

Gebroken Glas, Falende Camera's: Simuleren van Fysisch-gebaseerde Adversariële Steekproeven voor Autonome Rijsystemen

Manav Prabhakar, Jwalandhar Girnar, Arpan Kusari* University of Michigan Transportation Research Institute 2901 Baxter Road, Kamer 202, Ann Arbor, MI-48103 {prmanav, jwala, kusari}@umich.edu

Samenvatting

Hoewel recent veel onderzoek zich heeft gericht op het genereren van fysiek-gebaseerde adversariële steekproeven, is er een kritieke, maar vaak over het hoofd geziene categorie die voorkomt uit fysieke storingen binnen boordcamera's—onderdelen die essentieel zijn voor de perceptiesystemen van autonome voertuigen. Camerastoringen, of ze nu worden veroorzaakt door externe spanningen die hardwarestoringen veroorzaken of interne componentfouten, kunnen direct de veiligheid en betrouwbaarheid van autonome rijsystemen in gevaar brengen. Ten eerste motiveren we de studie met behulp van twee afzonderlijke experimenten in de echte wereld om aan te tonen dat glasstoringen inderdaad zouden leiden tot het falen van op detectie gebaseerde neurale netwerkmodellen. Ten tweede ontwikkelen we een simulatie-gebaseerde studie met behulp van het fysische proces van glasbreuk om verstoerde scenario's te creëren, die een realistische klasse van fysiek-gebaseerde adversariële steekproeven vertegenwoordigen. Met behulp van een eindige-elementenmodel (Eindige Elementen Methode)-gebaseerde benadering genereren we oppervlaktebarsten op de camera-afbeelding door een spanningsveld toe te passen dat wordt gedefinieerd door deeltjes binnen een driehoekig raster. Ten slotte gebruiken we fysiek-gebaseerde rendering (PBR) technieken om realistische visualisaties te bieden van deze fysiek plausibele breuken. Om de veiligheidsimplicaties te beoordelen, passen we de gesimuleerde gebroken glaseffecten toe als afbeeldingsfilters op twee datasets voor autonoom rijden - KITTI en BDD100K - evenals de grootschalige afbeeldingsdetectiedataset MS-COCO. We evalueren vervolgens de detectiefalingspercentages voor kritieke objectklassen met behulp van op CNN gebaseerde objectdetectiemodellen (YOLOv8 en Faster R-CNN) en een transformer-gebaseerde architectuur met Pyramid Vision Transformers. Om de distributie-impact van deze visuele vervormingen verder te onderzoeken, berekenen we de Kullback-Leibler (K-L) divergentie tussen drie verschillende datadistributies, waarbij we verschillende gebroken glasfilters toepassen op een aangepaste dataset (vastgelegd door een gebarsten voorruit), evenals de KITTI en Kaggle Cats and Dogs datasets. De K-L divergentieanalyse suggereert dat deze gebroken glasfilters geen significante distributieververschuingen introduceren. Ons doel is om een robuuste, fysiek-gebaseerde methodologie te bieden voor het genereren van adversariële steekproeven die reële camerastoringen weerspiegelen, met als overkoepelend doel de veerkracht en veiligheid van autonome rijsystemen tegen dergelijke fysieke bedreigingen te verbeteren.

Code —

<https://github.com/manavprabhakar/camera-failure>

Inleiding

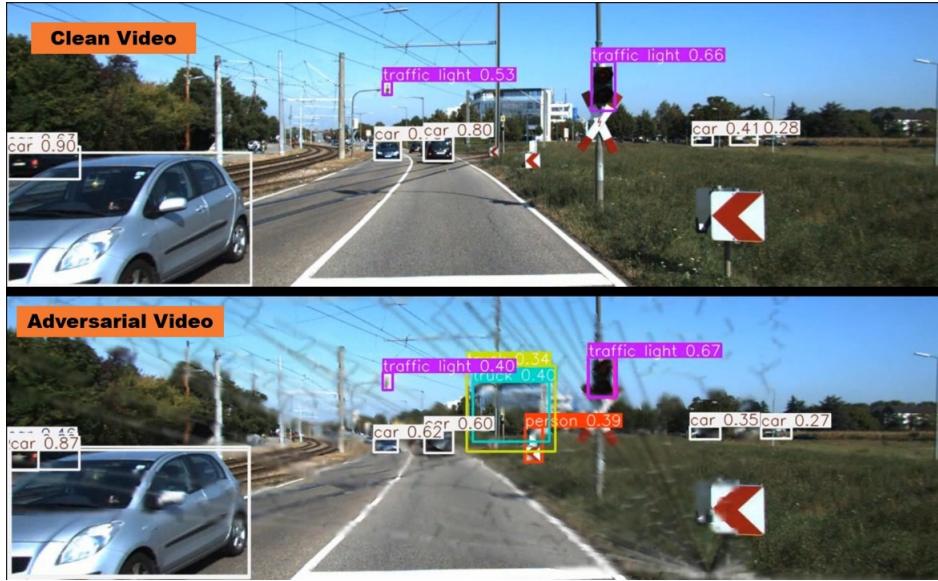
Camera's zijn alomtegenwoordig als afstandssensoren, die gegevens verzamelen uit een ongestructureerde en dynamische externe omgeving, vaak onder zware omstandigheden. Een storing of fout in een sensor is een afwijking van de functionele staat in ten minste één gegeven parameter van het systeem (van Schrick 1997). Deze fouten kunnen optreden door interne (zoals slijtage) of externe (temperatuur, vochtigheid, enz.) oorzaken. Voor RGB-camera's omvatten interne oorzaken dode pixels, terwijl externe oorzaken gebroken behuizingen of buitenlenzen en condensatie omvatten. Deze abrupte storingen zijn moeilijk te detecteren en hebben een negatieve invloed op objectdetectie-algoritmen, waardoor de nauwkeurigheid afneemt en vaak tot hallucinaties leidt, zoals getoond in Fig. 1. De storingen die bijvoorbeeld in een geautomatiseerd voertuig (AV) optreden, kunnen leiden tot kritieke veiligheidsproblemen die resulteren in ongelukken en in sommige gevallen, dodelijke slachtoffers.

Op dit moment, voor zover de auteurs weten, zijn er geen rigoureuze methoden voor het genereren van op camera gebaseerde sensorstoringen (Ceccarelli en Secci 2022).

In dit werk richten we ons op het falen van sensoren dat optreedt door breuken in elk glas dat een camera (of camera-omhulsel) bedekt, hoewel het proces dat in dit artikel wordt beschreven, kan worden gebruikt voor elk van de camerafouten die worden vermeld in (Ceccarelli en Secci 2022). Deze glasbreukeffecten in een camera kunnen worden veroorzaakt door een extern object dat de camera raakt of als gevolg van plotselinge ontwikkeling van hitte en/of druk binnen het omhulsel. In de terminologie van neurale netwerken wordt een afbeelding die onder dergelijke omstandigheden is vastgelegd, beschouwd als een adversarial sample. Eerder onderzoek (Akhtar en Mian 2018; Carlini en Wagner 2017; Szegedy et al. 2013) toont aan dat zelfs kleine hoeveelheden verstoringen, soms moeilijk te zien met het menselijk oog, voldoende zijn om de neurale netwerken volledig te misleiden, waarbij een subtiele verandering van invoer kan leiden tot een drastische verandering in uitvoer. We willen opmerken dat (Li, Schmidt en Kolter 2019) een fysiek cameragebaseerd adversarial attack-paradigma hebben gepresenteerd, dat dient als het meest verwante werk in dit domein. Zij presenteerden een aanpassing van de afbeelding met behulp van een overlay van een doorschijnende, zorgvuldig vervaardigde sticker die leidde tot verkeerde classificatie.

Om het effect van deze breuken op de resulterende camerabeelden te begrijpen, hebben we twee verschillende experimenten uitgevoerd: één

*Correspondentieauteur Copyright © 2026, Vereniging voor de Bevordering van Kunstmatige Intelligentie(www.aaai.org). Alle rechten voorbehouden.



Figuur 1: Een kwalitatieve vergelijking van schone versus adversariële video gegenereerd met behulp van onze simulatie- en renderingsmethode op KITTI. Dit frame toont valse positieven en verminderde vertrouwensniveaus voor ware positieven. Raadpleeg het aanvullend materiaal voor de volledige video.

