**Onderzoeksverslag IPASS**

**Inhoudsopgave**

[Inleiding 3](#_Toc11855990)

[Uitleg van het onderzoek 3](#_Toc11855991)

[Lijst van libraries 4](#_Toc11855992)

[OPEN-ALPR 4](#_Toc11855993)

[Beschrijving 4](#_Toc11855994)

[Algoritme 4](#_Toc11855995)

[Methodes 4](#_Toc11855996)

[PYOCR 4](#_Toc11855997)

[Beschrijving 4](#_Toc11855998)

[Algoritme 4](#_Toc11855999)

[Textract 5](#_Toc11856000)

[Beschrijving 5](#_Toc11856001)

[Algoritme 5](#_Toc11856002)

[Methodes 5](#_Toc11856003)

[Onderzoeksresultaten 6](#_Toc11856004)

[Conclusie 7](#_Toc11856005)

[Bronnenlijst en verwijzingen 8](#_Toc11856006)

# Inleiding

Voor mijn Induvidual Propedeuse Assignment voor HBO-ICT Applied Artificial Intelligence zal ik een onderzoek doen naar OCR’s in Python. OCR staat voor Optical Character Recognition, dit is de techniek waarbij tekst uit een foto van tekst door middel van patroonherkenning herkend en opgeslagen wordt. De eerste vormen van OCR komen uit de jaren 20, maar pas in de jaren 70 werd te toepassing van OCR populair.

Het beroepsproduct van mijn IPASS zal een onderzoeksverslag zijn, een plan van aanpak en een poster.

# Uitleg van het onderzoek

Voor het onderzoek van het IPASS zal ik 5 Python-OCR-libraries met elkaar vergelijken en testen op hun prestaties tegen over elkaar. De libraries zullen in 2 verschillende situaties tegen elkaar getest worden, de eerste situatie is het herkennen van tekst in een natuurlijke omgeving[zie afbeelding 1]. De tweede situatie is tekst herkennen uit het bestand “Lorem-Ipsum.pdf”, zie map Bijlagen. Voor het uitvoeren van de tests zal ik de PyCharm idea gebruiken.

Bij iedere test zal de library beoordeelt worden op accuraatheid en op snelheid, hieruit zal een score berekent worden. Na uitkomst van alle onderzoeksresultaten zal een conclusie worden geschreven met een mogelijke aanbeveling van een of meerdere libraries naar de resultaten van het onderzoek.

# Lijst van libraries

## OPEN-ALPR

### Beschrijving

OPEN-ALPR is een library ontwikkelt door het bedrijf openALPR. OPEN-ALPR is geschreven in C++, maar kan ook worden gebruikt in de talen C#, Java, Node.js, Go en Python. Open-ALPR is gebouwd voor het herkennen van kentekenplaten uit natuurlijke foto’s. OpenALPR’s eerste release was in 2014 en was gratis verkrijgbaar tot eind 2015.

### Algoritme

Voor detectie van kentekenplaten in openALPR wordt gebruikt gemaakt van het LBP (local binary pattern) algoritme[2]. Dit algoritme wordt vaak gebruikt voor gezichtsherkenning. Door een kleine aanpassing in de constanten van het algoritme kan het ingezet worden voor het vinden van mogelijke kentekenplaten op de foto’s.

### Methodes

Voor de binarisatie van foto’s wordt gebruikt gemaakt van de Wolf-Jolien methode en Sauvola methode[3]. Dit resulteert in meerdere binaire foto’s voor de beste kans van het vinden van kentekenplaten.

## PYOCR

### Beschrijving

PYOCR is een Python library voor het eerst uitgebracht in mei 2013. Het team van ontwikkelaars bestaat uit 21 mensen en werken er nog regelmatig aan (laatste release was in mei 2019).

### Algoritme

PYOCR maakt gebruikt van een “zelfgemaakt”-algoritme, dat goed van toepassing is bij tekst herkennen vanaf foto’s. Hiervoor is een bijlage te vinden in de map bijlagen onder het bestand “Detecting Text in Natural Scenes with Stroke Width Transform”. Deze methode van tekst herkenning is uitzonderlijk goed voor de toepassing in natuurlijke foto’s en om daar tekst uit te herkennen. Eigenlijk is dit algoritme ook gelijk de methode van het binariseren van de foto, dus zal ik de methode van PYOCR niet uitleggen.

## Textract

### Beschrijving

Textract is een Python interface voor verschillende Python pakketten: tesseract-ocr,ps2text,sox etc.. Textract is gespecialiseerd in het uitlezen van documenten, maar ondersteund ook verschillende foto formaten en kan zelfs tekst uit geluidsbestanden halen[6]. Textract’s eerst release was in juli 2014, maar is sinds juni 2017 niet meer geüpdatet. Voor het onderzoek zal ik alleen gebruik maken van de .jpg functies van textract, deze functie maakt gebruik van Google’s tesseract-ocr. Voor een gedetaileerde uitleg van het fungeren van tesseract-ocr zie “AnoverviewofTesseractOCREngine.pdf” in de map Bijlagen.

### Algoritme

Google’s tesseract-ocr maakt gebruik van Otsu’s algoritme en Otsu’s methode. Het algoritme van Otsu staat ook wel bekend als het Firefly-algorithm[7]. Het Firefly algoritme is geïnspireerd door het knipperende gedrag van vuurvliegjes. De lichtintensiteit wordt geassocieerd met de aantrekkingskracht van een vuurvliegje, de mogelijkheid om groepen te verdelen in kleine groepen en elke subgroep zwerm rond lokale eigenschappen te categoriseren. Daarom vuurvlieg algoritme bijzonder geschikt voor multimodale optimalisatieproblemen.

### Methodes

Voor de binarisatie van een foto wordt gebruik gemaakt van Otsu’s methode. Otsu’s formule is als volgt: .

In deze formule zijn ω0 en ω1  de mogelijkheid van de 2 kleuren classes onderscheiden door drempelwaarde *t.* en zijn de variaties tussen de 2 kleuren classes.

# Onderzoeksresultaten

# Conclusie

# Bronnenlijst en verwijzingen

1. <https://github.com/openalpr/openalpr>
2. <https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b>
3. <https://www-sciencedirect-com.hu.idm.oclc.org/science/article/pii/S0167865511002522>
4. [www.bavtailor.com/wp-content/uploads/2018/10/Lorem-Ipsum.pdf](http://www.bavtailor.com/wp-content/uploads/2018/10/Lorem-Ipsum.pdf)
5. <https://libraries.io/pypi/pyocr>
6. [https://textract.readthedocs.io/en/stable/](https://textract.readthedocs.io/en/stable/index.html)
7. <https://www.hindawi.com/journals/mse/2014/794574/>
8. (<https://www.sciencedirect.com/topics/engineering/firefly-algorithm>)