

Zerbrochenes Glas, versagende Kameras: Simulation physikbasierter adversarialer Muster für autonome Fahrsysteme

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

Zusammenfassung

Während sich die jüngste Forschung stark auf die Erzeugung physikalisch basierter adversarialer Muster konzentriert hat, stammt eine kritische, jedoch oft übersehene Kategorie aus physischen Ausfällen innerhalb der Bordkameras – Komponenten, die für die Wahrnehmungssysteme autonomer Fahrzeuge unerlässlich sind. Kameraausfälle, sei es durch äußere Belastungen, die zu Hardwareausfällen führen, oder durch interne Komponentenfehler, können die Sicherheit und Zuverlässigkeit autonomer Fahrsysteme direkt gefährden. Erstens motivieren wir die Studie durch zwei separate Experimente in der realen Welt, um zu zeigen, dass Glasfehler tatsächlich dazu führen würden, dass die auf Erkennung basierenden neuronalen Netzmodelle versagen. Zweitens entwickeln wir eine simulationsbasierte Studie, die den physikalischen Prozess des Glasbruchs nutzt, um gestörte Szenarien zu schaffen, die eine realistische Klasse physikalisch basierter adversarialer Muster darstellen. Mit einem auf dem Finite-Elemente-Modell (FEM) basierenden Ansatz erzeugen wir Oberflächenrisse auf dem Kamerabild, indem wir ein Spannungsfeld anwenden, das durch Partikel innerhalb eines Dreiecksnetzes definiert ist. Schließlich verwenden wir physikalisch basierte Rendering-Techniken (PBR), um realistische Visualisierungen dieser physikalisch plausiblen Brüche bereitzustellen. Um die Sicherheitsimplikationen zu bewerten, wenden wir die simulierten gebrochenen Glaseffekte als Bildfilter auf zwei autonome Fahrdatensätze an - KITTI und BDD100K - sowie auf den groß angelegten Bilddetektionsdatensatz MS-COCO. Anschließend bewerten wir die Erkennungsfehlerraten für kritische Objektklassen mit CNN-basierten Objekterkennungsmodellen (YOLOv8 und Faster R-CNN) und einer transformerbasierten Architektur mit Pyramid Vision Transformers. Um die verteilungsmäßigen Auswirkungen dieser visuellen Verzerrungen weiter zu untersuchen, berechnen wir die Kullback-Leibler (K-L)-Divergenz zwischen drei verschiedenen Datenverteilungen, indem wir verschiedene gebrochene Glasfilter auf einen benutzerdefinierten Datensatz (aufgenommen durch eine gesprungene Windschutzscheibe) sowie auf die KITTI- und Kaggle-Katzen-und-Hunde-Datensätze anwenden. Die K-L-Divergenzanalyse legt nahe, dass diese gebrochenen Glasfilter keine signifikanten verteilungsmäßigen Verschiebungen einführen. Unser Ziel ist es, eine robuste, physikalisch basierte Methodik zur Erzeugung adversarialer Muster bereitzustellen, die reale Kameraausfälle widerspiegeln, mit dem übergeordneten Ziel, die Widerstandsfähigkeit und Sicherheit autonomer Fahrsysteme gegen solche physischen Bedrohungen zu verbessern.

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

Einleitung

Kameras sind allgegenwärtig als Fernsensoren, die Daten aus einer unstrukturierten und dynamischen externen Umgebung sammeln, oft unter rauen Bedingungen. Ein Ausfall oder Fehler in einem Sensor ist eine Abweichung vom funktionalen Zustand in mindestens einem bestimmten Parameter des Systems (van Schrick 1997). Diese Fehler können aufgrund interner (wie Abnutzung) oder externer (Temperatur, Feuchtigkeit usw.) Ursachen auftreten. Bei R GB-Kameras umfassen interne Ursachen tote Pixel, während externe Ursachen gebrochene Gehäuse oder äußere Linsen und Kondensation umfassen. Diese abrupten Ausfälle sind schwer zu erkennen und beeinträchtigen die Objekterkennungsalgorithmen negativ – sie verringern die Genauigkeit und führen oft zu Halluzinationen, wie in Abb. 1 gezeigt. Die in einem automatisierten Fahrzeug (AV) auftretenden Ausfälle können beispielsweise zu kritischen Sicherheitsproblemen führen, die zu Unfällen und in einigen Fällen zu Todesfällen führen.

Derzeit gibt es nach bestem Wissen der Autoren keine strengen Methoden zur Erzeugung von kamerabasierten Sensorfehlern (Ceccarelli und Secci 2022).

In dieser Arbeit konzentrieren wir uns auf den Sensorausfall, der durch Risse in einem Glas, das eine Kamera (oder ein Kamera-Gehäuse) bedeckt, verursacht wird, obwohl der in diesem Papier beschriebene Prozess für alle in (Ceccarelli und Secci 2022) aufgeführten Kameraausfälle verwendet werden kann. Diese Glasbruch-Effekte in einer Kamera können durch ein äußeres Objekt, das die Kamera trifft, oder als Folge von plötzlicher Hitze- und/oder Druckentwicklung innerhalb des Gehäuses verursacht werden. Im Jargon der neuronalen Netze wird ein unter solchen Bedingungen aufgenommenes Bild als adversariales Beispiel betrachtet. Frühere Forschungen (Akhtar und Mian 2018; Carlini und Wagner 2017; Szegedy et al. 2013) zeigen, dass selbst kleine Mengen an Störungen, die manchmal schwer mit dem menschlichen Auge zu erkennen sind, ausreichen, um die neuronalen Netze vollständig zu täuschen, wobei eine subtile Änderung der Eingaben zu einer drastischen Änderung der Ausgaben führen kann. Wir möchten darauf hinweisen, dass (Li, Schmidt und Kolter 2019) ein physikales, kamerabasiertes adversariales Angriffsparadigma bereitgestellt haben, das als die am engsten verwandte Arbeit in diesem Bereich dient. Sie präsentierten eine Modifikation des Bildes durch eine Überlagerung eines durchsichtigen, sorgfältig gestalteten Aufklebers, der zu einer Fehlklassifizierung führte.

Um die Auswirkungen dieser Brüche auf die resultierenden Kamerabilder zu verstehen, führen wir zwei unterschiedliche Experimente durch: eines

*Korrespondierender Autor Copyright © 2026, Association for the Advancement of Artificial Intelligence(www.aaai.org). Alle Rechte vorbehalten.

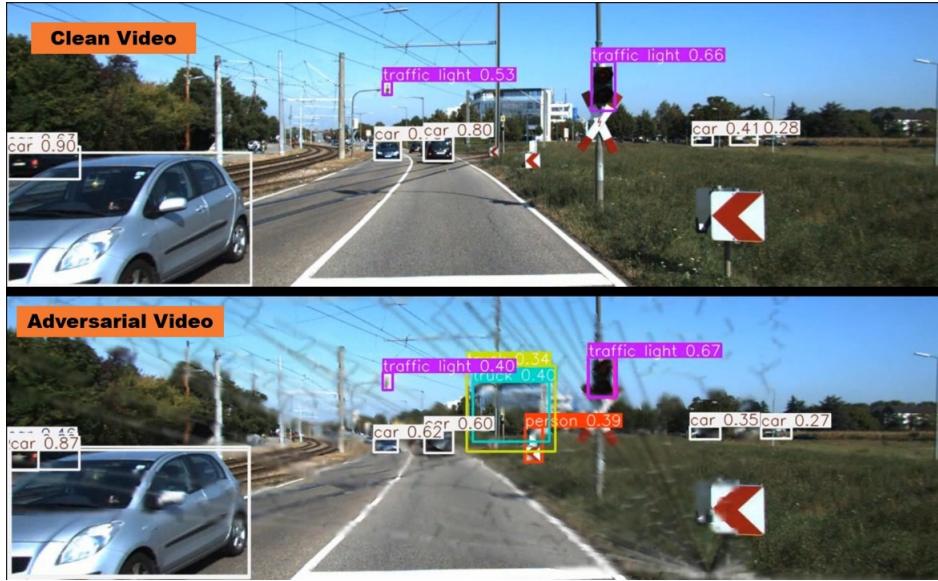


Abbildung 1: Ein qualitativer Vergleich von sauberen vs. adversarialen Videos, die mit unserer Simulations- und Rendering-Methode auf KITTI erzeugt wurden. Dieses Bild zeigt Fehlalarme und verringerte Vertrauensniveaus für echte Positive. Siehe das Ergänzungsmaterial für das vollständige Video.

in einer statischen Innenumgebung und das andere in einer dynamischen Außenumgebung. Das erste Experiment beinhaltete das Zerbrechen von gehärtetem Glas und das Platzieren vor der Kamera (siehe Abb. 2(a)) mit einem statischen Fahrzeug in der Szene, um zu verstehen, wie verschiedene Bruchmuster die Qualität und das Erscheinungsbild der Szene beeinflussen. Wir erfasssten Bilder bei unterschiedlichen Brennweiten, um die Variabilität solcher Störungen zu beurteilen. Dies half uns, bestimmte qualitative Fragen über das visuelle Erscheinungsbild dieser Brüche in Bezug auf ihre Ausbreitung und Intensität zu beantworten, was unseren Ansatz in den Abschnitten Fokalebene und Physische Angriffssimulation motivierte. Der experimentelle Aufbau und die detaillierten experimentellen Ergebnisse sind in Abschnitt Statisches Experiment der Ergänzung zu finden. Das zweite Experiment (Abb. 2(b)) bestand darin, ein Outdoor-Video mit dynamischen Fahrzeugen bei Tageslichtbedingungen aufzunehmen, indem eine MobileEye-Kamera neben einem in Abb. 2 dargestellten Windschutzscheibenriss (oben links gezeigt) platziert wurde und eine Inferenz mit YOLOv8 (Jocher, Chaurasia, und Qiu 2023) durchgeführt wurde, um ein grundlegendes Verständnis der Auswirkungen solcher Szenarien auf Objekterkennungsnetzwerke zu gewinnen. Wir beobachteten, dass das Modell das Fahrzeug in einem sauberen Bild leicht erkennen kann, während es bei der Erkennung versagt (unten rechts) oder falsche Positive erzeugt (unten links). Interessanterweise kann das Vorhandensein eines Risses auch unerwartet das Vertrauen in die Vorhersage des Autos mit einer klar definierten Kante erhöhen (0,92 unten links vs. 0,75 oben links). Die detaillierten Inferenz-Ergebnisse mit Fahrzeug- und Person-Klasse sind in Abschnitt Dynamisches Experiment der Ergänzung angegeben.

