TAXITELLINGEN MET BEELDHERKENNINGSCAMERA'S IN AMSTERDAM

Maarten Groen
Hogeschool van Amsterdam
m.n.groen@hva.nl

Nanda Piersma
Hogeschool van Amsterdam
n.piersma@hva.nl

Hung Chu
Hogeschool van Amsterdam
d.h.chu@hva.nl

Laura Rijlaarsdam
Hogeschool van Amsterdam

January 30, 2020

SAMENVATTING

In dit rapport worden de uitkomsten en geleerde lessen beschreven van een project waarin de toepassing van state-of-the-art Computer Vision algortimes in complexe stadsomgeving is onderzocht. Het project wordt uitgevoerd in de context van het verschonen van het taxibestand in Amsterdam. Op twee plekken in Amsterdam is data verzameld en geanalyseerd. Op een snellaadplek is gekeken naar het gebruik van een viertal laadplekken door taxi's en de mogelijke wachttijden op piekmomenten. Op een taxistandplaats is de doorloop van taxi's onderzocht en de mogelijkheden om een inschatting te doen van het aantal ingestapte personen. Het blijkt dat de tellingen op zowel de snellaadplek als de taxistandplaats een goede indicatie kunnen geven van de drukte en doorloop. Complexere vraagstukken, zoals het bepalen van het aantal instappende passagiers op een taxistandplaats brengt nog veel uitdagingen met zich mee.

1 Introductie

De gemeente Amsterdam neemt steeds meer maatregelen om de taxisector te verschonen. Het doel van deze maatregelen is om de luchtkwaliteit in de stad te verbeteren. Een belangrijke maatregel is om taxi's die niet fossiele brandstof gebruiken, maar elektrisch zijn of op biogas rijden, voorrang te geven op taxistandplaatsen. Op dit moment zijn bijvoorbeeld de taxistandplaatsen Centraal Station en Leidseplein nog slechts toegankelijk voor schone taxi's; diesels mogen er niet meer komen. Er zijn plannen om meer taxistandplaatsen te verschonen en de gemeente Amsterdam vraagt zich af in welke mate het groeiende aantal schone taxi's in staat is om de sterk in tijd variërende aantallen passagiers op die taxistandplaatsen te bedienen. Om deze beleidsvraag te kunnen beantwoorden wordt in dit project onderzocht of er met behulp van Computer Vision algoritmes betrouwbare gegevens over aantallen taxi's en passagiers bij de te verschonen taxistandplaatsen verkregen kunnen worden. Daarnaast wordt onderzocht of met soortgelijke Computer Vision algoritmes de drukte rondom een snellaadplek voor elektrische auto's en de mogelijke wachttijden van taxi's in kaart kunnen worden gebracht. Dit om informatie te verkrijgen of de laadinfrastructuur voldoende is om het groeiend aantal elektrische taxi's aan te kunnen.

Het project onderscheidt zich door de toepassing van Computer Vision algoritmes op real-world casuïstiek, waarbij naast tellingen wordt gekeken of nog meer contextuele informatie uit het beeld kan worden onttrokken. De focus ligt op de praktische implementatie en uitdagingen die er zijn om dit om een robuuste en overdraagbare manier te doen. Naar onze kennis is er nog weinig onderzoek gedaan naar de effectiviteit van hedendaagse Computer Vision technieken in een complexe stadsomgeving met onvoorspelbaar gedrag. Veel onderzoeken richten zich op een deel van het probleem, bijvoorbeeld het beter kunnen detecteren van objecten [1] of het kunnen tracken van meerdere objecten op hetzelfde moment [2]. In stedelijke omgevingen worden deze algoritmes toegepast bij tellingen of snelheidsbepalingen van objecten, maar volgt er vaak geen extra contextuele duiding [3]. Daarnaast is het vaak een omgeving waar objecten op

een voorspelbare manier door het beeld bewegen, zoals op een doorgaande weg [4]. In dit project proberen we juist extra complexiteit, met een mix van logische en complexere situaties, op te zoeken.

Verder ligt er ook nadruk op het snel, bij voorkeur real-time, kunnen analyseren van beelden. In de toekomst worden er idealiter geen beelden opgeslagen van slimme camera's en worden alleen relevante data doorgegeven. Om dit te kunnen doen zullen beelden in real-time geanalyseerd moeten worden. De ontwikkelde software zal in dat geval op schaalbare, en relatief goedkope hardware moeten kunnen draaien. In de afgelopen jaren is er met de doorbraak van verschillende Deep Learning technieken veel vooruitgang geboekt op Computer Vision gebied. Echter, door het real-time aspect van dit project valt de toepasbaarheid van een deel van de state-of-the-art algoritmes af omdat deze momenteel nog te veel rekenkracht vereisen. Bij de wel bruikbare algoritmes is een afweging gemaakt dat deze sneller zijn, maar daarbij ook minder betrouwbaar zijn. Er moet een balans worden gevonden tussen aan de ene kant de nauwkeurigheid van een model en aan de andere kant de snelheid ervan.

Daarnaast kijken we naar de uiteindelijke overdraagbaarheid van de algoritmes. Als een oplossing wordt toegespitst op één situatie, om bijvoorbeeld de nauwkeurigheid van het algoritme op een bepaalde locatie te vergroten, kan het moeilijk zijn om het algoritme over te dragen naar een andere locatie. Bij het overdragen zouden dan bepaalde contextuele regels opnieuw moeten worden bepaald.

In dit rapport bespreken we twee casestudies die in dit project zijn uitgevoerd. De twee casussen gaan respectievelijk over het gebruik van een snellaadplek en het gebruik van een taxistandplaats. Bij beide casestudies is data op locaties verzameld en zijn bestaande state-of-the-art Computer Vision methodes gebruikt om relevante informatie te ontsluiten. De gebruikte technieken, gemaakte keuzes, geleerde lessen en resultaten worden per casestudy beschreven. Afsluitend volgt een algemene discussie.

2 Gerelateerd werk

In het project worden verschillende technieken gebruikt die samen het volledige algortime vormen. In deze sectie worden deze bouwblokken die samen uiteindelijke algoritme vormen besproken.

2.1 Object Detection

Object Detection is binnen het project een van de belangrijkste bouwstenen, waar de andere technieken sterk afhankelijk van zijn. Met Object Detection wordt een algoritme getraind om specifieke objecten, zoals auto's of personen, te detecteren op basis van zeer veel voorbeelden.. Het algoritme kan daarna gebruikt worden om in nieuwe beelden deze objecten ook te detecteren. Bij deze algortimes wordt daarbij ook de plek waar de gevonden object zich in het beeld bevinden teruggegeven, de zogenaamde bounding boxes. Voor Object Detection wordt op dit moment voornamelijk Deep (Convolutional) Neural Networks gebruikt [5]. Een andere manier om objecten uit beelden te halen is Instance Segmentation [6]. Hierbij wordt de gehele vorm van de objecten teruggegeven, dit in tegenstelling tot alleen de bounding boxes. Hierdoor kan dus fijnmaziger iets worden gezegd over het gedetecteerde object. Voor deze fijnmazigheid wordt wel vaak snelheid ingeboet ten opzichte van Object Detection algoritmes. Omdat in dit project de snelheid waarop het gehele algoritme zijn taak kan uitvoeren van belang is wordt in dit geval Object Detection algoritmes gebruikt [1]. De andere algoritmes die in het project worden gebruikt bouwen voort op de gedetecteerde objecten. Het is dus belangrijk dat dit onderdeel goed werkt.

Het trainen van een Object Detection algoritme kost veel tijd en is kostbaar. Er moet een dataset aangelegd worden die handmatig moeten worden geannoteerd, waarna de training pas kan beginnen. Door de relatief korte tijdsduur van dit project is gekeken naar algoritmes die al op grote datasets met relevante objecten zijn getraind. Er zijn een aantal voorgetrainde algoritmes die gebruikt kunnen worden om objecten te detecteren [1]. Deze algoritmes zijn al getraind op gestandaardiseerde en geannoteerde grote datasets, zoals COCO [7]. De getrainde gewichten en configuratie van het model kunnen worden gedownload en worden gebruikt om direct uit nieuwe beelden de objecten te detecteren die ook in de eerder genoemde gestandaardiseerde datasets zaten.

