UNIVERSITY OF CALIFORNIA,
IRVINE

Novel Vulnerability Discoveries, Measurements, and Attack Designs for Safety-Critical Autonomous Systems from Practicality Perspectives

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Ningfei Wang

2024

DEDICATION

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Page

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Qi Alfred Chen, for his steadfast support and invaluable guidance throughout my Ph.D. journey. His enduring patience, unwavering motivation, and profound empathy have significantly influenced me as a researcher over these past five years. Recalling on our initial chat in 2018, I was deeply inspired by his visionary security research of autonomous systems and it was this moment that attract me to join his research group. Throughout my Ph.D. studies, Dr. Chen skillfully introduced me to the foundational aspects of security research, fostering my research mindsets, and encouraging a comprehensive understanding of complex problems within the field. Moreover, his guidance was not limited to academics alone. He provided personal mentorship that taught me the art of strategic planning, thoughtful decision-making, courageous confrontation of challenges, and continual self-improvement. As I look back over these years, my appreciation deepens for his willingness to mentor a nascent Ph.D. student like me and I am immensely fortunate to have had Dr. Chen as my advisor. His exceptional guidance has been pivotal to my journey, and it is with sincere gratitude that I acknowledge his profound impact on my personal and professional growth.

Next, I would like to extend my deepest appreciation to Dr. Gene Tsudik, Dr. Marco Levorato, Dr. Sangeetha Abdu Jyothi, and Dr. Zhou Li for serving on my advancement to candidacy committee; and Dr. Mohammad Al Faruque and Dr. Habiba Farrukh for serving on my Ph.D. dissertation committee. Their insightful feedback and suggestions were essential in improving my dissertation and broadening my perspective on the research field. I also wish to thank my intern manager and mentor: Yupin Huang and Dr. Han Cheng at Amazon A9 and Dr. Leihao Wei at Meta for their valuable guidance.

My doctoral journey has been joyful and fruitful, huge thanks to all my collaborators and friends. Foremost among them, I wish to express my gratitude to my labmates: Dr. Junjie Shen, Dr. Takami Sato, Ziwen Wan, Shaoyuan Xie, Yunpeng Luo, Fayzah Alshammari, Trishna Chakraborty, Junchi Lu, Chi Zhang, Li-Chen Cheng. We shared cherishable memories in research, life, hobbies, etc. I am also thankful to Dr. Ting Wang, Dr. Yinzhi Cao, Dr. Mooi Choo Chuah, Dr. Kaidi Xu, Dr. Chaowei Xiao, Dr. Yulong Cao, Dr. Bo Li, Dr. Dawei Yang, and Dr. Mingyan Liu for introducing me to and guiding me in security and machine learning research. I feel incredibly fortunate to have worked and to continue working with great collaborators, such as Dr. Shujiang Wu, Chen Ma, Dr. Chao Shen, Dr. Zhisheng Hu, Dr. Kang Li, Dr. Ziming Zhao, Dr. Chunming Qiao, Dr. Yueqiang Cheng, Dr. Shouling Ji, Dr. Xinyang Zhang, Dr. Hua Shen, Dr. Xiapu Luo, Dr. Yunhan Jack Jia, Dr. Xue Lin, Dr. Qi Zhu, Dr. Xiaojie Wang, Vivek Mittal, Justin Yue, Nanze Chen, Rong Mu, Christopher Joseph Dipalma, Zeyuan Chen, Han Wang, Chen Wang, Kanglan Tang, Dr. Yiheng Feng, Xiangguo Liu, Anthony Goeckner, Ruochen Jiao, Yixuan Wang, Dr. Jin Fang, Dr. Ruigang Yang, Dr. Shengjian Guo, Dr. Zhenyu Zhong, Pallavi Garg, Nitesh Gupta, Shubham Bhanudas Abhale, Aditya Sanjay Dikshit, Yiwen Zhu, Chaoran Yuan, Zhiqi Xu.

Furthermore, I wish to express my profound gratitude to my friends enriching my life over the past five years. My thanks go to Boran Wen, Dr. Dan Luo, Junze Liu, Shutong Li, Dr.

Andrew Searles, Xuesong Bai, Qifan Zhang, Weiheng Bai, Dr. Anthony Tsai, Chang Song, Shanshan Han, Zhanhang Liang, Dr. Liang Luo, Dr. Zhen Zhang, Dr. Shiqin He, Qingyuan Zheng, Jiacen Xu, Zhongqi Yang, Yiyang Min, Hanning Chen, Ziyu Wang, Dr. Haoyu Ma, Dr. Xiangyi Yan, Dr. Jiri Gesi, Dr. Aodong Li, Dr. Yunhan Zhao, Dr. Xi Lu, Dr. Zhe Wang, Jiashu Fan, Zifan Zhang, Xinrong Chi, Yingjie Chen, Hao Chen, Lu Lu, along with many others.

My Ph.D. research was supported in part by NSF under grants CNS-1850533, CNS-1929771, CNS-1932351, CNS-1932464, CNS-2145493; USDOT under grant 69A3552047138 for CARMEN UTC; USDOT under Grant 69A3552348327 for the CARMEN+ University Transportation Center; Donald Bren School of Information and Computer Sciences and Department of Electrical Engineering and Computer Science at University of California, Irvine. Thanks to all of them for their kind support.

Additionally, I want to thank my girlfriend, Tong Wu, who has been my unfaltering pillar of support and belief. Without her unconditional love and support, I do not think I can even have the courage to take on the challenge of pursuing a Ph.D. degree. Having her alongside me is the best thing that ever happened to me in my whole life. As we continue our journey together, my hope is that we continue to stand by each other, strengthening our bond as a family and jointly navigating the seas of life.

Finally, my sincere gratitude goes to my loving and supportive parents, Hongliang Wang and Jianyun Qi, and all other family member. It has been a long journey for me to pursue a Ph.D. degree in a distant land without seeing them in person. Their video/voice calls and unconditional support have given me the courage and determination to reach heights I had never imagined. They have been a steady source of strength for me during the difficult two years of the COVID-19 pandemic. Words fall short in expressing my appreciation for their lifelong support. The regret for not having been by their side over the past five years weighs heavily on me. As I continue my journey, their love and sacrifices will continue to motivate me to be the best I can be. I will always be in their debt.

VITA

Ningfei Wang

EDUCATION

Doctor of Philosophy in Computer Science **2019 – 2024**
University of California, Irvine *Irvine, CA*

Master of Science in Computer Science **2017 – 2019**
Lehigh University *Bethlehem, PA*

Bachelor of Engineering in Information Engineering **2013 – 2017**
Beijing University of Posts and Telecommunications *Beijing, China*

RESEARCH EXPERIENCE

Graduate Research Assistant **2019 – 2024**
University of California, Irvine *Irvine, CA*

Meta (Intern) **2024**
BCR Ranking Core ML Team *Menlo Park, CA*

Amazon (Intern) **2023**
Search Relevance Team *Palo Alto, CA*

Research Assistant **2018–2019**
APLS lab at Lehigh University *Bethlehem, PA*

Research Assistant **2018**
SEC lab at Lehigh University *Bethlehem, PA*

Research Assistant **2017**
WiNS lab at Lehigh University *Bethlehem, PA*

Research Assistant **2016**
Department of Automation at Tsinghua University *Beijing, China*

TEACHING EXPERIENCE

Teaching Assistant **2022 & 2023**
University of California, Irvine *Irvine, CA*

SELECTED PUBLICATIONS

Revisiting Physical-World Adversarial Attack on Traffic Sign Recognition: A Commercial Systems Perspective **2025**
The Network and Distributed System Security Symposium 2025 (NDSS 2025)

Towards Robustness Analysis of E-Commerce Ranking System **2024**
The ACM Web Conference 2024 (WWW 2024)

SlowTrack: Increasing the Latency of Camera-based Perception in Autonomous Driving Using Adversarial Examples **2024**
The 38th Annual AAAI Conference on Artificial Intelligence (AAAI 2024)

Intriguing Properties of Diffusion Models: A Large-Scale Dataset for Evaluating Natural Attack Capability in Text-to-Image Generative Models **2024**
The IEEE/CVF Conference on Computer Vision and Pattern Recognition 2024 (CVPR 2024)

A Cross-Verification Approach with Publicly Available Map for Detecting Off-Road Attacks against Lane Detection Systems **2024**
Inaugural Symposium on Vehicle Security and Privacy 2024 (VehicleSec 2024)

Does Physical Adversarial Example Really Matter to Autonomous Driving? Towards System-Level Effect of Adversarial Object Evasion Attack **2023**
International Conference on Computer Vision 2023 (ICCV 2023)

Towards the Practicality of the Adversarial Attack on Object Tracking in Autonomous Driving **2023**
Inaugural Symposium on Vehicle Security and Privacy 2023 (VehicleSec 2023)

Waving the Double-Edged Sword: Building Resilient CAVs with Edge and Cloud Computing **2023**
The 60th ACM/IEEE Design Automation Conference 2023 (DAC 2023)

Infrastructure-Aided Defense for Autonomous Driving Systems: Opportunities and Challenges **2022**
The 4th International Workshop on Automotive and Autonomous Vehicle Security (AutoSec 2022)

Invisible for both Camera and LiDAR: Security of Multi-Sensor Fusion based Perception in Autonomous Driving Under Physical-World Attacks **2021**
The 42nd IEEE Symposium on Security and Privacy (IEEE S&P 2021)

Dirty Road Can Attack: Security of Deep Learning based Automated LaneCentering under Physical-World Attack **2021**
The 30th USENIX Security Symposium (USENIX Security 2021)

Deployability Improvement, Stealthiness User Study, and Safety Impact Assessment on Real Vehicle for Dirty Road Patch Attack **2021**
The 3rd International Workshop on Automotive and Autonomous Vehicle Security (AutoSec 2021)

Interpretable Deep Learning under Fire **2020**
The 29th USENIX Security Symposium (USENIX Security 2020)

Rendered Private: Making GLSL Execution Uniform to Prevent WebGL-based Browser Fingerprinting **2019**
The 28th USENIX Security Symposium (USENIX Security 2019)

Integration of Static and Dynamic Code Stylometry Analysis for Programmer De-anonymization **2018**
The 11th ACM Workshop on Artificial Intelligence and Security (AISec 2018)
Best Paper Award

ABSTRACT OF THE DISSERTATION

Novel Vulnerability Discoveries, Measurements, and Attack Designs for Safety-Critical Autonomous Systems from Practicality Perspectives

By

Ningfei Wang

Doctor of Philosophy in Computer Science

University of California, Irvine, 2024

Assistant Professor Qi Alfred Chen, Chair

Autonomous systems, such as autonomous driving (AD), rely heavily on real-time perception systems to detect and interpret their surroundings, such as traffic cones, pedestrians, traffic signs, vehicles, etc. These perception systems predominantly employ Deep Neural Networks (DNNs) for tasks such as real-time object detection due to their superior performance. However, DNNs are inherently vulnerable to adversarial attacks—maliciously crafted inputs designed to cause the DNNs to malfunction. Given the safety- and mission-critical nature of autonomous systems, it is crucial to systematically investigate the potential security vulnerabilities of these systems in real-world settings.

So far, one of the most general yet crucial limitations for prior research works in this area is their limited practicality in real-world autonomous system setups, either due to their sole focus on the AI component alone, which makes it non-trivial to transfer their component-only attack effects to the system level, or due to their research scopes limited to academic prototypes instead of real-world systems. For example, almost all prior adversarial attacks on Traffic Sign Recognition (TSR) systems have only assessed the effects on academic TSR models, leaving the impacts on real-world commercial TSR systems largely unexplored. While a few recent works have attempted to evaluate the impact on commercial TSR systems, these efforts are typically confined to a single vehicle model, sometimes even an unidentified one, raising questions about both the generalizability and representativeness of their findings.

In this dissertation, I present a suite of research efforts toward novel vulnerability discoveries, measurements, and attack designs for safety-critical autonomous systems from practicality perspectives. By systematically discovering and understanding the security vulnerabilities at both the DNN model level and autonomous system level, these research efforts aim to provide new and useful insights that can inspire further exploration of this largely under-explored aspect in this research area.

Chapter 1

Introduction

Autonomous systems are now a reality in our daily life, such as a wide variety of commercial and private Autonomous Driving (AD) vehicles are already driving on the road. For instance, millions of Tesla [158] are running on public road and Waymo taxi [288] is publicly available in Los Angeles, Phoenix, etc. To ensure safety, a fundamental pillar in the autonomous system is perception, which leverages sensors such as cameras and LiDARs (Light Detection and Ranging) to detect surroundings in real time. Due to the recent superior performance of Deep Neural Networks (DNNs), such perception systems predominantly employ DNNs for tasks such as real-time object detection.

Despite their advancements, DNNs are inherently vulnerable to adversarial attacks: maliciously crafted inputs designed to cause the DNNs to malfunction [123, 76, 198, 301, 323, 192]. Given the safety- and mission-critical nature of autonomous systems, it is crucial to systematically investigate the potential security vulnerabilities of them in real-world settings. Thus, various prior works have studied the security of AD perception such as the attacks that aim at causing the evasion of critical physical road objects (e.g., STOP signs and pedestrians) [148, 305, 83, 293, 330, 70, 110, 186, 69]. However, all the prior research studies have general yet crucial limitations in this area: their limited practicality in real-world autonomous system setups, either due to 1) their sole focus on the AI component alone, which makes it non-trivial to transfer their component-only attack effects to the system level, or 2) their research scope limited to academic prototypes instead of real-world systems. For example, almost all prior studies [110, 187, 330, 83, 213, 335, 283, 237, 186, 104, 332, 195] have only assessed the effects of adversarial attacks on academic Traffic Sign Recognition (TSR) models, leaving the impacts on real-world commercial TSR systems largely unexplored. While a few recent works [148, 243] have attempted to evaluate the impact on commercial TSR systems, these efforts are typically confined to a single vehicle model, sometimes even an unidentified one, raising questions about both the generalizability and representativeness of their findings and even meaningfulness.

This dissertation aims to bridge these research gaps by focusing on novel vulnerability discoveries, measurements, and attack designs for safety-critical autonomous systems from practicality perspectives. AD systems are chosen as a primary case study due to their critical importance to safety.

Gap between AD system level and AI component level: To systematically address this research gap, my research focuses on two main security properties: *integrity* and *availability*. The research studies in each area are detailed as follows.

Integrity: Security of Multi-Sensor Fusion (MSF) based perception. All of prior attacks on AD perception are limited to attacks on a single source of AD perception, i.e., camera- or LiDAR-based AD perception *alone* [110, 329, 187, 324, 83, 221, 268, 72, 261]. By contrast, production high-level AD systems such as Waymo, Pony.ai, and Baidu Apollo, typically adopt an MSF-based design [34, 53, 7, 25], which fuses the results from different perception sources, e.g., LiDAR and camera, to achieve overall higher accuracy and robustness [115, 180, 86, 304, 179, 102, 163, 196, 101]. In such a design, under the assumption that not all perception sources are (or can be) attacked simultaneously, there *always* exists a possible

MSF algorithm that can rely on the unattacked source(s) to detect or prevent such an attack. This basic security design assumption is believed to hold in general [228, 128], and MSF is thus widely recognized as a general defense strategy against existing attacks on AD perception [228, 128, 72, 254].

This research presents a first study on the security property of MSF-based perception in AD systems today. I directly challenge the above basic security design assumption by demonstrating the possibility of effectively and simultaneously attacking *all* perception sources used in state-of-the-art MSF-based AD perception, i.e., camera and LiDAR [115, 180, 86, 304, 179, 102, 163, 196, 101]. This for the first time allows us to gain a concrete understanding of how much security guarantee the use of MSF can fundamentally provide as a general defense strategy for AD perception. Specifically, I consider physical-world attack vectors for high attack practicality, and target an attack goal with the most direct safety consequence for autonomous driving: cause a victim AD vehicle to fail in detecting a front obstacle.

Integrity: Physical-World hijacking attack on visual perception. Previous studies have highlighted the potential for adversarial attacks, including the use of adversarial patch [289, 148, 330, 283, 111], to fool object detection in AD perception. Such attacks cause the AD systems to ignore objects, posing significant safety risks. However, it is essential to recognize that AD perception extends beyond object detection to include Multiple Object Tracking (MOT) [53, 160, 150, 252]. MOT plays a pivotal role in AD perception by enhancing robustness against object detection errors. It ensures that only objects detected with consistent and stable accuracy across multiple frames are considered in the tracking results and, consequently, the driving decisions. This multi-frame consistency requirement presents a significant challenge to attacks that solely target object detection [150]. Therefore, a digital adversarial hijacking attack [150] to fool the entire AD perception has been proposed with adversarial patches as the attack vector. However, this prior attack [150] has shown limited effectiveness even in the digital space and is ineffective in the physical world.

This research proposes the first physical-world adversarial hijacking attack on the *entire AD perception system*. This attack adopts a strategic two-stage approach. In the initial stage, I focus on finding the most effective location for placing the adversarial patch to facilitate successful hijacking attacks. Subsequently, the second stage is to generate the adversarial patch, guided by the optimal locations identified in the preceding phase. I propose two loss functions in the second stage and given the inherent challenges arising from the interdependence of these loss functions, I propose a novel optimization strategy, which demonstrates superior performance compared to existing methods in prior works [150, 148, 283, 330]. Due to these, the attack can significantly outperform the existing hijacking attack [150].

Integrity: System-Level effect of adversarial object evasion attack. Various prior works have studied security of AD perception, especially the ones that aim at causing the evasion of critical physical road objects (e.g., STOP signs and pedestrians) [148, 305, 83, 293, 330, 70, 110, 186, 69]. Although these attacks are all motivated by causing erroneous driving behaviors at the AD system level (e.g., vehicle collisions and traffic rule violations), I find that so far they predominately only evaluate the attack success *at the targeted AI component level alone* (e.g., judged by per-frame object misdetection rates [83, 110, 305, 330, 148]), without further evaluation *at the system level*. Specifically, to systematically perform such system-level evaluation, I need to measure the end-to-end system-level attack success metrics (e.g., collision rates) with the full system-level attack context enclosing the attack-targeted AI component, for example, the remaining AD system pipeline such as object tracking, planning, and control, closed-loop control, and the attack-targeted driving scenario. In this paper, I call such system-level attack context *system model* for such adversarial attacks. This thus raises a critical research question: *can these existing works on physical adversarial object evasion attacks effectively achieve the desired system-level attack effects in the realistic AD system settings?*

To systematically answer this critical research question, I conduct the first measurement study on representative prior object-evasion attacks with regard to their capabilities in causing system-level effects. I propose a general framework, i.e., a system model, including perception modeling from the physical world, to measure STOP sign-evasion attack which is my target due to its high representativeness [252] and its direct impacts on driving correctness and road safety. The results show that *all* the representative existing works cannot cause any STOP sign traffic rule violation within the system model including a representative closed-loop control AD system in the common speed range for STOP sign-controlled roads in the real world even though the most effective attack can achieve more than 70% average attack success rate at the AI component alone.

I further investigate the potential root causes and find that all the existing works have design limitations on achieving effective system-level effects due to the lack of a system model in AD context for attack design: 1) physical model-inconsistent object size distribution in pixel sampling and 2) lack of vehicle plant model and AD system model consideration. A novel system-driven attack design is proposed, which can be integrated with all state-of-the-art attack methods to significantly improve system-level effects by overcoming the two limitations.

Availability: Increasing the latency of visual perception. While a plethora of prior works study the integrity of the AD perception, the *availability* aspect (real-time performance) of the system, which is crucial for safety (e.g., causing vehicle collision [275]) has been relatively underexplored, especially for the complete AD perception pipeline. While some existing AD security analysis has studied availability in object detection [249, 80], they do not encompass the entire AD perception since usually, object detection is a part of the AD perception [150]. Thus, these studies leave a critical research gap: *their proposed attack strategies may not be effective enough to conduct system-level effects in end-to-end AD systems.* As I demonstrate later, existing attacks targeting only object detection do not consistently produce highly potent system-level effects due to lack of entire AD perception consideration.

To fill in this critical research gap, I am the first to study availability-based adversarial attacks across the entire camera-based AD perception including both object detection and tracking. The proposed novel attack framework is designed to increase the latency of camera-based AD perception. Instead of solely targeting object detection, which might not yield potent system-level effects due to the limited increase of the latency, I realize the untapped potential of object tracking response time to generate a much more effective latency attack. To illustrate, an attacker focusing only on object detection might attempt to dramatically increase the number of proposed bounding boxes [80]. Object tracking might filter out a majority of these boxes and in common object detection post-processing [155, 327], the maximum number of detection is provided to ensure performance. Thus, the effectiveness of these attacks is limited. Due to the importance of object tracking, I first perform availability attack surface analysis by analyzing the time complexity of the state-of-the-art representative tracking algorithms. Then, I propose a two-stage attack strategy and formulate the attack as an optimization problem. Additionally, the novel loss function designs, encompassing score loss, bounding box area loss, and feature match loss, fully leverage the entire tracking-by-detection pipeline to generate effective latency-based attack.

Gap between academic prototypes and real-world systems: To deepen the understanding of this gap, I conduct the first large-scale measurement of prior physical-world adversarial attacks against commercial TSR systems. The study include four different commercial vehicle models from the top 15 best-selling brands in the US to ensure representativeness. The testing results reveal that it is actually possible for existing attack works from academia to have a highly-reliable attack success rate against certain commercial TSR system functionality, which is much higher than expected if compared to the transfer attack success rates reported by the corresponding original papers (e.g., for one such case the reported was <20%). Meanwhile I do not see generalizability of such attack capabilities over different commercial vehicle models and sign types. The much lower-than-expected attack success rate on commercial systems suggests the potential existence of deeper challenges for such attacks to take effect at the TSR system level. This measurement further prove the gap mentioned above.

Through further investigations, one major factor might be an unexpected *spatial memorization design* that commonly exists in commercial TSR systems. Specifically, this design exhibits an effect that once a sign is detected, both the detected sign type and the detected location are persistently memorized until the sign's reaction task is finished. Such a spatial memorization design can significantly impact the success of existing adversarial attacks at the TSR system level. For example, for hiding attacks, to achieve a system-level success in which the TSR system is unable to show the sign display at the sign's reaction task period, the attack has to be continuously successful at *all* possible detection moments that can trigger such memorization before the vehicle passes the sign. I further mathematically model its impact on the TSR system-level attack success for both hiding and appearing attacks, which results in new attack success metric designs that can systematically consider the spatial memorization effect.

By systematically discovering and understanding the security vulnerabilities at both the DNN model level and autonomous system level, these research efforts aim to provide new and useful insights that can inspire further exploration of this largely under-explored aspect in this research area.

Based on the extensive research outlined above, I have formulated my dissertation statement as follows.

Dissertation statement: Vulnerabilities of AI components in autonomous systems can be both (1) physically realizable, and (2) capable of damaging multiple types of system-level security properties (e.g., not only integrity but also availability). Moreover, existing security analysis methods that focus solely on AI components alone or academic prototype systems can be fundamentally insufficient.

Research impact: My research has left a mark on both academia and industry. To date, I have authored *19 research papers (4 under submission), of which 8 were accepted in top-tier conferences*—5 in top-tier computer security conferences such as *IEEE Security & Privacy, NDSS, and USENIX Security*, 2 in the top-tier computer vision conference, *ICCV and CVPR*, 1 in the top-tier machine learning conferences, *AAAI*, and 1 in the top-tier the web & information retrieval conferences, *WWW*. My work has also garnered *media coverage*, *interviews*, and led to *vulnerability disclosures.* Notably, I received the *best paper award at AISec 2018* [281] and the *best technical poster award at NDSS 2020.* To date, over 30 leading AD companies—including Tesla, GM, Volkswagen, Zoox, Hyundai, Baidu, Bosch, TuSimple, Lyft, Nuro, Toyota, and more—have initiated investigations into the security vulnerabilities in AD perception algorithms identified in my research. Several of these corporations have further engaged in discussions, exploring potential testing within their products and devising mitigation strategies.

1.1 Dissertation Roadmap

This dissertation is structured as follows. Chapter 2: This chapter lays the background and related work required to understand the research conducted in this dissertation. Chapters 3 to 6 – addressing the research gap between AD system level and AI component level: Chapter 3 introduces the security of MSF based perception; Chapter 4 proposes a physical-world hijacking attack on visual perception; Chapter 5 presents the system-level effect of adversarial object evasion attack; and Chapter 6 reveals an novel attack to increase the latency of the visual perception. Then, Chapter 7 focuses on addressing another research gap, i.e., gap between academic prototypes and real-world systems. The first large-scale measurement study of prior physical-world adversarial attacks against commercial TSR systems is conducted and significant insights are uncovered. This dissertation is further concluded in

Chapter 8 with a detailed discussion of future work.

Chapter 2

Background and Related Work

2.1 Autonomous Driving (AD) System Designs

The detailed overview of AD system designs is depicted in Fig. 2.1. This system gather data from various sensors, such as cameras and LiDAR, to perceive the physical world. The sensor data is then processed by the AD system to make crucial control decisions, such as steering adjustments. The core components of an AD system include perception, prediction, and planning. *Prediction*: This component estimates the future states (position, heading, velocity, acceleration, etc.) of nearby objects, which is vital for dynamic environments. *Planning*: Responsible for making trajectory-level driving decisions (e.g., cruising, stopping, lane changing), this module ensures that actions are safe and adhere to traffic regulations. *Perception*: Acting as the "eyes" of the AD system, this module processes real-time environmental data through functions such as object detection, object tracking, multi-sensor fusion (MSF), lane detection, and traffic sign detection. Given its critical role in vehicle safety and operational integrity, the perception module is selected as the primary and representative research focus of this dissertation.

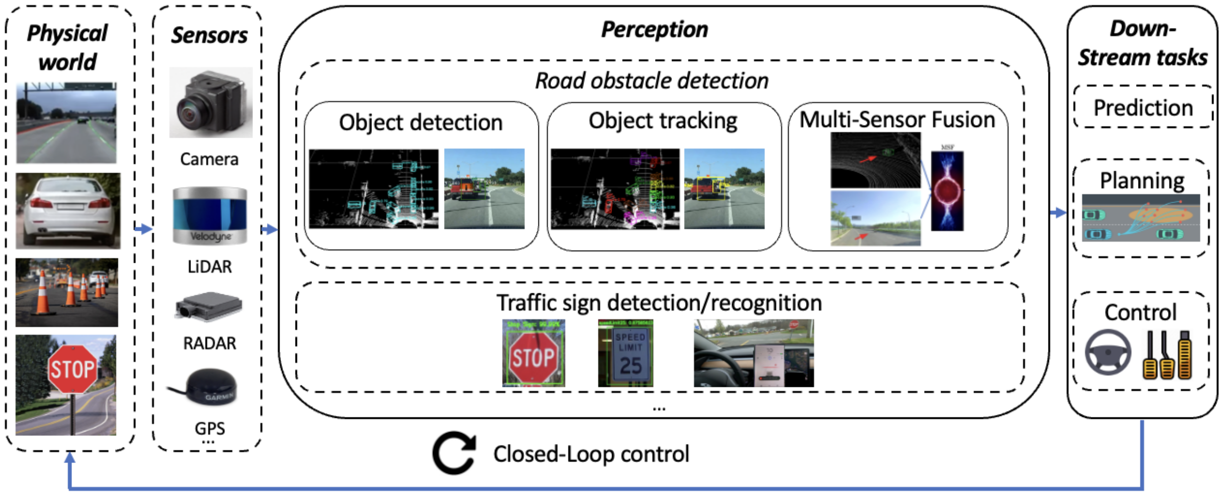


Figure 2.1: Overview of AD system designs.

2.2 Adversarial Attacks

Recent works find that DNN models are generally vulnerable to *adversarial attacks* [123, 220, 299, 302, 298, 300, 227]. Some works further explored such attacks in the physical world [110, 173, 64, 267, 324, 83, 187, 329, 56, 195]. However, all the prior research studies have general yet crucial limitations in this area: their limited practicality in real-world autonomous system setups, either due to 1) their sole focus on the AI component alone, which makes it non-trivial to transfer their component-only attack effects to the system level, or 2) their research scope limited to academic prototypes instead of real-world systems. For example, almost all prior studies [110, 187, 330, 83, 213, 335, 283, 237, 186, 104, 332, 195] have only assessed the effects of adversarial attacks on academic Traffic Sign Recognition (TSR) models, leaving the impacts on real-world commercial TSR systems largely unexplored. While a few recent works [148, 243] have attempted to evaluate the impact on commercial TSR systems, these efforts are typically confined to a single vehicle model, sometimes even an unidentified one, raising questions about both the generalizability and representativeness of their findings and even meaningfulness. In this dissertation, I present a suite of research efforts toward novel vulnerability discoveries, measurements, and attack designs for safety-critical autonomous systems from practicality perspectives.

Chapter 3

Security of Multi-Sensor Fusion based Perception in Autonomous Driving Under Physical-World Attacks

3.1 Introduction

Today, high-level (e.g., Level-4 [92]) self-driving cars, or Autonomous Vehicles (AV) [2], are under rapid development. Some of them, e.g., Google Waymo [33] and TuSimple [30], are already providing services on public roads. To ensure correct and safe driving, a fundamental pillar in the Autonomous Driving (AD) system is *perception*, which leverages sensors such as cameras and LiDARs (Light Detection and Ranging) to detect surrounding obstacles in real time. Due to the direct impact on safety-critical driving decisions such as collision avoidance, various prior works have studied the security of AD perception under realistic physical-world attack vectors such as adding stickers, posters, or paintings to traffic signs [110, 329, 187, 324, 83], or shooting lasers to the LiDAR [72, 261].

All of these studies, however, are limited to attacks on a single source of AD perception, i.e., camera- or LiDAR-based AD perception *alone* [110, 329, 187, 324, 83, 221, 268, 72, 261]. By contrast, production high-level AD systems such as Waymo, Pony.ai, and Baidu Apollo, typically adopt a Multi-Sensor Fusion (MSF) based design [34, 53, 7, 25], which fuses the results from different perception sources, e.g., LiDAR and camera, to achieve overall higher accuracy and robustness [115, 180, 86, 304, 179, 102, 163, 196, 101]. In such a design, under the assumption that not all perception sources are (or can be) attacked simultaneously, there *always* exists a possible MSF algorithm that can rely on the unattacked source(s) to detect or prevent such an attack. This basic security design assumption is believed to hold in general [228, 128], and MSF is thus widely recognized as a general defense strategy against existing attacks on AD perception [228, 128, 72, 254].

This paper presents a first study on the security property of MSF-based perception in AD systems today. We directly challenge the above basic security design assumption by demonstrating the possibility of effectively and simultaneously attacking *all* perception sources used in state-of-the-art MSF-based AD perception, i.e., camera and LiDAR [115, 180, 86, 304, 179, 102, 163, 196, 101]. This for the first time allows us to gain a concrete understanding of how much security guarantee the use of MSF can fundamentally provide as a general defense strategy for AD perception. Specifically, we consider physical-world attack vectors for high attack practicality, and target an attack goal with the most direct safety consequence for autonomous driving: cause a victim AV to fail in detecting a front obstacle.

