MEETRAPPORT SNELHEID

ImageShell & Intensity

Christiaan van den Berg & Mike Hilhorst
1660475 & 1676029

Inhoud

Doel
Wij willen graag met dit experiment bepalen welke implementatie het snelst is van de twee.

Hypothese
Wij vermoeden dat onze eigen implementatie sneller is dan de standaard implementatie, omdat Singel color channel maar naar kleur kanaal kijkt, zoals de naam doet vermoeden. Omdat er maar een kanaal wordt aan gesproken, zou het ook minderlang moeten duren om de afbeelding te aan te passen. Dus het totaal zou sneller moeten zijn.

Werkwijze
Wij hebben een lijst met test afbeeldingen opgesteld van verschillende groten om onze en de originelen implementatie te testen. Deze bestaat uit de volgende onderdelen:

Honderd afbeeldingen met de resolutie:

- 128 bij 128
- 256 bij 256
- 512 bij 512
- 1024 bij 1024

Tien afbeeldingen met veel gebruikte computerscherm resoluties:

- 640 bij 480
- 1280 bij 720
- 1920 bij 1080
- 2160 bij 1440

Wij gaan deze afbeeldingen hun snelheid testen doormiddel van de diagnostic tools van Visual Studio 2017. Elke set afbeeldingen worden apart getest waarna de runtime wordt opgeslagen en genoteerd.

De CPU(Intel(R) Core(TM) i5-6600K CPU @ 3.50GHz, 3504 Mhz, 4 Core(s), 4 Logical Processor(s)) van de computer waar het op getest wordt, heeft een maximaal gebruik van 25% bij elke test, om de resultaten constant te houden tussen de implementaties.

Dit doen we voor beide implementaties. De resultaten zullen wij met elkaar vergelijken en verwerken tot overzichtelijke vorm zoals een tabel en/of diagram.

Resultaten

Geef de meetresultaten overzichtelijk weer in de vorm van een tabel en/of diagram.

Gemiddeld	Originele implementatie	Onze implementatie
(200 afbeeldingen)128x128	1.498	1.133
(200 afbeeldingen)256x256	5.392	4,114
(54 afbeeldingen)512x512	6.065	4.569
(80 afbeeldingen)1024x1024	35.567	26.168

Tijd is in seconden(s)

Gemiddeld	Originele implementatie	Onze implementatie
(9 afbeeldingen)640x480	1.034	0.905
(9 afbeeldingen)1280x720	3.558	2.708
(9 afbeeldingen)1920x1080	8.226	6.020
(10 afbeeldingen)2160x1440	15.599	11.270

Tijd is in seconden(s)

Verwerking

Wij hebben meerder metingen gedaan over de originele en onze implementatie. Wij hebben verschillende groten afbeeldingen gebruikt, dus wij krijgen hier van veel data. Deze data nemen wij het gemiddelde van en met die resultaten gaan wij tussen de implementaties vergelijken.

Per resolutie gaan we een mini conclusie trekken die we samen voegen tot een grote(eind conclusie)

Conclusie

Met onze meetresultaten zijn we tot de volgende mini conclusie gekomen. Per resolutie staat hier onder hoeveel procent onze implementatie sneller is, tegen over de standaard implementatie:

Resolutie	Procent sneller
128x128	24.4%
256x256	24.7%
512x512	24.7%
1024x1024	26.4%
640x480	12.5%
1280x720	23.9%
1920x1080	26.8%
2160x1440	27.8%

Uit deze metingen kunnen wij halen dat onze implementatie gemiddelde ~25% sneller is dan de standaard implementatie. Dus wij kunnen als conclusie stellen dat onze implementatie daadwerkelijk sneller is.

Evaluatie

Wij hebben best veel metingen gedaan om goed te kunnen bepalen welke implementatie het snelst is(~600 afbeeldingen). Wij hebben ook testen meerder keren uitgevoerd om ruis te verkleinen in de resultaten. We hadden deze testen nog groter kunnen doen om onze vertrouwbaarheid te vergroten. Dit hebben wij niet gedaan door tijds gerelateerde zaken.

Ergens waar wij geen invloed op hebben is Windows, Windows 10 staat er vooral om bekent dat deze ongevraagd veel ressources gebruikt op computers die deze OS een tijd geïnstalleerd hebben staan. Dit kan in vloed hebben op de resultaten, wat wij hadden kunnen doen is onze code testen op een verse installatie van Windows. Dit is voor de volgende keer handig om naar te kijken.

De conclusie is dat onze implementatie beter is met wel 25% gemiddeld. Ons doel was om een snellere implementatie te maken. Met alle onzekerheden in gedachte kunnen we zeggen dat dit een succes was, onze implementatie is sneller dan de standaard implementatie. Het project was dus een succes.