Implementatieplan: ImageShell en Intensity

Agterberg, Ole – 1651981

Stoeltie, Ferdi – 1665045

Image: http://bubble.ro/How_to_convert_an_image_to_grayscale_using_PHP.html

1. Doel

De opdracht bestaat uit twee doelen. Het eerste doel is om Imageshells te maken voor RGB en intensiteit. Hiermee kunnen alle pixels van een afbeelding opgeslagen worden. Het tweede doel is om RGB om te zetten naar intensiteit (Grayscale). Dit zorgt er voor dat latere image processing stappen beter werken. Het tweede doel is afhankelijk van het eerste doel. Om deze doelen te realiseren zijn de volgende punten opgesteld:

- Documenten opstellen bestaande uit: Diagrammen, modellen, code en bijbehorende documentatie;
- Maken Imageshell klasse voor RGB en intensiteit;
- RGB-waarden (pixels en afbeelding) om te zetten naar grijs-waarden (Grayscale).
    - Onderzoek doen naar conversie van RGB-waarden naar grijs-waarden;
        - Verschillende methoden vergelijken en testen.
    - Testen in de GUI applicatie;
    - Het maken van een ImageShell klasse voor RGB en voor Intensiteit.

2. Methoden

Zoals hier boven beschreven zijn er twee doelen. Om de doelen te behalen zijn ook twee verschillende methoden nodig.

2.1. Data opslag

De pixel data van de image kunnen op verschillende manieren opgeslagen worden, zoals;

- Array
    - Snel opzoeken van elementen;
    - Snel opslaan van elementen;
    - Constante grootte.
- Vector;
    - Toevoegen van elementen kan lang duren;
    - Dynamische grootte;
    - Gaat niet efficiënt om met geheugen;
    - Snel elementen opzoeken.
- List;
    - Toevoegen van element gaat snel;
    - Dynamische grootte;
    - Gaat efficiënt om met geheugen;
    - Opzoeken van elementen duurt lang.
- Set – Een gesorteerde lijst met unieke waarden;
    - Is niet bruikbaar voor deze toepassing.
- Map – Een gesorteerde lijst met key, value;
    - Is niet bruikbaar voor deze toepassing.

2.2. Conversie

Voor de conversie naar grijs-waarden zijn twee methoden;

- Gemiddelde;
  $Grijs = \frac{(R + G + B)}{3}$
  - Snel door weinig berekeningen;
  - Komt niet overeen met werkelijkheid.
- Gewogen gemiddelde;
  $Grijs = 0{,}2126R + 0{,}7152G + 0{,}0722B \ \wedge \ 0{,}299R + 0{,}587G + 0{,}114B$
  - Extra berekening nodig;
  - Komt beter overeen met werkelijkheid.
- Enkele kleur kanaal.
  $Grijs = R \ \wedge \ G \ \wedge B$
  - Is niet bruikbaar voor deze toepassing.

(RGB to Grayscale Conversion, sd)

3. Keuze

3.1. Data opslag

Gekozen is om gebruik te maken van een *std::vector* met een vooraf – op basis van de hoeveelheid pixels - gedefinieerde grootte. Een *std::list* is traag om specifieke elementen op te zoeken. Twee nadelen van een vector zijn het inefficient omgaan met geheugen en het toevoegen van nieuwe elementen. Aangezien er alleen elementen (pixels) worden toegevoegd bij initialisatie vormen de nadelen geen probleem (Wicht, 2012).