Although prior works have designed successful physical-world attacks on AD perceptions based only on camera or only on LiDAR, we find that simply combining their designs does not achieve our goal. First, we need to identify a physical-world attack vector effective for both camera and LiDAR, which can not be satisfied by those popular ones used in prior works. For example, adding stickers changes an object's texture (e.g., color) but not its shape; this can be effective for camera [110, 329, 187, 324, 83] but not LiDAR. Conversely, laser shooting has been shown to be effective for LiDAR-based AD perception [72, 261], but not for camera-based ones. Second, no matter what attack vector we use, we need to further address 2 design challenges: (1) We need to differentiably synthesize the physical attack impacts simultaneously and consistently onto both camera images and LiDAR point clouds. For certain attack vectors, e.g., differentiably modelling the impact of lasers on camera images, this can be very challenging. (2) To improve run-time performance, the state-of-the-art LiDAR-based AD perception uses aggregated features of the 3D points grouped at the level of 2D or 3D cells [107, 333, 165, 53, 60, 312, 86]; however, their calculation is by nature non-differentiable (§3.3.2), which makes the attack difficult to optimize.

Towards this end, we design a novel physical-world adversarial attack method, MSF-ADV, which addresses the challenges above and thus fundamentally challenges the basic MSF design assumption in AD perception. We employ *adversarial 3D object* as the attack vector, with the key observation that different shapes of a 3D object can lead to both point position changes in LiDAR point clouds and pixel value changes in camera images. Thus, an attacker can leverage *shape manipulations* to introduce input perturbations to both camera and LiDAR simultaneously. To achieve the attack goal, the attacker simply places such an object on the roadway; this can be conveniently accomplished with modern 3D printing services and an object type commonly expected on the roadway, e.g., a traffic cone but with a slightly worn or broken look.

To systematically generate effective adversarial 3D objects, we adopt an optimization-based approach that starts with a 3D mesh of a normal object, e.g., a normal traffic cone, and performs shape manipulation by changing its vertex positions. Under this attack vector, we address design challenge 1 by constructing differentiable 3D rendering functions to synthesize the attack-influenced point clouds and camera images. For design challenge 2, we find that all commonly-used cell-level aggregated features can be differentiably derived by the *point-inclusion* property (§3.4.4). Thus, we first design a differentiable and accurate approximation for such property, and then use it as a building block to differentiably compute the gradient of the cell-level aggregated features during the optimization. We also employ domain-specific designs for attack robustness, stealthiness, and physical-world realizability.

We evaluate MSF-ADV with MSF algorithms included in 2 open-source full-stack AD systems, Baidu Apollo [53] and Autoware.AI [7], that have high representativeness in practice, e.g., Apollo is ranked as the top 4 leading AD developers along with Waymo, Ford, and Cruise [22]. We select 3 object types and evaluate each on 100 real-world driving scenarios from the KITTI dataset [119]. Our results show that the generated adversarial objects achieve more than 91% success rate across different object types and MSF algorithms. We also find that our attack is (1) stealthy from the driver's view based on a user study, (2) robust to different victim approaching positions and angles, with over 95% average success rates, and (3) transferable across different MSF algorithms, with an average transfer attack success rate of around 75%.

To understand the attack realizability in the physical world, we 3D-print our adversarial objects, and evaluate them using real LiDAR and camera devices. Using a vehicle with a LiDAR mounted, we find that our 3D-printed adversarial object can successfully evade LiDAR detection in 99.1% (107) of the total 108 collected frames. Using a miniature-scale experiment setting (§3.5.5.2), we find that our adversarial object has a 85-90% success rate to evade both LiDAR and camera detection at 20 randomly-sampled positions.

To understand the end-to-end safety impact, we further evaluate our method using a production-grade AD simulator, and find that our adversarial traffic cone can cause a 100% vehicle collision rate for an Apollo AV across 100 runs. In contrast, the collision rate with a normal traffic cone is 0%. Demo videos are at our project website: **https://sites.google.com/view/cav-sec/msf-adv**. We also evaluate various existing DNN-level defense strategies (e.g., input transformation and augmenting training data), and discuss future defense directions. Our code and data are released at our website [23].

In summary, this work makes the following contributions:

- We are the first to study security issues of MSF-based AD perception and the first to challenge the basic MSF design assumption in the AD context. We successfully design and engineer a physical-world adversarial attack aiming at generating adversarial 3D object to mislead a victim AV to fail in detecting it and thus crash into it.
- We adopt an optimization-based approach that addresses two main design challenges: non-differentiable target camera and LiDAR sensing systems, and non-differentiable cell-level aggregated features used by LiDAR. We also design strategies to enhance the attack robustness, stealthiness, and physical-world realizability.
- We evaluate on MSF algorithms included in representative open-source industry-grade AD systems in real-world driving scenarios. Our attack is shown to achieve over 91% success rates across different object types and MSF algorithms. Such high effectiveness can also be achieved with (1) high stealthiness, (2) high robustness to victim positions, (3) high transferability across MSF algorithms, and (4) high physical-world realizability after being 3D-printed and captured by LiDAR and camera devices.
- To understand the end-to-end safety impact, we further evaluate the proposed attack on a production-grade simulator, and show that our attack can cause a 100% vehicle collision rate to an industry-grade AD system. We also evaluate and discuss defense strategies.

While MSF is widely recognized as a promising and general defense strategy for existing attacks on AD perception [228, 128, 72, 144, 254, 308, 310, 222, 202, 79], prior works have neither studied the security of existing MSF algorithms in practical AD settings, nor made attempts to understand whether the very basic security design assumption for MSF can fail. In this paper, we make the first attempt towards this direction, and we hope that our findings and insights can inspire more future research into this largely overlooked research perspective.

3.2 Background

3.2.1 MSF-based AD Perception

In high-level (e.g., Level 4 [92]) AD systems, *perception* is a critical module that detects surrounding objects in real time. Due to its direct impact on safety-critical driving decisions such as collision avoidance, AD perception in production high-level AD systems such as Google Waymo, Pony.ai, and Baidu Apollo predominantly adopts a Multi-Sensor Fusion (MSF) based design [34, 53, 7, 25]. In this paper, we call such design *MSF-based AD perception*, or *MSF* for short. In this paper, we focus on MSF designed for *in-road obstacle detection*, e.g., front cars, which is the most basic task for AD perception.

MSF design principle and basic assumption. In MSF-based AD perception, the final object detection results are obtained by fusing multiple perception sources such as camera and LiDAR, with the goal of leveraging their strengths while compensating their weaknesses to achieve overall higher accuracy and robustness than those achievable by a single perception source [115, 180, 86, 304, 179, 102, 163, 196, 101]. For example, LiDAR is a ranging-based sensor by shooting lasers, which thus is more difficult to capture the texture information (e.g., color) of an object compared to cameras [115]. Camera images, on the other hand, cannot directly provide the depth information of an object [180, 196], which can be overcome by LiDARs. Thus, an MSF algorithm can be designed to leverage both the depth information from LiDAR point clouds and the texture information from camera images to achieve higher object detection performance than those using either camera or LiDAR alone [86, 180]. To achieve such overall higher accuracy and robustness, the basic design assumption is that *there generally exists at least one source that can provide the correct results.* In this paper, we are the first to challenge such assumption in the AD context.

Representative MSF algorithm design. In AD perception, state-of-the-art MSF algorithms predominately use 2 perception sources: camera and LiDAR [115, 180, 86, 304, 179, 102, 163, 101]. Fig. 3.1 shows an overview of a typical MSF-based AD perception design. In industry AD systems, before running the MSF, the raw camera and LiDAR inputs, i.e., camera images and LiDAR point clouds, are usually first pre-processed [53, 7] to prepare the camera- and LiDAR-side MSF inputs, which can improve the run-time algorithm performance (detailed later).

In the MSF algorithm, state-of-the-art designs predominantly adopt DNN networks to process the LiDAR-side and camera-side MSF inputs [53, 7, 115, 180, 86, 304, 179, 102, 163, 101], due to the recent superior performance of deep learning [331, 183, 185, 59, 43, 44, 41, 42]. In this paper, we call them *LiDAR perception networks* and *camera perception networks* inside the MSF algorithm. Next, the processing results from these two networks are fused using (1) DNNs [86, 304, 115, 180, 179, 102, 163], or (2) hard-coded matching and prioritization rules [53, 7]. Rule-based fusion is usually a late fusion, i.e., applied to the end results of the two networks, while DNN-based one can be a late or early fusion, i.e., at the intermediate perception results, which can be fused more deeply and thus potentially lead to higher accuracy. Meanwhile, rule-based fusion has two unique benefits. First, it is more modular and thus can flexibility combine different camera and LiDAR perception models [316]. Second, it is easier to debug and interpret than DNNs [88], and also to hard-code safety rules and measures [316]. In our attack design later in §3.4, we comprehensively consider both fusion designs.

When preparing the camera- and LiDAR-side MSF inputs, typical pre-processing steps include data transformation such as rotations and shifting, applying Region of Interest (ROI) filter to remove unrelated input portions, and extracting aggregated features from the raw

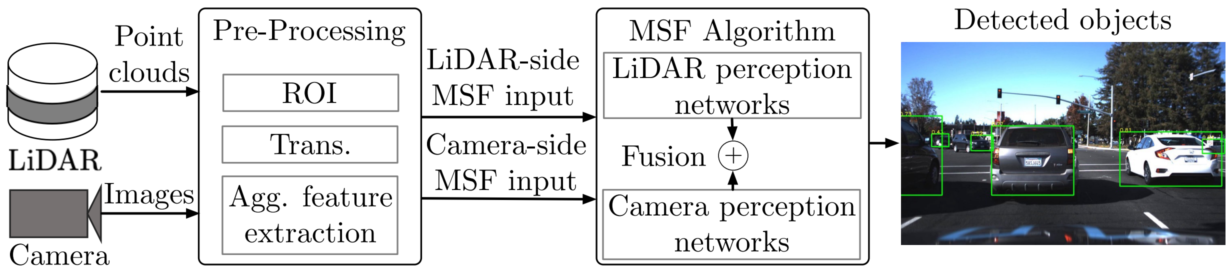


Figure 3.1: Overview of MSF-based AD perception design.

input. These pre-processing steps can largely reduce the sizes and dimensions of the MSF algorithm inputs, which can thus greatly improve the run-time algorithm performance [129]. Considering that the raw point cloud data can include millions of 3D points per second [32], such pre-processing is especially beneficial for LiDAR perception. Thus, many state-of-the-art LiDAR-based AD perception model designs choose to use aggregated input features such as average height and intensity of the 3D points grouped at the level of *3D cells*, or *voxels* [107, 279, 333, 165]. Some state-of-the-art designs even choose to further aggregate the features in such 3D cells to 2D cells in Bird's-Eye View (BEV) to further improve the real-time detection performance [312], which is thus the most popularly adopted in industry-grade AD systems [53, 60, 312, 86]. As detailed in §3.3.2, such popular adoption of cell-level aggregated features for LiDAR introduces a unique challenge to our attack design.

3.2.2 Physical-World Adversarial Attack

Recent works find that DNN models are generally vulnerable to adversarial example, or *adversarial attacks* [123, 220, 299, 302, 298, 300, 227]. Some works further explored such attacks in the physical world [110, 173, 64, 267, 324, 83, 187, 329, 56]. In the AD context, previous works have designed successful physical-world adversarial attacks on the camera-based AD perception *alone* [110, 329, 187, 324, 83, 221, 268], or the LiDAR-based one *alone* [72, 261]. However, none of them have considered MSF-based AD perception, which is predominantly adopted in industry AD systems today (§3.2.1) and in principle can be more robust against these attacks (§3.1). Also, as detailed later in §3.3.2, blindly combining these prior designs cannot directly lead to successful attacks on MSF due to various new and unique challenges.

3.3 Problem Formulation and Design Challenges

3.3.1 Attack Goal and Threat Model

Attack goal: Fundamentally defeat MSF design assumption. In this paper, we target an attack goal with the most direct safety impact on driving: fool the MSF-based AD perception in the victim AV to fail in detecting a front obstacle and thus crash into it. Even when the vehicle has a fail-safe Automatic Emergency Brake (AEB) system, e.g., based on RADAR or ultrasonic sensors, such a crash is still possible for two reasons. First, today's AEB systems are not perfect. For example, a recent study shows that the ones in popular vehicle models today fail to avoid crashes 60% of the time [36]. Second, even if they can successfully perform emergency stop, they cannot avoid being hit by rear vehicles that fail to yield on time. To achieve this goal, in this paper we target physical-world attack vectors in the AD context for high practicality and realism.

Due to the basic design assumption of MSF (§3.2.1), as long as there still exists at least one perception source that is not attacked, it is always possible for the unattacked source(s) to correct the final fused perception results and thus defeat our attack goal. Thus, in this paper we aim at designing an attack that can effectively attack *all* perception sources used in the MSF-based AD perception. This can enable our design to *fundamentally* defeat the MSF design assumption and thus most generally achieve our goal above. As the combination of camera and LiDAR is most popularly adopted in state-of-the-art MSF-based AD perception (§3.2.1), in this paper our design needs to attack both camera and LiDAR simultaneously.

Threat Model. As the first study to achieve the attack goal above, in this work we mainly focus on a white-box attack setting, i.e., assuming that the attacker has a full knowledge of the MSF algorithm used in the victim AD system. This is the same assumption made in most prior adversarial attacks on camera- or LiDAR-based AD perception [72, 111, 110, 329]. To achieve this, the attacker may obtain a victim AV model, e.g., by purchasing or renting [8], and then reverse engineer its perception module, which has been shown as possible on Tesla Autopilot [15]. The attacker can also target the AVs using open-source MSF-based AD perception algorithms [53, 7]. In the attack preparation time, we assume that the attacker can collect camera images and LiDAR point clouds of a targeted road where she plans to launch the attack.

3.3.2 Design Challenges

As described in §3.2.1, in state-of-the-art MSF algorithms, the camera and LiDAR perception networks are DNN based. Although no prior works consider attacking MSF, many designed successful physical-world adversarial attacks on camera- *or* LiDAR-based AD perception DNN models. However, we find that blindly combining these prior designs cannot directly achieve our goal due to 3 unique challenges:

C1. Lack of a single physical-world attack vector effective for both camera- and LiDAR-based AD perception. To achieve our attack goal, we need to find physical-world attack vectors for both camera- and LiDAR-based perception networks in MSF. However, so far none of the attack vectors used in previous physical-world adversarial attacks in the AD context have shown effectiveness in affecting both. For camera-based AD perception, previous works predominately consider adding stickers/posters [110, 329], painting [324, 83], or changing brightness [221, 268], which can only change the *texture* of an obstacle but not its *shape* and thus can barely affect the LiDAR point clouds. On the LiDAR side, LiDAR spoofing [72, 261], which shoots lasers to LiDAR, has shown to be effective in the

AD context. Although lasers can also affect camera inputs [310], no prior work has studied its effectiveness for fooling camera-based AD perception models. One possible solution is to use separate attack vectors for them, e.g., using stickers for camera and laser shooting for LiDAR. However, this not only adds up the attack deployment costs and thus lowers the realizability and stealthiness, but also requires precise synchronizations across the attack processes. Thus, it is highly desired to identify one single attack vector that can effectively attack both at the same time.

C2. Need to differentiably synthesize physically-consistent attack impacts onto both camera and LiDAR. To systematically generate adversarial inputs, prior works generally adopt optimization-based approaches, which have shown both high efficiency and effectiveness [329, 56]. Since adversarial attack generation typically takes thousands of optimization iterations [200, 76], it is almost impossible in practice to physically drive vehicles on the target road to obtain the attack-influenced camera images and LiDAR point clouds every time the adversarial inputs are updated in an iteration. Thus, we need to digitally synthesize the impacts of the adversarial stimulus from the physical world onto both camera images and LiDAR point clouds, and such synthesizing needs to be differentiable to enable effective optimization. As discussed in **C1**, no single attack vector has been studied for both camera- and LiDAR-based AD perception so far in prior works. Thus, no matter what attack vector we identify to address **C1**, we need to design a new differentiable synthesizing function for at least one of the perception sources, which can be quite challenging for certain physical-world attack vectors, e.g., differentiably modelling the impact of lasers on camera inputs from different distances and angles. Meanwhile, since such attack impacts come from the same physical-world stimulus, the synthesized impacts to the camera images and the LiDAR point clouds need to be *physically consistent*, e.g., conforming to their different mounting positions in the AV.

C3. Need to handle non-differentiable pre-processing steps in AD perception. As introduced in §3.2.1, in industry AD systems, images and point clouds are usually pre-processed before fed into the MSF algorithm. In particular, state-of-the-art LiDAR-based AD perception models popularly use aggregated features of 3D points grouped at level of 2D or 3D cells (§3.2.1). To calculate such cell-level aggregated features, the necessary first step is to calculate whether an input point is inside a cell or not. In this paper, we call it a *point-inclusion* property. By nature, such property is discontinuous, i.e., 0 and 1 for outside and inside a cell. This causes the calculation of any cell-level aggregated features non-differentiable with regard to the LiDAR point clouds, which thus makes our optimization difficult to be effective. So far, no prior works have considered a general design to handle such non-differentiable pre-processing steps for LiDAR.

3.4 Attack Design: MSF-ADV

In this paper, we are the first to address all the 3 challenges in §3.3.2 by designing a novel physical-world adversarial attack method, *MSF-ADV*, which thus can fundamentally defeat the MSF design assumption in AD perception.

3.4.1 Design Overview

To address the challenges in §3.3.2, our MSF-ADV method has the following novel designs:

Adversarial 3D object: physically-realizable and stealthy attack vector for MSF-based AD perception. To address **C1**, we identify *adversarial 3D object* as the physical-world attack vector against MSF-based AD perception. Our key insight is that different shapes of a 3D object can lead to not only point position changes in LiDAR point clouds but also pixel value changes in camera images. Thus, the attacker can leverage *shape manipula-*

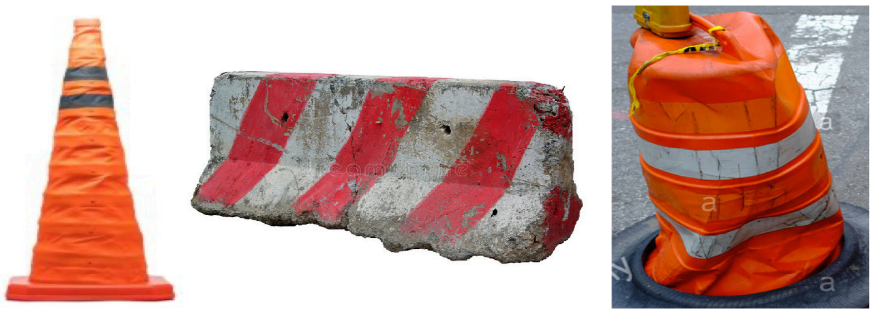


Figure 3.2: Real-world traffic objects with worn or broken looking shapes, which can be mimicked by our physical-world attack vector: adversarial 3D object with *shape manipulations*.

tions of such an object to introduce adversarial input perturbations simultaneously to both camera and LiDAR perception networks in the MSF algorithm. To achieve the attack goal, the attacker simply places such an object in the roadway to trick the victim AV to crash into it.

Beside satisfying **C1**, such an attack vector also has 2 other advantages. First, it is easily realizable and deployable in the physical world. For example, the attacker can construct it digitally in a 3D mesh and 3D-print it, which is convenient today through online services [1]. Second, it can achieve high stealthiness by mimicking a normal traffic object that can legitimately appear in the middle of the road, e.g., a traffic cone or barrier, but with a worn or broken look, which is not uncommon in the real world as shown in Fig. 3.2. In our design (§3.4.5), we also constrain the degree of the shape changes from the normal object to achieve high stealthiness. Note that although it is possible to manipulate the object texture (e.g., color) together with the shape in our design, we intentionally choose to not consider it in this paper as it can greatly harm stealthiness and also incur additional printability issues, which is a common challenge for physical-world adversarial attacks using stickers/posters [109, 110, 141].

Causing road safety threats. To make such an object both easy to deploy and able to cause severe crashes, the attacker can choose smaller objects such as a rock or traffic cone but fill it with granite or even metal to make it harder and heavier. For example, a 0.5

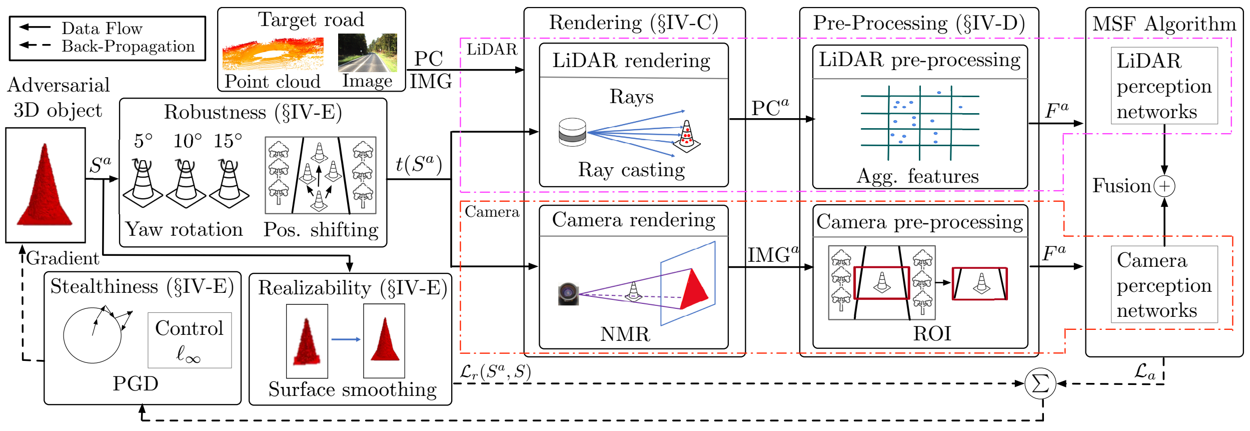


Figure 3.3: Overview of the optimization-based adversarial 3D object generation in MSF-ADV.

cubic-meter rock or a 1-meter high traffic cone [27] filled with some aluminum can easily weigh over 100 kg, which can trip the victim AV to lose control, damage the chassis, or break the windshield glass if bounced up when driving at a high speed. Besides causing damages by the crash itself, the attackers can also exploit the *semantic meaning* of certain road object types such as traffic cones. For example, the attacker can design an AV-specific attack by placing nails or glass debris behind an adversarial traffic cone object so that failing to detect it can lead to tire blowout of a targeted AV. Here, the safety damages are not directly caused by the traffic cone crash itself, and thus in this case the adversarial traffic cone can be small and lightweight like normal ones to make it easier to 3D-print, carry, and deploy.

Optimization-based adversarial 3D object generation. To systematically generate adversarial 3D objects, we adopt an optimization-based approach similar to prior works [110, 329, 187, 324, 83, 72, 261]. We start with a 3D mesh of a normal 3D object, e.g., a normal traffic cone, and then introduce shape manipulations by changing its vertex positions. To address **C2**, due to the choice of adversarial 3D objects as the attack vector, we can conveniently leverage existing 3D rendering techniques in computer graphics to simulate the functionalities of the physical equipment, i.e., camera and LiDAR, and thus systematically synthesize the attack-influenced camera images and LiDAR point clouds. Specifically, to enable the end-to-end optimization process, we perform differentiable constructions of these rendering processes, and use the relative positions to the 3D object to ensure the physical consistency with the corresponding camera and LiDAR mounting positions.

With the synthesized raw camera images and LiDAR point could, next we design the differentiable approximation function for the non-differentiable pre-processing step (non-differentiable cell-level aggregated feature calculation) to enable the end-to-end optimization. To address this, our key insight is that all the commonly-used cell-level aggregated features can be differentiably derived by the point-inclusion property (detailed later in §3.4.4). Thus, we first design a novel and accurate differentiable function to approximate the calculation of the point-inclusion property, and then use it as a building block to achieve differentiable computations of the pre-processing steps for LiDAR. In the optimization process, we also have other domain-specific designs, e.g., for attack robustness, stealthiness, and physical-world realizability, which will be detailed in the following sections.

3.4.2 MSF-ADV Methodology Overview

In this section, we provide an overview of our MSF-ADV method, and will detail its components in later sections.

Problem formulation. We formulate the attack generation process as the following optimization problem:

$$\min_{S^a} \ \mathbb{E}_{t\sim T}[\mathcal{L}_a(t(S^a);\mathcal{R}^l,\mathcal{R}^c,\mathcal{P},\mathcal{M}) + \lambda \cdot \mathcal{L}_r(S^a,\ S)] \tag{3.1}$$

$$\text{where} \ \ \mathrm{PC}^a = \mathcal{R}^l(t(S^a),\ \mathrm{PC}) \tag{3.2}$$

$$\mathrm{IMG}^a = \mathcal{R}^c(t(\ S^a),\ \mathrm{IMG},\mathrm{C}) \tag{3.3}$$

$$F^a = \mathcal{P}(\mathrm{PC}^a,\ \mathrm{IMG}^a) \tag{3.4}$$

$$\mathcal{L}_a(t(S^a);\mathcal{R}^l,\mathcal{R}^c,\mathcal{P},\mathcal{M}) = \mathcal{O}(\mathcal{M}(F^a)) \tag{3.5}$$

$$\text{subject to} \ \ \Delta(S^a,\ S) \le \epsilon \tag{3.6}$$

S is the original benign object and S^a is the adversarial one. We use vertex-face ($\boldsymbol{v}$-$\boldsymbol{f}$) meshes to represent them, i.e., $S = (\boldsymbol{v}, \boldsymbol{f})$ and $S^a = (\boldsymbol{v}^a, \boldsymbol{f}^a)$. In Eq. (3.1), the optimizing parameter is the adversarial object S^a, and we only change its vertices $\boldsymbol{v}^a$. The objective function includes: (1) an adversarial loss $\mathcal{L}_a$, which is designed to achieve our attack goal by misleading the MSF algorithm $\mathcal{M}(\cdot)$ to fail in detecting S^a, and (2) a realizability loss $\mathcal{L}_r(\cdot)$, which is designed to improve smoothness of the S^a surface to benefit both the printability and stealthiness (§3.4.5). To improve the robustness of S^a in the physical world, we apply Expectation over Transformation (EoT) [56] by introducing a set of 3D transformation T to S^a and optimize the expectation of their objective function values in Eq. (3.1). λ is a balancing hyper-parameter.

In Eq. (3.2) and Eq. (3.3), $\mathcal{R}^l(\cdot)$ and $\mathcal{R}^c(\cdot)$ are the differentiable LiDAR and camera rendering functions respectively (§3.4.3). They generate the attack-influenced point clouds PC^a and images IMG^a given the corresponding backgrounds of the target road (PC and IMG). PC^a and IMG^a are then fed into the differentiable pre-processing approximation function $\mathcal{P}(\cdot)$ to obtain the attack-influenced MSF input features F^a (§3.4.4). F^a is fed into MSF algorithm $\mathcal{M}(\cdot)$ in Eq. (3.5), and $\mathcal{O}(\cdot)$ is designed to extract the output features related to the object's confidence score of the adversarial object. To achieve high stealthiness, in Eq. (3.6) we limit the shape deformation between S and S^a within a threshold ϵ by using a distance metric $\Delta(\cdot)$ (e.g., L_p distance metric: $\Delta(S^a, S) = ||S^a - S||_p$).

Optimization process overview. Fig. 3.3 overviews our optimization process. As shown, given a 3D object S^a initialized with S, we first apply 3D transformations (e.g. rotation and position shifting) T to generate multiple samples $t(S)$ to improve the robustness of the adversarial object against environment's variation. Next, each one of them, along with the LiDAR point clouds (PC) and camera image (IMG) background from the target road, are fed into the rendering functions ($\mathcal{R}^l(\cdot), \mathcal{R}^c(\cdot)$), pre-processing approximation functions ($\mathcal{P}(\cdot)$), and the MSF algorithm ($\mathcal{M}(\cdot)$) to calculate $\mathcal{L}_a$. Additionally, the realizability loss $\mathcal{L}_r(S^a, S)$ is added to $\mathcal{L}_a(\cdot)$ together using Eq. (3.1) to construct our loss function. To solve it, we use Projected Gradient Descent (PGD). Specifically, we compute its gradients with respect to the vertex positions $\boldsymbol{v}^a$ of S^a and constrain the gradients with a stealthiness threshold ϵ. We then update S^a using these gradients. We iteratively apply this process until S^a cannot be detected by the MSF algorithm.

3.4.3 Differentiable Rendering

In this section, we detail the differentiable rendering functions $\mathcal{R}^l(\cdot)$ and $\mathcal{R}^c(\cdot)$ in Eq. (3.2) and Eq. (3.3). To ensure physical consistency, we define S^a in the LiDAR coordinate system, which is convenient as it is by nature 3D. For camera rendering, we then use a calibration matrix C to transform S^a from the LiDAR coordinate system to the camera coordinate system. C can be obtained by measuring the relative positions between the camera and the LiDAR of AV. To achieve differential rendering, we leverage existing differentiable ray-casting methods [19] for LiDAR and NMR [159] for camera.

Table 3.1: Summary of commonly-used cell-level aggregated features in state-of-the-art LiDAR-based object detection models. Our novel soft point-inclusion property calculation (§3.4.4) can be used to differentiably derive all of them.

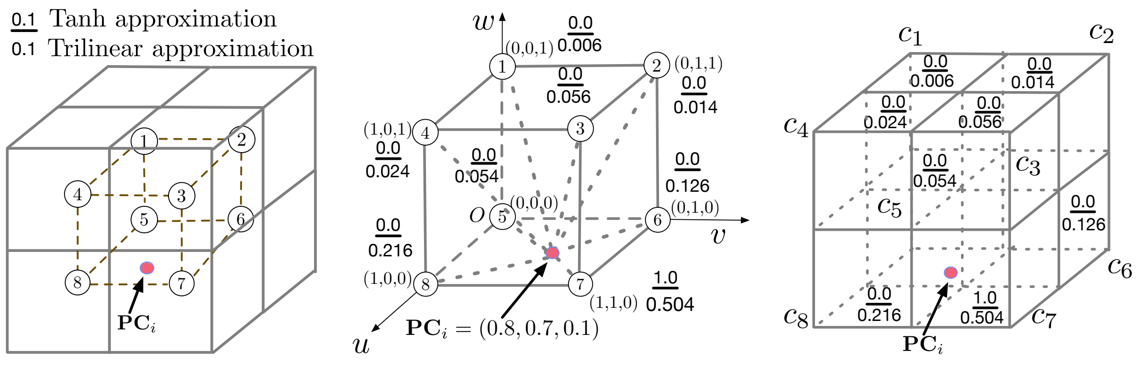
Cell-level Aggregated Features	Used in
Occupancy	[53, 312, 7, 107, 279, 205]
Count	[53, 7]
Height (min/max/mean)	[53, 7, 86, 60, 163]
Intensity (min/max/mean)	[53, 7, 312, 86, 60, 107, 279]
Density	[86, 60, 163]

3.4.4 Pre-Processing Step Approximation

In this section, we detail the construction of the differentiable pre-processing function $\mathcal{P}(\cdot)$. Most of the pre-processing steps such as ROI, rotation, and position shifting (§3.2.1) can be directly constructed differentiably using projective and affine transformations. However, such construction is especially challenging for the calculation of cell-level aggregated features such as cell occupancy and the mean height of the points inside a cell, due to the discontinuity of the point-inclusion property as discussed in **C3** (§3.3.2). However, such features are commonly used in state-of-the-art LiDAR-based AD perception as summarized in Table 3.1, for achieving high run-time performance (§3.2.1). This thus makes it necessary to address this to ensure the generality of our attack method.

