2/13/2020

Kevin Patist, Wietse ten Dam

Implementatieplan

Plan van implementatie

Table of Contents

[Doel 2](#_Toc33170854)

[Hoofdvraag 2](#_Toc33170855)

[Methoden 3](#_Toc33170856)

[Canny edge detection 3](#_Toc33170857)

[Canny met Gaussian filter 3](#_Toc33170858)

[Sobel operator 3](#_Toc33170859)

[Prewitt 4](#_Toc33170860)

[Voor-en nadelen 4](#_Toc33170861)

[Keuze 5](#_Toc33170862)

[Implementatie 6](#_Toc33170863)

[Evaluatie 7](#_Toc33170864)

[Bronnen 8](#_Toc33170865)

# Doel

* Om van langs lopende mensen de gezichten te kunnen vinden en daarop herkenning te kunnen uitvoeren.
* Tijdens de epidemie van het coronavirus willen we graag een camera met gezichtsherkenning en een warmte camera samen laten werken. Dan kunnen we snel de mensen identificeren die een te hoge lichaamstempratuur hebben zodat die zo snel mogelijk medische hulp krijgen.
* De warmte camera en het koppelen daarvan doen wij niet, wij zorgen voor een programma dat gezichten detecteert en herkent. Wij hebben geen live view wij krijgen foto’s van mensen binnen en detecteren een gezicht en voeren herkenning daarop uit. Om dit optimaal te kunnen toe te passen moeten wij een gang hebben met goede verlichting en een camera in het midden van de gang.
* Wij willen de edge detection verbeteren om de volgende reden: Als er betere edge detection is krijg je een accurater beeld van de gezichtskenmerken wat het makkelijker kan maken om de persoon te kunnen detecteren en te kunnen herkennen.

# Hoofdvraag

Hoe kunnen wij de huidige edge detection verbeteren zodat wij een hogere accuraatheid krijgen?

# Methoden

## Canny edge detection

Bij canny edge detection word de output van de sobel operator gebruikt. De sobel operator is ook een edge detection algoritme maar Canny perfectioneert die uitkomst. De output van sobel ziet er meestal wel oké uit maar voor een computer is die nog niet heel goed. Canney neemt dus de uitkomst van sobel en gaat daarop aan de slag. Alle lijnen die sobel gemaakt heeft daar gaan Canney 1 pixel dikke lijnen van maken om echt gebruik te maken van een hoge resolutie als die er is want sobel maakt puur gebruik van een edge en als die edge dik blijkt te zijn dan is die lijk ook dik en niet precies. Canny maakt gebruikt van de lijnen van sobel en zoekt voor een local optimum, waar de objecten zich echt scheiden en legt daar de 1 pixel dikke lijn op. Door dit toe te passen heb je na het goede algoritme van sobel wat extra detail.

Het is ook belangrijk om een threshold toe te voegen aan de Canny edge detection omdat als je dat niet doet kan die alle kleine in perfecties ook zien als een lijn en dat willen we niet. Door een threshold toe te voegen kan je alleen maar lijnen toestaan boven een bepaalde kracht. Een dubbel threshold is nog beter om toch wat meer detail te behouden, Een dubbel threshold betekend dat je een top waarde kiest en een laag waarde. Alles wat boven de top waarde zit word sowieso getoond daarin tegen alles wat onder de laag waarde zit word sowieso niet getoond. Tussen de top en laag lijn zit een grijs gebiedje, Alles wat in die grijze laag ligt en alles wat vast zit aan de top laag word wel getoond maar als het niet aan de top laag vast zit en dus aan de laag lijn, dan word het niet getoond. Zo word de edge detectie toch preciezer maar blijf je toch de kleine in perfecties voor.

Dit is afkomstig van “Canny edge detection” <http://justin-liang.com/tutorials/canny/>

# Canny met Gaussian filter

In <https://link.springer.com/chapter/10.1007/978-3-642-33478-8_7> “The Canny Edge Detection and Its Improvement” staat dat Gaussian filter ervoor kan zorgen dat de grijs levels smoother over gaan in elkaar en dat je dan false hits makkelijker kan tegen gaan. Een false hit is natuurlijk niet wat je wil, een Gaussian filter is een tool om het plaatje een beetje onscherp te maken om zo alle kleine lijtjes die als edge gezien zouden kunnen worden worden vervaagt en blijven echt alleen de dikke echt edges over. Een edge detectie programma kan dan beter zijn best doen en ook accurater zijn

## Sobel operator

De sobel operator gebruikt twee masks, een voor de x as en een voor de y as. Hiermee zoekt hij op de desbetreffende as naar edges. Als beide masks over de image zijn gehaald wordt het resultaat samengevoegd.

