Inhoudsopgave

Inhoudsopgave 1
Mijn stageplaats 2
1 Mijn stagecontract 3
1.1 Stage-opdracht 3
1.2 Input 3
1.3 Output 3
1.4 Criteria om succes te meten 3
1.5 Wat zal jouw stage-opdracht niet realiseren? 3
1.6 Planning 3
2 Stageverslag 4
2.1 Verslag stageweek 1 & 2 4
2.2 Verslag stageweek 3 & 4 7
2.3 Verslag stageweek 5 & 6 9
2.4 Verslag stageweek 7 & 8 11
3 Tussentijdse evaluatie 13
3.1 Verslag 13
3.2 Werkpunt 13
4 Eindreflectie 14

Mijn stageplaats

Stagebedrijf:	HITlab
Adres:	Graaf Karel de Goedelaan 32
Stagebegeleider bedrijf:	Demanet Jelle
Contactgegevens:	jelle.demanet@gmail.com
Stagebegeleider Howest:	Segers Nathan
Website:	http://www.hitlab.be/

1 Mijn stagecontract

Deze gegevens worden ook aangevuld op https://stage.howest.be

1.1 Stage-opdracht

Bij het HITlab verwerken ze data van zowel vaste als mobiele eyetrackers, het manueel labelen hiervan is een tijdrovend proces. Het project bestaat daarom uit het automatiseren hiervan. Via AI objectherkenning kunnen Area of Interests automatisch gelabeld worden. De huidige software die het HITlab gebruikt heeft nog andere tekortkomingen, er kunnen geen heatmaps mee gemaakt worden die aan de eisen voldoen. Dit project zal deze functionaliteit dan ook uitbreiden in de vorm van Python code.

1.2 Input

De data komt uit de Tobii Pro Labs software in de vorm van een .tsv data export. Alle software zal zodanig geschreven worden opdat ze kunnen draaien op de aanwezige hardware in het HITlab. De software zal voornamelijk bestaan uit python code en enkele ipython notebooks.

1.3 Output

Een programma geschreven in python dat met behulp van object herkenning eyetracking automatisch data kan labelen alsook heatmaps kan maken. De user zal deze software kunnen bedienen door middel van een grafische interface.

1.4 Criteria om succes te meten

Op het einde van de stage zal ik minstens volgende zaken gemaakt hebben.

Installatie en gebruikershandleiding
Python scripts om:

tsv eyetracking data te parsen
heatmaps te maken en overlayen op video files
eyetracking data automatisch te labelen door object herkenning

1.5 Wat zal jouw stage-opdracht niet realiseren?

Mijn opdracht is voldoende afgebakend.

1.6 Planning

heatmap scripts + tsv parsing	04/03/2022
Verschillende object herkenning AI testen en vergelijken	11/03/2022
Area of Interests tracking in mobiele eyetracking videos + tussentijdse evaluatie	29/04/2022
eyetracking data automatisch labelen	20/05/2022
GUI maken + tijd voorzien voor onverwachte problemen / bugs	10/06/2022

