Implementatieplan

Bram van Bergeijk
1691083
Leo Jenneskens
1679950
23-2-2017

Doel:

Het praktikum heeft als doel om één of meer onderdelen van een gezichtsherkennings-framework aan te passen. Het is de bedoeling dat dit gebeurd met eigen code die geen gebruik maakt van ondersteunende libraries.
Het gerichte einddoel van dit project is vervangende code te schrijven voor de Image Shell voor de face recognition applicatie en een code voor de conversie van RGB naar Intensity (ook wel Grayscale) die equivalent of beter (ondermeer op het gebied van snelheid en memory efficiency) is.

Methoden:

Om een RGB abeelding naar grayscale te converteren hebben wij een aantal algoritmes tot onze beschikking. De meest gebruikte is Average, “the most common grayscale conversion routine” (Helland, 2011). Average is een algoritme waarbij simpelweg het gemiddelde van de RGB waarde neemt, om die vervolgens om te zetten naar grayscale. Het simpelste algoritme is met afstand het single color algoritme “ the

fastest computational method for grayscale reduction” (Helland, 2011). Een overzicht van zes methoden en hun voor en nadelen voor de RBG naar grayscale conversie.

Methoden	Voordelen	Nadelen
Luminance	Correct voor het menselijke oog.	Zwaarder met meer berekeningen.
Average	Simpel en duidelijk algoritme.	Weinig finesse.
Single colour channel	Snelst en kost het minste geheugen.	Praktisch twee/derde van de afbeelding wordt niet gebruikt.
Decomposition	Neemt de waarde met de meeste invloed en voert single colour per pixel uit waardoor er duidelijke verschillen tussen donker en licht zijn.	Ziet niet goed verschil tussen felle kleuren naast elkaar.
Desaturation		Weinig contrast en veel berekening.
Lightness		Relatief weinig contrast.

Keuzes:

Er is voor gekozen om tot een drietal aan algoritmen te implementeren, afhankelijk van de resultaten na elke test. Single colour channel (zowel met blauw, groen als rood) is de eerste wegens het gebrek aan daadwerkelijke berekeningen waardoor dit ook de snelste is. Mocht dit niet het gewenste resultaat opleveren, dan is het de beurt aan de standaard Average. Hierbij wordt simpelweg het gemiddelde van de drie RGB waarden gemiddeld. In het geval dat ook dit niet werkt, wordt er overgestapt naar Luminance, “a more sophisticated version of the average method”

(Cook, More on colors and grayscale, 2009). Die de RBG niet gelijkwaardig laat meewegen in de middeling.

Implementatie:

Voor de implentatie van dit onderzoek gaan we in drietal classen implementeren. Voor de imageshell zullen we implementeren in de class intensityimagestudent en in de class rgbimagestudent. Hierin zullen de afbeeldingen worden opgeslagen en kunnen worden opgevraagd. Voor het rgb naar grayscale conversie zullen we gaan implementeren in de class studentpreproccesing . hierin zullen we de verschillendealgoritmes testen die gekozen zijn voor het onderzoek.

Evaluatie:

Het is de bedoeling dat wanneer het laatste stadium van het project is afgerond, De nieuw geschreven code beter presteert op het gebied van snelheid en/of memory efficiency. Voor het meten van de snelheid van de code kan een vergelijking worden gemaakt met de eerste clock pulse van het programma en het laatste clock pulse van het programma hiermee kan de relatieve snelheid worden gemeten van de code. Hierin zal hoogstwaarschijnlijk het Single colour channel algoritme als snelste en meest efficiënte uit de test komen, mits deze de test naar behoren doorstaat.

Bibliografie

Cook, J. D. (2009, August 24). *More on colors and grayscale*. Opgehaald van johndcook.com: https://www.johndcook.com/blog/2009/08/24/more-on-colors-and-grayscale/

Cook, J. D. (2009, August 24). *Three algorithms for converting color to grayscale*. Opgehaald van johndcook.com: https://www.johndcook.com/blog/2009/08/24/algorithms-convert-color-grayscale/

Helland, T. (2011, october 1). *Seven grayscale conversion algorithms*. Opgehaald van tannerhelland.com: http://www.tannerhelland.com/3643/grayscale-image-algorithm-vb6/