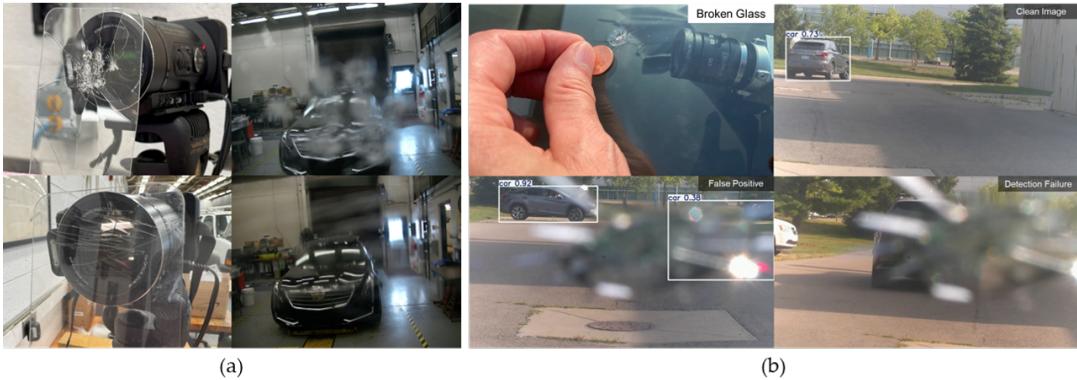
in een statische binnenomgeving en de andere in een dynamische buitenomgeving. Het eerste experiment betrof het breken van gehard glas en het voor de camera plaatsen (zie Fig. 2(a)) met een statisch voertuig in de scène om te begrijpen hoe verschillende breukpatronen de kwaliteit en het uiterlijk van de scène beïnvloeden. We maakten opnamen bij verschillende brandpuntsafstanden om de variabiliteit van dergelijke verstoringen te beoordelen. Dit hielp ons bepaalde kwalitatieve vragen te beantwoorden over het visuele uiterlijk van deze breuken met betrekking tot hun verspreiding en intensiteit, wat onze aanpak in de sectie Brandpuntsvlak en Simulatie van fysieke aanval motiveerde. De experimentele opstelling en de gedetailleerde experimentele resultaten zijn te vinden in Sectie Statisch Experiment van Aanvullend. Het tweede experiment (Fig. 2(b)) bestond uit het opnemen van een buitenvideo met dynamische voertuigen onder daglichtomstandigheden door een MobileEye-camera naast een voorruitbreuk te plaatsen zoals gepresenteerd in Fig. 2 (getoond in de linkerbovenhoek) en het uitvoeren van inferentie met behulp van YOLOv8 (Jocher, Chaurasia, en Qiu 2023) om een primitief begrip te krijgen van de impact van dergelijke scenario's op objectdetectienetwerken. We observeerden dat het model het voertuig gemakkelijk kan detecteren in een schoon beeld, terwijl het last heeft van detectiefouten (rechtsonder) of valse positieven genereert (linksonder). Interessant genoeg kan de aanwezigheid van een breuk ook onverwacht het vertrouwen in de voorspelling van de auto verhogen, waarbij een duidelijk gedefinieerde rand wordt gepresenteerd (0,92 linksonder vs. 0,75 linksboven). De gedetailleerde inferentieresultaten met voertuig- en persoonsklasse zijn te vinden in Sectie Dynamisch Experiment van Aanvullend.

We zochten vervolgens online naar echte afbeeldingen van gebroken glas (Sectie Echte glasbreukafbeeldingen van Aanvullend), maar slaagden er niet in een dataset groot genoeg te bouwen om een data-gedreven benadering voor adversariële verdediging onder deze omstandigheden mogelijk te maken. Daarnaast experimenteerden we met CGI-tools zoals Maya en Blender voor

het creëren van dergelijke effecten, maar ze missen de flexibiliteit, controle, schaal en fysica om deze omstandigheden te simuleren. De dichtstbijzijnde simulatieoptie in de bestaande literatuur is ArcSim (Pfaff et al. 2014). Hun simulatie-uitvoer met hoge resolutie is echter extreem traag (≈ 20 uur), wat het moeilijk maakt om op te schalen. Als gevolg daarvan richten we onze inspanningen op het creëren van een schaallbare simulatie-gebaseerde pijplijn voor het genereren van breuken die kunnen worden gebruikt om de perceptiestapel te verbeteren.

Voor een glasbreuk kunnen het hoofdpunt, de kracht en de invalshoek willekeurig zijn, maar de verspreiding en het resulterende patroon volgen een inherent fysiek proces (zijdne lineair of radiaal). We bouwen daarom een breuksimulatie gebaseerd op deeltjes in een willekeurig gegenereerd driehoekig netwerk en voeren spanningsoortplanting door het netwerk uit. Onze simulatie stelt ons in staat om de breuken binnen een driehoekig netwerk te produceren in elke discrete tijdstoestand δt . We gebruiken OpenCV om het gegeven netwerk om te zetten in een overeenkomstig gebroken glaspatroonbeeld. Vervolgens maken we gebruik van fysiek-gebaseerde rendering (PBR) (Pharr, Jakob, en Humphreys 2023) om de oppervlaktebreuken realistisch te renderen met behulp van de bidirectionele reflectieverdelingsfunctie (BRDF) door de hoeveelheid licht te berekenen die wordt gereflecteerd vanaf een bepaald punt op een oppervlak als gevolg van lichtbron(nen) die erop invallen.

Door onze renderingsaanpak te combineren met drie populaire open source datasets - KITTI (Geiger et al. 2013), BDD100K (Yu et al. 2020) en MS-COCO (Lin et al. 2014), zijn we in staat om efficiënt adversariële afbeeldingen te genereren. Een gebruikelijke methode om de gegenereerde adversariële afbeeldingen te testen, is het aantal false positives/negatives in de beeldruimte te vinden. Echter, in ons geval, vanwege het lokale adversariële effect, kunnen we niet eenvoudigweg op een beeldgebaseerde maatstaf vertrouwen. Daarom gebruiken we de adversariële afbeeldingen (vergelijkbaar met de figuur linksunder in Fig. 2) en extraheer we de objecten



Figuur 2: (a) Indoor statisch experiment. Links: Camera met 2 verschillende patronen van gebroken gehard glas; rechts - beelden van het voertuig onder de verschillende breuken. (b) Outdoor dynamisch experiment. Linksboven - een muntgrote voorruitsschade; rechtsboven - schoon beeld met het voertuig gedetecteerd met YOLOv8; linksonder - vals positief door de barst; rechtsonder - detectiefout door het glas. Meer voorbeelden van deze experimenten zijn opgenomen in het aanvullend materiaal.

die zich bevinden binnen de regio waar de breuk bestaat met behulp van de ground truth begrenzingskaders. We maken vervolgens gebruik van YOLOv8, Faster R-CNN (Ren et al. 2016) en Pyramid Vision Transformer (PVTv2) (Wang et al. 2022) om het percentage objecten te vinden dat faalt wanneer de adversariële filters worden toegepast. We bieden ook ablatiestudies aan om de distributieverschillen tussen de drie sets afbeeldingen te begrijpen: Echt gebroken glasafbeeldingen die experimenteel zijn verzameld, echt gebroken glasafbeeldingen die online zijn verzameld en de gegenererde afbeeldingen. We berekenen de Kullbeck-Liebler (K-L) divergentie voor deze afbeeldingsdistributies om de gelijkenis van de gegenererde afbeeldingen met de echte gebroken glasafbeeldingen te bewijzen. We gebruiken kattenafbeeldingen uit de Kaggle Cats and Dogs dataset als controle om het verschil tussen afbeeldingsdistributies (PK) te begrijpen.

De belangrijkste bijdragen van het artikel kunnen als volgt worden samengevat:

- We bieden een nieuwe manier om glasbreuk te abstracteren door een combinatie van spanningspropagatiemethoden en minimale opspannende bomen, om fysiek kloppende gebroken glaspanelen te genereren.
- We presenteren een PBR-benadering om een realistische weergave van camerafouten te faciliteren die kan worden gebruikt met elk soort bestaand computervisie-dataset - zowel afbeeldingen als video's.
- Onze simulatie- en renderpijplijnen zijn schaalbaar en computationeel efficiënt ($\approx 1.6s$), waardoor ze zowel door de academische wereld als de industrie kunnen worden gebruikt voor het verbeteren van robuustheid en bescherming tegen out-of-distribution voor een breed scala aan toepassingen.

Achtergrond

Fysisch gebaseerde adversariële steekproeven

Het probleem van een adversariële steekproef kan als volgt worden gedefinieerd: voor een model M dat een invoersteekproef X correct classificeert naar zijn aangewezen klasse, d.w.z. $M(X) = y_{true}$, leidt het toevoegen van een ϵ aan de invoersteekproef X tot een gewijzigde steekproef X' zodanig dat $M(X') \neq y_{true}$. Dus, de injectie van de fout ϵ resulteert in een adversariële steekproef die ervoor zorgt dat het model faalt.

Hoewel het idee van adversariële manipulatie van het model al geruime tijd geleden is geïdentificeerd in de context van machine learning (Dalvi et al. 2004), is in het afgelopen decennium de focus volledig gericht geweest op de adversariële aanvallen op neurale netwerken (Szegedy et al. 2013; Goodfellow, Shlens, en Szegedy 2014). In deze artikelen toonden de onderzoekers aan dat een kleine gerichte injectie van ruis, bijna onmerkbaar voor het menselijk oog, de labels volledig veranderde (Szegedy et al. 2013) en omgekeerd, dat er beelden konden worden gegenereerd die voor mensen volledig onherkenbaar leken, maar die perfecte classificaties hadden van de DNN's (Nguyen, Yosinski, en Clune 2015).

Hoewel deze adversariële steekproeven het model testen op mogelijke fouten, ontbreekt het hen aan fysieke realiteit bij hun generatie en hebben ze toegang tot het model nodig. Om dit aan te pakken, heeft recent onderzoek zich gericht op het ontwikkelen van fysiek relevante adversariële steekproeven. Een van de eerste pogingen hiertoe werd gedaan door (Kurakin, Goodfellow en Bengio 2018), die zich richtten op de nauwkeurigheid van de modellen in de fysieke wereld door ruisachtige beelden van een mobiele telefooncamera te gebruiken, waardoor het model een groot deel van de steekproeven verkeerd classificeerde. In dezelfde lijn toonden (Eykholt et al. 2018) aan dat echte verkeersborden kunnen worden verstoord met eenvoudige fysieke stickers die strategisch zijn geplaatst om geavanceerde DL-algoritmen bijna perfect te misleiden, zelfs met veranderingen in het gezichtspunt. Andere onderzoekers hebben adversariële beelden (Kong et al. 2020), Translucent Patches op camera (Zolfi et al. 2021) of kunstmatige LiDAR-oppervlakken (Tu et al. 2020) geplaatst om steekproeven te genereren die objectdetectoren misleiden. Hoewel dit eerdere onderzoek fysica gebruikt bij het genereren van de steekproeven, komen ze niet voor uit het modelleren van een rigoureus fysiek proces en we streven ernaar deze kloof in dit werk te vullen.

