1. Verbetering Grayscale Algorithm

1.1. Namen en datum

Baartman, Michel
Bout, Nick
03 April 2018

1.2. Doel

Het doel is te meten van de snelheid van zowel de vijf nieuw geïmplementeerde grayscale algoritmes als het oude grayscale algoritme en hiermee de onderstaande hypothese te bewijzen.

1.3. Hypothese

Elk van het vijftal nieuwe algoritmes zijn sneller dan het oude algoritme. Ook zullen de nieuwe algoritmes meer tijdwinst opleveren bij grotere afbeeldingen.

1.4. Werkwijze

De snelheid van de algoritmes zal getest worden met behulp van van de c++ library “chrono”. Elk algoritme zal op meerdere afbeeldingen 10.000 keer uitgevoerd worden en vervolgens zal de gemiddelde tijdsduur van de functie per afbeelding genoteerd worden. De afbeeldingen die gebruikt gaan worden zijn een 640x640 afbeelding genaamt *carlos.jpg* en een 198x255 afbeelding genaamt *male-3.png*. Beide afbeeldingen kunnen gevonden worden in de testset map van de locatie. Er is voor de *male-3.png* afbeelding gekozen omdat het oude algoritme werkte met deze afbeelding. Hiermee kan dus gegarandeerd worden dat het mogelijk is om een grayscale algoritme uit te voeren op het bestand met de huidige applicatie.Daarnaast is voor *carlos.jpg* gekozen omdat er een afbeelding nodig was met andere afmetingen dan *male-3.png.* Dit is nodig om te kunnen testen of de grote van de afbeelding impact heeft op de tijdswinst van één van de nieuwe algoritmes. Omdat alle afbeeldingen in de testset dezelfde afmetingen hadden moest er een nieuwe afbeelding komen en *carlos.jpg* was de eerst gevonden afbeelding van een persoon. Het meten van deze tijdsduur zal gedaan worden in de main.cpp van de geleverde software. De begintijd wordt met behulp van chrono opgeslagen voor de “***if (!executor->executePreProcessingStep1(true)) {***“ regel in de functie: ***executeSteps***. Na de if conditie wordt de eindtijd opgeslagen en kunnen we de duratie bepalen door de begintijd van de eindtijd af te tellen. Voor het oude algoritme zal van te voren de implementatie op default gezet worden door middel van ***ImageFactory::setImplementation(ImageFactory::DEFAULT);*** bij de nieuwe algoritmes wordt de implementatie op student gezet door middel van ***ImageFactory::setImplementation(ImageFactory::STUDENT);*** Daarnaast wordt voor het oude algoritme de functie ***executePreProcessingStep1()*** met de parameter false uitgevoerd en de nieuwe algoritmes met true. De logica van het oude algoritme is niet in te zien maar het algoritme kan wel gebruikt worden.

1.5. Resultaten

Zoals in de werkwijze beschreven zijn de algoritmes getest op twee afbeeldingen en 10.000 keer uitgevoerd per afbeelding. De afbeeldingen zijn gekozen omdat te meten of er meer tijdswinst was bij grotere afbeeldingen. In de onderstaande tabel worden de gemiddelde metingen in milliseconden per algoritme per afbeelding getoond. Eén van de testen werd (bij toeval) twee keer uitgevoerd. Hierbij bleek dat zelfs met 10.000 iteraties er nog steeds een afwijking in het gemeten gemiddelde aanwezig kan zijn.

Algoritme	**carlos.jpg, 640 x 640**	**male-3.png, 198 x 255**
Oude algoritme	10.2047ms	0.93243ms
Averaging	3.97743ms	0.47540ms
Luma/Luminance	4.39395ms	0.50958ms
Desaturation	5.93243ms	0.72865ms
Decomposition (min)	4.23714ms	0.51868ms
Decomposition (max)	4.21205ms	0.51065ms

1.6. Verwerking

De metingen hebben aangetoond dat er tot wel 6.22727 milliseconde verschil kan zitten tussen het oude algoritme en één van de nieuwe algoritmes. Ook is er aangetoond dat hoe groter de afbeelding hoe meer tijdswinst er wordt gemaakt. Dit wordt extra weergeven in de onderstaande tabel.

Algoritme	**Aantal keren sneller dan oude algoritme, carlos.jpg**	**Aantal keren sneller dan oude algoritme, male-3.png**
Averaging	2.57	1.96
Luma/Luminance	2.32	1.83
Desaturation	1.72	1.28
Decomposition (min)	2.41	1.78
Decomposition (max)	2.42	1.83

1.7. Conclusie

Het oude algoritme is het langzaamste algoritme van de zes geteste algoritmes. De averaging is het snelst. Ook kan er geconcludeerd worden dat hoe groter de afbeelding hoe meer tijdswinst er gemaakt wordt op de nieuwe algoritmes.

1.6. Evaluatie

De nieuwe algoritmes zijn inderdaad sneller dan het bestaande algoritme. Maar omdat de verkregen applicatie niet heel stabiel is, is zelf met het runnen van 10.000 iteraties de tijdsduratie nog erg flexibel. Dit kan zorgen voor eventuele meetfouten. Daarnaast werkte de rest van de image processing stappen niet altijd met de nieuwe grayscale image die resulteerde uit het algoritme. Hier zou nog een meetrapport over moeten komen.