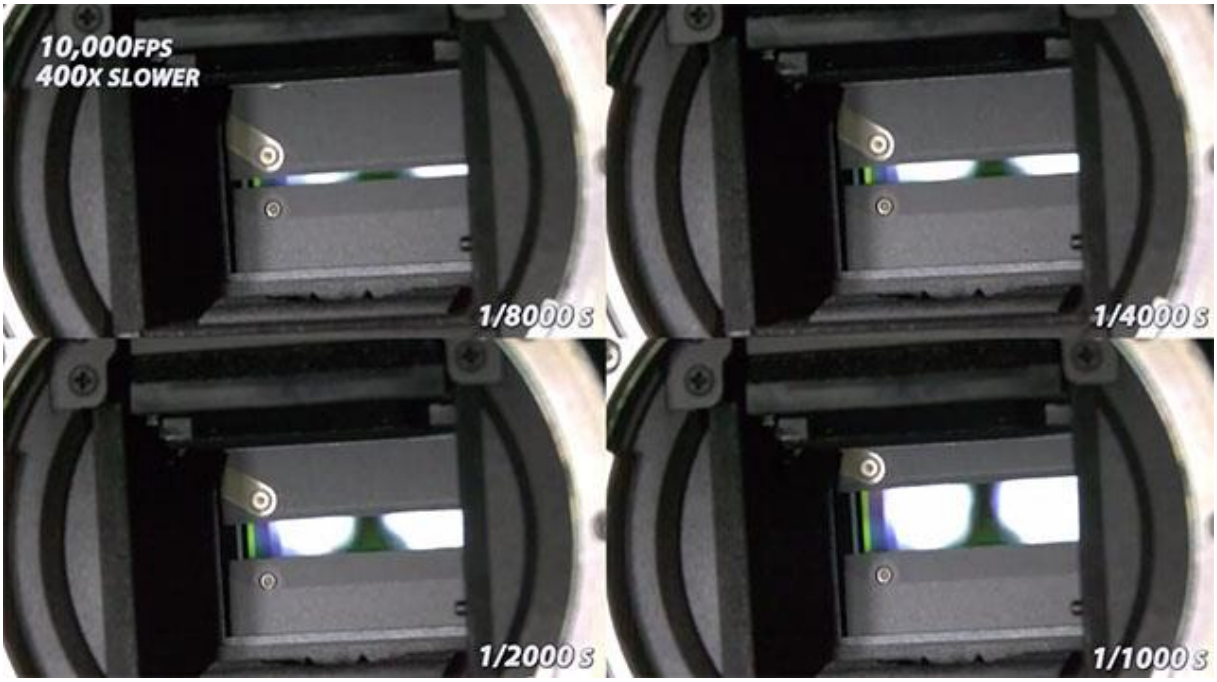
Figuur 2: Smeereffect op water door verandering van belichtingstijd (Bron: http://ilovehaagendazs.com)

2.2.1. Mechanische rolling shutter

De mechanische rolling shutter gebruikt letterlijk rolluiken die naar beneden vallen om de belichtingstijd van de sensor te controleren. Dit vertaalt zich dan ook in dat typische klikgeluid. De belichtingstijd wordt uitgedrukt in de tijd tussen het opengaan en sluiten van de rolluiken of anders gezegd de opening tussen de rolluiken van de sluiter. Een grote opening betekent een lange belichtingstijd en een kleine opening een korte belichtingstijd. Een shutterspeed van 1/4000 seconden komt overeen met een lijn op de sensor die 0,25 milliseconden lang belicht wordt. Dit betekent niet dat het hele beeld op 0,25 milliseconden tijd genomen wordt. De snelheid van de rolluiken zorgt ook voor de tijd die nodig is om de hele sensor te belichten. De snelheid voor een mechanische rolling shutter ligt in de orde van 1 milliseconde. Het beeld wordt dus in een tijd van 1,25 milliseconden opgenomen. Des te groter de shutterspeed, des te minder de snelheid van de shutter een rol speelt.

Omdat er fysiek rolluiken aanwezig zijn, kan er in sommige gevallen ook rolling shutter distortion optreden. Dit wordt veroorzaakt door het naar omlaag trekken van de rolluiken en de daarbij horende bewegingen.

De bovenstaande waarde van de sluitertijd zal in de praktijk niet snel worden toegepast. In de fotografie bijvoorbeeld wordt de sluitertijd vaak gelimiteerd tot minimum 1/125 seconden. Door de zeer korte sluitertijd in het voorbeeld van 1/4000, wordt de sensor te weinig belicht. Dit vertaalt zich in een heel donker beeld. Externe lampen en het aanpassen van de ISO-waarde zouden hierbij eventueel helpen. [3],[5]



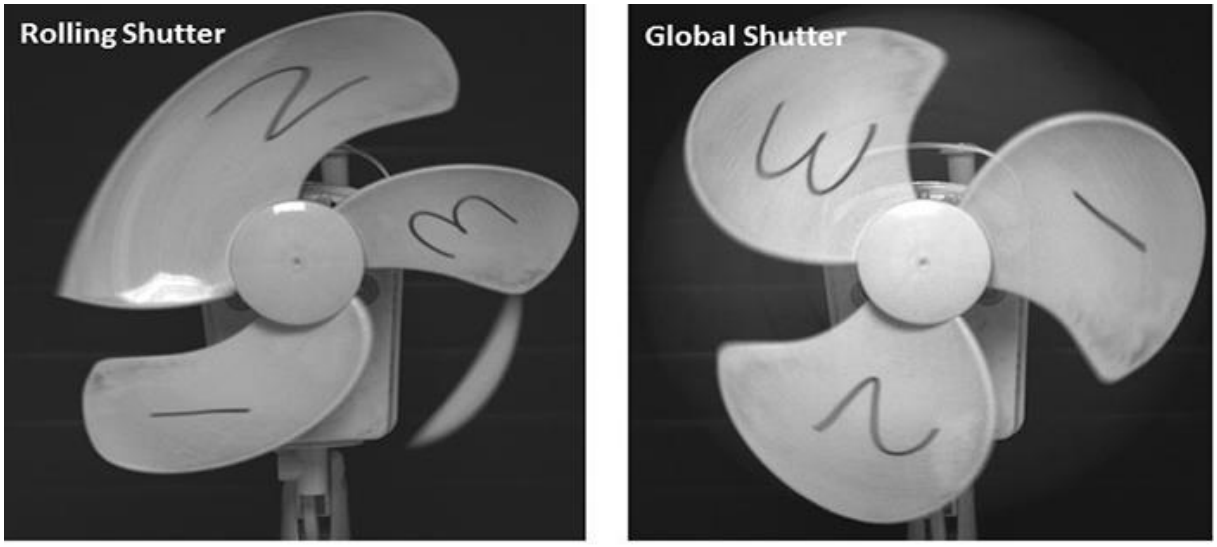
Figuur 3: Grootte van het venster door aanpassen van shutterspeed in een mechanische rolling shutter (Bron: http://petapixel.com)

2.2.2. Elektronische rolling shutter en global shutter

Een elektronische rolling shutter is vergelijkbaar met die van een mechanische, waarbij de mechanische rolluiken worden vervangen door extra aanstuurbare lijnen op de sensor. Een voordeel van een elektronische shutter is het nemen van stille foto's. Er zijn namelijk geen onderdelen aanwezig die geluid maken. Het nadeel van een elektronische rolling shutter is dat de snelheid nodig om de hele sensor te belichten, veel trager is dan een mechanische sluiter. De snelheid ligt in de orde van 30 milliseconden. Dit is dus 30 keer trager dan een mechanische. Eén van de snelste elektronische rolling shutters is deze van de RED Epic 6K camera en bedraagt 5 milliseconden. Dit is nog steeds 5 keer trager dan een mechanische. [5]

Het nadeel van een trage rolling shutter is het fotograferen van een bewegend object. Omdat de rolluiken van boven naar onder vallen, zal eerst de bovenkant van de beeldsensor belicht worden. In fucntie van de tijd betekent dit dat de onderkant van de sensor veel later belicht wordt en dus het bewegend object al een nieuwe positie heeft. Figuur 4 geeft het effect weer van een traag werkende rolling shutter bij een snel bewegend object.

Een global shutter daarentegen heeft helemaal geen last van deze traagheid van de rolling shutters. Zoals de naam al doet vermoeden wordt bij een global shutter de hele sensor in één keer belicht en niet door middel van vallende rolluiken. Hierdoor is er wel extra hardware nodig op de sensor en dat ten koste van de oppervlakte aan lichtgevoelige sensoren. Microlenzen zijn een oplossing om toch zoveel mogelijk licht op te vangen. Global shutters worden hard shutters genoemd als ze abrupt van belichten naar versperren overgaan of soft shutters als ze geleidelijk overgaan.

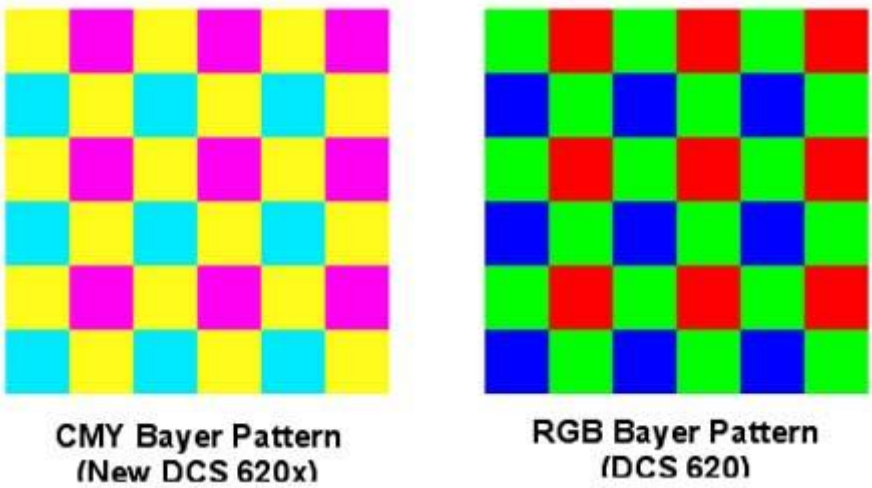


Figuur 4: Verschil tussen een rolling shutter en een global shutter bij een bewegend object (Bron: http://www.andor.com)

2.3. Opbouw beeldsensoren – CMOS versus CCD

Het hart van een camera is de beeldsensor. Een beeldsensor is een rechthoekige chip met een matrix van kleine lichtgevoelige fotodiodes. Bij een fotodiode wordt er een lichtvenster gemaakt boven een PN-junctie. Deze PN-junctie zet licht om in een elektrische lading. Boven de fotodiodes wordt een filter geplaatst voor het bepalen van de kleuren.

Er zijn verschillende soorten filters die kunnen gebruikt worden om bovenop de fotodiodes te plaatsen. Zolang men maar uit de filter de basiskleuren rood, groen en blauw kan achterhalen. De meest gebruikte filter is de Bayerfilter, bestaande uit een patroon van twee groene, een rode en een blauwe filter. Er is een extra groene filter aanwezig omdat het menselijk oog gevoeliger is voor deze kleur. Net zoals op een beeldscherm kan met behulp van deze drie basiskleuren, elke mogelijke kleurencombinatie gecreëerd worden, mits ze maar dicht genoeg bij elkaar staan zodat het menselijk oog ze niet meer van elkaar kan onderscheiden. Eén pixel bestaat uit de net vernoemde vier kleurenfilters en de onderliggende fotodiodes. [4],[5],[12]



Figuur 5: Patroon voor filters van een beeldsensor (Bron: http://www.robgalbraith.com)

2.3.1. CMOS-beeldsensor

Een CMOS-sensor staat voor Complementary Metal Oxide Semiconductor. Elke diode op de sensor verwerkt zijn eigen data. Elke fotodiode op de beeldsensor heeft een aparte elektrische lading naar spanning converter aan boord die vervolgens de spanning nog versterkt om er tot slot een digitaal signaal van te maken. De oppervlakte van het lichtgevoelige deel per fotodiode is kleiner door de extra hardware die moet worden toegevoegd. Omdat elke diode zijn eigen elektrische lading converteert naar een digitaal signaal wordt er een minder uniform beeld verkregen. Dit omdat niet elke converter perfect aan elkaar gelijk is en dus niet elke conversie gelijk is afgestemd. Aan de andere zijde doet elke fotodiode zijn eigen conversie en is het één groot parallel proces waardoor hoge snelheden worden behaald.

De meeste DSLR maken gebruik van een CMOS-beeldsensor omdat dit soort sensor wordt geproduceerd als massaproductie.

2.3.2. CCD-beeldsensor

Een CCD-sensor staat voor Charge Coupled Device. Op dezelfde manier wordt ook hier licht omgezet in een elektrische lading. Deze lading wordt vervolgens doorgegeven naar een algemene rijconverter die de elektrische lading omzet naar een spanning. Deze converter zorgt voor een grotere uniformiteit van het beeld doordat er veel meer fotodiodes door dezelfde convertor worden behandeld. Omdat er minder hardware aanwezig is, kan het lichtgevoelige gedeelte van de fotodiode ook groter gemaakt worden waardoor ze op zich weer meer licht kunnen opvangen. Dit heeft als voordeel dat er minder ruis aanwezig is bij het nemen van een foto bij een slechte belichting.

Als toevoeging wordt er ook beslist of een rolling shutter of global shutter wordt geproduceerd. Deze zal uiteraard in het proces van de beeldsensor moeten ingebouwd worden als extra hardware. Een global shutter wordt voornamelijk toegepast bij een CCD-beeldsensor.

2.4. Lens

Een lens is, buiten de beeldsensor, het belangrijkste onderdeel van een camera. De lens is het onderdeel dat het licht gaat focussen op de beeldsensor. Zij zorgt dus door middel van het aanpassen van de brandpuntsafstand tot de beeldsensor voor een scherp beeld of een wazig beeld.

Lenzen zijn niets anders dan een aantal gekromde glazen die achter elkaar staan. Er zijn verschillende soorten lenzen die variëren in focuslengte. Zo zijn er macrolenzen die optisch heel sterk kunnen inzoomen of wide-angle lenzen die gebruikt worden bij close-up shots. Lenzen communiceren zelfs met de body van de camera. Zo zal de optie autofocus de lens automatisch scherpstellen door het aanpassen van de brandpuntsafstand. Dit door middel van het verplaatsen van een gekromd glas in de lens zelf. Een andere mogelijkheid is optical image stabilization waarbij een beeld stabiel wordt gehouden door actuators in de lens. Dit laatste wordt nog verder in de paper aangehaald.

Ook op de lens kunnen filters geplaatst worden. Niet zoals de bayer filters van de beeldsensor, maar UV (C) filters, kleurenfilter, enzovoort voor een extra toegevoegd effect.