2 Stageverslag

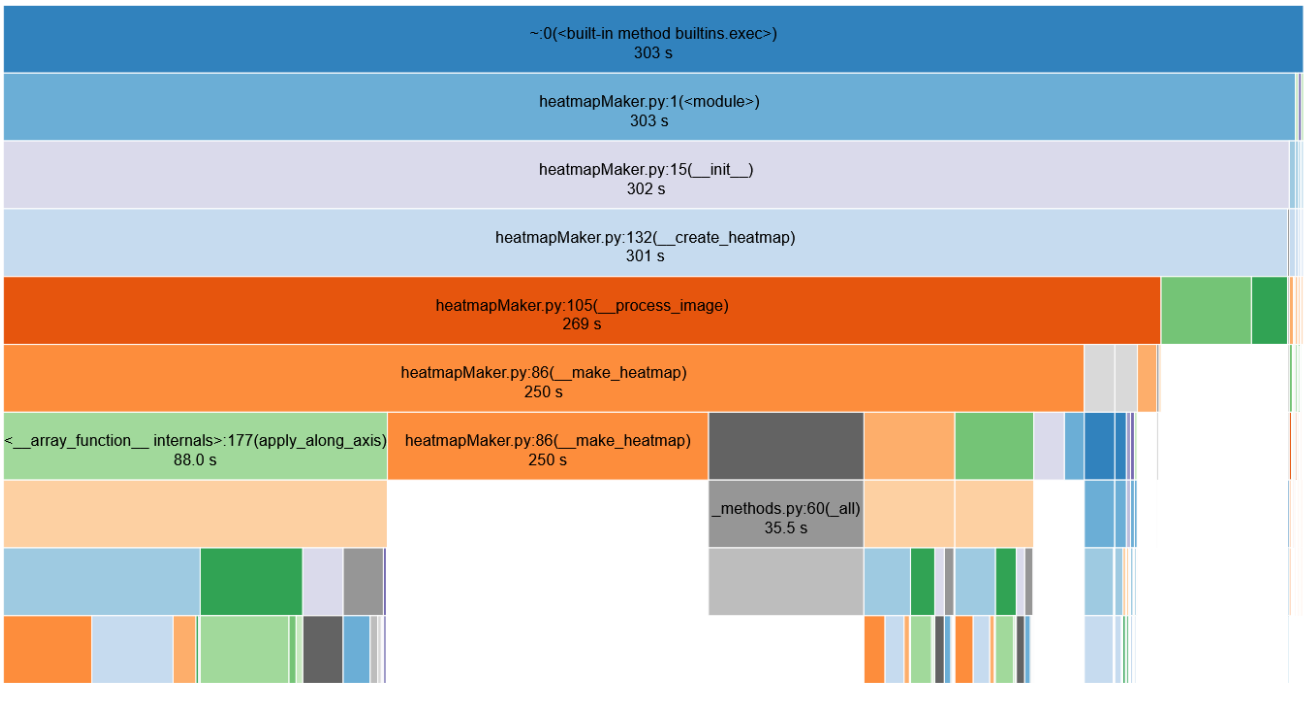
2.1 Verslag stageweek 1 & 2

Realisaties

Tijdens de eerste week van mijn stage heb mij vooral ingelezen bij HITlab en mijn stageopdracht beter kunnen begrijpen aan de hand van real world data. Ik heb de eerste dagen vooral research gedaan naar hoe eyetracking effectief werkt. Meer daarover in het research hoofdstuk.

De 2de week heb ik al een paar mooie dingen kunnen maken. Ik heb al verschillende miniprojectjes gemaakt die de basis zullen vormen voor mijn stageopdracht.

Naam	Wat het doet	Waarom ik het heb gemaakt	Werkt het al?
Blinkdetector	Detecteert oogknippers en logt die naar de console	Eerste test met de pupil labs core eyetracker en API	✓
EyetrackingEffnet	Annoteert objecten in een video van de world camera van de pupil core bril	Word gebruikt door mij om de verschillende efficientnet architecturen te testen op snelheid en accuraatheid	✓
EffNet	Gebruikt google's automl modellen om object herkenning te doen	Eerste implementatie van efficientnet, dit projectje heb ik gebruikt om vertrouwd te geraken met de werkwijze en de API	✓
VasteEyetrackerHeatmap	Creert een heatmap en projecteert deze over de video die de proefpersoon heeft bekeken	Hiermee kan HITlab heatmaps genereren op basis van Tobii Pro Labs tsv exports. Dit zal uitgebreid worden om ook heatmaps van mobiele eyetracking data te maken	✓ (hier en daar nog een paar kleine aanpassingen nodig)
MovingEyetrackerHeatmap	Dit project zal heatmaps kunnen maken van bewegende eyetracking data	Dit is eigenlijk het einddoel van mijn stage, er is nog heel veel werk aan.	✗



Figuur 1 - Optimazing Code

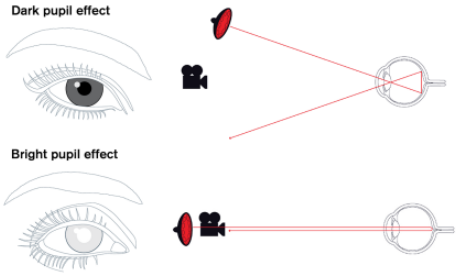
Iets wat ik op school nog niet heb moeten doen is code optimaliseren op snelheid. Op figuur 1 zie je hoelang elke functie call er over doet om uitgevoerd te worden. Hieruit kan je dan afleiding waar je je code moet herschrijven.

Ik heb ook al verschillende meetings kunnen meedoen. Bij een daarvan ben ik naar Waak geweest, zij zijn bezig om met behulp van eyetracking een test te ontwikkelen die hun medewerkers kan helpen ondersteunen, bijvoorbeeld bij het maken van duidelijkere instructies. De eyetracker die ze daarvoor gebruiken is een mobiele eyetracker van Tobii.