Wir suchten dann online nach echten Bildern von zerbrochenem Glas (Abschnitt Echte Glasbruchbilder der Ergänzung), konnten jedoch keinen ausreichend großen Datensatz erstellen, um einen datengetriebenen Ansatz für die Abwehr von Angriffen unter diesen Bedingungen zu ermöglichen. Zusätzlich experimentierten wir mit CGI-Tools wie Maya und Blender,

um solche Effekte zu erzeugen, aber sie fehlen an Flexibilität, Kontrolle, Maßstab und Physik, um diese Bedingungen zu simulieren. Die nächstliegende Simulationsoption in der vorhandenen Literatur ist ArcSim (Pfaff et al. 2014). Allerdings sind ihre hochauflösenden Simulationsausgaben extrem langsam (≈ 20 Stunden), was es schwierig macht, sie zu skalieren. Daher richteten wir unsere Bemühungen darauf, eine skalierbare, simulationsbasierte Pipeline zu schaffen, um Brüche zu erzeugen, die zur Weiterentwicklung des Wahrnehmungstapels verwendet werden können.

Bei einem Glasbruch können der Hauptpunkt, die Kraft und der Einfallswinkel zufällig sein, aber die Ausbreitung und das resultierende Muster folgen einem inhärent physikalischen Prozess (entweder linear oder radial). Wir erstellen daher eine Bruchsimulation basierend auf Partikeln in einem zufällig erzeugten Dreiecksnetz und führen die Spannungsverteilung durch das Netz durch. Unsere Simulation ermöglicht es uns, die Brüche innerhalb eines Dreiecksnetzes zu jedem diskreten Zeitpunkt δt zu erzeugen. Wir verwenden OpenCV, um das gegebene Netz in ein entsprechendes zerbrochenes Glas-Musterbild umzuwandeln. Anschließend nutzen wir physikalisch basiertes Rendering (PBR) (Pharr, Jakob, and Humphreys 2023), um die Oberflächenbrüche realistisch darzustellen, indem wir die bidirektionale Reflexionsverteilungsfunktion (BRDF) verwenden, um die Menge des von einem bestimmten Punkt auf einer Oberfläche reflektierten Lichts zu berechnen, die durch einfallende Lichtquellen verursacht wird.

Indem wir unseren Rendering-Ansatz mit drei beliebten Open-Source-Datensätzen kombinieren - KITTI (Geiger et al. 2013), BDD100k (Yu et al. 2020) und MS-COCO (Lin et al. 2014), sind wir in der Lage, adversariale Bilder effizient zu generieren. Ein gängiger Prozess zum Testen der generierten adversarialen Bilder besteht darin, die Anzahl der falsch-positiven/negativen über den Bildraum zu finden. In unserem Fall können wir jedoch aufgrund des lokalen adversarialen Effekts nicht einfach auf eine bildbasierte Messung vertrauen. Daher verwenden wir die adversarialen Bilder (ähnlich der unteren linken Abbildung in Abb. 2) und extrahieren die Objekte

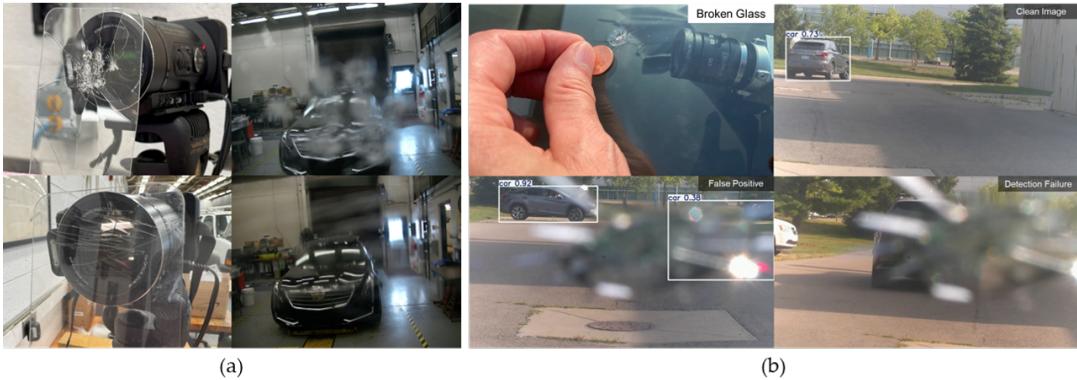


Abbildung 2: (a) Statisches Experiment im Innenbereich. Links: Kamera mit 2 verschiedenen Mustern von gebrochenem Sicherheitsglas; rechts - Bilder des Fahrzeugs unter den verschiedenen Brüchen. (b) Dynamisches Experiment im Außenbereich. Oben links - ein münzgroßer Riss in der Windschutzscheibe; oben rechts - sauberes Bild mit dem Fahrzeug, das mit YOLOv8 erkannt wurde; unten links - falsch positives Ergebnis durch den Riss; unten rechts - Erkennungsfehler durch das Glas. Weitere Beispiele aus diesen Experimenten sind im Ergänzungsmaterial enthalten.

die sich in dem Bereich befinden, in dem der Bruch existiert, unter Verwendung der Ground-Truth-Bounding-Boxen. Wir nutzen dann YOLOv8, Faster R-CNN (Ren et al. 2016) und Pyramid Vision Transformer (PVTv2) (Wang et al. 2022), um den Prozentsatz der Objekte zu finden, die fehlschlagen, wenn die adversarialen Filter angewendet werden. Wir bieten auch Ablationsstudien an, um die verteilungsbedingten Unterschiede zwischen den drei Bildersätzen zu verstehen: Experimentell gesammelte echte zerbrochene Glasbilder, online gesammelte echte zerbrochene Glasbilder und die generierten Bilder. Wir berechnen die Kullbeck-Liebler-Divergenz für diese Bildverteilungen, um die Ähnlichkeit der generierten Bilder mit den echten zerbrochenen Glasbildern zu beweisen. Wir verwenden Katzenbilder aus dem Kaggle Cats and Dogs Datensatz als Kontrolle, um den Unterschied zwischen Bildverteilungen (PK) zu verstehen.

Die wesentlichen Beiträge des Papiers lassen sich wie folgt zusammenfassen:

- Wir bieten eine neuartige Methode zur Abstraktion von Glasbrüchen durch eine Kombination von Spannungsverbreitungsmethoden und minimalen Spannbäumen, um physikalisch fundierte gebrochene Glasstrukturen zu erzeugen.
- Wir präsentieren einen PBR-Ansatz, um ein realistisches Rendering von Kameraausfällen zu ermöglichen, das mit jeder Art von bestehenden Computer-Vision-Datensätzen - sowohl Bilder als auch Videos - verwendet werden kann.
- Unsere Simulations- und Rendering-Pipelines sind skalierbar und recheneffizient ($\approx 1.6s$), sodass sie sowohl von der Wissenschaft als auch von der Industrie zur Verbesserung der Robustheit und des Schutzes vor Verteilungen für eine Vielzahl von Anwendungen genutzt werden können.

Hintergrund

Physikbasierte adversarielle Muster

Das Problem des adversarialen Beispiels kann wie folgt definiert werden: Für ein Modell M , das eine Eingabeprobe X korrekt in ihre vorgesehene Klasse klassifiziert, d.h. $M(X) = y_{true}$, führt das Hinzufügen eines „Fehlers“ ϵ zur Eingabeprobe X zu einer veränderten Probe X' , sodass $M(X') \neq y_{true}$. Somit führt die Injektion des Fehlers ϵ zu einem adversarialen Beispiel, das das Modell zum Scheitern bringt.

Obwohl die Idee der adversarialen Manipulation des Modells im Kontext des maschinellen Lernens schon vor einiger Zeit identifiziert wurde (Dalvi et al. 2004), lag der Fokus im letzten Jahrzehnt eindeutig auf den adversarialen Angriffen auf neuronale Netzwerke (Szegedy et al. 2013; Goodfellow, Shlens und Szegedy 2014). In diesen Arbeiten zeigten die Forscher, dass eine kleine gezielte Injektion von Rauschen, die für das menschliche Auge fast nicht wahrnehmbar ist, die Labels vollständig veränderte (Szegedy et al. 2013) und umgekehrt Bilder erzeugt werden konnten, die für Menschen völlig unkenntlich aussahen, aber von den DNNs perfekt klassifiziert wurden (Nguyen, Yosinski und Clune 2015).