Voor dit project is het YoloV3 [8] algoritme gebruikt. Dit algoritme maakt een goede afweging tussen accuraatheid en snelheid. Daarnaast kan het algoritme zowel auto's als personen herkennen en kan daardoor in beide casussen worden gebruikt.

[1] Tensorflow detection model zoo: `https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md`

2.2 Object Tracking

Om gevonden objecten over verschillende frames in een video te kunnen volgen is het nodig om een tracking algoritme te gebruiken. Multiple Object Tracking (MOT) algoritmes [9] proberen op basis van de output van bijvoorbeeld een Object Detection algoritme de gedetecteerde objecten in een nieuw beeld van een video te koppelen aan gedetecteerde objecten in eerdere beelden. Door objecten op deze manier te kunnen volgen kunnen er diepgaandere analyses gedaan worden, zoals tellingen [10, 11]. De grootste uitdaging op dit gebied is het nauwkeurig kunnen analyseren van omgevingen met zeer veel objecten waarbij objecten mogelijk achter andere objecten verstopt zitten [12, 13, 14]. Net als bij Object detection moet er een balans worden gevonden tussen nauwkeurigheid en snelheid. Er zijn eenvoudigere algoritmes, zoals Centroid Tracking of Kalman filters [15] die relatief snel zijn, maar meer problemen ervaren bij de eerder genoemde grote uitdagingen op Object Tracking gebied dan de complexere en betrouwbaardere, maar ook langzamere Deep Learning trackers [16].

2.2.1 Kentekenlokalisatie

Voor de eerste casus is het niet genoeg om alleen te weten waar de auto's zich bevinden in het beeld. Er moet ook een onderscheid gemaakt worden tussen taxi's en auto's met normale kentekens. In Nederland is de indicator die altijd dit onderscheid kan maken het blauwe kenteken van de taxi. Een veel gebruikte techniek om kentekens te vinden en af te lezen wordt Automatic Licence Plate Recognition (ALPR) [17] genoemd. Bij ALPR wordt in een afbeelding eerst de locatie van een kenteken bepaald, om vervolgens met een Character Recognition algoritme het kenteken af te lezen. Er zijn vele onderzoeken gedaan naar ALPR. ALPR wordt al uitgebreid gebruikt, bijvoorbeeld door de overheid om controles te kunnen uitvoeren [18].

Aangezien een kenteken privacygevoelig kan zijn, is er voor gekozen om te onderzoeken in hoeverre het mogelijk is om te kunnen bepalen of een object een taxi is zonder het kenteken af te lezen. In dat geval is alleen het lokaliseren van een kenteken belangrijk, het daadwerkelijke aflezen wordt dus niet gedaan. Door na het lokaliseren de kleur van het kenteken te analyseren, kunnen we voorspellen of het gedetecteerde object een taxi of een normale auto is [19].

2.3 Smart Cities

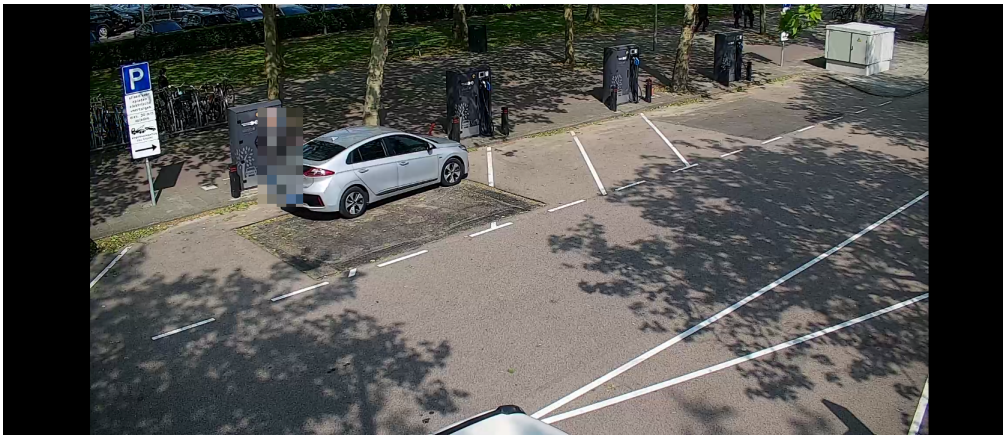
Een mix van de hierboven genoemde technieken wordt al op verschillende manieren gebruikt om verkeer in steden te analyseren. Vooral op het gebied van zelfrijdende auto's gebeurt er veel rondom de toepassing van Computer Vision technieken om bijvoorbeeld de omgeving van de auto in de gaten te houden [20] of de auto in de juiste baan op de weg te houden [21]. Daarnaast zijn er ook onderzoeken waarin een deel van een weg in de gaten wordt gehouden doormiddel van een vaste camera om bijvoorbeeld onregelmatigheden in het verkeer automatisch te kunnen herkennen [4, 22]. Zo kan een inschatting van de snelheid worden gemaakt en kan worden herkend of een auto plotseling stopt. Ook zijn er voorbeelden waarbij kruispunten worden gemonitord om een beter zicht te krijgen op de verkeersstromen en eventuele knelpunten [23, 24]. Naar onze kennis is er weinig onderzoek gedaan naar de toepassingen van dit soort algoritmes om het verloop rondom een snellaadplek of taxistandplaats te analyseren.

2.4 Privacy

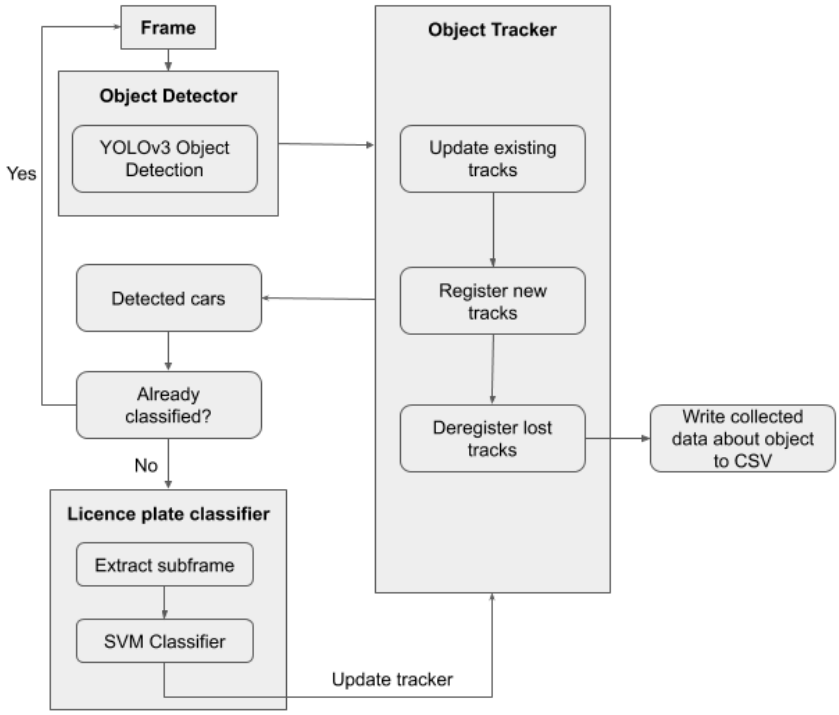
De vooruitgang in Computer Vision toepassingen is mogelijk gemaakt door de toename van het rekenvermogen om de beeldgegevens efficiënter te analyseren. Hoewel er veel mogelijk is met voldoende rekenkracht, zijn de kosten ook hoog. Daarnaast brengt het verzenden van bepaalde gegevens naar de cloud, zoals beeldinformatie verzameld in een stedelijke omgeving, privacy risico's met zich mee. Er kunnen personen of kentekens herkenbaar in beeld komen dat wordt gezien als een persoonsgegeven. Volgens het Privacy by Design principe [25] wordt er standaard gedurende het hele ontwerp en ontwikkelproces van een project proactief rekening gehouden met mogelijke privacygevoelige informatie en hoe daarmee om te gaan. Om de privacy risico's zoveel mogelijk te kunnen uitsluiten wordt daarom in dit project de snelheid waarin een beeld kan worden geanalyseerd als een belangrijk eigenschap meegenomen. Met een algoritme dat snel genoeg is zouden alle analyses in real-time op een lokale machine kunnen worden uitgevoerd. Dit zou betekenen dat er geen beelden hoeven worden opgeslagen of over het internet te worden verstuurd en dat alleen de uitkomsten van het algoritme kunnen worden opgeslagen. Hierdoor wordt ook voorkomen dat er mogelijk beelden met persoonsgegevens worden opgeslagen of doorgestuurd.