Figuur 6: Opbouw van een lens (Bron: http://www.ufunk.net)

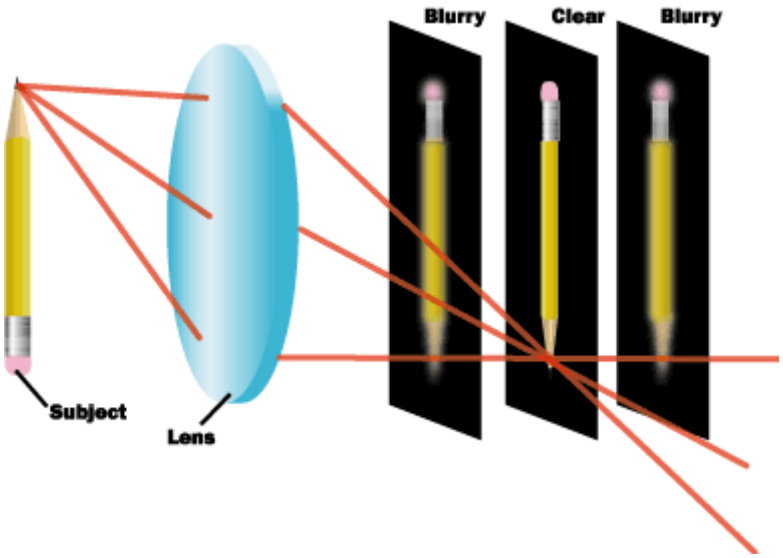
3. Verschillende soorten blur-effecten en stabilisatietechnieken

3.1. Verschillende soorten blur-effecten

In de fotografie is het belangrijk om een gedetailleerde en scherpe foto te nemen. Soms kan dit bij de minste beweging sterk tegenvallen en zal de foto er eerder wazig en onscherp uitzien. Een onscherpe foto kan liggen aan een divers aantal mogelijkheden. Zo zijn er de verkeerde instellingen, is er een slechte belichting of is er bewogen met de camera. De eerste twee zijn vrij snel oplosbaar in geval van een goede fotograaf, maar zelfs de meest ervaren fotografen hebben moeilijkheden met het laatste puntje. Dit komt omdat dit allemaal te maken heeft met de snelheid van de camera en de beweging ervan. [2],[3],[12]

3.1.1. Out of focus blur – zoom blur

Dit zijn de eerder meer voor de hand liggende voorbeelden waarom een foto wazig is. Out of focus blur betekent dat de brandpuntsafstand niet correct bepaald is waardoor het verkregen beeld na de lens niet perfect op de beeldsensor belandt. De afstand kan zowel te kort als te lang zijn. Beide effecten zijn zeer snel oplosbaar door de focuslens aan te passen tot de lichtinval van het object wel perfect op de lens invalt. Zoom blur is een ander woord voor out of focus blur.



Figuur 7: Out of focus of zoom blur (Bron: http://electronics.howstuffworks.com)

3.1.2. Motion blur

Motion blur is een effect dat een stilstaande camera heeft als er een object met een grote snelheid voorbijkomt. Omdat de snelheid van de beeldsensor niet sneller is dan de beweging van het object, zal tijdens het nemen van de foto het object verder bewegen. Het object wordt als het ware uitgesmeerd over de foto. Zowel globale als rolling shutters hebben hier last van. Een elektronische rolling shutter heeft hier net iets meer last van dan een mechanische door de snelheid van zijn rolluiken. Figuur 8 geeft een voorbeeld waarbij de dubbeldekker sneller heeft bewogen dan de shutterspeed, maar de achtergrond ervan niet. [2]

Figuur 8: Motion blur bij een bewegend object (Bron: http://webneel.com)

3.1.3. Handshake blur

Handshake blur is het effect op een foto als de persoon in kwestie geen vaste hand heeft. Omdat de camera in functie van de tijd die nodig is om de hele beeldsensor op te nemen geen vaste positie heeft, zal heel het beeld uitgesmeerd worden over de sensor. Elke pixel van de foto is uitgesmeerd over het gebied dat de camera bewogen is. Er zijn verschillende mogelijkheden om dit soort blur effect van een foto te verwijderen. Er bestaan hier zowel hardwarematige als softwarematige oplossingen voor. [11]

3.2. Stabilisatietechnieken - Image Stabilization

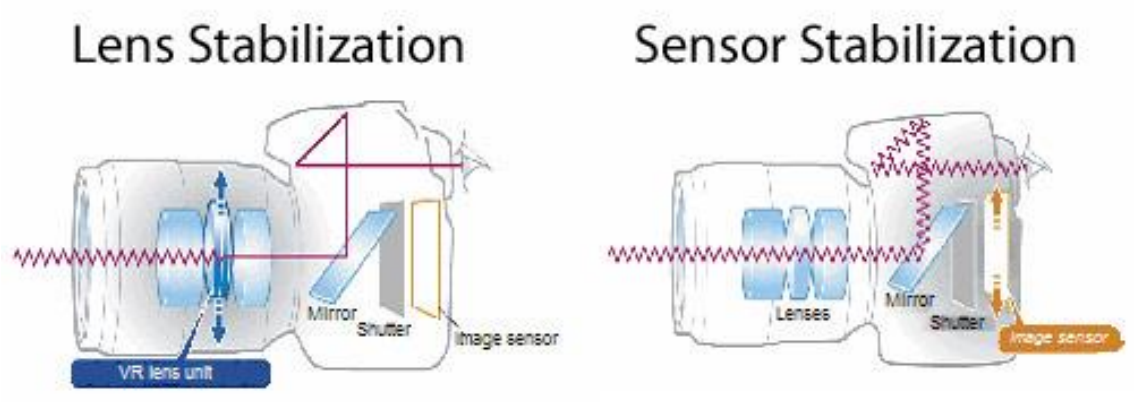
Er zijn verschillende technieken op de markt die helpen om een zo perfect als mogelijk stabiel beeld te verkrijgen tijdens het nemen van een foto. Deze technieken berusten op het verwijderen van handshake blur. Zo heeft SIGMA Optical Stabilization, NIKON Vibration Reduction, SONY Optical Steady Shot, enz. Allemaal zorgen ze voor het stabiliseren van het beeld. Hier bestaan twee verschillende technieken voor. [6]

3.2.1. Lens gebaseerd

Optische beeldstabilisatie is een techniek ingebouwd in de lens van de camera. In de lens bevindt zich een zwevend element dat is opgehangen tussen actuators. Dit zwevend element kan het optisch pad naar de beeldsensor verplaatsen zodat tijdens het opnemen van het beeld, de beweging van de camera wordt tegengewerkt. De beweging wordt opgenomen door twee gyroscopen die de horizontale en verticale beweging van de camera detecteren. [3],[7]

3.2.2. Sensor gebaseerd

Sensor gebaseerde beeldstabilisatie is een techniek die, net zoals optische beeldstabilisatie, de beweging van de camera detecteert en tenietdoet. Deze techniek wordt ook wel mechanische beeldstabilisatie genoemd. Het verschil is hier dat niet het optisch pad wordt bewogen door middel van de lens, maar de beeldsensor zelf. De sensor wordt geplaatst in een frame dat is opgehangen door actuators. Het voordeel van het bewegen van de sensor is dat elke lens kan gebruikt worden, niet zoals bij optische beeldstabilisatie waar de lenzen speciaal voor moeten gemaakt worden. [2],[3]



Figuur 9: Image stabilization bij handshake blur (Bron: https://cdn.photographylife.com)

3.2.3. Steadicam

Een steadicam is een hulpmiddel voor een cameraman dat extern wordt bevestigd aan de camera en de persoon in kwestie. Het is een harnas waaraan een mechanische arm is verbonden met een gewricht of cardanische ophanging (gimbal) zodat een camera in elke richting kan bewegen. Een steadicam maakt initieel niet gebruik van gyroscopen en berust op het verschuiven van het zwaartepunt en het massatraagheidsmoment naar de cameraman. Dit zorgt voor het neutraliseren van kleine bewegingen. Omdat het zwaartepunt verschoven wordt naar de cameraman, zweeft de camera als het ware. Instellingen en het scherpstellen van de camera door de cameraman zorgt voor extra bewegingen en wordt voornamelijk draadloos door een extern persoon gedaan.

3.2.4 Digitale beeldstabilisatie

Digitale stabilisatie is een heel andere manier van stabilisatie en wordt vaak nog extra toegepast op bovenstaande technieken. Deze techniek gebruikt de rand van het beeld als buffer voor te bewegen. Heel het beeld van de beeldsensor wordt gebruikt voor het opnemen, maar niet alles wordt getoond. Het getoonde beeld kan in deze buffer langs de rand vrij rondbewegen zodat een niet stabiel opgenomen beeld, wel stabiel lijkt te zijn. Via software, vandaar de naam digitale stabilisatie, wordt apart elke frame bekeken hoe de achtergrond ten opzichte van de vorige frame verplaatst is.

Figuur 10: Digitale beeldstabilisatie van drie opeenvolgende frames (Bron: http://elinetechnology.com)

3.3. Blind deconvolution – MAP estimation en Richardson-Lucy

Bovenstaande stabilisatietechnieken berusten allemaal op het feit dat het beeld nog moet genomen worden of wordt genomen. Al deze technieken, buiten digitale stabilisatie, vereisen extra toegevoegde componenten die voor sommigen niet beschikbaar zijn. Er zijn ook andere technieken ontworpen die een wazig genomen beeld omzetten in een scherper beeld met behulp van image processing. Met deze technieken wordt een bewegingspad gezocht dat probeert te achterhalen wat de beweging van de camera was toen de foto werd genomen. Dit bewegingspad wordt ook wel de Point Spread Function (PSF) of blur kernel genoemd. Als het bewegingspad en de originele wazige foto bekend zijn, kan er een scherper beeld bekomen worden door middel van een deconvolutie algoritme. Dit soort van technieken valt onder beeldrestoratie. [1],[10]

Als een beeld wordt opgenomen op een sensor ontstaat er een zogezegde ideale puntbron per pixel. Indien de camera beweegt tijdens het nemen van een beeld, wordt de ideale puntbron als het ware gesmeerd over een aantal pixels. De som van al deze niet ideale puntbronnen kan voorgesteld worden als volgende functie:

$$d_i = \sum_j p_{ij}.u_j \quad (1)$$

Hier staat p_{ij} voor de point spread function, u_j voor de pixelwaarde van het te achterhalen beeld en d_i voor het bekomen wazig beeld. In de wiskunde wordt verondersteld dat het beeld is opgenomen volgens een Poissonverdeling. Een Poissonverdeling is een discrete kansverdeling voor het tellen van bepaalde voorvallen over een gegeven tijdsinterval, in dit geval het aantal pixels.

Deconvolutie is een wiskundige term die staat voor het berekenen van een onbekende functie uit een convolutieproduct. Dit convolutieproduct bestaat uit één bekende en één onbekende functie. In de fotografie worden algoritmes gebruikt dat van een gegeven bewegingspad (onbekende functie) en een wazig beeld (bekende functie), een scherper beeld maakt (onbekende functie). Indien het bewegingspad niet gekend is, wordt de term blind deconvolution gebruikt, indien wel wordt de term deconvolution gebruikt.

Een ander soort stabilisatietechniek is het verscherpen van randen van objecten in het beeld. Met behulp van de eerste en tweede afgeleide worden randen van objecten gedetecteerd. Dit komt omdat een kleurenovergang een verandering in de afgeleide geeft. Een andere techniek is het toepassen van een onscherp masker. Het verschil tussen het originele en het onscherpe beeld zorgt ervoor dat details naar voren komen. Dit beeld wordt samen met het originele gebruikt voor het verscherpen van het beeld. Dit soort technieken worden beeldverbeteraars genoemd. Hier wordt verder niet dieper op ingegaan in deze paper. [12],[15]

3.4. MAP-estimation algoritme

Een eerste techniek voor het achterhalen van de PSF van een foto voor beeldrestoratie is de maximum a postpriori (MAP) estimatie. De letter y staat voor de originele en gekende wazige foto, x staat voor de onbekende scherpe foto en k staat voor het onbekende bewegingspad. Omdat niet elk beeld perfect opgenomen wordt, is er ook nog n ruis aanwezig. Hoofdletters worden gebruikt voor de Fourrier getransformeerde. Convolutie komt namelijk overeen met vermenigvuldigen in het Fourrier domein.

$$y = k \otimes x + n \qquad \text{en} \qquad Y_w = K_w . X_w + N_w \tag{2)(3}$$

Blind deconvolution gaat met slechts één bekende variabele, namelijk de originele wazige foto, de PSF proberen zo goed als mogelijk te achterhalen en daaruit een beeld zonder blur creëren. De meest simpele en gebruikte techniek van MAP-estimatie is een maximale schatting maken voor x en k zodat:

$$P(x,k|y)\ \alpha\ P(y|x,k)P(x,k) = P(y|x,k)P(x)P(k) \tag{4}$$

Waarbij P(y|x, k) staat voor de verdeling van de ruis, en P(x), P(k) staan voor het voorafgaand beeld en blur kernel. Een verdere wiskundige uitleg wordt niet aangehaald in deze paper.

Met behulp van deze kansberekening wordt iteratief een zo goed als mogelijk bewegingspad achterhaald. Deze MAP-estimatie geeft een goede benadering indien er geen of zeer weinig noise aanwezig is.

In deze paper wordt echter het bewegingspad achterhaald met behulp van sensoren waardoor blind deconvolution algoritmes niet nodig zijn. [13],[15]

3.5. Richardson-Lucy algoritme

Het Richardson-Lucy algoritme is een algoritme dat een onsherp beeld verscherpt waarbij het bewegingspad gekend is. De idee achter het deconvolutie algoritme is opnieuw van een wazig beeld en een gekend bewegingspad het meest waarschijnlijke scherp beeld te creëren, waarbij er iteratief gezocht wordt volgens volgende formule:

$$u_j{}^{(t+1)} = \sum_j p_{ij} u_j{}^{(t)} \text{ met } c_i = \sum_j p_{ij} u_j{}^{(t)} \tag{5}$$

Uit onderzoek is gebleken dat naarmate de iteratie convergeert, de maximale waarschijnlijkheid als oplossing bekomen wordt onder een Poisson verdeelde grafiek. In Matlab bestaat er de functie deconvolucy waarbij de input I als frame wordt gebruikt en de sensorwaarden als PSF. Deze sensorwaarden moeten eerst omgezet worden in een beeld waarbij de breedte van het bewegingspad wordt omgezet in een breedte in pixels van de PSF. Het algortime kan zeer effectief zijn indien de toegevoegde ruis niet of slecht gekend is. De standaard iteratiewaarde voor Matlab bedraagt 10.