To address this, we find that as long as we can obtain the point-inclusion value of each 3D point to a given cell, all the commonly-used features in Table 3.1 can be mathematically calculated in closed form. Thus, we first design an accurate and differentiable approximation of the point-inclusion property calculation, or a *soft* point-inclusion calculation, and then use it as a *building block* to differentiably derive the features.

Building block: Soft point-inclusion calculation. Given a point $\mathbf{PC}_i$ with coordinate (u_i, v_i, w_i) from the point cloud PC and a 3D cell c_m of length L, width W, and height H, the direct point-inclusion value of $\mathbf{PC}_i$ for c_m, denoted as $\text{PI}(\mathbf{PC}_i, c_m)$, is 1 if $\mathbf{PC}_i$ is inside c_m,



(a) 8 cells & formed cube (b) Soft point-inclusion calc. (c) Result assigned to 8 cells

Figure 3.4: Illustration of the soft point-inclusion calculation with trilinear and tanh approximations. $\mathbf{PC}_i$ is a point in PC, and c_1 to c_8 are the 8 3D cells closest to $\mathbf{PC}_i$.

and 0 if not. To differentiably approximate this function, we estimate the *point-inclusion probability* of the point among the 8 cells closest to it by calculating the interpolation of it to these 8 cell center positions. Fig. 3.4 (a) illustrates these 8 cells, which are indexed as $m = 1...8$. The center position of a cell c_m is denoted as (u_m, v_m, w_m). These 8 center positions form a cuboid that encloses $\mathbf{PC}_i$. We can then calculate the interpolation of this point to these center positions using trilinear interpolation [28]:

$$\text{softPI}(\mathbf{PC}_i, c_m) = (1 - \frac{d(u_m, u_i)}{L}) \cdot (1 - \frac{d(v_m, v_i)}{W}) \cdot (1 - \frac{d(w_m, w_i)}{H}) \tag{3.7}$$

where $d(u_1, u_2) = |u_1 - u_2|$ and $\sum_{m=1}^{8} \text{softPI}(\text{PC}_i, c_m) = 1$. Thus, this is similar to calculating the probabilities of whether PC$_i$ is inside each of these 8 cells. Fig. 3.4 (b) illustrates the calculation process and the example calculation for $\mathbf{PC}_i$ at $(0.8, 0.7, 0.1)$ when $L = W = H = 1$ (i.e., each cell is a cube) and the center coordinate of c_5 is the origin $(0, 0, 0)$. The calculation results are the numbers without underline at the 8 center positions. In Fig. 3.4 (c), the interpolation value at the center position of each cell is then used as the point-inclusion probability for such cell. As shown, since $\mathbf{PC}_i$ is inside c_7, it is the closest to the center of c_7 at $(1, 1, 0)$, and thus the interpolation value is the highest for c_7. This thus is able to correctly assign the highest point-inclusion probability to c_7.

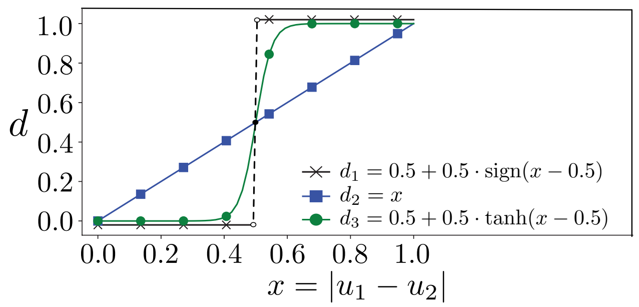


Figure 3.5: Line d_1 is the ground truth, while d_2 and d_3 are trilinear and tanh approximation. As shown, the tanh one is much closer to the ground truth.

Approximation accuracy improvement. In Fig. 3.4 (b), while the point-inclusion probability is indeed the highest for c_7, the probability value is only 0.504 and thus still has a non-negligible gap to the ground-truth value 1. We find that the cause of this gap is at the $d(u_1, u_2)$ function in Eq. (3.7). As shown in Fig. 3.5, the ground-truth function for $d(u_1, u_2)$ when $L = W = H = 1$ is $0.5 + 0.5 \cdot \mathrm{sign}(|u_1 - u_2| - 0.5)$, since if the distance between the point and the cell center at any dimension is over 0.5, it is outside of cell and thus $(1 - d(u_1, u_2))$ should be 0 in Eq. (3.7). Since $\mathrm{sign}(x)$ is not differentiable when $x = 0$, such ground-truth function cannot be directly used in softPI($\cdot$). Using $d(u_1, u_2) = |u_1 - u_2|$ as in classic trilinear interpolation is differentiable, but its curve has a gap to the ground truth as shown in Fig. 3.5 so that it is more difficult for the optimized S^a to succeed. To address this, we use tanh($\cdot$) to differentiably and accurately approximate sign($\cdot$). For example, for the u dimension, it becomes:

$$d(u_1, u_2) = \frac{L}{2} + \frac{L}{2} \cdot \tanh(\mu \cdot (\mid u_1 - u_2 \mid - \frac{L}{2})) \tag{3.8}$$

For the v and w dimensions of $\mathbf{PC}_i$ we replace L with W and H. Fig. 3.5 shows the curve of Eq. (3.8) when $L = 1$. As shown, the difference between Eq. (3.8) approximation and the ground truth is much smaller. In this paper, we call SoftPI($\cdot$) using $d(u_1, u_2) = |u_1 - u_2|$ and Eq. (3.8) *trilinear* and *tanh approximation* respectively. The numbers with underline in Fig. 3.4 (b) and (c) are the results with tanh approximation. As shown, with tanh approximation the point-inclusion probability for c_7 becomes 1.0, which is directly the ground-truth

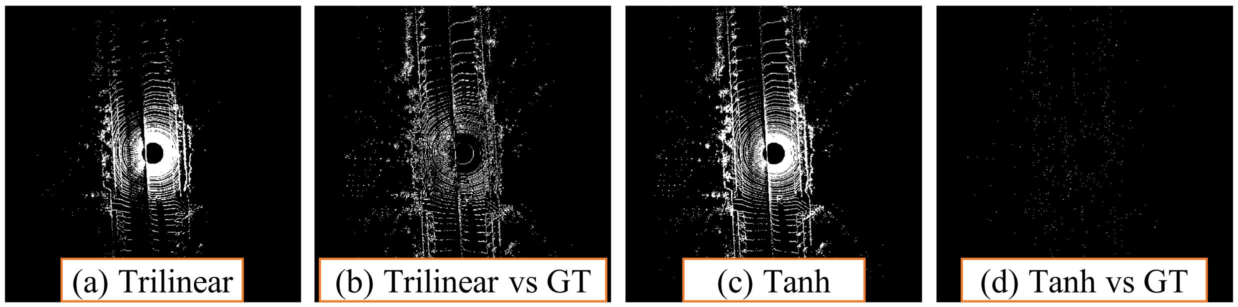


Figure 3.6: Accuracy benefit of tanh approximation over trilinear approximation for the count feature (number of points per cell). Count values are visualized using a gray-scale heatmap. GT denotes the ground-truth count value.

value and thus much more accurate than trilinear approximation.

To more concretely show the benefit of tanh approximation, Fig. 3.6 shows the calculation results for the count feature in Table 3.1 based on softPI$(\cdot)$ using real-world point cloud data. The count feature calculates the number of points in a cell (derivation of it from softPI$(\cdot)$ is described later). In Fig. 3.6, the count values are visualized using a gray-scale heatmap in BEV. Fig. 3.6 (a) and (c) shows the count values calculated using trilinear and tanh approximations respectively, and (b) and (d) shows their differences to the ground-truth count value. As shown, the count values using trilinear approximation have clear differences to the groundtruth, while the differences for the ones using tanh approximation is almost invisible.

Derivation of cell-level aggregated features. With an accurate SoftPI$(\cdot)$, we can then differentiably approximate all the cell-level aggregated features in Table 3.1 as follows:

•*Count and density.* The count feature calculates the number of points in a cell. With softPI$(\cdot)$, we can differentiably derive the count value as $\mathrm{CNT}(c_m) = \sum_{\mathbf{PC}_i \in \mathrm{PC}} \mathrm{softPI}(\mathbf{PC}_i, c_m)$. The density feature calculates the density of points in a cell. Thus, we can directly calculate it by dividing CNT$(\cdot)$ by the cell size.

•*Occupancy.* The occupancy feature calculates whether a cell has points or not. With CNT$(\cdot)$

above, it can be calculated as sign(CNT($\cdot$)). Note that since the sign($\cdot$) is not differentiable, we approximate it using $\text{sign}(x) = x$ during the backward pass of the optimization.

$\bullet$*Height and intensity.* The max/min/mean height features calculate the maximum, minimum, and the average height of the points inside a cell. Thus, the max and min height features are directly $\max_{\mathbf{PC}_i \in \text{PC}} \text{softPI}(\mathbf{PC}_i, c_m) \cdot w_i$ and $\min_{\mathbf{PC}_i \in \text{PC}} \text{softPI}(\mathbf{PC}_i, c_m) \cdot w_i$. The mean height feature can be calculated as $\frac{\sum_{\mathbf{PC}_i \in \text{PC}} \text{softPI}(\mathbf{PC}_i, c_m) \cdot w_i}{\text{CNT}(c_m) + \epsilon}$, where ϵ is small number to prevent division by zero. The max/min/mean intensity features can be calculated similarly by replacing w_i with the intensity value of $\mathbf{PC}_i$.

The calculations above are performed for 3D cells. To obtain features for 2D cells, we just need to add an aggregation of these 3D cell features in one dimension, e.g., the vertical dimension for BEV 2D cells (§3.2.1), into these calculations.

3.4.5 Objective Function Design

Adversarial loss $\mathcal{L}_a$. For the adversarial loss $\mathcal{L}_a$ in Eq. (3.1), similar to prior attacks on object detection [110, 329], we extract and minimize the confidence value (which reflects the confidence that the region contains an object) of the regions of S^a. As introduced in §3.2.1, the fusion process of the LiDAR and camera perception networks in the MSF algorithm can be DNN-based or rule-based. For the former, we directly extract the confidence values in the MSF output [86, 304, 115, 180, 179, 102, 163]. For the latter, since the rule-based fusion logic is not directly differentiable, we extract the confidence values in the outputs of the LiDAR and camera perception networks separately, and minimize the sum of them. This is because if we can prevent S^a from being detected in the outputs of both the LiDAR and camera perception networks, S^a will not appear in the MSF output no matter what the rule-based logic is.

Realizability loss $\mathcal{L}_r(\cdot)$. To realize our attack goal in §3.3.1, S^a needs to be 3D-printed and placed on top of the road surface in the physical world. To facilitate this, we design the realizability loss $\mathcal{L}_r(\cdot)$ in our objective function to (1) improve the printability of S^a at 3D printers by maximizing its surface smoothness using a Laplacian loss [176], and (2) prevent the generation of S^a that is underneath the road surface. The detailed loss formulations are in Appendix 3.A.

Stealthiness designs. Our optimization process has two designs for improving the stealthiness of S^a. First, the realizability loss above can improve its surface smoothness, which can thus allow it to look normally in practice. Second, we solve Eq. (3.6) by using Project Gradient Descent (PGD) with L_∞ distance constraint during the gradient update step in Fig. 3.3, which thus ensures that the per-dimension moving distance for each vertex in S is smaller than ϵ. We can then use ϵ to control how similar S^a looks compared to the benign one S, and thus the smaller ϵ is, the stealthier S^a is.

Attack robustness improvement. To achieve the end-to-end attack success in our setting, it is ideal if S^a can be continuously undetected by the MSF algorithm when the victim AV is approaching the object, until their distance is smaller than the brake distance [11] so that it is too late to brake to avoid the crash. Thus, we need to improve the robustness of S^a against different victim approaching distances and angles of the target road. To achieve this, we implement Transformation T via random yaw-dimension rotations and ground-plane position shifting of S^a, which is illustrated in Fig. 3.3.

3.5 Attack Evaluation

3.5.1 Evaluation Methodology and Setup

MSF algorithm selection. In our evaluation, we target MSF algorithms included in open-source industry-grade AD systems to ensure high practicality and realism of our evaluation results. In particular, we select the ones included in 2 open-source full-stack AD systems, Baidu Apollo [53] and Autoware.AI [7], due to their (1) *representativeness* among industry-grade AD systems today, as Apollo has been recently ranked among the top 4 leading industrial AD developers along with Waymo, Ford, and Cruise [22], and Autoware is adopted by the USDOT in their AV fleet [12]; (2) *practicality*, since both systems can be readily installed on real vehicle models [6, 10] for driving on public roads. In particular, Apollo has been providing self-driving taxi services in China for months [9]; and (3) *ease to experiment with*, since they are the only full-stack AD systems that are open-sourced.

Both AD systems use rule-based fusion in their MSF algorithms, i.e., the LiDAR and camera perception networks are separated DNN models, and their individual perception outputs are fused based on hard-coded matching and prioritization rules. As described in §3.2.1, such design has high modularity and is easy to debug, interpret, and hard-code safety rules/measures [316]. These can greatly benefit system development in industry, which might be the reasons why it is adopted in both Apollo and Autoware.AI. As described in §3.4.5, for such fusion type, our optimization objective is to make our adversarial object undetected in both the outputs of the LiDAR and camera perception models to allow our attack to succeed no matter what rule-based fusion logic is used.

Due to such modular fusion designs, the MSF algorithms in both Apollo and Autoware.AI allow different combinations of LiDAR and camera perception models. Thus, in our evaluation we also evaluate our attack against different such combinations to understand the generality of our attack. In this paper, we call each such combination an *MSF combination* and use $\bigoplus$ to denote such combination operation. In particular, we select 2 different models for LiDAR and 2 for camera, which forms 4 MSF combinations in total. On the LiDAR side, the LiDAR perception model in Apollo is also included in Autoware.AI. Thus, we choose 2 models in different Apollo versions that have substantially different DNN designs: one from the latest version, v5.5, denoted as *A5-L*, and another from an older version, v2.5, denoted as *A2-L*. At the DNN design level, *A5-L* differs greatly from *A2-L* with 43.9% more deep layers and 65.0% more trainable parameters. On the camera side, we select the one from the latest version of Apollo, denoted as *A5-C*, and the pre-trained YOLO v3 [35], denote it as *Y3*, which is included in the latest version of Autoware.AI.

3D object type selection. Considering the supported object types for the LiDAR and camera models, we experiment with 3 types of objects for the above 4 MSF combinations: (1) a *traffic cone* of size 0.5 m $\times$ 0.5 m $\times$ 1.0 m, for A5-L$\bigoplus$A5-C and A2-L$\bigoplus$A5-C, (2) a *bench* of size 0.6 m $\times$ 0.5 m $\times$ 1.5 m, for A5-L$\bigoplus$Y3 and A2-L$\bigoplus$Y3, and (3) a *toy car* of size 0.6 m $\times$ 0.7 m $\times$ 1.6 m (for kids to sit inside), for all 4 MSF combinations. We intentionally avoid large objects like cars since they are much harder to 3D-print and deploy. Among the 3 object types, we consider traffic cone as the most attractive for attacker since it is much more common to appear on the roadway than the other two and thus the most stealthy. Thus, majority of our experiments are focused on traffic cone.

Attack scenario selection. For each object type, we select 100 real-world driving scenarios from the KITTI dataset [119] in which such object in benign case can be 100% detected by the MSF combinations. Each scenario is one frame of sensor inputs including the camera image, the LiDAR point cloud, and the calibration matrix. These scenarios has high diversity with different types of objects (e.g., cars, trucks, traffic lights) and roads (e.g., local, high-way, to rural roads).

Object placement. For most experiments, we place the benign and adversarial objects 7 meters (m) in front of the victim. We choose 7 m because it is the braking distance [11] when the vehicle speed is 25 mph, almost the lowest one in normal driving. Since such distance is larger for higher vehicle speeds, 7 m represents the *smallest* distance at which the object has to be detected by the victim to avoid a crash in normal driving scenarios. In §3.5.4, we also evaluate our attack among different victim distances and angles. More detailed attack parameter settings are in Table 3.A.1 in Appendix.

3.5.2 Attack Effectiveness

In this section, we evaluate the effectiveness of our attack on the attack scenarios described in §3.5.1.

Evaluation metrics. Given an MSF combination and an attack scenario, we render our generated 3D adversarial object into the background point cloud and image, and test whether it can be detected by the MSF combination. We determine our attack as success if and only if the adversarial 3D object is undetected by *both* the LiDAR and camera models in such MSF combination. Under this criterion, the successful attacks can generally defeat *any* rule-based fusion logic that can be applied to fuse the outputs of these two models. Thus, the calculated success rate is a *lower bound* when a specific fusion logic is used, e.g., the ones in Apollo and Autoware.AI. We perform evaluation on 100 scenarios and report success rate.

Results. The results for the 4 MSF combinations are shown in Table 3.2. For all object types and all MSF combinations, success rates are at least 91%, and those for traffic cone and bench are all 100% among 100 driving scenarios. This shows that MSF-ADV is an effective method. Among these results, the 100% success rates for traffic cone is especially important, since it is the most attractive object type among the three from the attacker's view due to its small size and the ability to disguise as a normal traffic object in the middle of the road.

Table 3.2: Attack success rate and average vertex perturbation distance of MSF-ADV on different MSF combinations in 100 driving scenarios. A5-L, A5-C: LiDAR and camera models in Baidu Apollo v5.5. A2-L: LiDAR model in Apollo v2.5. Y3: YOLO v3. All objects can be 100% detected by each MSF combination in the benign case.

MSF Comb.		A5-L$\bigoplus$A5-C		A5-L$\bigoplus$Y3		A2-L$\bigoplus$A5-C		A2-L$\bigoplus$Y3	
Object Type		Traffic cone	Toy car	Bench	Toy car	Traffic cone	Toy car	Bench	Toy car
Success Rate		100%	91%	100%	93%	100%	96%	100%	97%
Dist. (cm)	$\Delta\ell_1$	5.92	5.95	5.93	5.97	5.93	5.63	5.90	5.61
	$\Delta\ell_2$	3.28	3.46	3.39	3.37	3.43	3.34	3.30	3.25
	$\Delta\ell_\infty$	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
LPIPS		0.06	0.02	0.20	0.04	0.07	0.17	0.20	0.06

Note that our method can achieve 91% attack success rates even for the toy car of which the object type (car) has been heavily explored in training data of the model. Among the 4 MSF combinations, A5-L$\bigoplus$A5-C has the lowest attack success rates, which shows that the models from the latest version of Apollo are the most robust among the 4.

Stealthiness. We also measure the stealthiness of our object using the average per-vertex $\Delta\ell_p$ distances and the LPIPS (Learned Perceptual Image Patch Similarity) metric [321]. Table 3.2 shows the results with stealthiness parameter $\epsilon = 2$ cm (§3.4.5). As shown, our attack only needs to move each vertex by 3.4 cm on average ($\Delta\ell_2$) to achieve at least 91% success rates on all MSF combinations. For LPIPS, we use the official implementation from [321] to measure the LPIPS value between the driving image with benign object rendered and the same image with the adversarial one rendered at the same location. As shown, the average LPIPS value is 0.10 across the 3 object types. This is at the same level as those achieved in latest GAN-based image restoring methods [154], which are generally considered as indistinguishable for human. In Table 3.3, we further evaluate our attack under different ϵ values on A5-L$\bigoplus$A5-C using traffic cone. As shown, the attack success rates are still over 93% even when the average moved distance per vertex ($\Delta\ell_2$) is as small as 1.5 cm.

Attack stealthiness user study. To more directly evaluate the attack stealthiness, we

Table 3.3: Stealthiness evaluation results of MSF-ADV on MSF combination A5-L$\bigoplus$A5-C with the traffic cone object under different stealthiness levels of ϵ (§3.4.5).

Stealthiness Level (cm)	Succ. Rate	$\Delta\ell_p$ Dist. (cm)			LPIPS
		$\Delta\ell_1$	$\Delta\ell_2$	$\Delta\ell_\infty$	
$\epsilon = 2.0$	100%	5.92	3.28	2.00	0.06
$\epsilon = 1.0$	93%	2.84	1.51	1.00	0.05
$\epsilon = 0.5$	76%	1.38	0.54	0.50	0.05

also conduct a user study for traffic cone with 105 participants from Amazon Mechanical Turk [4]. The results show that the generated adversarial traffic cone is generally viewed (1) as innocent as the original benign cone, and (2) less suspicious than certain benign ones with broken shapes. More details are in Appendix 3.B.

Effectiveness under different attack settings. We also perform evaluation under different attack parameter settings. We find that our attack is most sensitive to μ, which show that the differentiable approximation design in §3.4.4 is critical to the attack success. More details are in Appendix 3.C.

Printability. We also evaluate the printability of our attack using commercial printability checking tool and geometry metrics such as *watertightness* [271, 16], *self-intersection* [14], and *curvature* [14]. Our results show that our generated objects are 100% printable, and our printability improvement designs in §3.4.5 substantially reduce the printing difficulties from 58.9% to 74.3%. Detailed are in Appendix 3.D.

Transferability. We also evaluate the attack transferability among the 4 MSF combinations with the toy car object. We find that the transfer attack among them is generally effective, with success rates around 75% on average.

Table 3.4: Comparison between MSF-ADV and baseline attack methods in attack success rate and object perturbation degrees. GN: Gaussian noise. GA: genetic algorithm.

Attack Method	Success Rate	$\Delta\ell_p$ Dist. (cm)		
		$\Delta\ell_1$	$\Delta\ell_2$	$\Delta\ell_\infty$
GN	8%	21.8	3.35	10.3
GA	9%	2.85	1.84	2.00
Ours	100%	5.92	3.28	2.00

3.5.3 Comparison with Baseline Attack Methods

While our attack shows high effectiveness in the previous section, it is unclear how much of it is due to the specific designs in MSF-ADV. To understand this, in this section we compare our method with possible baseline attack methods.

Evaluation methodology. We consider 2 baseline attack methods: (1) Gaussian noise based shape perturbation, denoted as *GN*, and (2) Genetic algorithm [208] based attack generation, denoted as *GA*. GN is used to understand whether the success of our attack is due to our optimization-based design (§3.4), or simply due to the nature of that level of shape perturbations. GA still uses our objective function design in §3.4.5 as fitness function, but does not need differentiability, which is thus used to understand whether our differentiable approximation function designs in §3.4.4 are actually useful.

Experimental setup. We perform comparison with our attack on A5-L$\bigoplus$A5-C MSF combination with the traffic cone object using the same setup in §3.5.1. We implement GN and GA using the corresponding standard Python libraries [3, 17]. For GN, we apply a Gaussian noise with $\mu = 0$ and $\sigma = 2.1$ cm to each vertex dimension to generate a similar level of perturbation as MSF-ADV with $\epsilon = 2$ cm. For GA, we set the population size to 50, a common value used in genetic algorithm based adversarial attacks [51, 113]. We configure it to use 2 cm as the per-dimension perturbation bound for each vertex, the same as ϵ in MSF-ADV. To achieve a fair comparison, we run GA using similar CPU and GPU resources

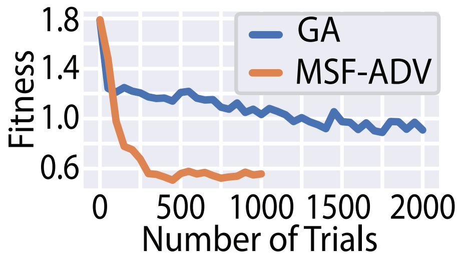


Figure 3.7: Fitness values of GA and MSF-ADV during the optimization process. The curve for MSF-ADV stops at 1000 since it can already succeed for all scenarios at that point.

as MSF-ADV, and ensure that it runs longer than our method.

Result. Table 3.4 summarizes the attack success rates of GN, GA, and our method, and the corresponding shape perturbation degrees. As shown, for GN, the average moved distance per vertex is 3.35 cm ($\Delta\ell_2$), which is larger than those generated by our method (3.28 cm). However, only 8% of the ones from GN succeed, which is a magnitude lower than ours (100%). This thus shows that our high attack effectiveness is mainly due to our optimization-based design, instead of the nature of a similar-level shape perturbation. For GA, we stop it after it generates 2000 adversarial objects for each attack scenario, which is twice the number for our method (1000). However, the success rate is only 9%, which is also a magnitude lower than ours. Fig. 3.7 shows the fitness value trend during the optimization process, which is averaged over the 100 attack scenarios. As shown, the fitness value decrease for GA is much slower than ours: its fitness value drop after 2000 trials is achieved after only 133 trials using our method, which is 15$\times$ more efficient. This thus concretely shows that benefit of our differentiable approximation function designs in §3.4.4, which allows the use of gradient-based optimizations to significantly improve both the attack efficiency and effectiveness.

3.5.4 Attack Robustness

In this section, we evaluate our attack robustness against different victim approaching positions and angles.

Table 3.5: Average success rate on A5-L$\bigoplus$A5-C with traffic cone in different victim approaching distance ranges.

	Y = (-0.1 m, 0.1 m)		
	X = (5 m, 15 m)	(15 m, 25 m)	(25 m, 35 m)
w/o EoT	80.3%	79.2%	79.9%
w/ EoT	96.3%	95.5%	96.6%

Evaluation methodology. We still use the attack scenarios in §3.5.1 for evaluation. To synthesize different relative positions between the victim and the object when the victim is approaching the object, we render the object at different locations ahead of the victim in both the camera and LiDAR frames given an attack scenario.

Experimental setup. As described in our attack design (§3.4.1), the adversarial object is placed in the middle of the traffic lane in which the victim is driving. In this section, X and Y denote the relative distance between the victim and the object in the longitudinal (i.e., forward and backward) and lateral (i.e., left and right) directions respectively. For X, we consider 3 distance ranges from 5 to 35 m, which correspond to the brake distances for speed from ∼20 to 55 mph [11]. For Y, the deviations to the center of the lane usually need to be within 0.1 m for smooth and safe driving [49, 98]. Thus, we consider $Y \in (-0.1 \text{ m}, 0.1 \text{ m})$. For each position range, we randomly sample 20 different positions.

Results. Table 3.5 shows the average attack success rates for A5-L$\bigoplus$A5-C with traffic cone in the 3 position ranges over the 100 evaluation scenarios (§3.5.1). As described in §3.4.5, we use EoT to improve robustness. As shown, this improves the average success rates in all position ranges by 20.5% on average. Overall, with EoT the average attack success rates are over 95% across different position ranges, which shows a high robustness of our attack against different victim approaching positions and angles at common driving speeds.

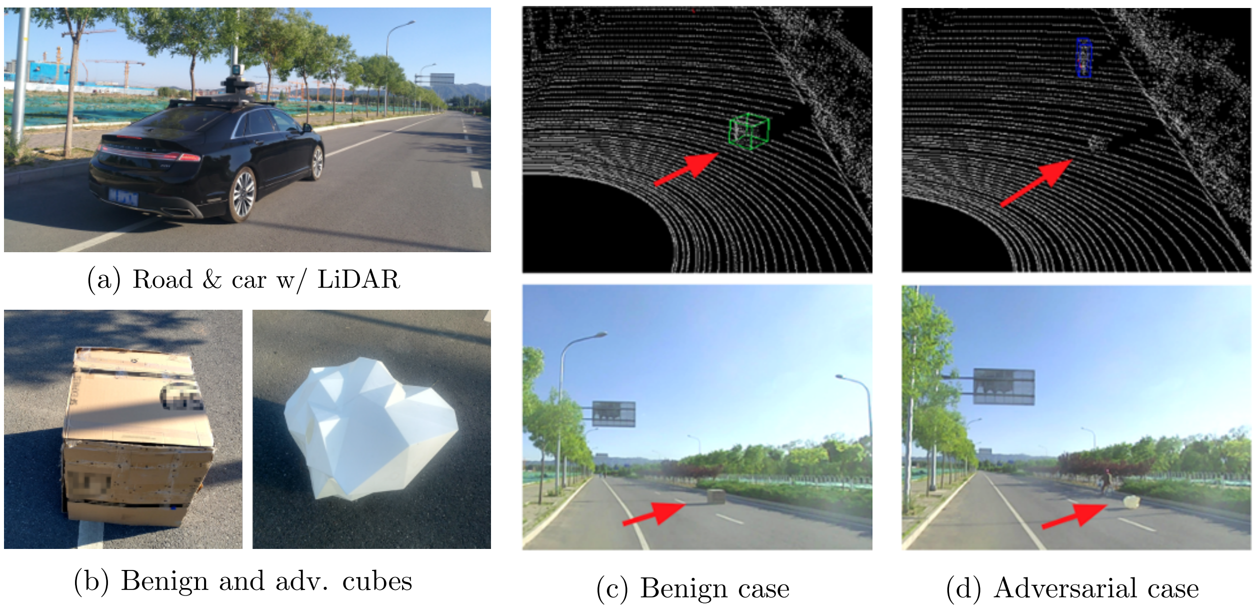
3.5.5 Physical-World Attack Realizability Evaluation

While the results in prior sections show high effectiveness and robustness of our attack, the experiments are performed by digitally rendering the objects into camera and LiDAR inputs. Thus, it is unclear whether such high effectiveness can still be achieved after the adversarial object is 3D-printed and placed in the physical world. Thus, in this section we evaluate such physical-world realizabilty of our attack.

3.5.5.1 Real Vehicle based Experiments

At the early stage of this project, we had access to a real vehicle equipped with a high-end Velodyne HDL-64E LiDAR, and used it to perform physical-world experiments for LiDAR models. Unfortunately, later we lost the access to it and only have such real vehicle based experiments for the LiDAR-side evaluation. In this section, we report these results for LiDAR side, and will detail in the next section the physical-world experiments for both LiDAR and camera using a miniature-scale experiment setup.

