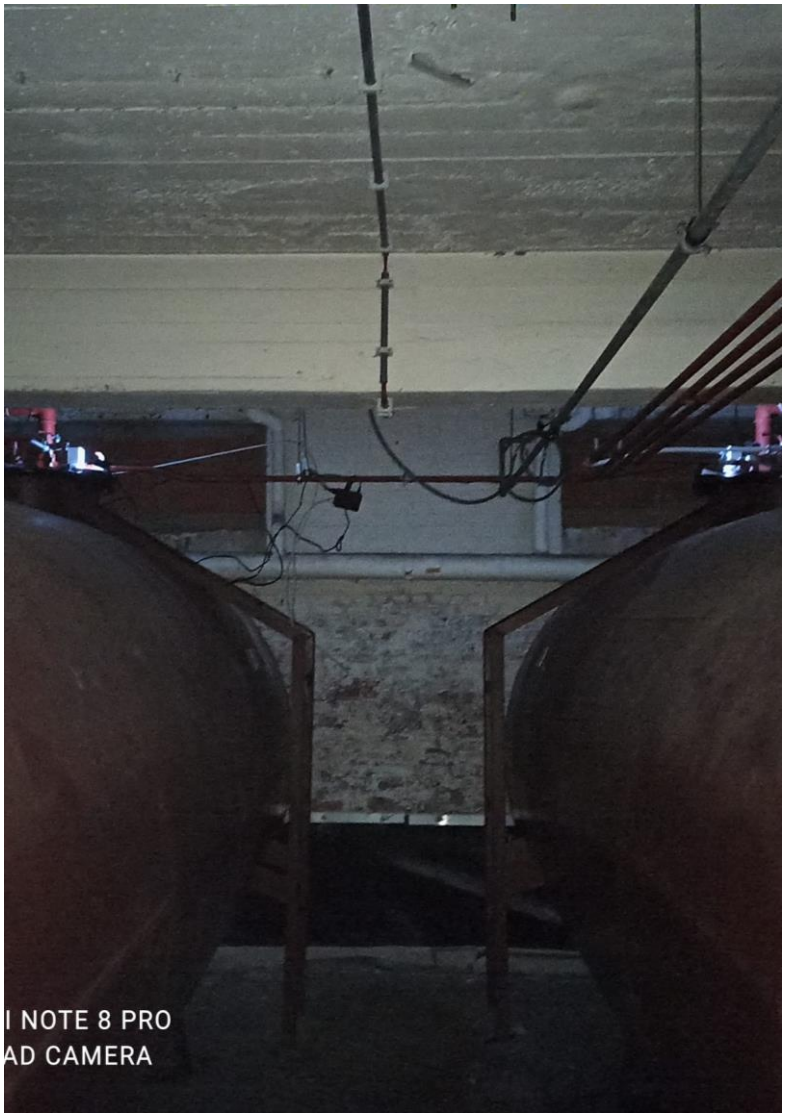


Mazouttank

REALISATIEDOCUMENT

Florian Neise | Stage | 20/05/2021

Inhoudsopgave

1 Doel

Het uiteindelijke doel van dit project is het nemen van een foto van een mazouttankmeter en deze te hosten op een website, daarna wordt er ook een lijnherkenning uitgevoerd om automatisch een mail te sturen als er te weinig mazout is.

In dit document ga ik de vooruitgang en de verschillende stappen van het project uitleggen en beschrijven.

2 Client side

De client side is in dit geval de Raspberry Pi waarop de Raspberry Pi OS draait, aangezien het eerst de bedoeling was alles op de raspberry pi te doen bestond de eerste versie van het programma, uit het opnemen van een foto en een directe lijnherkenning waarop dan een mail kan worden gestuurd. Aangezien dit last minute werd veranderd ga ik hieronder de aparte stukken van het vervolledigde programma in chronologische volgorde bespreken. De uitleg van de code staat in commentaar (alles met een "#" ervoor) mee in de screenshots.