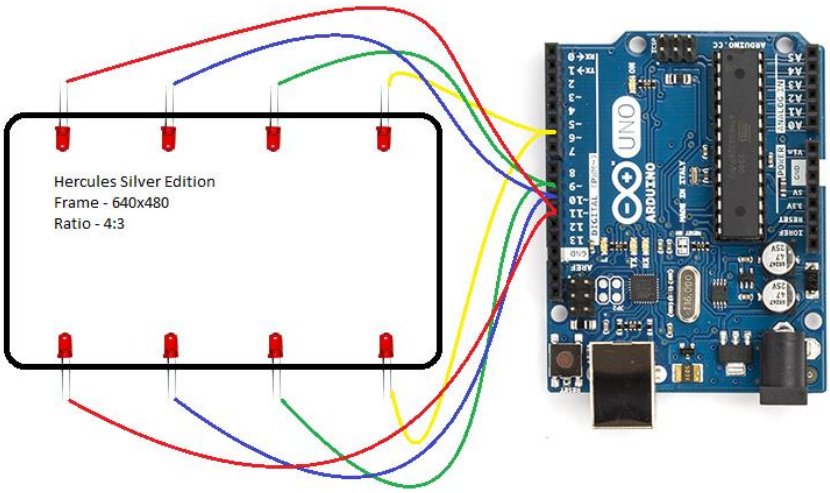
Ruisversterking is een algemeen gekend probleem bij algoritmes die een maximale waarschijnlijkheid methode gebruiken. Na vele iteraties kan een hersteld beeld een gespikkelde verschijning hebben. Dit geldt zeker voor een glad oppervlak in het beeld onder lowlight omstandigheden. Deze spikkels behoren niet tot een vaste structuur van het beeld maar worden gecreëerd als neveneffect van het algoritme om de ruis in het beeld te verwerken. [14],[15]

4. Bepalen van traagheid beeldsensor

De cameramodule waarop getest wordt is de Hercules silver edition. Als eerste is er onderzocht hoelang het duurt voor de camera om de gehele beeldsensor uit te lezen. Er wordt namelijk gebruikgemaakt van een rolling shutter. De traagheid van de beeldsensor is niet meteen gelijk aan de shutterspeed. Om hier achter te komen wordt er een ledje aan de onderkant en aan de bovenkant van het beeld geplaatst. De ledjes zijn verbonden met de analoge outputs van een Arduino Uno die via een PWM-signaal de intensiteit van de ledjes verandert. Beide leds worden door dezelfde pin op de Arduino aangestuurd. Figuur 11 geeft de opstelling weer. Door de tijd te veranderen die de intensiteit van de ledjes opvoert, wordt bepaald hoelang de sensor nodig heeft voor het beeld uit te lezen. Uit de codec-informatie van een vooraf opgenomen video wordt de maximale framerate gehaald. Deze bedraagt 29.97fps wat neerkomt op een nieuw frame elke 33.367ms. De beeldsensor wordt dus zeker binnen deze tijd uitgelezen.

Om een controleruimte te creëren wordt een donkere ruimte zonder buitenlicht gebruikt. Enkel de leds zorgen voor een verandering van de intensiteit. Om de traagheid van de beeldsensor te bepalen, worden zowel de onderstaande als bovenstaande leds iteratief verlaagt. Het verschil in intensiteit tussen de bovenstaande en onderstaande leds bepaalt de snelheid van de beeldsensor. Het verschil in tijd die bij de intensiteitswaarde horen is dan de tijd nodig om de hele beeldsensor uit te lezen.

Omdat er enkel rode leds worden gebruikt, wordt voor het bepalen van de intensiteit naar het rode beeld gekeken van een 8-bit RGB kleurenfoto. De intensiteit wordt berekend op het aantal pixels die boven een bepaalde tresholdwaarde liggen. Hoe hoger de tresholdwaarde, hoe sneller de grafiek in verzadiging komt bij de maximale intensiteitswaarde. Hoe lager de waarde, hoe sneller instabiele veranderingen worden opgenomen. Een ideale tresholdwaarde ligt dus ergens in het midden tussen deze uitersten.

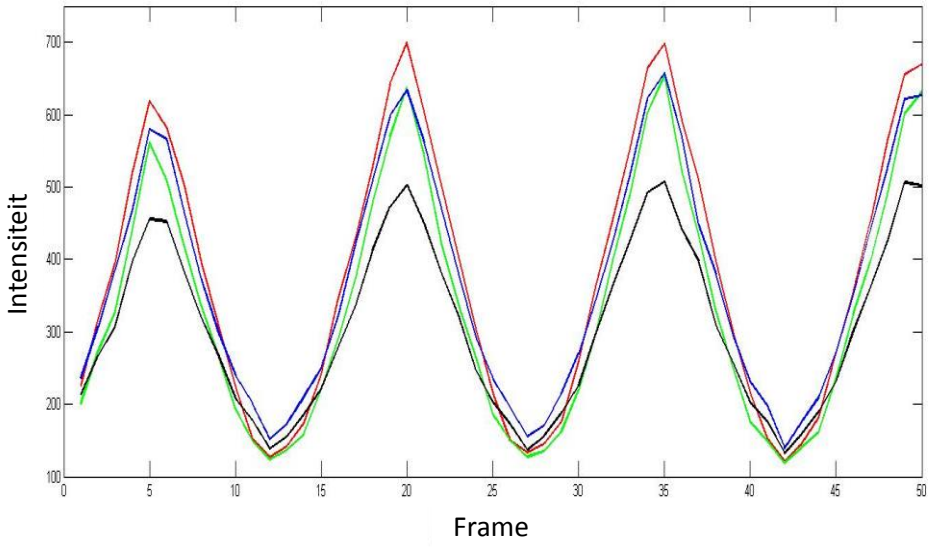


Figuur 11: Opstelling ledjes aangestuurd door de Arduino Uno

Uit de formule voor het bepalen van de intensiteit door gebruik te maken van het vermogen, achterhalen we dat de intensiteitswaarde lineair verandert ten opzichte van het vermogen. Dit vermogen wordt aangestuurd door een 8bit analoog PWM-signaal te sturen naar de leds waarbij de led op zijn maximum brandt bij 255 en op zijn minimum brandt bij 0. De noemer is de oppervlakte van een bol als we aannemen dat het licht in alle kanten evenredig verdeeld wordt.

$$|I| = \frac{P}{4\pi r^2} \tag{6}$$

Een voorbeeld hiervan is figuur 12 waarbij de verandering in tijd 10 milliseconden bedraagt en de verandering in intensiteit 5. Aan de onderkant van de grafiek treedt er lichte verzadiging op. Dit is merkbaar door de aanwezigheid van de ronding. Ter illustratie is enkel de verandering van intensiteit van de vier bovenstaande leds getoond. De tresholdwaarde van intensiteit bedraagt er 100.



Figuur 12: Bepalen van traagheid beeldsensor door verandering in intensiteit; bovenstaande leds

Om de leestijd te bepalen van de beeldsensor wordt iteratief de verandering in intensiteit aangepast over een bepaalde tijd. Er loopt een vrij goed lineair verband tussen de maximale en minimale intensiteitswaarde en uit de formule van intensiteit wordt een lineair verband gehaald. Vervolgens wordt de formule voor het bepalen van de richtingscoëfficient voor een lineaire rechte gebruikt.

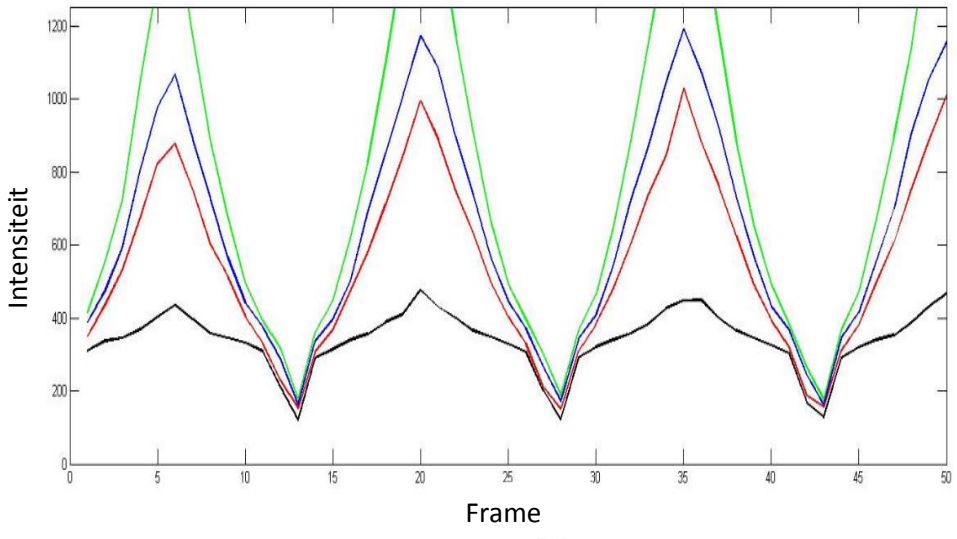
$$y = \frac{(y_b - y_a)}{(x_b - x_a)} x + b \qquad met\ b = 0 \tag{7}$$

Hierin is y het aantal pixels per intensiteit en x de tijd nodig tussen de minimum en maximum intensiteitswaarde. Het PWM-signaal daalt met een constante waarde van 5 per tijdsinterval van 10ms. Dit tijdsinterval moet kleiner zijn dan de maximale framefrate en niet kleiner dan de maximale leessnelheid van de Arduino om de sensorwaarde uit te lezen. Daarom is een waarde van 10ms gekozen. Dit betekent dat het totale tijdsinterval 510ms bedraagt. Onderaan bevindt zich de globale framerate en traagheid van de beeldsensor per leds (bovenstaand en onderstaand). De video is rechtstreeks opgenomen in Matlab waarbij de framerate 15fps bedraagt. De steekproef van dit onderzoek bedraagt 100 frames.

Tabel 1: Globale framerate en traagheid van beeldsensor

	LEDS 1	LEDS 2	LEDS 3	LEDS 4
GEMIDDELDE FRAMERATE:	**15,02**	**14,94**	**15,23**	**20,70**
GEMIDDELDE TRAAGHEID:	**30,03**	**28,33**	**26,72**	**67,33**

De gemiddelde framerate uit de onderzochten waarde bedraagt ongeveer 15fps en de gemiddelde traagheid van de beeldsensor bedraagt 28ms. Buiten leds 4 (onderstaand en bovenstaand) zijn alle data rond deze twee waarden gelegen. Een reden waarom de leds 4 andere waarden geven is omdat deze op de rand van het beeld liggen. Hierdoor kan het zijn dat niet alle intensiteitwaarden evenredig opgenomen worden. Indien er gekeken wordt naar de onderstaande leds op figuur 13, valt er direct op dat de lineairiteit van led 4 zeer slecht is aan de onderzijde. Vanaf een bepaalde waarde verandert de rico van de rechte waardoor de grafiek minder snel stijgt dan de andere leds. Onderstaande led 4 heeft dus een slecht globaal lineair verband.



Figuur 13: Bepalen van traagheid beeldsensor door verandering in intensiteit; onderstaande leds

Merk op dat de traagheid van de beeldsensor slechts voor één opstelling is achterhaald. De shutterspeed is niet geweten van de camera. Dit houdt in dat de traagheid korter kan zijn indien er veel meer licht aanwezig is in het beeld, immers de shutterspeed bepaalt ook grotendeels de traagheid van de beeldsensor.

Ook maakt de camera gebruik van auto adjust brightness. Bij deze opstelling heeft auto adjust brightness niet opgetreden omdat de gemiddelde intensiteitswaarde van het beeld boven een bepaald niveau moet komen. In een andere opstelling kan deze wel opspringen tijdens het meten van de intensiteit van de leds. Dit betekent dat er een gelijkaardige sprong zal voorkomen in de grafiek zoals bij leds 4. De rico van het lineair verband verandert dus naarmate de intensiteitswaarde de auto adjust brightness verandert. Een voorbeeld van het optreden van auto adjust brightness wordt later nog aangehaald.

5. Ophalen sensordata

Voor het ophalen van de sensordata wordt een MPU-6050 module gebruikt die is verbonden aan een Arduino Uno. De MPU-6050 heeft 3 accelerometers en 3 gyroscopen. Enkel de accelerometer in de respectievelijke x- en y-richting worden gebruikt voor het bepalen van het bewegingspad. De Arduino gaat via een I^2C-protocol communiceren met de MPU-6050 voor het uitlezen van de sensorwaarden. De kloksnelheid van de Arduino bedraagt 16MHz en de uitleessnelheid van de accelerometers bedraagt 1KHz. Theorethisch gezien is het mogelijk om elke milliseconde de sensorwaarde op te vragen indien de sensorwaarde steeds in de buffer van de MPU-6050 geplaatst wordt alvorens het uit te lezen van de samples. De nauwkeurigheid van de Arduino bedraagt 1 microseconde. Onderstaande tabel toont 5 samples van de gemiddelde uitleestijd voor een steekproef van 100 sensorwaarden.

Tabel 2: Gemiddelde uitleestijd van de Arduino met samples van MPU-6050

AANTAL [#]:	100	100	100	100	100
GEMIDDELDE TIJD [µS]:	2124,24	2124,16	2124,48	2124,48	2124,36
GEMIDDELDE TIJD [µS]:	**2124,344**				

Indien de snelheid voor het veranderen van de intensiteit van de led er nog bij op wordt geteld, wordt een globale traagheid bekomen voor het uitlezen van de Arduino. Deze tijd is echter zeer klein en in de orde van enkele microseconden zodat ze bijna onbestaande is ten opzichte van de uitleessnelheid. Door de traagheid van de beeldsensor te delen door de gemiddelde snelheid van de sensordata, kan het gemiddeld genomen aantal datapunten per bewegingspad worden vastgesteld. In de gecontroleerde ruimte bedraagt deze afgerond 13 datapunten.

Dit betekent dat er maximaal 13 sensorwaarden per genomen frame kunnen verzameld worden. De accelerometers geven een waarde weer die de versnelling voorstelt in een bepaalde richting. Uit de formule van de versnelling kan de snelheid van de camerabeweging gehaald worden door te delen door de gemiddelde tijd nodig voor het uitlezen van de sensorwaarden. Vervolgens wordt het bewegingspad opgesteld uit de afgelegde weg.