Evaluation methodology and setup. In this experiment, we 3D-print the adversarial object and conduct the experiment by using the vehicle mentioned above to collect its LiDAR point clouds on the real road. Fig. 3.8 (a) shows the vehicle and road. We selected a rarely-used road and no other vehicles passed by during this experiment. Since this experiment was performed at the early stage of this project, the selected object type was a 75cm cube, and the targeted model was A2-L, the latest version of the Apollo LiDAR model at that time. Fig. 3.8 (b) shows the box of the same size used as the benign cube, and the 3D-printed adversarial cube. This setup mimics the attack scenario by placing an adversarial rock-shaped object (§3.4.1).


(a) Road & car w/ LiDAR

(b) Benign and adv. cubes

(c) Benign case

(d) Adversarial case

Figure 3.8: Physical-world experiment settings and evaluation results for LiDAR-side physical-world attack realizability. We use a Velodyne HDL-64E LiDAR mounted on a real vehicle. The adversarial cube is 3D-printed at 1:1 scale.

Results. We manually drive the vehicle around the cube and collect traces in front of it and on the left of it. In total, there are 99 LiDAR frames with the benign cube, and A2-L is able to correctly detect it in 84.8% (84) frames. In comparison, we find that the adversarial cube is detected in only *0.9% (1)* of the 108 LiDAR frames including it. Fig. 3.8 (c) and Fig. 3.8 (d) show examples of the frames and detection results for the benign and adversarial cubes respectively. These results show that our attack is still effective in the physical-world setting for the LiDAR side of MSF. Experiment videos and images are at **https://sites.google.com/view/cav-sec/msf-adv**.

3.5.5.2 Miniature-Scale Experiments

Since we lost the access to the experiment vehicle, in this section we design a miniature-scale experiment in our lab environment to perform physical-world experiments for both LiDAR and camera.

Evaluation methodology. In this experiment, we still 3D-print the adversarial object and

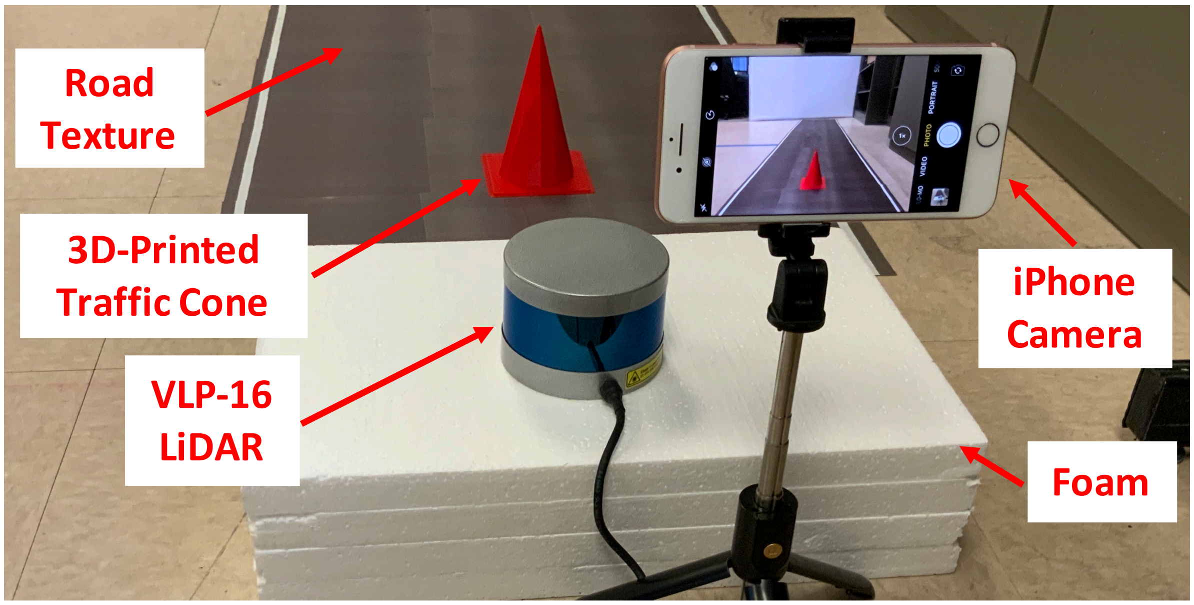


Figure 3.9: Miniature-scale experiment setup with camera and LiDAR. Road and traffic cone are at 1:6.67 scale.

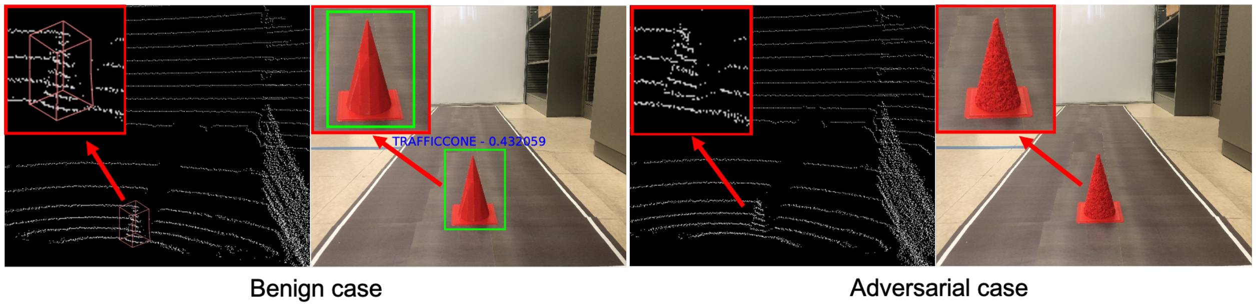


Figure 3.10: Visualization of the LiDAR and camera perception results for A5-L$\bigoplus$A5-C in miniature-scale experiments.

obtain its point clouds and images using physical LiDAR and camera devices like in the actual physical-world attack settings. However, the main difference is that the adversarial object and the road are set up in a *miniature scale* as shown in Fig. 3.9. As shown, the adversarial object is 3D-printed at 1:6.67 scale and placed on a miniature-scale straight road created by printing a real-world high-resolution BEV road texture on multiple A4 papers and concatenating them together. Here, the obtained point clouds of the object and road are scaled up accordingly following the physical rule of LiDAR to obtain the point clouds in real-world scale. The benefit of such miniature-scale setup is that it can not only obtain physical-world point clouds and images following the same physical rules of LiDAR and camera, but also more easily fit into the budget of a university-level research lab (e.g., 3D-

Table 3.6: Evaluation results for A5-L$\bigoplus$A5-C and A2-L$\bigoplus$A5-C at 20 randomly-sampled positions in miniature-scale experiments. Results for A2-L$\bigoplus$A5-C is a transfer attack since the adversarial traffic cone is generated for A5-L$\bigoplus$A5-C.

	A5-L$\bigoplus$A5-C	A2-L$\bigoplus$A5-C (transfer attack)
Benign detection rate	19/20 (95%)	16/20 (80%)
Attack success rate	18/20 (90%)	17/20 (85%)
Attack success rate when benign can be detected	18/19 (94.7%)	14/16 (87.5%)

printing our 1-meter high traffic cone at 1:1 scale requires industry-grade 3D printers [20]).

Experimental setup. We use an iPhone 8 Plus back camera and a Velodyne VLP-16 LiDAR to collect images and point clouds as shown in Fig. 3.9. For the adversarial object, we generate the adversarial traffic cone mesh using the image and point cloud collected in our miniature-scale setup as the background. We 3D-print the benign and adversarial traffic cones with 380 um precision at 1:6.67 scale. The road size, traffic cone size, and the camera and LiDAR positions are chosen to represent the scenario where these sensors are installed on a car driving on a standard 3.6-meter wide highway road [39].

In the experiment, we try 20 different positions on the miniature road, which are randomly sampled in a 6.0 cm $\times$ 6.0 cm area at the road center and $\sim$45 cm far from the camera and LiDAR. We choose this area because we find the highest detection rate of the benign cone in this area. In real-world scale, this represents the scenario where the adversarial cone is roughly at the road center and 3-3.5 m far from the camera and LiDAR on the victim. Since the object type is traffic cone, we consider A5-C on camera side, and the VLP-16 versions of A5-L and A2-L in Apollo, which has the same model architecture as their corresponding HDL-64 versions [5].

Results. Table 3.6 shows the results. As shown, for A5-L$\bigoplus$A5-C, the benign traffic cone can achieve 95% detection rate at the 20 random positions. However, after we place the adversarial one at exactly these 20 positions, the detection rate is only 10%, leading to a

90% success rate. Specifically, at the 19 positions that the benign cone can be successfully detected, the attack success rate is around 95%. Fig. 3.10 visualizes the LiDAR and camera perception results of the benign and adversarial cones. More images and dynamic moving videos are at **https://sites.google.com/view/cav-sec/msf-adv**.

Since this adversarial cone is generated for A5-L$\bigoplus$A5-C, we also evaluate it against A2-L$\bigoplus$A5-C to understand whether such attack effectiveness can transfer. As shown, the success rate of such a transfer attack is very similar: the success rate among the 20 positions is 85%, and that among the positions where the benign cone can be detected is 87.5%. These results thus show that our generated adversarial objects can still be effective against both LiDAR and camera in a physical-world environment, and such effectiveness can transfer.

3.6 End-to-End Attack Simulation Evaluation

To more concretely understand the end-to-end safety consequences, we further evaluate on a concrete attack scenario using a production-grade AD simulator.

Evaluation methodology and metrics. We perform an end-to-end attack evaluation on Baidu Apollo using LGSVL simulator [38]. LGSVL is an open-source Unity-based simulator designed for testing and development of industry-grade AD systems, and has already supported Apollo. In our evaluation, we use a map of a single-lane road in LGSVL, and set up Apollo to control a vehicle to drive along this lane. To launch our attack, we imported the 3D mesh of our adversarial traffic cone into Unity, set its physical properties, and then re-build the simulator and the map. We control the position of this adversarial cone to set it to the lane center, and LGSVL will provide Apollo with the raw camera and LiDAR inputs with the adversarial objects using its simulation engine. As described in §3.4.1, crashing into such an adversarial traffic cone can lead to severe safety damages as the attacker can fill it with denser materials such as granite or metal, or put nails or glass debris behind it. Considering such concrete attack scenarios, we directly use the vehicle collision rate with the adversarial cone to evaluate the attack effectiveness.

Experimental setup. We evaluate on Apollo v5.0, the latest Apollo version supported by LGSVL so far [38]. We use the default camera and LiDAR device configurations in this support. The LiDAR and camera models in Apollo v5.0 are the same as those in the latest version, Apollo v5.5. Thus, we directly use the adversarial traffic cone generated in §3.5 for this evaluation. The vehicle speed is set to 30 km/h. For both benign and adversarial scenarios, we perform 100 runs of experiments and each lasts around 20 seconds to allow the vehicle to arrive at the traffic cone placement position and finish executing the driving decision.

Results and demo videos. The results show that our adversarial traffic cone can *always* fool the Apollo system in the entire trip across the 100 runs, leading to a *100%* vehicle collision rate. We inspect the experiment log and find that the adversarial cone evades both the camera and LiDAR perception pipelines at *every* frames before fusion, which thus fundamentally defeats the basic design assumption of using MSF for defense. In contrast, in the benign case, Apollo is *always* able to correctly detect the benign cone and stop in front of it to avoid collision (i.e., 0% crash rate). Across different runs, the vehicle driving trajectories differ slightly due to the simulation randomness and sensor messaging delay/dropping, but our attack shows a high robustness against such trajectory variances when the victim is approaching.

Fig. 3.11 shows the key screenshots on both the LGSVL and Apollo sides during the simulation. As shown, in the benign case, the victim can detect the traffic cone and successfully make a stop decision to decrease its speed to 0 km/h. However, in the adversarial case, the victim cannot detect the traffic cone even when it is right in front of it. Thus, it maintains the original speed and directly crashes into it. We also record short demo videos from the

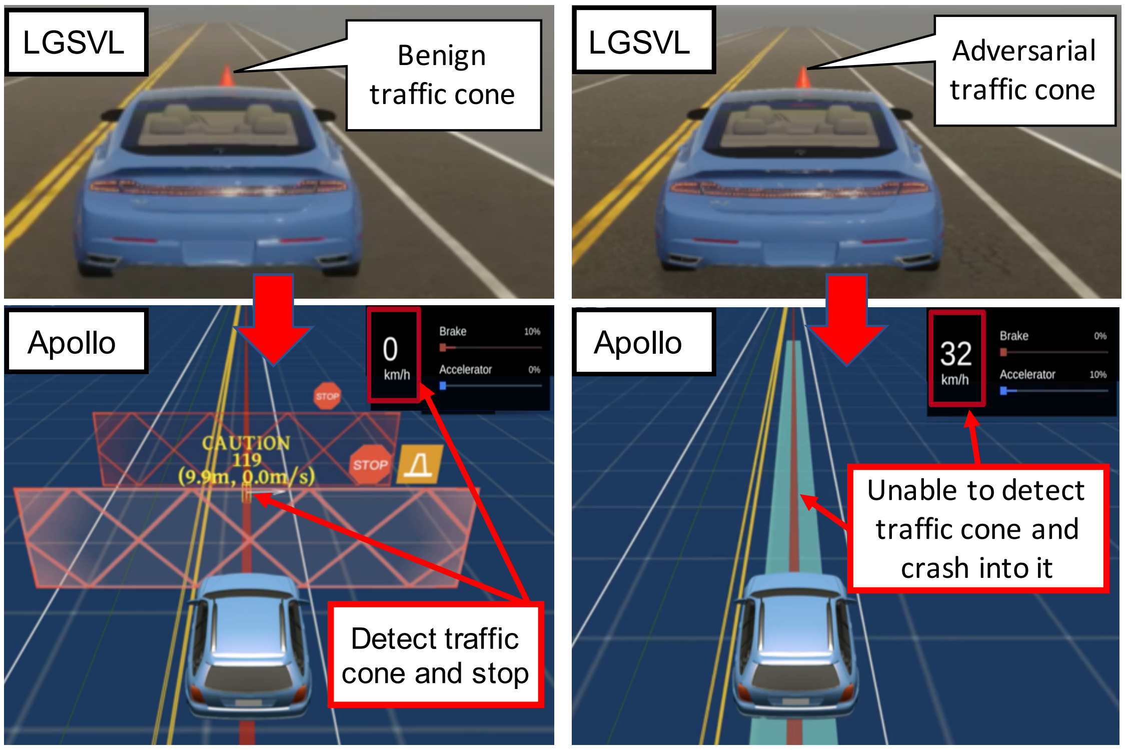


Figure 3.11: Screenshots of Apollo and LGSVL in the end-to-end attack evaluation with benign and adversarial traffic cones. Across 100 runs, the crash rate is 100% for adversarial case, and 0% for benign case.

simulation, available at [23].

3.7 Limitations and Defense Discussion

3.7.1 Limitations of Our Study

End-to-end physical-world evaluation. In this work, our attack is designed with a practical attack model (§3.4.1) and evaluated on real-world driving dataset and miniature-scale physical-world settings (§3.5.5). However, we did not perform an end-to-end attack evaluation on a real AV in the physical world due to the cost and safety considerations. As a best effort, we evaluate such end-to-end attack impacts using a production-grade AD simulator (§3.6). Note that AD companies such as Waymo also heavily rely on simulation-based eval-

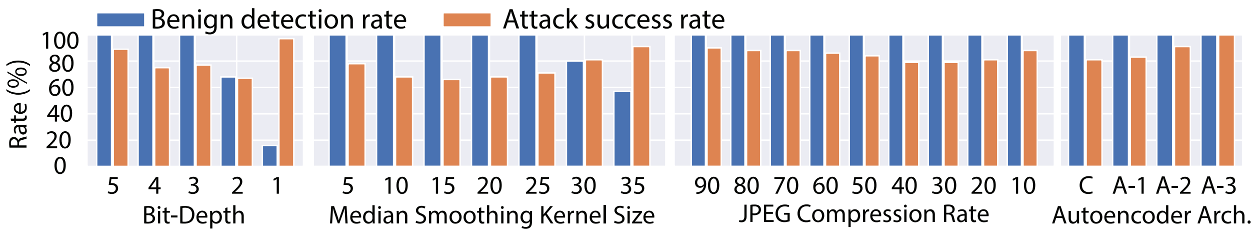


Figure 3.12: Evaluation results of 4 DNN input transformation based defense methods for our attack on A5-L$\bigoplus$A5-C with traffic cone object. Benign detection rates mean detected by either LiDAR or camera. Attack success rate means that both LiDAR and camera fail to detect. For all x-axes, values from left to right mean higher to lower camera/LiDAR input quality (e.g., more smoothing or compression). Detailed setup in Appendix 3.E.

uations when developing and testing AD systems for safety and budget considerations [18].

Attack generality evaluation. In our evaluation, we target the MSF algorithms used in representative industry-grade AD systems such as Baidu Apollo [53], which generally adopt a rule-based fusion design. As introduced in §3.2.1, there also exists other types of fusion design: DNN-based fusion [86, 304, 115, 180, 179, 102, 163] or even context-aware sensor fusion [201], which improve both the efficiency and robustness of the AD systems. Thus, it is still unclear how effective MSF-ADV can be for these MSF algorithms. Note that this is not a limitation of our attack methodology: as described in §3.4.5, our design is generally applicable to different fusion designs. Also, since rule-based fusion design is more preferable for the system development in the industry (§3.5.1), our current evaluation results can potentially lead to more impacts to AD systems in practice. Thus, we left the attack evaluation of other MSF algorithms as future work.

3.7.2 Defense Discussion

3.7.2.1 DNN-Level Defense

Our attack exploits vulnerability in DNNs used in MSF, and thus a direct defense direction is to secure these DNNs. In the recent arms race between adversarial attacks and defenses, various defense/mitigation techniques have been proposed, e.g., input transformation [306, 206, 105], adversarial training [200], and certified robustness [174, 168]. However, almost all of them focus on image classification models under digital-space attacks, instead of object detection models under physical-world attacks. To the best of our knowledge, no prior works has considered defending against adversarial 3D objects in MSF context.

Experiment methodology. In this case, as a best effort to understand the effectiveness of existing defenses in our attack setting, we perform experiments mainly on two easily-adaptable defense strategies: (1) camera/LiDAR input transformation without model re-training, for which we evaluate 4 popular methods: bit-depth reduction [306], median smoothing [306], JPEG compression [105], and autoencoder reformation [206]; and (2) augmenting training data, denoted as AUG, which re-trains the model with adversarial inputs mixed in training dataset [262, 123, 221]. AUG is only applied to YOLO v3 (Y3) since Apollo does not release training dataset for its models. Additionally, we also explored adversarial training [318] for Y3, but different from the standard adversarial training, we only applied *2 steps PGD attack* to approximate the solutions of inner maximal problem for efficiency due to the complexity of our attack pipeline (e.g., rendering, pre-processing, and attacking two models together) caused by our problem settings and evaluated system. Such a strategy has been adopted in some recent works to improve efficiency of adversarial training [291, 318]. More details are in Appendix 3.E. Note that we do not evaluate certified robustness [174, 168] since its designs today focus on small 2D digital-space perturbations (e.g. ℓ_2=0.5 on ImageNet [90, 313]), and their extensions to either 3D space or physical-world attacks are still open research problems.

Results. Fig. 3.12 shows the results for the 4 input transformation based defenses on our attack on A5-L$\bigoplus$A5-C for traffic cone. For each method, we explore different parameters to explore the trade-off between benign detection rate and attack success rate. As shown, with the decrease of the LiDAR/camera input quality (left to right for all the x-axes), the attack success rate will eventually increase for all 4 methods since the input quality becomes so low that both camera and LiDAR models cannot detect the object even in the benign case. For some methods, the attack success rate first decreases before such increase, which is likely because the input quality reduction disrupts our adversarial shape perturbations. Overall, median smoothing achieves the highest defense effectiveness by decreasing the attack success rate to 66% without affecting the benign detection rate. Note that it is known that all these methods can be bypassed by adaptive attacks [74, 323, 134, 250]. Thus, an interesting future work is to explore the effectiveness of these methods under adaptive attack designs of MSF-ADV.

Table 3.7 shows the results for AUG. For a fair comparison, the original model in the table is also newly-trained using the same setup. As shown, AUG is able to decrease the attack success rate to 69% with 100% benign detection rate. Our preliminary exploration of adversarial training with 2-step PGD does not show higher effectiveness: even with 900 epoch of training, the attack success rate is only reduced to 95% with 100% benign detection rate. The potential reason of the lower effectiveness is that 2 steps PGD is not enough to generate effective adversarial objects during training. Compared to some prior works [318, 291], this suggests that our attack poses more challenges in balancing the trade-off between efficiency and effectiveness in adversarial training. We plan to systematically investigate this in the future.

Overall, the most effective defense found in these experiments can only decrease the attack success rate to 66%, which is not quite enough to render this attack vector practically unexploitable. Leveraging the analysis insights, we plan to explore more effective defense designs by exploring (1) other input transformation considering the success of medium smoothing, and (2) more efficient and effective adversarial training designs for our attack. As certified robustness can provide strong theoretical guarantees, we also plan to explore the extensions of it to 3D space and physical-world attacks.

Table 3.7: Results of augmenting training data (AUG) for our attack on A5-L$\bigoplus$Y3 compared to original model for bench object. Detailed setup in Appendix 3.E.

	Y3 test set mAP	Benign det. rate	Attack succ. rate
Original	44%	100%	100%
AUG	32%	100%	69%

3.7.2.2 Fuse More Perception Sources

At MSF algorithm level, one defense direction is to fuse more perception sources, e.g., more cameras/LiDARs sharing an overlapped view but mounted at different positions, assuming that our attack may be more difficult to optimize if the fused camera/LiDAR perception results are from very different viewing angles and positions. Also, we may consider including RADAR into MSF, which is less preferred in state-of-the-art MSF designs (§3.2.1) but may help improve their security. Note that this cannot fundamentally defeat our attack since RADAR point clouds may also be affected by shape manipulations and their state-of-the-art object detection algorithms are still DNN-based [280]. Nevertheless, including RADAR may make it more difficult to attack if the RADAR perception model is more robust. We leave a systematic exploration of these to future work.

3.8 Related Work

Autonomous Driving (AD) system security. Since AD systems heavily rely on sensors, prior works have studied *sensor attacks* in AD context such as spoofing/jamming attacks on camera [310, 213], LiDAR [254, 72], RADAR [310], ultrasonic [310], and IMU [272]. In comparison, these works mainly focus on vulnerabilities at sensor level, while we focus on those at the higher *autonomy software* level, i.e., the "brain" of AD systems. At such level, prior works have studied the security of camera/LiDAR object detection [110, 83, 329, 72, 37] and tracking [149], localization [253], lane detection [240, 242, 178], traffic light detection [263], and end-to-end AD [221, 268]. However, so far *all* of them only consider attacks on camera or LiDAR perception *alone*, while we are the first to study the security of MSF-based AD perception and address the corresponding design challenges (§3.3.2).

Adversarial attacks. Various adversarial attacks have been proposed to generate adversarial attacks in the digital space [123, 76, 220, 209, 262, 221, 301, 271, 296, 169, 298, 227, 300, 302, 299]. In comparison, we focus on physical-world attack vectors. Multiple prior works have designed and evaluated adversarial attacks in the physical world [110, 173, 64, 267, 324, 83, 187, 329, 56]. However, none of them have considered MSF-based AD perception, and as described in §3.3.2, blindly combining their designs cannot directly achieve our goal due to various unique design challenges.

3.9 Conclusion

This paper presents a first study on the security issues of MSF-based AD perception, that challenges the basic design assumption for MSF as a defense strategy in AD context. We design a novel attack method, MSF-ADV, with adversarial 3D object as the attack vector, and address design challenges in non-differentiable target camera and LiDAR sensing systems and non-differentiable computation of cell-level aggregated features for LiDAR. We perform evaluations on MSF algorithms included in industry-grade AD systems using real-world driving scenarios. Our results show that our attack achieves over 90% success rates across different object types and MSF algorithms, while being stealthy, robust, transferable and physical-world realizable. In simulation evaluation, our attack can cause 100% vehicle collision rate. We also evaluate and discuss defenses. Considering the critical role of perception for safe AV driving, we hope that our findings and insights can help the community develop effective defenses in practice.

Appendix 3.A Realizability loss $\mathcal{L}_r(\cdot)$ in §3.4.5

To realize our attack goal in §3.3.1, S^a needs to be 3D-printed and placed on top of the road surface in the physical world. To facilitate this, we design the realizability loss $\mathcal{L}_r(\cdot)$ in our objective function to (1) improve the printability of S^a by 3D printers, and (2) prevent the generation of S^a that is underneath the road surface. Our formulation of $\mathcal{L}_r(\cdot)$ is in Eq. (3.9), where the first and second parts are for achieving (1) and (2) respectively. The first part is a Laplacian loss [176], where V^a is the vertex set of S^a, and for $\boldsymbol{v}_i^a \in V^a$, $\Gamma(\boldsymbol{v}_i^a)$ denotes the set of connected neighboring vertices of $\boldsymbol{v}_i^a$. Since our attack generation is performed by only moving the vertex positions in the benign object S (§3.4.1), there is always a corresponding vertex $\boldsymbol{v}_i$ in the vertex set V of S that $\boldsymbol{v}_i^a$ is moved from. The distance between $\boldsymbol{v}_i^a$ and $\boldsymbol{v}_i$ is denoted as $\Delta\boldsymbol{v} = \boldsymbol{v}_i^a - \boldsymbol{v}_i$. Thus, the first part in Eq. (3.9) penalties the differences between the position change of each vertex in S^a and those of its neighboring vertices. This can thus improve the smoothness of the surface of S^a, which can lower the precision requirements of the 3D printer [29] and thus improve the printability of S^a. We also use a popular mesh simplification method, Quadric Edge Collapse Decimation (QECD), as an optional post-processing step to further improve printability.

In the second part, z_i^a and z_i denotes the height values of $\boldsymbol{v}_i^a$ and $\boldsymbol{v}_i$. This part minimizes the distance between the lowest height among all vertices in S^a and that in S, which thus penalties the moving of the vertices in S^a towards under the road surface. β_1 is a hyper-parameter for this part in Eq. (3.9).

$$\mathcal{L}_r(S^a, S) = \sum_{\boldsymbol{v}_i^a \in V^a} \sum_{\boldsymbol{v}_q^a \in \Gamma(\boldsymbol{v}_i^a)} \left\|\Delta\boldsymbol{v}_i^a - \Delta\boldsymbol{v}_q^a\right\|_2^2 + \beta_1 \cdot \| \min_{\boldsymbol{v}_i^a \in V^a} z_i^a - \min_{\boldsymbol{v}_i \in V} z_i \|_2^2 \tag{3.9}$$

Table 3.A.1: Detailed settings for attack parameters in §3.5.

Parameter	Value
PGD initial point (§3.4.5)	0.01
PGD constraint (§3.4.5)	0.02
Tanh approximation parameter μ (§3.4.4)	100
Preventing division by zero ε (§3.4.4)	10^{-7}
X sample range (§3.4.5)	$(5, 35)$
Y sample range (§3.4.5)	$(-0.3, 0.3)$
yaw sample angles (§3.4.5)	$(-5°, 5°)$
Learning rate (§3.4.5)	0.001
$\mathcal{L}_r(\cdot)$ coefficient λ (§3.4.5)	20
Height loss coefficient β_1 (Appendix 3.A)	0.001
Precision of 3D printer used in §3.5.5	0.38mm

Appendix 3.B Attack Stealthiness User Study

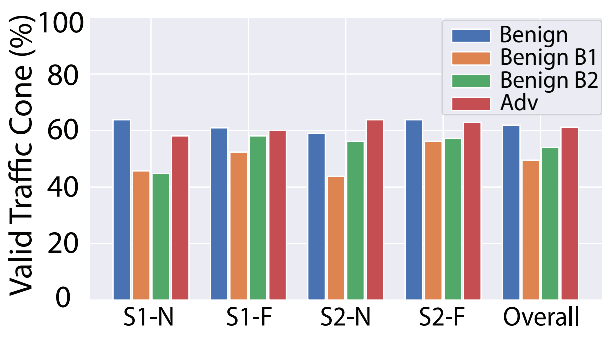
In this section, we conduct a user study to evaluate the stealthiness of the adversarial 3D objects. We go through the IRB process and our study is determined as the IRB Exempt, due to not involving collection of any Personally Identifiable Information (PII) or target any sensitive population.

Evaluation methodology. In this study, we select traffic cone as the evaluation target due to its high attractiveness for the attacker (§3.5.1). We evaluate 4 red traffic cone with different shapes: the benign shape (*Benign*) the adversarial shape generated by MSF-ADV (*Adv*), and two benign but broken shapes similar to Fig. 3.2 (*Benign B1* and *B2*). We consider *Benign B1* and *B2* since our attack is designed to mimic benign traffic objects with a broken look (§3.4.1). We randomly select two images (*S1*, *S2*) from KITTI and render these shapes into these two images at two different positions (near or far way from the victim AV, denoted as *N* or *F*) to generate four realistic driver's scenario (*S1-N*, *S1-F*, *S2-N*, *S2-F*) on the roadway for each shape.

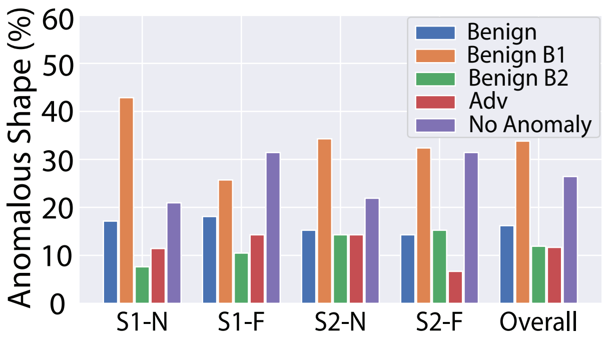
For each of the 4 rendered images above, we ask whether the red traffic object in the image is a valid traffic cone. Note that the driving images are selected by ensuring no extra red object. Images of them are on our website [23]. Among the 4 rendered images, we also ask in which one the red traffic object has the most anomalous shape compared to a normal traffic cone. "No Anomaly" option is included to avoid randomly picking from the participants. To understand the distribution of the participant's background, we also ask for demographic information and background information related to driving.

Evaluation setup. We use Amazon Mechanical Turk [4] to perform the user study. In total, we collected results from 105 participants (55.24% male and 44.76% female) with 35.3 average age. We confirmed that all of them have driving experience by asking them the age when first licensed and the weekly driving mileage. All the benign objects, including *Benign B1* and *B2* can be correctly detected by the latest Apollo MSF combination (A5-L$\bigoplus$A5-C) while *Adv* cannot. The full survey is available at [31].