Theorie van gebarsten/gebroken glas

Het onderwerp van hoe glas breekt en hoe het zich verspreidt, is nog steeds een open onderzoeksraag en een die controversieel is met meerdere fysieke theorieën die zijn voorgesteld (Rouxel en Brow 2012). Terwijl de microscopische procedure van glasbreuk wordt bediscussieerd, is op macroniveau het barsten

dynamica goed begrepen. (Liu et al. 2021) analyseerden het proces van het barsten van een glazen lens in de preciesieglasvormtoepassing met behulp van de Eindige Elementen Methode met een driedimensionaal model in een fysiek simulatiesoftware. De fysieke parameters werden ingevoerd in de software en de barstpaden werden geanalyseerd met behulp van de simulatie-resultaten. De auteurs voerden een temperatuur- en spanningssimulatie uit van een hoogprecisie driedimensionaal meshmodel van het gevormde glas. (Iben en O'Brien 2009) boden een manier om oppervlaktebreuken te genereren in verschillende materialen, waaronder glas. Zoals al in de inleiding vermeld, leverde (Pfaff et al. 2014) de simulatie van glasbreuk als een dunne plaat, wat het meest verwante werk vormt met onze voorgestelde methode.

Methodologie

Het genereren van realistische glasschade vereist het creëren van grootschalige natuurkundige simulaties door breukdynamica op te lossen op een getriëerd eindig-elementenrooster met glas eigenschappen.

Simulatie van gebroken glas

We vertegenwoordigen glas met behulp van deeltjes die zijn bemonsterd uit een uniforme verdeling verspreid over een vlak, beperkt in de vorm van een 2D-rooster met behulp van beperkte Delaunay-triangulatie. Dit verwijdert slecht gevormde driehoeken en voorkomt ongelijke en onrealistische randen.

Elk deeltje p_i heeft een positie x_i en heeft nabije buren k_i binnen een straal r die bestaande randen hebben met p_i . Wiskundig gezien vertegenwoordigt het triangulatierooster \mathcal{M} een eindige verzameling van 2-simplexen zodanig dat als

$$\forall (K, K') \in \mathcal{M} \times \mathcal{M}, |K| \cap |K'| = |K \cap K'|. \quad (1)$$

De scheurpatronen in glas ontstaan door spanning van de externe kracht (F) op het initiële impactpunt p_I door een specifieke vervormingswet (elasticiteit en plasticiteit) van het glas (G) aan te nemen (Kuna 2013). We berekenen vervolgens de sterkteparameters in de vorm van effectieve spanning σ_V op het impactpunt (V) als de spanningsstaat van het impactpunt. De kritische spanningswaarden voor de sterkte van glas σ_C worden gevonden met behulp van tests op eenvoudige monsters met elementaire belastingcondities (bijv. trekproef). De breuk treedt dan op wanneer de effectieve spanning groter is dan de kritische spanning gedeeld door de veiligheidsfactor (S):

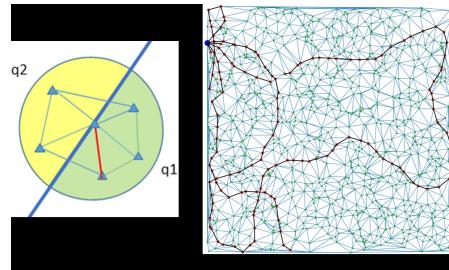
$$\sigma_V(G, F) > \frac{\sigma_C}{S}. \quad (2)$$

Uit de klassieke theorie van sterkteleer weten we dat het falen in de meeste gevallen wordt beheerst door de hoofdspanningen σ_I en σ_{II} voor 2D-elementen. De initiële scheur ontstaat ofwel door de normaal-vlakte scheur waarbij de breukvlakken zich loodrecht op de richting van de hoogste hoofdspanning σ_I bevinden (Rankine 1857), of door de schuif-vlakte scheur waarbij de breukvlakken samenvallen met de snijvlakken van de maximale schuifspanning $\tau_{max} = (\sigma_I - \sigma_{II}) / 2$ (Coulomb 1776). In het geval van glas gaan we ervan uit dat de initiële breuk loodrecht op de richting van de maximale hoofdspanning plaatsvindt.

Vanaf het initiële impactpunt p_I is de spanningsvoortplanting door glas onstabiel, aangezien de scheur abrupt groeit zonder dat er een toename van externe belasting nodig is. Vanaf p_I ,

propageert de spanning in de buurt van de hoek k_i als de spanning in de richting $\vec{p_I p_j}$ waar $p_j \in k_i$ als

$$\sigma_{p_j} = \sigma_V * \frac{\vec{p_I p_j} \cdot \vec{n}}{|\vec{p_I p_j}| |\vec{n}|}. \quad (3)$$



Figuur 3: (a) Voor een splijtvlek aangegeven in blauw, worden de opgetelde positieve spanning q_1 en opgetelde negatieve spanning q_2 vergeleken en vindt de voortplanting plaats aan de zijde met de grotere opgetelde spanning en de gekozen knoop die het dichtst bij het splijtvlek ligt (aangegeven in rood). (b) Toont hoe we een breuk simuleren in een mesh die ontstaat vanuit het impactpunt (gemarkeerd in blauw) naar de knopen die spanning ervaren boven hun drempelsterkte (gemarkeerd in rood).

Met de berekende spanning voor elke rand kan de som van de positieve spanning (getoond in Fig. 3) dan worden gegeven als:

$$q_1 = \int_{\partial\Omega} \sigma_{p_j} \mathbb{I}(\sigma_{p_j} > 0) dA \quad (4)$$

voor een continue oppervlakte Ω waar \mathbb{I} de indicatorfunctie is. De som van de positieve spanning voor discrete simplices in het overeenkomstige gebied A met straal R wordt gegeven als

$$q_1 = \sum_{K \in A_R} \sigma_K \mathbb{I}(\sigma_K > 0). \quad (5)$$

Op dezelfde manier wordt de som van de negatieve spanning q_2 berekend. Dan, voor een grotere magnitude $\max(|q_1|, |q_2|)$, kiezen we de overeenkomstige rand met de hoogste concentratie van spanning in het gegeven segment als het optimale splitsingsvlak, omdat dat de maximale spanningsverlichting biedt. Zo reist de spanning langs de maarsranden en verspreidt de spanning zich bij elk knooppunt.

De recursieve toepassing van de spanningspropagatie wordt uitgevoerd totdat de spanning in alle staten convergeert, d.w.z. $\sigma_p^{(t)} \approx \sigma_p^{(t-1)} \forall p \in V$.

Het propageren van de spanning in alle richtingen over alle knooppunten resulteert in terugbarsten zoals uitgelegd in (O'Brien en Hodgins 1999). Om dit te voorkomen, propageren we alleen langs de randen waar de spanningsniveaus maximaal zijn, maar voeren we een spanningsupdate uit op alle naburige knooppunten. We gebruiken dan een minimale omspannende boom (MST) op een maas die is gemaakt met deze gespannen knooppunten. We combineren deze MST met ons initiële spanningspropagatieveld langs de randen om het uiteindelijke barstpatroon te berekenen. De MST is een effectieve abstractie omdat het de knooppunten verbindt die dichter bij elkaar liggen en binnen het hoge spanningsveld, terwijl het overbodigheden verwijdt.

Ons computationele proces van spanningsvoortplanting is gedefinieerd in Algoritme 1 in Aanvullend.

Fysiek-gebaseerde rendering

Zodra we de breuken op het mesh-niveau hebben gegenereerd, is ons volgende doel om een visuele weergave van deze breuken te creëren. Zoals alle PBR-technieken is onze methode gebaseerd op de microfacet theorie, die stelt dat elk oppervlak kan worden beschreven door piepkleine, perfect reflecterende spiegels genaamd microfacetten (Pharr, Jakob, en Humphreys 2023).

In overeenstemming met de microfacet theorie en energiebesparing, gebruiken we de reflectantievergelijking,

$$L_o(x, \omega_o, \lambda, t) = L_e(x, \omega_o, \lambda, t) + L_r(x, \omega_o, \lambda, t) \quad (6)$$

waarbij $L_o(x, \omega_o, \lambda, t)$ de totale spectrale stralingssterkte is van golflengte λ die naar buiten is gericht langs richting ω_o op tijdstip t , vanaf een bepaalde positie x . ω_o is de richting van het uitgaande licht. t is tijd. L_e is de uitgezonden spectrale stralingssterkte en L_r is de gereflecteerde spectrale stralingssterkte.

