IMAGE SCALING
VERGELIJKING MET DE DEFAULT IMPLEMENTATIE
MEETRAPPORT

FIGUUR 1: ORIGINELE AFBEELDING

FIGUUR 2: AFBEELDING NA CONVERSIE NAAR GRIJSWAARDEN EN SCALING

FIGUUR 3: AFBEELDING NA EDGE-DETECTION

FIGUUR 4: AFBEELDING NA THRESHOLDING

FIGUUR 5: DE GEZICHTSKENMERKEN WEERGEGEVEN OP DE VERKLEINDE ZWART-WIT AFBEELDING

Gemaakt door: Patrick van der Bend

Datum: 13 Februari 2019

Versie: 1.2

Versiebeheer

Versie	Wijzigingen	Datum
1	Eerste versie	9 Februari 2019
1.1	Document visueel mooier gemaakt: - Afbeeldingen op de voorpagina toegevoegd - Typefouten weggehaald	10 Februari 2019
1.2	Typefouten weggehaald	13 Februari 2019

Doel

In dit document wordt de voorbereiding, verwachtingen (hypothese), uitvoering, verwerking en evaluatie van een experiment beschreven. Het doel van dit experiment is het meten van de prestatie van de huidige implementatie van het scaling-algoritme en deze dan te vergelijken met de prestatie van de default implementatie. Specifiek wordt er gekeken naar snelheid en betrouwbaarheid van de algoritmes.

Hypothese

De hypothese is dat de huidige implementatie er ongeveer 100% langer over doet dan de default implementatie, gebaseerd op metingen door Anthony Tanbakuchi in 2016: http://tanbakuchi.com/posts/comparison-of-openv-interpolation-algorithms/ (default gebruikt linear)

Wel wordt er verwacht dat de huidige implementatie het ongeveer net zo goed zal doen als de default implementatie en dat de huidige implementatie minstens 80% zo betrouwbaar is als de default implementatie.

Werkwijze

Bij dit experiment wordt de snelheid en betrouwbaarheid gemeten van twee algoritmes. Om de snelheid te meten zal er een timer geprogrammeerd worden, die voor het beginnen van een algoritme start en wordt gestopt als het algoritme klaar is. Elke meting zal vijf keer gedaan worden en daar wordt het gemiddelde van genomen. De resultaten worden gemeten in nanoseconden.

Om de betrouwbaarheid te meten, wordt gekeken of de rest van het gezichtsherkenning-framework werkt op de afbeelding die verwerkt is door het algoritme. Hoe betrouwbaar het algoritme is, hangt af van hoe goed de resultaten van het framework zijn.

Deze tests worden op zeven afbeeldingen uitgevoerd bij beide algoritmes, dus de tests worden in totaal 14 keer uitgevoerd. De afbeeldingen hebben de volgende dimensies:

Afbeelding naam	Dimensies (B x H)
child-1	225 x 255
female-1	195 x 258
female-2	112 x 149
female-3	198 x 255
male-1	194 x 259
male-2	194 x 259
male-3	198 x 255

De resultaten zullen worden genoteerd onder Resultaten in dit document.

Belangrijk: De default implementatie is lichtelijk aangepast (zie Appendix A) voor dit experiment: door de aanpassing verandert de default implementatie de afbeelding naar dezelfde grootte als de huidige implementatie. Dit is gedaan om de vergelijkingen zo eerlijk mogelijk te maken.

Resultaten

Resultaten bij de default implementatie

Afbeelding	Gemiddelde tijd(ns)	Resultaat framework
child-1	11046943.4	5/5 keer succesvol
female-1	8257984.2	5/5 keer succesvol
female-2	2942936.6	5/5 keer succesvol
female-3	8670981.0	5/5 keer succesvol
male-1	8100231.0	5/5 keer succesvol
male-2	10024456.0	0/5 keer succesvol (Allen faalden bij *Localization step 2*)
male-3	9190607.6	0/5 keer succesvol (Bij alle pogingen werden de ogen niet gevonden)