Während diese adversariellen Muster das Modell auf mögliche Fehler untersuchen, fehlt ihnen jeglicher physikalischer Realismus bei ihrer Erstellung und sie benötigen Zugriff auf das Modell. Um dies zu adressieren, hat sich die jüngste Forschung darauf konzentriert, physikalisch relevante adversarielle Muster zu entwickeln. Einer der ersten Vorstöße in diesem Bereich wurde von (Kurakin, Goodfellow und Bengio 2018) unternommen, die die Genauigkeit der Modelle in der physischen Welt ins Visier nahmen, indem sie verrauchte Bilder von einer Handy-Kamera einspeisten, die das Modell dazu brachten, einen großen Teil der Muster falsch zu klassifizieren. In ähnlicher Weise demonstrierten (Eykholt et al. 2018), dass echte Verkehrsschilder mit einfachen, strategisch platzierten physischen Aufklebern so gestört werden können, dass sie selbst bei Blickwinkeländerungen die modernsten DL-Algorithmen nahezu perfekt täuschen. Andere Forscher haben adversarielle Bilder (Kong et al. 2020), Translucent Patches auf der Kamera (Zolfi et al. 2021) oder künstliche LiDAR-Oberflächen (Tu et al. 2020) platziert, um Muster zu erzeugen, die Objektdetektoren täuschen. Während diese vorherige Forschung Physik zur Erzeugung der Muster verwendet, stammen sie nicht aus der Modellierung eines rigorosen physikalischen Prozesses, und wir zielen darauf ab, diese Lücke in dieser Arbeit zu schließen.

Theorie des gebrochenen/zerbrochenen Glases

Das Thema, wie Glas bricht und wie es sich ausbreitet, ist nach wie vor eine offene Forschungsfrage, die mit mehreren vorgeschlagenen physikalischen Theorien umstritten ist (Roussel und Brow 2012). Während das mikroskopische Verfahren des Glasbruchs diskutiert wird, ist auf makroskopischer Ebene das Rissverhalten

gut verstanden. (Liu et al. 2021) analysierten den Prozess des Glaslinsenbruchs in der Präzisionsglasformungsanwendung unter Verwendung von FEM mit einem dreidimensionalen Modell in einer physikalischen Simulationssoftware. Die physikalischen Parameter wurden in die Software eingegeben und die Risspfade anhand der Simulationsergebnisse analysiert. Die Autoren führten eine Temperatur- und Spannungssimulation eines hochpräzisen dreidimensionalen Netzmodells des geformten Glases durch. (Iben und O'Brien 2009) boten eine Möglichkeit, Oberflächenbrüche in verschiedenen Materialien, einschließlich Glas, zu erzeugen. Wie bereits in der Einleitung erwähnt, bot (Pfaff et al. 2014) die Simulation des Glasbruchs als dünnes Blatt, was die am engsten verwandte Arbeit zu unserer vorgeschlagenen Methode darstellt.

Methodik

Um realistische Glasbrüche zu erzeugen, müssen groß angelegte physikbasierte Simulationen erstellt werden, indem die Bruchdynamik auf einem triangulierten Finite-Elemente-Netz mit Glaseigenschaften gelöst wird.

Simulation von gebrochenem Glas

Wir repräsentieren Glas durch Partikel, die aus einer gleichmäßigen Verteilung über eine Ebene entnommen werden, die in Form eines 2D-Netzes mit eingeschränkter Delaunay-Triangulation eingeschränkt ist. Dies entfernt schlecht geformte Dreiecke und vermeidet ungleichmäßige und unrealistische Kanten.

Jedes Partikel p_i hat eine Position x_i und hat nächste Nachbarn k_i innerhalb eines Radius r , die bestehende Kanten mit p_i haben. Mathematisch stellt das Triangulationsnetz \mathcal{M} eine endliche Menge von 2-Simplizes dar, so dass, wenn

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

Die Rissmuster im Glas entstehen durch die Spannung der äußeren Kraft (F) am anfänglichen Aufschlagpunkt p_1 , indem ein spezifisches Deformationsgesetz (Elastizität und Plastizität) des Glases (G) angenommen wird (Kuna 2013). Wir berechnen dann die Festigkeitsparameter in Form von effektiver Spannung σ_V am Aufschlagpunkt (V) als Spannungszustand des Aufschlagpunkts. Die kritischen Spannungswerte für die Festigkeit des Glases σ_C werden durch Tests an einfachen Proben mit elementaren Belastungsbedingungen (z.B. Zugversuch) ermittelt. Der Bruch tritt dann auf, wenn die effektive Spannung größer ist als die kritische Spannung, geteilt durch den Sicherheitsfaktor (S):

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

Aus der klassischen Festigkeitslehre wissen wir, dass das Versagen in den meisten Fällen durch die Hauptspannungen σ_I und σ_{II} für 2D-Elemente kontrolliert wird. Der anfängliche Riss entsteht entweder durch den normal-planaren Riss, bei dem die Bruchflächen senkrecht zur Richtung der höchsten Hauptspannung σ_I (Rankine 1857) liegen, oder durch den Scher-planaren Riss, bei dem die Bruchflächen mit den Schnittflächen der maximalen Scherspannung $\tau_{max} = (\sigma_I - \sigma_{II}) / 2$ (Coulumb 1776) zusammenfallen. Im Fall von Glas nehmen wir an, dass der anfängliche Bruch senkrecht zur Richtung der maximalen Hauptspannung erfolgt.

Vom anfänglichen Aufschlagpunkt p_1 aus ist die Spannungsverteilung durch das Glas instabil, da der Riss abrupt wächst, ohne dass eine Erhöhung der äußeren Belastung erforderlich ist. Ab p_1 ,

breitet sich die Spannung in der Nachbarschaft des Scheitelpunkts k_i aus, da die Spannung entlang der Richtung $\overrightarrow{p_i p_j}$ verläuft, wo $p_j \in k_i$ aus

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

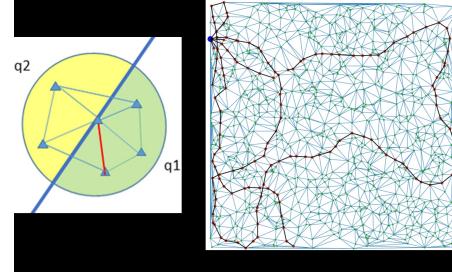


Abbildung 3: (a) Für eine in Blau gegebene Spaltebene werden die aufsummierte positive Spannung q_1 und die aufsummierte negative Spannung q_2 verglichen, und die Ausbreitung erfolgt auf der Seite mit der größeren aufsummierten Spannung und dem gewählten Knoten, der der Spaltebene am nächsten liegt (in Rot angegeben). (b) Zeigt, wie wir einen Bruch in einem Netz simulieren, der von seinem Aufschlagpunkt (in Blau markiert) zu den Knoten ausgeht, die Spannungen über ihrer Grenzfestigkeit erfahren (in Rot markiert).

Mit dem für jede Kante berechneten Stress kann der aufsummierte positive Stress (in Abb. 3 gezeigt) dann wie folgt angegeben werden:

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

für die kontinuierliche Oberfläche Ω , wobei die Indikatorfunktion ist. Der aufsummierte positive Stress für diskrete Simplizes im entsprechenden Bereich A mit Radius R wird wie folgt angegeben

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

Ähnlich wird der aufsummierte negative Stress q_2 berechnet. Dann wählen wir für die größere Größe $\max(|q_1|, |q_2|)$ die entsprechende Kante mit der höchsten Stresskonzentration im gegebenen Segment als die optimale Trennebene, da dies die maximale Stressentlastung bietet. Somit wandert der Stress entlang der Maschenkanten und dissipiert den Stress an jedem Knotenpunkt.

Die rekursive Anwendung der Stressausbreitung wird durchgeführt, bis die Konvergenz des Stresses in allen Zuständen erreicht ist, d.h. $\sigma_p^{(t)} \approx \sigma_p^{(t-1)} \forall p \in V$.

Die Ausbreitung des Stresses in alle Richtungen über alle Knoten führt zu Rückrisen, wie in (O'Brien und Hodgins 1999) erklärt. Um dies zu vermeiden, propagieren wir nur entlang der Kanten, wo die Stressniveaus maximal sind, führen jedoch ein Stress-Update an allen benachbarten Knoten durch. Wir verwenden dann einen minimalen Spannbaum (MST) auf einem Netz, das mit diesen gestressten Knoten erstellt wurde. Wir kombinieren diesen MST mit unserem anfänglichen Stressausbreitungsfeld entlang der Kanten, um das endgültige Rissmuster zu berechnen. Der MST ist eine effektive Abstraktion, da er die Knoten verbindet, die näher beieinander liegen und sich im Hochstressfeld befinden, während Redundanzen entfernt werden.

Unser Berechnungsprozess der Spannungsverteilung ist in Algorithmus 1 in der Ergänzung definiert.