Ik heb ook samengezeten met studenten van UHasselt. Zij gebruiken ook een Tobii eyetracker en hadden daar enkele problemen mee.

Research

De eyetrackers waar ik mee werk gebruiken 2 principes om je ogen te tracken, het dark en bright puil effect (figuur 2). Ze zenden infrarood licht in je ogen en meten de weerspiegeling. Afhankelijk van de weerkaatsingshoek wordt bepaald waar je naar kijkt.



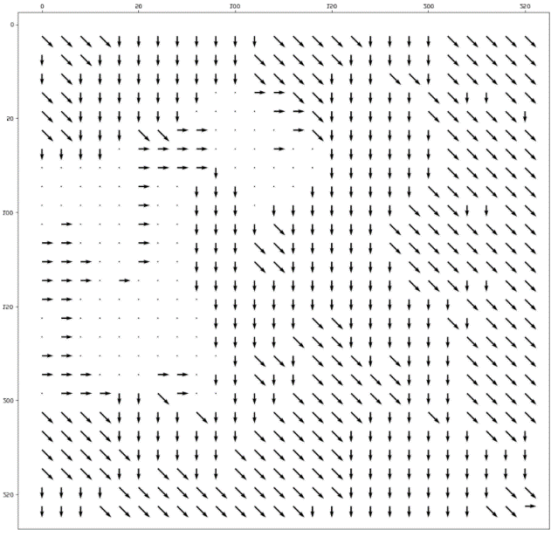
Figuur 2 – Dark vs Bright pupil effect

Je hebt meerdere soorten oogbewegingen. Voor mijn project ben ik vooral geïnteresseerd in fixaties, dit zijn momenten waarin je ogen op 1 vast punt fixeren om informatie te verwerken. Deze fixatiepunten gebruik ik om heatmaps te plotten.

Voor heatmaps te maken van data afkomstig van mobiele eyetrackers is er nogal veel wiskunde nodig. Bij een vaste eyetracker heb je geen 'ego motion'.

Alle datapunten vallen binnen hetzelfde vlak en zijn daar gemakkelijk op te plotten. Het is ook onmogelijk om uit 2d videobeelden de 3d bewegingen van de camera te verkrijgen. Het proces om uit beelden beweging en oriëntatie in te schatten heet Visual Odometry.

Er zijn verschillende methodes om de beweging te verkrijgen, waaronder optical flow. Waarbij de verschuiving van elke pixel gemeten wordt en op die manier een vectorveld kan worden gemaakt (zie figuur 2).



Figuur 3 – Vector field from optical flow

Er is ook veel AI-research over Visual Odometry. De meeste projecten werken met stereocamera's omdat je dan dieptezicht hebt. De mobiele eyetrackers waar ik mee werk hebben maar 1 camera, dit is een extra moeilijkheid.

Een aanpak die ik wel interessant vind heet Unsupervised Structure-from-motion (SfM). Echter ga ik het eerst proberen met de opencv library van python. Hier zitten functies in de onder andere de transformatie en translatie matrices kunnen bereken van 2 frames. Die matrices kan ik dan toepassen op de datapunten.

Feedback

Voor mijn eerste project dat met data van een vaste eyetracker heatmaps maakt kreeg ik als feedback dat dit nog niet helemaal accuraat was. Verschillende punten werd fout geprojecteerd op de video. Ondertussen is dat al verbeterd maar het is nog niet perfect.

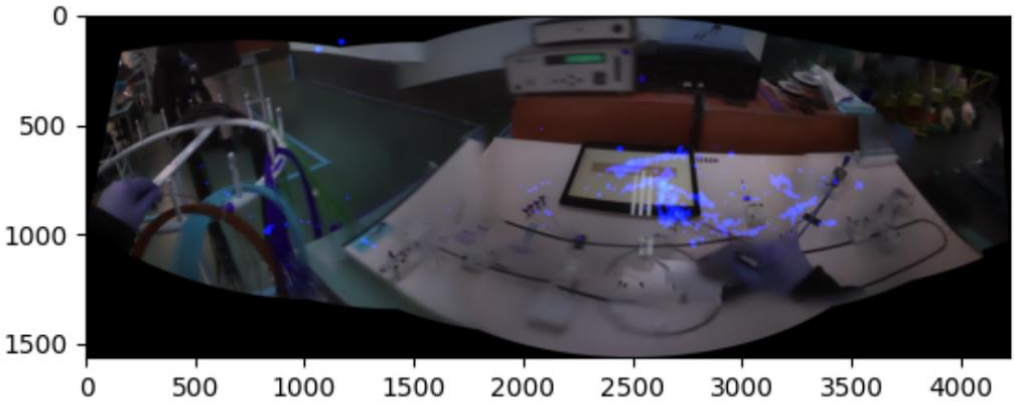
Planning voor de komende twee weken

De volgende twee weken ga ik focussen op de heatmaps van een mobiele eyetracker, dit is niet zoals de originele planning maar ik heb ondervonden dat dit een stuk moeilijker is dan gedacht. Daarom wil ik dit deel eerst afhebben voor ik verder ga.