De totale gemiddelde tijd is afgerond 8319162.83ns

Resultaten bij de huidige implementatie

Afbeelding	Gemiddelde tijd(ns)	Resultaat framework
child-1	72914765.6	0/5 keer succesvol (Allen faalden bij *Localization step 2*)
female-1	67004349.4	5/5 keer succesvol
female-2	1075122.0	5/5 keer succesvol
female-3	66252813.0	5/5 keer succesvol
male-1	72717064.4	5/5 keer succesvol
male-2	74098782.0	0/5 keer succesvol (Allen faalden bij *Localization step 5*)
male-3	67784581.6	0/5 keer succesvol (Bij alle pogingen werden de ogen niet gevonden)

De totale gemiddelde tijd is afgerond 60263925.43ns

Verwerking

Bij dit experiment ging het om de snelheid en betrouwbaarheid van de huidige implementatie in vergelijking tot de default implementatie. Hieronder de resultaten:

Tijd:

Tijd Default implementatie (ns)	Tijd Huidige implementatie (ns)	Tijd huidig / tijd default * 100%
8319162.83	60263925.43	724.40%

Betrouwbaarheid:

Slagingspercentage default	Slagingspercentage huidig	Huidig / default * 100%
71.43%	57.14%	80.0%

De huidige implementatie doet er dus ongeveer 7.25 keer langer over dan de default implementatie. Het slagingspercentage is 20% lager dan dat van de default implementatie.

Conclusie

De conclusie die uit de meetresultaten getrokken kan worden is dat de huidige implementatie niet goed presteert in vergelijking tot de default implementatie. Vooral de performance is veel te laag. Het slagingspercentage van de huidige implementatie is ook lager dan de default implementaties slagingspercentage. Een meetresultaat dat opvalt is het meetresultaat bij afbeelding *male-2*. Bij de default implementatie kwam het framework tot *step 2*, terwijl het framework tot *step 5* kwam bij de huidige implementatie. Het kan dus zijn dat het verhoogde contrast bij de huidige implementatie het makkelijker maakt om bepaalde gezichtskenmerken te herkennen.

Evaluatie

De minimale eisen voor het huidige algoritme, gedefinieerd in het implementatieplan:
Max. 200% de berekentijd vergeleken met de default implementatie
Min. 80% het slagingspercentage vergeleken met de default implementatie

Aangezien de berekentijd 700% keer dat van de default implementatie was, is hier het maximum ver overschreden, terwijl het slagingspercentage op het minimum ligt. Hierom moet er nagedacht worden of de huidige implementatie geoptimaliseerd moet worden, of dat er een ander algoritme gekozen en uitgewerkt moet worden.

Tijdens dit experiment zijn er Meetonzekerheden ontstaan doordat de timer die geïmplementeerd is niet precies start op het moment dat het algoritme begint. De timer start bij de aanroep van een andere functie die een paar flags checkt en dan de algoritmes uitvoert. De timer stop wanneer deze functie een waarde teruggeeft. Hierdoor zit er dus nog wat berekentijd tussen de start van de timer en de start van het algoritme. Er zit ook berekentijd tussen het eindigen van het algoritme en het stoppen van de timer.

Appendix A

In deze appendix worden de veranderingen in de default implementatie beschreven.

De enige verandering was in DefaultPreProcessing.cpp onder de functie:
*IntensityImage * DefaultPreProcessing::stepScaleImage(const IntensityImage &src) const*

Onder de if statement *if (ThoroughBushThoroughBrier < OverParkOverPale)*

Originele code:

```
if (ThoroughBushThoroughBrier < OverParkOverPale){
                double ThoroughFloodThoroughFire = 1.0 / sqrt(OverParkOverPale / ThoroughBushThoroughBrier);
                cv::resize(OverHillOverDale, OverHillOverDale, cv::Size(), ThoroughFloodThoroughFire, ThoroughFloodThoroughFire, cv::INTER_LINEAR);
        }
```

Nieuwe code (veranderingen in het rood):

```
if (ThoroughBushThoroughBrier < OverParkOverPale){
                double ThoroughFloodThoroughFire = 1.0 / sqrt(OverParkOverPale / ThoroughBushThoroughBrier);
                // for testing
                float Bob = std::sqrt((src.getWidth() * src.getHeight()) / 40000.0f);
                int Henk = int(src.getWidth() / Bob);
                int Hans = int(src.getHeight() / Bob);
                // testing code ends here
                cv::resize(OverHillOverDale, OverHillOverDale, cv::Size(Henk, Hans), ThoroughFloodThoroughFire, ThoroughFloodThoroughFire, cv::INTER_LINEAR);
        }
```