Implementatie plan RGB ->Grayscale.

Geschreven door Mike Hoogendoorn & Richard Janssen.

Datum: 21-02-2020

Datum laatste wijziging: 27-03-2020

Inhoudsopgave:

1. Doel: 2

2. Methoden: 2

3. Keuze: 3

4. Implementatie: 3

5. Evaluatie: 3

1. Doel:

Wij hebben code gekregen van een gezichtsherkennings programma. Een stap die er voor nodig is om een gezicht te herkennen uit een afbeelding is het converteren naar een grijswaarden afbeelding.

Ons doel is om deze stap te verbeteren op het resultaat zonder een significant langzamer snelheid of meer geheugen gebruik in vergelijking tot de bestaande functie.

Met resultaat wordt bedoeld dat we gaan kijken of we er voor kunnen zorgen dat een groter bereik van afbeeldingen met het programma werken (zie Evaluatie voor details).

2. Methoden:

Na een uitgebreid literatuuronderzoek naar de meerdere mogelijkheden om RGB om te zetten naar grijswaarden hebben we verschillende drie methodes gevonden. Elk van deze methodes geeft elke basiskleur(R,G,B) een prioriteit. Deze weging van de basiskleuren zorgt ervoor dat je bijvoorbeeld kleuren met veel blauw tijdens de conversie intenser(witter) maakt dan kleuren met veel rood.

Twee van deze methodes hebben wij in onderzoeksverslagen gevonden op research gate. Optie 1 uit het onderzoek van tarun Kumar & Karun Verma geeft rood een prioriteit van 33,3%, groen van 50% en blauw van 16,66% om tot de volgende formule uit te komen:

Iy=0.333*190+0.5*183+0.1666*175 =183.925

Bovenstaande formule is gebaseerd op de formule van het onderzoek van Raja Bala & Karen M. Braun die de prioriteiten rood als 30%, groen als 59% en blauw als maar 11% gebruiken om tot deze formule te komen:

GRAY = 0.30 R + 0.59 G + 0.11 B

Vervolgens komen we uit op optie 3, een formule zonder ook maar enige complexe wiskunde. In deze formule pakken we simpelweg het gemiddelde van de 3 kleuren. (Alle kleuren een prioriteit van 33,3%) Een duidelijk nadeel aan deze methode is dat er geen rekening wordt gehouden met dat er in de meeste kleurenruimtes weinig blauw en heel veel mogelijke groentinten bestaan. Beide bovenstaande methodes wijzen de prioriteiten toe op basis van de hoeveelheid mogelijke kleuren in elke groep (R, G & B)

(R + G + B) / 3 = G

Alle 3 deze methodes (+ de originele implementatie van de docent) zullen worden vergeleken op de volgende vlakken:

1. Voor het menselijk oog op welke van de 4 grijze plaatjes het fijnst “oogt” en op zichtbare details, hiervoor gaan we een poll opzetten op googleforms waarin we aan zoveel mogelijk mensen vragen hun mening door te geven.
2. Voor computer vision worden er op de plaatjes meerdere gezichtsherkenning operaties uitgevoerd, deze zullen worden vergeleken op accuraatheid voor alle 4 implementaties en voor snelheid en de hoeveelheid bewaarde details tussen onze implementatie met de 3 bovenstaande formules in vergelijking met de originele implementatie van de docent.

Research paper 1: Tarun Kuma & Karun Verma

Research paper 2: Raja Bala & Karen M. Braun

3. Keuze:

We hebben gekozen om alle drie de bovenstaande formules te implementeren. Omdat we niet zeker weten welke formule ook een beter resultaat zal leveren in de gezichtsherkennings context. Ook hebben wij deze keuze gemaakt in plaats van verder literatuur onderzoek doen omdat we de implementatie rondom de formule toch al schrijven. Het testen van de verschillende formules zou dan niet veel tijd meer moeten innemen. En dan kunnen wij na onze metingen beter overwegen welke formule we het beste kunnen gebruiken.

4. Implementatie:

De functie voor het converteren van RGB naar grijswaarden zal worden vervangen door onze eigen functie binnen de bestaande code van “Externaldll2017.sln” in het bestand “studentPreProcessing.cpp” in de functie “stepToIntensityImage”. Er zullen geen nieuwe hulpklassen worden gebruikt.

Er kunnen geen realistische aannames gemaakt worden of onze implementatie beter zal zijn omdat de originele code die we vervangen verborgen is. Wij hopen / verwachten dat onze implementatie betere resultaten zal opleveren maar dit kunnen wij niet ondersteunen.

5. Evaluatie:

We willen aantonen dat de bestaande codebase in zijn geheel verbeterd kan worden door de “fundering” ofwel de eerste stap te optimaliseren. Dit is relevant omdat de resultaten van deze eerste stap in alle volgende stappen gebruikt wordt. We gaan de snelheid van de volledige ketting aan stappen meten en vergelijken met het origineel, er bestaat al code die deze snelheid voor ons opmeet.

Ook gaan we de eindresultaten van de oude tegenover de nieuwe code vergelijken op het correct herkennen van de persoon. We zijn van plan om het programma te testen met 5 verschillende personen. Van elk van deze personen nemen we verschillende foto’s (portret, portret met accessoires, portret met drukke achtergrond, foto van langere afstand / lagere resolutie).

We zullen ook een test doen om voor het geheugengebruik en calculatietijd. We hopen / verwachten dat de accuraatheid van de resultaten merkbaar wordt verbeterd. En dat het geheugengebruik en calculatietijd niet sterk zal verschillen.

Alle resultaten (Voor de tests die getallen opleveren.) zullen in plots worden gezet om de verschillen duidelijk weer te geven. Uit deze plots kunnen we ook duidelijk de standaardafwijking (standard deviation.) terugzien. Voor de vergelijkingen uitgevoerd door het menselijk oog zullen de resultaten van de verscheidene gray scaling formules naast elkaar gezet worden, In dit geval geldt wel dat dit voornamelijk een mening bevat, iedereen heeft een voorkeur voor welke grayscaling zij fijner vinden. Hier kan eventueel een poll voor opgezet worden om te achterhalen of het merendeel een duidelijke voorkeur heeft.

Voor gezichtsherkenning geven we elk type foto een bepaald aantal punten. Als een operatie succesvol verloopt. Dan krijgt de implementatie de volledige hoeveelheid punten. Dan gaan wij aan de hand van

deze punten het gemiddelde, variatie, en standaardafwijking met elkaar vergelijken welke implementatie de beste consistente prestaties levert.