2.2 foto's opslaan

Het volgende dat ik heb gedaan was het opslaan van een frame die ik heb opgenomen via de cameras die op de raspberry pi zijn aangesloten

```
#Importeren van de cv2 library -> openCV gebruikt voor
#alles omrent beelden en beeldverwerking
import cv2 as cv

#de videocapture wordt een variable toewijzen en dan wordt
#de gelezen frame opgeslaan in de variable frame
cap = cv.VideoCapture(0)
_, frame = cap.read()

#ten slotte wordt de zojuist in een variable opgeslagen frame
#naar een png bestand in de map "Resources" weggeschreven.
cv.imwrite('Resources/meterstand.png', frame)
```

2.3 doorsturen van foto

Nadat de beelden werden opgeslagen in een png bestand gaan ze worden doorgestuurd naar de externe server door behulp van volgende code:

```python
#importeren van requests, deze library zorgt ervoor om op een
# heel gemakkelijke manier HTTP requests te sturen
import requests

#in dit deel wordt er een informatie(de naam)
# van het doorgestuurde bestand mee gegeven.
info = {'tank': 'links'}

#Hier wordt de file meterstand.ng geopend en naar de vooraf
# gedefinierde ip adres/website doorgestuurd

with open('Resources/meterstand.png', 'rb') as fp:
    files = {'image': fp}
    response = requests.post('http://172.16.16.126:5000/image', auth=('DeLift', '[redacted]'), params=info,
                             files=files)
    print(response)
```

Daarmee is het programma dat op de Pi draait klaar. Hieronder is een screenshot van het programma in een stuk voor beide camera's:

```python
import requests
import cv2 as cv


cap = cv.VideoCapture(0)
cap2 = cv.VideoCapture(2)
_, frame = cap.read()
_, frame2 = cap2.read()

cv.imwrite('Resources/meterstand.png', frame)
cv.imwrite('Resources/meterstand2.png', frame2)

info = {'tank': 'links'}
info2 = {'tank': 'rechts'}

with open('Resources/meterstand.png', 'rb') as fp:
    files = {'image': fp}
    response = requests.post('http://172.16.16.126:5000/image', auth=('DeLift', '[redacted]'), params=info,
                             files=files)
    print(response)

with open('Resources/meterstand2.png', 'rb') as fp2:
    files = {'image': fp2}
    response = requests.post('http://172.16.16.126:5000/image', auth=('DeLift', '[redacted]'), params=info2,
                             files=files)
    print(response)
```