Laat I_1 de bidirectionele reflectantieverdelingsfunctie zijn,

$$I_1 = f_r(x, \omega_i, \omega_o, \lambda, t)$$

en laat I_2 de spectrale stralingssterkte zijn die naar binnen komt richting x vanuit richting ω_i op tijdstip t .

$$I_2 = L_i(x, \omega_i, \lambda, t)$$

Dan kan L_r worden gedefinieerd als

$$L_r(x, \omega_o, \lambda, t) = \int_{\Omega} I_1 \cdot I_2 \cdot (\omega_i \cdot \mathbf{n}) d\omega_i \quad (7)$$

waarbij Ω de eenheidshemisfeer is geцentreerd rond de oppervlakte-normaal \mathbf{n} over ω_i zodat $\omega_i \cdot \mathbf{n} > 0$.

Door de reflectantievergelijking te abstracteren, willen we een visuele weergave van ons gebroken glasnet maken. We hebben $L_e = 0$ aangezien glas geen licht uitstraalt. Nu, voor het berekenen van L_r , beschouwen we elke barst tussen de knooppunten als een microfacet. Dan kunnen we L_r voor elke barst definiëren als:

$$L_r = L_i(\omega_i \cdot \hat{\mathbf{n}}) \quad (8)$$

Gegeven de eenheidsvectoren $(\hat{\omega}_\alpha)$ en $(\hat{\omega}_\theta)$ die respectievelijk overeenkomen met de azimut (α) en zenith (θ) hoeken, berekenen we de gemiddelde energie die op de barst valt als

$$\mathbb{E}(L_r) = \frac{|\hat{\omega}_\alpha \cdot \hat{\mathbf{n}}| + |\hat{\omega}_\theta \cdot \hat{\mathbf{n}}|}{2} \quad (9)$$

waarbij $\hat{\mathbf{n}}$ de eenheidsoppervlakte-normaal van de barst is.

Laat (I_r, I_g, I_b) de gemiddelde intensiteit van de lichtbron zijn. Dan wordt de barstintensiteit, I_c gedefinieerd als

$$I_c = (I_r, I_g, I_b) \cdot \frac{\mathbb{E}(L_r)}{\sum L_r} \quad (10)$$

Brandpuntsvlak en Simulatie van fysieke aanval Hoewel we in staat zijn realistische breuken te simuleren, is het primaire gebruik van ons werk het genereren van gesimuleerde voorbeelden die over bestaande datasets (KITTI, BDD100k, MS-COCO) worden gelegd en deze te vergelijken met de echte dataset op de weg die we hebben gecreëerd.

Elk vastgelegd beeld zal scherpe kenmerken vertonen van de objecten in het brandpuntsvlak. De glazen behuizing die de camera bedekt, is extreem dichtbij en maakt daarom geen deel uit van het brandpuntsvlak.

Wanneer de barst ontstaat, weerkaatsen de lichtstralen ongelijkmatig langs de barst en ontstaat er een waas (voorbeld gegeven in Fig. 4). We maken een binaire masker op basis van het barstpatroon en vervagen vervolgens de breuken die op het beeld zijn gelegd. Dit produceert een veraf-focus beeld. Voor een kort-focus beeld vervagen we het beeld en richten we ons op de voorgrond, d.w.z. de barst.

Experimentatie

Dataset

We benchmarken twee soorten gebroken glaspatronen - echt en gesimuleerd - op drie populaire open-source datasets - KITTI (Geiger et al. 2013), BDD100K (Yu et al. 2020) en MS-COCO (Lin et al. 2014). De eerste twee vertegenwoordigen specifieke domeinen van autonoom rijden, terwijl de laatste een algemeen beeldendataset is. De echte gebroken glaspatroonafbeeldingen zijn verzameld van de FreePik-website¹ en vormen de basislijn in ons geval. We hebben in totaal 65 afbeeldingen verzameld en deze uitgebreid tot een set van 10.000 afbeeldingen via beeldvergroting met behulp van willekeurige verschuivingen, beeldomkeringen en bijnijdechnieken. We genereren ook 10.000 afbeeldingen met behulp van onze fysicasimulator. Vervolgens leggen we deze gebrosten glaspatronen met onze PBR-pijplijn over elke validatieafbeelding in de datasets en verzamelen de geaggregeerde resultaten. We gebruiken drie modelarchitecturen YOLOv8, Faster R-CNN en PVTv2 model met voorgetrainde gewichten om objectdetectieresultaten te genereren.

Implementatie

Ons simulatiemodel is ontwikkeld door willekeurig 10^4 deeltjes te bemonsteren uit een uniforme ruimtelijke verdeling in het gegeven frame in een CPU. Een KD-boom uit het SciPy Python-pakket (Virtanen et al. 2020) met standaardparameters wordt geconstrueerd om de benaderde dichtstbijzijnde buren van elk deeltje te vinden. Vervolgens wordt een Delaunay-triangulatie uitgevoerd op de deeltjes om een beperkt driehoekig netwerk te creëren. We gebruiken een impactkracht van 500 eenheden met een willekeurig impactpunt en een willekeurige impactvector. De spanningspropagatie vindt plaats totdat een drempel van 300 eenheden is bereikt. De PBR wordt uitgevoerd op de CPU door de methoden te implementeren die in de vorige sectie zijn beschreven met behulp van OpenCV en Python.

Resultaten en Discussie

Een belangrijke verschuiving ten opzichte van de meeste eerdere werken in adversariële voorbeelden is dat onze gegenereerde adversariële patronen niet alle pixels in een afbeelding universeel beïnvloeden. Daarom moet de vergelijking alleen worden gemaakt voor het afbeeldingsgebied waar het patroon zich bevindt. Voor dit doel maken we een binaire maskering van elk patroon en geven we de resultaten weer van de objecten die alleen in dat patroon bestaan.

Tabel 1 toont de resultaten van de gemiddelde precisie (AP) onder de adversariële afbeeldingen die zijn gegenereerd met behulp van de twee soorten scheurpatronen (online verzameld en gesimuleerd) voor verschillende klassen. Voor KITTI daalt de AP van andere klassen zoals verwacht met de afname in AP die overeenkomt met het percentage van de afbeelding dat wordt ingenomen door de vrachtwagenklasse, die de grootste daling registreert. Voor BDD100K met PVTv2-B0 zien we dat de daling in AP het grootst is in de gesimuleerde afbeeldingen, maar over het algemeen,

¹<https://www.freepik.com>



Figuur 4: (a) Toont het gesimuleerde beeld met de weg en voertuigen in het brandpuntsvlak (PBR en ver-focus). (b) geeft het gesimuleerde barstpatroon in het brandpuntsvlak weer (PBR en kort focus).

Tabel 1: Gemiddelde precisie (in procenten) van verschillende klassen in KITTI, BDD100k en MS-COCO onder verschillende adversariële afbeeldingen. x geeft de overlay-relatie tussen dataset en glasscheurtype weer. Schoon x Dataset - verwijst direct naar de specifieke afbeeldingen zonder enige adversariële steekproef. RO x Dataset - verwijst naar echte afbeeldingen van gebroken glas die online zijn verzameld en over schone afbeeldingen zijn gelegd. Sim x Dataset - verwijst naar gesimuleerde scheur patronen die over schone afbeeldingen zijn gelegd.

Dataset	IoU-drempel	Categorie	Schoon x Dataset	RO x Dataset	Sim x Dataset
KITTI (YOLOv8)	0.5	Voetganger	25.64	69.72	17.84
		Vrachtwagen	12.39	3.59	3.76
		Car	58.99	50.7	57.73
	0.75	Voetganger	6.83	33.88	6.02
		Vrachtwagen	11.29	2.67	2.79
		Car	31.25	23.85	30.15
BDD100k (PVTv2)	0.5	Voetganger	66.47	54.33	25.95
		Vrachtwagen	61.97	52.83	52.02
		Car	80.37	70.14	56.78
	0.75	Voetganger	27.06	22.72	10.60
		Vrachtwagen	47.03	38.23	42.52
		Car	46.23	45.97	42.99
MS- COCO (Faster R-CNN)	0.5	Persoon	0.035	0.024	0.024
		Voertuigen	2.14	1.45	1.87
		Food	35.34	28.07	30.65
	0.75	Persoon	0.032	0.022	0.023
		Voertuigen	1.56	1.05	1.07
		Food	24.59	18.85	22.00

de trend wordt voortgezet met de voetgangersklasse die de sterkste daling vertoont. Voor MS-COCO hebben we de AP voor de supercategorieën: persoon, voertuigen en voedsel geaggregeerd. Dit komt omdat veel objecten in MS-COCO een kleiner gebied in het beeldkader innemen, waardoor het moeilijk is om betekenisvolle resultaten uit alle categorieën te halen. Een zeer intrigerend resultaat is dat de voetgangersklasse een veelvoudige toename in AP vertoont onder de echte gebroken glaspatronen. Hoewel deze trend misschien contra-intuïtief lijkt, sluit het aan bij de resultaten in Fig. 2 waar het vertrouwen van de auto toeneemt vanwege een rand. Dit toont in feite aan dat de AP sterk afhankelijk is van het barstpatroon, waardoor het uiterst belangrijk is om verdedigingsmethodologieën te creëren om deze adversariële aanvallen te mitigeren.