2.2 Verslag stageweek 3 & 4

Realisaties

Deze 2 weken ben ik erin geslaagd om heatmaps te maken van mobiele eyetracker data. Er is nog een beetje werk de heatmaps er beter te laten uitzien, dit is vooral wat parameters tunen.



Figuur 4 - Heatmap overlay of real-world data

Op figuur 4 kun je het resultaat zien. Het proces om de panorama foto maken vanuit de frames in de input video is heel tijdrovend. Om dit te versnellen heb ik de opencv module zelf gecompiled met Nvidia CUDA ondersteuning, hierdoor kan ik mijn grafische kaart gebruiken om complexe berekening te doen en kan de applicatie in realtime draaien.

```
Queue size: 0     Read 100% of frames
CPU Time: 229.8082811832428
Queue size: 0     Read 100% of frames
0.9831460674157303
GPU Time: 74.30465340614319
```

Figuur 5 - Speed difference CPU & GPU

Ik heb ook een projectmap gemaakt voor de uiteindelijke applicaties met daarin volgende python klasses:

```
├── assets
│   └── placeholder.png
├── Converter.py
├── ConverterGPU.py
├── gputest.ipynb
├── GUI.py
├── HeatmapMaker.py
└── Panorama.py
```

Figuur 6 - folder structure

ConverterGPU verwerkt de ruwe data naar een universeel formaat dat geschikt is om op de panorama te plotten, Converter doet het zelfde maar dan op de CPU, zoals je in figuur 5 kunt zien is de GPU sneller.

De converter klasses erven van Datahandler, die zal bij initialisatie de ruwe eyetracking data verwerken.

HeatmakMaker is de overkoepelende klasse van de converter klasses, deze maakt met de data van de converters de heatmaps

GUI klasses is de klasse waar de gebruikersinterface in gebouwd zal worden.

Er zullen nog klasses bijkomen om analyses op de data te doen, zoals Area of Interests maken.

Research

Deze 2 weken heb ik vooral research gedaan naar gpu programming en de sift en surf algoritmes.

SIFT (Scale Invariant Feature Transform) kan features in afbeelding detecteren, beschrijven en herkennen.

Een goede beschrijving van hoe het werkt kun je hier vinden.

Ik gebruik sift in de Converter klasse om per frame van de eyetracker video keypoints te localiseren en deze te matchen met de panorama foto. Vanuit de 2 lijsten met keypoints bereken ik de transformatiematrix M waarmee ik elke willekeurig punt in het frame kan transformeren naar de panorama foto. Deze matrix pas ik dan toe op de data van de eyetracker.

Dit proces is echter traag en vergt veel rekenkracht. In opencv kun je ook SURF gebruiken, dit is een gelijkaardig aan sift, alleen werkt het veel sneller en kan draaien op de GPU wat het nog sneller maakt.

Een probleem met de GPU gebruiken is de architectuur van de computer. De CPU en GPU hebben alle 2 een apart geheugen en communiceren via een PCI-E bus. Deze communicatie zorgt voor veel overhead. Elk frame moet eerst geüpload worden naar de GPU, daar verwerkt worden en dan terug gedownload worden naar de CPU. Ik heb dit onderzocht en het blijkt dat de GPU gebruiken toch de betere manier is, de extra overhead en de omslachtige manier van code (zie figuur 7) wegen niet op tegen de rekenkracht van de GPU (zie figuur 5).

```
# Load the image onto the GPU
cuMatFrame = cv2.cuda_GpuMat()
cuMatFrame.upload(frame, stream=frame_stream)

# Convert the color on the GPU
cuMatFrame = cv2.cuda.cvtColor(cuMatFrame, cv2.COLOR_BGR2GRAY, stream=frame_stream)

# Detect keypoints/descriptors
kp1, des1 = self.surf.detectWithDescriptors(cuMatFrame, None)
gpu_matches = self.matcher.knnMatchAsync(des1, self.des2, k=2, stream=frame_stream)

# Download the keypoints and matches from GPU memory
kp1 = cv2.cuda_SURF_CUDA.downloadKeypoints(self.surf, kp1)
frame_stream.waitForCompletion()

matches = self.matcher.knnMatchConvert(gpu_matches)
```

Figuur 7 - Code to acces GPU

Ik heb ook een idee om mijn algoritme helemaal te veranderen om het nog sneller en efficiënter te doen draaien.