Physikalisch basiertes Rendering

Sobald wir die Brüche auf der Mesh-Ebene erzeugt haben, besteht unser nächstes Ziel darin, eine visuelle Darstellung dieser Brüche zu erstellen. Wie alle PBR-Techniken basiert unsere Methode auf der Mikrofacet-Theorie, die besagt, dass jede Oberfläche durch winzige, perfekt reflektierende Spiegel, sogenannte Mikrofacetten, beschrieben werden kann (Pharr, Jakob, and Humphreys 2023).

In Übereinstimmung mit der Mikrofacet-Theorie und der Energieerhaltung verwenden wir die Reflexionsgleichung,

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

wobei $L_o(x, \omega_o, \lambda, t)$ die gesamte spektrale Strahldichte der Wellenlänge λ ist, die entlang der Richtung ω_o zum Zeitpunkt t von einer bestimmten Position x nach außen gerichtet ist. ω_o ist die Richtung des ausgehenden Lichts. t ist die Zeit. L_e ist die emittierte spektrale Strahldichte und L_r ist die reflektierte spektrale Strahldichte.

Sei I_1 die bidirektionale Reflexionsverteilungsfunktion,

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

und sei I_2 die spektrale Strahldichte, die von Richtung ω_i zum Zeitpunkt t nach innen zu x kommt.

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

Dann kann L_r definiert werden als

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

wobei Ω die Einheits-Halbkugel ist, die um die Oberflächennormalen \mathbf{n} über ω_i zentriert ist, sodass $\omega_i \cdot \mathbf{n} > 0$.

Abstrahieren der Reflexionsgleichung, zielen wir darauf ab, ein visuelles Rendering unseres gebrochenen Glasgitters zu erstellen. Wir haben $L_e = 0$, da Glas kein Licht emittiert. Nun, um L_r zu berechnen, betrachten wir jeden Riss zwischen den Knoten als eine Mikrofläche. Dann können wir L_r für jeden Riss definieren als:

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

Gegeben die Einheitsvektoren ($\hat{\omega}_\alpha$) und ($\hat{\omega}_\theta$), die den Azimut- (α) und Zenit- (θ) Winkel entsprechen, berechnen wir die mittlere Energie, die auf den Riss trifft, als

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

wobei \hat{n}_i die Einheits-Oberflächennormale des Risses ist.

Sei (I_r, I_g, I_b) die mittlere Intensität der Lichtquelle. Dann wird die Rissintensität, I_c , definiert als

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

Fokalebene und Physische Angriffssimulation Während wir in der Lage sind, realistische Brüche zu simulieren, ist der primäre Anwendungsfall unserer Arbeit, simulierte Beispiele zu erzeugen, die auf bestehenden Datensätzen (KITTI, BDD100k, MS-COCO) überlagert sind, und diese mit dem realen Straßendatensatz zu vergleichen, den wir erstellt haben.

Jedes aufgenommene Bild wird scharfe Merkmale der Objekte in seiner Fokalebene aufweisen. Das Glasgehäuse, das die Kamera abdeckt, ist extrem nah und gehört daher nicht zur Fokalebene.

Wenn der Riss entsteht, prallen die Lichtstrahlen ungleichmäßig entlang des Risses ab und erzeugen eine Unschärfe (Beispiel in Abb. 4). Wir erstellen eine Binärmaske basierend auf dem Rissmuster und verwischen dann die überlagerten Brüche auf dem Bild. Dies erzeugt ein Bild mit Fernfokus. Für ein Bild mit Nahfokus verwischen wir das Bild und konzentrieren uns auf den Vordergrund, d.h. den Riss.

Experimentieren

Datensatz

Wir bewerten zwei Arten von zerbrochenem Glas - real und simuliert - auf drei beliebten Open-Source-Datensätzen: KITTI (Geiger et al. 2013), BDD100k (Yu et al. 2020) und MS-COCO (Lin et al. 2014). Die ersten beiden repräsentieren spezifische Domänen des autonomen Fahrens, während der letzte ein allgemeiner Bilddatensatz ist. Die Bilder mit realen zerbrochenen Glasstrukturen werden von der FreePik-Website¹ gesammelt und stellen in unserem Fall die Basislinie dar. Wir haben insgesamt 65 Bilder gesammelt und sie durch Bildaugmentation mit zufälligen Verschiebungen, Bildspiegelungen und Zuschneide-techniken auf einen Satz von 10,000 Bildern erweitert. Wir erzeugen auch 10,000 Bilder mit unserem Physiksimulator. Anschließend überlagern wir diese zerbrochenen Glasstrukturen mit unserer PBR-Pipeline auf jedes Validierungsbild in den Datensätzen und sammeln die aggregierten Ergebnisse. Wir verwenden drei Modellarchitekturen: YOLOv8, Faster R-CNN und PVTv2-Modell mit vorge trainierten Gewichten, um Objekterkennungsergebnisse zu erzeugen.

Implementierung

Unser Simulationsmodell wird entwickelt, indem 10^4 Partikel zufällig aus einer gleichmäßigen räumlichen Verteilung im gegebenen Rahmen auf einer CPU entnommen werden. Ein KD-Baum aus dem SciPy-Python-Paket (Virtanen et al. 2020) mit Standardparametern wird erstellt, um die ungefähren nächsten Nachbarn jedes Partikels zu finden. Anschließend wird eine Delaunay-Triangulation auf den Partikeln durchgeführt, um ein eingeschränktes dreieckiges Netz zu erstellen. Wir verwenden eine Einschlagkraft von 500 Einheiten mit einem zufälligen Aufschlagpunkt und einem zufälligen Aufschlagvektor. Die Spannungsverbreitung erfolgt, bis ein Schwellenwert von 300 Einheiten erreicht ist. Das PBR wird auf der CPU durchgeführt, indem die im vorherigen Abschnitt beschriebenen Methoden unter Verwendung von OpenCV und Python implementiert werden.

Ergebnisse und Diskussion

Ein wesentlicher Unterschied zu den meisten früheren Arbeiten zu adversarialen Beispielen besteht darin, dass unsere generierten adversarialen Muster nicht alle Pixel in einem Bild universell beeinflussen. Daher muss der Vergleich nur für den Bildbereich durchgeführt werden, in dem das Muster existiert. Zu diesem Zweck erstellen wir eine binäre Maske jedes Musters und geben die Ergebnisse der Objekte aus, die nur in diesem Muster existieren.

Tabelle 1 zeigt die Ergebnisse der durchschnittlichen Präzision (AP) unter den adversarialen Bildern, die mit den beiden Arten von Rissmustern (online gesammelt und simuliert) für verschiedene Klassen erzeugt wurden. Für KITTI sinkt die AP der anderen Klassen erwartungsgemäß mit dem Rückgang der AP, der dem Prozentsatz der vom Lkw besetzten Bildfläche entspricht, wobei der Lkw die größte Abnahme verzeichnet. Für BDD100K mit PVTv2-B0 sehen wir, dass der Rückgang der AP bei den simulierten Bildern am größten ist, aber insgesamt,

¹<https://www.freepik.com>

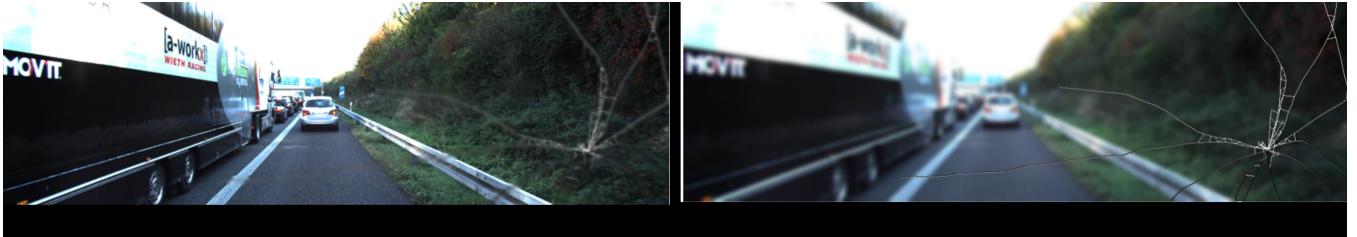


Abbildung 4: (a) Zeigt das simulierte Bild mit der Straße und den Fahrzeugen in der Fokalebene (PBR und Fernfokus). (b) Bezeichnet das simulierte Rissmuster in der Fokalebene (PBR und Nahfokus).

Tabelle 1: Durchschnittliche Präzision (in Prozent) verschiedener Klassen in KITTI, BDD100k und MS-COCO unter verschiedenen adversarialen Bildern. x bietet die Überlagerungsbeziehung zwischen Datensatz und Glasriss-Typ. Sauber x Datensatz - bezieht sich direkt auf die speziellen Bilder ohne adversariale Probe. RO x Datensatz - bezieht sich auf echte Bilder von online gesammeltem, rissigem Glas, das auf saubere Bilder überlagert wurde. Sim x Datensatz - bezieht sich auf simulierte Rissmuster, die auf saubere Bilder überlagert wurden.

