1. Implementatieplan Edge-detection

1.1. Namen en datum

Namen: Diana Huisen & Amber Kramer
Datum: 6-3-2020

1.2. Doel

Het doel is om een edge-detection methode te maken om de edges in een gezicht te herkennen. Met edges moet je iets als een outline van het gezicht voorstellen. Hieronder valt niet alleen de buitenste lijnen van het hoofd, maar ook van de onderdelen/features zoals ogen, neus en mond. Dit algoritme zal toegepast worden op bewakingscamera's om verdachte personen snel te kunnen detecteren en identificeren. Dit algoritme moet relatief snel zijn om snel personen te kunnen identificeren. Het moet ook werken bij zowel RGB als zwart-wit afbeeldingen, zelfs als er niet veel verschil in contrast is. De features moeten dan ook gewoon te herkennen zijn. Daarna wordt er een thresholding methode gemaakt, wat vooral bedoeld is om eventuele "ruis" weg te laten vallen door de pixels om te zetten naar alleen zwart of wit.

1.3. Methoden

Er zijn heel veel verschillende methodes. De meeste methodes zijn (kleine) aanpassingen gebaseerd op andere (bekendere) methodes. Hieronder hebben we de meest voorkomende methodes uitgelegd en daar de voor- en nadelen van benoemd.

Sobel

De Sobel methode zet op basis van een klein filter in horizontale en verticale richting de afbeelding om in een nieuwe afbeelding waarop de edges te zien zijn.

Pluspunten:
Deze methode is relatief goedkoop met het aantal berekeningen.
Er is een verbeteringsfilter om differentatie en smoothing toe te passen.
Kan bij zowel RGB als zwart-wit afbeeldingen gebruikt worden.
Meer focus op het midden.

Minpunten:
Werkt alleen met afbeeldingen met weinig/geen ruis.
Langzamer dan Prewitt.
Je moet het filter zowel verticaal als horizontaal los toepassen.

Prewitt

Deze methode is vergelijkbaar met Sobel en wordt gebruikt om horizontale en verticale edges in afbeeldingen te herkennen. Alleen het masker is anders, hier wordt namelijk geen nadruk gelegd op de pixels die dichter bij het midden van het masker liggen.

Pluspunt:
Sneller dan Sobel

Minpunten:
Minder focus op het midden
Werkt alleen met afbeeldingen met weinig/geen ruis en een groot of duidelijk contrast.
Je moet het filter zowel verticaal als horizontaal los toepassen.

Laplacian (+ Gaussian)

De Laplacian methode lijkt op de vorige methodes, maar gebruikt maar een filter in plaats van twee.

Pluspunt:
Je hebt in een keer de edges, in plaats van los horizontaal en verticaal te moeten doen.

Minpunt:
Zeer gevoelig voor ruis. Heeft over het algemeen eerst nog (Gaussian) smoothing nodig.

Canny

Canny is een stuk ingewikkelder dan de vorige methodes. Er moet eerst een Gaussian filter worden toegepast om de ruis te verminderen. Daarna wordt Sobel of Prewitt gebruikt om de algemene edges te vinden. Daarna worden alle edges gecheckt en zwakke pixels worden er uitgehaald. Edges kunnen hierdoor gebroken worden en moeten daarna weer gerepareerd worden. Dit wordt gedaan door per pixel te kijken of het een edge is, en daarna in de omgeving kijken of er nog een edge in dezelfde richting is. Hierbij wordt ook nog rekening gehouden met de threshold, die al in dit algoritme verwerkt is. Je hebt hier dus geen aparte threshold algoritme of formule voor nodig.

Pluspunt:
Meest effectief/beste resultaat

Minpunten:
Zeer complex
Duurt langer

1.4. Keuze

Sobel is relatief snel, er zijn verbeteringsfilters, het kan gebruikt worden bij zowel RGB als zwart-wit afbeeldingen, er is een focus op het midden waardoor we verwachten dat de features (ogen en neus etc.) ook nog goed te zien zijn.

Het grootste nadeel van Sobel is dat de afbeeldingen weinig/geen ruis mogen hebben om goed te werken. Dit is echter bij elk algoritme het geval. Sobel is minder gevoelig voor ruis dan Laplacian en kan beter omgaan met minder grote verschillen in contrast dan Prewitt.

Over het algemeen zou Canny de beste keus voor gezichtsherkenning zijn, maar dit algoritme wordt gezien als zeer complex en werkt niet zo snel als Sobel. Bij bewakingscamera's speelt de snelheid echter wel een grote rol. Vanwege deze reden en het feit dat Sobel aan onze eisen voldoet, hebben we voor dit algoritme (Sobel) gekozen.

1.5. Implementatie

We gaan in onze code eerst de afbeelding omzetten naar zwart-wit. Dit omdat je RGB afbeeldingen makkelijk kan omzetten naar zwart-wit, maar zwart-wit afbeeldingen niet naar RGB.
Daarna passen we het algoritme van Sobel toe. Dit gebeurd onder andere met behulp van de link in de bronnen.
Tenslotte kijken we of er een threshold nodig is en welke instellingen er het beste werken om zo duidelijk mogelijk, maar ook bij veel verschillende soorten afbeeldingen, deze zo goed mogelijk te herkennen.

1.6. Evaluatie

We gaan met zowel zwart-wit afbeeldingen als met RGB afbeeldingen testen of de gezichten herkend worden. Ook kijken we naar afbeeldingen met veel en afbeeldingen met weinig contrast. Er wordt bijgehouden hoeveel er van elke afbeelding herkend wordt.

We gebruiken hiervoor 10 afbeeldingen per categorie, hierbij worden naar de volgende categorieën gekeken:
Afbeelding zelf:

- 20 RGB en 20 zwart-wit.
    - Waarvan beide categorieën 10 wel een gezicht laten zien en 10 niet.
- 10 met ruis en 10 zonder ruis.

Personen/huidskleur:

- 10 blank, 10 donker en 10 met personen met een huidskleur dat er tussenin zit. (voor het verschil in contrast)

Hoe gaan we de snelheid meten?

- Timing tool in Visual Studio, en anders via code (mocht de timing tool niet werken)

Hoe meten we het verschil tussen ons algoritme en het oude algoritme?

- Snelheid
- Resultaten vergelijken (hoeveel van de details zijn er herkend)
    - Gemiddelde resultaten per categorie met differentiatie vergelijken
    - Gemiddelde resultaten per dataset (met verschillende categorieën), die worden omgezet naar een totaal gemiddelde. Op basis daarvan wordt de standaardafwijking berekend en door de confidence interval kan je zien hoe significant de algoritmes zijn.

Bronnen

Duidelijke plaatjes van vergelijkingen tussen de algoritmes zijn hier te vinden:
https://medium.com/@nikatsanka/comparing-edge-detection-methods-638a2919476e

De papers die we hebben gebruikt zijn te vinden op de github:
https://github.com/DianaHuisen/Vision-HU-2020/tree/master/implementatieplannen/working