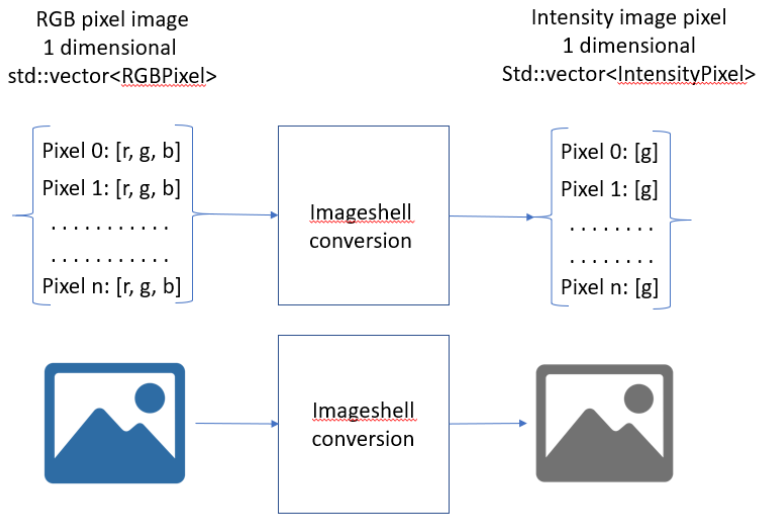
3.2. Conversie

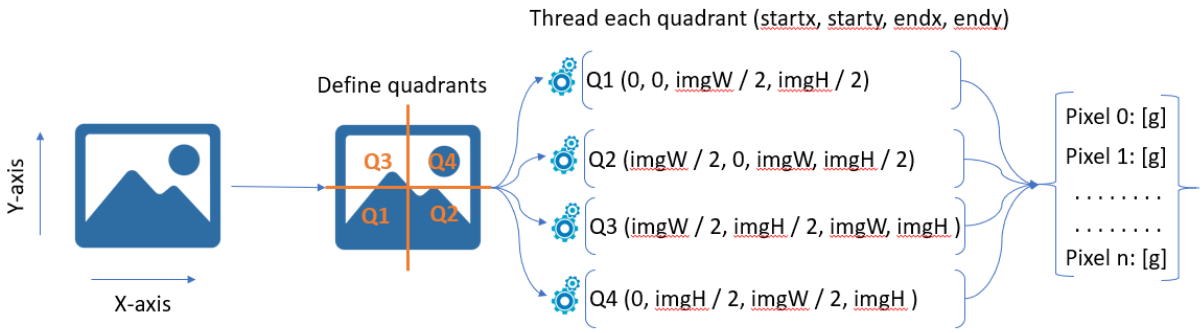
De methode die gekozen voor de conversie is het gewogen gemiddelde. Deze methode zorgt wel voor een paar extra berekeningen, maar het representeert de werkelijkheid beter. (Tutorialspoint, 2020). De keuzen valt voor de exacte formule voor het gewogen gemiddelde is nog niet vast te stellen. Dit is erg afhankelijk van de manier waarop de foto genomen is. Het is erg afhankelijk van de saturatie (woord) per kleur in de foto's. Bij een oververzadiging van rood is het handig om '$0{,}299R + 0{,}587G + 0{,}114B$' te gebruiken omdat de resulterende waarde hierin meer gelijk zijn kijkende naar een histogram. (**Voorbeeld**).

4. Implementatie

De implementatie van de imagesshell is vrij simpel. De klassen hebben een private std::vector met de typen Intensity of RGB. De public functies van de klassen kunnen vervolgens de vector beïnvloeden.

De conversie functie is iets ingewikkelder. De functie krijgt een RGB image die vervolgens wordt omgezet in een Intensity image, door een formule toe te passen op de drie RGB waarden. De RGB image wordt in vier delen opgesplits, zie afbeeldingen:





5. Evaluatie

In de meetrapporten is te zien de snelheid waarmee de image wordt omgezet en wordt opgeslagen verbeterd is, ten opzichte van de default implementatie. Daarnaast is de conversie van RGB naar Intensity versneld door het toepassen van meerdere threads. Vooral bij grootere afbeeldingen is onze implementatie sneller.

6. Verwijzingen

RGB to Grayscale Conversion. (sd). Opgehaald van Profs:
https://profs.info.uaic.ro/~ancai/DIP/lab/Lab_2_DIP.pdf

Tutorialspoint. (2020). *Grayscale to RGB Conversion*. Opgehaald van Tutorials point:
https://www.tutorialspoint.com/dip/grayscale_to_rgb_conversion.htm

Wicht, B. (2012, November 26). *C++ benchmark - std::vector VS std::list*. Opgeroepen op 2020, van Baptiste Wicht: https://baptiste-wicht.com/posts/2012/11/cpp-benchmark-vector-vs-list.html