Ablatiestudies

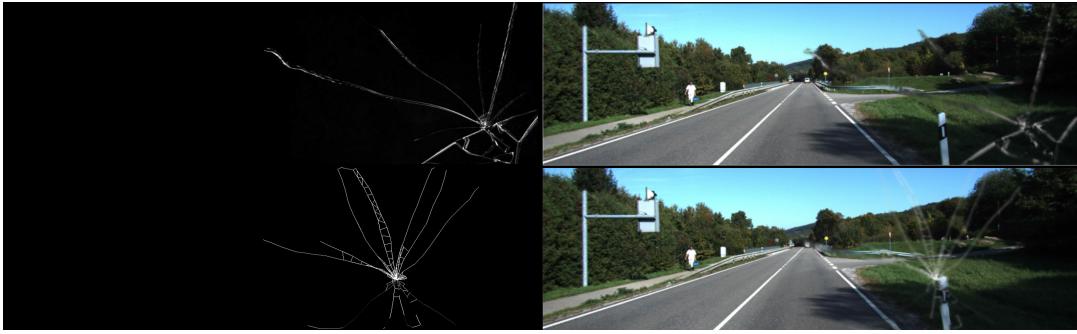
Onze resultaten geven aan dat de gesimuleerde beelden een vergelijkbaar adversarisch effect hebben als de echte beelden. Daarom is een belangrijke ablatiestudie voor ons om te begrijpen hoe dicht de gesimuleerde barstpatronen bij de echte gebroken glaspatronen en die online verzameld zijn. We vormen 5 distributies

- Echt dataset van de weg (afgebeeld in Fig. 2)

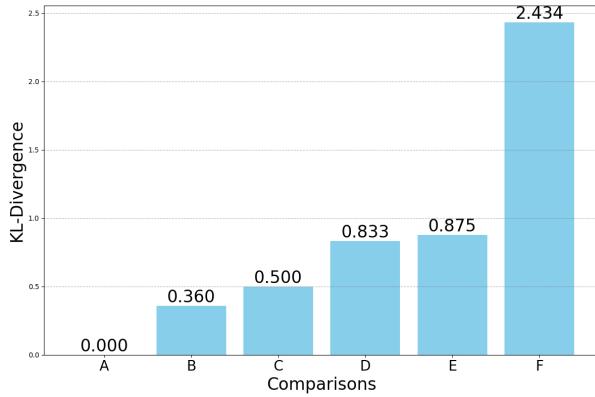
- Barstpatronen verzameld online (Fig. 5 linksboven)
- Gesimuleerde barstpatronen (Fig. 5 linksonder)
- Gesimuleerde barstpatronen overgelegd op KITTI (Fig. 5 rechtsonder)
- Barstpatronen online verzameld, overgelegd op KITTI (Fig. 5 rechtsboven)

We berekenen nu de K-L divergentie tussen al deze distributies om te bepalen hoe vergelijkbaar ze zijn met elkaar (zie Fig. 6). Om een controle te bieden, vergelijken we KITTI met afbeeldingen van katten uit de Kaggle dataset, wat een K-L divergentie van 2.434 oplevert. Op die schaal hebben de PBR-afbeeldingen van gebroken glas een verschil van 0.36 met de echte gebroken glaspatronen, terwijl de gebroken-glasfilters overgelegd op KITTI-afbeeldingen een vergelijkbare K-L divergentie hebben.

Fig. 7 toont een analyse van de rekentijd voor elk van onze modules en over verschillende aantal deeltjes. We voeren deze analyse uit op 100 runs, waarbij we willekeurige impactpunten, impacthoeken en meshstructuren genereren met een vast aantal deeltjes. Het verschil in rekentijd voor verschillende runs kan worden toegeschreven aan het impactpunt en de impacthoek.



Figuur 5: Linksboven - Scheurpatroon online verzameld op Freepik; rechtsboven - online scheurpatroon overgelegd op KITTI; linksonder - gesimuleerd scheurpatroon met PBR; rechtsonder - gesimuleerd scheurpatroon overgelegd op KITTI.



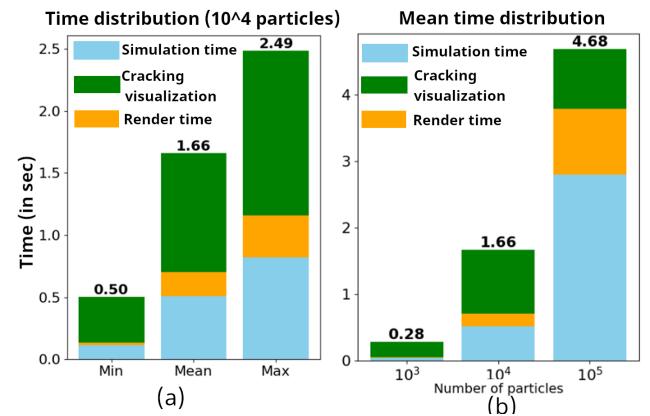
Figuur 6: K-L divergentie van verschillende paren van afbeeldingsdistributies. Datasets: RC - Reëel dataset op de weg (zie Fig. 2), KITTI en Katten. Filters: RO - Reëel (online verzameld) en Sim - Gesimuleerd. K-L divergentie tussen (x - overlay relatie): A - (Sim x KITTI) vs (Sim x KITTI); B - (Sim vs RO); C - (Schoon RC vs KITTI); D - (Gebroken RC) vs (RO x KITTI); E - (Gebroken RC) vs (Sim x KITTI); F - KITTI vs Katten.

De visualisatie van het barsten en de rendertijd variëren ook door verschillende groottes van maskers die worden gevormd door variërende breuk patronen. We variëren ook het aantal deeltjes en zien hoe de looptijd exponentieel toeneemt met de toename van de deeltjes. Al deze runs werden gerenderd op afbeeldingen uit de KITTI dataset met afmetingen van (375 × 1242 × 3).

Conclusie en Toekomstige Scope

We hebben een nieuwe klasse van adversariële fouten geïntroduceerd die voortkomen uit het fysieke proces van fouten in de camera. In dit artikel bieden we een benadering om een realistisch gebroken glaspatroon te genereren vanuit een fysieke simulatie en dat vervolgens in bestaande afbeeldingsdatasets te integreren met behulp van fysiek gebaseerde rendering. We tonen aan dat de gesimuleerde adversariële afbeeldingen kunnen leiden tot significante fouten in objectdetectie.

In dit werk behandelen we black-box adversariële aanvallen die voortkomen uit fysieke fenomenen die in de echte wereld natuurlijk voorkomen, en niet kunstmatig zijn gemaakt om specifieke model



Figuur 7: (a) Gemiddelde tijd die door verschillende modules van onze pijplijn wordt genomen over 100 runs. (b) De minimale, maximale en gemiddelde tijd die door verschillende modules wordt genomen over 100 runs voor een 10^4 deeltjesnet. Voor deze grafieken tonen we de tijd die nodig is voor simulatie (simulatietijd), het omzetten van het net naar glas (barstvisualisatie) en uiteindelijk het renderen (rendertijd).

kwetsbaarheden uit te buiten. We gaan uit van geen kennis van de modelattributen, gewichten of architectuur, waardoor aanvallen overdraagbaar zijn tussen verschillende modellen. Fysieke adversariële methoden (Translucent Patch, RP2) kunnen allemaal worden aangeduid als oclusies van ofwel de camera of de objecten die worden vastgelegd. De adversariële aard komt voort uit het effect van de modelinference als gevolg van deze oclusies. Onze PBR-pijplijn mengt de barsten met bronafbeeldingen als translucente, wazige patronen, wat invloed heeft op de latente ruimte-encoding in plaats van directe oclusie te veroorzaken, resulterend in onjuiste detecties.

Hoewel dit werk specifiek een op fysica gebaseerde methode voor het genereren van gebroken glaspatronen introduceert, omvatten camerastoringen ook andere effecten zoals zonneschittering, overbelichting, onderbelichting, condensatie, enz. Ons toekomstig werk zal zich richten op het creëren van een adversariële gereedschapskist voor de realistische generatie van deze effecten met behulp van fysica en deze vervolgens plaatsen op bestaande afbeeldingsdatasets en auto-simulatieplatforms om verder onderzoek op dit gebied van gedeeltelijke camerastoringen te bevorderen.

