Lichtintensiteit, Resolutie en saturatie invloed op grayscale algoritmes gebruikt in facial recognition

# Namen en Datum

Mick Bos  
Ties Brouwer

12-03-2020

# Doel

Het doel van dit onderzoek is om te onderzoeken of er voor facial recognition software gebruikt in securitycamera’s uit een selectie grayscale conversie algoritme een effectiever algoritme te vinden is in vergelijking met de standaard implementatie. Dit wordt gebaseerd op hoe de verschillende algoritmes onderscheiden wanneer zij worden getest met portretten die verschillen in lichtintensiteit, saturatie en resolutie, aangezien de software voor camera situaties die hierop veel kunnen verschillen.

# Hypothese

Dit onderzoek werkt met verschillende grayscale conversie algoritmes die onderling verschillende kwaliteitseigenschappen hebben (zie: Werkwijze). Voor de verschillen in Lichtintensiteit, Saturatie en Resolutie waardes maken wij de volgende verwachtingen voor de algoritmes

* **Lichtintensiteit**Wij verwachten dat het *Luma algoritme* de beste resultaten zal leveren. Dit algoritme verkrijgt zijn resultaat, net zoals *Luminance,* door de RGB-waardes met vaste waardes te vermenigvuldigen. Het verschil ligt erin dat *Luma* haar resultaten duidelijkere verschillen tussen lichtere kleuren bevatten.
* **Saturatie**

Wij verwachten dat het *Value algoritme* de beste resultaten zal leveren. Value haalt zijn werking door de RGB-kanalen gelijk te trekken, wat overheen komt met het verhogen/verlagen van het *value* kanaal uit het HSV color space. De werking van dit algoritme zou moeten kunnen vermijden dat saturatie verschillen impact op het resultaat zullen leveren.

* **Resolutie**Aangezien resolutie verschil geen invloed heeft op de individuele RGB-kanalen verwachten wij dat de algoritmes in hun resultaat het meest zullen verschillen in hun mogelijkheid tot het creëren van geleidelijk kleurverschil. Hierdoor verwachten wij, net als bij lichtintensiteit, dat het *Luma algoritme* de hoogste Hit/miss ratio zal leveren. Er wordt bij de *Luma* resultaten extra nadruk overkleurverschil gezet, waardoor de facial recognition software gemakkelijker features zou moeten kunnen herkennen.

# Werkwijze

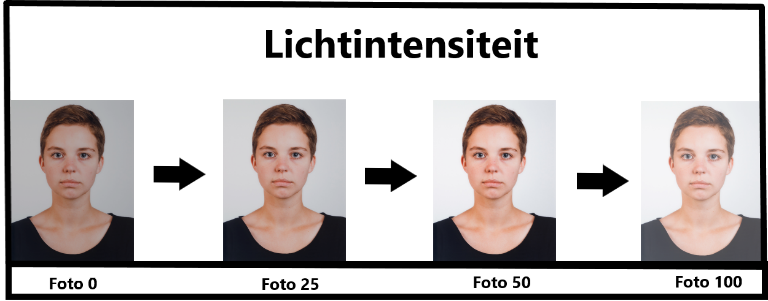
Voor dit onderzoek worden de volgende algoritmes1 getest:

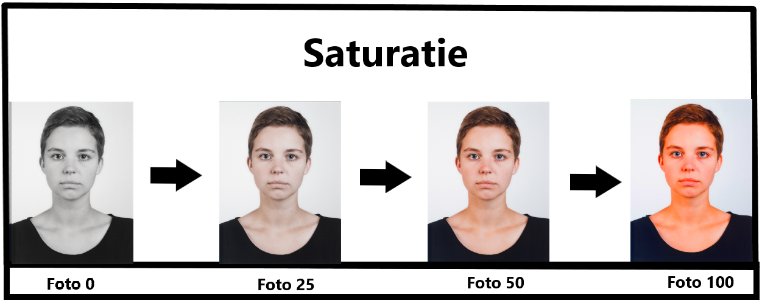
* **Intensity Algorithm**   
   Result = ((1/3) \* (R + G + B))
* **Value Algorithm**  
   Result = MAX(R, G, B)
* **Luminance Algorithm**   
   Result = (R \* 0.3) + (G \* 0.59) + (B \* 0.11)
* **Luma Algorithm**   
   Result = (R \* 0.2126) + (G \* 0.7152) + (B \* 0.0722)
* **Luster Algorithm**  
   Result = (1/2) \* (MAX(R, G, B) + MIN(R, G, B))