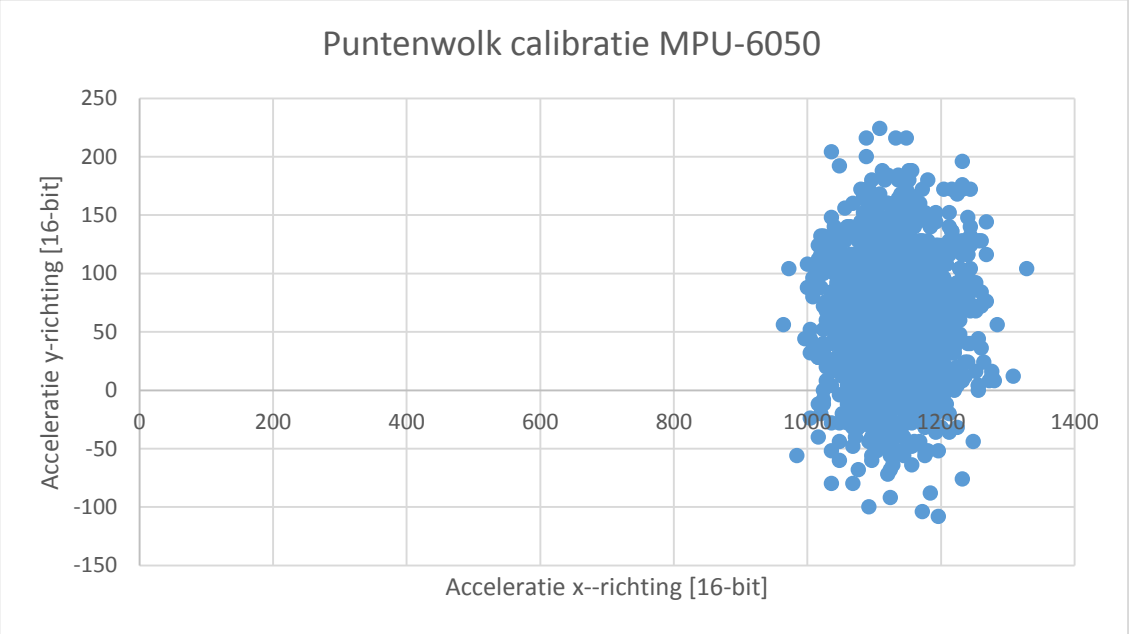
$$a = \frac{dv}{dt}, \quad dv = a.dt \qquad \text{oftewel} \qquad v(t) = v_0 + at \tag{8}$$

$$x = x_0 + v_0 t + \frac{1}{2}at^2 \tag{9}$$

Bij het ophalen van de eerste sensorwaarde is v_0 en x_0 steeds gelijk aan nul, dit omdat er voor het ophalen van een beeld nog geen pixelwaarden aanwezig zijn in de beeldsensor. Vervolgens schuiven deze waarden iteratief op. De waarde van de versnelling wordt, omgerekend en gekalibreerd, gehaald uit de sensordata van de MPU-6050. De sensorwaarden kunnen gelijk gesteld worden aan het venster van de rolling shutter die op dat ogenblik passeert voor de beeldsensor. De x- en y-richting worden bepaald door apart de sensordata te interpreteren in een orthogonaal assenstelsel.

Omdat de shutterspeed echter niet gekend is, kan de grootte en dus de breedte in pixels van het shuttervenster niet achterhaald worden om zo erachter te komen welke sensorwaarden allemaal gelezen zijn binnen dit venster van de rolling shutter. Daarom worden alle sensordata voor het bewegingspad gebruikt voor het hele frame. De snelheid alleen heeft geen betekenis bij de sensorwaarden want deze moet immers eerst gelinkt worden aan een breedte in pixels.

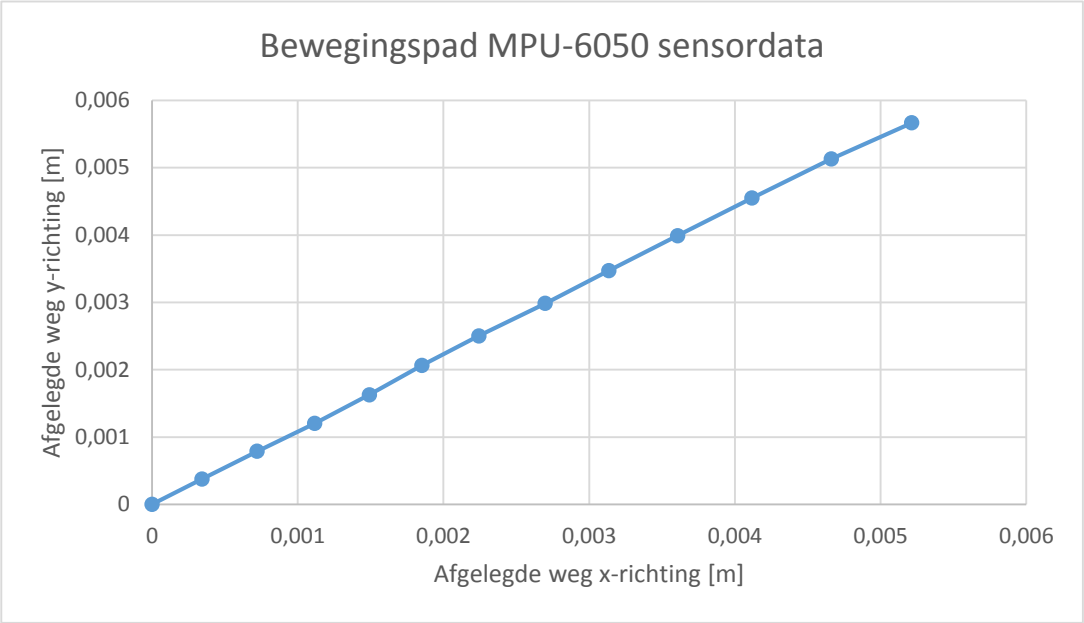
Vooraleer deze data mag geïnterpreteerd worden, moet er een calibratietest gedaan worden. Deze test bepaalt in hoeverre er beweging wordt gedetecteerd bij een vooraf gekende controlebeweging. Als controlebeweging wordt er simpelweg niet bewogen zodat de minimum- en maximumwaarde kunnen bepaald worden aan de hand van de uitersten. De MPU-6050 werd geplaatst op een vaste ondergrond (tegelvloer) voor zo weinig mogelijk externe vibraties te meten. Indien er een beweging wordt geregistreerd, is er de kans dat de sensor helemaal niet bewogen heeft. Deze maximale bewegingen horen bij hetzelfde punt en behoren dus toe aan dezelfde pixel. Er moet dus een minimum aan acceleratie worden gemeten alvorens er kan gesproken worden over een beweging. De MPU-6050 geeft een 16-bit integer breukwaarde weer tussen 0 en 2g acceleratie. Dit betekent dat 0 staat voor geen acceleratie beweging en [-32768, 32767] staat voor respectievelijk -2g of 2g. Figuur 14 geeft de uiterste waarden weer van een nulbeweging. De steekproef bestaat uit meer dan 1000 sensorwaarden. Merk op dat deze waarden dus tegengewerkt moeten worden alvorens een bewegingspad kan worden opgesteld. [16]



Figuur 14: Puntenwolk calibratie voor bepaling van uitersten bij de MPU-6050

Omdat enkel in de x- en y-richting de sensorwaarden worden uitgelezen is het niet nodig om de afstand te bepalen tussen het object en de camera. Het gehele beeld verschuift immers hetzelfde aantal pixels. Indien er bewogen wordt volgens een rolbeweging over de x-as en y-as (pitch en yawn beweging) is de afstand tot het voorwerp of de persoon wel nodig. Dit wordt echter niet aangehaald in deze paper.

Figuur 15 geeft een bewegingspad weer van 13 opeenvolgende sensordata waarvan de MPU-6050 bewogen is in een schuine opwaartse rechte.



Figuur 15: Bewegingspad van een opwaartse schuine rechte van 13 datapunten

Een datapunt komt overeen met een waarde van de x-beweging en y-beweging die op éénzelfde moment worden uitgelezen door de Arduino. Zo worden er in totaal 26 datawaarden opgevraagd waaruit 13 datapunten worden opgesteld.

Als de bewegingssnelheid van de beeldsensor gekend is, is het volgende belangrijke punt de afgelegde weg linken aan een breedte in pixels in het PSF. De lengte van het bewegingspad vertaalt zich in het aantal pixels dat er bewogen zijn. De methode voor het bepalen van de afstand is enkel iteratief te bepalen en hangt van persoon tot persoon af. Iemand kan een foto als scherp ervaren terwijl een andere deze nog als wazig ervaart. Indien het bewegingspad uit de sensordata kan gehaald worden en een wazig beeld uit handshake blur wordt opgenomen, kan er een deconvolutie algortime worden toegepast. De belangrijkste factor die nog niet aan bod is gekomen is de timing. Immers de start wanneer het beeld wordt opgenomen moet dezelfde zijn als de start van de sensorwaarde.

6. Onderzoek voor timing van camera en sensordata

Bij het toepassen van deconvolutie zijn er twee belangrijke factoren, namelijk het bewegingspad of de point spread function (blur kernel) en de wazig genomen foto. Blind deconvolutie algoritmes gaan een bewegingspad achterhalen uit de foto zelf. Er wordt dus uit de foto zelf een bewegingspad gehaald en niet uit externe sensorwaarden. Op vlak van timing is hier geen sprake van aangezien het algoritme pas kan werken als er al een foto is genomen. In deze paper worden echter twee aparte systemen aan elkaar gelinkt. Enerzijds de hercules camera module en anderzijds de sensorwaarden van de MPU-6050. Het verschil zit er dus in dat het bewegingspad niet uit het beeld wordt achterhaald, maar uit externe sensorwaarden.

6.1. Onderzoek n°1: Gezamenlijk startsignaal voor camera en sensordata

Een eerste experiment hield in om serieel een gezamenlijk startsignaal te versturen naar de beeldsensor en de Arduino met de onderlinge gedachte dat beide op hetzelfde moment beginnen zolang de pc maar snel genoeg is. Om dit te verifiëren worden in het beeld van de beeldsensor ledjes verlicht op het startsignaal dat gegeven wordt door de pc. De ledjes worden als een rollend licht aangestuurd van links naar rechts telkens er data worden uitgelezen of met andere woorden telkens er een frame wordt opgehaald. In de video die vervolgens ontstaat moeten de ledjes dus frame per frame van links naar rechts verschuiven als een rollend licht.

In de veronderstelling dat een pc snel genoeg is om een signaal te versturen naar de Arduino om direct daarna een signaal te sturen naar de beeldsensor, worden deze onder elkaar geplaatst in een loopfunctie. De ledjes branden echter niet in sequentie. Frames worden als het ware willekeurig gekozen, ledjes branden soms helemaal niet of zijn aan in twee opeenvolgende frames. In Matlab bestaat er een functie voor het parallel lopen van twee aparte processen genaamd *'parfor'*. Indien beide processen apart worden behandeld en aangestuurd worden met een gezamenlijk signaal, zou dit een betere oplossing zijn als bovenstaande seriële for loop. Dit is echter niet mogelijk omdat in een *parfor* alle parameters moeten staan. Met andere woorden moet Matlab steeds opnieuw connectie maken met de arduino per keer dat een frame wordt opgehaald. Er moet dus steeds opnieuw een initialisatiefase doorlopen worden die enkele seconden duurt.

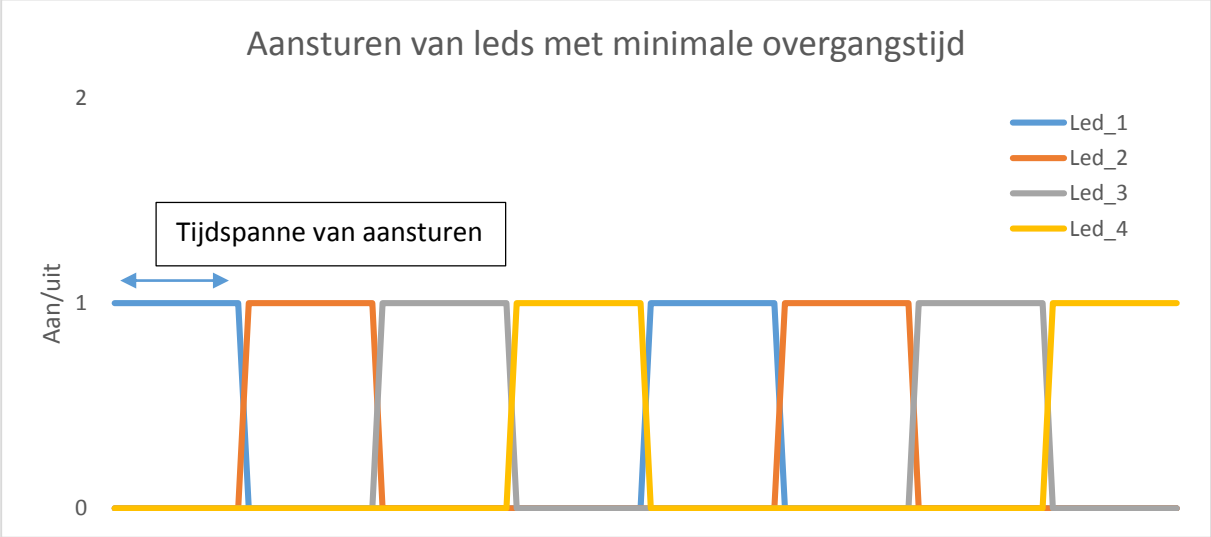
Een mogelijke verklaring voor het fout aflopen van de seriële loop is het afhandelen van het USB-protocol. De snelheid van usb2.0 bedraagt 480Mbits/sec voor high speed en 1.5Mbits/sec voor low speed. Voor het versturen van een simpele integer naar de Arduino is het dus snel genoeg om kort daarna de beeldsensor aan te sturen. De sensordata van de Arduino worden pas opgehaald als het beeld van de camera al is doorgestuurd. De Arduino bepaalt aan de hand van een vooraf opgestelde waarde hoelang er data worden opgehaald uit de sensoren. De afhandeling van meerdere connecties in het USB-protocol zal het dus niet mogelijk maken om snel achtereen twee verschillende apparaten aan te sturen. Een andere mogelijkheid is dat de cameramodule te laat reaggeert op een commando, maar dan zou er een steeds voorkomend verschil in de ledjes te merken zijn. Dit is echter niet het geval.

6.2. Onderzoek n°2: Aparte systemen met driftcompensatie voor sensordata

Een andere mogelijkheid voor het timing probleem is het proces als twee aparte systemen beschouwen. Het eerste systeem bestaat uit de Arduino waarop de ledjes bevestigd zijn. De ledjes veranderen van staat als een rollend licht van links naar rechts, zoals in het vorige experiment. Hierbij gaat de Arduino door middel van een wachttijd de ledjes aansturen. Deze wachttijd wordt bepaald door de framerate van de camera. Het tweede systeem is dus de camera die wordt aangestuurd door de native software van de fabrikant zelf. De software behaalt een framerate van 29,97 fps, bekomen uit de codec informatie van een vooraf opgenomen testvideo. Er wordt dus een frame elke 33.3667 microseconde opgehaald.

Voor de eerste 20 frames komt dit goed overeen, maar daarna treedt er een drift op. Deze drift vertaalt zich in de ledjes die niet meer per frame veranderen, maar beginnen te versnellen of te vertragen in opvolgende frames. Dit merk je aan een ledje dat normaal brandt in een bepaalde frame dat nu plots in een opvolgende frame brandt. Of twee ledjes die branden in de overgang van het rollend licht. Dit kan liggen aan een foutieve framerate door de codec, of aan het feit dat de timing in de Arduino niet optimaal is. De drift is compenseerbaar door een juistere framerate te bepalen. Volgens de codec informatie van Windows Live Moviemaker wordt er slechts een 24.89fps behaald. Dit komt neer op een nieuwe frame elke 40.17ms. De beste methode die meer dan 100 frames stabiel blijft haalt echter een nieuw frame op elke 35.33ms. Er blijft dus steeds drift optreden naarmate er langer wordt gefilmd.