Results. Fig. 3.B.1 (a) shows the ratio of users thinking that the given traffic cone object is a valid traffic cone. As shown, *Benign* and *Adv* have similar ratios (around 60%) and are higher than *Benign B1* and *B2* since the broken shapes may be more obvious than that of *Adv* after our surface smoothing and PGD-based perturbation bounding (§3.4.5). Note that even for *Benign* there are around 40% users thinking that it is invalid. This might be because the rendered color and shading inevitably have infidelity compared to the real-world background images. Fig. 3.B.1 (b) shows the selection ratios for the cone object with the most anomalous shape. As shown, *Benign B1* and "No Anomaly" are the most popular choices across and *Adv* is always the lowest. The results show that our generated adversarial traffic cone is generally viewed at least as innocent as the original benign cone, and also less suspicious than certain benign ones with broken shapes.



(a) Validity of the traffic cone



(b) Most anomalous shape

Figure 3.B.1: User study results of the attack stealthiness. (a) shows the ratio of users thinking a given object is a valid traffic cone; (b) shows the selection ratios for traffic cones with the most anomalous shape. *S1* and *S2*: 2 different real-world driving images; N and F: near and far rendering positions.

Appendix 3.C Attack Effectiveness under Different Attack Settings

In this section, we perform experiments to understand how sensitive our attack is to different attack parameter settings.

Experimental setup. We target 5 key attack parameters in Table 3.A.1 to perform experiments: μ, λ, β_1, Learning rate, and PGD initial. For each parameter, we experiment with values that are one magnitude higher or lower than the default value. The experiments are performed on A5-L$\bigoplus$A5-C with traffic cone. The results are averaged over 20 attack scenarios randomly selected from the 100 scenarios in §3.5.1.

Results. Table 3.C.1 shows the results. As shown, our attack is most sensitive to μ. Considering that μ is used in our differentiable approximation of the point-inclusion calculation (Eq. (3.8) in §3.4.4), these results show that our differentiable approximation design is critical to the attack success. Different learning rates, λ, and PGD initial are also shown to impact the results, but such impacts are limited to when the values are above or below a certain magnitude.

Table 3.C.1: Attack success rate for A5-L$\bigoplus$A5-C with traffic cone under different attack parameter settings. Descriptions of these attacker parameters are in Table 3.A.1. Gray cells are default settings. KAP: Key attack parameters; ASR: Attack success rate

	μ			λ			β_1			Learning rate			Initialization		
KAP	10	100	1000	2.0	20.0	200.0	0.01	0.001	0.0001	0.01	0.001	0.0001	0.1	0.01	0.001
ASR	75%	100%	60%	100%	100%	65%	100%	100%	100%	75%	100.0%	100%	45%	100.0%	100%

Appendix 3.D Printability Evaluation

To perform our attack, the adversarial objects generated digitally need to be (1) printable by today's 3D printers, and (2) the easier to be printed the better, e.g., requiring less printing precision and thus printable by cheaper 3D printers. In this section, we evaluate such printability of our attack.

Evaluation metrics. To evaluate whether it is *printable or not*, we first use *PreForm*, a commercial printability checking tool [16] that can determine whether their 3D-printing service can print a given 3D mesh. We also leverage the object *watertightness* [271] as another metric, which measures whether the object mesh could hold water if filled. Thus, any 3D object needs to be watertight to have a volume and thus can validly exist (and thus 3D-printed) in physical world. This is the most basic metric for any object meshes to be printable.

For whether the object is *easy to print*, we use the *self-intersection ratio* and *curvature*. *Self-intersection ratio* measures the percentage of the object mesh's 2D faces that have intersections with its other faces. High self-intersection ratio means the mesh need to be printed by a higher precision printer with higher cost. The second metric we use is the *curvature* of the object, which measures how smooth the object surface is. The more smooth the surface is, the less printing precision is required and thus the easier to print. We calculate this metric using the average per-vertex Gaussian curvature value.

Table 3.D.1: Printability evaluation results of MSF-ADV on A5-L$\bigoplus$A5-C with traffic cone. LP: Laplacian loss in Eq. (3.9).

Technique used	Printable?		Easy to Print?		Success Rate
	PreForm	Watertight	Self-intersect	Curvature	
None	100%	100%	88.73%	1.68 ± 1.56	100%
LP	100%	100%	14.43%	0.69 ± 0.65	100%
QECD	100%	100%	38.96%	1.42 ± 1.30	90%
LP + QECD	100%	100%	0.46%	0.67 ± 0.50	92%

Experimental setup. As described in §3.4.5 and Appendix 3.A, our design includes two methods to improve the printability: the Laplacian loss (LP) in $\mathcal{L}_r(\cdot)$ in Eq. (3.9), and QECD [21] as an optional post-processing step. Thus, in our experiment we evaluate the printability of our adversarial objects with and without these two methods. We use the same scenario and parameter settings as §3.5.2.

Results. Table 3.D.1 shows the evaluation results for A5-L$\bigoplus$A5-C using traffic cones. As shown, with or without using any printability improvement methods, the objects generated by our method are all watertight and determined as printable by the PreForm software since our attack method only manipulates the vertex positions of the benign object without changing the original vertex connection relationships.

For the two metrics on whether the object is easy to print, both the self-intersection ratio and the average curvature value are greatly reduced by applying either LP or QECD. LP alone is particularly cost-effective: it is able to substantially reduce the self-intersection ratio by 74.3% and the curvature value by 58.9% without hurting the attack success rate. However, QECD alone hurts self-intersect ratio, curvature value and attack success rate. The decrease of the attack success rate is because QECD is a mesh simplification method that may slightly change the object shape, which thus may interfere with the originally well-optimized shape of the adversarial object. Combining QECD and LP together achieves the highest reduction in both metrics, with only 0.46% self-intersection ratio and 0.67 curvature value. Note that, the average curvature for the benign traffic cone object is 0.72. Thus, both LP alone and the combination achieve a similar level of surface smoothness comparable to a normal real-world object, which thus are printable enough in practice. While the combination reduces the self-intersection ratio compared to LP alone, it incurs 8% success rate decrease due to the use of QECD. Thus, there exists a trade-off. If the attacker does not care about the printing cost, they can choose to use LP alone to better ensure the attack success; otherwise, they can combine it with QECD to reduce the printing costs.

Appendix 3.E Details of the DNN-Level Defenses Evaluation

We describe the details of the defense methods.

Bit-Depth reduction [306]. We follow the setting in prior work [306]. We reduce the bit depth for the image input and the LiDAR point cloud. For a camera image, it consists of RGB channel with 8-bit depth (0-255) for each of them. For a LiDAR point cloud, each point has 4 fields: x, y, z, i, where x, y, and z represents the 3D position, and i is intensity. Each field is a floating point with 32-bit. We use the formulation: $\frac{\text{round}(x*(2^{\text{bit}}-1))}{(2^{\text{bit}}-1)}$ to reduce the bit-depth. In our experiments, we evaluate 5 different bit-depths ranging from 5-bits to 1-bit for both camera and LiDAR inputs. Higher bit-depth number means higher input quality after the bit-depth reduction.

Median smoothing [306]. We follow the setting in prior work [306] and apply the median smoothing to both LiDAR and camera inputs by taking a median around each LiDAR point or camera pixel with a different kernel size. We evaluate 7 different kernel sizes ranging from 5 to 35. Larger the kernel size means higher smoothness and lower input quality.

JPEG compression [105]. We follow the setting in [105] and apply the JPEG only to images since JPEG compression is specific to images. Our attack is successful only when it succeeds for both camera and LiDAR models, therefore, securing the camera model alone is still an effective defense strategy. We use Python Image Library (PIL) [24] to control the compression quality with argument "quality". We use 9 values from 10 to 90 with step 10 to explore the defense effectiveness at different compression rates. Lower values means higher compression rates and thus lower image quality.

Autoencoder reformation [206]. Autoencoder reformation is a part of MagNet defense [206]. We apply it only on image since it is designed for camera-based adversarial examples.

We evaluate 4 different autoencoder architectures, denoted as *C*, *A-1*, *A-2*, and *A-3*. C is the same architecture in the MagNet paper [206] for the CIFAR-10 dataset. Since the input size in our setting is much larger than that in CIFAR-10, we also evaluate 3 other architectures, A-1, A-2, and A-3, by adding 1, 2, 3 average pooling layers to C. From C, to A-3, the latent space dimension size decreases, which thus means more compression and lower input quality. All the autoencoder are trained with real-world images in KITTI dataset [119](§3.5).

Adversarial training (AT) [318]. Since Apollo does not release the training dataset for its models, we can only evaluate this method on Y3 (YOLO v3). Since our attack needs to succeed for both camera and LiDAR, a secure camera model is still an effective defense strategy. We adapt our method to the state-of-the-art adversarial training-based method for camera-based object detection [318]. We follow their algorithm but change the attack in the training loop to ours. Since our attack is performed by adding an object instead of perturbing an existing one, an additional challenge is how to assign ground-truth bounding boxes and labels to our adversarial objects. To address this, we render benign object to the same position and use its detection results as ground-truth results for adversarial one. In AT, we only use bench object to perform experiment.

While adversarial training can be highly robust, it is known to be expensive and nearly intractable for large-scale problems [248, 291]. In our case, this problem further exacerbates as the cost of our attack is higher than 2D digital-space attacks. Thus, we employ an acceleration method found in a recent work [291] that allows a much smaller number of PGD steps (instead of a full optimization cycle) in each training iteration by randomly initializing the adversarial inputs. Specifically, we use a PGD with 2 step and randomly initialize the adversarial mesh during each training iteration. Besides, we train our model from a pre-trained Y3 model, which can converge much faster and also improve robustness [136, 84]. We use the original Y3 training set COCO [13]. We train the model for over 900 epoch, and the model converges after $\sim$83 epoch.

Augmenting training data (AUG) [262, 123, 221]. Prior works show that re-training the model with adversarial inputs mixed in the original training data can improve the model robustness [262, 123, 221]. Same as for adversarial training, this method is only applied to Y3, and we use the same method as in adversarial training to generate the adversarial inputs and their ground-truth bounding boxes and labels. We use same COCO training dataset and the number of training epoch. In this case, the model converges at $\sim$48 epoch.

Chapter 4

ControlLoc: Physical-World Hijacking Attack on Camera-based Perception in Autonomous Driving

4.1 Introduction

Autonomous Driving (AD) vehicles, also known as self-driving cars, are increasingly becoming an integral part of our daily lives [89, 143, 225]. Various companies [166], such as Tesla, are at the forefront of developing AD technologies. To ensure security and safety, AD vehicles such as Tesla employ camera-based perception to detect environmental elements such as traffic signs, pedestrians, and other vehicles in real time. These camera-based perception systems predominantly involve Deep Neural Networks (DNNs) [231, 53, 160] such as object detection, owing to the superior performance of DNNs [94, 58, 142, 48, 184].

Given that failing to detect the objects can lead to violent crashes [319, 258], camera-based perception in AD (referred to as AD perception throughout this paper) in ensuring safety and security has prompted extensive research into exploring its vulnerabilities. For instance, previous studies have highlighted the potential for adversarial attacks, including the use of adversarial patch [289, 148, 330, 283, 111], to fool object detection in AD perception. Such attacks cause the AD systems to ignore objects, posing significant safety risks.

However, it is essential to recognize that AD perception extends beyond object detection to include Multiple Object Tracking (MOT) [53, 160, 150, 252]. MOT plays a pivotal role in AD perception by enhancing robustness against object detection errors. It ensures that only objects detected with consistent and stable accuracy across multiple frames are considered in the tracking results and, consequently, the driving decisions. Specifically, MOT tracks detected objects, estimates their velocities, and generates movement trajectories, called trackers. The tracker management module adds a layer of robustness against detection inaccuracies by not hastily discarding unmatched trackers or instantly creating new ones for newly detected objects. This multi-frame consistency requirement presents a significant challenge to attacks that solely target object detection. For instance, for an adversarial attack on object detection alone to significantly impact the AD perception pipeline, it must achieve at least a 98% success rate across 60 consecutive frames [150], which is infeasible for previous attacks on object detection [289, 330, 148, 283].

Therefore, a digital adversarial hijacking attack [150] to fool the entire AD perception has been proposed with adversarial patches as the attack vector. This hijacking attack necessitates precise control over the position and shape of the object, which is more challenging compared to prior attacks [71, 239, 335, 148] that focus solely on manipulating an object confidence or classification scores. This attack is also powerful since it can achieve a persistent attack effect lasting for dozens of frames with just a few frames of successful attacks. Such a lasting impact is particularly valuable, as existing attacks on object detection alone require consistent success to achieve similar significant attack impacts. Despite this potential, the prior attack [150] has shown limited effectiveness even in the digital space and is

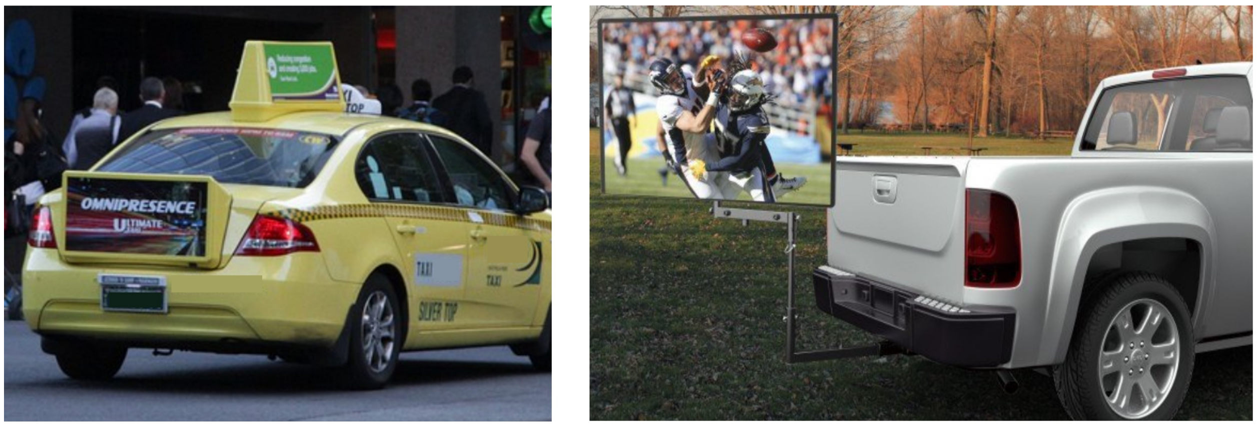


Figure 4.1.1: Illustration of our attack vector in the physical world: advertising monitor, mimicking taxi advertisement (left) or a car with a monitor on its back (right).

ineffective in the physical world. This can be attributed to the lack of precise detection bounding box (BBOX) filtering and the inability to address the coupling between the score and shape of BBOX. The detailed analysis and experiment results of the ineffectiveness for this adversarial hijacking attack [150] in both digital and physical domain are demonstrated in §4.4, §4.5.3, and §4.5.4.

In this paper, we propose the first physical-world adversarial hijacking attack named ControlLoc on the *entire AD perception*. To perform tracker hijacking attacks against MOT, the first step is to shift the target object's BBOX location a certain distance in a specified direction, and then disappear the surrounding BBOXes, as illustrated in Fig. 4.2.2 (c). This places higher demands on the attack capability and requires an attack vector able to display dynamically, as common static patches, such as printed ones, are insufficient to meet these requirements. We employ a monitor as the attack vector, and physical-world experiments show that it performs effectively under diverse lighting conditions. This approach also enhances stealth, as embedding adversarial patches into just a few frames (4-5) of a benign advertisement video makes the attack almost indistinguishable from standard roadside billboards or vehicle advertisements, as depicted in Fig. 4.1.1.

ControlLoc adopts a strategic two-stage approach. In the initial stage, we focus on finding the most effective location for placing the adversarial patch to facilitate successful hijacking attacks. Subsequently, the second stage is to generate the adversarial patch, guided by the optimal locations identified in the preceding phase. This step involves erasing the target object's BBOX from the detection outputs, with a fabricated BBOX of a similar shape in a direction specified by the attacker based on the attack goals and scenarios. This process is designed to simulate movement in a chosen direction, deceiving the AD perception. We propose two loss functions in the second stage, introduced in §4.4.5, aimed at generating the adversarial patch to achieve the attack goal: a score loss, which controls the appearing or disappearing of the bounding boxes, and a regression loss, which is for shape and positioning of fabricated BBOX. Given the inherent challenges arising from the interdependence of these loss functions, we propose a novel optimization strategy, detailed in §4.4.5, which demonstrates superior performance compared to existing methods in prior works [150, 148, 283, 330]. Due to these, ControlLoc can significantly outperform the existing hijacking attack [150].

Our evaluation results demonstrate that ControlLoc achieves outstanding performance across all different AD perceptions, including the combinations between four object detectors and four MOT algorithms with the two attack goals mentioned above. Note that we include the AD perception adapted in open-source industry-grade full-stack AD systems [53, 160]. On two driving datasets, ControlLoc achieves an impressive average attack success rate (ASR) of 98.1%. Furthermore, when compared with a baseline attack [150], the ASR of ControlLoc is quadruple that of the baseline. Additionally, our newly proposed optimization method in this problem domain surpasses the previous method by demonstrating the trend of different loss function values.

To understand the attack effectiveness in the physical world, we further evaluate ControlLoc with a real vehicle, where we put the generated adversarial patch on the rear of the car (and the location is specified by our patch location preselection in the first stage). The results show a 79% average ASR across different outdoor backgrounds, light conditions, hijacking directions, attack goals, angles, and backgrounds while the baseline attack [150] shows ineffectiveness, i.e., 0%? average ASR. To assess how ControlLoc affects the AD behavior, such as collisions or unnecessary emergency stops, we conduct tests using Baidu Apollo [53], an industry-grade full-stack AD system with the LGSVL simulator [234], a production AD simulator with an average effectiveness of 84.4%. We also evaluate various existing directly adaptable DNN-level defenses and discuss future defense directions.

To sum up, we make the following contributions:

- We propose the first practical hijacking attack on AD perception using the monitor as the attack vector, to alter the location and shape of objects. This attack can cause vehicle collisions or unnecessary emergency stops.
- We introduce a novel attack framework, ControlLoc, to generate physical-world adversarial patches. This includes patch location preselection, BBOX filters, loss function designs, and a novel optimization method.
- We evaluate ControlLoc on multiple AD perception systems including industry-grade ones. ControlLoc is effective in the real world with a real vehicle across different backgrounds, outdoor light conditions, hijacking directions, and angles. It causes AD system-level effects like collisions in a production AD simulator.

4.2 Background and Related Work

4.2.1 Camera-based Perception in AD

Camera-based perception in AD critically depends on object detection and multiple object tracking (MOT) to accurately recognize and classify surrounding entities, such as cars. As depicted in Fig. 4.2.1, the process initiates with a series of images. The AD perception algorithm employs an object detector [338] to generate a bounding box (BBOX) and classify the object. Subsequently, the results from object detection, combined with existing tracking data, are input into the MOT [78]. This is tasked with updating the tracking information, such as the BBOX, object velocity, and track identification (track id). Finally, this data is relayed to other downstream modules in AD, such as the planning module [317], which facilitates decision-making processes. Since only the detection results with sufficient consistency and stability across multiple frames can be included in the tracking results, the MOT module can improve the robustness of AD perception.

Object Detection Object detection plays a pivotal role in AD perception, predominantly utilizing Deep Neural Networks (DNN) to identify or categorize various road objects [77]. State-of-the-art DNN-based object detectors are fundamentally divided into two main categories: one-stage and two-stage detectors [338]. One-stage detectors, such as YOLO [156, 231, 230], are renowned for their rapid detection speeds, making them highly suitable for real-time applications such as AD systems. In contrast, two-stage detectors, such as Faster R-CNN[232], are celebrated for their accuracy in detection. Given the real-time requirement of AD systems, industry-grade full-stack AD systems, such as Baidu Apollo [53], predominantly employ one-stage detectors. Furthermore, object detection can be categorized into anchor-based and anchor-free approaches [322]. Anchor-based detection methods leverage a large number of preset anchors, then predict the category and refine the coordinates of these anchors, and finally output these refined anchors as detection results. Conversely, anchor-free detection [269], directly predicts the bounding boxes of objects, offering a more generalizable solution. This paper explores both anchor-based and anchor-free object detection methods.

Multiple Object Tracking (MOT) The current state-of-the-art MOT can be broadly classified into two main approaches [190, 95]: detection-based tracking, also known as tracking-by-detection, and detection-free tracking. The former method employs object detectors to identify objects, which are then used as inputs for MOT, while the latter relies on manually cropped or marked objects as inputs [190]. Tracking-by-detection has emerged as the predominant technique in MOT, particularly within the context of AD [190, 95, 326]. This predominance is attributed to the inherent unpredictability of the number and locations of objects, coupled with the expectation that objects can periodically enter and exit the camera field of view [95]. These conditions render tracking-by-detection algorithms especially well-suited for integration into AD systems [95]. In this paper, we concentrate on the tracking-by-detection paradigm. As illustrated in Fig. 4.2.1, this methodology involves associating the results of object detection at time t with existing trackers from the previous time step ($track|t-1$) and forecasting the current state of the trackers at time t ($track|t$), which includes the velocity and location of every tracked object. To mitigate the impact of false positives and missed detection by the object detectors, MOT modules typically initiate a tracker for an object only after it has been consistently detected across H frames. Similarly, a tracker is removed only after the object has not been detected for R consecutive frames [336, 53, 160, 150]. Consequently, merely compromising the object detection component may not sufficiently disrupt the AD perception [252, 150]. Therefore, this paper introduces a novel and practical physical-world adversarial hijacking attack strategy targeting the entire AD perception including both object detection and MOT.

4.2.2 Attacks on AD Camera-based Perception

Attacks on Object Detection Recent studies have highlighted the vulnerability of DNN models to adversarial examples or attacks, a vulnerability that has been extensively explored [124, 75, 219, 323, 303]. Further investigations have extended these findings to adversarial attack in the physical world [289, 148, 111, 330, 283, 186, 71, 239, 305]. Specifically, within the context of AD, prior research has successfully executed physical-world adversarial attacks targeting camera-based object detection alone [289, 148, 283, 330]. However, the entire AD perception framework encompasses both object detection and MOT. Given the

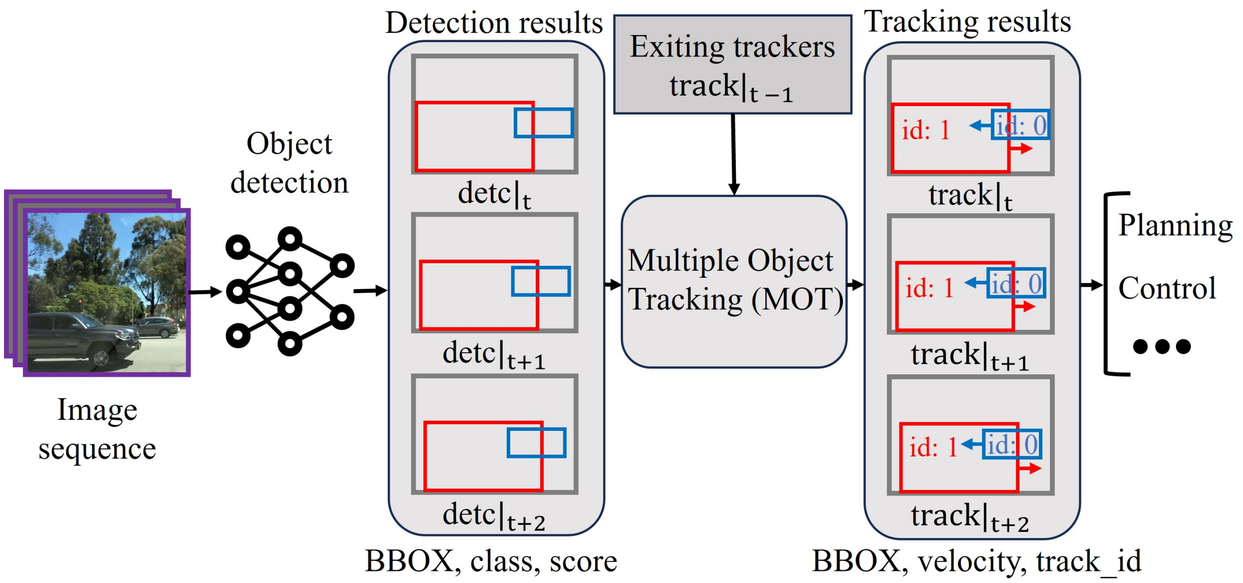
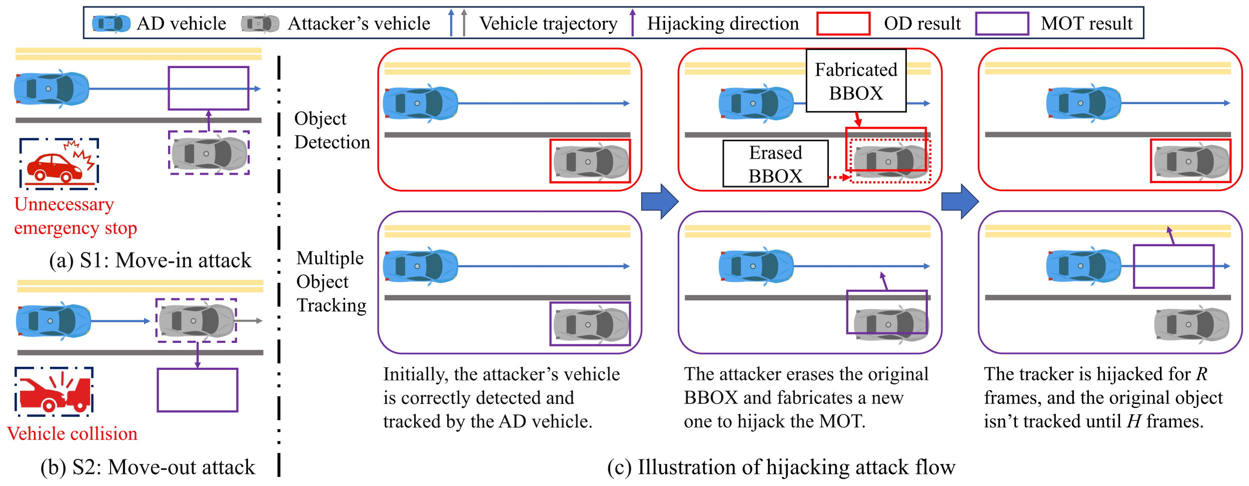


Figure 4.2.1: AD system pipeline: The camera captures images for object detection and multiple object tracking (MOT). Results are sent to the planning and then control modules.



AD vehicle
Attacker's vehicle
Vehicle trajectory
Hijacking direction
OD result
MOT result
Unnecessary emergency stop
(a) S1: Move-in attack
Vehicle collision
(b) S2: Move-out attack
Object Detection
Multiple Object Tracking
Fabricated BBOX
Erased BBOX
Initially, the attacker's vehicle is correctly detected and tracked by the AD vehicle.
The attacker erases the original BBOX and fabricates a new one to hijack the MOT.
The tracker is hijacked for R frames, and the original object isn't tracked until H frames.
(c) Illustration of hijacking attack flow

Figure 4.2.2: Attack goals: (a) move-in attack and (b) move-out attack. (c) shows hijacking attack flow. Rather than aiming for a stable and continuous attack success, ControlLoc achieves a sustained attack effect by successfully attack just a few frames. As shown in the figure, this brief success can have a lasting impact on the MOT, even if the OD recovers to benign detection.

nature of MOT, for an attack targeting only object detection to be effective, it must achieve at least a 98% success rate across 60 consecutive frames—a highly challenging task that for existing object detection attacks to meet [289, 150, 211]. Thereby, this paper proposes a novel, effective, and practical method for the physical-world adversarial hijacking attack on the entire AD perception.

Attacks on Object Tracking Various attacks targeting object tracking have been proposed, spanning both the digital [85, 311] and physical domains [211, 96]. Among them, AttrackZone [211] represents a notable physical domain attack against siamese-based tracking [172, 171], which is a single object tracking (SOT) [328], employing a projector to introduce adversarial perturbations. However, contemporary AD systems employ MOT rather than SOT [53, 160, 81, 170, 252] due to the requirement to identify and track multiple objects simultaneously [170]. Given the limited application of SOT in AD contexts, the end-to-end impact of AttrackZone on AD vehicles is uncertain, casting doubts on its practical value. In contrast, we introduce a novel attack against the realistic entire AD perception, i.e., object detection plus MOT, leveraging an adversarial patch effective in different light conditions, angles, and backgrounds with a novel attack optimization method.

Attacks on the Entire AD Perception Attacks targeting the entire AD perception, such as on availability [193] and integrity [150], have been documented in the literature. Notably, Jia et al. [150] introduce a digital adversarial hijacking attack for the entire AD perception, encompassing object detection and MOT. Despite the innovative approach, the effectiveness of their attack is fundamentally limited as shown in §4.5.3 even in the digital space not to mention in the physical domain. In contrast, our work presents a physical-world hijacking attack that not only breaks the AD perception but also enhances effectiveness compared to the prior attack.

4.3 Attack Goal and Threat Model

Attack Goal. In this paper, we primarily focus on attack goals with significant safety implications for AD, such as vehicle collisions or unnecessary emergency stops [276]. We specifically explore physical-world attack vectors within the AD landscape, employing the adversarial patch due to its high practicality and realism [330, 283, 111, 335, 203]. Our research outlines two main hijacking attack goals: the move-in attack and the move-out attack, shown in Fig. 4.2.2 (a) and (b), respectively. The move-in attack is designed to deceive the victim AD vehicle into an unnecessary emergency stop by inducing a false perception of an object on its current trajectory. On the other hand, the move-out attack manipulates the AD system to overlook actual obstacles by altering the perceived location of these obstacles to the roadside, thereby leading the vehicle into a collision. These tactics aim to demonstrate the potential for adversarial interventions to disrupt the safety and operational integrity of AD systems.

Threat Model. To achieve the attack goals outlined above, this paper delves into a white-box threat model for AD perception consists of object detection and MOT. This threat model assumes that the attacker possesses detailed knowledge of the target object detection, including its architecture and parameters, a promising threat model that aligns with the ones in the existing literature on adversarial vulnerabilities of AD perception [150, 148, 71, 239, 211, 330, 111]. For MOT, the threat model only assume the attacker knows the key parameter settings, i.e., under what conditions the tracking results and detection results are successfully associated, without needing to know other MOT algorithms, such as which Kalman filter is used, etc. This enables our attack to exhibit transferability across various MOT algorithms [326, 53, 160, 46, 103] To improve the attack effectiveness, especially in the real world, we assume that the attacker can collect videos of a targeted road where she plans to launch the attack [71, 239] for attack preparation. To effectively attack AD perception, we employ a monitor as an attack vector, a method with significant potential for dynamically displaying adversarial patches, despite being rarely explored in the context of physical-world adversarial attacks. From the perspective of attack mechanisms, this endows attackers with enhanced capabilities by enabling the display of patches with varying attack effects, particularly in scenarios involving MOT, where continuous video frames serve as input.