3 Casus Snellaadplek

Aan de Europaboulevard in Amsterdam is in mei 2019 een nieuwe snellaadplek in gebruik genomen. Op deze plek zijn vier snellaadstations geplaatst waar auto's maximaal 30 minuten mogen laden. In de periodes van 23 tot en met 30 mei



Figuur 1: Perspectief casus snellaadplek.



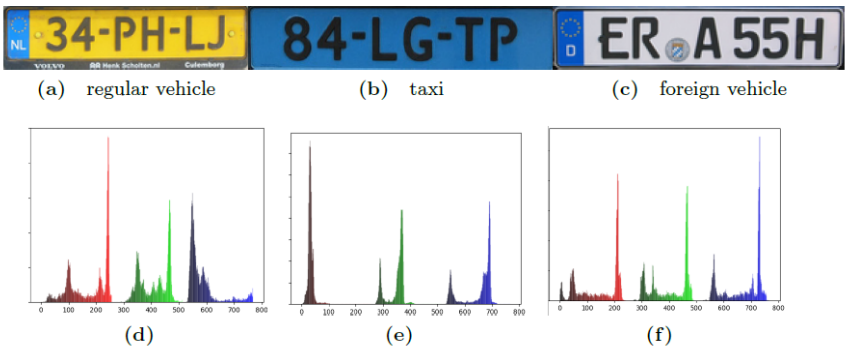
Figuur 2: Overzicht stappen algoritme.

en 3 tot en met 9 september is hier 24 uur per dag videomateriaal verzameld vanuit het perspectief zoals te zien is in figuur 1.

Er is een algoritme ontwikkeld dat auto's kan detecteren, classificeren als taxi of normale auto, en de gevonden auto's kan volgen. Op basis van deze informatie kan data worden vergaard om de beleidsvragen van de gemeente te beantwoorden zoals de hoeveelheid taxi's die gebruik maken van deze snellaadplek en de mogelijke wachttijden als alle laders bezet zijn.

3.1 Algoritme

Het algoritme dat gebruikt wordt in deze eerste casus bestaat uit verschillende samenwerkende algoritmes die ieder een eigen taak hebben. In figuur 2 is een versimpelde weergave te zien van de werking van het algoritme om de relevante data uit het beeld te kunnen halen. De verzamelde data zijn opgedeeld in videobestanden van een kwartier en zijn



Figuur 3: Voorbeelden kleurcodes kentekens.

opgenomen in 25 frames per seconde (fps). Het is mogelijk om elke frame te analyseren, echter is dit funest voor de snelheid van het algoritme. Om de snelheid van het algoritme te verbeteren is besloten ieder vijfde frame te analyseren. Doordat de auto's op deze locatie relatief langzaam rijden verwachten we dat het verlies aan geanalyseerde frames weinig impact heeft op de nauwkeurigheid van het algoritme.

Ieder vijfde frame wordt dus doorgegeven aan het algoritme voor analyse. In de eerste stap wordt uit een frame de auto's gehaald met behulp van een voorgetraind Object Detection model genaamd YOLOv3 [8]. Dit algoritme geeft voor ieder gedetecteerde object een bounding box, een bijbehorend label en een betrouwbaarheidsscore terug. De lijst met objecten met de hoogste betrouwbaarheidsscore die aan onze eisen voldoen worden doorgegeven aan een ander algoritme, namelijk de Object Tracker. De Object Tracker is een Centroid Tracker die ervoor zorgt dat objecten over verschillende frames kunnen worden gevolgd. Er wordt geprobeerd om de gedetecteerde objecten uit het huidige frame te koppelen aan de gedetecteerde objecten in eerdere frames.

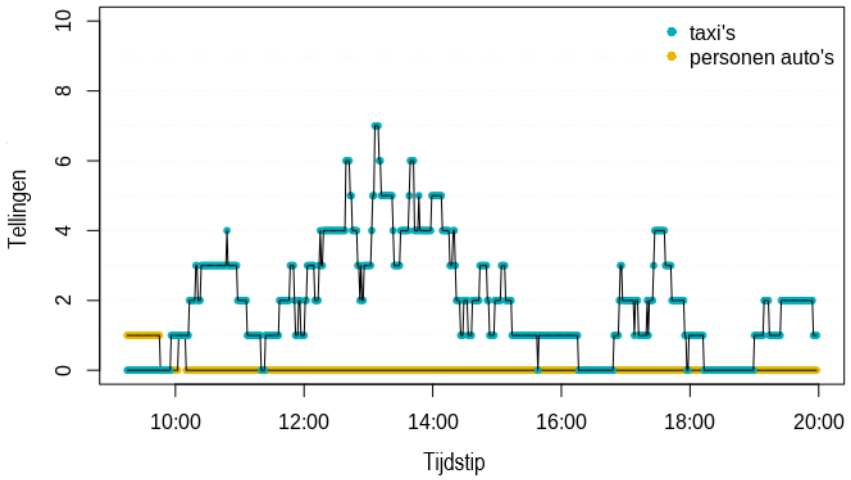
De Object Tracker kan gezien worden als het geheugen van het algoritme. Voor ieder gedetecteerd en gevolgd object wordt relevante data bijgehouden. Via deze tracker is voor ieder object de relevante informatie op te vragen waarop acties kunnen worden gedaan. Dit kan bijvoorbeeld informatie zijn over of het object op een laadplek staat, hoelang deze daar staat en of deze beweegt of niet.

Nadat de tracker volledig is geüpdate op basis van de nieuwe gevonden objecten wordt de lijst met nog bestaande en gekoppelde objecten teruggestuurd. In het algoritme wordt voor ieder object in de tracker gekeken of deze al verder is geclassificeerd. We weten in eerste instantie alleen of het een auto is, maar niet of het een taxi of normale auto is.

Als een object nog niet is geclassificeerd als taxi wordt er een subframe uit het frame gehaald van alleen de gedetecteerde auto. Op dit subframe wordt het WPOD-net algoritme [26] toegepast om kentekens te lokaliseren. Van het gelokaliseerde kenteken wordt voor ieder kleurkanaal (rood, groen en blauw) een histogram gemaakt (zie figuur 3). Op basis van deze histogrammen wordt voor ieder gelokaliseerd kenteken een vector van lengte negen bepaald, bestaande uit het gewogen gemiddelde, het gemiddelde van meest felle kleuren en meest voorkomende kleurwaarde van ieder kleurkanaal. Een lineair regressiemodel (Ridge), dat al getraind is op zulke vectoren om kentekenplaten te classificeren, wordt toegepast om de kentekenplaten te classificeren als taxi, reguliere auto of buitenlandse auto. Als een gedetecteerd object driemaal dezelfde classificatie voorspelling krijgt, dan wordt deze classificatie aangenomen.

Zoals aangegeven wordt voor ieder object dat wordt gevolgd ook nog extra data verzameld. Zo wordt bijgehouden of de auto op een snellaadplek staat, hoe lang deze blijft staan en of deze mogelijk aan het wachten is omdat alle snelladers in gebruik zijn. Zodra een auto uit beeld verdwijnt wordt het object uit de tracker verwijderd en wordt de achterliggende data weggeschreven naar een CSV bestand. In het CSV bestand is de volgende data te vinden:

- ID. Het door de Object Tracker meegegeven unieke identiteitsnummer voor de gedetecteerde auto.
- Classificatie. De door het algoritme voorspelde classificatie van het gedetecteerde kenteken. Dit kan een taxi (blauw), Nederlandse auto (geel), buitenlandse auto (overig) of onbekend (bij infrarood beeld) zijn.
- Tijdstip inrijden beeld. Het moment dat de auto het beeld inrijdt.
- Tijdstip beginnen met laden. Het moment dat de auto in de Region Of Interest (ROI) van een snellader komt (zie figuur x).
- Tijdstip stoppen met laden. Het moment dat de auto de ROI verlaat.