Het startschot voor het nemen van de foto voor deze methode is ook zeer onduidelijk. Indien de timing wel perfect zou zijn, is het nog niet mogelijk om juist één frame aan bepaalde sensordata te koppelen. Dit komt omdat het ophalen van de frame gebeurt in een tijdspanne gelijk aan de frames per seconde, net zoals de tijd nodig om de ledjes te veranderen als een lopend licht. Met andere woorden indien er op een frame een ledje aan staat, is het onmogelijk om uit dat frame te halen wanneer dat ledje precies is begonnen met 'aan' te staan. Figuur 16 geeft hier een duidelijker beeld over. Het ophalen van een frame kan dus binnen deze tijd gebeuren.



Figuur 16: Tijdspanne van aansturen bij driftcompensatiemethode

Hoe dan ook is na een bepaalde tijd deze methode niet meer beschikbaar door de optredende drift. Er is zelfs ook nog een mogelijkheid op een dropped frame. Dit betekent dat er tijdens het filmen een frame niet volledig wordt opgehaald. Drop outs zijn frames die zelfs volledig weg zijn. Drop outs kunnen eventueel worden opgevangen door te kijken of het lopen licht van de ledjes een tel heeft overgeslagen. Het verschil van een drop out frame ten opzichte van drift is de overgang van de leds. Door de trage shutterspeed van de camera zal een drift zich vertalen door een lopen licht dat voorop huppelt. Dit is merkbaar indien er twee (de ene zal wat zachter branden dan de andere door de overgangsfase) in plaats van één led brandt. Een dropped frame daarentegen zorgt er gewoon voor dat in een video, tussen de frames, een led wordt overgeslagen.

Er zal een calibratie nodig zijn in functie van de tijd die kan bepalen waar precies de frame wordt opgenomen in de tijdspanne van de led of zelfs tijdens een overgangstijdspanne van een drift tussen de leds. Dit is mogelijk door de intensiteitswaarden van de leds aan te passen in functie van de tijd, omdat de leds deel uitmaken van het beeld en de sensordata. Er wordt gebruikgemaakt van een soort gemeenschappelijke factor, namelijk de leds in beeld.

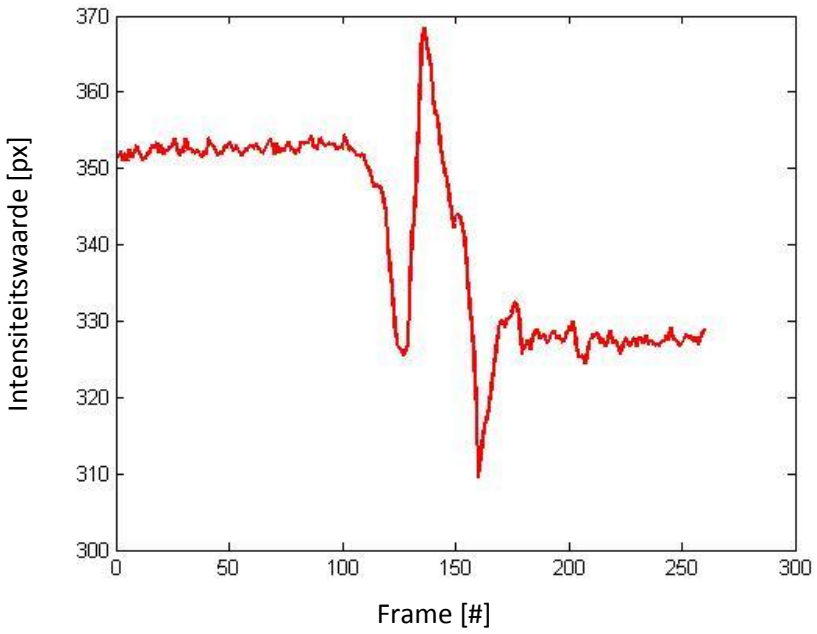
6.3. Onderzoek n°3: Meten van intensiteitswaarden voor tijdsbepaling van een frame

Door de drift en het niet weten wanneer de frame binnen een tijdspanne wordt opgehaald, is het zo goed als onmogelijk om deze data aan elkaar te linken. Er is een techniek nodig die op elke frame kan vaststellen waar in de tijd de frame genomen is en welke sensordata met deze tijd overeenkomt. Een techniek die een verandering van tijd weergeeft.

Net zoals bij het bepalen van de traagheid van de beeldsensor, kan de intensiteitswaarde van de leds bepaald worden per frame. De intensiteit die voor een lineair verband zorgt, moet duidelijk zichtbaar zijn in de frame en hangt af van het histogram van de de frame zelf. Dit komt omdat elke frame al een bepaalde intensiteit met zich meebrengt door het omgevingslicht.

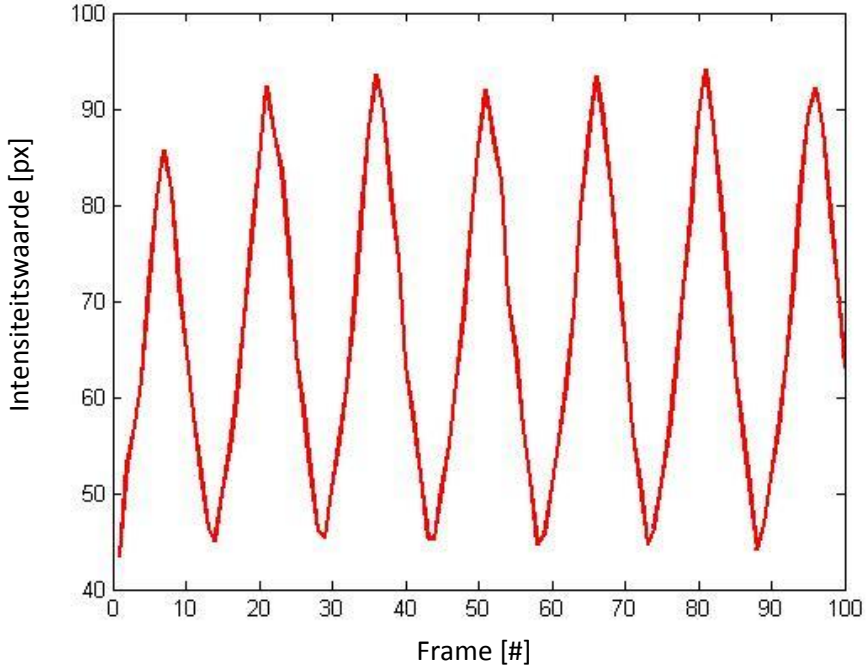
Omdat de driver van de cameramodule automatisch gebruikmaakt van auto adjust brightness is het zeer moeilijk in een omgeving met buitenlicht te bepalen of een verandering van intensiteitswaarden door de leds worden veroorzaakt of door de driver zelf.

Figuur 17 geeft het ingrijpen van de auto adjust brightness weer in een frame met buitenlicht waar de globale intensiteitswaarde wordt gemeten. De pieken en dalen op de eerder linieare gedeeltes van de grafiek zijn de verandering in intensiteit van de leds. De grote piek in het midden is de verandering van auto adjust brightness. In een gecontroleerde ruimte mag deze piek en dus het ingrijpen van de auto adjust brightness niet voorkomen.



Figuur 17: Intensiteitswaarde per frame van buitenlicht waarop auto adjust brightness ingrijpt

Figuur 18 geeft de globale intensiteit per frame weer van een gecontroleerde ruimte. Hierbij zijn de pieken enkel te wijten aan de verandering van intensiteit van de leds. De globale intensiteit is veel kleiner in de donkere gecontroleerde ruimte. Merk op dat hier de lineairiteit duidelijk zichtbaar is.



Figuur 18: Intensiteitswaarde per frame van gecontroleerde ruimte zonder auto adjust brightness

Enkel de tijdsbepaling van de bovenstaande leds zijn belangrijk, immers de frame begint met opnemen aan de bovenkant. Door een gemiddelde begintijd uit de intensiteitswaarden van de vier leds te halen, is het mogelijk in functie van de globale tijd te weten wanneer de frame opgenomen is. Daarom moet telkens het vermogen per led worden doorgestuurd samen met de sensorwaarden van de MPU-6050. Nu kunnen de sensordata in functie van de tijd aan een intensiteitswaarde van de leds gelinkt worden per frame.

De gecontroleerde ruimte is, zoals eerder vermeld, een donkere kamer waarbij enkel de leds voor een verandering van intensiteit zorgen. Indien in deze gecontroleerde ruimte al geen goede benadering van tijdsbepaling mogelijk is, zal dit zeker niet lukken bij een ongecontroleerde ruimte met een tegenactie van auto adjust brightness.

Tabel 3 geeft een aantal random frames weer waarbij de tijsbepaling in functie van de intensiteit wordt bepaald. De tijdsbepaling van alle vier de bovenstaande respectievelijk onderstaande leds moeten allemaal dezelfde waarde aangeven voor een gemeten intensiteit (niet alle acht dezelfde waarde mits traagheid van de beeldsensor). Indien er niet wordt gekeken naar leds 4 (slecht in beeld), verschillen de tijdsbepalingswaarden zeer hard van elkaar. Zo zijn er waarden die tot meer dan 30ms van elkaar verschillen. Dit is langer dan de traagheid van de beeldsensor, wat betekent dat er zelfs 2 frames in dezelfde berekende tijdswaarde kunnen optreden. De tijdsbepaling is niet accuraat genoeg in de gecontroleerde ruimte.

De framerate bedraagt 15 fps wat betekent dat er per frame een tijd van 66,67ms tussenzit. Ook hier moet het verschil in tijd van de opeenvolgende frames rond deze waarde liggen. Bij de eerste en laatste framewaarde ligt het verschil in tijd onder de 66ms en bij de tweede framewaarde boven de 66ms.

Tabel 3: Tijdsbepaling van intensiteitswaarde op random achtereenvolgende frames

FRAME 56 EN 57:	BOVENSTAANDE LEDS				ONDERSTAANDE LEDS			
56. INTENSITEIT [PX]:	**319**	**383**	**363**	**304**	**151**	**146**	**188**	**174**
TIJDSBEPALING [MS]:	94,57	60,60	84,58	196,07	25,54	22,61	51,23	43,25
57.	**176**	**236**	**219**	**158**	**116**	**117**	**137**	**131**
	14,07	17,57	23,68	39,64	0,00	0,00	11,64	0,00
FRAME 34 EN 35:	**Bovenstaande Leds**				**Onderstaande Leds**			
34. INTENSITEIT [PX]:	**852**	**1433**	**1047**	**427**	**663**	**602**	**622**	**492**
TIJDSBEPALING [MS]:	394,60	368,01	373,83	327,86	399,10	378,21	388,13	363,14
35.	**1047**	**1700**	**1193**	**448**	**698**	**654**	**657**	**507**
	504,37	446,18	435,57	350,36	424,64	418,76	415,30	378,22
FRAME 89 EN 90:	**Bovenstaande Leds**				**Onderstaande Leds**			
89. INTENSITEIT [PX]:	**318**	**388**	**360**	**298**	**197**	**179**	**223**	**191**
TIJDSBEPALING [MS]:	94,01	62,07	83,31	189,64	59,10	48,35	78,40	60,36
90.	**404**	**507**	**429**	**324**	**286**	**241**	**291**	**243**
	142,42	96,91	112,49	217,50	124,03	96,70	131,19	112,66

Omdat bij het meten van de intensiteit de shutterspeed ook een rol speelt, wordt op het laatst mogelijke moment, wanneer het rolluik dus sluit, de hoogste of laagste intensiteit gemeten. Dit hangt af van de stijgende of dalende intensiteit van de leds. Dit komt dus overeen met het einde van de opname van het frame. Aangezien het begin van het opnemen van de frame nodig is, en de shutterspeed niet bekend is, kan er nog steeds geen beginpunt van het frame worden bepaald. Indien de traagheid van de beeldsensor dezelfde waarde behoudt (inclusief de shutterspeed), is het wel mogelijk maar is het meten van de intensiteitswaarde nog steeds niet accuraat genoeg.

Indien drift compensatie met tijdsbepaling via intensiteitswaarde wordt toegepast, waarbij de intensiteitswaarde van het rollend licht verandert, zal er per frame ongeveer dezelfde accuraatheid van tijdsbepaling verkregen worden. Buiten de gecontroleerde ruimte zal opnieuw de auto adjust brightness voor dezelfde problemen zorgen en drift zal blijven voorkomen na een bepaalde tijd.

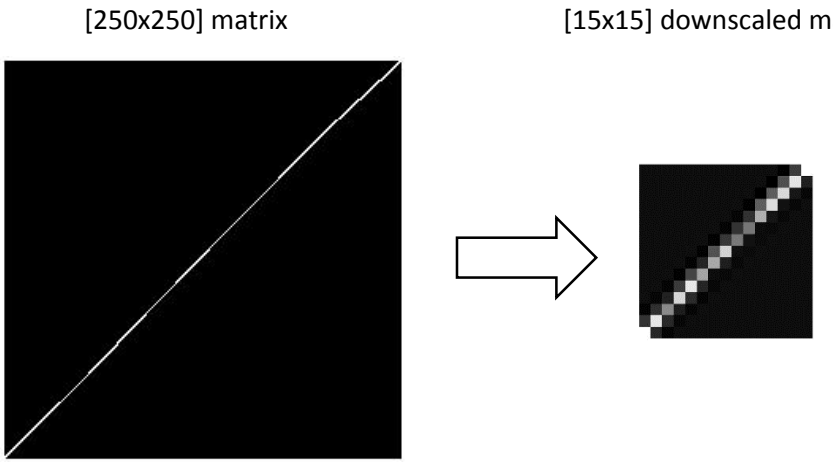
7. Onderzoek voor toepassing van bewegingspad uit sensordata

7.1. Bewegingspad bepalen uit sensordata van MPU-6050

Voor deconvolutie wordt toegepast, moet een bewegingspad omgevormd worden naar een beeld van grijswaarden. Hierbij moet de som van het bewegingspad gelijk zijn aan 1. Zij bepalen namelijk in welke mate pixels van het origineel beeld doorwegen bij deconvolutie. Het bewegingspad wordt opgesteld in een vooraf gedimensionaliseerde matrix waarbij de sensordata, omgerekend naar snelheid (7), als relatieve punten op de matrix worden aangeduid. Tussen de datapunten worden vervolgens rechte lijnen getrokken. Indien de grootte van de matrix te klein wordt genomen, zal de nauwkeurigheid van het bewegingspad dalen. Dit omdat de lijnen tussen de sensordatapunten niet nauwkeurig genoeg kunnen getekend worden. Andersom heeft een veel te grote matrix weinig zin. Het bewegingspad zal niet fel afwijken naarmate er een grotere gedimensionaliseerde matrix wordt gekozen. Voor het reconstrueren van het datapad wordt een matrix van 500x500 gebruikt.

Bij het tekenen van het bewegingspad is het belangrijk om in het midden van de vooropgestelde matrix te beginnen, immers een opwaartste beweging is positief en een neerwaartse beweging negatief. De helft van de matrix wordt gebruikt als er enkel een positieve beweging plaatsvindt. Het is dus mogelijk dat hierdoor niet de hele matrix benut wordt voor het datapad. Dit heeft geen nadelig effect immers deze zwarte stukken hebben als waarde nul en worden niet gebruikt. Er kan wel een probleem onstaan als de matrix wordt gelinkt aan een breedte in pixels, omdat niet de matrix maar het bewegingspad gelinkt moet worden aan een breedte in pixels. Tot slot wordt het bewegingspad gedownscaled naar een formaat afgeleid uit de maximale verplaatsing in de x- en y-richting.