Referenties

- Akhtar, N.; en Mian, A. 2018. Bedreiging van adversariële aanvallen op deep learning in computervisie: Een overzicht. *Ieee Access*, 6: 14410–14430.
- Carlini, N.; en Wagner, D. 2017. Adversarial voorbeelden zijn niet gemakkelijk te detecteren: Het omzeilen van tien detectiemethoden. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 3–14.
- Ceccarelli, A.; en Secci, F. 2022. Falen van RGB-camera's en hun effecten in toepassingen voor autonoom rijden. *IEEETransactions on Dependable and SecureComputing*. Coulumb, C.-A. 1776. Essai sur une application des regles des maximis et minimis a quelques problemes de statique relatifs, a la architecture. *Mem. Acad. Roy. Div. Sav.*, 7: 343–387. Dalvi, N.; Domingos, P.; Mausam; Sanghai, S.; en Verma, D. 2004. Adversarial classificatie. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 99–108. Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; Xiao, C.; Prakash, A.; Kohno, T.; en Song, D. 2018. Robuuste fysieke-wereld aanvallen op diepe leermodellen voor visuele classificatie. In *Proceedings ofthe IEEE conference on computer visionand patternrecognition*, 1625–1634. Geiger, A.; Lenz, P.; Stiller, C.; en Urtasun, R. 2013. Vision ontmoet Robotica: De KITTI Dataset. *International Journalof RoboticsResearch(IJRR)*. Goodfellow, I. J.; Shlens, J.; en Szegedy, C. 2014. Verklaren en benutten van adversarial voorbeelden. *arXiv preprintarXiv:1412.6572*. Iben, H. N.; en O'Brien, J. F. 2009. Genereren van oppervlakte scheurpatronen. *GraphicalModels*, 71(6): 198–208. Jocher, G.; Chaurasia, A.; en Qiu, J. 2023. Ultralytics YOLO. Kong, Z.; Guo, J.; Li, A.; en Liu, C. 2020. Physgan: Genereren van fysieke-wereld-resistente adversarial voorbeelden voor autonoom rijden. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14254–14263. Kuna, M. 2013. Eindige elementen in breukmechanica. *Solidmechanics andits applications*, 201: 153–192. Kurakin, A.; Goodfellow, I. J.; en Bengio, S. 2018. Adversarial voorbeelden in de fysieke wereld. In *Artificial intelligence safety and security*, 99–112. Chapman and Hall/CRC. Li, J.; Schmidt, F.; en Kolter, Z. 2019. Adversarial camerastickers: Een fysieke camera-gebaseerde aanval op diepe leersystemen. In *International conferenceon machine learning*, 3896–3904. PMLR. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Doll'ar, P.; en Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V* 13, 740–755. Springer.
- Liu, Y.; Xing, Y.; Li, C.; Yang, C.; en Xue, C. 2021. Analyse van lensbreuk in precisieglastypen met de eindige-elementenmethode. *Applied Optics*, 60(26): 8022–8030.
- Nguyen, A.; Yosinski, J.; en Clune, J. 2015. Diepe neurale netwerken zijn gemakkelijk te misleiden: Hoge vertrouwensvoorspellingen voor onherkenbare beelden. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 427–436. O'Brien, J. F.; en Hodgins, J. K. 1999. Grafische modellering en animatie van broze breuk. In *Proceedingsof ACM SIGGRAPH 1999*, 137–146. ACM Press/Addison-Wesley Publishing Co. Pfaff, T.; Narain, R.; De Joya, J. M.; en O'Brien, J. F. 2014. Adaptief scheuren en barsten van dunne platen. *ACM Transactions on Graphics (TOG)*, 33(4): 1–9. Pharr, M.; Jakob, W.; en Humphreys, G. 2023. *Fysiek gebaseerde rendering: Van theorie tot implementatie*. MIT Press. PK, A. ??? Kaggle katten en honden mini dataset.
h
t
t
p
s://
W
W
w.
kaggle.com/datasets/aleemaparakatta/cats-and-dogs-mini-dataset.
Geraadpleegd: 2024-09-30. Rankine, W. J. M. 1857. II. Over de stabiliteit van losse aarde. *Philosophical transactions of the Royal Society of London*, (147): 9–27. Ren, S.; He, K.; Girshick, R.; en Sun, J. 2016. Snellere R-CNN: Naar real-time objectdetectie met regio-voorstelnetwerken. *IEEEtransactions on pattern analysis and machineintelligence*, 39(6): 1137–1149. Rouxel, T.; en Brow, R. K. 2012. De stroming en breuk van geavanceerde glazen—een overzicht. *International Journal of Applied Glass Science*, 3(1): 1–2. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; en Fergus, R. 2013. Intrigerende eigenschappen van neurale netwerken. *arXiv preprintarXiv:1312.6199*. Tu, J.; Ren, M.; Manivasagam, S.; Liang, M.; Yang, B.; Du, R.; Cheng, F.; en Urtasun, R. 2020. Fysiek realiseerbare vijandige voorbeelden voor LiDAR objectdetectie. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 13716–13725. van Schrik, D. 1997. Opmerkingen over terminologie op het gebied van supervisie, foutdetectie en diagnose. *IFAC Proceedings Volumes*, 30(18): 959–964. Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; van der Walt, S. J.; Brett, M.; Wilson, J.; Millman, K. J.; Mayorov, N.; Nelson, A. R. J.; Jones, E.; Kern, R.; Larson, E.; Carey, C. J.; Polat, I.; Feng, Y.; Moore, E. W.; VanderPlas, J.; Laxalde, D.; Perktold, J.; Cimrman, R.; Henriksen, I.; Quintero, E. A.; Harris, C. R.; Archibald, A. M.; Ribeiro, A. H.; Pedregosa, F.; van Mulbregt, P.; en SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamentele algoritmen voor wetenschappelijk rekenen in Python. *Nature Methods*, 17: 261–272. Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; en Shao, L. 2022. Pvt v2: Verbeterde baselines met pyramid vision transformer. *Computational visualmedia*, 8(3): 415–424.

- Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V.; en Darrell, T. 2020. BDD100K: Een divers rijdataset voor heterogeen multitask leren. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2636–2645.
- Zolfi, A.; Kravchik, M.; Elovici, Y.; en Shabtai, A. 2021. De translucent patch: Een fysieke en universele aanval op objectdetectoren. In *Proceedings of the IEEE/CVF conference on computervision andpattern recognition*, 15232–15241.

Algoritme van spanningspropagatie

Algoritme 1 beschrijft de procedure voor het simuleren van de propagatie van spanning door een materiaal na een impactgebeurtenis. Het algoritme neemt als invoer de locatie van de impact (pt), de grootte van de impactkracht (F), de impactrichtingsvector (v) en de ouderzijde (PE) die geassocieerd is met de impactlocatie. Het gebruikt ook een nabijste buurstraal R om de set van kandidaatlocaties voor spanningspropagatie te bepalen.

Algoritme 1: Spanningspropagatie

```
1:  $pt \leftarrow$  Impactpunt
2:  $F \leftarrow$  Impactkracht
3:  $PE \leftarrow$  Ouderzijde
4:  $v \leftarrow$  Impactvector
5:  $R \leftarrow$  Nabijste buurstraal
6:
7: procedure PROPAGEREERSTRESS( $Pt, F, V, PE$ )
8:    $frontiers \leftarrow KDTTree - queryRadius(R)$ 
9:    $NN \leftarrow \frac{frontiers - pt}{\|frontiers - pt\|}$ 
10:   $\cos(\theta) \leftarrow NN \cdot v$ 
11:   $stress \leftarrow calculateStress(\cos(\theta) F)$ 
12:   $frontiers \leftarrow frontiers[argmax(stress)]$ 
13:   $v \leftarrow v[argmax[stress]]$ 
14:   $PE \leftarrow PE[argmax[stress]]$ 
15:  PROPAGATESTRESS( $Pt, F, V, PE$ )
16: einde procedure
```

Eerst gebruikt het een KD-boom datastructuur om efficiënt alle punten (grenzen) binnen een gegeven straal R van het impactpunt op te vragen. Voor elke grens berekent het een eenheidsrichtingsvector van het impactpunt naar de grens (NN). Vervolgens projecteert het de impactvector v op deze richting om de cosinusgelijkenis $\cos(\theta)$ te verkrijgen, waarmee de hoekrelatie tussen de impactrichting en de kandidaat-propagatierichting wordt vastgelegd. Voor elke kandidaat wordt de resulterende waarde gebruikt, samen met de impactkracht, om de overeenkomstige spanning op dat punt te berekenen. Het algoritme selecteert vervolgens de kandidaat met de maximale spanningswaarde. De impactvector v en ouderzijde PE worden bijgewerkt om overeen te komen met deze nieuwe richting. Het proces wordt recursief herhaald, waardoor de gesimuleerde spanningsgolf iteratief door het materiaal kan voortplanten langs het pad van de grootste spanningsoverdracht.

Deze benadering is bedoeld om na te bootsen hoe spanning vanaf een impactpunt zich waarschijnlijk door een materiaal zal verspreiden—voorkeur gevend aan paden die worden gedefinieerd door zowel geometrische nabijheid als mechanische uitlijning met de oorspronkelijke impact.

De uiteindelijke output van de simulatie is de realisatie van het mesh als een afbeelding die overeenkomt met het patroon van een gebroken lens (eindafbeelding van Fig. 8).

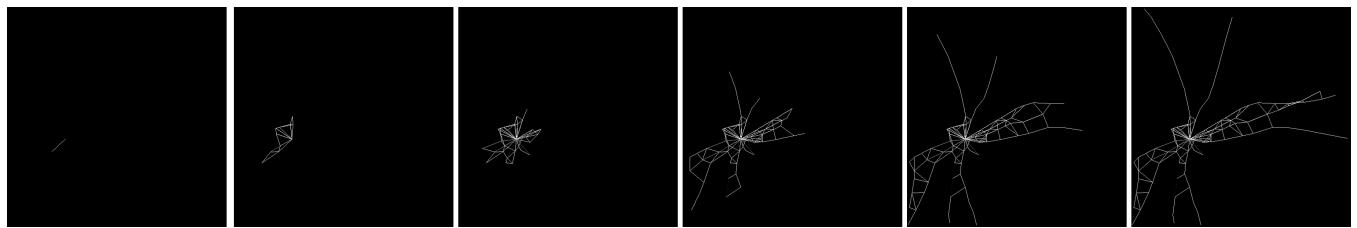


Figure 8: Een animatie van het breken van een lens gesimuleerd door het spanningsveld in te stellen en PB toe te passen R.

