1. Snelheidstest

1.1. Namen en datum

Willem de Groot & Henrike Kraan-Bos, 13-04-2018

1.2. Doel

Hoe verhouden de snelheden van de verschillende methodes zich tot elkaar?

1.3. Hypothese

We verwachten dat de middelste pixel methode het snelst is, dat Gaussian methode langzamer is dan de gemiddelde methode en dat de mediaan methode iets langzamer zal zijn dan de gemiddelde methode.

1.4. Werkwijze

Geef een korte beschrijving van het experiment. (Het overschrijven van de practicumhandleiding is niet nodig.) Maak indien nodig een tekening van de proefopstelling, waarin grootheden kunnen worden aangegeven.

We hebben voor de tijdmetingen de standaard library `<ctime>` gebruikt. In de functie `stepScaleImage` hebben we de code om de tijd te starten en te stoppen respectievelijk aan het begin en eind van die functie toegevoegd. Dat is de volgende code:

```
std::clock_t start;
double duration;
start = std::clock();

...

duration = (std::clock() - start) / (double)CLOCKS_PER_SEC;
std::cout << "Time: " << duration << std::endl;
```

Hiermee meten we hoe lang het duurt om het volledige proces uit te voeren.

We hebben drie verschillende test images gebruikt. Hieronder staan de details:

Naam	Afmetingen	Bestandsgrootte
custom1.jpg	990x1260 pixels	110,607 bytes
custom2.jpg	995x1396 pixels	669,759 bytes
custom3.jpg	2007x2671 pixels	516,159 bytes

1.5. Resultaten

Custom1

	Default	Middelste pixel	Mediaan	Gemiddelde	Gaussian
Tijd (s)	0,087	0,140	1,445	0,216	0,251
	0,098	0,141	1,510	0,214	0,250
	0,087	0,143	1,573	0,214	0,250
	0,089	0,144	1,461	0,222	0,266
	0,094	0,144	1,508	0,221	0,258
	0,088	0,142	1,563	0,209	0,260
Gemiddelde	0,091	0,142	1,510	0,216	0,256

Custom2

	Default	Middelste pixel	Mediaan	Gemiddelde	Gaussian
Tijd (s)	0,112	0,143	1,505	0,216	0,253
	0,097	0,143	1,555	0,219	0,249
	0,097	0,142	1,636	0,217	0,245
	0,090	0,138	1,598	0,215	0,253
	0,094	0,141	1,537	0,213	0,241
	0,107	0,147	1,522	0,219	0,252
Gemiddelde	0,100	0,142	1,559	0,217	0,249

Custom 3

	Default	Middelste pixel	Median	Gemiddelde	Gaussian
Tijd (s)	0,436	0,554	5,245	0,862	1,051
	0,416	0,534	5,282	0,853	1,025
	0,402	0,549	5,432	0,842	1,040
	0,414	0,536	5,398	0,838	1,031
	0,422	0,528	5,412	0,837	1,005
	0,465	0,532	5,390	0,837	1,030
Gemiddelde	0,426	0,539	5,360	0,845	1,030

1.6. Verwerking

We hebben per methode per image de gemiddelde tijd berekend met de formule:

$$\frac{t1 + t2 + \cdots + t6}{6}$$

Vervolgens hebben we per image de verhouding tussen de snelheden berekend met de formule:

$$\text{verhouding} = \frac{\text{gemiddelde methodewaarde}}{\text{gemiddelde defaultwaarde}}$$

Verhouding	Default	Middelste pixel	Mediaan	Gemiddelde	Gaussian
custom1	1	1,573	16,685	2,387	2,827
custom2	1	1,430	15,667	2,176	2,501
custom3	1	1,265	12,587	1,984	2,420

1.7. Conclusie

We zien in de meetresultaten terug de default implementatie het snelst werkt. Van onze eigen methodes werkt de middelste pixel methode het snelst, anderhalf keer zo langzaam als de default methode. De mediaan is opmerkelijk langzaam, deze doet er meer dan 12 keer zo lang over als de default implementatie. We zien wel dat het verschil bij een groter image terugloopt, wat overigens voor de andere methodes ook geldt. Verder zien we dat de gemiddelde methode ongeveer tweemaal zo langzaam is als de default methode. De Gaussian methode is ongeveer 2.5 keer zo langzaam.

1.8. Evaluatie

We zijn nu achter de snelheden van de verschillende methodes gekomen bij afbeeldingen van verschillende afmetingen en hoe deze zich tot elkaar verhouden.

Onze verwachting was dat de middelste pixel methode het snelst is van onze methodes, dit bleek inderdaad zo te zijn. De mediaan methode bleek veel langzamer dan gedacht. Waarschijnlijk komt dit door het sorteeralgoritme dat is gebruikt. Wij hebben de `std::sort` gebruikt, en aangezien bij grotere afbeeldingen de snelheidsverhouding kleiner wordt, denken we dat de standaard library QuickSort gebruikt, wat beter werkt bij grotere aantallen. Wat wel het geval bleek was dat de Gaussian methode iets langzamer is dan de gemiddelde methode.

Aangezien de verschillende metingen van dezelfde opstelling weinig van elkaar afwijken, achten wij deze testopstelling betrouwbaar. Het is wel mogelijk om nog iets dichter op de methode te meten, maar dit slaat alleen een paar vaste stappen over.

Verder kunnen de verschillende methodes nog geoptimaliseerd worden. Zo hoeft voor de Gaussian methode maar een kwart van de waardes te worden uitgerekend, aangezien deze in iedere richting hetzelfde zijn. Daarnaast zijn alle box samples van een image dezelfde grootte, dus hoeft dit ook niet voor iedere box sample opnieuw berekend te worden. Ook hoeven voor het kiezen van de middelste pixel niet alle pixels in een box sample geplaatst te worden, maar om de code universeler te houden doen wij dit wel. Voor de mediaan methode zou nog een ander sorteeralgoritme kunnen worden gekozen, afhankelijk van het aantal pixels.