Figuur 19 geeft het bewegingspad van een naar omhoog gaande schuine rechte weer in een matrix van 250x250 en het gedownscaled formaat van 15x15 ervan. Dit gedownscaled formaat wordt gebruikt als bewegingspad waarbij de grootte van het formaat afhangt van de maximale verplaatsing van pixels in de met handshake blur genomen foto. Deze waarde kan automatisch afgeleid worden van de sensorwaarden nadat deze eenmalig is bepaald per camera. De intensiteit van de pixels in het bewegingspad bepalen welke waarde van pixels meer representatief waren tijdens de beweging bij het nemen van de foto.



Figuur 19: PSF van 13 accelerometersensorwaarden

Het is niet mogelijk in de gebruikte Hercules Silver Edition cameramodule om met behulp van een exif-tool onderliggende data van een foto te achterhalen zoals de shutterspeed, ISO-waarde of diafragma. In de meeste gevallen kan men met een exif-tooltje de shutterspeed en dergelijke achterhalen omdat deze embedded in de foto zitten als extra informatie.

Camera:	Canon EOS 450D
Lens:	Canon EF 22-55mm f/4-5.6 USM Shot at 49 mm
Exposure:	Auto exposure, Program AE, 1/64 sec, f/5.6, ISO 400
Flash:	On, Fired
Focus:	One-shot AF, with a depth of field of from 0 m to infinity. AF Area Mode: Auto
Date:	**December 1, 2014** 9:50:33PM (timezone not specified) (1 year, 16 days, 2 hours, 10 minutes, 48 seconds ago, assuming image timezone of US Pacific)
File:	**3,088** × **2,056** JPEG (**6.3** megapixels) 1,939,189 bytes (1.8 megabytes)

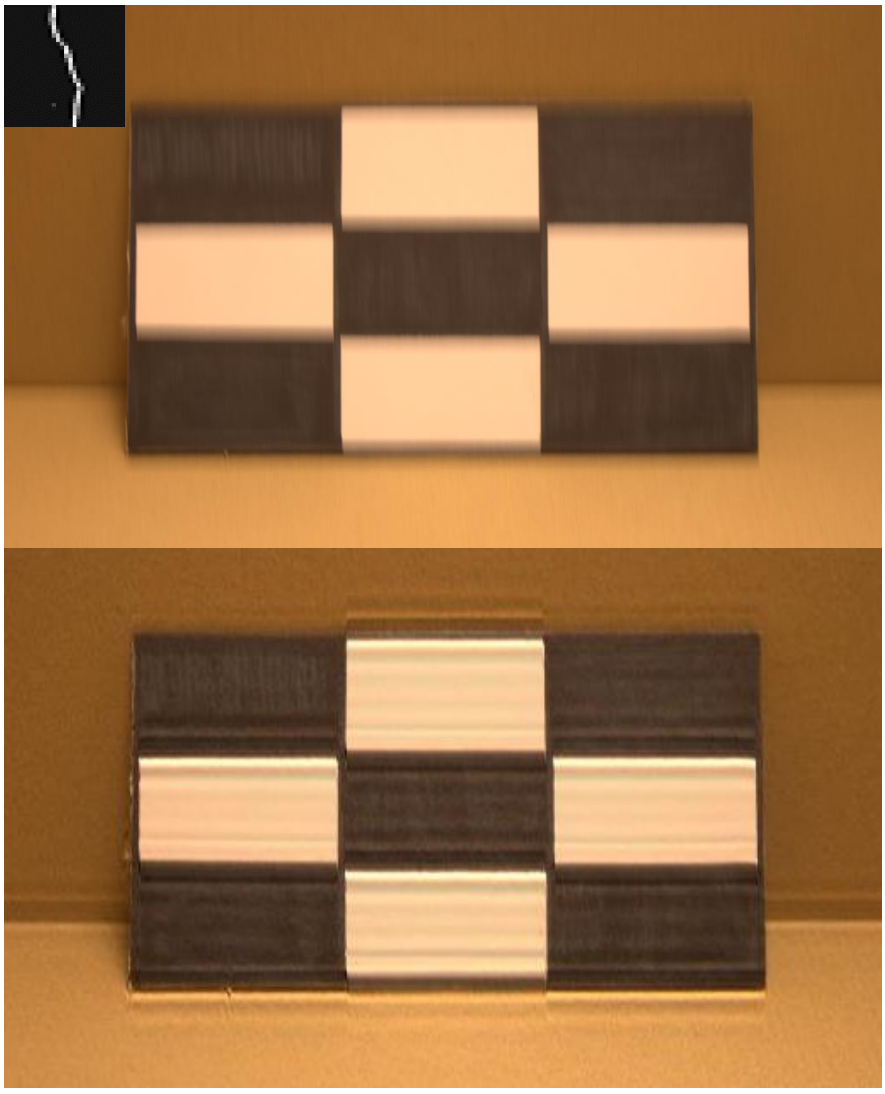
Figuur 20: Exif-tool voor het bepalen van onderliggende data in een foto

Indien de shutterspeed bekend is kan met behulp van de traagheid van de sensor een aantal sensorwaarden gelinkt worden aan een aantal pixels van het frame. Zo kan er, omdat er gebruik wordt gemaakt van een rolling shutter, een pixelband worden opgesteld die overeenkomt met een aantal datapunten. Let op dat de traagheid van de beeldsensor ook weer afhankelijk is van de shutterspeed. Immers bij een grotere shutterspeed van bijvoorbeeld 500ms zal de sensor al zeker 500ms lang belicht worden plus de extra tijd nodig om de rolluiken open en dicht te trekken.

Omdat in de cameramodule geen shutterspeed bekend is, wordt het testen van de deconvolutie toegepast op het beeld van een spiegelreflexcamera. De camera maakt gebruik van een mechanische rolling shutter waarbij de traagheid van de beeldsensor in de orde van 1ms ligt. De snelheid van de shutterspeed bepaalt hoeveel datapunten er kunnen genomen worden. Een shutterspeed van 1/10sec komt overeen met een beeld dat wordt opgenomen in een tijdspanne van 100ms. Omdat de traagheid van de beeldsensor ten opzichte van de shutterspeed heel klein is, worden alle datapunten voor het frame gebruikt. De traagheid van de beeldsensor is sneller dan de tijd waarin een sensordatapunt wordt uitgelezen. Het bewegingspad in het gebruikte voorbeeld is opgesteld uit 47 sensordatapunten en is achterhaald door binnen een tijdspanne een foto te nemen en daaruit iteratief te bepalen welke sensorwaarden het meest representatief zijn.

7.2. Deconvolutie – Richardson-Lucy algoritme & Wiener Filter

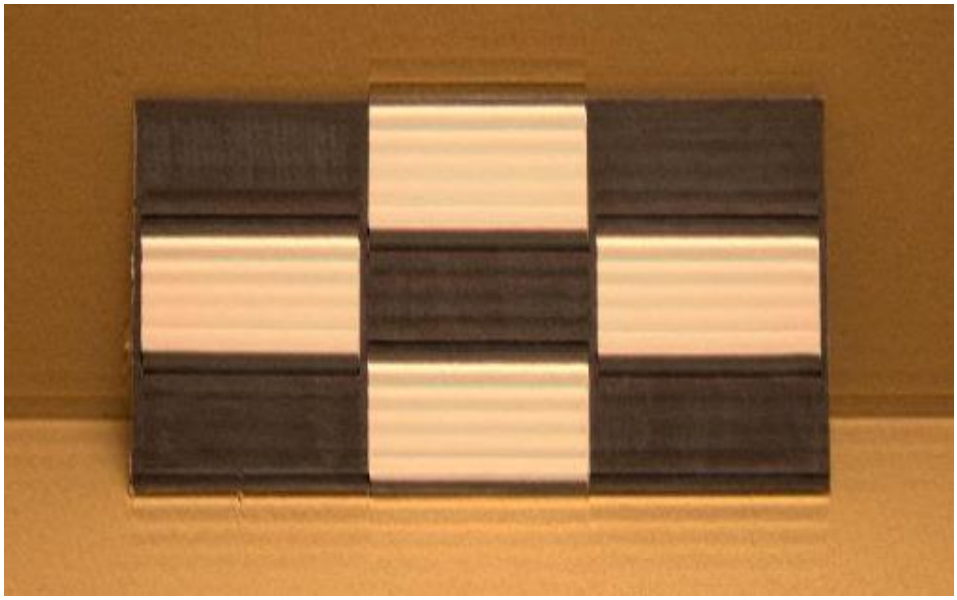
Als eerste wordt het Richardson-Lucy algoritme toegepast. Dit algoritme geeft een zeer goede benadering indien de ruis van de foto niet gekend is. Figuur 21 geeft de originele foto met bewegingspad en het resultaat van het algoritme weer. Het bewegingspad heeft een hoogte van 85 pixels en een breedte van 8 pixels. Als controlebeeld is er een foto genomen met een Canon EOS 700D met dimensies 5208x3476, shutterspeed 1/10sec, diafragma f/5.6 en ISO 200. De foto werd genomen met een lichte handshake blur.



Figuur 21: Richardson-Lucy algoritme (bovenstaand: origineel, onderstaand: deconvolutieproduct)

In het resultaat is het bekomen beeld duidelijk verscherpt en zijn accenten meer zichtbaar op de foto. De overgang van voornamelijk de zwart-wit kleuren, geeft wel een ringingeffect weer. Het Richardson-Lucy algoritme heeft in totaal 390 seconden de tijd nodig gehad voor image processing, wat het een behoorlijk tijdsafhankelijk algortime maakt. Daarom is er gezocht naar een alternatieve deconvolutiemethode genaamd de Wiener filter. Een Wiener filter wordt gebruikt in de signaalverwerking om een schatting te maken van een willekeurig proces door lineaire tijdsinvariante filtering toe te passen. Dit soort filter wordt in de praktijk ook het vaakst gebruikt voor de reconstructie van een blurry foto. Merk wel op dat dit enkel voor meer lineaire bewegingen geldt. Indien de shutterspeed klein is, zal een beweging voornamelijk lineair geneigd te zijn. Figuur 22 geeft het effect weer op hetzelfde controlebeeld.

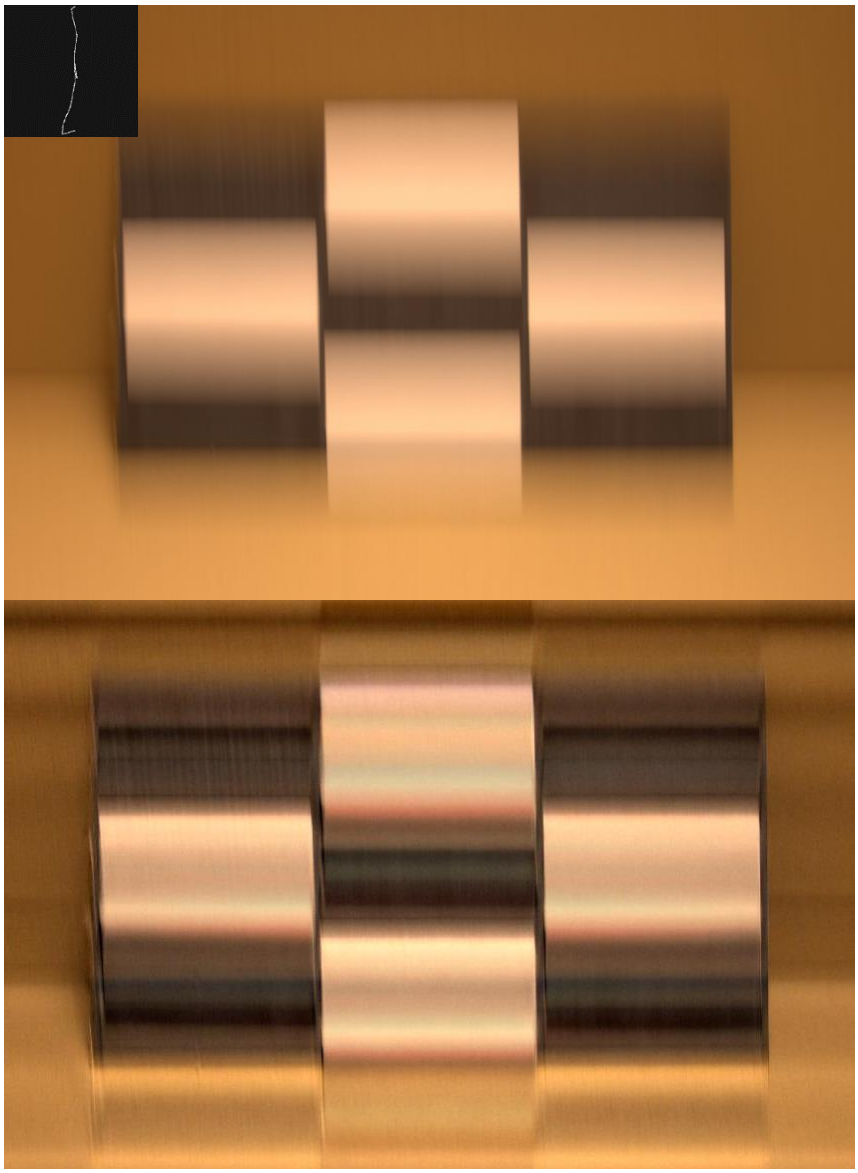
Omdat het Wiener filter algoritme een schatting nodig heeft van de hoeveelheid ruis aanwezig in het beeld, wordt er extra gekende ruis aan toegevoegd. Als resultaat bekomt men opnieuw een verscherpt beeld met duidelijkere accenten, maar met een iets groter ringingeffect. De tijd die nodig is voor dit algoritme is echter beduidend kleiner en bedraagt 29 seconden.



Figuur 22: Wiener filter algoritme (deconvolutieproduct)

Ter illustratie wordt er ook een zware handshake blur foto verscherpt. Het controlebeeld is een foto genomen met dimensies 5208x3476, shutterspeed 1/10sec, diafragma f5.6 en ISO 200. Op figuur 22 wordt de originele foto met bewegingspad en het Wiener-Filter algoritme getoond als resultaat. Het bewegingspad heeft een hoogte van 405 pixels en een breedte van 34 pixels.

Indien er zwaardere handshake blur optreedt zal het resultaat een meer globaal beeld geven van het voorwerp of persoon in beeld, details daarentegen zullen nauwelijks zichtbaar zijn. Het ringingeffect is in dezelfde maat aanwezig, enkel groter in dimensie. De grootte van het ringingeffect hangt dus af van de grootte van het bewegingspad dat is gebruikt voor deconvolutie.