Statisch Experiment

Om het effect van deze breuken op de resulterende beelden te begrijpen, voeren we eerst een statisch experiment binnenshuis uit zoals vermeld in Sectie Inleiding. Voor dit experiment gebruiken we verschillende geharde glasplaten, die we willekeurig breken met een kleine hamer met één of meerdere breekpunten. Vervolgens plaatsen we een 36 MP JVC GC-PX10 hybride camera gemonteerd op een statief met een klem voor het geharde glas.

Fig. 9(a) toont de gedetailleerde opstelling met de camerabevestiging en het geharde glas dat met een klem op zijn plaats wordt gehouden. Fig. 9(b) toont het beeld dat door de camera is vastgelegd en Fig. 9(c) toont het enkele voertuig dat als het primaire object wordt vastgelegd door de camera door het geharde glas. De scène wordt verlicht met bovenliggende fluorescentielampen.

Fig. 10 toont enkele van de breuk- en kraspatronen op het geharde glas. Deze patronen werden opzettelijk gerandomeerd, waarbij meerdere brandpunten en verschillende niveaus van kracht werden gebruikt om de onvoorspelbare en gevarieerde aard van echte glasschade na te bootsen. Door diverse krachten toe te passen, konden we een spectrum van breuken en krassen produceren, variërend van fijne oppervlaktebeschadigingen tot meer uitgesproken breeuken. Deze aanpak werd gekozen om de soorten schade die glassurfaces in werkelijke omstandigheden kunnen tegenkomen—zoals die veroorzaakt door impact, puin of omgevingsstressoren—nauwkeurig te repliceren, waardoor de relevantie en realisme van onze experimentele opstelling worden gewaarborgd. Deze representatieve schadepatronen stellen ons in staat om effectiever de invloed van glasimperfecties op sensorprestaties en objectdetectie-algoritmen te analyseren.

Twee verschillende breeukpatronen en hun resulterende beelden worden getoond in Fig. 11 en Fig. 12. We willen opmerken dat we de brandpuntsafstanden van de camera aanzienlijk hebben gevarieerd om te begrijpen hoe de beelden eruitzien bij dichtbij- en veraf-focus. De resultaten laten zien dat zelfs kleine kraspatronen zichtbaar zijn in de beeldoutput, terwijl een veel sterker multi-breeukpatroon bijna het hele beeld kan vervagen. Dit experiment biedt de intuïtie waarop ons simulatie- en visualisatieframework is gebouwd.

Verhoogde AP voor voetgangers in KITTI

We willen erop wijzen dat de verhoogde AP voor de voetgangersklasse iets was waar zelfs wij aanvankelijk door verrast waren. Echter, een zorgvuldige kwalitatieve diepgaande analyse hielp ons te begrijpen dat dit gebeurde als gevolg van de glasbreuken die het voor het model gemakkelijker maakten om voetgangers te classificeren vanwege verbeterde randen rondom hen. Dit was geen randartefact, maar in plaats daarvan fungeerde de glasbreuk als een extra randgrens die de voetganger en de achtergrond duidelijk scheidde. Een vergelijkbaar resultaat werd ook waargenomen in [1] waar de algehele AP werd verhoogd in adversariële beelden.



(a)



(b)

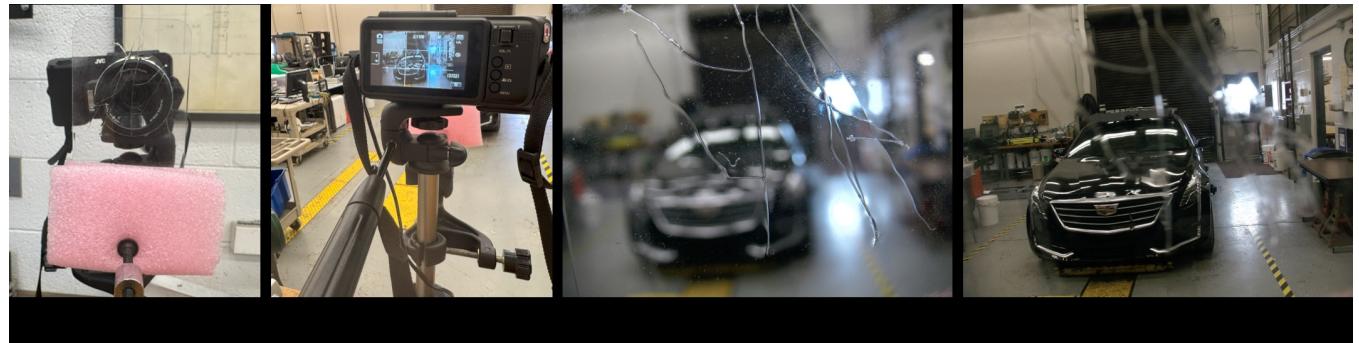


(c)

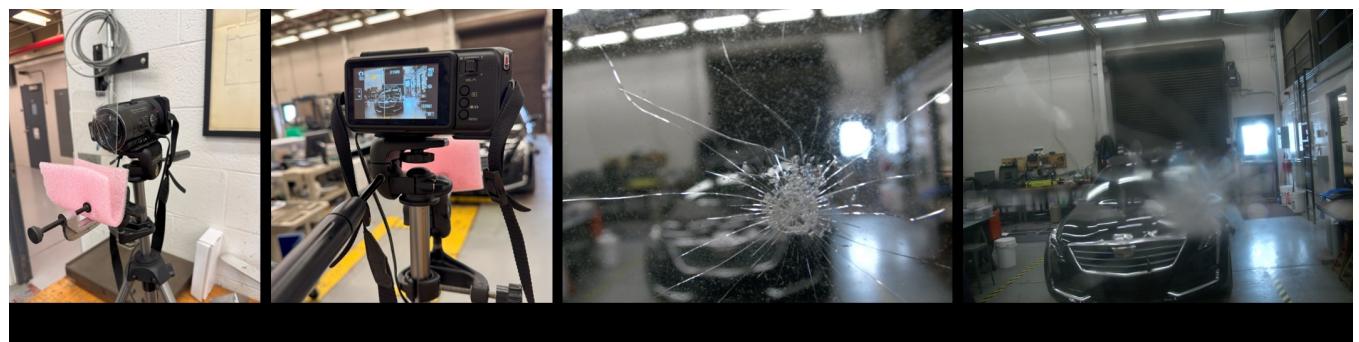
Figuur 9: Experimentele opstelling voor het verzamelen van beelden die worden beïnvloed door bekraaste/gebroken buitenlagen voor een camera. (a) toont de volledige opstelling voor het maken van adversariële beelden. (b) toont de positie van de camera ten opzichte van de scène die wordt vastgelegd. (c) toont de scène die door de camera wordt vastgelegd.



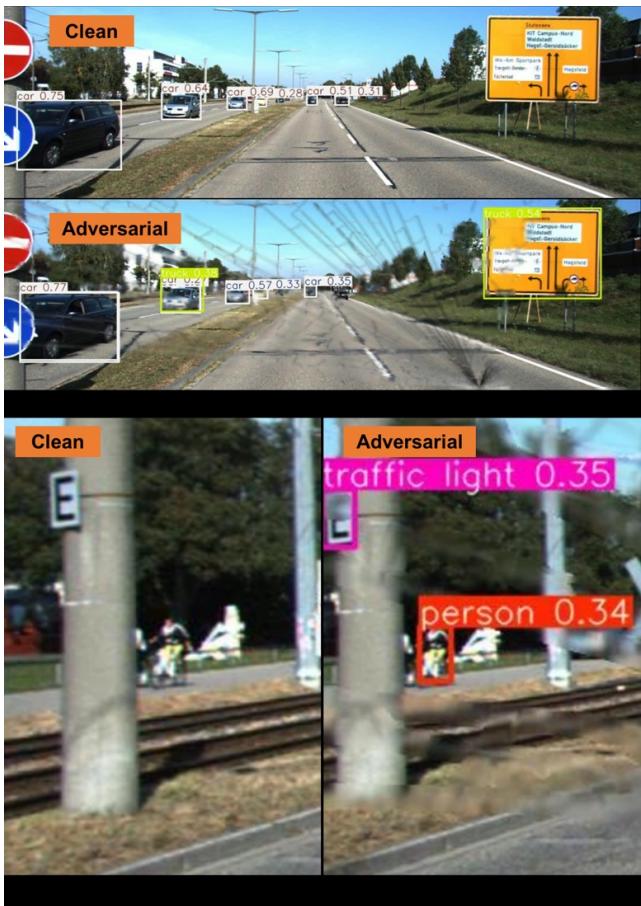
Figuur 10: Enkele breuken/krassenpatronen op het glas dat we gebruikten voor het verzamelen van de afbeeldingen. (a) Een scherpe kracht die loodrecht op het glasoppervlak wordt uitgeoefend, waardoor breuken radiaal ontstaan. (b) en (c) repliceren een glas met krasen.



Figuur 11: (a) Toont het krasenpatroon geplaatst voor de camera, (b) toont het camerastandpunt. (c) toont de afbeelding vastgelegd door de camera (kortere focus). (d) toont de afbeelding vastgelegd door de camera (verre focus).



Figuur 12: (a) Toont het gebroken glaspatroon voor de camera, (b) toont het camerastandpunt. (c) toont de afbeelding vastgelegd door de camera (kortere focus). (d) toont de afbeelding vastgelegd door de camera (verre focus).