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

Der Trend wird beibehalten, wobei die Fußgängerklasse den stärksten Rückgang zeigt. Für MS-COCO haben wir den AP für die Superkategorien aggregiert: Person, Fahrzeuge und Essen. Dies liegt daran, dass viele Objekte in MS-COCO eine kleinere Fläche im Bildrahmen einnehmen, was es schwierig macht, aussagekräftige Ergebnisse aus allen Kategorien zu erhalten. Ein sehr interessantes Ergebnis ist, dass die Fußgängerklasse unter den realen gebrochenen Glasmustern einen mehrfachen Anstieg des AP aufweist. Während dieser Trend kontraintuitiv erscheinen mag, stimmt er mit den Ergebnissen in Abb. 2 überein, wo das Vertrauen des Autos aufgrund einer Kante steigt. Dies zeigt tatsächlich, dass der AP stark vom Rissmuster abhängt, was es äußerst wichtig macht, Verteidigungsmethoden zu entwickeln, um diese adversarialen Angriffe zu mildern.

Ablationsstudien

Unsere Ergebnisse zeigen, dass die simulierten Bilder einen ähnlichen adversarialen Effekt wie die realen Bilder erzielen. Daher ist eine wichtige Ablationsstudie für uns zu verstehen, wie nah die simulierten Rissmuster an den realen Rissmustern und den online gesammelten Mustern sind. Wir bilden 5 Verteilungen

- Realer On-Road-Datensatz (dargestellt in Abb. 2)

- Rissmuster, die online gesammelt wurden (Abb. 5 oben links)
- Simulierte Rissmuster (Abb. 5 unten links)
- Simulierte Rissmuster überlagert auf KITTI (Abb. 5 unten rechts)
- Rissmuster, die online gesammelt wurden, überlagert auf KITTI (Abb. 5 oben rechts)

Wir berechnen nun die K-L-Divergenz zwischen all diesen Verteilungen, um zu ermitteln, wie ähnlich sie einander sind (siehe Abb. 6). Um eine Kontrolle zu bieten, vergleichen wir KITTI mit Bildern von Katzen aus dem Kaggle-Datensatz, was eine K-L-Divergenz von 2,434 ergibt. In dieser Skala haben die PBR-Bilder von zerbrochenem Glas einen Unterschied von 0,36 zu den echten zerbrochenen Glasmustern, während die zerbrochenen Glasfilter, die auf KITTI-Bilder überlagert sind, eine ähnliche K-L-Divergenz aufweisen.

Abb. 7 zeigt eine Analyse der Rechenzeit für jedes unserer Module und über verschiedene Anzahlen von Partikeln. Wir führen diese Analyse über 100 Durchläufe durch, wobei wir zufällige Aufschlagpunkte, Aufschlagwinkel und Netzstrukturen mit einer festen Anzahl von Partikeln generieren. Der Unterschied in der Rechenzeit für verschiedene Durchläufe kann dem Aufschlagpunkt und dem Aufschlagwinkel zugeschrieben werden.

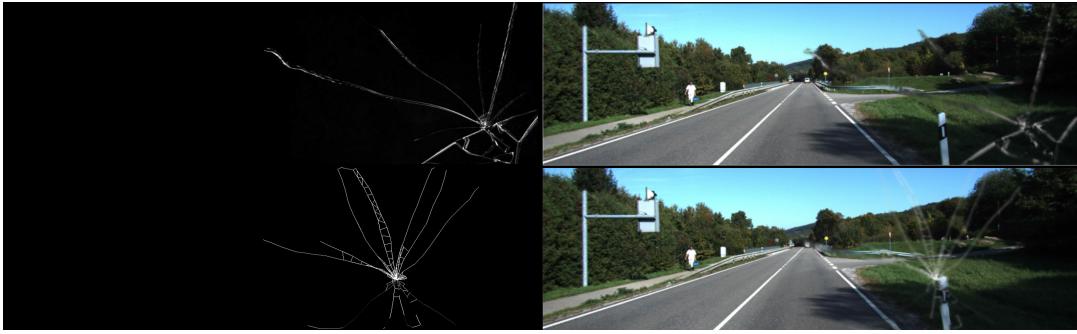


Abbildung 5: Oben links - Rissmuster online auf Freepik gesammelt; oben rechts - online Rissmuster überlagert auf KITTI; unten links - simuliertes Rissmuster mit PBR; unten rechts - simuliertes Rissmuster überlagert auf KITTI.

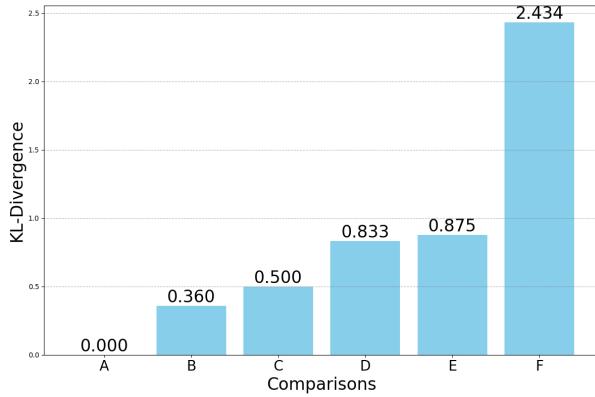


Abbildung 6: K-L-Divergenz verschiedener Paare von Bildverteilungen. Datensätze: RC - Reales On-Road-Datensatz (siehe Abb. 2), KITTI und Katzen. Filter: RO - Real (online gesammelt) und Sim - Simuliert. K-L-Divergenz zwischen (x - Überlagerungsbeziehung): A - (Sim x KITTI) vs (Sim x KITTI); B - (Sim vs RO); C - (Sauberes RC vs KITTI); D - (Beschädigtes RC) vs (RO x KITTI); E - (Beschädigtes RC) vs (Sim x KITTI); F - KITTI vs Katzen.

Der Rissvisualisierung und die Renderzeit variieren ebenfalls aufgrund unterschiedlich großer Masken, die durch verschiedene Bruchmuster entstehen. Wir variieren auch die Anzahl der Partikel und beobachten, wie die Laufzeit exponentiell mit der Zunahme der Partikel steigt. Alle diese Durchläufe wurden auf Bildern aus dem KITTI-Datensatz mit den Abmessungen von $(375 \times 1242 \times 3)$ gerendert.

Fazit und zukünftiger Umfang

Wir haben eine neuartige Klasse von adversarialen Fehlern eingeführt, die aus dem physikalischen Prozess von Fehlern in der Kamera resultieren. In diesem Papier stellen wir einen Ansatz vor, um ein realistisches zerbrochenes Glas-Muster aus einer physikalischen Simulation zu erzeugen und dieses anschließend in bestehende Bilddatensätze mittels physikalisch basierter Darstellung einzubetten. Wir zeigen, dass die simulierten adversarialen Bilder zu erheblichen Fehlern in der Objekterkennung führen können.

In dieser Arbeit befassen wir uns mit Black-Box-Angriffen, die aus realen, natürlich vorkommenden physikalischen Phänomenen resultieren, die nicht künstlich entwickelt wurden, um spezifische Modellschwachstellen auszunutzen.

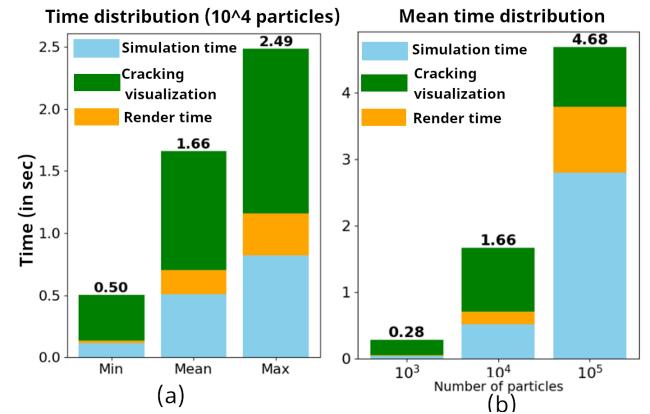


Abbildung 7: (a) Durchschnittliche Zeit, die von verschiedenen Modulen unserer Pipeline über 100 Durchläufe benötigt wird. (b) Die minimale, maximale und durchschnittliche Zeit, die von verschiedenen Modulen über 100 Durchläufe für ein 10^4 Partikelgitter benötigt wird. In diesen Diagrammen zeigen wir die für die Simulation benötigte Zeit (Simulationszeit), die Umwandlung des Gitters in Glas (Rissvisualisierung) und schließlich das Rendering (Renderzeit).

Wir gehen davon aus, dass keine Kenntnisse über die Modellattribute, Gewichte oder Architektur vorliegen, wodurch Angriffe auf verschiedene Modelle übertragbar sind. Physische adversarielle Methoden (Translucent Patch, RP2) können alle als Verdeckungen entweder der Kamera oder der erfassten Objekte bezeichnet werden. Die Adversarialität ergibt sich aus der Wirkung der Modellinferenz aufgrund dieser Verdeckungen. Unsere PBR-Pipeline mischt die Risse mit den Quellbildern als durchscheinende, verschwommene Muster, die den latenten Raumcodierung beeinflussen, anstatt eine direkte Verdeckung zu verursachen, was zu falschen Erkennungen führt.