Figuur 23: Wiener filter algoritme (bovenstaand: origineel, onderstaand: deconvolutieproduct)

Het onderzoek met het MAP-estimatie blind deconvolution algoritme van Matlab is niet verder uitgewerkt. De intentie was om een vergelijkende deconvolutietijd te bepalen voor een blind deconvolution algoritme, maar na een half uur rekentijd met een iteratie van 10 stappen werd er nog steeds geen verscherpt beeld bekomen. Uiteindelijk is het algoritme stopgezet omdat het systeem vastliep en is hier dus geen exacte tijdsbepaling van, buiten het feit dat het algoritme langer dan 30 minuten nodig heeft.

Tabel 4 geeft een overzicht van de snelheden van de deconvolutiemethode bij verschillende beeldformaten met eenzelfde grootte van bewegingspad van 50x50 en éénzelfde foto, aangepast naar de juiste formaten. Momenteel zijn de meest gebruikte formaten voor video-opname 720p (HD-formaat) en 1080p (Full HD-formaat). Er wordt binnenkort wel overgeschakeld naar 2160p (4k-formaat). Het convolutie algoritme wordt gerund in Matlab met aan boord een Intel Core i5 M480 @2.67GHz van het jaar 2011Q1.

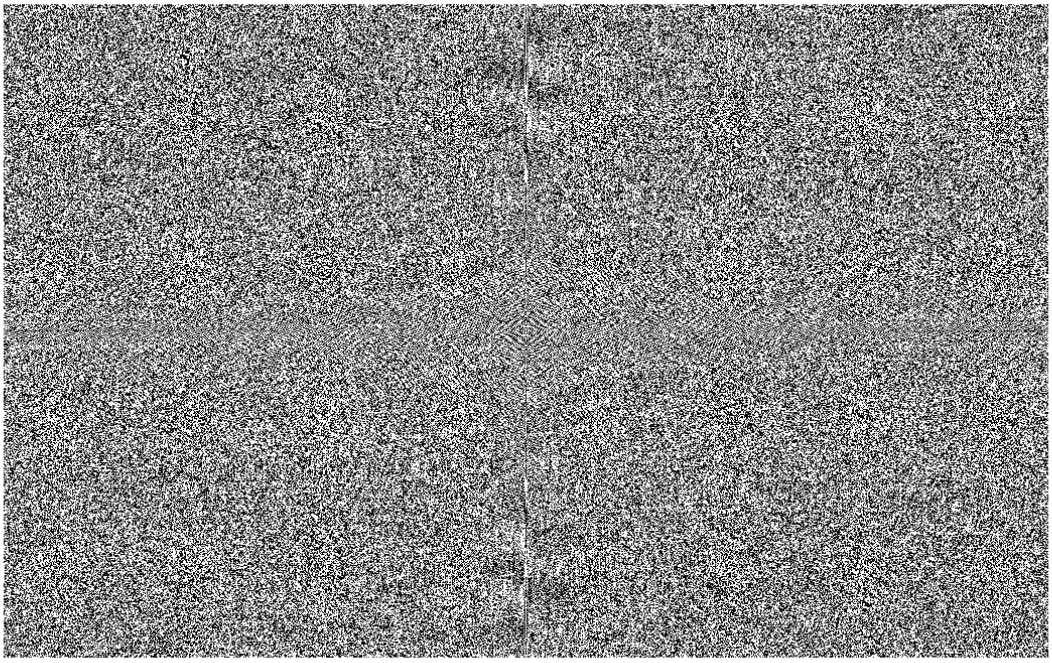
Tabel 4: Convolutietijd van Richardson-Lucy en Wiener Filter algortime op verschillende beeldformaten

	RICHARDSON-LUCY [SEC]	**WIENER FILTER [SEC]**
[640X480]	5,25	0,60
[1280X720] 720P	11,8	1,60
[1920X1080] 1080P	24,6	3,25
[2560X1440] 1440P	55,1	5,80
[3840X2160] 2160P	138,4	11,80

Indien er een krachtige processor aan boord is voor deconvolutie, is het mogelijk om real-time aan motion-compensatie te doen door gebruik te maken van de Wiener Filter zonder extra filtering. Het beeldformaat is dan echter wel zeer klein en behoort niet toe aan de standaard van vandaag.

7.3. Onderzoek ringingeffect na deconvolutie

Dit ringingeffect lijkt in eerste instantie een aantal horizontale balken die over het gehele beeld liggen en enkel duidelijker zichtbaar worden bij de overgang van kleuren. Er zou gedacht kunnen worden dat dit een soort van periodische ruis is. Dit is echter niet het geval als er gekeken wordt naar de Fourrier getransformeerde van dit beeld in grijswaarde. Indien er periodische ruis aanwezig is, zouden er sterretjes in het beeld moeten liggen die de frequentie van de ruis weergeven. Dit is echter niet het geval. Het middelste sterretje is de dc-component (0Hz) van het beeld en heeft geen belang. Figuur 23 geeft de fourrier getransformeerde van de met lichte handshake blur genomen foto.



Figuur 24: Fourrier getransformeerd deconvolutieproduct voor bepaling van periodische ruis

Omdat het ringingeffect voornamelijk voorkomt bij de overgang van kleuren kan er ook gekeken worden naar de edge detectie. Edge detectie zijn filters die de overgang van kleuren weergeven. In Matlab bestaan er hier een verschillend aantal filters voor. Hierdoor kan het ringingeffect al gedeeltelijk worden afgezonderd indien er enkel apart hierop een filter wordt toegepast. Omdat niet alle overgangen van kleuren te wijten zijn aan het ringingeffect, moet er een onderscheid gemaakt kunnen worden tussen een gewenste en een ongewenste kleurenovergang.

In dit voorbeeld werd er enkel een horizontaal ringingeffect verkregen omdat het bewegingspad dit veroorzaakte. Een complexer bewegingspad zorgt voor een ander soort ringingeffect, maar is steeds achterhaalbaar door edge detectie. Verder onderzoek naar dit ringingeffect is nog nodig voor het extra verscherpen van het beeld.

8. Besluit van het onderzoek

Als eerste werd onderzocht wat de traagheid van de beeldsensor van de cameramodule was. Deze is belangrijk om te weten hoeveel sensordatapunten van de MPU-6050 nodig zijn voor het construeren van het bewegingspad. Hierbij werd de verandering van intensiteit getest in een gecontroleerde ruimte, dit omdat extra instellingen zoals shutterspeed niet gekend waren. De traagheid van de beeldsensor in de gecontroleerde ruimte bedroeg 28 milliseconden. Vervolgens werd de snelheid per datapunt van de MPU-6050 uitgelezen door de Arduino. Deze bedroeg 2,124 milliseconden, wat neerkomt op een 13 datapunten per beeld.

De volgende stap hield in om van de sensordatapunten een bewegingspad op te stellen voor deconvolutie. De acceleratie van de data wordt gemeten door accelerometers. Deze data worden omgezet naar een snelheid en afgelegde weg om in een vooraf gedimensionaliseerde matrix geplaatst te worden, relatief ten opzichte van de maximum waarde van de x-richting en de y-richting. Vervolgens wordt een lijn tussen de datapunten getrokken, beginnend bij het nulpunt in het midden van de matrix.

Als tweede werd er onderzocht of het mogelijk is om twee aparte systemen, namelijk de cameramodule en de Arduino met de MPU-6050, samen te laten werken met een juiste timing. Er werd onderzocht of de software van Matlab snel genoeg een gezamenlijk startsignaal kon sturen naar de Arduino en de camera module. Dit werd visueel getoond als een lopend licht in de video. Elke keer een frame werd opgevraagd, ging het volgende ledje branden. De resultaten kwamen echter helemaal niet overeen met een lopend licht. Vervolgens werd er gezocht naar een manier om een timing te creëren in de Arduino die gelijk loopt met de framerate van de native software van de cameramodule. Uiteindelijk kon, met behulp van driftcompensatie, het lopend licht van frame tot frame gevolgd worden. Dit was echter maar goed voor een aantal frames om uiteindelijk terug beginnen achter of voor te driften. Ook kon er niet precies bepaald worden wanneer de frame werd opgehaald binnen de tijd van de led. Als laatste werd de verandering van de intensiteitswaarden gebruikt voor een tijdsbepaling van een frame. Hierbij werd de verandering van intensiteit van de led achterhaald in functie van de tijd, om dit vervolgens te koppelen aan een frame. Resultaten van een aantal random frames gaven aan dat de techniek niet accuraat genoeg is om toe te passen.

Het is zeer belangerijk om een gezamenlijk startsignaal te verkrijgen voor beide systemen die afhankelijk zijn van elkaar. Bij een digitale spiegelreflexcamera, waar de shutterspeed en dergelijke wel gekend zijn, is het ook niet mogelijk om een gezamenlijk startsignaal te creëren voor video. Er zijn immers geen signalen die van buitenaf uit de camera gehaald worden die vertellen wanneer precies een frame wordt opgehaald. De enige accurate optie is het zelf aansturen van een sensor op hardwareniveau waar er zelf bepaald wordt wanneer een nieuwe frame wordt opgehaald. Verder onderzoek naar deze techniek is nog nodig.

Als laatste werd het Richardson-Lucy algoritme en het Wiener Filter algoritme toegepast voor deconvolutie. Bij beide algoritmen is het resulterend beeld duidelijk verscherpt met een nadelig toegevoegd ringingeffect. Gemiddeld genomen ligt de computatie tijd van een 1080p beeld voor het Richardson-Lucy algoritme rond de 24,60 seconden en voor het Wiener Filter algoritme rond de 3,25 seconden. Het Wiener Filter algoritme is beduidend sneller in deconvolutie, maar is meer onderhevig aan het ringingeffect doordat er een gekende ruis aan het frame moet worden toegevoegd.

Na deconvolutie is onderzocht of het ringingeffect mogelijk verwijderbaar is door te kijken naar de fourrier getransformeerde van dit beeld in de veronderstelling dat het ringingeffect ontstaat door periodische ruis. Dit is echter niet het geval. Vervolgens is er onderzocht of er via edgde-detectie filtering het ringingeffect kon geïsoleerd worden. Er werd echter niet alleen het ringing effect geïsoleerd. Verder onderzoek voor het scheiden van het ringing effect is nog nodig.

Voor het toepassen van real-time videocompensatie is de snelheid van de rekenkracht belangrijk. Het Wiener Filter algoritme is beduidend sneller dan het Richardson-Lucy algoritme, maar niet snel genoeg voor video compensatie met de huidige standaard.

Bibliografie

[1]: Anat Levin, e. a. (sd). Understanding and evaluating blind deconvolution algorithms. *Adobe*, 8.

[2]: Baek, J. (2012). Computational Photography. *CS 478 Lecture*, 83.

[3]: Baek, J. (2012). Computational Photography Image Stabilization. 83.

[4]: Caspe, B. (sd). How an Electronic Shutter Works. 6.

[5]: Communications, A. (2014). Image stabilization - improving camera usability. 6.

[6]: DigitalRev. (2008, 10 10). *DigitalRev*. Opgehaald van http://www.digitalrev.com/article/lens-vs-sensor-shift-image/MzU4Mg_A_A

[7]: Embedded, I. (sd). Optical Image Stabilization (OIS). *Rohm Semiconductor*, 13.

[8]: Hammann, J. T. (1992). Image Stabilization Performance Optimization Using An Optical Reference Gyroscope. *University of Colorado*, 79.

[9]: Karpenko, A. (2011). Digital Video Stabilization and Rolling Shutter Correction using Gyroscopes. *Stanford University*, 7.

[10]: Neel Joshi, e. a. (sd). Image deblurring using inertial measurement sensors. *ACM SIGGRAPH*, 8.

[11]: Rob Fergus, e. a. (sd). Removing Camera Shake from a Single Photograph. *University of Toronto*, 8.

[12]: Truyen, N. (sd). Image deblurring. *UHasselt*, 61.

[13]: Kotera Jan, e. a. (2011). Blind deconvolution using alternating maximum a posteriori estimation with heavy-tailed priors. *ucsc*, 8.

[14]: Yu-Wing Tai, e. a. (2010). Richardson-Lucy deblurring for scenes under a projective motion path. IEEE, 15.

[15]: Levin, A (2011). Understanding blind deconvolution algorithms. IEEE, 13

[16]: InvenSense Inc. *Alldatasheet.* Opgehaald van http://pdf1.alldatasheet.com/datasheet-pdf/view/517744/ETC1/MPU-6050.html

Bijlagen

A. [Matlab]: Ophalen data Arduino via COM

```
Clc; close all; clear all; delete(instrfindall);

%% Werkt enkel samen met automatisch verzenden van sensorwaarde
%% Initialisatie voor serialObject COM1 - Arduino
disp('Initialisiatie en dumpflush van sensorwaarde')
serialObject = serial('COM1', 'BaudRate', 115200);
fopen(serialObject);

% Flush voor zekerheid en goede startwaarde
% Indien timeout, hard reset Arduino
dummyflush = 0;
while dummyflush < 5
    % 49 verzenden als startsignaal (In Arduino COMpoort '1' verzenden)
    fwrite(serialObject, 49);
    text = fgets(serialObject);
    dummynum = str2num(text);
    dummyflush = dummyflush +1;
end
% % Extra dummyflush voor geschreven waarden te verwijderen
% dummyflush = 0;
% while dummyflush < 5
%     text = fgets(serialObject);
%     dummynum = str2num(text);
%     dummyflush = dummyflush +1;
% end
disp('Dumpflush succesvol')

%% Initialisatie loop communicatie Arduino
% Create matrix uit datapunten van x,y groot
% nFrames is het totaal aantal punten dat moet worden binnengehaald, hangt
% af van de lengte van de video
fCount = 1;
nFrames = 1000;

lengte = length(dummynum);
lengte = (lengte/2 -1)

x = zeros(lengte, nFrames);
y = zeros(lengte, nFrames);
disp('Start binnenhalen data...')

while fCount < nFrames
    text = fgets(serialObject);
    array = str2num(text); %str2double is sneller maar geeft NaN
    i = 1;
    j = 1;

    while i < (lengte *2)
        x(j, fCount) = array(i);
        y(j, fCount) = array(i+1);
        i = i +2;
        j = j +1;
    end
    fCount = fCount +1;
end

% 48 verzenden als stopsignaal (In Arduino COMpoort '0' verzenden)
fwrite(serialObject, 48);
delete(instrfindall);
```