De verschillende algoritmes worden voor Lichtintensiteit, Saturatie en Resolutie ieder getest op een bijpassende testset. Elke testset bestaat uit een verzameling van 5 verschillende portretten (figuur 1), waarvoor voor elk portret 100 versies worden getest. Deze versies lopen van een factor (0) voor foto 0 naar een factor (100) voor foto 100, waarbij de factor gelijkstaat aan de waarde levels van hun corresponderende invloed. (figuur2)

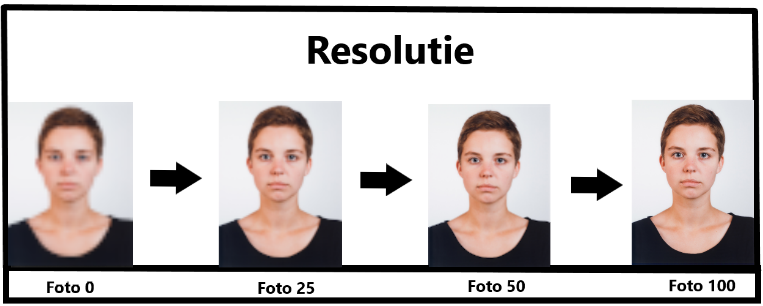


*Figuur 1*





*Figuur 2*



De verschillende software steps waar het resultaat van de algoritmes op wordt getest bestaat uit:

1. PreLocalization
2. Localization (1)
3. Localization (2)
4. Localization (3)
5. Localization (4)
6. Localization (5)
7. Prepare extraction
8. Extraction (1)
9. Extraction (2)
10. Extraction (3)
11. Extraction (4)
12. PostProcessing

De algoritmes worden op de volgende wijze individueel getest:

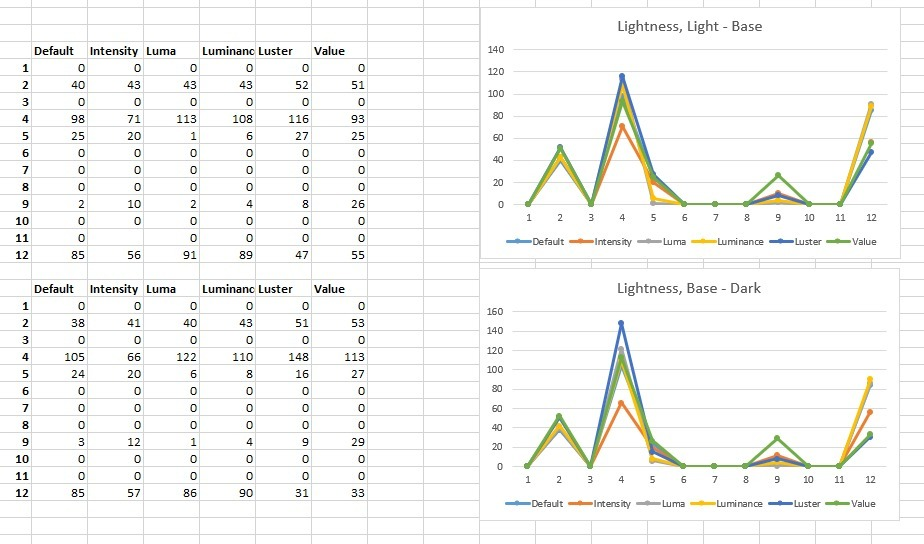
* Voer voor lichtintensiteit, saturatie en resolutie de volgende stappen uit:

1. Verwerk het eerste algoritme in de facial recognition software als grayscale conversie wijze.
2. Laad de eerste testset in en gebruik het huidige algoritme om dit portret te verwerken tot grayscale. Het algoritme slaat nu op welke software steps er mogelijk waren voor de resulterende grayscale.
3. Herhaal stap 2 tot alle testsets gemeten zijn.
4. Bereken en noteer voor het algoritme hoe vaak een software step is behaald over de verschillende portret testsets.
5. Herhaal stap 1 tot en met 3 voor de overige algoritmes.