Während diese Arbeit speziell eine physikbasierte Methode zur Erzeugung von zerbrochenen Glasmustern einführt, umfassen Kameraausfälle auch andere Effekte wie Sonnenblendung, Überbelichtung, Unterbelichtung, Kondensation usw. Unsere zukünftige Arbeit wird sich darauf konzentrieren, ein adversariales Toolkit für die realistische Erzeugung dieser Effekte mithilfe von Physik zu erstellen und sie anschließend auf bestehende Bilddatensätze und Auto-Simulationsplattformen zu platzieren, um die weitere Forschung in diesem Bereich der partiellen Kameraausfälle zu fördern.

Referenzen

- Akhtar, N.; und Mian, A. 2018. Bedrohung durch adversariale Angriffe auf Deep Learning in der Computer Vision: Eine Umfrage. *Ieee Access*, 6: 14410–14430.
- Carlini, N.; und Wagner, D. 2017. Adversarielle Beispiele sind nicht leicht zu erkennen: Umgehung von zehn Erkennungsmethoden. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 3–14.
- Ceccarelli, A.; und Secci, F. 2022. Ausfälle von RGB-Kameras und ihre Auswirkungen auf Anwendungen im autonomen Fahren. *IEEETransactions on Dependable and SecureComputing*. Coulumb, C.-A. 1776. Essai sur une application des règles des maximis et minimis à quelques problèmes de statique relatifs, à l'architecture. *Mem. Acad. Roy. Div. Sav.*, 7: 343–387. Dalvi, N.; Domingos, P.; Mausam; Sanghi, S.; und Verma, D. 2004. Adversarielle Klassifikation. 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.; Kohn, T.; und Song, D. 2018. Robuste physische Angriffe auf die visuelle Klassifikation durch Deep Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1625–1634. Geiger, A.; Lenz, P.; Stiller, C.; und Urtasun, R. 2013. Vision trifft Robotik: Der KITTI-Datensatz. *International Journal of Robotics Research(IJRR)*. Goodfellow, I. J.; Shlens, J.; und Szegedy, C. 2014. Erklärung und Nutzung adversarieller Beispiele. *arXiv preprint arXiv:1412.6572*. Iben, H. N.; und O'Brien, J. F. 2009. Erzeugung von Oberflächenrissmustern. *GraphicalModels*, 71(6): 198–208. Jocher, G.; Chaurasia, A.; und Qiu, J. 2023. Ultralytics YOLO. Kong, Z.; Guo, J.; Li, A.; und Liu, C. 2020. Physgan: Erzeugung physisch-weltresistenter adversarieller Beispiele für autonomes Fahren. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14254–14263. Kuna, M. 2013. Finite Elemente in der Bruchmechanik. *Solid Mechanics and Its Applications*, 201: 153–192. Kurakin, A.; Goodfellow, I. J.; und Bengio, S. 2018. Adversarielle Beispiele in der physischen Welt. In *Artificial Intelligence Safety and Security*, 99–112. Chapman and Hall/CRC. Li, J.; Schmidt, F.; und Kolter, Z. 2019. Adversarielle Kamerasticker: Ein physischer, kamerabasierter Angriff auf Deep-Learning-Systeme. In *International Conference on Machine Learning*, 3896–3904. PMLR. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; und 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.; und Xue, C. 2021. Analyse des Linsenbruchs in der Präzisionsglasformung mit der Finite-Elemente-Methode. *Applied Optics*, 60(26): 8022–8030. Nguyen, A.; Yosinski, J.; und Clune, J. 2015. Tiefe neuronale Netzwerke lassen sich leicht täuschen: Hochsichere Vorhersagen für nicht erkennbare Bilder. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 427–436. O'Brien, J. F.; und Hodgins, J. K. 1999. Grafische Modellierung und Animation von sprödem Bruch. In *Proceedings of ACM SIGGRAPH 1999*, 137–146. ACM Press/Addison-Wesley Publishing Co. Pfaff, T.; Narain, R.; De Joya, J. M.; und O'Brien, J. F. 2014. Adaptives Reißen und Brechen dünner Blätter. *ACM Transactions on Graphics (TOG)*, 33(4): 1–9. Pharr, M.; Jakob, W.; und Humphreys, G. 2023. *Physically based rendering: From theory to implementation*. MIT Press.
- PK, A. ??? Kaggle Katzen und Hunde Mini-Datensatz.
h
t
t
p
s://
w
www.kaggle.com/datasets/aleemaparakatta/cats-and-dogs-mini-dataset. Zugriff: 2024-09-30. Rankine, W. J. M. 1857. II. Über die Stabilität von lockerem Erdreich. *Philosophical transactions of the Royal Society of London*, (147): 9–27. Ren, S.; He, K.; Girshick, R.; und Sun, J. 2016. Faster R-CNN: Auf dem Weg zur Echtzeit-Objekterkennung mit Region-Vorschlagsnetzwerken. *IEEETransactions on pattern analysis and machine intelligence*, 39(6): 1137–1149. Rouxel, T.; und Brow, R. K. 2012. Der Fluss und Bruch von fortschrittlichen Gläsern—ein Überblick. *International Journal of Applied Glass Science*, 3(1): 1–2. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; und Fergus, R. 2013. Interessante Eigenschaften von neuronalen Netzwerken. *arXiv preprint arXiv:1312.6199*. Tu, J.; Ren, M.; Manivasagam, S.; Liang, M.; Yang, B.; Du, R.; Cheng, F.; und Urtasun, R. 2020. Physisch realisierbare adversarielle Beispiele für Lidar-Objekterkennung. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 13716–13725. van Schrick, D. 1997. Bemerkungen zur Terminologie im Bereich der Überwachung, Fehlererkennung und 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.; und SciPy 1.0 Contributors. 2020. SciPy 1.0: Grundlegende Algorithmen für wissenschaftliches Rechnen in Python. *Nature Methods*, 17: 261–272. Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; und Shao, L. 2022. Pvt v2: Verbesserte Baselines mit Pyramidal Vision Transformer. *Computational visualmedia*, 8(3): 415–424.

- Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V.; und Darrell, T. 2020. BDD100k: Ein vielfältiger Fahrdatensatz für heterogenes Multitask-Lernen. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2636–2645.
- Zolfi, A.; Kravchik, M.; Elovici, Y.; und Shabtai, A. 2021. Der Translucent Patch: Ein physischer und universeller Angriff auf Objektdetektoren. In *Proceedings of the IEEE/CVF conference on computervision andpattern recognition*, 15232–15241.

Algorithmus der Spannungsverbreitung

Algorithmus 1 beschreibt das Verfahren zur Simulation der Verbreitung von Spannung durch ein Material nach einem Einschlagereignis. Der Algorithmus nimmt als Eingaben den Ort des Aufschlags (pt), die Größe der Einschlagkraft (F), den Aufschlagsrichtungsvektor (v) und die übergeordnete Kante (PE), die mit der Aufschlagstelle verbunden ist. Er verwendet auch einen Nächster-Nachbar-Radius R , um die Menge der Kandidatenorte für die Spannungsverbreitung zu bestimmen.

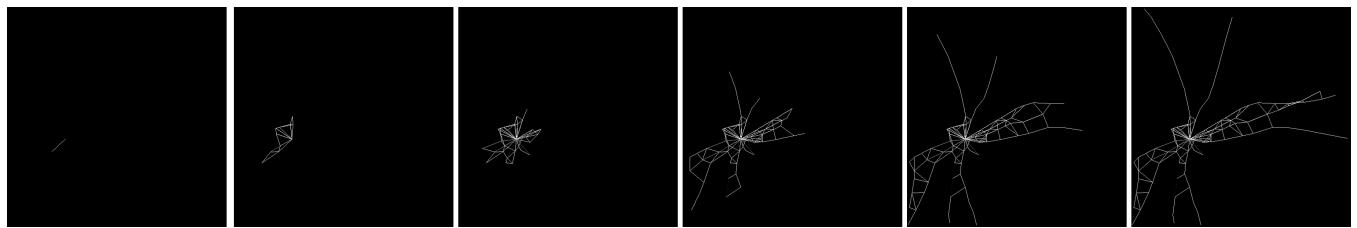
Algorithmus 1: Spannungsverbreitung

```
1:  $pt \leftarrow$  Aufschlagpunkt
2:  $F \leftarrow$  Einschlagkraft
3:  $PE \leftarrow$  Übergeordnete Kante
4:  $v \leftarrow$  Aufschlagsvektor
5:  $R \leftarrow$  Nächster-Nachbar-Radius
6:
7: Prozedur STRESSVERBREITUNG( $Pt, F, V, PE$ )
8:    $frontiers \leftarrow KDT\ree - queryRadius(R)$ 
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:  PROPAGIERESTRESS( $Pt, F, V, PE$ )
16: Prozedur beenden
```

Zuerst verwendet es eine KD-Baum-Datenstruktur, um effizient alle Punkte (Grenzflächen) innerhalb eines gegebenen Radius R vom Aufschlagpunkt abzufragen. Für jede Grenzfläche berechnet es einen Einheitsrichtungsvektor vom Aufschlagpunkt zur Grenzfläche (NN). Dann projiziert es den Aufschlagsvektor v auf diese Richtung, um die Kosinusähnlichkeit $\cos(\theta)$ zu erhalten, die die Winkelbeziehung zwischen der Aufschlagsrichtung und der Kandidaten-Ausbreitungsrichtung erfasst. Für jeden Kandidaten wird der resultierende Wert zusammen mit der Einschlagkraft verwendet, um die entsprechende Spannung an diesem Punkt zu berechnen. Der Algorithmus wählt dann den Kandidaten mit dem maximalen Spannungswert aus. Der Aufschlagsvektor v und die übergeordnete Kante PE werden aktualisiert, um dieser neuen Richtung zu entsprechen. Der Prozess wird rekursiv wiederholt, sodass die simulierte Spannungswelle iterativ durch das Material entlang des Pfades der größten Spannungsübertragung propagiert.