Figuur 4: Voorbeeld drukte op snellaadplek van een dag uit de verzamelde data.

- Tijdstip wegrijden uit beeld. Het moment dat de auto volledig uit beeld verdwijnt.
- Laadtijd: De tijd in seconden dat de auto nabij een snellader in de ROI staat.
- Wachttijd: De tijd in seconden dat de auto moeten wachten voordat een snellader beschikbaar komt bij een volledige bezetting van de snellaadplekken.

3.2 Rapportage

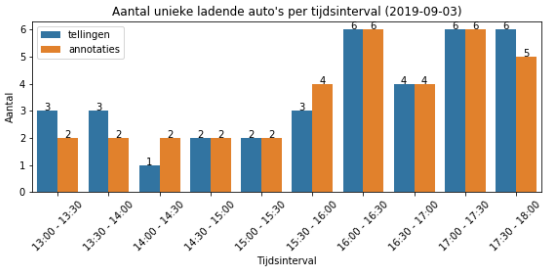
Op basis van het CSV bestand kan een rapportage worden gegenereerd. In deze rapportage kan per dag worden bekeken hoeveel taxi's en auto's van de plek gebruik hebben gemaakt, hoe lang er werd geladen, en of er lang moest worden gewacht. In figuur 4 is een voorbeeld te zien van het gebruik van de snellaadplek voor één dag en de piekmomenten in aantallen getelde auto's.

3.3 Resultaten

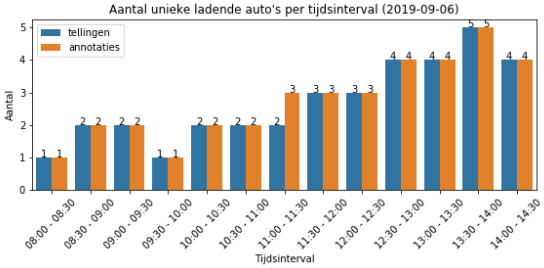
Er is op twee verschillende momenten data verzameld. Het algoritme is ontwikkeld op basis van de eerste week aan data die in mei 2019 is verzameld. De resultaten die hieronder worden beschreven zijn op basis van de toepassing van het ontwikkelde algoritme op basis van de tweede week aan data uit september 2019. In de data uit september is twintig en een half uur aan data handmatig geannoteerd verdeeld over drie blokken in drie verschillende dagen. Deze informatie wordt gebruikt om de uitkomsten van het algoritme te toetsen aan de daadwerkelijke situatie. In figuur 5 is een overzicht te zien van de verschillen in uitkomst van het algoritme (tellingen) en de daadwerkelijke situatie (annotaties) voor de verschillende dagen opgedeeld in blokken van 30 minuten.

De verschillen tussen de uitkomsten van het algoritme en de annotaties zijn op verschillende manieren te verklaren. Er kunnen verschillen ontstaan door fouten die gemaakt worden in de Object Detection of Object Tracking en die vervolgens doorwerken in de rest van het gehele algoritme. Dit kan een kleine impact hebben, bijvoorbeeld in het geval dat de tijd van de telling niet helemaal overeenkomt met de daadwerkelijk situatie doordat het algoritme een auto door lichtomstandigheden even kwijt is. Fouten hebben grotere impact als het Object Detection algoritme door een samenloop van omstandigheden één auto detecteert als twee of meer verschillende auto's. Het algoritme voorspelt dan dus dat het drukker is dan het daadwerkelijk was. In de filters op de output van het algoritme worden ook nog een aantal keuzes gemaakt die kunnen bijdragen aan de gevonden verschillen. Zo wordt een filter ingesteld dat de tijd aangeeft die een auto minimaal op een snellaadplek heeft moeten staan voordat deze wordt meegerekend als een auto die heeft geladen.

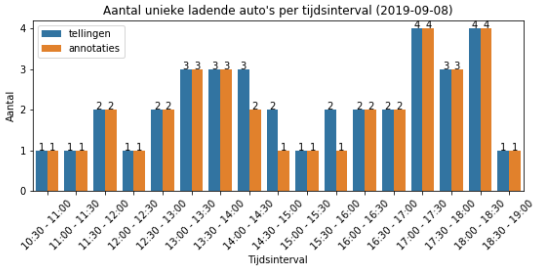
Gekeken naar de resultaten in figuur 5 weet het algoritme de tellingen goed te doen, waarbij soms een afwijking van maximaal één auto te zien is. Zo is op dag 1 in het eerste uur een afwijking te zien. Dit komt door een bus die het beeld van de camera blokkeert (zie figuur 6). Hierdoor kan het algoritme een taxi niet herkennen en dus ook niet tracken. Sommige afwijkingen komen door licht afwijkende berekende eindtijden van laden, waardoor deze ten onrechte niet worden meegenomen in een nieuw blok van 30 minuten.



(a) Resultaat dag 1

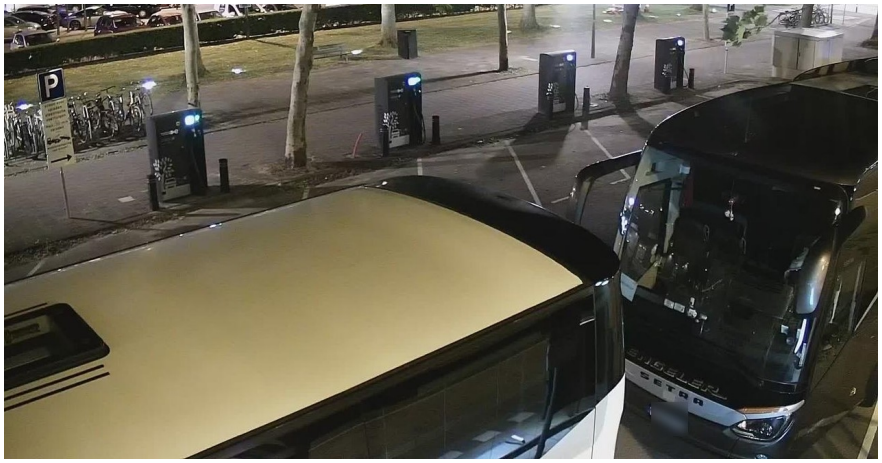


(b) Resultaat dag 2

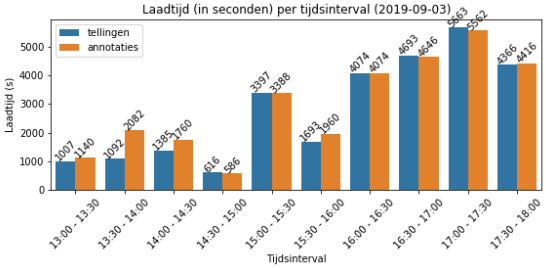


(c) Resultaat dag 3

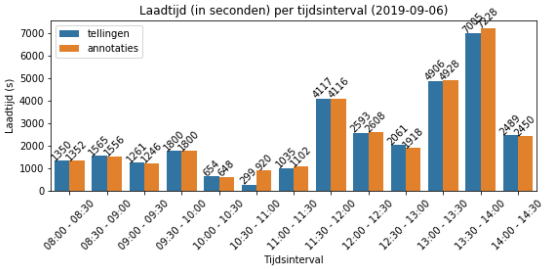
Figuur 5: Overzicht van de verschillen in aantallen tussen de uitkomsten van het algoritme (tellingen) en de daadwerkelijke situatie (annotaties).



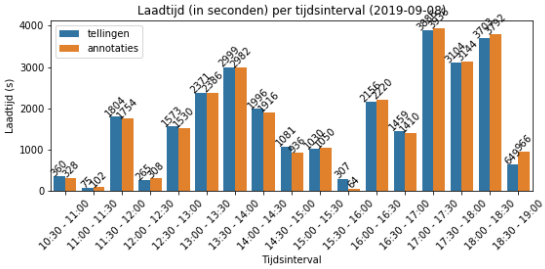
Figuur 6: Blokkade beeld door bus.



(a) Resultaat dag 1



(b) Resultaat dag 2



(c) Resultaat dag 3

Figuur 7: Overzicht van de verschillen in laadtijden tussen de uitkomsten van het algoritme (tellingen) en de daadwerkelijke situatie (annotaties).