In plaats van elk frame apart te verwerken zou ik frames per batch op de GPU willen plaatsen, gelijkaardig aan een neuraal netwerk trainen, dit is helaas iets dat in python moeilijk te implementeren valt. Python is te high level hiervoor. Daarom zou ik CUDA-C willen gebruiken, dat is een C taal voor grafische kaarten. Dit zal eerst verder onderzocht moeten worden.

Feedback

De feedback die ik tot nu toe ontvangen heb is vrij positief. Ik werk niet alleen aan mijn eigen project maar heb ook al verschillende keren collega's geholpen met python code te debuggen en hun projecten te testen. Op 18 maart had ik een meeting om mijn stand van staken iets uitgebreider te tonen dan op de team meetings. Daar kreeg ik als feedback de vraag om een 360 graden camera te gebruiken om de panorama's te maken. Dit ga ik deze week onderzoeken of dit mogelijk is.

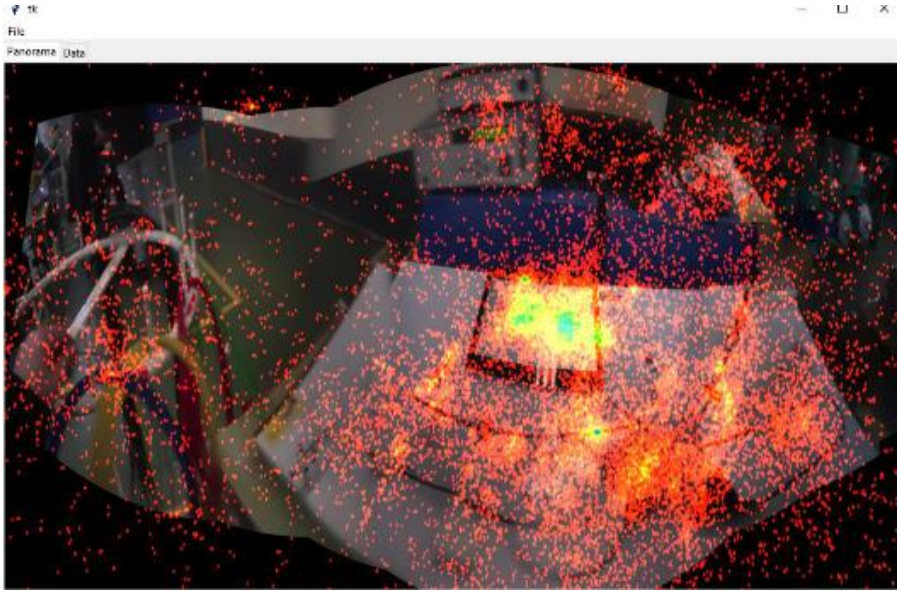
Planning voor de komende twee weken

Ondertussen zit ik terug op schema, de volgende weken ga ik mogelijkheden bouwen om de eyetracker data te analyseren en te onderzoeken. Er staan ook een paar workshops gepland om de projecten van het HITlab voor te stellen.