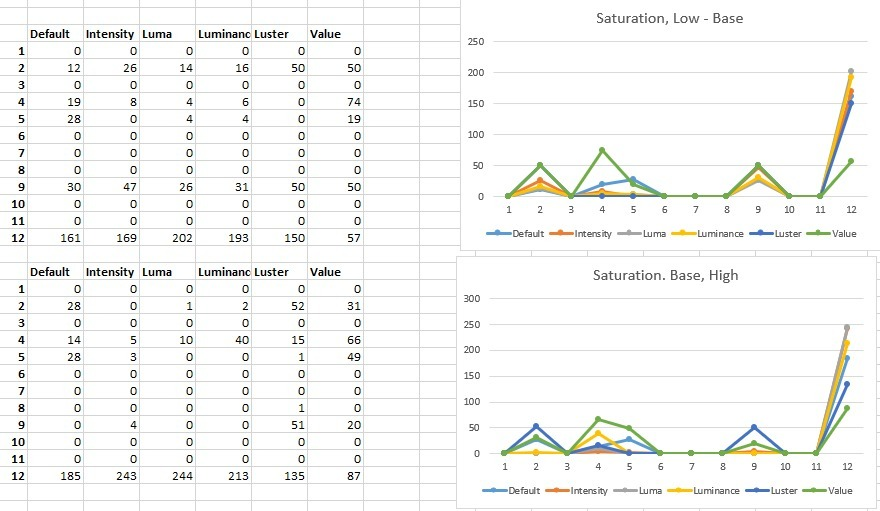
# Resultaten

De volgende 3 meetresultaten zijn afleidingen van de verkregen data. De tabellen aan de linkerkant staan voor hoeveel steps een algoritmes over al zijn testsets heeft verkregen Deze tabellen zijn opgesplitst in 2, waarvan de bovenste staat voor alle portretten oplopend van een factor(0) tot factor (50), en de onderste voor alle portretten van factor(50) tot factor(100). Rechts staan de grafieken die erop gebaseerd zijn.

# Resultaten Lichtintensiteit



# Resultaten Saturatie



# Resultaten Resolutie

# 

# Verwerking

Uit de meetresultaten is een conclusie te trekken op basis vanwaar de pieken in de grafiek te vinden zijn. Hierbij is het belangrijk hoe de verschillende algoritmes met elkaar vergelijken op de grafiek pieken.

# Verwerking Lichtintensiteit

Het is duidelijk te zien dat het *default algoritme, luster* en *luminance* erg gelijklopen met hoe effectief zij werken wanneer het gaat op lichtere portretten. Dit is te vinden op basis van hoeveel portretten zij het gehele proces hebben laten doorlopen. Dit transleert zichzelf ook naar de tweede grafiek, waarin te zien is dat de verschillen in resultaten nog steeds gelijklopen. Daarbij is aan te merken dat het *value* algoritme vaak vastliep bij de extracties, maar er wel een heel aantal doorheen heeft gekregen.

# Verwerking Saturatie

Uit de grafieken is duidelijk te lezen dat ieder algoritme behalve value hier een stuk minder last had van de verschillen in saturatie. Daarbij behalen *Luminance* en *Luma* op een aantal haperingen na van de verschillende algoritmes de meeste portretten die door het gehele proces zijn gekregen. Verder behaalt *default implementatie* resultaten die ver achterwegen blijven hangen in vergelijking met deze twee. Bij de tweede grafiek valt dit anders, aangezien *Luminance* dit keer lagere eindresultaten behaald dan *Luma. Daarnaast behaalt Intensity* dezelfde resultaten als *Luma*, waarbij het bij de lichtere portretten juist mindere resultaten leverde.

# Verwerking Resolutie

Waarbij het bij Lichtintensiteit en saturatie verschillen een nek aan nek race was is van de resolutie grafieken overduidelijk af te lezen dat het *Luminance* algoritme de beste resultaten voor de software heeft gemaakt. Bij elke onderlinge stap heeft *Luminance* de hoogste resultaten.

# Conclusie Lichtintensiteit

# Conclusie Saturatie

# Conclusie Resolutie

# Evaluatie

Om de hypothese van dit meetrapport te beantwoorden: Het beste algoritme om de default implementatie te verbeteren is Luminance. Luminance heeft de hoogste hit ratio van alle geteste algoritmes.