B. [Matlab]: Meten intensiteitswaarde leds

```
Clc; clear; close all

%% Openen van video en opslaan in nieuw video na integratie met sensordata
video = VideoReader(filename.extension');
writerObj = VideoWriter('newfilename.avi');
open(writerObj);

%% Ophalen data video
nFrames = video.NumberOfFrames;
for k = 1 : nFrames
    vid(k).cdata = read(video, k);
end

grootte = vid(1);
[h, w] = size(grootte.cdata(:,:,1));
imhist(grootte.cdata(:,:,1));

%% Initialisatie waarde voor loop
% Beginnend van links naar rechts, waardes van de leds
iUp1 = 0;iUp2 = 0;iUp3 = 0;iUp4 = 0;
iDo1 = 0;iDo2 = 0;iDo3 = 0;iDo4 = 0;

first = 0;
first_frame = 0;
q = 1;
ibw = zeros(h, w);

disp('Start conversie intensiteit')
% Tresholdwaarde voor bepaling minimum intensiteit
inputValue = 100;

%% Intensiteitswaarde halen uit data van video per frame
for z = 1 : nFrames
    u = vid(z);
    % imshow(u.cdata), pause(0.5); %

    red = u.cdata(:,:,1);
    I = u.cdata;
    % autoAdjust(z) = mean(mean(sum(I,3))); % Voor bepalen van globale
intensiteit

    i = 1;
    j = 1;
    while i < h
        while j < w
            iValue = red(i, j);
            if (iValue > inputValue)
                ibw(i, j) = 255;

                %% Bepalen intensitent waarde led voor timing
                % Intensiteit aparte led bovenkant - aangepast beeldformaat
                if (i < h/4.8)
                    if (j < w/4)
                        iUp1 = iUp1 +1;
                    elseif (j > w/4  &&  j < w/2)
                        iUp2 = iUp2 +1;
                    elseif (j > w/2 && j < (3*w/4))
                        iUp3 = iUp3 +1;
                    elseif (j > (3*w/4))
```

```
                    iUp4 = iUp4 +1;
                end
            % Intensiteit aparte led onderkant - aangepast beeldformaat
            elseif (i > (3.6*h/4.8))

                if (j < w/4)
                    iDo1 = iDo1 +1;
                elseif (j > w/4  &&  j < w/2)
                    iDo2 = iDo2 +1;
                elseif (j > w/2 && j < (3*w/4))
                    iDo3 = iDo3 +1;
                elseif (j > (3*w/4))
                    iDo4 = iDo4 +1;
                end
            end

            %% Bepalen startwaarde voor meting (indien nodig)
            sw = 500;
            if (first == 0 && (iUp1+iDo1 > sw || iUp2+iDo2 > sw ||
iUp3+iDo3 > sw || iUp4+iDo4 > sw))
                first_frame = z
                first = 1;
            end

        end
        j = j +1;
    end
    % Breedte resetten
    j = 1;
    i = i +1;
end

%% Puur visueel voor achterhalen of led's in juiste verdeling zit
k = 1;
while k < h
    ibw(k, w/4) = 255;
    ibw(k, w/2) = 255;
    ibw(k, (3*w/4)) = 255;
    k = k +1;
end

%% Tonen van resultaat en herinitialisatie
% Pas toepassen zodra de eerste frame gevonden is waar de ledjes
branden
% q gebruiken voor niet op random getal te beginnen in matrix, q init
uit loop
if (first == 1)
    cUp1(q) = iUp1;
    cUp2(q) = iUp2;
    cUp3(q) = iUp3;
    cUp4(q) = iUp4;

    cDo1(q) = iDo1;
    cDo2(q) = iDo2;
    cDo3(q) = iDo3;
    cDo4(q) = iDo4;

    q = q +1;
end
% Herinitialiseren, anders wordt alles opgeteld bij elkaar
% imshow(ibw);pause(0.3);
iUp1 = 0;iUp2 = 0;iUp3 = 0;iUp4 = 0;
```

```
    iDo1 = 0;iDo2 = 0;iDo3 = 0;iDo4 = 0;
    ibw = zeros(h, w);
end

disp('Intensiteit conversie gestopt')
Aantal_geconverteert = q
disp('Curvefitting gestart')

%% Plotten en zoeken van peakwaardes dal en top
% Top en dal peaks
pks = mean(findpeaks(cUp1));
ipks = mean(findpeaks(cUp1 * -1) *-1);

% Plotten
figure, plot(cUp1, 'r-'); hold on;
plot(cUp2, 'g-'); hold on;
plot(cUp3, 'b-'); hold on;
plot(cUp4, 'k-');
title('Leds bovenaan: 1. Rood 2. Groen 3. Blauw 4. Zwart');
xlabel('Frame'); ylabel('Intensiteit');

% figure, plot(autoAdjust, 'r-', 'linewidth',2)
figure, plot(cUp1(10:50), 'r-','linewidth',2)
hold on, plot(cUp2(10:50), 'g-','linewidth',2)
hold on, plot(cUp3(10:50), 'b-','linewidth',2)
hold on, plot(cUp4(10:50), 'k-','linewidth',2)
figure, plot(cDo1, 'r-'); hold on;
plot(cDo2, 'g-'); hold on;
plot(cDo3, 'b-'); hold on;
plot(cDo4, 'k-');
title('Leds onderaan: 1. Rood 2. Groen 3. Blauw 4. Zwart');
xlabel('Frame'); ylabel('Intensiteit');


%% Wegschrijven data naar excel voor verder onderzoek
% Aangepaste excelfile met cUp1,… als tabblad
disp('Data wegschrijven excel...')
filename = 'filename.xlsx';
xlswrite(filename,cUp1,'cUp1','D1')
xlswrite(filename,cDo1,'cUp1','D2')
%
xlswrite(filename,cUp2,'cUp2','D1')
xlswrite(filename,cDo2,'cUp2','D2')
%
xlswrite(filename,cUp3,'cUp3','D1')
xlswrite(filename,cDo3,'cUp3','D2')
%
xlswrite(filename,cUp4,'cUp4','D1')
xlswrite(filename,cDo4,'cUp4','D2')
disp('Done.')
```

C. [Matlab]: Reconstrueren van bewegingspad uit sensordata

```
close all; clear all; clc

%% Data handmatig ingevoerd voor testfile
x = [data van bewegingspad hier invullen];
y = [data van bewegingspad hier invullen];

%% Instellingen aanpassen voor matrix
% maxlen = grootte van de matrix,
% scale = grootte van bewegingspad indien handmatig invoer
maxlen = 500;
scale = [50 50];


%% Grootte bepalen van het aantal sensorwaarden
matlen = max(max(size(x), size(y)));

% Relatieve afstand bepalen door absolute maximale waarde te achterhalen
xmax = max(abs(max(x)),abs(min(x)));
ymax = max(abs(max(y)),abs(min(y)));

% Breedte in pixels bepalen aan de hand van de afstand tussen minimale en
% maximale waarde
calibratiex = 178.716;
calibratiey = 159.101;

% Afstand van beweging in pixelsgrootte voor aanpassen grootte bewegingspad
% Elke camera heeft aparte waarden
xbewinpix = 20;
ybewinpix = 20;

% Grootte van bewegingspad na berekening van camera
xdiff = round(((xmax -min(x)) -calibratiex) /xbewinpix);
ydiff = round(((ymax -min(y)) -calibratiey) /ybewinpix);
%scale = [ydiff xdiff];

%% Initwaarde voor filters
% Enkel gebruikt voor horizontale benadering
hFilter = fspecial('prewitt');

%% Tekenen van sensordatapunten en lijnen in voorgedimensionaliseerde
matrix
% Init waarde
max2 = maxlen /2;

PSF1 = zeros(maxlen, maxlen);
PSF1 = im2double(PSF1);
xset = zeros(1, matlen); yset = zeros(1, matlen);

for i = 1:matlen
     xused = x(i);
     yused = y(i);

     xset(i) = max2 +round((max2 /xmax) *xused); % Aanpassen percentage
gewijs
     yset(i) = max2 +round((max2 /ymax) *yused);

     if i == 1
```

```
        % Source: http://stackoverflow.com/questions/2464637/matlab-
drawing-a-line-over-a-black-and-white-image
        xpts = linspace(max2, xset(i), 1000);
        ypts = linspace(max2, xset(i), 1000);
        index = sub2ind([maxlen maxlen], round(xpts), round(ypts));
        PSF1(index) = 1;
    elseif i >= 2
        xpts = linspace((xset(i -1)), xset(i), 1000);
        ypts = linspace((yset(i -1)), yset(i), 1000);
        index = sub2ind([maxlen maxlen], round(xpts), round(ypts));
        PSF1(index) = 1;
    end
end

%% Aanpassen bewegingspad en downscale bewegingspad
Iframe = imread('filename.extension');
figure, imshow(Iframe); title('Input beeld voor deconvolutie');

PSF2 = imcrop(PSF1, [min(yset) min(xset) max(yset) max(xset)]);
PSF2 = imrotate(PSF2, 90); % Indien x en y waarde zijn verwisseld, anders
comment
PSF5 = imresize(PSF2, scale);

%Aanpassen filter zodat som matrix 1 is
lighting = sum(sum(PSF5));
PSF5 = PSF5/lighting;



%% Algoritme van Richardson Lucy - weinig ringing maar nog wat blurry
% 398.5928 seconden voor little ** 464.6257 seconden voor big [5208x3476]
figure, imshow(PSF5, [], 'InitialMagnification', 'fit');

tic;
result = deconvlucy(Iframe, PSF5);
Deconvolucy_time = toc

figure, imshow(result); title('Outputbeeld na deconvolutie van Richardson
Lucy');

%% Algoritme van de Wiener filter - veel ringing maar scherp
% 28.9951 seconden voor little ** 30.6396 seconden voor big [5208x3476]
figure, imshow(PSF5, [], 'InitialMagnification', 'fit');

tic;
I = im2double(Iframe);
noise = 0.001;
mean = 0;

I = imnoise(I, 'gaussian', mean, noise);
snr = noise / var(I(:));
result = deconvwnr(Iframe, PSF5, snr); % 0.001 of snr gebruiken
Wiener_time = toc



figure, imshow(result); title('Outputbeeld na deconvolutie met Wiener
filter');

%% Extra filtering voor achterhalen perdiodic noise effect bij Wiener
filter
```

```
% Source: http://stackoverflow.com/questions/20531110/removing-pattern-and-
noise-in-an-image-using-fft-in-matlab
% I = rgb2gray(result);
% I = im2double(I);
% f = fftshift(fft2(I));
% figure, imshow(f)

%% Extra filtering (beta)
I = rgb2gray(result);
E = edge(I, 'sobel');

% Dilate edges
Ed = imdilate(E, strel('disk', 5));
Ed = repmat(Ed, [1 1 3]);

filter = fspecial('average', 10);
Ifilt = imfilter(result, filter);
result(Ed) = Ifilt(Ed);
% figure, imshow(result); title('Outputbeeld na deconvolutie en extra
filtering op ringing effect');
```

D. [Arduino]: Bepalen sensorwaarde en verandering van intensiteit leds

```
/*
Code voor het bepalen van de intensiteit van een ledje door gebruik te maken
van een pwm signaal. Via Matlab een tresholdwaarde bepalen voor intensiteit.
*/

#include<Wire.h>

const int MPU = 0x68;  // I2C address of the MPU-6050
int16_t Accel_X, Accel_Y, Accel_Z, Temp, Gyro_X, Gyro_Y, Gyro_Z;

// Analoge uitgangen van Arduino
const int led1 = 11;
const int led2 = 10;
const int led3 = 9;
const int led4 = 6;

signed int ledState1 = 0;
signed int ledState2 = 0;
signed int ledState3 = 0;
signed int ledState4 = 0;

// Tweemaal indien apart flikering wordt aangestuurd
signed int ledValue1 = 255;
boolean verwissel1 = false;
boolean switched1 = false;
signed int ledValue2 = 125;
boolean verwissel2 = false;
boolean switched2 = false;

// Interval tijd in micros voor verandering van leds
unsigned long previousMicros = 0;
const long interval = 1000;

// Minimumwaarde zichtbaar als intensiteitsverandering
const long minWaarde = 5;
unsigned int data = 0;

void setup() {
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);  // PWR_MGMT_1 register
  Wire.write(0);     // Slaapstand ontwaken (wakes up the MPU-6050)
  Wire.endTransmission(true);
  Serial.begin(115200); //zowel apparaatbeheer als hier aanpassen

  pinMode(led4, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led1, OUTPUT);
}

void loop() {
  /*
   Hier wordt de data van de sensor steeds opgehaald zo snel als dat de
Arduino dit aankan
  */
```

```
  if (Serial.available() > 0) {
    data = Serial.read();
    Serial.println(data);
  }

  if (data == 49) {
    unsigned long currentMicros = micros();
   // Moet er staan volgens Arduino Code, maar is sneller zonder
   Wire.beginTransmission(MPU);
   Wire.write(0x3B);  // starting with register 0x3B (ACCEL_XOUT_H)
   Wire.endTransmission(false);
   Wire.requestFrom(MPU, 14, true);  // request a total of 14 registers

   Accel_X = Wire.read()<<8|Wire.read();   // 0x3B (ACCEL_XOUT_H) & 0x3C
(ACCEL_XOUT_L)
   Accel_Y = Wire.read()<<8|Wire.read();   // 0x3D (ACCEL_YOUT_H) & 0x3E
(ACCEL_YOUT_L)
   Serial.print(ledState1);Serial.print("                                    ;
");Serial.print(ledState2);Serial.print(" ; ");
   Serial.print(ledState3);Serial.print("                                    ;
");Serial.print(ledState4);Serial.print(" ; ");
   Serial.print(Accel_X);Serial.print(" ; ");Serial.println(Accel_Y);

  /*
    Intensiteit aanpassen van de leds.
    Mogelijkheid tussen waarde_1 en nt_waarde_1, en waarde_2 en nt_waarde_2
  */
  //-------------------------------------------------------------------------
---------------------------------
  if (currentMicros - previousMicros >= interval) {
    previousMicros = currentMicros;
    if (verwissel1 == true) {
      ledValue1 = ledValue1 + minWaarde;
    } else {
      ledValue1 = ledValue1 - minWaarde;
    }
    switched1 = false;

    if (verwissel2 == true) {
      ledValue2 = ledValue2 - minWaarde;
    } else {
      ledValue2 = ledValue2 + minWaarde;
    }
    switched2 = false;

  }
  if (ledValue1 > (255 - minWaarde) && switched1 == false) {
    ledValue1 = 255;
    verwissel1 = !verwissel1;
    switched1 = true;
  } else if (ledValue1 < (0 + minWaarde) && switched1 == false) {
    ledValue1 = 0;
    verwissel1 = !verwissel1;
    switched1 = true;
  }
    if (ledValue2 > (255 - minWaarde) && switched2 == false) {
    ledValue2 = 255;
    verwissel2 = !verwissel2;
```

```
    switched2 = true;
  } else if (ledValue2 < (0 + minWaarde) && switched2 == false) {
    ledValue2 = 0;
    verwissel2 = !verwissel2;
    switched2 = true;
  } // ------------------------------------------------------------------
-----------------------------------

  // Aanpassen aan ledwaardes: Keuze uit ledValue1, -ledValue1, ledValue2 en
-ledValue2
  ledState4 = ledValue1;
  ledState3 = ledValue1;
  ledState2 = ledValue1;
  ledState1 = ledValue1;

  } else if (data == 48) {

  }

  analogWrite(led4, ledState4);
  analogWrite(led3, ledState3);
  analogWrite(led2, ledState2);
  analogWrite(led1, ledState1);
}
```

Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Real-Time Motion Compensated Video Camera

Richting: **master in de industriële wetenschappen: elektronica-ICT**
Jaar: **2016**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

Wilms, Leendert

Datum: **15/01/2016**