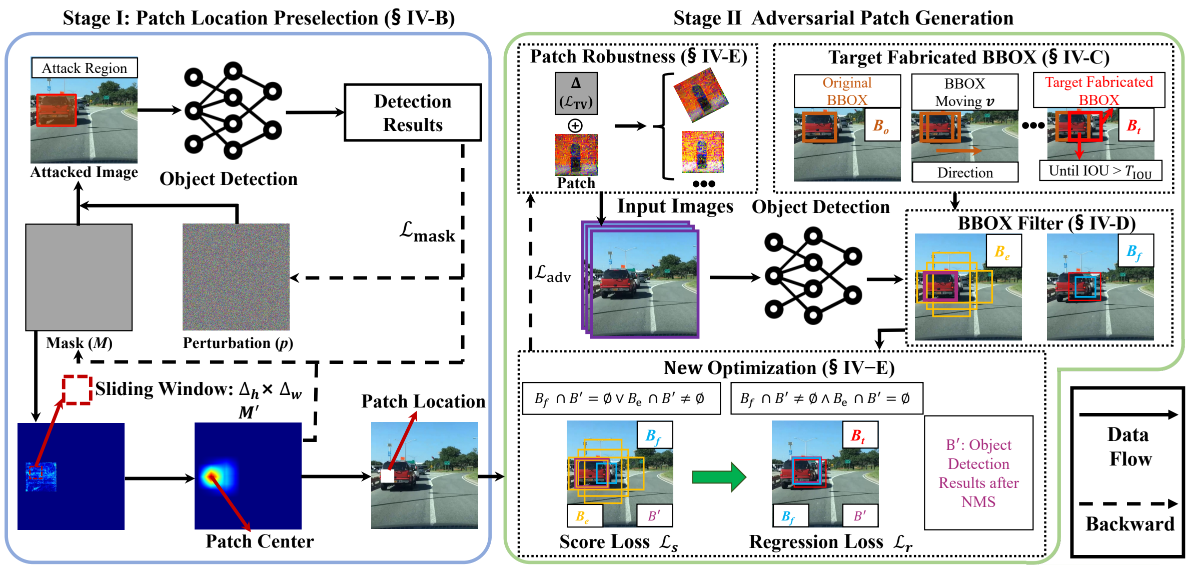

Stage I: Patch Location Preselection (§ IV-B)
Attack Region
Attacked Image
Object Detection
Detection Results
$\mathcal{L}_{\text{mask}}$
Mask (M)
Perturbation (p)
Sliding Window: $\Delta_h \times \Delta_w$
M'
Patch Location
Patch Center
Stage II Adversarial Patch Generation
Patch Robustness (§ IV-E)
Δ ($\mathcal{L}_{\text{TV}}$)
Patch
Target Fabricated BBOX (§ IV-C)
Original BBOX
BBOX Moving v
Direction
Target Fabricated BBOX
Until IOU > T_{IOU}
B_o
B_t
Input Images
Object Detection
$\mathcal{L}_{\text{adv}}$
BBOX Filter (§ IV-D)
B_e
B_f
New Optimization (§ IV−E)
$B_f \cap B' = \emptyset \vee B_e \cap B' \neq \emptyset$
$B_f \cap B' \neq \emptyset \wedge B_e \cap B' = \emptyset$
B'
B′: Object Detection Results after NMS
Score Loss $\mathcal{L}_s$
Regression Loss $\mathcal{L}_r$
Data Flow
Backward

Figure 4.4.1: Overview of our ControlLoc, a two-stage hijacking attack via adversarial patches.

4.4 ControlLoc Attack Methodology

4.4.1 Attack Design Overview

We provide a detailed overview of our ControlLoc. This hijacking attack flow is illustrated in Fig. 4.2.2 (c). As depicted, the process begins with the AD perception system correctly detecting and tracking the object. When the vehicle enters the effective attack range, ControlLoc removes the bounding box (BBOX) of the target object from the detection results and fabricates a similar-shaped BBOX, which is slightly shifted with an attacker-desired direction. This fabricated BBOX is then associated with the original tracker of the target object, effectively hijacking the tracker. Although the tracker hijacking typically lasts for only a few frames, its adversarial effects can persist longer, depending on the design of the MOT, particularly the common H and R shown in Fig. 4.2.2 (c) and introduced in §4.2. To achieve the above attack strategy, we propose a dual-stage attack method, of which overview is in Fig. 4.4.1.

Stage I: This stage shown in Fig. 4.4.1 is an optimization-based approach to preselect the patch location. The details of this part will be introduced in §4.4.2. This strategy leverages masks and adversarial perturbations to identify areas that are most conducive to successful attack execution. These areas are then further refined based on potential patch placement locations, such as the rear of the vehicle. Subsequently, a sliding window is utilized to precisely obtain the optimal location. This process (Stage I) can be a pre-processing step to enhance the efficiency and effectiveness of attack generation.

Stage II: This stage as shown in Fig. 4.4.1 can be divided into several distinct steps, outlined below, focusing on generating a physical-world adversarial patch for hijacking attacks.

Step 1: Finding Target Fabricated Bounding Box. In Fig. 4.4.1, an iterative process is employed to find the target fabricated BBOX based on the Intersection over Union (IOU) value between the candidate and the original BBOX. The key insight is that the fabricated BBOX should closely match the original BBOX, but with a shift as large as possible towards the attack direction. The details are outlined in §4.4.3.

Step 2: Bounding Box Filter. In DNN-based object detection, many proposed BBOXes are irrelevant for attack generation, often identifying background elements or unrelated objects. To ensure the effective generation of the patch, it is crucial to filter the relevant BBOXes. This BBOX filter process is conducted based on the understanding of the object detection designs and is elaborated upon in §4.4.4.

Step 3: Loss Function Design and Optimization Method. This step introduces novel loss functions and a new optimization method detailed in §4.4.5. The designed loss function includes score loss and regression loss to create or remove BBOXes. We propose a new optimization strategy that markedly enhances the effectiveness of the traditional standard Lagrangian relaxation method. To bolster attack robustness, we integrate Expectation over Transformation (EoT), drawing upon prior research [283, 330].

4.4.2 Patch Location Preselection

To effectively generate our attack, it is crucial to strategically position a patch in the most vulnerable area near the vehicle. We formulate this problem as an optimization problem to find the ideal region for patch placement [87]. The detailed process is illustrated in Fig. 4.4.1. The objective function, denoted as $\mathcal{L}_{\text{mask}}$, is formulated as follows:

$$\arg\min_{p,m} \mathcal{L}_{\text{adv}}(x') + \alpha \cdot \mathcal{L}_{M'}(m, \Delta_h, \Delta_w) \tag{4.1}$$

$$\textbf{where } \mathcal{L}_{M'} = \| \max(M') - \frac{1}{hw} \sum_{j=1}^{h} \sum_{i=1}^{w} M'[i,j] \|_1 \tag{4.2}$$

$$M'[i,j] = \sum_{z_1=0}^{\Delta_h - 1} \sum_{z_2=0}^{\Delta_w - 1} M[i+z_1, j+z_2] W[z_1, z_2] \tag{4.3}$$

$$M[i,j] = \frac{1}{2} \times \tanh(\gamma \cdot m[\lfloor \frac{i}{s} \rfloor, \lfloor \frac{j}{s} \rfloor]) + \frac{1}{2} \tag{4.4}$$

$$x' = x \odot (1 - M) + p \odot M \tag{4.5}$$

Equation equation 4.1 is to identify the most vulnerable region leveraging the mask denoted as M, which controls the strength of the perturbation p. The final patch location aims to contain as many pixels with high values of M as possible, to cover the most vulnerable areas. When using this method, two main concerns must be addressed. First, the values of M need to be kept as close to 0 or 1 as possible to reflect the binary decision of either applying or not applying the patch. Second, it is important to keep that the high-value pixels of M are clustered closely, as the patch needs to form a contiguous block.

To address the first concern, M is computed by unconstrained mask parameters m, as shown in Equation equation 4.4. The transformation using the tanh function in Equation equation 4.4 constrains the mask M within [0, 1] range. Tuning the hyperparameter γ drives mask values closer to 0 or 1 and modulates the convergence speed of the process. The hyperparameter s modulates the mask granularity. The variable p signifies the perturbations applied to the original image x through Equation equation 4.5 to obtain the input x' for $\mathcal{L}_{\text{adv}}$. The details of $\mathcal{L}_{\text{adv}}$ will be introduced in §4.4.5. The variables h and w represent the height and width of the image x.

To address the second concern, upon generating a sensitivity mask indicative of the perturbation mask M, a sliding window W of the same size (Δ_h, Δ_w) as the patch is applied to process this mask. The calculated averaged values within the window are referred to as M' shown in Equation equation 4.3, which scores each potential location by averaging the values within the window. Furthermore, leveraging the mask M', we formulate a novel loss function $\mathcal{L}_{M'}$, which plays a pivotal role in determining the unique and most effective patch location. Specifically, by minimizing $\mathcal{L}_{M'}$, we can encourage M to cluster within a uniquely rectangular box of dimensions (Δ_h, Δ_w). The clustering effect is super important for the effectiveness of an adversarial patch, as the patch must form a contiguous block. Therefore, ControlLoc focuses on finding the optimal placement for a tightly grouped continuous block rather than scattered discrete points. α is a hyperparameter.

Moreover, recognizing physical constraints on the capabilities of the attackers, only designated areas are considered viable for patch placement. Therefore, consistent with prior works [203, 212], we restrict the attack areas to the rear side of the vehicle in both digital and physical-world experiments. Thereby, we limit the mask M to these regions. Notably, selecting the patch location can precede attack generation steps, serving as a potential and effective pre-processing step. Furthermore, our patch selection method incurs negligible computational overhead, as we only required 20 iterations in our experiments to determine the optimal location shown in §4.5.3.

4.4.3 Finding Target Fabricated Bounding Box

The core idea behind finding a target fabricated BBOX is to create a scenario where, when our attack has ended, the tracking system loses track of the original object. This is achieved

Algorithm 4.1: Find target fabricated BBOX location

```
Input: $B_o$: Original object BBOX; $\vec{v}$: Attacker desired directional vector; $T_{\text{IOU}}$:
       IOU threshold for data association between trackers and detection results.
Output: $B_t$: Target fabricated BBOX location.
1 $k \leftarrow 1$
2 $B_t \leftarrow B_o$
3 while $\text{IOU}(B_t, B_o) > T_{\text{IOU}}$ do
4     $B_t \leftarrow B_o + \vec{v} \cdot k$
5     $k = k + 1$
6 end
7 $B_t \leftarrow B_o + \vec{v} \cdot (k - 1)$
8 return $B_t$
```

by manipulating the BBOX of the target object to maximize its deviation from the benign, within its original data association range, directing towards a directional vector $\vec{v}$ determined by the attack goal. Unlike previous research [150], which seeks the optimal BBOX location based on the specific tracking algorithm, we employ a tracking-agnostic strategy since the adversarially modified BBOX does not require precise alignment with the adversarial patch's physical location.

To achieve that, the key insight of this approach is that the fabricated BBOX should match the original BBOX, but with a shift as large as possible towards the direction $\vec{v}$. Thus, the fabricated BBOX must overlap the benign BBOX with an IOU above a predefined threshold T_{IOU}, while also being slightly shifted towards the original BBOX position. It's noteworthy that this IOU threshold generally remains consistent across different MOT [103, 326, 150, 53, 160], enabling the application of a general threshold that facilitates a black-box attack model. This general property is critical, as it does not require detailed knowledge of the specific MOT algorithms in use. Our method for iteratively determining the target fabricated BBOX location is outlined in Algorithm 4.1 to find the desired deviation and illustrated in Fig. 4.4.1.

4.4.4 Bounding Box Filter

In DNN-based object detection, most proposed BBOXes do not contribute to patch generation, as they frequently identify irrelevant objects or background elements. To generate the patch effectively, it requires the selection of appropriate BBOXes for fabrication or erasure. This selection hinges on the understanding of object detection. Our approach is adaptable to both anchor-based and anchor-free detection (§4.2.1).

The mainstream object detectors, including one-stage detectors such as the YOLO series, or two-stage detectors such as the RCNN series, introduced in §4.2, can adopt grid-based designs [231, 230, 156, 189, 133, 233]. Grid-based detectors separate the input image into fixed-size grids, with each cell responsible for predicting BBOXes for objects within its vicinity. To ascertain the location of these BBOXes, an offset is calculated from the top-left corner of each cell. A detailed illustration and example for this process is provided in Appendix 4.C, which precisely obtains the BBOX location.

By leveraging the intrinsic property of grid-based detectors above, we introduce the Center bounding box filter (C-BBOX), an effective method for filtering BBOX adaptable for both anchor-based and anchor-free object detection detailed in §4.2. The details of the C-BBOX process are in Algorithm 4.2. C-BBOX first calculates the scaling ratio scale between the input image size and the feature map size, i.e., the size of each grid cell. Then the C-BBOX extracts the grid cell corresponding to B_t (§4.4.3) based on scale.

C-BBOX is compatible with anchor-based and anchor-free models. For anchor-based detectors, where each grid corresponds to multiple anchors, C-BBOX extracts the BBOX having the largest IOU with B_t as B_f in Algorithm 4.2 (top(A) is to obtain the index of maximum value in vector A). For anchor-free detectors, where each grid has a unique anchor, C-BBOX applies a corrective vector in the hijacking direction to accurately filter the BBOXes since such detectors allow for greater flexibility in BBOX placement.

Algorithm 4.2: C-BBOX

```
Input: $B_t$: Target BBOX; $\text{size}_f$: Feature map size; $\text{size}_x$: Image size; $\vec{v}$: Attack
       directional vector, $k_s$: Step size.
Output: $B_f$: BBOX needed to be fabricated.
1  $(c_x, c_y) \leftarrow$ center point of $B_t$ ;
2  $\text{scale} = \text{size}_x\ \text{size}_f \text{id}_{\text{grid}} = \textbf{int}(c_x/\text{scale}, c_y/\text{scale})$ ;
3  grid $\leftarrow$ grid cell corresponding to $\text{id}_{\text{grid}}$ ;
4  if detector is anchor_based then
5      anchors $\leftarrow$ all anchors of grid ;
6      $\text{index}_{\text{anchor}} = \textbf{top}(\text{IOU}(B_t, \text{anchors}))$ ;
7      $B_f = \text{anchors}[\text{index}_{\text{anchor}}]$ ;
8  else if detector is anchor_free then
9      $c_x = c_x + k_s \cdot \frac{\vec{v}}{|\vec{v}|}$ ;
10     $\text{id}_{\text{grid}} = \textbf{int}(c_x/\text{scale}, c_y/\text{scale})$ ;
11     $B_f \leftarrow$ the anchor of grid corresponding to $\text{id}_{\text{grid}}$;
12 return $B_f$;
```

Moreover, C-BBOX assists in pinpointing BBOXes for erasure in anchor-based models by identifying the cell corresponding to the original BBOX, thereby enabling the precise removal of undesired BBOXes. For anchor-free detectors, we use the IOU BBOX filter, similar to previous research [148], to identify BBOXes for erasure. This method initially eliminates predictions with confidence below the NMS threshold. Subsequently, it filters the BBOX by the IOU between each remaining proposal BBOX and B_t.

For detectors not based on grid structures, bipartite matching [73], is used to distinguish between BBOXes for fabrication and those for erasure. This approach ensures our method's applicability across various object detection designs. With the filter methods in Equation equation 4.6, we extract the BBOXes needed to be fabricated B_f and erased B_e.

$$B_f, B_e = F(O_{\text{bbox}}, B_t, B_o) \tag{4.6}$$

where O_{bbox} is all proposal BBOXes before NMS, B_o is the original BBOX, and $F(\cdot)$ is the BBOX filter function.

Algorithm 4.3: Generating Adversarial Patch

```
Input: $x$: Input image; $B_t$: Target BBOX; $B_o$: Original object; $D(\cdot)$: Object
       detector; $N$: Attack iterations; NMS$(\cdot)$: NMS function; $T_{conf}$: Score
       threshold.
Output: $\Delta$: Adversarial patch.
1  Initial $\Delta \leftarrow \Delta_0$ ;
2  for $n = 1$ to $N$ do
3      $O_{\text{bbox}} = D(x + \Delta)$ ;
4      $B_f, B_e = F(O_{\text{bbox}}, B_t, B_o)$ ;
5      $B' = \text{NMS}(O_{\text{bbox}})$ ;
6      if $B_f \cap B' \neq \varnothing$ and $B_e \cap B' = \varnothing$ then
7          $\mathcal{L}_{\text{adv}} = \mathcal{L}_r(B_f, B_t)$ ;
8      else
9          $\mathcal{L}_{\text{adv}} = \mathcal{L}_s(B_f, B_e, T_{conf})$ ;
10     $\mathcal{L} = \mathcal{L}_{\text{adv}} + \mu_2 \cdot \mathcal{L}_{\text{TV}}$ ;
11     $\Delta = \text{Adam}(\Delta, \mathcal{L})$ ;
12 return $\Delta$ ;
```

4.4.5 Loss Design and Optimization Method

As detailed in Algorithm 4.3, ControlLoc involves enhancing the confidence score of B_f to ensure its preservation after NMS, while concurrently adjusting its dimensions and location to closely match B_t. Conversely, it is imperative to diminish the confidence scores of B_e to preclude their inclusion in the detection outcomes. Similar to the existing adversarial patch attacks [330, 283], we also formulate the adversarial patch generation as an optimization problem. The optimization of this attack poses a multiple-objective problem, requiring the simultaneous optimization of the score loss $\mathcal{L}_s$ for the extracted boxes as well as the shape and location loss, collectively referred to as regression loss $\mathcal{L}_r$. Specifically, for an input image x and an object detection model $D(\cdot)$ that excludes NMS, the optimization task can be represented in Equation equation 4.7, aiming to minimize Δ subject to the conditions that B_f is encompassed within B' and B_e is excluded from B', where B' is all BBOXes after

NMS.

$$\begin{aligned} &\arg\min_{\Delta} \mathcal{L}\{F[D(x,\Delta), B_t, B_o]\} \\ &\quad \text{s.t. } B_f \in B' \text{ and } B_e \cap B' = \varnothing \end{aligned} \tag{4.7}$$

where Δ is adversarial patch and F is from Equation equation 4.6.

Score Loss. To effectively manipulate BBOXes in ControlLoc, adjusting their scores is essential. This adjustment aims to enhance the scores of newly generated BBOXes denoted as $\mathcal{L}_f$ while simultaneously reducing the scores of removed BBOXes denoted as $\mathcal{L}_e$. To accomplish this, we introduce a novel score loss, defined in Equation equation 4.8.

$$\mathcal{L}_s = \underbrace{\frac{1}{|B_e|}\sum_{c\in B_e} \mathbb{1}^c \cdot c_{conf}^2}_{\mathcal{L}_e} + \underbrace{\mu_1 \cdot \frac{1}{|B_f|}\sum_{c\in B_f}(1-c_{conf})^2}_{\mathcal{L}_f} \tag{4.8}$$

$$c_{conf} = c_{obj} \cdot \max\{c_{class_i}\},\ i \in [1, N_c] \tag{4.9}$$

where N_c is the number of classes; the indicator function $\mathbb{1}^c$ checks whether the score of a BBOX c exceeds the score threshold T_{conf}; it is set to 1 if true, and 0 otherwise. This formulation aims to adjust scores, enhancing the detection of relevant objects while minimizing the impact of irrelevant ones. Hyperparameters μ_1 is to balance $\mathcal{L}_e$ and $\mathcal{L}_f$.

Regression Loss. To optimize the position and shape of the fabricated BBOXes B_f—aiming to effectively redirect the tracking from the target object—we introduce a regression loss function, as delineated in Equation equation 4.10.

$$\begin{aligned} \mathcal{L}_r = &\overbrace{\frac{1}{|B_f|}\sum_{c\in B_f} -\log(\mathrm{IOU}(c, B_t))}^{\mathcal{L}_{\mathrm{IOU}}} \\ &+ \beta \cdot \underbrace{\frac{1}{|B_f|}\sum_{c\in B_f}(\mathrm{center}(c) - \mathrm{center}(B_t))^2}_{\mathcal{L}_{\mathrm{center}}} \end{aligned} \tag{4.10}$$

where the regression loss $\mathcal{L}_r$ comprises two components: $\mathcal{L}_{\text{IOU}}$ and $\mathcal{L}_{\text{center}}$. The IOU loss aims to reduce the discrepancy in the overlap between the fabricated BBOX B_f and the target BBOX B_t, ensuring accurate coverage and alignment. Thus, the center loss, weighted by a factor β, seeks to minimize the distance between the centroids of B_f and B_t such that the tracker can be moved away.

Total variation Loss. To make the generated adversarial patch smooth, and thus increase the effective range of the attack, the total variation loss in Equation equation 4.11 is used to reduce the color changes between the adjacent pixels.

$$\mathcal{L}_{\text{TV}} = \sum_{i,j} \sqrt{\left|\Delta_{i+1,j} - \Delta_{i,j}\right|^2 + \left|\Delta_{i,j+1} - \Delta_{i,j}\right|^2} \tag{4.11}$$

Optimization Method. Simultaneously optimizing multiple loss functions, particularly $\mathcal{L}_s$ and $\mathcal{L}_r$, requires a sophisticated strategy. Existing literature [150] typically employs the standard Lagrangian relaxation method for this task. This approach involves aggregating the different loss functions into a single objective, each modulated by predetermined coefficients, followed by gradient descent for an optimal solution.

In our case, this method is fundamentally ineffective. Notably, it does not perform well across various coefficient configurations, as detailed in §4.5.3. The inefficacy of simultaneously optimizing multiple loss functions, i.e., $\mathcal{L}_s$ and $\mathcal{L}_r$, is largely attributed to the negative coupling effects in gradients. Essentially optimizing L_f in L_s determines the location of B_f at a coarse-grained level. Subsequent optimization of L_r refines the location and shape of B_f. Thus, the appropriate sequence of optimization should initially focus on L_f in L_s, to ensure that B_f is correctly identified in the detection results after NMS. Then, the subsequent step involves adjusting the location and shape of B_f. However, employing the standard Lagrangian relaxation method to achieve dual optimization presents challenges. The interaction between L_r and L_f in L_s often leads to a negative coupling effect in our problem space, where an excessive gradient on one side can restrict improvements in the other, hindering effective optimization.

Thus, we propose a new optimization method to address the limitations in the standard Lagrangian relaxation method for hijacking attack generation mentioned above:

$$
\begin{aligned}
&\arg\min_{\Delta} \mathcal{L}_{\text{adv}} + \mu_2 \cdot \mathcal{L}_{\text{TV}} \\
\textbf{where } \mathcal{L}_{\text{adv}} &= \mathbb{1}[B_f \cap B' \neq \varnothing \text{ and } B_e \cap B' = \varnothing] \cdot \mathcal{L}_r \\
&+ \mathbb{1}[B_f \cap B' = \varnothing \text{ or } B_e \cap B' \neq \varnothing] \cdot \mathcal{L}_s
\end{aligned} \tag{4.12}
$$

Our method optimizes either $\mathcal{L}_s$ or $\mathcal{L}_r$ based on the condition specified shown in Equation equation 4.12, rather than attempting to minimize a combination of the two loss functions simultaneously. This selective approach ensures that the optimization process is more targeted and effective. The purpose of optimizing $\mathcal{L}_s$ in our selective approach is to satisfy the non-linear constraint in the equation. In other words, $\mathcal{L}_s$ only needs to be optimized to the point where B_f becomes the sole BBOX around the object, rather than being minimized as much as possible. This approach avoids waste of perturbation [65]. Another advantage lies in its ability to address the issue of imbalanced gradients between the two loss functions, particularly in the context of the coupled problem of location, shape and score of BBOX in object detection. The attack generation is in Algorithm 4.3. μ_2 is a hyperparameter.

Attack Robustness Enhancement. To enhance the attack robustness, particularly in physical world, we incorporate the Expectation over Transformation (EoT) [55, 111, 148, 283] illustrated in Fig. 4.4.1. This involves applying various transformations, such as color modification. Our attack does not rely on a large EoT distribution across varying distances, unlike previous methods that aim to conceal objects across varying distances. This is a key advantage of our approach, as we only require short-term success to create a lasting impact. For angle transformations, we incorporate perspectives from behind the vehicle (to

Table 4.5.1: The effectiveness of attacks on four object detection (OD) models, i.e., ApoD, Y3, Y5, and YX, with four MOT algorithms, i.e., ApoT, BoT-SORT, ByteTrack, and StrongSORT. The evaluation metrics include the attack success rate (ASR) and the average number of frames to execute an effective attack (Frame #). BS: BoT-SORT; BT: ByteTrack; SS: StrongSORT.

Attack scenario	OD\MOT		BDD dataset [315] ApoT	BS	BT	SS	KITTI dataset [120] ApoT	BS	BT	SS	Ave.
Move-in	ApoD	ASR	100%	100%	100%	90%	90%	100%	100%	90%	96.3%
		Frame #	3.1	2.8	2.6	2.5	2.4	2.4	2.7	2.6	2.6
	Y3	ASR	100%	100%	100%	100%	100%	100%	100%	100%	100%
		Frame #	3.2	2.9	3.4	2.7	2.5	2.6	3.1	2.4	2.9
	Y5	ASR	100%	100%	100%	100%	100%	100%	100%	100%	100%
		Frame #	3.1	2.8	2.9	2.6	2.7	2.3	3.0	2.9	2.8
	YX	ASR	100%	100%	100%	100%	100%	100%	100%	90%	98.8%
		Frame #	3.8	3.0	3.1	2.7	2.6	2.9	3.5	2.3	3.0
Move-out	ApoD	ASR	100%	100%	100%	100%	90%	100%	100%	100%	98.8%
		Frame #	3.6	2.6	2.5	2.4	2.9	2.6	2.4	2.8	2.7
	Y3	ASR	100%	100%	100%	100%	100%	100%	100%	100%	100%
		Frame #	4.0	2.8	2.7	2.2	2.9	2.4	2.4	2.2	2.7
	Y5	ASR	100%	100%	100%	100%	100%	100%	100%	100%	100%
		Frame #	3.5	2.7	2.6	2.3	2.8	2.2	2.4	2.2	2.6
	YX	ASR	90%	90%	90%	90%	80%	90%	100%	100%	91.3%
		Frame #	4.5	2.9	2.9	2.8	2.7	2.4	2.9	2.4	2.9
Ave.		ASR	98.8%	98.8%	98.8%	97.5%	95.0%	98.8%	100%	97.5%	98.1%
		Frame #	3.6	2.8	2.8	2.5	2.7	2.5	2.8	2.5	2.8

simulate 'move-out' attacks) and from adjacent lanes (to simulate 'move-in' attacks) into our transformation set. This ensures the attack remains effective under typical driving camera angles. The disappearing patch attack is detailed in Appendix 4.B.

4.5 Evaluation

4.5.1 Evaluation Methodology and Setup

AD Perception. We include different AD perception systems, i.e., different object detection

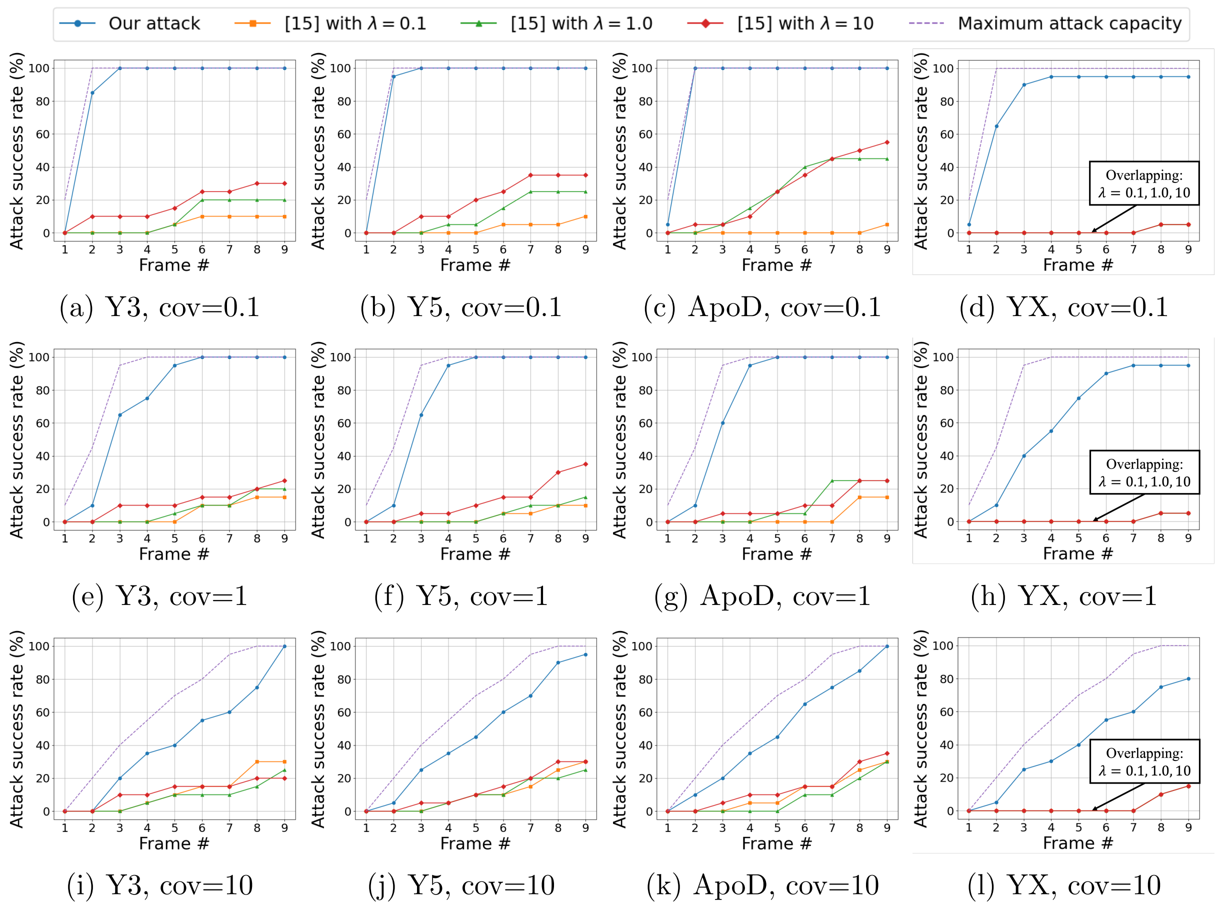


Figure 4.5.1: Comparison between our attack and the baseline attack [150] under four different object detection models (Y3, Y5, ApoD, and YX) with three different parameter values of ApoT (cov = 0.1, 1, 10). λ is the hyperparameter in [150]. Maximum attack capability assumes the attacker can arbitrarily control BBOX locations.