Het algoritme maakt soms fouten met meer impact doordat een auto door de lichtomstandigheden twee bounding boxes krijgt en daardoor dubbel wordt geteld Eveneens als er veel auto's aan het laden zijn op de achterste laders vanaf het camerastandpunt gezien kan dit ervoor zorgen dat bounding boxes worden omgewisseld, of dat er auto's dubbel worden geteld. Dit is mogelijk op te lossen door een complexere tracking algoritme te gebruiken dat meer rekening houdt met de historie van de positie van de objecten.

Op dag 3 is tussen 14.00 - 16.00 uur een aantal keer een afwijkende telling te zien. Dit komt door de auto's die kort op een snellaadplek staan en dus na enkele minuten weer wegrijden. Er is een threshold ingesteld van 120 seconden voordat een auto wordt meegeteld. Door deze threshold te verhogen zouden deze fouten kunnen worden gefilterd, met het risico dat daadwerkelijke korte laders niet worden meegerekend.

In figuur 7 is een overzicht te zien van de totale laadtijden per 30 minuten. Ook hier is het verschil te zien tussen de uitkomsten van het algoritme (tellingen) en de daadwerkelijke situatie (annotaties). De grote verschillen aan het begin van dag 1 is te verklaren door de bus die het zicht op een ladende taxi blokkeert, zoals hierboven beschreven. Zie ook tabel 1 voor de totale laadtijd per dag.

In de drie dagen uit de testdata zijn er in totaal 59 auto's geannoteerd, waarvan 49 (83%) taxi's. Het algoritme telde in totaal 62 auto's, waarvan er 48 (77%) als taxi waren geclassificeerd. De verschillen komen door het volledig missen van een auto, door bijvoorbeeld de blokkade van een bus of het dubbel tellen door een fout in het Object Detection algoritme. De classificatie of een auto een taxi kenteken of normale kenteken heeft is in deze testset 100% accuraat. Dus alle auto's die als taxi waren geclassificeerd waren ook daadwerkelijk een taxi. Deze goede resultaten worden ook onderschreven in de scriptie van Otten[19] waar op een grotere testset (374 kentekens) 98.5% accuuraatheid werd behaald op een dataset van dezelfde locatie.

In de verzamelde data en dus ook in de geannoteerde data waren er weinig momenten waarop er meer dan vier autos tegelijk aan het laden waren en er andere moesten wachten. In de testdata is op 3 september in totaal 308 seconden (5 minuten) gewacht en op 6 september in totaal 556 seconden (10 minuten). De tellingen van het algortime kwamen op 3 september uit op 203 seconden en op 6 september op 575 seconden. Op 8 september zijn in zowel de annotaties als de tellingen geen wachttijden geregistreeerd. Er zijn nog meer voorbeelden nodig om hier verdere conclusies aan te kunnen verbinden.

Dag	**Totaal # unieke auto's**		# unieke taxi's		# unieke normale auto's		Totale laadtijd (seconden)	
	Annotaties	**Tellingen**	Annotaties	Tellingen	Annotaties	Tellingen	Annotaties	Tellingen
3 sept	**19**	**21**	18	19	1	2	29614	28899
6 sept	**20**	**18**	16	14	4	4	31872	30560
8 sept	**20**	**23**	15	15	5	8	28824	28871

Tabel 1: Overzicht van de verschillen tussen de uitkomsten van het algoritme (tellingen) en de daadwerkelijke situatie (annotaties) per dag.

3.3.1 Overdraagbaarheid

Om de resultaten van de gedetecteerde en gevolgde objecten te duiden moeten er contextuele informatie worden toegevoegd om iets te kunnen zeggen over wat er in het beeld afspeelt. Veel van de contextuele informatie die moet worden meegegeven is casus specifiek. Het zijn vaststaande regels die iets zeggen over gebieden in het beeld die interessant zijn. Zo moet er worden aangegeven wat de ROI is. In het geval van de snellaadplek is de ROI de plek waar we verwachten dat auto's staan als ze aan het laden zijn. Als een gedetecteerde auto in deze ROI komt nemen we aan dat deze auto aan het laden is. Daarop kan actie worden ondernomen, zoals het doen van tellingen en het bijhouden van tijdstippen.

Bij deze casus is er naast de ROI ook een parameter aangemaakt die aangeeft hoe snel bij het uit het zicht raken van een object deze moet worden vergeten omdat deze echt weg is (en eventueel weggeschreven naar de CSV). Op basis van informatie na het bekijken van de data en de locatie zijn er specifieke regels opgenomen. Bijvoorbeeld:

- Als een auto uiterst links uit beeld verdwijnt verwachten we dat deze weg rijdt en wordt deze snel geregistreerd als 'weggereden'. Als dit niet wordt gedaan is er het risico dat een nieuwe binnenrijdende auto vanaf links het ID van de vorige auto krijgt.
- Als een auto uiterst rechts (doodlopende weg) uit beeld verdwijnt verwachten we dat deze weer terug komt, dus wordt er langer gewacht voordat de track wordt vergeten.

Het is praktisch onmogelijk om met elke situatie rekening te houden. In een complexe stadsomgeving kunnen ontelbaar veel onverwachte situaties optreden. Het toevoegen van een context afhankelijke regel geeft altijd het risico dat het op een situatie voor verbetering zorgt, maar juist in een andere situatie nieuwe problemen geeft. Er zal hier een balans in moeten worden gevonden. Daarnaast zorgt elke toegevoegde regel ervoor dat het algoritme complexer wordt en per locatie aangepast moet worden aan de nieuwe context.

3.3.2 Snelheid

Op een laptop (Intel Core i9 @ 2.90Ghz, 16GB RAM en GeForce GTX 1050 Ti Max-Q) is het mogelijk om beelden in real-time te analyseren. Een 15 minuten durende video duurt gemiddeld 13 minuten en 32 seconden om te analyseren. Het is op dit moment nog niet haalbaar om bij een camera op locatie dit soort software te draaien op goedkopere schaalbare hardware.

3.4 Uitdagingen

Als een plek in de nacht niet goed is verlicht gaat de camera automatisch over naar het opnemen in infrarood (grijstinten) beeld. Zodra dit gebeurt kan de kleur van het kenteken niet worden bepaald. Het is dan niet mogelijk om te classificeren of een auto een taxi is. Er zijn geen andere kenmerken dat voor alle taxi's geldt, zo heeft bijvoorbeeld niet iedere taxi een daklicht. Wel worden er door het algoritme nog steeds tellingen bijgehouden en krijgen deze auto's een aparte label. Op deze manier kan bij drukte wel worden gedetecteerd of er veel auto's moeten wachten. Drukte in de nachtelijke periodes komt in de dataset niet voor. Er is dus een randvoorwaarde voor het 's nachts inzetten van deze techniek om een goed verlichte plek te hebben met een camera die zichzelf hier goed op kan instellen.

4 Casus Taxistandplaats

Op de taxistandplaats bij Centraal Station (De Ruijterkade) in Amsterdam is onderzocht of op basis van beeldmateriaal de doorloop van taxi's bepaald kan worden. Zo wordt er geteld hoeveel taxi's er voorbij komen en hoe lang zij wachten. Daarnaast is er ook gekeken naar de mogelijkheden om tellingen te doen van het aantal personen dat in een taxi stapt. In de periode van 3 tot en met 9 september is hier 24 uur per dag videomateriaal verzameld vanuit het perspectief zoals te zien is in figuur 8. De weg is een eenrichtingsweg, waarbij auto's links aan komen rijden en rechts weer uit beeld

Figuur 8: Perspectief casus taxistandplaats, inclusief ingetekende Region of Interest

wegrijden. Van mensen die een taxi zoeken wordt verwacht dat zij in de voorste taxi stappen. Daarna mogen de taxi daarachter een plekje opschuiven. Nieuwe taxi's sluiten achteraan aan.

De resolutie van het oorsponkelijke beeld is 1920x1080 pixels. Aangezien we geïnteresseerd zijn in de daadwerkelijke taxistandplaats onderin het beeld, wordt alleen een deel van de afbeelding geanalyseerd. Hiermee is de uiteindelijk resolutie van het beeld waar het algoritme mee werkt 1920x580 pixels.

