1. Implementatieplan Imageshell & Intensity

1.1. Namen en datum

Jesper Obbes

Sander van Sterkenburg

Datum: 12/04/2019

1.2. Doel

De aanleiding voor het schrijven van dit implementatieplan is tot stand gekomen omdat wij als studenten van de Hogeschool Utrecht de opdracht hebben gekregen om een onderdeel van een gezichtherkennings-framework aan te passen.

In dit onderdeel zal er gewerkt worden aan een Imageshell voor de face recognition applicatie en zal er code geschreven worden voor de conversie RGB → Intensity. Dit wordt ook wel Grayscale genoemd. De resultaten van deze veranderingen en toevoegingen zullen er voor moeten zorgen dat de resultaten vergelijkbaar of beter zijn dan de aangeleverde code.

1.3. Methoden

1.3.1 Imageshell

Het belangrijkste van de imageshell is het opslaan van de pixels. Hier zijn diverse methoden voor. Zo kan er gebruik gemaakt worden van een van de STL-containers zoals std::Vector<> of std::Array<> of van een C array. Eenmaal als deze keuze is gemaakt moet er ook nog gekeken worden hoe de pixels worden opgeslagen in een van de containers. Worden alle pixels in 1 container gezet of worden er containers in containers gezet met voor elke rij en/of kolom dan een andere container die de pixels daarvan bevat.

1.3.2 RGB → Intensity

Om van RGB naar Intensity te gaan moet er gebruik worden gemaakt van een algoritme. Er is gekeken naar drie verschillende algoritmes die wij kunnen toepassen.

Het eerste algoritme is een simpele intensity algoritme. De formule is: $$\mathcal{G}_{Intensity} \leftarrow \frac{1}{3}(R+G+B).$$

Het tweede algoritme is Luminance. Dit algoritme wordt veel gebruikt in image processing software zoals GIMP. De formule is: $$\mathcal{G}_{Luminance} \leftarrow 0.3R+0.59G+0.11B.$$

Het derde en laatste algoritme is Value. Hierbij wordt gekeken naar de maximum van de RGB channels. De formule is: $$G_{Value} = \max(R,G,B).$$

1.4. Keuze

Voor de imageshell is er gekozen voor voor een array. Dit is een stuk sneller dan het gebruiken van een vector aangezien dit veel overhead levert.

Voor RGB → Intensity is er gekozen voor om alle drie de formules te testen. Op basis van deze resultaten zullen we het beste algoritme gebruiken.

1.5. Implementatie

De imageshell wordt zo geïmplementeerd dat de constructor en de set functies al gelijk genoeg bytes alloceren op de heap om de afbeelding op te slaan. Om ervoor te zorgen dat data verwijdert wordt als het nodig is zal de deconstructor deze verwijderen en de set functies ook.

Voor RGB→Intensity wordt er met een for-loop door de afbeelding gelopen. Voor elke pixel in de afbeelding worden de kleurwaardes opgehaald en wordt daarop de intensity berekent. Dit wordt uiteindelijk opgeslagen in de IntensityImage klasse.

1.6. Evaluatie

Voor de evaluatie van de Imageshell zal er gekeken worden naar de werking ervan en de snelheid van het proces vergeleken met de standaard implementatie. Als dit sneller of beter is weten we dat de werking ervan goed is.

Voor de evaluatie van RGB→Intensity zal er gekeken worden naar de kwaliteit en naar de snelheid van de formules vergeleken met de standaard implementatie.

1.7. Bronnen

https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0029740