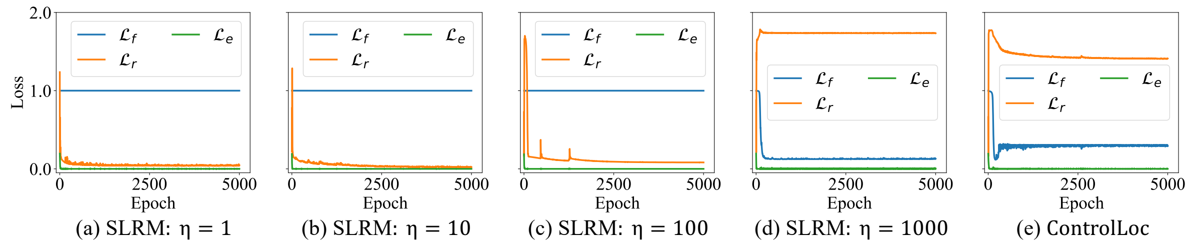


Figure 4.5.2: Comparison of loss value between our optimization method in ControlLoc and the standard Lagrangian relaxation method (SLRM) with different hyperparameter values η. The detailed loss designs are in §4.4.5: Equation equation 4.8 and Equation equation 4.10.

models and MOT. For object detection, we encompass both anchor-based and anchor-free detectors. Our examination mostly leverages algorithms in open-source industry-grade full-stack AD systems to affirm the practicality and representativeness of our findings. We select a variety of object detection models, including the Baidu Apollo Object Detection (ApoD) [53]; YOLO v3 (Y3) [231] as incorporated in Autoware.AI [160]; YOLO v5 (Y5) [156] which is highlighted in recent security research on AD [148]; and YOLOX (YX) [118], an anchor-free detector in the latest Baidu Apollo Beta. For MOT, our focus extends to leading and representative algorithms that underscore the diversity and advancement in the field. This includes the the MOT used in Baidu Apollo [53](ApoT); BoT-SORT [46]; ByteTrack [326], and StrongSORT [103]. We all use their default configurations.

Datasets. We select two widely recognized datasets in the AD research [71, 150, 315, 120]: the Berkeley Deep Drive (BDD) dataset [315] and the KITTI dataset [120]. Within the BDD dataset, we randomly chose 20 clips specifically for their relevance to our attack goals: 10 clips are for the object move-in scenario, and another 10 are chosen for the object move-out scenario. A similar selection process is applied to the KITTI dataset. We manually identify a target vehicle within each clip. To align our study with realistic conditions, we impose restrictions on adversarial patch size, for which our patch average size is 12% of the target vehicle in pixels.

4.5.2 Attack Effectiveness

Evaluation Metrics. The success of the attack is defined as *the attack is considered successful when, at the end of the attack, the detection BBOX of the target object can no longer be associated with any existing trackers.* Such metric is widely used in the security analysis of tracking [150, 211]. We measure the attack success rate (ASR) and the average number of frames to conduct an effective attack (Frame #). Note that the Frame # is within the attack successful cases.

Results. The attack effectiveness on four object detectors and four MOT algorithms across two datasets, aiming for two specific attack goals, is detailed in Table 4.5.1. The attack boasts an average success rate of 98.1% and necessitates an average of 2.8 frames to achieve efficacy in general. Among the MOT algorithms evaluated across the two datasets, ApoT emerges as the most robust one, evidenced by its lowest average attack success rate of 96.9% and the highest average of 3.2 frames required for a successful attack. These findings suggest that attacking ApoT demands a higher frame count and has a lower attack success rate, rendering it less vulnerable compared to other MOT algorithms. Regarding object detection, YX demonstrates the lowest attack success rate at 95.1% and requires the highest average of 3.0 frames for a successful attack. This robustness could be attributed to its anchor-free object detection design, which appears more robust against hijacking attacks. Within the anchor-based object detection models, ApoD shows the lowest attack success rate at 97.6%, suggesting that the design of object detection and MOT in Apollo tends to be more robust. Note that YX is also adapted in Apollo as introduced in § 4.5.1. An additional observation is that the move-in attack achieves a higher success rate of 98.8% but generally requires more frames (average of 2.8) compared to the move-out attack, which has a success rate of 97.5% with an average of 2.7 frames. This suggests that, although move-in attacks might be easier to successful than move-out attacks, the latter tend to reach attack goals faster within successful cases. From Table 4.5.1, StrongSORT exhibits greater robustness compared to others, except ApoT. This is likely due to a Noise Scale Adaptive Kalman filter [103] design, which adjusts measurement noise covariance based on confidence scores of detection results.

4.5.3 Comparison with Baselines

4.5.3.1 Comparison with Prior Attack [150]

Methodology and Setup. For the camera-based perception pipeline, we chose different object detectors coupled with ApoT due to their adoption in an industry-grade full-stack AD system. The evaluation utilizes the BDD dataset as outlined in §4.5.1. Following the methodology of prior research as our baseline [150], we employ λ to denote the weighting factor between two loss functions in the baseline method, $\mathcal{L}_1$ and $\mathcal{L}_2$, thus defining the combined loss function as $\mathcal{L} = \mathcal{L}_1 + \lambda \cdot \mathcal{L}_2$. The $\mathcal{L}_1$ is for erasure and $\mathcal{L}_2$ is for fabrication. We also explore the impact of varying λ values: 0.1, 1.0, and 10.0. Additionally, for ApoT, we investigate its performance across different tracking parameters, cov: noise covariance in Kalman filter [150] following the same setup as the baseline [150].

Results. The results, as depicted in Fig. 4.5.1, unequivocally demonstrate the superior efficacy of ControlLoc, achieving an impressive 99.4% attack success rate on Y3, ApoD, and Y5 models, and a 90% attack success rate on the YX model. This starkly contrasts with the outcomes from existing research [150], which has an 8.3% attack success rate on the YX model and 24.8% on the other models tested. This substantial discrepancy underscores the enhanced capability of our ControlLoc to manipulate the target object's position effectively, thereby hijacking its tracker. A critical observation from our analysis reveals that prior research [150] tends to fail in maintaining the target's BBOX: at low λ values, leading to its disappearance, or conversely, at high λ values, resulting in no significant change or generating multiple BBOXes. In stark contrast, our ControlLoc demonstrates remarkable effectiveness and robustness to different *cov* values of ApoT. In certain instances, ControlLoc achieves similar performance to maximum attack capacity, which assumes the attacker can arbitrarily manipulate the BBOXes.

4.5.3.2 Comparison with Traditional Optimization

This part compares our novel optimization method with the traditional standard Lagrangian relaxation method (SLRM) in this hijacking attack context. Our method, delineated in Equation equation 4.12, diverges from SLRM, which merges score loss ($\mathcal{L}_s$) and regression loss ($\mathcal{L}_r$) using a hyperparameter η in the form $\mathcal{L}_r + \eta \cdot \mathcal{L}_s$. Notably, the score loss encompasses two components, $\mathcal{L}_f$ and $\mathcal{L}_e$, as specified in Equation equation 4.8. To facilitate a detailed comparison, we use $\mathcal{L}_f$ and $\mathcal{L}_e$ for $\mathcal{L}_s$. Previous research leveraging SLRM [150]

Table 4.5.2: ASR comparison between our patch location preselection (§4.4.2) in ControlLoc and a random location preselection on the rear of the vehicle.

	Random		Ours	
	Move-in	Move-out	Move-in	Move-out
ASR	20%	20%	**90%**	**80%**

demonstrates its inadequacy in generating effective adversarial patches for tracker hijacking. This limitation is illustrated through the three losses, $\mathcal{L}_r$, $\mathcal{L}_f$, and $\mathcal{L}_e$, which fail to optimize simultaneously under varying hyperparameter η settings, as depicted from Fig. 4.5.2 (a) to (d). The primary challenge arises from the low initial score of the fabricated BBOX (B_f), resulting in a correspondingly weak gradient. Thus, SLRM hinders the minimization of $\mathcal{L}_f$, particularly when with high regression loss. This typically leads to a negligible reduction in $\mathcal{L}_f$, as evidenced in Fig. 4.5.2 (a) to (c), where $\mathcal{L}_f$ barely decreases unless η is substantially increased, for example, to around 1000, as shown in Fig. 4.5.2 (d). However, elevating the η introduces a new problem: the regression loss ($\mathcal{L}_r$) fails to be well optimized, shown in Fig. 4.5.2 (d). This damages the attack's effectiveness, preventing the fabricated BBOX from associating with the target tracker. However, our optimization approach successfully mitigates these issues, showing its efficacy in Fig. 4.5.2 (e).

4.5.3.3 Baseline Evaluation for Stage I in §4.4.2.

This part assesses the benefit of Stage I by comparing two scenarios: Stage I for patch location preselection and a random patch location on the rear of the vehicle. For a fair comparison, we maintain consistent patch sizes and conduct 1,000 iterations for each attack generation. Specifically, for Stage I, we involve 20 iterations to determine the optimal patch location. The results, in Table 4.5.2, reveal that attacks employing Stage I achieve an average attack success rate of 85.0% across two attack goals, whereas those with a random patch location exhibit a significantly lower attack success rate of 20.0%. This underscores the importance of Stage I in ControlLoc.

Table 4.5.3: Physical-world attack evaluation regarding ASR for ControlLoc and baseline attack [150] under different outdoor light conditions, i.e., sunny, cloudy, and night; angles, i.e., L1 to L4 defined by the lateral distance shown in Fig. 4.5.3, where L1 and L2 correspond to the move-out attack, while L3 and L4 are used for the move-in attack; and background, i.e., B1 to B6, including common roadside and intersection scenarios encountered during driving. The results are averaged over 5 videos.

	B1		B2		B3		B4		B5		B6	
	[150]	Ours	[150]	Ours	[150]	Ours	[150]	Ours	[150]	Ours	[150]	Ours
Sunny day (∼20,000 lux)												
L1	0%	60%	0%	80%	0%	60%	0%	60%	0%	60%	0%	40%
L2	0%	100%	0%	100%	0%	100%	0%	80%	0%	80%	0%	80%
L3	0%	100%	0%	80%	0%	80%	0%	60%	0%	80%	0%	80%
L4	0%	80%	0%	80%	0%	80%	0%	80%	0%	60%	0%	80%
Cloudy day (∼8,000 lux)												
L1	0%	60%	0%	80%	0%	80%	0%	60%	0%	60%	0%	80%
L2	0%	80%	0%	80%	0%	80%	0%	80%	0%	80%	0%	80%
L3	0%	100%	0%	100%	0%	100%	0%	100%	0%	80%	0%	100%
L4	0%	100%	0%	100%	0%	100%	0%	100%	0%	100%	0%	100%
Night (∼70 lux)												
L1	0%	60%	0%	60%	0%	60%	0%	40%	0%	60%	0%	40%
L2	0%	80%	0%	80%	0%	80%	0%	60%	0%	60%	0%	60%
L3	0%	80%	0%	100%	0%	100%	0%	80%	0%	80%	0%	80%
L4	0%	80%	0%	80%	0%	80%	0%	80%	0%	80%	0%	80%

4.5.4 Physical-World Attack Evaluation

Evaluation Setup and Methodology. To systematically evaluate the effectiveness of ControlLoc in the physical world, we conduct experiments on real driving roads under various physical-world factors, including angle, light conditions, and background. Specifically, we conduct experiments on driving routes at four different angles, as illustrated in Fig. 4.5.3, capturing video at a constant speed as the vehicle approaches from a distance to achieve both move-in and move-out attack goals depending on the positioning of the vehicle. The angles are defined based on the lateral distance from the vehicle, i.e., L1 to L4 shown in Fig. 4.5.3 with around 1 m per segment. Among them, L1 and L2 correspond to the move-

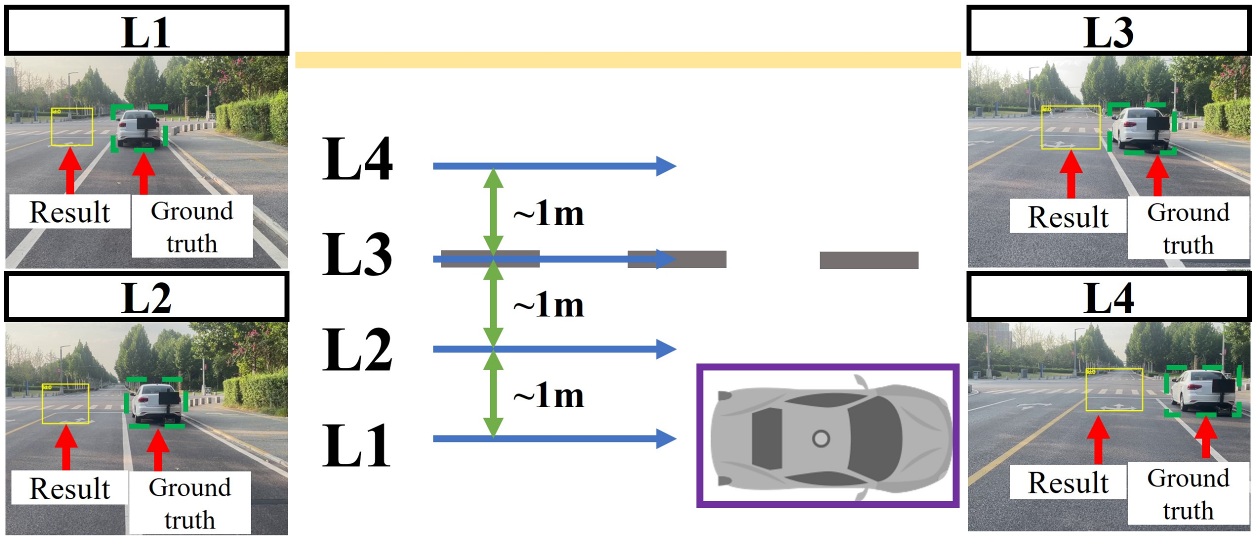


Figure 4.5.3: Physical-World attack evaluation setups.

out attack, while L3 and L4 are used for the move-in attack. Our attack goals align with the angles, as introduced in §4.3. Additionally, we evaluate the system under three different light conditions: the sunny day (approximately 20,000 lux), the cloudy day (approximately 8,000 lux), and nighttime (approximately 70 lux). Furthermore, we experiment with six different backgrounds B1 to B6, including common roadside and intersection scenarios encountered during driving, to explore hijacking attacks from both directions: right-to-left (B1-B3) and left-to-right (B4-B6). The combination of these various physical-world factors results in a total of 72 scenarios. We conduct physical-world attack evaluation in controlled environments. For our attack, we insert a single-frame adversarial patch for moving the target object and a three-frame adversarial patch for hiding the target object in a benign advertisement video, with a resolution of 1080P. The video is displayed on a 32-inch monitor, which is smaller than the physical monitors typically used in existing research on AD security [203]. For each scenario, we collect five video clips for analysis. For object detection and MOT, we utilize the systems implemented in Baidu Apollo, specifically ApoD and ApoT, due to their representativeness. The camera used for these recordings has the same configurations—such as focal length and video resolution—as the camera used in the Baidu Apollo project [53].

For the baseline [150], the hyperparameter λ is set to 10, as this value yields the best performance in §4.5.3, and we utilize the same EoT as used in ControlLoc. We ensure that all other factors that could influence the results, such as video recording, are consistent.

Results and Visualization. The effectiveness of ControlLoc comparing with baseline attack [150], under variations in angle, light conditions, and background, is presented in Table 4.5.3. Our ControlLoc achieves a 79% average attack success rate, while the baseline [150] shows no effectiveness, evidenced by a 0% attack. This ineffectiveness arises from their lack of a precise BBOX filtering method and the shortcomings of their optimization approach, which makes it difficult to address the imbalanced gradients between the score loss and regression loss. As a result, the patch used to manipulate the BBOX fails to function effectively. Notably, on cloudy day, ControlLoc has better effectiveness, yielding an 87% attack success rate compared to a 77% attack success rate on the sunny day and a 73% attack success rate at night. The root cause is that the patches displayed on the monitor experience minimal distortion under cloudy light conditions. In contrast, the stronger light on sunny day lowers the brightness of the displayed patches, while the weaker light at night increases their brightness, causing glare. Additionally, the attack success rate shows limited variation across different backgrounds, indicating that our attack is largely insensitive to background changes. As for the attack goals, the move-in attack (L3, L4) achieves a higher success rate of 87.0%, making it more effective than the move-out attack (L1, L2), which has a success rate of 70.5%. This observation is consistent with the findings from digital-space evaluations (§4.5.2).

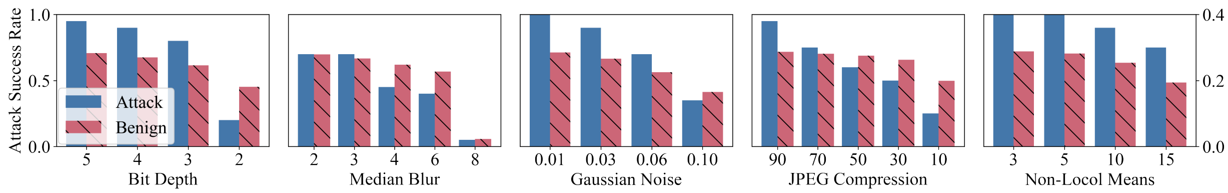


Figure 4.5.4: Attack effectiveness regarding attack success rate and model utility regarding mAP (mean Average Precision) of five common input transformation-based defenses. The x-axis represents the strength of each defense.

Table 4.5.4: AD system-level evaluation (vehicle collision or unnecessary emergency stop rate) under different speeds using Baidu Apollo and LGSVL. 20 runs for each cell.

	Move-out				Move-in			
	L1		L2		L3		L4	
Speed (km/h)	20	40	20	40	20	40	20	40
Benign	0%	0%	0%	0%	0%	0%	0%	0%
ControlLoc	65%	75%	70%	80%	100%	95%	100%	90%

4.5.5 System-Level Attack Effect Evaluation

Evaluation Setup and Methodology. To study the AD system-level attack effects of ControlLoc, we perform an attack evaluation on Baidu Apollo [53], an industry-grade full-stack AD system, using LGSVL simulator [234], a production-grade AD simulator. Our experiments are conducted on the Borregas Ave map and the Lincoln2017MKZ AD vehicle. To enhance the perception fidelity of simulators, we model the location of the tracker after it has been hijacked and inject it into the AD system from our physical-world attack evaluation results in §4.5.4 including the four different drive trajectories with different angles/lateral distances as shown in Fig. 4.5.3. Our evaluation focuses on two representative scenarios as shown in Fig. 4.2.2. Move-in attack (Fig. 4.2.2 (a)) is a common scenario for other vehicles to park on the side of the road and move-out attack (Fig. 4.2.2 (b)) is another common driving scenario. We perform 20 runs on each scenario with 20 km/h and 40 km/h.

Results. The outcomes are summarized in Table 4.5.4. Our ControlLoc achieves an average AD system-level attack effectiveness rate of 84.4% for critical scenarios such as vehicle collisions or unnecessary emergency stops while maintaining normal operation in benign situations with a 0% incidence of attack effects on the AD system. The efficacy of the move-in attack at 96.3% is notably superior to that of the move-out attack, which has a 72.5% rate. Notably, the attack effectiveness at high speeds (40 km/h) reaching 85% surpasses that at lower speeds (20 km/h), which is 83.8% in the move-out scenarios. This is critical as high speed poses significant safety risks.

4.6 Discussion

4.6.1 Defenses

DNN-Based Defense. Prior research has focused on enhancing the robustness of DNNs against adversarial attacks. Such efforts fall into two broad categories: certified defenses [175, 297, 294] and non-certified defenses [307, 105, 325]. Certified defenses offer provable guarantees of robustness but are generally time-intensive, rendering them impractical for real-time systems, like AD systems. Furthermore, there is a notable absence of certified defenses specifically designed to defend against attacks on the entire AD perception. Thus, we evaluate several non-certified defense strategies: input-transformation defenses, which are directly adaptable. These include JPEG compression [105], bit depth reduction [307], Gaussian noise [325], median blur [307], and non-local means [307, 323]. Due to their easily adaptable nature, these methods have been assessed in recent security studies [71, 239, 335, 323]. We use the BDD dataset and the perception module in Baidu Apollo, i.e., ApoD and ApoT.

The effectiveness of these defense measures is quantified by the attack success rate, while the impact on benign performance is assessed using the mean Average Precision (mAP). As shown in Fig. 4.5.4, we observe that median blur can partially mitigate the attacks, particularly with large kernel sizes (e.g., 8). However, it remains possible for the attack to succeed, and importantly, this harms the model, which inevitably causes serious consequences in safety-critical applications [335]. Thus, these defenses are not practically applicable.

Sensor Fusion Based Defense. Employing multi-sensor fusion (MSF) for improving perception robustness, such as integrating LiDAR, represents a strategic defensive approach. However, incorporating additional sensors like LiDAR substantially raises system costs.

Thus, many AD systems primarily utilize camera-based perception, such as Tesla [153] and OpenPilot [91]. Additionally, relying solely on MSF may not adequately defend against ControlLoc. This vulnerability is attributed to the potential for attackers to simultaneously attack all perception sources [71]. Furthermore, recent research [87] shows the feasibility of attacking the MSF-based perception by attacking only camera-based perception. While the MSF may compromise the attack, its defensive potential remains to be systematically explored in future research.

Collision Avoidance System Based Defense. Collision avoidance systems (CAS), like Autonomous Emergency Braking (AEB), use RADAR or ultrasonic sensors to prevent or reduce the severity of collisions [71, 40, 164]. While helpful, they cannot fully prevent collisions or eliminate the need for robust defense methodologies against ControlLoc. First, AD systems must independently handle as many safety hazards as possible, rather than relying solely on CAS, which should serve as backup safety measures for emergencies. Second, CAS is insufficient for defending against move-in attacks, making them inadequate against ControlLoc. Additionally, these systems are not perfect, achieving only a 27% reduction in bodily injury claim frequency and a 19% reduction in property damage frequency [106, 314]. Thus, AD systems must be designed to handle safety hazards independently.

4.6.2 Limitation and Future Work

Despite promising outcomes in physical-world tests, the full impact of end-to-end attacks on AD systems, especially commercial vehicles like Tesla, remains to be fully understood. Constraints such as cost and safety lead us to use simulations [71], a common industry practice, for preliminary AD system-level assessments [71, 283]. More comprehensive and realistic evaluations can be a future work. Our research predominantly examines one-stage detectors used in industry-grade AD systems. However, considering the existence and application of two-stage detection, systematic investigation can be a future direction for attack generality.

4.7 Conclusion

In this paper, we present ControlLoc, a novel physical-world adversarial patch attack to exploit vulnerabilities in AD perception including object detection and MOT. With a two-stage attack methodology, ControlLoc significantly outperforms the existing attack, achieving an impressive average success rate of 98.1% across diverse AD perception systems. The effectiveness of ControlLoc is validated in real-world conditions with a 79% average attack success rate. AD system-level impact such as vehicle collisions is also evaluated using a production AD simulator with 84.4% attack effectiveness.

Appendix 4.A Ethics Considerations

When addressing the ethics related to the evaluation of physical-world attacks, it is imperative to underscore the evaluation taken to ensure both safety and responsibility. Our experimental setup is located on a low-traffic road within our institute, under controlled conditions to ensure minimal traffic. All equipment was operated by individuals experienced in outdoor vehicle experiments. This can effectively avoid the risk of unintended consequences to the uninvolved public. Additionally, we confirm that no harm is caused to the commercial vehicles in our physical-world experiments. These vehicles are for data collection.

Acknowledging the potential ramifications our findings may have on the security of AD systems, we commit to executing a responsible vulnerability disclosure. We will ensure that all necessary precautions are taken before making our research publicly available.

Appendix 4.B Disappearing Patch Attack Generation

We adopt a dual-patch attack strategy to perform the hijacking attack. This approach consists of one patch designed for moving the BBOX's location of the target object and a second for making the object disappear. The disappearing patch attack can draw upon methodologies similar to previous studies [148, 283, 330] detailed as follows.

Utilizing the BBOX filter described in §4.4.4, we can efficiently identify and remove the BBOXes that should be excluded. We denote these BBOXes as the set B'_e. To achieve the goal, we reduce the scores of all BBOXes in B'_e, as formulated in Equation equation 4.13, aligning with the methodology used in previous research [330, 283, 148, 111].

$$\mathcal{L}_{\mathrm{e}'} = \frac{1}{|B'_e|} \sum_{c \in B'_e} c^2_{conf} \tag{4.13}$$

where the c_{conf} is detailed in Equation equation 4.9. The attacker can display these patches on a monitor.

Appendix 4.C Location of BBOXes for Grid-Based Object Detector

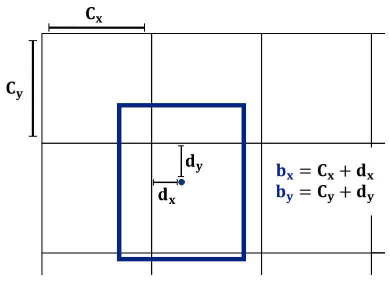


Figure 4.C.1: The center coordinates of bounding boxes prediction method adopted by grid-based detection algorithms.

To ascertain the location of BBOXes for grid-based object detectors, an offset is calculated from the top-left corner of each cell as shown in Fig. 4.C.1. For instance, YOLO v5 [156] determines the location coordinates relative to each cell's position. The detection model computes t_x and t_y for each BBOX within the output feature map. If a cell is positioned away from the image's top-left corner by $(c_x,\, c_y)$, these predictions will be adjusted based on Equation equation 4.14.

$$\begin{aligned} b_x &= d_x + c_x, \text{ where } \ d_x = \sigma(t_x) \\ b_y &= d_y + c_y, \text{ where } \ d_y = \sigma(t_y) \end{aligned} \tag{4.14}$$

where $\sigma(.)$ is the Sigmoid function. This process, utilized by grid-based detectors, calculates the offset (d_x, d_y) to the center coordinates (b_x, b_y) of each BBOX, ensuring that the center of any predicted BBOX remains within the confines of its cell.

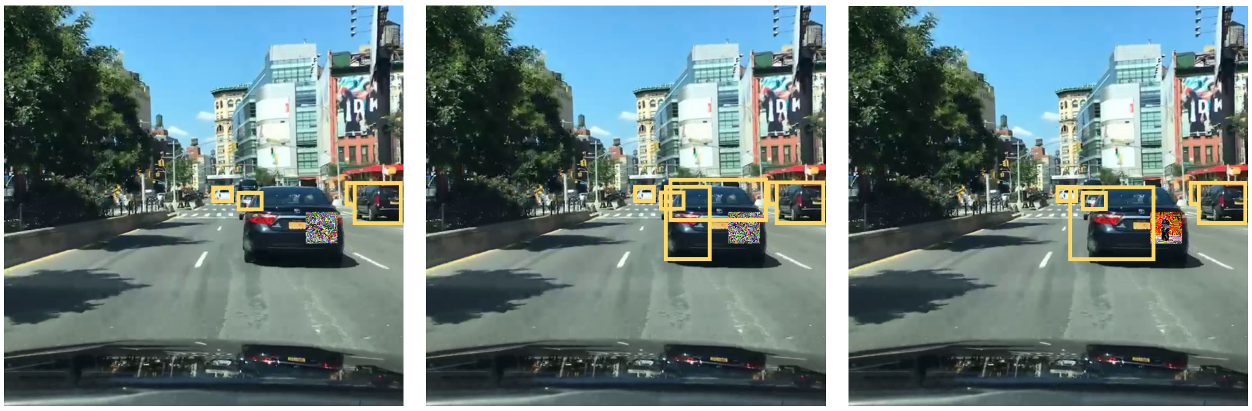

(a) [150] with small λ (b) [150] with suitable λ (c) ours

Figure 4.C.2: Comparison of detection results between the baseline method [150] and our method.

Appendix 4.D Comparison Figure

Fig. 4.C.2 illustrates comparison with [150]. When λ is small, the BBOXes around the object are disappear. And when λ is set to a appropriate value, more than one BBOXes appear around objects, interfering with the association between the tracker and the target BBOX.

This issue arises because previous works lack precise BBOX filtering, unlike our proposed BBOX filter. Furthermore, without our new optimization method to address the imbalance gradients between the score loss and regression loss, even if the target BBOX appears, its shape cannot successfully associate and hijack the tracker.

Appendix 4.E Patch Location under Different Frames

We select several consecutive frames from a video clip and obtain their respective optimal patch positions via the method in Stage I, as illustrated in Fig. 4.E.1. The results demonstrate that the patch positions do not vary significantly across these frames and can be encompassed within the positional transformation distribution in the EoT. This observation indicates that our method is effective for continuous video frames. The results demonstrate that our method can maintain relatively stable patch positions as the patch sizes (Δ_h and Δ_w) vary with the changing relative distance between the AD vehicle and the Attacker's vehicle. Furthermore, such minor changes in position can be encompassed within the positional transformation distribution in the EoT. This observation indicates that our method is effective for continuous video frames.

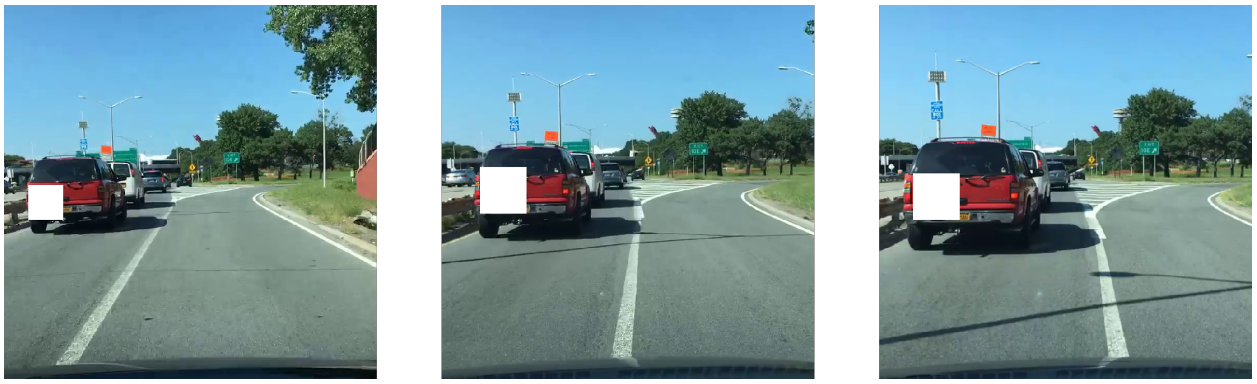


Figure 4.E.1: Patch location under different frames. The white square in each image frame indicates the patch's optimal location.

Chapter 5

Does Physical Adversarial Example Really Matter to Autonomous Driving? Towards System-Level Effect of Adversarial Object Evasion Attack

5.1 Introduction

Autonomous Driving (AD) vehicles are now a reality in our daily life, where a wide variety of commercial and private AD vehicles are driving on the road. For instance, the millions of Tesla cars [158] equipped with Autopilot [265] are publicly available. To ensure safe and correct driving, a fundamental pillar is *perception*, which is designed to detect surrounding objects in real time. Due to the safety- and security-criticality of AD perception, various prior works have studied its security, especially the ones that aim at causing the evasion of critical physical road objects (e.g., STOP signs and pedestrians), or *physical adversarial object evasion attack* [148, 305, 83, 293, 330, 70, 110, 186, 69].