2.3 Verslag stageweek 5 & 6

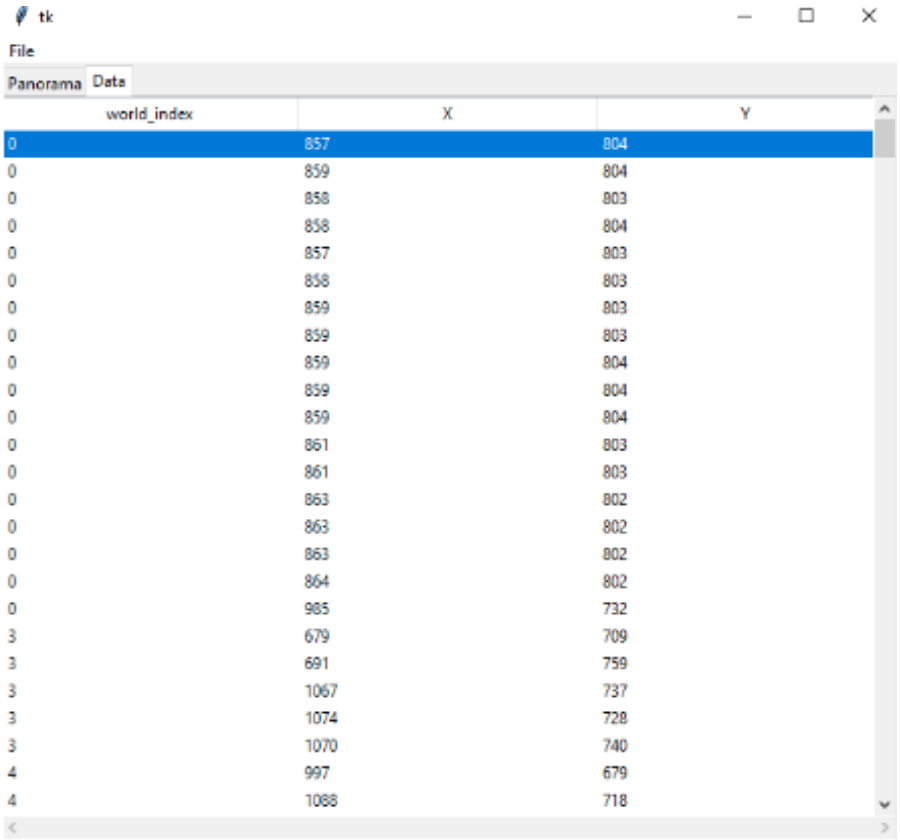
Realisaties

Deze weken ik heb verschillende demo's over projecten van HITLab gegeven en over mijn project gesproken. Uit de demo's hebben we ondervonden dat er grote nood is bij bedrijven om op een efficiënte manier eyetracking data te verwerken. Ik ben nu ook bezig om een applicatie te bouwen om de analyse te doen.



Figuur 8 - GUI

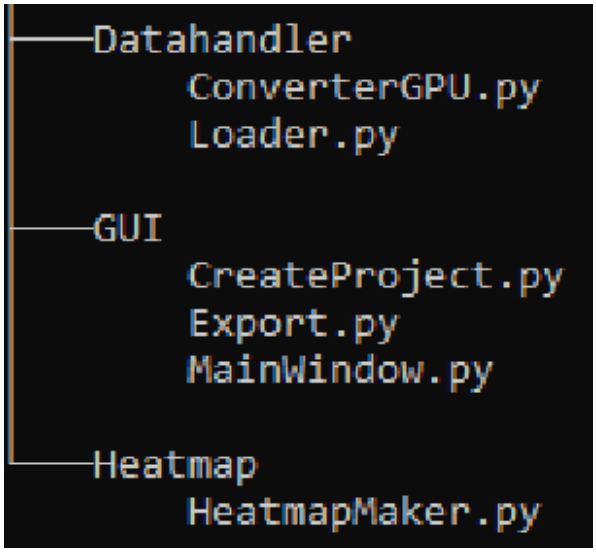
Mijn idee is om de gebruiker in het tabje 'panorama' gebieden op de foto te laten aanduiden waarvoor dan analyses gemaakt worden zoals countplots, heatmaps, aantal fixaties tegenover saccades, enz. De mogelijkheden zijn praktisch eindeloos. In het tabje 'data' kan de gebruiker de ruwe data bekijken.



world_index	X	Y
0	857	804
0	859	804
0	858	803
0	858	804
0	857	803
0	858	803
0	859	803
0	859	803
0	859	804
0	859	804
0	859	804
0	861	803
0	861	803
0	863	802
0	863	802
0	863	802
0	864	802
0	985	732
3	679	709
3	691	759
3	1067	737
3	1074	728
3	1070	740
4	997	679
4	1088	718

Figuur 9 - Data tab

Momenteel kun je al een nieuw project aanmaken, openen en opslaan. De gebruiker zal gevraagd worden om al de nodige bestanden te selecteren waarna het programma deze in een projectmap steekt. Hierna zal het programma de data transformeren naar het perspectief van de panorama foto.
Ondertussen is mijn projectstructuur weeral veranderd, ik heb alles onderverdeeld per klasse, wat veel logischer oogt dan op figuur 6.



Figuur 10 - Folder Structuur

Research

Ik heb de CUDA-C taal verder onderzocht en dat lijkt me te risicovol om het te proberen te implementeren, maar ik zou toch nog willen weten of ik via python batches kan uploaden en verwerken.
Wel ik heb nagedacht en op mijn stage het idee laten vallen om een centrale server te hebben die alle data verwerkt. Dit zal ook nodig voor veel bedrijven dit niet aan alle werknemers zware laptops en desktop willen voorzien maar hun bestaande infrastructuur willen gebruiken. Ik ben nu aan het onderzoeken of er een manier is die gelijkaardig aan de python Queue library opdrachten kan communiceren, maar dan over het netwerk. Dan worden alle GUI's die gebruikt worden producers en maak ik een consumer klasse die centraal op een server draait, er zouden zelfs meerdere consumers kunnen draaien afhankelijk van het aantal servers.