<https://www.researchgate.net/publication/297736749_Comparison_of_Edge_Detection_Techniques>

## Prewitt

Prewitt is zeer gelijk aan de Sobel operator en is gebruikt om randen te detecteren op de x en y assen. Prewitt legt in tegenstelling tot Sobel minder de nadruk op de pixels in de buurt van het midden. De kernel is bij Prewitt heeft gelijke nummer met min en plus maar sobel heeft meedere nummers ook in plus en min.

ijser.org/researchpaper/Edge-Detection-by-Using-Canny-and-Prewitt.pdf

## Fuzzy logic

### Voor-en nadelen

|  |  |  |
| --- | --- | --- |
| Methode | Voordelen | Nadelen |
| Canny edge detection | Hele scherpe edge detection | Sobel of Prewitt moet sowieso geimplementeerd worden.  Lastig te implementeren.  Kost meer tijd en resources. |
| Canny met Gaussian filter | Het plaatje word soft gemaakt waardoor de ruis minder invloed heeft.  Hele goede edge detection en reactie op randen.  Zeer precies | Sobel moet en het Gaussian filter.  Lastig om te implementeren  Kost veel processor tijd  Is niet tijd efficent. |
| Sobel operator | Edge detection met lichte smoothing  Kost weinig tijd/resources  Dectecteerd ook de orentatie van de randen | Werkt minder goed als er meer ruis is. |
| Prewitt | Detecteerd de randen en de orentatie van de egde.  Simpel | Hij is niet zo accuraat en werkt slecht met ruis |

# Keuze

Er is gekozen voor het gebruikt van de sobel operator omdat het een efficient en snelle methode. Sobel is boven Prewitt gekozen omdat Sobel ook automatisch wat smoothing toepast op de afbeelding, dit zorgt ervoor dat het wat minder gevoelig is voor ruis. Ook kunnen we de kernel van Sobel aanpassen waardoor het flexibeler is dan Prewitt. In verband met de tijd en verwachtingen voor het implementeren van Canny is besloten het bij Sobel te laten.

# Implementatie

Om de Sobel operator te gaan gebruiken moet er een C++ implementatie geschreven worden. Dit zal gedaan worden door de volgende bestanden aan te passen:

* StudentPreProcessing.h
* StudentPreProcessing.cpp

Binnen deze bestanden zullen de stepEdgeDetection en stepThresholding functies worden aangepast. In de stepEdgeDetection functie komt een nieuwe implementatie van Sobel en de stepThresholding wordt een thresholding functie die afgestemd staat om optimaal samen te werken met de stepEdgeDetection functie. Ook is het mogelijk dat de private variabelen van de StudentPreProcessing klasse aangepast worden als er wordt ontdekt dat dit nodig is voor het goed implementeren van Sobel.

Om de nieuwe Sobel implementatie optimaal te laten werken gaat er onderzocht worden wat de beste kernel en thresholding waarden zijn. De uitkomst van dit onderzoek bepaald welke kernel en thresholding waarden in het eindproduct gebruikt gaan worden.

Bij deze implementatie nemen wij aan dat de rest van het facial recognition proces in werkende staat is.

# Evaluatie

# Bronnen

“Canney edge detection”(15-02-2020): <http://justin-liang.com/tutorials/canny/>

“The Canny Edge Detection and Its Improvement”(16-02-2020):

<https://link.springer.com/chapter/10.1007/978-3-642-33478-8_7>

“Edge Detection by Using Canny and Prewitt” (21-02-2020):

ijser.org/researchpaper/Edge-Detection-by-Using-Canny-and-Prewitt.pdf

“Edge Detection in Digital Images Using Fuzzy Logic Technique” (21-02-2020)

<https://www.freeprojectsforall.com/wp-content/uploads/2018/10/Edge-Detection-Techniques-using-Fuzzy-Logic.pdf>