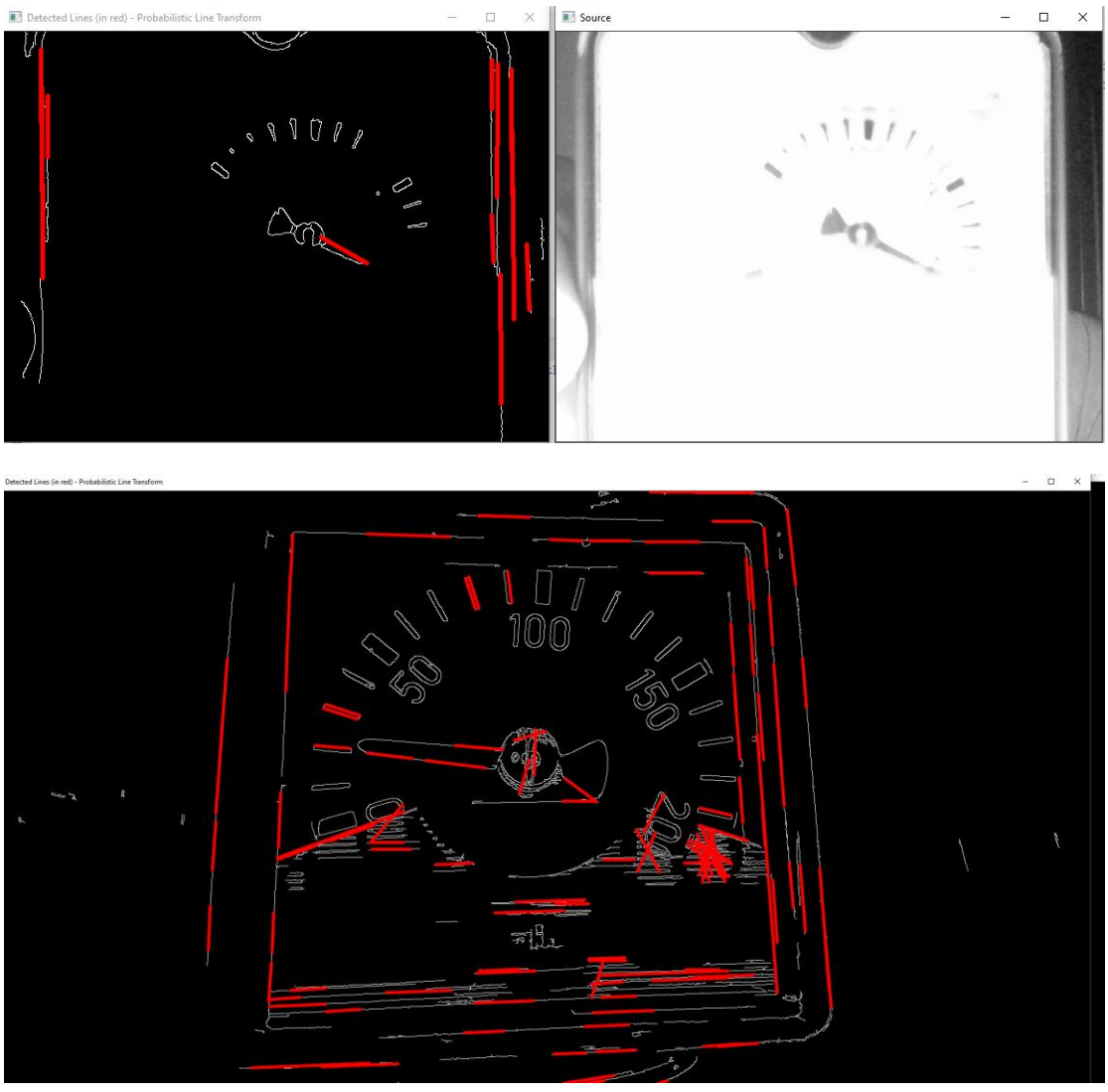
3 server side

3.1 foto's ontvangen

Voor het ontvangen van de foto's heb ik code van mijn stagementor gekregen, deze runt op een server en er kan dan iets naar daar worden gestuurd, zoals ook de rest van dit project is dit programma in Python geschreven.

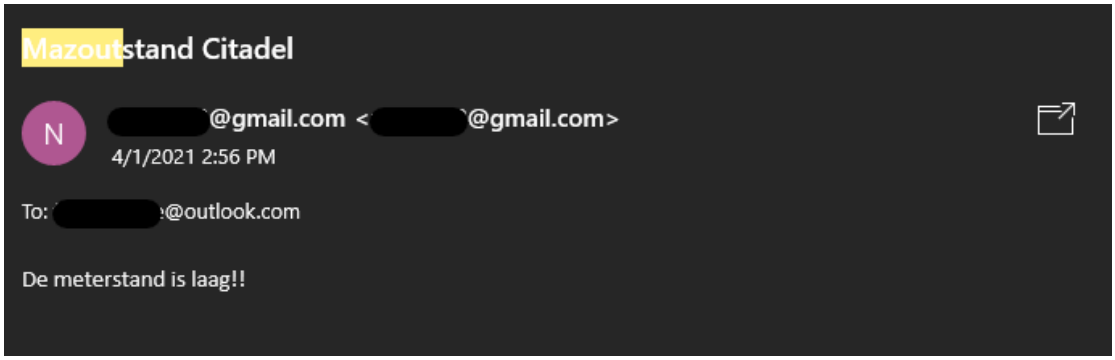
3.2 lijnen herkennen

Nadat de foto's werden ontvangen en in een folder werden opgeslagen als png afbeeldingen, wordt er een lijnherkenning op uitgevoerd zoals in de foto hieronder te zien.