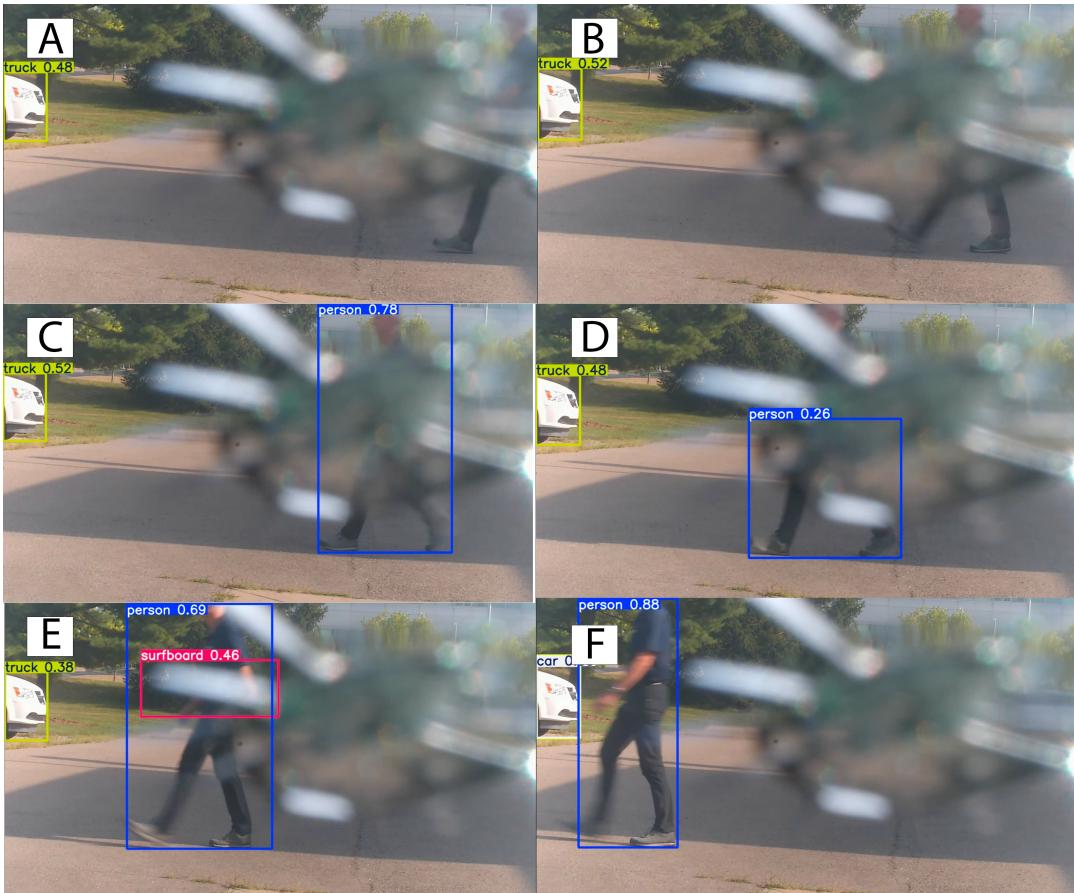
Figuur 13: (a) Boven - detecties op een schoon beeld; onder - detecties op een adversarief beeld. (b) YoLo slaagt er niet in de persoon te detecteren (c) Glasbreuken stellen het model in staat de persoon te detecteren.

Dynamisch Experiment

In deze sectie beschrijven we het dynamische experiment dat wordt genoemd in Sectie Inleiding. We voeren dit experiment uit om de temporele verstoring te begrijpen die door een barst wordt geïntroduceerd. We gebruiken een barst in de voorruit van een voertuig en plaatsen een kleine camera op het dashboard achter de barst. Vervolgens fotograferen we twee dynamische objecten - een voertuig en een voetganger terwijl ze door de scène bewegen. Fig. 14 biedt enkele specifieke beeldframes met inferentie van YOLOv8 voor de voertuigklasse. We tonen aan dat met de barst het voertuig in de meeste frames onopgemerkt blijft. Bovendien bevat bijna elk frame een vals-positieve detectie. Overeenkomstig presenteren we Fig. 15 als de frames met een persoon die door de scène loopt. We tonen aan dat het af en toe detectie biedt en soms met een verkeerde klasse (surfplank).



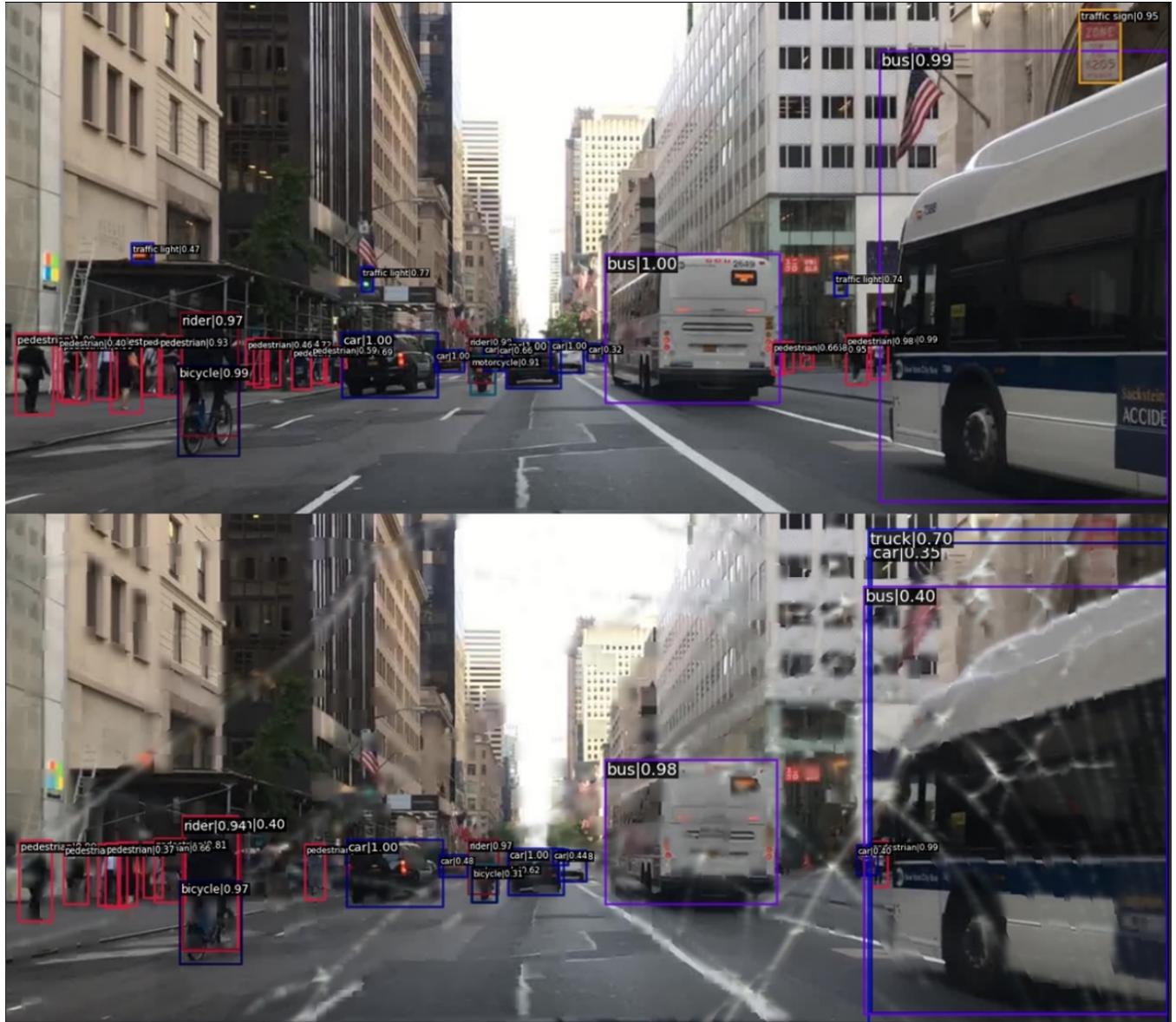
Figuur 14: Specifieke frames van de beelden genomen met de voorruitbarst met YOLOv8-inferentie voor de voertuigklasse. A - vals positief zonder object in de scène; B - geen inferentie op voertuig; C - geen inferentie op voertuig; D - eerste detectie op voertuig; E - twee verschillende detecties op hetzelfde voertuig; F - verkeerde begrenzingsvakgebied.



Figuur 15: Specifieke frames van de beelden genomen met de voorruitbarst met YOLOv8-inferentie voor de klasse persoon. A - eerste verschijning van persoon in scène zonder detectie; B - geen inferentie van persoon; C - eerste detectie van persoon; D - gedeeltelijke detectie van persoon; E - detectie van persoon met andere klasse; F - volledige detectie van persoon.

Echte glasbreukafbeeldingen

We presenteren een voorbeeld van de glasbreukafbeeldingen verzameld van de FreePik-website, overgelegd op het KITTI Dataset samen met YOLOv8-inferentie (Fig. 16). We tonen aan dat de breuk enkele detecties verwijderd en het vertrouwen in andere detecties vermindert.



Figuur 16: Boven - Inferentie van PTv2 op een schoon beeld uit BDD100K. Onder - Inferentie voor een echte gebroken glas afbeelding over BDD100K gelegd ter vergelijking. We zien twee extra fout-positieven aan de rechterkant (vrachtwagen, auto) en verschillende fout-negatieven voor de voetgangersklasse aan de linkerkant van de adversariële afbeelding.