# 1D vs 2D array’s

## Namen en datum

Hamza ait Messaoud, 01-01-2018

## Doel

In dit experiment wordt nagegaan welke vorm van dataopslag sneller is. Hierbij wordt een 1-dimensionale array vergeleken met een multidimensionale array. In dit voorbeeld maken we gebruik van een 2-dimensionale array.

## Hypothese

De verwachting is dat een 1-dimensionale array sneller werkt, deze vorm van opslag vergt minder geheugen instructies. Bij een multidimensionale array moet voor elke dimensie het adres opgehaald worden voordat de data opgehaald kan worden.

## Werkwijze

Eerst worden 20 testen uitgevoerd met de 1-dimensionale array en daarna worden dezelfde testen uitgevoerd met de 2-dimensionale array. Vervolgens wordt gemiddelde genomen om de testen te kunnen vergelijken.

Het volgende stuk code is gebruikt om te testen. Om memory allocatie niet mee te nemen in de test is gekozen om van tevoren al de image aan te maken. Maar om de memory allocatie van de array wel mee te nemen wordt de image telkens weer terug gezet op 0 bij 0. Bij het zetten van de afbeelding is gekozen om een copy te maken van een andere afbeelding waardoor er 20 keer een read en 20 keer een write wordt uit gevoerd.

RGBImageStudent \* copyImage = new RGBImageStudent();

Clock\_t startTime, endTime;

for(int i = 0; i < 20; i++) {

startTime = clock();

copyImage->set(\*image);

endTime = clock();

std::cout << "Time: " << endTime - startTime << "\n";

copyImage->set(0, 0);

}

## Resultaten

|  |  |  |
| --- | --- | --- |
|  | 1-Demensionaal | 2-Demensionaal |
|  | 126 μs | 10867 μs |
|  | 119 μs | 11515 μs |
|  | 118 μs | 10663 μs |
|  | 120 μs | 10914 μs |
|  | 124 μs | 11137 μs |
|  | 137 μs | 10714 μs |
|  | 123 μs | 10867 μs |
|  | 120 μs | 10670 μs |
|  | 119 μs | 10774 μs |
|  | 120 μs | 10708 μs |
|  | 125 μs | 10694 μs |
|  | 122 μs | 10560 μs |
|  | 122 μs | 10294 μs |
|  | 121 μs | 10378 μs |
|  | 120 μs | 10154 μs |
|  | 119 μs | 10620 μs |
|  | 119 μs | 10507 μs |
|  | 120 μs | 10684 μs |
|  | 121 μs | 10460 μs |
|  | 118 μs | 10558 μs |
| Gemiddeld | 121 μs | 10686 μs |

## Verwerking

Hieronder bevindt zich een grafiek met een gemiddelde opslagtijd van hetzelfde beeld bij de verschillende methoden.

## Conclusie

Uit dit onderzoek kunnen we concluderen dat de 1-dimensionale array efficiënter werkt dan de 2-dimensionale array.

## Evaluatie

Onze hypothese klopt. De hypothese was gemaakt op basis van de beschikbare kennis van assembly. Een 1-dimensionale array maakt minder geheugen aanvragen dan een 2-dimensionale array. Van een 2-dimensionale array bevat de eerste array alleen pointers naar de tweede array. In de tweede array zijn de pixels en informatie opgeslagen. Dus als een pixel veranderd moet worden kost dat meer instructies. Daarnaast kan er bij een 1-dimensionale array meer snelheid geboekt worden met het kopiëren van data.