3.3 mail

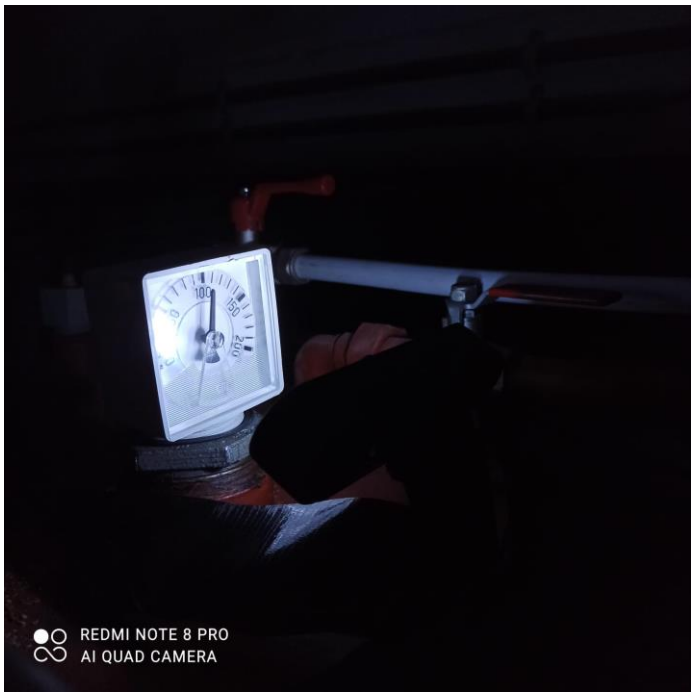
Tenslotte moet er automatisch een mail worden gestuurd als de tank te leeg is zoals hieronder te zien wordt er een bestaand Gmail account gebruikt om de mail te versturen naar mijn eigen outlook account. Een belangrijke stap was het instellen van het account omdat ook minder veilige programma's toegang moesten hebben naar het Gmail account.



4 hardware installatie

4.1 webcams

Bij het installeren van de hardware heb ik eerst de camera's aan de mazouttank vast gemaakt doormiddel van duct tape omdat dit de gemakkelijkste manier was en iedereen er mee content was.



4.2 ethernet aansluiting

Om uiteindelijk de Raspberry Pi met het netwerk te kunnen verbinden, om de foto's door te kunnen sturen, heb ik RJ45 stekkers op een UTP kabel gezet en in de ruimte ernaast aangesloten.

5 sources

De gebruikte tutorials en hulpmiddelen:

https://qengineering.eu/install-opencv-4.5-on-raspberry-pi-4.html

https://www.pyimagesearch.com/start-here/

https://www.pyimagesearch.com/2016/02/08/opencv-shape-detection/

https://www.pyimagesearch.com/2016/02/01/opencv-center-of-contour/

https://medium.com/@nayak.abhijeet1/analogue-gauge-reader-using-computer-vision-62fbd6ec84cc

https://stackoverflow.com/questions/46513323/how-to-read-utility-meter-needle-with-opencv

https://www.pyimagesearch.com/2015/09/21/opencv-track-object-movement/

https://www.pyimagesearch.com/2018/07/19/opencv-tutorial-a-guide-to-learn-opencv/

https://www.youtube.com/watch?v=WQe0O7MI0Bs&t=5247s&ab_channel=Murtaza%27sWorkshop-RoboticsandAI