4.1 Algoritme

Het gehele algoritme bij deze casus werkt met grotendeels dezelfde bouwblokken als de eerste casus zoals te zien in figuur 2. Hetzelfde Object Detection algoritme wordt gebruikt, al worden in dit geval naast informatie over de in het beeld bevindende auto's ook informatie van alle personen in het beeld teruggegeven. Daarnaast worden er naast de tracker die informatie verzameld over alle gevonden auto's een andere complexere tracker geïnitialiseerd op basis van Kalman Filtering [15] die betere voorspellingen kan doen over waar een persoon zich waarschijnlijk in een volgend beeld bevindt. In deze casus komt het voor dat er zeer veel personen rondom de taxistandplaats lopen en daarnaast ook regelmatig in onvoorspelbare patronen. Om deze reden is er gekozen om een algoritme in te zetten die in theorie beter met deze omstandigheden kan omgaan, maar daarbij ook nog snel werkt.

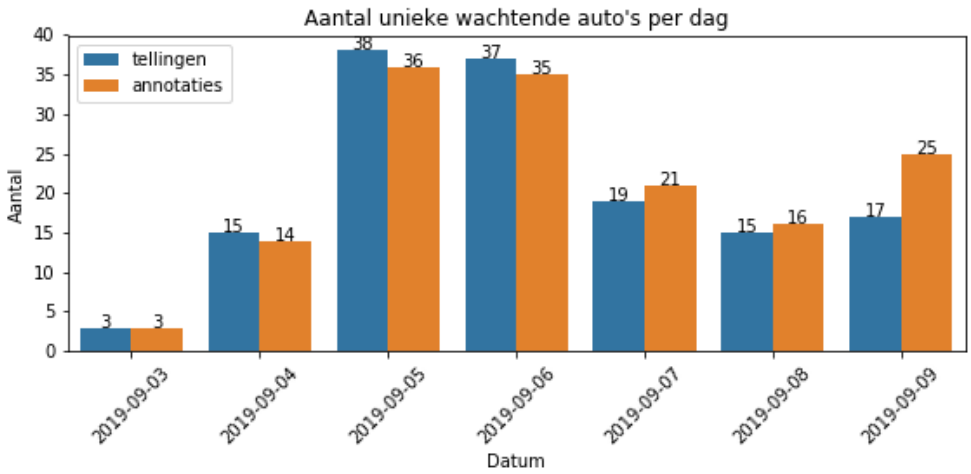
De gevonden auto's en personen in ieder nieuw beeld worden aan de trackers meegegeven om de objecten over tijd te kunnen volgen. Omdat alleen taxi's op deze locatie mogen staan wordt er in deze casus vanuit gegaan dat alle auto's die op de taxistandplaats inderdaad taxi's zijn. De kentekenlokatie en bijbehorende kentekenkleur worden dus in dit geval niet bepaald.

Er is een Region Of Interest (ROI) ingesteld rondom de vier taxi taxistandplaatsen zoals te zien in figuur 8 waar in dit geval drie taxi's wachten. Zodra een taxi de ROI inrijdt wordt er bijgehouden hoe lang deze zich daar bevindt. De ROI omvat dus alle vier de plekken, pas als een taxi daarbuiten komt wordt de tijdstelling stopgezet. Zodra een taxi aan de rechterkant uit beeld verdwijnt worden de resultaten weggeschreven.

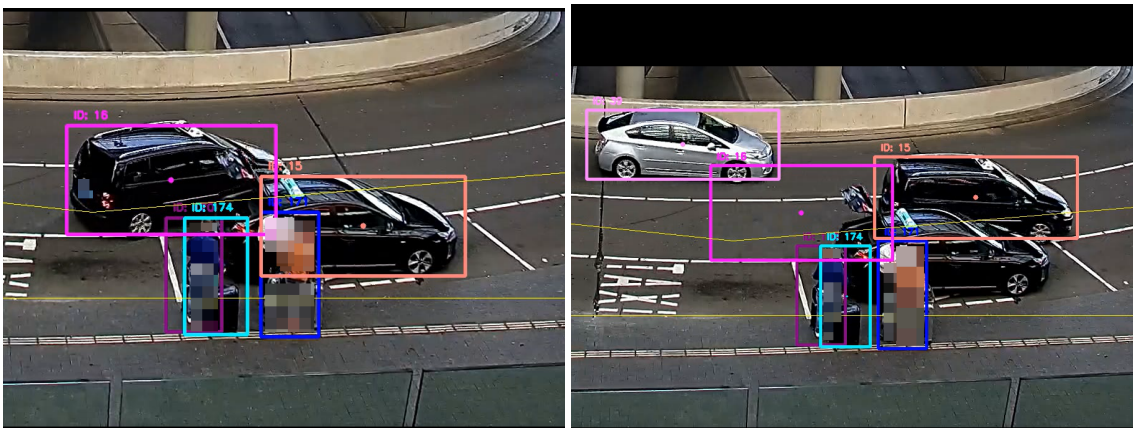
Terwijl een taxi aan het wachten is wordt er ook een voorspelling gemaakt van het aantal personen dat is ingestapt. Zodra het tracking algoritme, dat personen volgt, iemand enige tijd kwijt is geraakt wordt er gekeken of deze zich in de buurt van een wachtende taxi bevindt. In het meest naïeve scenario wordt er dan vanuit gegaan dat deze persoon in de taxi is gegaan. In sectie 4.3 wordt ingegaan op verschillende invalshoeken hoe verder naar dit probleem is gekeken.

Zodra een auto uit beeld verdwijnt wordt het object uit de tracker verwijderd en wordt de achterliggende data weggeschreven naar een CSV bestand. In het CSV bestand is de volgende data te vinden:

- ID. Het door de Object Tracker meegegeven unieke identiteitsnummer de gedetecteerde auto.
- Wachttijd. Het moment dat de auto het beeld inrijdt.
- Aantal ingestapte personen.



Figuur 9: Overzicht van aantal getelde taxi's door algoritme (tellingen) en de annotaties.



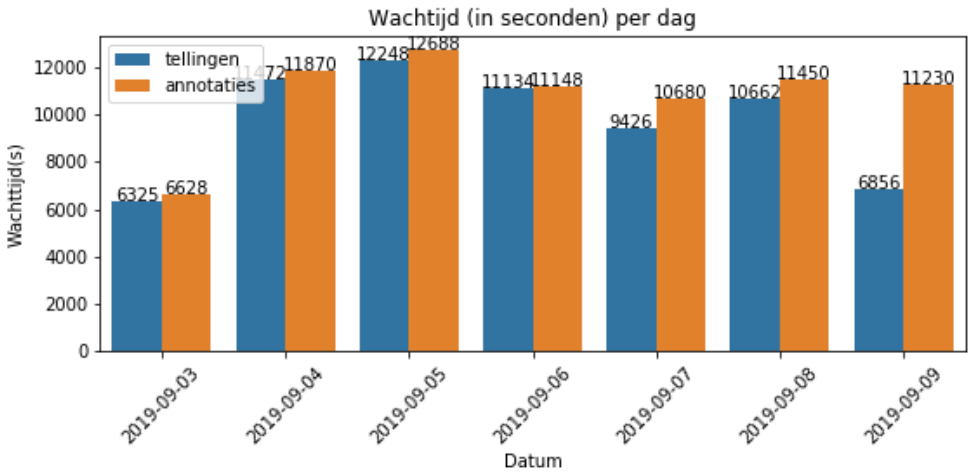
Figuur 10: Voorbeeld van fout in tracking algoritme