Dieser Ansatz zielt darauf ab, nachzuahmen, wie sich die Spannung von einem Aufschlagpunkt am wahrscheinlichsten durch ein Material ausbreitet—bevorzugt entlang von Pfaden, die sowohl durch geometrische Nähe als auch durch mechanische Ausrichtung mit dem ursprünglichen Aufschlag definiert sind.

Das endgültige Ergebnis der Sim ist die Darstellung des Netzes als Bild, das dem Muster einer zerbrochenen Linse entspricht (Endbild von Abb. 8).



Figu^{Abb. 8}: Eine Animation des Bruchs einer Linse, simuliert durch das Setzen des Spannungsfeldes und die Anwendung von PB R.

Statisches Experiment

Um die Auswirkungen dieser Brüche auf die resultierenden Bilder zu verstehen, führen wir zunächst ein statisches Experiment in Innenräumen durch, wie im Abschnitt Einführung erwähnt. Für dieses Experiment verwenden wir verschiedene gehärtete Glasscheiben, die wir zufällig mit einem kleinen Hammer an einem oder mehreren Bruchpunkten zerbrechen. Dann platziieren wir eine 36 MP JVC GC-PX10 Hybridkamera, die mit einer Klemme auf einem Stativ vor dem gehärteten Glas montiert ist.

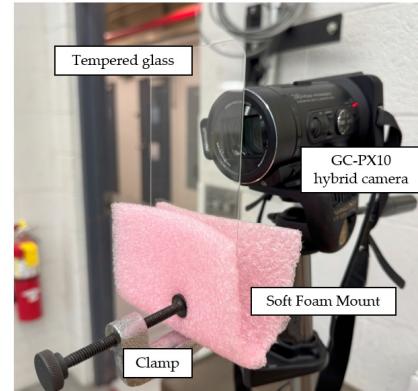
Abb. 9(a) zeigt den detaillierten Aufbau mit der Kamerahalterung und dem gehärteten Glas, das mit einer Klemme fixiert ist. Abb. 9(b) zeigt das von der Kamera aufgenommene Bild und Abb. 9(c) zeigt das einzelne Fahrzeug, das als Hauptobjekt durch das gehärtete Glas von der Kamera erfasst wird. Die Szene wird mit Deckenleuchtstofflampen beleuchtet.

Abb. 10 zeigt einige der Bruch-/Kratzer-Muster auf dem gehärteten Glas. Diese Muster wurden absichtlich randomisiert, indem mehrere Brennpunkte und unterschiedliche Kraftstärken verwendet wurden, um die unvorhersehbare und vielfältige Natur von Glasschäden in der realen Welt nachzuahmen. Durch die Anwendung unterschiedlicher Kraftstärken konnten wir ein Spektrum von Brüchen und Kratzern erzeugen, das von feinen Oberflächenabnutzungen bis zu ausgeprägteren Brüchen reicht. Dieser Ansatz wurde gewählt, um die Arten von Schäden, die Glasoberflächen unter realen Bedingungen erleiden können—wie durch Stöße, Trümmer oder Umwelteinflüsse verursacht—möglichst genau zu replizieren und so die Relevanz und Realitätsnähe unseres experimentellen Aufbaus sicherzustellen. Diese repräsentativen Schadensmuster ermöglichen es uns, den Einfluss von Glasunvollkommenheiten auf die Sensorleistung und Objekterkennungsalgorithmen effektiver zu analysieren.

Zwei verschiedene Bruchmuster und ihre resultierenden Bilder sind in Abb. 11 und Abb. 12 dargestellt. Wir möchten darauf hinweisen, dass wir die Brennweiten der Kamera erheblich variiert haben, um zu verstehen, wie die Bilder bei Nah- und Fernfokus aussehen. Die Ergebnisse zeigen, dass selbst geringfügige Kratzmuster im Bildausgang sichtbar werden, während ein viel stärkeres Mehrfachbruchmuster fast das gesamte Bild verwischen kann. Dieses Experiment liefert die Intuition, auf der unser Simulations- und Visualisierungsrahmen basiert.

Erhöhte AP für Fußgänger in KITTI

Wir möchten darauf hinweisen, dass die erhöhte AP für die Fußgängerklasse etwas war, das selbst uns zunächst überrascht hat. Eine sorgfältige qualitative Tiefenanalyse half uns jedoch zu verstehen, dass dies aufgrund der Gläserne geschah, die es dem Modell erleichterten, Fußgänger zu klassifizieren, da die Kanten um sie herum verstärkt wurden. Dies war kein Kantenartefakt, sondern vielmehr der Gläserne, der als zusätzliche Kantenbegrenzung fungierte und den Fußgänger klar vom Hintergrund trennte. Ein ähnliches Ergebnis wurde auch in [1] beobachtet, wo die gesamte AP in adversarialen Bildern erhöht wurde.



(a)



(b)



(c)

Abbildung 9: Experimenteller Aufbau zur Erfassung von Bildern, die durch zerkratzte/gebrochene Außenschichten einer Kamera beeinträchtigt sind. (a) zeigt den gesamten Aufbau zur Aufnahme von adversarialen Bildern. (b) zeigt die Position der Kamera in Bezug auf die aufgenommene Szene. (c) zeigt die Szene, die von der Kamera erfasst wird.



Abbildung 10: Einige Bruch-/Kratzer-Muster auf dem Glas, das wir zur Bildersammlung verwendet haben. (a) Eine scharfe Kraft, die senkrecht zur Glasoberfläche angewendet wird, erzeugt radial verlaufende Brüche. (b) und (c) replizieren ein Glas mit Kratzern.

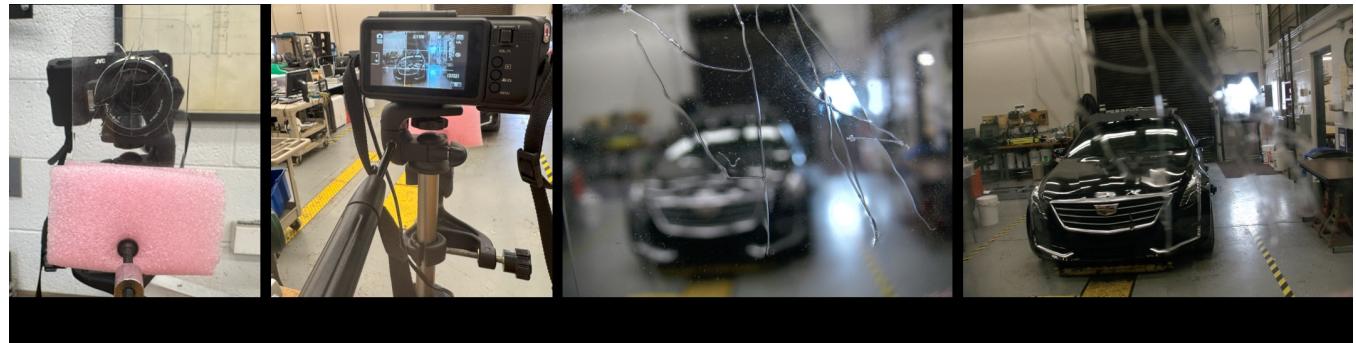


Abbildung 11: (a) Zeigt das Kratzmuster vor der Kamera, (b) zeigt die Kamera-Perspektive. (c) zeigt das von der Kamera aufgenommene Bild (Kurzfokus). (d) zeigt das von der Kamera aufgenommene Bild (Fernfokus).

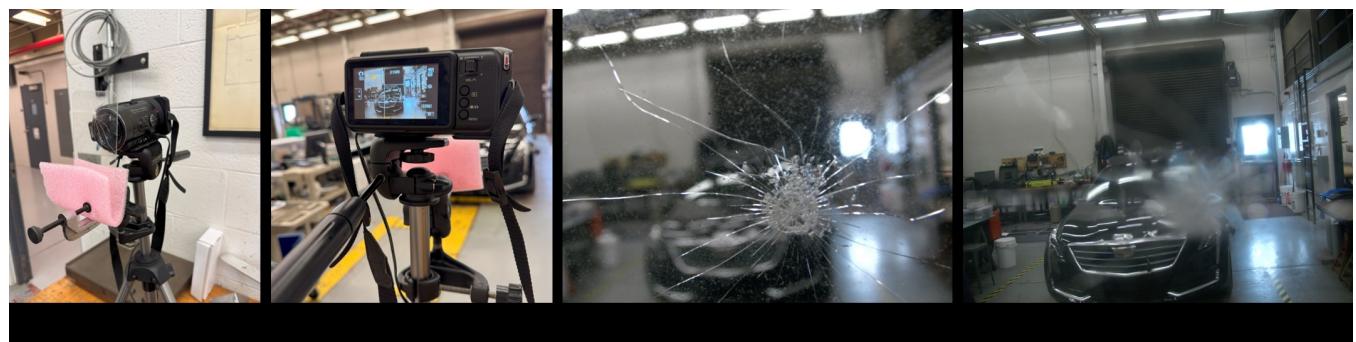


Abbildung 12: (a) Zeigt das zerbrochene Glas-Muster vor der Kamera, (b) zeigt die Kamera-Perspektive. (c) zeigt das von der Kamera aufgenommene Bild (Kurzfokus). (d) zeigt das von der Kamera aufgenommene Bild (Fernfokus).