Feedback

Op de verschillende demo's die ik heb gedaan kwam wel interessante feedback van bedrijven. Een onderwerp dat altijd terugkwam is dat toepassingen met mobiele eyetrackers soms niet praktisch haalbaar zijn omdat de gebruiker altijd hardware moet dragen. Zeker in het geval van een VR omgeving. Ook was er iemand die aan mij vroeg of het mogelijk was om het referentieframe van de panorama foto te kiezen. Ik heb hier zelf nog niet over nagedacht maar het is zeker iets dat ik de volgende periode ga onderzoeken.

Planning voor de komende 2 weken

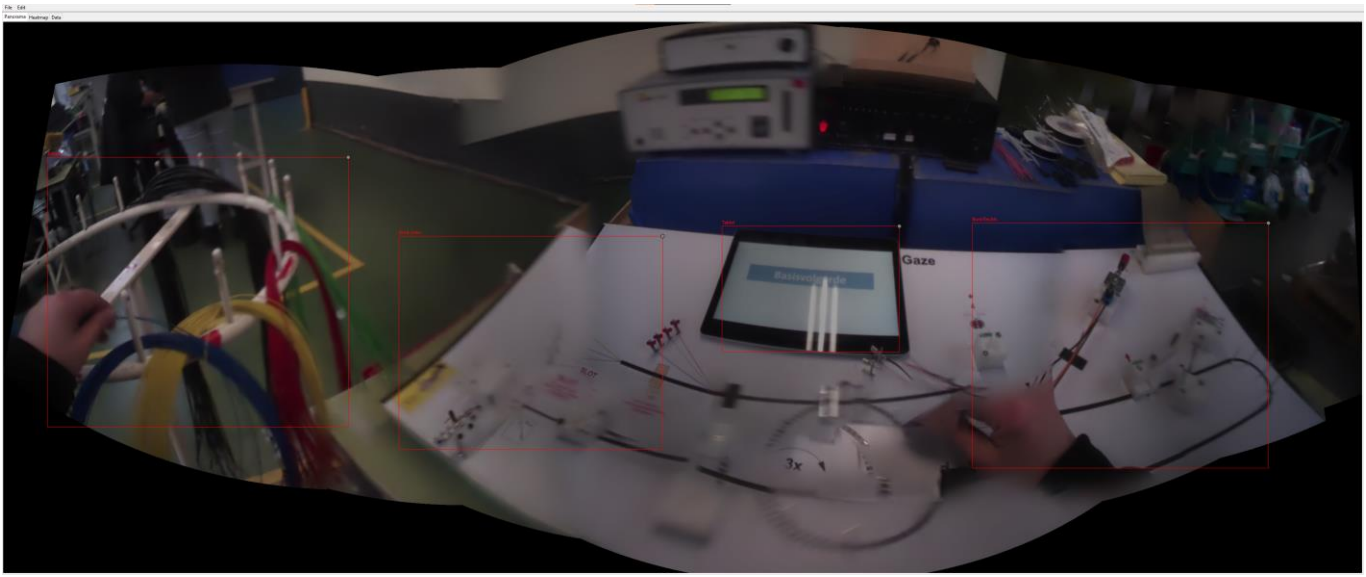
Op een van de demo's van vorige week heb ik met Nathan Segers gesproken over het gebruik van GPU-servers. Na de paasvakantie ga ik code schrijven hiervoor en met hem samenzitten om te kijken of we dit kunnen testen op een server van Howest. Verder moet de GUI applicatie verder gebouwd worden. Ook staat er nog een demo gepland. Ik ga ook nog eens samenzitten met Leatitia van HITlab, zij zit op het eyetracking project en ik zou haar input willen gebruiken voor het design van de GUI applicatie.

2.4 Verslag stageweek 7 & 8

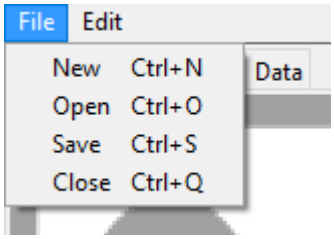
Realisaties

Ik heb de GUI verder uitgebreid, er zijn nu 3 tabjes; panorama, heatmap en data.