4.2 Resultaten

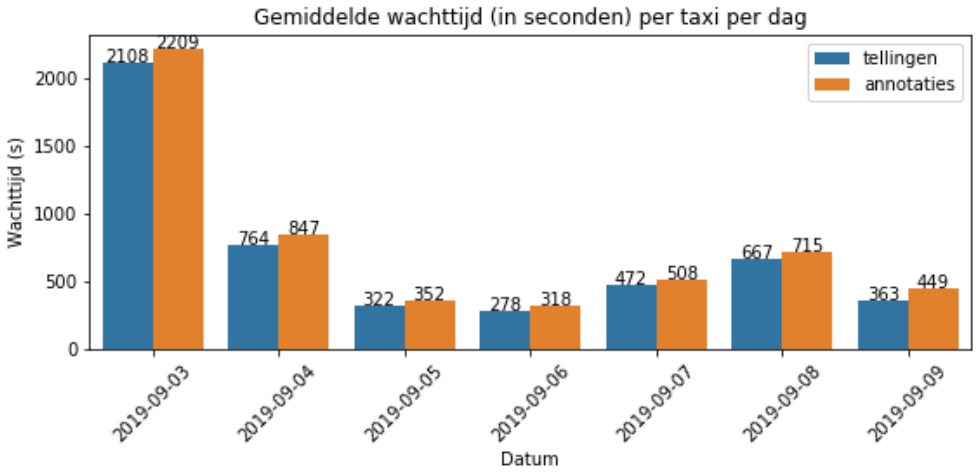
In totaal is van zeven dagen data verzameld. Om de uitkomsten van het algoritme te kunnen evalueren is in totaal 8 uur aan data handmatig geannoteerd. Op één dag is in de nacht twee uur data geannoteerd, op de rest van de dagen is een uur aan data geannoteerd. De geannoteerde data bestaat uit tijdstippen in zowel de ochtend, middag, avond en nacht.

In figuur 9 is voor ieder geannoteerd deel van de testset een overzicht te zien van de hoeveelheid getelde taxi's door het algoritme (tellingen) en de handmatig geannoteerde controle data (annotaties). In de grafiek is te zien dat op alle dagen, behalve de laatste, het algoritme een goede benadering heeft van het aantal taxi's dat op de taxistandplaats heeft gewacht. Er is een afwijking van één tot twee taxi's met een negatieve uitschieter op de laatste dag.

Op 5, 6 en 7 september is het verschil te verklaren doordat een aantal taxi's dubbel zijn geteld of niet goed zijn geteld door een fout in het tracking algoritme. In figuur 10 is een voorbeeld te zien waarop dit fout gaat. Op de afbeelding is een taxi te zien (roze bounding box) die op baan naast de taxistandplaatsen rijdt vlak langs een wachtende taxi (oranje bounding box). Doordat een aantal personen een deel van de wachtende taxi blokkeert is de object detectie niet optimaal. In dit geval wordt de wachtende taxi niet herkent, terwijl de voorbijrijdende taxi wel wordt herkend. Het algoritme koppelt hierdoor de verkeerde bounding boxes. Net als bij de vorige casus is ervoor gekozen om niet alle frames te analyseren, maar ieder vijfde frame, om snelheid te winnen. Dat betekent dat een auto meer afstand aflegt voordat er een nieuwe set aan gevonden objecten binnenkomt dan dat als op ieder frame Object Detection zou worden uitgevoerd. Doordat de afstand tussen objecten in twee opvolgende frames groter is, is er ook een grotere kans op fouten zoals hierboven beschreven.



Figuur 11: Overzicht van wachttijden



Figuur 12: Overzicht van gemiddelde wachttijden per dag

Op de laatste dag (9 september) wordt door het algoritme acht taxi's minder geteld dan er eigenlijk waren. De situatie wijkt af ten opzichte van de andere doordat het beeld zwart-wit is door het matige licht en het druk is. Het algoritme heeft op de meest rechter taxistandplaats erg veel moeite om de auto's nauwkeurig te detecteren. Doordat een deel van een gebouw dichtbij in het camerabeeld staat (zie figuur 8) heeft de camera geen goede focus op deze taxistandplaats in deze lichtomstandigheden. Het Object Detection algoritme heeft vooral moeite met het herkennen van taxi's op deze taxistandplaats als er rondom deze taxi's zich een groep personen bevindt.

In figuur 11 is een overzicht te zien van de totale wachttijd in seconden voor iedere geannoteerd deel van de dag in de testset. Net als bij de eerste casus komen de tellingen van het algoritme goed in de buurt van de daadwerkelijke tijden. Er zijn twee uitschieters op 7 en 9 september waar er volgens het algoritme veel minder is gewacht dan de daadwerkelijke situatie. Het grote verschil op 7 september komt doordat er driemaal uiterst rechts een object verloren gaat doordat er veel mensen rondom de auto staan. De auto wordt enige tijd niet meer gedetecteerd en verdwijnt dan. Het grote verschil op 9 september komt door het hierboven al verklaarde probleem op de meest rechter taxistandplaats, waarbij de taxi's slecht worden gedetecteerd.

In figuur 12 is de gemiddelde wachttijd voor een taxi per dag in de testset te zien. Hier is duidelijk af te lezen wat de drukke en rustige dagen zijn. Des te lager de waardes hier zijn, des te sneller er voor een taxichauffeur een klant is. In de testset op 6 september hoefde een taxi dus gemiddeld niet langer dan vijf minuten te wachten voor een klant.

Er zijn geen betrouwbare resultaten behaald om een iets te kunnen zeggen over het aantal personen dat in een taxi stapt. De taxi's volgen over het algemeen een logisch en te verwachten patroon. Echter, de personen die zich rondom de taxistandplaats begeven doen dit niet. Er is ook geprobeerd te herkennen hoeveel taxi's wegrijden zonder personen mee te nemen. Echter komt op deze plek het zeer zelden voor dat een taxi zonder klanten wegrijdt. Om iets over de nauwkeurigheid van het algoritme hierover te zeggen zijn meer voorbeelden hiervan nodig.

4.3 Uitdagingen

De grootste uitdaging is de koppeling van een persoon aan een taxi, dus wanneer een persoon daadwerkelijk instapt. De kwaliteit van de detectie van personen is niet altijd even goed door de hoge mate van occlusie, waarbij een object wordt geblokkeerd door een andere object zoals een andere persoon of een auto. Er zijn regelmatig beelden waarin een persoon niet meer kan worden gedetecteerd, waardoor ook de tracking moeilijker wordt. Detectie op meer frames zou kunnen helpen, maar dat betekent opoffering van de snelheid waarop het algoritme zijn taak kan uitvoeren.

Daarnaast bewegen personen zich in onvoorspelbare patronen, waardoor occlusie een nog grote probleem vormt. Ook is er veel ruis door personen die alleen door het beeld lopen en niet op zoek zijn naar een taxi. Omdat een enkel beeld de hele taxistandplaats omvat ontstaan er ook problemen aan de zijkanten van het beeld. Voor de meest rechter taxistandplaats is het complexer om te bepalen of iemand in de taxi is gestapt, omdat personen daar het beeld uit kunnen lopen en dus niet in de taxi stappen.

Het inprogrammeren van bepaalde regels zorgt ervoor dat op andere plekken mis kan gaan. Zo wordt er bijvoorbeeld een historie bijgehouden van personen die op enig moment door de tracker niet meer gevonden konden. Dit kan namelijk betekenen dat deze in een taxi is ingestapt. Op het moment dat een taxi wegrijdt wordt gekeken hoeveel personen in de historie in de buurt waren van de taxi vlak voordat deze wegreed. Het is echter moeilijk om te bepalen hoelang er terug moeten worden gekeken in de historie door de onvoorspelbaarheid van de situatie. Soms rijdt een taxi direct weg en soms duurt het enkele minuten voordat een taxi daadwerkelijk vertrekt.

De in sectie 4.2 genoemde fouten met betrekking tot het tracken van taxi's zouden op twee manieren kunnen worden verbeterd. Ten eerste zou de Object Detection stap vaker kunnen worden uitgevoerd, zodat de kans op verkeerde koppeling van objecten kleiner wordt. Ten tweede zou een geavanceerdere tracking algoritme kunnen worden gebruikt die beter rekening kan houden met de snelheid van een object. Deze tweede optie zou minder impact op de snelheid van het algoritme kunnen hebben dan de eerste optie, afhankelijk van de gekozen tracking techniek, en zou bij een vervolgproject de volgende verbeterstap kunnen zijn.

4.4 Snelheid

Op een laptop (Intel Core i9 @ 2.90Ghz, 16GB RAM en GeForce GTX 1050 Ti Max-Q) is het mogelijk om beelden in real-time te analyseren. Een 15 minuten durende video duurt gemiddeld 12 minuten en 22 seconden om te analyseren. Het is op dit moment nog niet haalbaar om bij een camera op locatie dit soort software te draaien op goedkopere schaalbare hardware.

