1. Week 5 02-06-2015 (Scaling)

1.1. Week3 26-05-2015

Gemaakt door Thijs van Tiem & Jos Roijakkers

1.2. Doel

Het doel van deze opdracht is de vision afbeelding te scalen naar een nieuw formaat. Hiervoor moet je kijken naar de mogelijkheden en een keuze maken. Aan de hand van je gemaakte keuze ga je de scaling toepassen met het algoritmen dat je hebt gekozen.

Het totale doel van de opdracht is meer te weten te komen over scaling.

1.3. Hypothese

Wij verwachten dat de code goed uitgevoerd word en gewoon naar de juiste scaling formaat word gezet zonder enige vervormingen in de afbeeldingen. Wel verwachten we dat deze code in het begin wat problemen met zich mee zal brengen maar dat het naderhand gewoon goed zal functioneren.

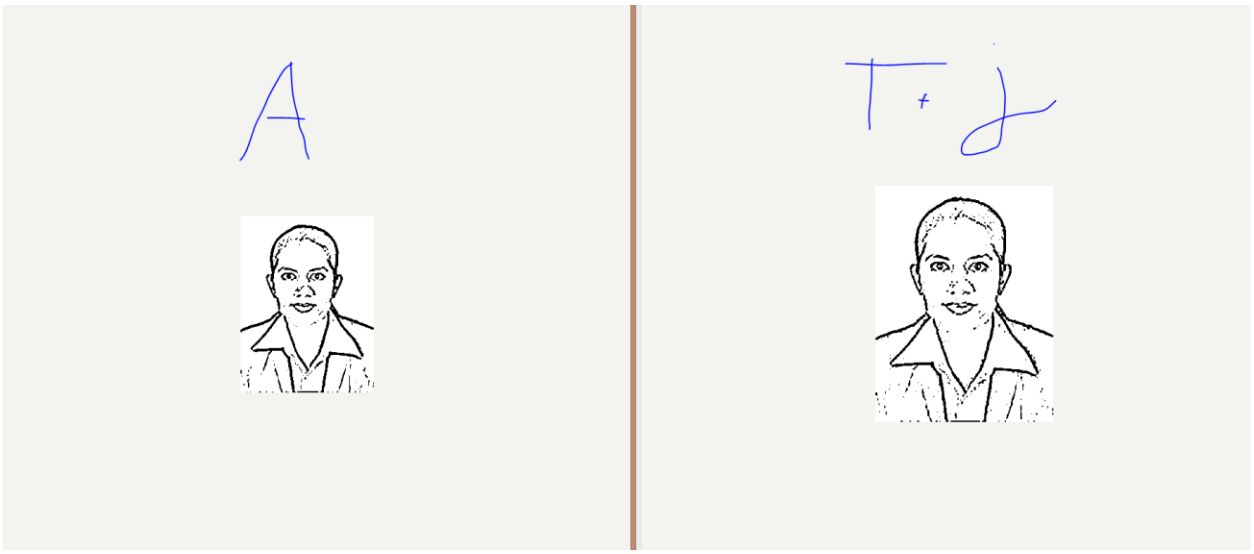
1.4. Werkwijze

We vergelijken de uitvoer met de code van de leraar met die van de student en kijken naar de verschillen in output bij de herkenning. Hiervoor word gekeken naar de gegeven output van de afbeeldingen en van de terug gegeven waardes. Hierna word er gekeken naar de formules en word er gekeken naar de originele code.

1.5. Resultaten

Deze keer word er niet naar de verschillende waardes gekeken in de uitvoer maar of de uitvoer van beide beide codes hetzelfde is. Hiervoor worden dus meerdere resultaten vergeleken en worden de gebruikte formules nog even toegelicht bij onderdeel 1.6.

Code leraar(A) & ons(T+J):



Zoals je ziet zijn al onze afbeeldingen grotere geschaald dan die van de leraar. Wel zie je als je inzoomt dat onze afbeeldingen wanneer ze groter worden meer ruis creëren waardoor de resultaten soms anders uitpakken dan dat ze bedoeld zijn. De afbeelding hieronder laat het beter zien.

Zoals je ziet zijn onze lijnen door de ruis dunner en vallen sommige lijnen daardoor niet even goed uit. Dit is wel een vergelijking op de afbeelding waarbij ze beiden even ver zijn ingezoomd en onze afbeelding oorspronkelijk groter is.

Uit deze resultaten valt dus tijdelijk te concluderen dat onze scaling van ons goed werkt maar meer ruis oplevert wat dus bij grote scaling problemen kan opleveren in de resultaten.

1.6. Verwerking

Het programma werkt op alle afbeeldingen maar zorgt er wel voor dat na de scaling de rest niet meer goed uitgevoerd kan worden. Dit is helaas het resultaat van teveel ruis waardoor de rest van de algoritmen niet meer uitgevoerd kunnen worden. De formules die wij gebruikt hebben in deze opdracht zijn terug te leiden aan de hand van de vermelden bronnen in de code. De formule die we gebruikt hebben werkt als volgt:

```
auto vergrootObject = ImageFactory::newIntensityImage(image.getWidth(), image.getHeight());
const int scaleX = 300; // de groote van de scale ((in de opdracht staat 200x200))
const int scaleY = 300;

double inputSize = image.getWidth()*image.getHeight(); // gewoon totale grote van de image
double realSize = static_cast<double>(scaleX)*static_cast<double>(scaleY); // Static cast om te zorgen dat de ints naar doubles gaan (200*200)
double schaal = 1.0 / sqrt(inputSize / realSize); // berekend de goede schaal

vergrootObject->set(static_cast<int>(image.getWidth()*schaal), static_cast<int>(image.getHeight()*schaal)); // bereken het nieuwe object in breedte en hoogte

for (int x = 0; x < vergrootObject->getWidth(); ++x){
    for (int y = 0; y < vergrootObject->getHeight(); ++y){
        auto Xold = round( x / schaal); //nearestneighbour berekening met round om kans op komma getallen te voorkomen.
        auto Yold = round( y / schaal);
        vergrootObject->setPixel(x, y, image.getPixel(Xold, Yold)); //scale de pixels x,y aan de hand van de oude x en y.
    }
}
```

Eerst zetten de scale van de afbeelding op 300*300 en lopen we alle pixels hiervan door. Daarna worden x en y rond naar beneden afgerond met een nearest neighbour berekening. Dit word gedaan door de schaal door x te delen. Dit word gedaan door 1 /(square root(originele afbeelding / werkelijke groter) uit te voeren. Aan de hand van het resultaat van de berekening worden de nieuwe pixels gezet en word dat object gevuld tot er geen pixels meer zijn. Naderhand word dat object geretourneerd.

1.7. Conclusie

De scaling van ons werkt prima maar levert helaas teveel ruis op om als vision algoritmen gebruikt te worden. Wanneer deze ruis hieruit gehaald zou worden zou het een perfect algoritmen zijn. Wij zijn in ieder geval tevreden met ons resultaat.

1.8. Evaluatie

De hypothese die bovenaan gebruikt word is correct. De code functioneert maar brengt helaas was problemen met zich mee met namen de ruis. Helaas is de gebruikte code dus niet optimaal voor Vision maar zou hij met wat aanpassingen wel gebruikt kunnen worden.