Op het panorama tab kan de gebruiker Area of Interests tekenen. Op het tabje AOI waar ik nog aan bezig ben zal dan per AOI verschillende berekeningen en grafieken te zien zijn. Ik heb ook een apart tabje gemaakt voor de heatmap. Hier kan de gebruiker in een oogopslag zien welke gebieden interessant zijn om verder te onderzoeken met de AOI's. Er is ook een eenvoudig menu om een nieuw project te maken, te openen en op te slaan.



Figuur 11 - AOI's



Figuur 12 - File Menu

Verder ben ik bezig met mijn python project om te zetten naar 1 executable. Dan kan ik gebruikers al laten testen. Nu gaat dat niet omdat de source code verschillende DLL's en libraries nodig heeft die gebruiker dan zou moeten installeren, dit is een tijdrovend proces. Het maken van de executable verloopt een beetje stroef en zal verdere research nodig hebben.

Research

Momenteel ben ik aan het onderzoeken hoe ik mijn custom opencv build in een docker image kan steken. Dit verloopt moeilijk omdat ik problemen heb met mijn wsl2 installatie. De bedoeling is om een docker image te maken met gpu support, zodat het gebruik en installatie van de software veel eenvoudiger wordt. Er kan dan ook gekeken worden om de image te deployen op een gpu server. Op HITlab hebben ze de hardware om zelf de software te runnen, maar veel bedrijven met wie ik contact had op demo's hebben geen capabele hardware.

Verder heb ik ook de kwaliteit van mijn data bekeken. Het blijkt dat wanneer de panorama foto die als referentie gebruikt wordt te weinig keypoints bevat het transformeren van de dataset erg lastig is. Ik heb 2 datasets vergeleken, 1 is een opname van 16 seconden, met een duidelijk referentieframe. De andere is een opname van 18 minuten, het bijbehorende referentieframe is gemaakt uit de videodata, er zitten veel scheuren en lijm errors in. Uit mijn test bleek dat de 1ste dataset voor 100% getransformeerd is, de 2de amper 20%. Het referentiekader is dus echt belangrijk, ik raad af om deze te generen uit de videodata. Het is beter om met een moderne smartphone een panoramafoto te maken.

Feedback

Ik ben nog eens bij Waak op bezoek geweest, daar kreeg ik verdere feedback voor hun use case. Ze zouden mijn applicatie qr codes willen laten scannen die in de videoframes van de eyetrackingdata voorkomen. Tegen volgende week hoop ik om ze een eerste build van mijn applicatie te kunnen sturen en meer feedback te krijgen. Ook is er tegen mij gezegd geweest om meer op voorhand te laten weten wanneer ik thuiswerk.

Planning voor de komende 2 weken

De komende 2 weken ga ik proberen om een eerste werkende build van mijn applicatie te maken en die te laten testen. Daarnaast zullen er ook extra functionaliteiten moeten toegevoegd worden aan de AOI's

3 Tussentijdse evaluatie

3.1 Verslag

Het inwerken op de stageplek verliep zeer vlot. Volgend Jelle had ik snel de kern van de opdracht vast en bouwde daar direct op voort. Ik heb project echt van mij gemaakt. Ik heb wel een beetje moeite gehad met GPU-acceleratie technieken. Voor de rest was alles vrij duidelijk. De software die ze op HITlab gebruiken ben ik vrij snel aan gewend geraakt. Ik heb zelfs verschillende keren op mezelf demo's gegeven aan bedrijven. Ik zit goed op schema, soms heb ik wel een paar dagen ergens vastgezeten maar ik ben er steeds in geslaagd om vooruitgang te blijven boeken. Ik kan ook altijd terecht met vragen bij het team. Ik vraag zelf wel te weinig om feedback. Indien ik op tijd klaar ben met mijn project kan ik nog extra uitbreidingen maken. Er zijn op HITlab nog tal van projecten waar ik mee aan de slag kan.

3.2 Werkpunt

Mijn werkpunt is meer communiceren over mijn werk en regelmatiger feedback vragen.

4 Eindreflectie

<Bespreek de eindrealisatie:
- Welke aspecten van jouw stage-opdracht zijn afgerond?
- Welke zaken zijn niet gerealiseerd?
- Welke zaken zijn anders verlopen dan verwacht?
- Wat heb je bijgeleerd?
- Zijn er extra zaken bij jouw takenpakket bijgekomen?
- Schrijf hier ook een dankwoord aan jouw stagebedrijf>