5 Discussie en Conclusie

In het project is binnen twee verschillende casussen onderzocht hoe Computer Vision algoritmes toegepast kunnen worden op complexe stadsomgeving om daaruit informatie te halen over relevante beleidsvragen. Daarbij zijn bij iedere casus verschillende technieken gecombineerd, om zo meer te kunnen zeggen over wat er zich in het beeld afspeelt. Van het detecteren van objecten zoals auto's en personen, tot het over tijd volgen van de gedetecteerde objecten en de bepaling of een auto een taxi is of niet.

Uit het onderzoek blijkt dat subproblemen zeer bepalend zijn voor de resultaten. Dit zijn subproblemen als het detecteren van auto's, en het classificeren en tracken ervan. In overzichtelijke situaties worden er goede resultaten behaald. Bij specifiekere en complexer vraagstukken, zoals het aantal personen dat instapt in een taxi op een zeer druk moment, wordt het moeilijk om met state-of-the-art algoritmes nauwkeurige resultaten te behalen.

Eén van de uitdagingen binnen het project was om het gehele algoritme zo te maken dat deze op een schaalbare manier, Privacy by Design uitgerold zou kunnen worden binnen een stad. Op basis van de resultaten op een laptop, ingericht om snel met dit soort algoritmes te kunnen werken is het nog niet haalbaar om bij dit soort type algoritmes op goedkopere schaalbare hardware te installeren. Als het doel alleen is om tellingen uit te voeren op een overzichtelijke plek worden daar wel mogelijkheden voor gezien.

De Object Detection stap is het bouwblok waarop de rest van de technieken in het project voortbouwen. In dit project is ervoor gekozen om niet ieder frame te analyseren zodat het gehele algoritme sneller werkt. Om nauwkeurigere Object Detection resultaten te krijgen zijn er verschillende mogelijkheden. Zo kan een Object Detection algoritme met nieuwe voorbeelden getraind worden vanuit de verzamelde data. Op deze manier kan de algemene nauwkeurigheid voor die specifieke casus worden verhoogd. Daar tegenover staat dat hierdoor de overdraagbaarheid naar andere situaties weer moeilijker wordt. Om diezelfde nauwkeurigheid te kunnen halen kan het zijn dat voor iedere casus het algoritme getraind moeten worden op nieuwe voorbeelden. Een tijdrovende klus. Een andere optie is om afhankelijk van de casus en de impact op privacy een mogelijkheid om de analyse in de cloud te doen. Hier is meer rekenkracht beschikbaar, wat het makkelijker maakt om wel elke frame te analyseren.

Het nauwkeurig kunnen volgen van object in een complexe stadssituaties, zoals geschetst in de taxistandplaats casus, geeft veel uitdagingen. Omdat het goed kunnen volgen van object een voorwaarde is voor nauwkeurige tellingen is hier nog een grote verbeterslag te maken. Voor state-of-the-art Object Tracking algoritmes zijn de grootste uitdagingen het omgaan met onvoorspelbare patronen en uit beeld verdwijnende objecten. Met complexere algoritmes zou wellicht beter resultaten kunnen worden behaald, maar hier geldt weer dat de snelheid waarop de analyse kan worden uitgevoerd snel omlaag gaat.

Met veel sleutelen aan bepaalde situaties en het inprogrammeren van context afhankelijke regels voor iedere casus is het mogelijk om goede resultaten te halen. Er moet een balans worden gevonden tussen de snelheid waarmee het algoritme werkt en de overdraagbaarheid naar andere locaties. Om dit soort technieken Privacy by Design toe te kunnen passen in de openbare ruimte is er snellere, betaalbare hardware voor plaatsing op locatie nodig en moet, gezien er geen data meer wordt opgeslagen om op te valideren, de nauwkeurigheid verder omhoog.

De casussen hebben veel inzicht gegeven in de huidige mogelijkheden van Computer Vision algoritmes zodra deze worden toegepast in complexe stadsomgevingen. Er doen zich oneindig veel onvoorziene situaties voor in de publieke ruimte die dit soort casussen zeer interessant maken voor dit type onderzoek. Het biedt naast de waardevolle inzichten ook nieuwe kansen voor relevante vervolgonderzoeken.

6 Acknowledgements

Dit project is gefinancierd binnen het NWO - Verbinden van Duurzame Steden (VerDus) programma. Met dank aan de Gemeente Amsterdam voor het meedenken en hulp bij de dataverzameling.

References

[1] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[2] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[3] Igor S Farias, Bruno JT Fernandes, Edison Q Albuquerque, and Byron LD Leite. Tracking and counting of vehicles for flow analysis from urban traffic videos. In *2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, pages 1–6. IEEE, 2017.

[4] Panagiotis Giannakeris, Vagia Kaltsa, Konstantinos Avgerinakis, Alexia Briassouli, Stefanos Vrochidis, and Ioannis Kompatsiaris. Speed estimation and abnormality detection from surveillance cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 93–99, 2018.

[5] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2147–2154, 2014.

[6] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[8] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[9] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]*, March 2016. arXiv: 1603.00831.

[10] Yi Wei, Nenghui Song, Lipeng Ke, Ming-Ching Chang, and Siwei Lyu. Street object detection/tracking for ai city traffic analysis. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 1–5. IEEE, 2017.

[11] Mehmet Kemal Kocamaz, Jian Gong, and Bernardo R Pires. Vision-based counting of pedestrians and cyclists. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8. IEEE, 2016.

[12] Mustansar Fiaz, Arif Mahmood, Sajid Javed, and Soon Ki Jung. Handcrafted and deep trackers: A review of recent object tracking approaches. *arXiv preprint arXiv:1812.07368*, 2018.

[13] Laura Leal-Taixé, Anton Milan, Konrad Schindler, Daniel Cremers, Ian Reid, and Stefan Roth. Tracking the trackers: an analysis of the state of the art in multiple object tracking. *arXiv preprint arXiv:1704.02781*, 2017.

[14] Yingkun Xu, Xiaolong Zhou, Shengyong Chen, and Fenfen Li. Deep learning for multiple object tracking: a survey. *IET Computer Vision*, 13(4):355–368, 2019.

[15] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[16] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[17] Shan Du, Mahmoud Ibrahim, Mohamed Shehata, and Wael Badawy. Automatic license plate recognition (alpr): A state-of-the-art review. *IEEE Transactions on circuits and systems for video technology*, 23(2):311–325, 2012.

[18] A. Homburg, G.and Schreijenberg, Tillaart J., and Y. van den, Bleeker. 2387-intelligente toepassingen van automatic number plate recognition. 2019.

[19] Tom Otten. Counting taxis in amsterdam using computer vision algorithms. B.s. thesis, University of Amsterdam, 8 2019.

[20] Bo Li, Tianlei Zhang, and Tian Xia. Vehicle detection from 3d lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*, 2016.

[21] Brilian Tafjira Nugraha, Shun-Feng Su, et al. Towards self-driving car using convolutional neural network and road lane detector. In *2017 2nd International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT)*, pages 65–69. IEEE, 2017.

[22] Chen Wang, Aibek Musaev, Pezhman Sheinidashtegol, and Travis Atkison. Towards detection of abnormal vehicle behavior using traffic cameras. In *International Conference on Big Data*, pages 125–136. Springer, 2019.

[23] Sokemi Rene Emmanuel Datondji, Yohan Dupuis, Peggy Subirats, and Pascal Vasseur. A survey of vision-based traffic monitoring of road intersections. *IEEE transactions on intelligent transportation systems*, 17(10):2681–2698, 2016.

[24] Robert Kerwin C Billones, Argel A Bandala, Laurence A Gan Lim, Edwin Sybingco, Alexis M Fillone, and Elmer P Dadios. Microscopic road traffic scene analysis using computer vision and traffic flow modelling. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 22(5):704–710, 2018.

[25] Ann Cavoukian et al. Privacy by design: The 7 foundational principles. *Information and Privacy Commissioner of Ontario, Canada*, 5, 2009.

[26] Sergio Montazzolli Silva and Claudio Rosito Jung. License plate detection and recognition in unconstrained scenarios. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 580–596, 2018.