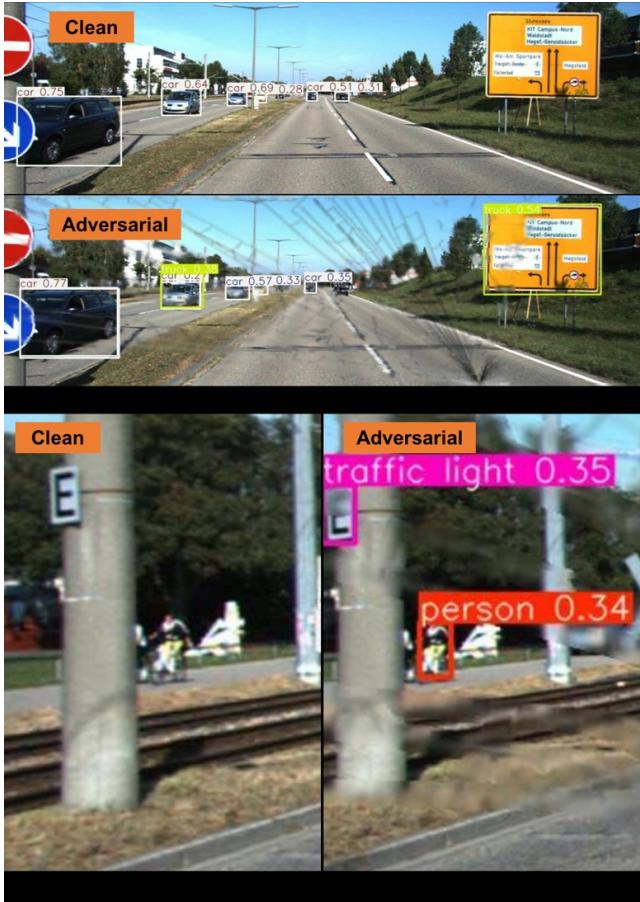


Abbildung 13: (a) Oben - Erkennungen auf einem sauberen Bild; unten - Erkennungen auf einem adversarialen Bild. (b) YoLo erkennt die Person nicht (c) Glasrisse ermöglichen es dem Modell, die Person zu erkennen.

Dynamisches Experiment

In diesem Abschnitt beschreiben wir das dynamische Experiment, das im Abschnitt Einführung erwähnt wird. Wir führen dieses Experiment durch, um die zeitliche Störung zu verstehen, die durch einen Riss eingeführt wird. Wir verwenden einen Windschutzscheibenriss eines Fahrzeugs und platzieren eine kleine Kamera auf dem Armaturenbrett hinter dem Riss. Dann fotografieren wir zwei dynamische Objekte - ein Fahrzeug und einen Fußgänger, während sie sich durch die Szene bewegen. Abb. 14 zeigt einige spezifische Bildrahmen mit Schlussfolgerungen von YOLOv8 für die Fahrzeugklasse. Wir zeigen, dass das Fahrzeug mit dem Riss in den meisten Rahmen unentdeckt bleibt. Zusätzlich enthält fast jeder Rahmen ein falsch positives Ergebnis. Entsprechend präsentieren wir Abb. 15 als die Rahmen mit einer Person, die in der Szene geht. Wir zeigen, dass es intermittierend Erkennungen liefert und gelegentlich mit einer falschen Klasse (Surfbrett).



Abbildung 14: Bestimmte Bilderrahmen der mit dem Windschutzscheibenriss aufgenommenen Bilder mit YOLOv8-Inferenz für die Fahrzeugklasse. A - Falsch positiv ohne Objekt in der Szene; B - keine Inferenz auf Fahrzeug; C - keine Inferenz auf Fahrzeug; D - erste Erkennung auf Fahrzeug; E - zwei verschiedene Erkennungen auf demselben Fahrzeug; F - falscher Bereich des Begrenzungsrahmens.



Abbildung 15: Spezifische Bilderrahmen der mit dem Riss in der Windschutzscheibe aufgenommenen Bilder mit YOLOv8-Inferenz für die Klasse Person. A - erster Eintritt der Person in die Szene ohne Erkennung; B - keine Inferenz der Person; C - erste Erkennung der Person; D - teilweise Erkennung der Person; E - Erkennung der Person mit anderer Klasse; F - vollständige Erkennung der Person.

Echte Glasbruchbilder

Wir präsentieren ein Beispiel der Glasbruchbilder, die von der FreePik-Website gesammelt und auf den KITTI-Datensatz überlagert wurden, zusammen mit der YOLOv8-Inferenz (Abb. 16). Wir zeigen, dass der Bruch einige Erkennungen entfernt und das Erkennungsvertrauen anderer verringert.

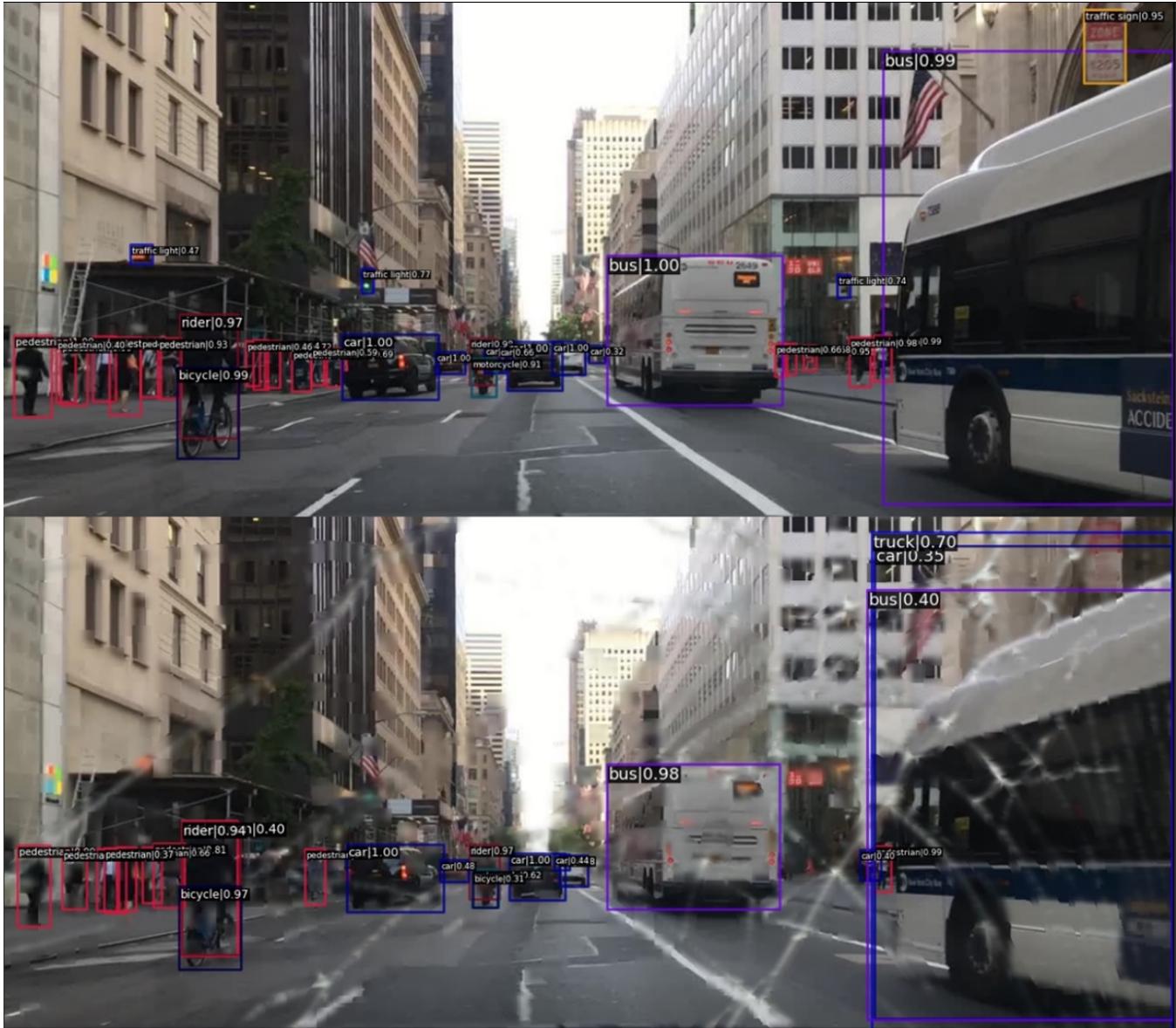


Abbildung 16: Oben - Inferenz von PTV2 auf einem sauberen Bild aus BDD100k. Unten - Inferenz für ein echtes Bild mit zerbrochenem Glas, das zur Vergleichszwecken auf BDD100k überlagert wurde. Wir sehen zwei zusätzliche Fehlalarme auf der rechten Seite (Lkw, Auto) und mehrere Fehlklassifikationen für die Fußgängerklasse auf der linken Seite des adversarialen Bildes.