Although these attacks are all motivated by causing erroneous driving behaviors at the AD system level (e.g., vehicle collisions and traffic rule violations), we find that so far they predominately only evaluate the attack success *at the targeted AI component level alone* (e.g., judged by per-frame object misdetection rates [83, 110, 305, 330, 148]), without further evaluation *at the system level.* Specifically, to systematically perform such system-level evaluation, we need to measure the end-to-end system-level attack success metrics (e.g., collision rates) with the full system-level attack context enclosing the attack-targeted AI component, for example, the remaining AD system pipeline such as object tracking, planning, and control, closed-loop control, and the attack-targeted driving scenario. In this paper, we call such system-level attack context *system model* for such adversarial attacks (§5.2). This thus raises a critical research question: *can these existing works on physical adversarial object evasion attacks effectively achieve the desired system-level attack effects in the realistic AD system settings?*

To systematically answer this critical research question, we conduct the first measurement study on representative prior object-evasion attacks with regard to their capabilities in causing system-level effects (§5.3). We propose a general framework, i.e., a system model, including perception modeling from the physical world, to measure STOP sign-evasion attack which is our target due to its high representativeness [252] and its direct impacts on driving correctness and road safety. Our results show that *all* the representative existing works cannot cause any STOP sign traffic rule violation within the system model including a representative closed-loop control AD system in the common speed range for STOP sign-controlled roads in the real world even though the most effective attack can achieve more than 70% average attack success rate at the AI component alone.

We further investigate the root causes and find that all the existing works have design limitations on achieving effective system-level effects due to the lack of a system model in AD context for attack design: 1) physical model-inconsistent object size distribution in pixel sampling and 2) lack of vehicle plant model and AD system model consideration (detailed in §5.4). We further propose SysAdv, a system-driven attack design, which can be integrated with all state-of-the-art attack methods to significantly improve system-level effects by overcoming the two limitations.

We evaluate our novel proposed attack design in our platform and show that the system-level effect can be significantly improved in §5.5, i.e., the system violation rate can be increased by around 70%. To further validate the generality of our attack, we also examine generality on different AD system parameters (§5.5.2) and different object types (§5.5.3), which shows improvement at both component- and system-level. Demo videos are at the project website: **https://sites.google.com/view/cav-sec/sysadv**.

To sum up, this paper makes the following contributions:

- We conduct the first measurement study on the system-level effect of the representative prior object-evasion attacks with our proposed novel evaluation framework (i.e., system model) including 4 popular object detectors and 3 state-of-the-art object-evasion attacks.
- We identify the limitations of prior works which hinder them in potently achieving system-level effects and propose SysAdv, a system-driven adversarial object-evasion attack with the system model in AD context.
- We further evaluate SysAdv and show that the system-level effect of SysAdv can be significantly improved, i.e., the system violation rate increases by around 70%.

5.2 Related Work and Background

Camera-based AD perception. Camera-based AD perception generally leverages DNN-based object detection to detect or recognize road objects of various categories (e.g., traffic signs, vehicles, and pedestrians) in consecutive image frames [77]. State-of-the-art DNN-based object detectors can be classified into two categories: one-stage object detector, and two-stage object detector [338]. The former, such as YOLO [230, 231, 156], usually has higher detection speed, while the latter, such as Faster R-CNN [232], usually has higher detection accuracy. In this paper, we focus on the security aspects of camera-based AD perception and perform the corresponding experiments on both object detector categories. We perform the measurement study of physical adversarial object evasion attack in AD perception §5.3 including these two kinds of object detectors.

Physical adversarial object evasion attacks in AD context. Recent works find that DNN models are generally vulnerable to adversarial attacks [123, 76, 198, 301, 323, 192]. Due to the direct reliance of camera-based AD perception on DNN object detectors, various prior works have explored the feasibility of adversarial attacks in AD context [148, 330, 305, 337, 285, 252, 282, 97, 192, 240, 191, 242]. Among them, *physical adversarial object evasion attacks*, which typically use physical-world attack vectors such as malicious patches to cause the disappearance of road objects (e.g., pedestrians and traffic signs) [148, 110, 330, 305, 83, 293], are especially severe due to their direct impacts on driving correctness and road safety. However, as detailed in later sections, we find that so far the considerations and integration of the corresponding *system models* (detailed below) in the prior works are *far from enough* in both attack designs and evaluation, which substantially jeopardizes the meaningfulness of their designs from the end-to-end AD driving perspective (§5.3).

Gap between AI component errors and their system-level effect. We do not intend to claim to be the first to point out, analyze, measure, or optimize the gap between

AI component errors and their system-level effect in general; there exists a large body of prior works in various other problem contexts (e.g., computer vision system [145, 229, 147], image analysis [131, 320], camera surveillance [126, 127], video analytics [266, 125], planning [224, 223, 270], and control [270]) across academia and industry that have studied the characterization and/or optimization of end-to-end system performance [122, 66] with regard to AI/vision component errors. Nevertheless, to the best of our knowledge, none of them 1) quantified such gaps in the context of adversarial attacks on autonomous systems, especially those in real-world system setups; and 2) identified novel designs that can systematically address or fill such gaps on autonomous systems, which we believe are our novel and unique contributions.

Systems model for AD AI adversarial attacks. To understand the end-to-end system-level impacts of an adversarial attack against a targeted AI component in an AD system (e.g., whether it can indeed effectively cause undesired AD system-level property violations), we need to systematically consider and integrate the overall system semantics and context that enclose such AI component into the security analysis [99, 247]. In this paper, we call a systematic abstraction of such system semantics and context the *system model* of such AD AI adversarial attacks. Specifically, in the AD context we identify 3 essential sub-components in such system model: 1) *the AD system model*, i.e., the full-stack AD system pipeline that encloses the attack-targeted AI components and closed-loop control, e.g., the object tracking, planning, and control pipeline for the object detection AI component; 2) *the vehicle plant model* [214, 99], which defines the physical properties of the underlying vehicle system under control, e.g., maximum/minimum acceleration/deceleration, steering rates, sensor mounting positions, etc.; and 3) *the attack-targeted operation scenario model*, which defines the physical driving environment setup, driving norms (e.g., traffic rules), and the system-level attack goal (e.g., vehicle collision, traffic rule violation, etc.) targeted by the AD AI adversarial attack.

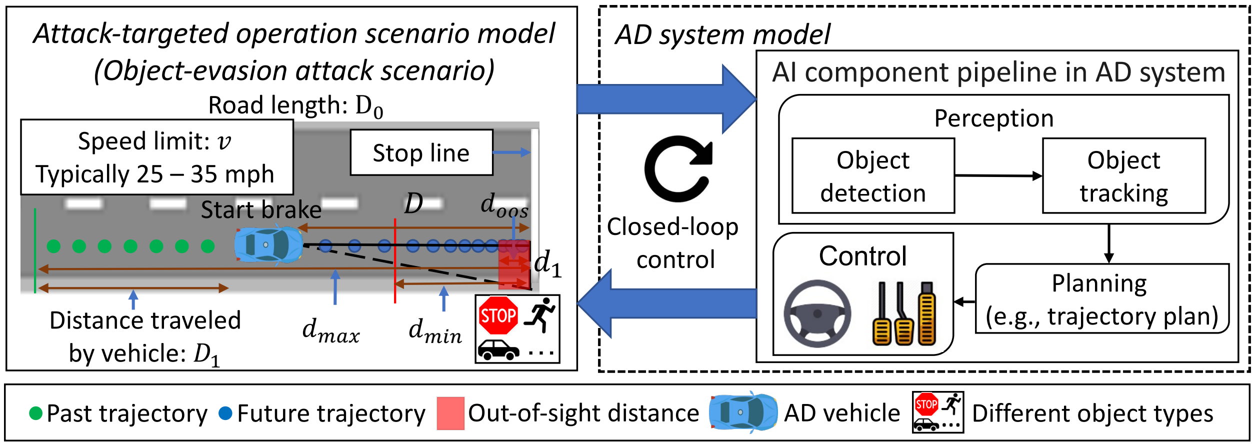


Figure 5.2.1: Illustration of the system model for adversarial object-evasion attacks in AD context.

System model for adversarial object-evasion attacks. Fig. 5.2.1 illustrates the aforementioned system model defined for the adversarial object-evasion attack. The AD system model for object detection, the targeted AI component in adversarial object-evasion attacks, mainly includes its downstream tasks of object tracking, planning, and control, and closed-loop control. The vehicle plant model mainly includes the physical properties related to longitudinal control, e.g., the minimum brake distance (d_{min}), and the distance to the stop line (stop to avoid violating traffic rules or crashes) where the stop line is out of sight in the camera image d_{oos} (depending on the hood length and the camera mounting position). The operation scenario model includes the speed limit, lane width, the relative positioning and facing of the object to the ego lane, the driving norm that the vehicle typically drives at constant speed before it starts to see the object (d_{max}), and the system-level attack goal that triggers the traffic rule violation (i.e., hit into the object or exceeding the stop line). We will use this system model in our studies in the following sections. There exists several example attacks for the system model such as STOP sign-evasion attack, which is the most extensively-studied physical adversarial object evasion attack in AD context [252], and thus will be the main focus of our study in later sections; pedestrian-evasion attack [305]; car-evasion attack [277]; etc.

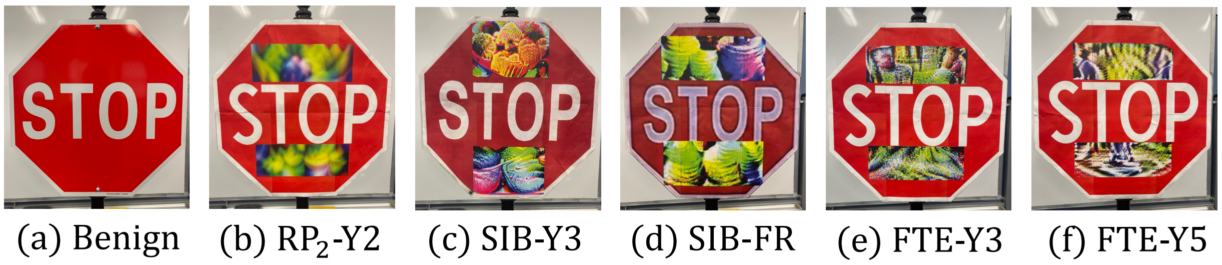


Figure 5.3.1: Visualisation of STOP signs attack reproduction (in Table 5.3.1) for measurement study in physical world.

5.3 System-Level Effect of Prior Works

Scientific gap in existing works: Lack of system-level evaluation. Despite a plethora of published attack works on physical adversarial object evasion attacks in AD context (§5.2), we find that actually *all of them* only evaluate their attack effect *at the targeted AI component level* (i.e., judged by per-frame object misdetection rates [110, 305, 330, 148]), without any evaluation *at the system level*, i.e., with the corresponding system models for such attacks as described in §5.2. However, in the Cyber-Physical System (CPS) area, it is widely recognized that in AD system, AI component-level errors do not necessarily lead to system-level effects (e.g., vehicle collisions) [99, 247, 150]. Thus, without system-level evaluation, it can be highly difficult to scientifically know whether the attack is actually meaningful from the end-to-end AD driving perspective. We view this as a critical scientific gap in this current research space, and to address this, we perform a measurement study on the existing works about their system-level effects. We choose to focus on *adversarial STOP sign-evasion attck* as our target due to its high representativeness in this research space and also its direct impacts on driving correctness and road safety (§5.2).

5.3.1 Attack Formulation and Selection

Attack formulation. We assume that the attacker can arbitrarily manipulate pixels within restricted regions known as adversarial patch attack [63, 330, 110]. Such a patch attack is easy to deploy in the real-world and very stealthy. We consider the patch stays on the STOP sign shown in Fig. 5.3.1.

Selection of prior STOP sign attack works and their reproduction. There are various prior works on physical adversarial STOP sign-evasion attacks [188, 148, 110, 330, 309, 187, 83]. To perform our system-level effect measurement, we select the most effective ones at AI component level as representative examples. Four model designs (including one-stage and two-stage object detectors in §5.2) have been covered. For each model, we select the most effective attack design published so far which are shown in Table 5.3.1. However, all the STOP sign attacks in Table 5.3.1 do not provide the source code. Since we tried to contact the authors of the attacks for the source code but they all cannot provide it, we try our best to reproduce some of the works. Currently, we only have the reproduction for RP_2 and FTE. For SIB, we directly use the STOP sign images shared by the authors of that paper used for their physical-world experiments. We print the high-resolution STOP signs on multiple ledger-size papers and concatenate them together to form full-size real STOP signs which are shown in Fig. 5.3.1.

To demonstrate the reproduction correctness, we utilize their original evaluation setups for our trials. Our results are generally similar to theirs confirming the correctness of reproduction. For instance, the original RP_2 paper [110] reports an attack success rate of approximately 63.5% from 0 to 30 feet. With the same setup (outdoor), our results provide a 61.0% attack success rate — nearly mirroring the original. Note that SIB attack on the FR in Table 5.3.3 seems anomalous: it records around 47% attack success rate only from 40 to 45 meters, while consistently registering 0% in others. Despite the patch being provided by the

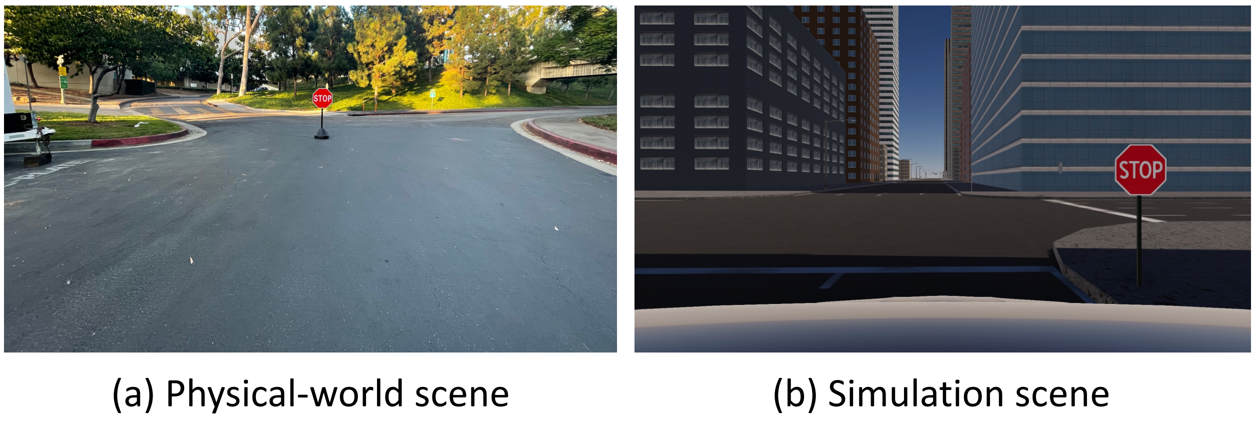


Figure 5.3.2: Experiment scenes. (a) Real-world scene with real road and injected STOP sign; (b) SVL simulation scene with the San Francisco map in a sunny day at noon.

authors, the pre-trained FR can be different, where we use MMDetection [82], a PyTorch-based object detection toolbox. Given such potential low transferability, the attack may be less effective compared to their original results. However, this is our best effort to reproduce their results faithfully.

5.3.2 Measurement Methodology and Setup

To measure system-level effects, we adopt a simulation-centric evaluation methodology, which has been widely adopted both in academia [275, 238] and in industry [287, 146] due to the inherent limitations of real-road AD testing in cost, safety, efficiency, and corner-case coverage. In this study, we use SVL, a production-grade high-fidelity AD simulator designed for AD systems [234]. As repeatedly demonstrated in various prior works, the end-to-end AD system-level evaluation results in SVL can highly correlate with the same setup tested in the physical world [275, 238]. To ensure the fidelity of our evaluation results, we improve the fidelity of the rendering process by modeling the perception results in the real world with a practical setup (details below). Note that the attacks themselves are agnostic to map and time by design, and thus are not generally affected by their changes. In SVL, we use San Francisco map on a sunny day at noon, which is the most representative setup.

Perception results modeling from physical world. To enhance the perception fidelity of simulators, we model the perception results using a practical setup in the real world. This approach represents our best effort to improve the fidelity of the simulation due to the experimental feasibility. Previous studies collect video frames by directly moving towards the STOP sign and simulate varying view angles by rotating the STOP sign itself. This approach is not practical since the vehicles do not directly drive towards the STOP sign, and the STOP sign should instead be located on the roadside as shown in Fig. 5.2.1. To improve such unrealistic setups, we follow the system model defined in §5.2. We recorded several pieces of video along the driving direction D using an iPhone 12 Pro Max starting from 45 m to 4 m (4 m is the d_{oos} in §5.2). We choose 45 m since 1) it is the minimal brake distance for speed above 50 mph, which exceeds the usual maximum speed of STOP sign areas, and 2) it is already much larger than the maximum distance evaluated in all the prior STOP sign-evasion attack works. We separate the whole range into 9 pieces, each spanning 5 m except the one near the STOP sign, which is 1 m long. Then, we record a video in each region and feed the video into the object detectors to model the perception results. We perform these experiments on sunny days as shown in Fig. 5.3.2. With that, we perform perception results injection at the output of the object detection task in the AD system, i.e., first read the ground-truth STOP sign detection results from the simulator and then drop/keep the detection results based on detection rate. For instance, if the attack success rate is 60% for a range, we will randomly drop the STOP sign detection results with a possibility of 60% in that range.

Evaluated AD system pipeline. The AD system pipeline includes representative downstream tasks after object detection, which contains 1) a tracking step using a general Kalman Filter based multi-object tracker [190]; 2) a planning step using a lane-following planner from Baidu Apollo [54], an industry-grade full-stack AD system; and 3) a control step using classic controllers, i.e., PID for longitudinal control used in OpenPilot [217], a production-grade Level-2 AD system, and Stanley [138] for lateral control.

Table 5.3.1: Selection of the representative prior works. Specifically, for each of the 4 model types targeted by prior works, we select the most effective attack design published so far.

Model	YOLO v5 (Y5)	YOLO v3 (Y3)	YOLO v2 (Y2)	Faster RCNN (FR)
Attack	FTE [148]	SIB [330]	RP_2 [110]	SIB [330]

Table 5.3.2: System-level violation rate in the simulation-based testing and component-level overall ASR for model Y2, Y3, Y5, and FR in benign and attacked scenarios. 10 runs for each cell with different initial AD position. B: benign; Sys: system; Comp: component; ASR: attack success rate.

Eval. level	Speed (mph)	Y2		Y3			Y5		FR	
		B	RP_2	B	SIB	FTE	B	FTE	B	SIB
Sys (violation)	25, 30, 35	0%	0%	0%	0%	0%	0%	0%	0%	0%
Comp (ASR)	Overall	-	71.2%	-	53.1%	53.3%	-	41.0%	-	5.2%

Speed selection. The driving speed is from 25 to 35 mph, with a step size of 5 mph, which is the most common speed range for STOP sign-controlled roads in the real world. 25 mph [57] is the common speed limit for the STOP sign-controlled road intersections, which is more likely to avoid a crash, and 35 mph [67] is the most common speed limit for city streets, which STOP signs are designed for.

5.3.3 Measurement Results

We evaluate the targeted AD system-level attack effect, i.e., STOP sign violation rate, by defining the STOP sign violation rate as $\frac{N_{\text{violation}}}{N_{\text{total}}}$, in which $N_{\text{violation}}$ means the number of runs where the AD vehicle exceeds the stop line and N_{total} is the number of total runs. Table 5.3.2 shows the results where each speed has 10 runs with random initialization of the AD vehicle position while the perception results modeling from real world is shown in Table 5.3.3. To our surprise, *none* of the existing representative attacks can trigger STOP sign violations in *any* of the common speeds for STOP sign-controlled roads when the benign performs well, though most of the attacks are effective in the component (i.e., with about 45% average attack success rate across the 5 attacks). After inspecting the details, we find that

Table 5.3.3: Detection rates of different objectors in benign, RP_2-, SIB-, and FTE-attacked scenarios tested in the physical world for perception results modeling (shown in §5.3.2). Each detection rate below is calculated with at least 400 video frames.

OD \ Dis. range (m)		4 - 5	5 - 10	10 - 15	15 - 20	20 - 25	25 - 30	30 - 35	35 - 40	40 - 45
YOLO v2 (Y2)	Benign	100%	100%	71.3%	31.3%	0%	0%	0%	0%	0%
	RP_2 [110]	58.2%	90.0%	76.2%	34.6%	0.1%	0%	0%	0%	0%
YOLO v3 (Y3)	Benign	100%	100%	100%	100%	80.1%	11.8%	6.7%	1.0%	0%
	SIB [330]	93.7%	100%	100%	90.4%	38.2%	0%	0%	0%	0%
	FTE [148]	89.9%	100%	100%	87.3%	42.9%	0.6%	0%	0%	0%
YOLO v5 (Y5)	Benign	100%	100%	100%	100%	98.7%	89.4%	52.3%	25.3%	0%
	FTE [148]	91.2%	100%	100%	99.7%	88.2%	48.4%	3.9%	0%	0%
Faster-RCNN (FR)	Benign	100%	100%	100%	100%	100%	100%	100%	100%	100%
	SIB [330]	100%	100%	100%	100%	100%	100%	100%	100%	53.2%

the STOP sign is always tracked at the object tracking step before reaching the minimum brake distance of the AD vehicle due to the low attack success rate in such regions. Taking SIB attack on Y3 as an example, the brake distance for 30 mph is around 15 m. In the benign scenario, the detection rate for 15-20 m is 100%, while the SIB attack can still have 90.4% detection rate as shown in Table 5.3.3, which is not enough to make the tracking vanish before the minimum braking distance.

5.4 System-Driven Attack Design

After realizing that existing works cannot provide any system-level violation in AD context, we propose SysAdv, a system-driven attack design, which can be integrated with all the existing attacks to improve system-level effects.

5.4.1 System-Driven Attack Design Framework

For the attack design in the prior works [110, 330, 148, 83], we can abstract the key part for attack generation:

$$\arg\min_{p_a} \quad \mathbb{E}_{s\sim\mathcal{S}}[\mathcal{L}(M(p_a, O, s, B), \gamma)] \tag{5.1}$$

$\mathcal{S}$ is the distribution to sample different object sizes in pixels, which is a very important factor in achieving the robust attack at different distances between the AD vehicle and the object. The $\mathcal{L}$ is the loss function used in the prior attacks to achieve high attack effectiveness, p_a is the adversarial patch, O is the object, and function $M(p_a, O, s, B)$ indicates applying p_a to O, then resizing object size in pixel to s, and applying O into the background B, γ means other inputs for loss function in the prior works (e.g., the bounding box information and threshold) related to the object detector alone. After investigation on Eq. (5.1), we find out that the system model can be involved into two parts, i.e., $\mathcal{S}$ and function $M(.)$, which do not rely on object detector alone.

After exploring all the prior works, we discover that all of them do not consider such a system model information into their attack designs, which hinder them to achieve potent system-level effects in AD context. Thus, we propose two novel system-driven designs to significantly improve the system-level effects. Specifically, we involve the system model information into $\mathcal{S}$ and function $M(.)$ from Eq. (5.1).

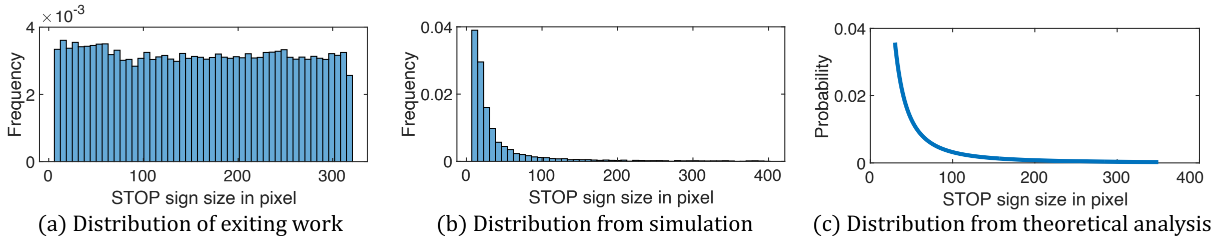


Figure 5.4.1: Different STOP sign size distribution: (a) state-of-the-art existing work [148], (b) our experimental analysis, and (c) our theoretical analysis.

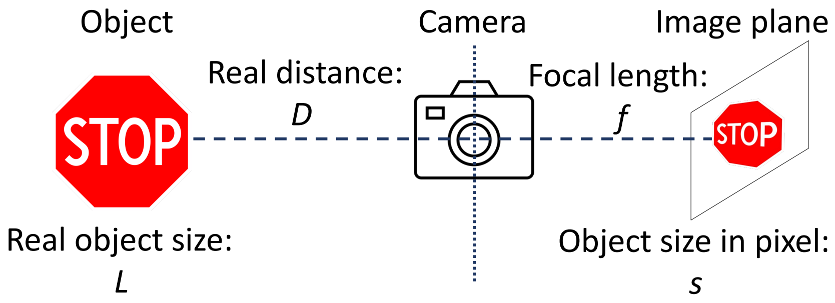


Figure 5.4.2: Theoretical analysis of §5.4.2, i.e., the camera pin-hole model.

5.4.2 Physical Model-Inconsistent Object Size Distribution in Pixel Sampling

In the prior works [330, 148, 110], to make the attack robust to different distance, Expectation over Transformation (EoT) [55] is used to *uniformly* sample the object size ($\mathcal{S}$ in Eq. (5.1)) in a certain range [83, 148, 55]. However, with system model (§5.2), we find that this assumption is not held, which leads to the first observation: *physical model-inconsistent object size distribution in pixel sampling.* To justify the observation, we perform the experimental and theoretical analysis with STOP sign system model as an example.

Experimental analysis. With the same setup in §5.3, we simulate the real driving scenario in SVL. The STOP sign size in pixels and the distance between the vehicle and the STOP sign can be directly obtained from SVL (§5.3.2). With that, we can get the frequency distribution histogram over different STOP sign sizes in pixels as shown in the Fig. 5.4.1 (b), in which the AD vehicle runs for 30 rounds at speed 25 mph. The distribution shown in Fig. 5.4.1 (b) is not uniform, which is wrongly assumed by the prior works [83, 148]. To compare, we sample the STOP sign size from the most recent prior work [148], which designs an algorithm to determine the STOP sign size in a *uniform* way. We run that algorithm 30k times and collect the STOP sign size shown in Fig. 5.4.1 (a). The difference between Fig. 5.4.1 (a) and (b) indicates that our observation is held experimentally.

Theoretical analysis. Assuming a uniform motion for AD vehicles, we leverage the camera pin-hole model (Fig. 5.4.2) for the theoretical analysis. From Fig. 5.4.2, we abstract the relationship of real object size (L), real distance(D), focal length(f), and object size in pixel(s) with similar triangles: $\frac{L}{D} = \frac{s}{f}$. With the system model shown in Fig. 5.2.1, we assume that the initial vehicle to STOP sign distance is the road length D_0 and the current vehicle to STOP sign distance is D. Due to uniform motion, the vehicle traveled distance can be formulated as $D_1 = v * t$, where v is the vehicle speed (usually it is the speed limit) and t is the time. To build the relationship between s and the sampled frequency (i.e., the frame number) F, we formulated the time t as $t = \frac{F}{\eta}$, where the η is the image capturing frequency from the camera. Due to $D_1 + D = D_0$ and the camera pin-hole model (Fig. 5.4.2), we can obtain the following equation:

$$D_0 = D + v * \frac{F}{\eta} = \frac{L * f}{s} + v * \frac{F}{\eta} \rightarrow F = (D_0 - \frac{L * f}{s}) * \frac{\eta}{v} \tag{5.2}$$

Eq. (5.2) is the CDF of s, since the F is accumulated frames. To obtain PDF, CDF's derivative is calculated:

$$F' = \frac{dF}{ds} = \frac{\eta * L * f}{v * s^2} \tag{5.3}$$

From Eq. (5.3), the probability distribution is definitely not uniform. We also plot Eq. (5.3) as shown in Fig. 5.4.1 (c) with $\eta = 20$, $L = 1.5$, $v = 25mph$, and $f = 25mm$ (commonly used in AD system such as Baidu Apollo). The distribution is similar to the distribution in the experimental analysis shown in Fig. 5.4.1 (b), which supports our observation.

Our system-driven solution (S1). With that, we propose our system-driven solution (S1) to address this inconsistency above. Leveraging the system model in Fig. 5.2.1, we define a novel object size distribution based on Eq. (5.3): $\mathcal{S} = \{s_1, s_2, ..., s_N\}$ as a discrete distribution, where s_i is the object size in pixels. Based on the Eq. (5.3), the probability of s_i can be abstract as $p(s_i) = \frac{1}{s_i^2} / \sum_{k=1}^{N} \frac{1}{s_k^2}$. Such new object size distribution can be used to address the inconsistency observation and easily integrated into the attack design (Eq. (5.1)). However, to get the detailed distribution, we have to know the range of $\mathcal{S}$, which will be addressed in the following system-driven solution (S2) in §5.4.3.

5.4.3 Lack of Vehicle Plant Model and AD System Model Consideration

In the EoT process, uniformly sampling the object size ($\mathcal{S}$ in Eq. (5.1)) in a *range* is generally used. In the prior works, they just treat it as hyper-parameters without any reasons [83, 148]. In practice, not every range is equivalently important to achieve system-level effects. Taking STOP sign case as an example, within d_{min} in Fig 5.2.1, despite applying maximum deceleration, AD vehicle still cannot fully stop before the stop line. Thereby, such a range is not important to achieve system-level effects. However, none of the prior works involve the system-critical range related to *the vehicle plant model* and *AD system model* in their attack designs, which leads to the second observation: *lack of vehicle plant model and AD system model consideration.*

Note that previous studies which indiscriminately utilize a broad range of object sizes for attacks have exhibited reduced effectiveness in comparison to those employing a small size range. For example, when the object size is small (implying the AD vehicle is far away from the object), attack convergence becomes challenging [148], which indicates that it is harder

Table 5.4.1: Attack success rate of RP$_2$ for Y2 evaluated in simulation with both small and large STOP sign pixel sizes.

		Distance (m)							Ave
		4 - 5	5 - 10	10 - 15	15 - 20	20 - 25	25 - 30	30 - 35	
Comp. ASR	Small	6.7%	37.1%	68.3%	81.1%	100%	100%	100%	70.5%
	Large	98.6%	6.1%	0%	1.0%	58.5%	99.1%	100%	51.9%

to attack. Therefore, generally, utilizing a more optimally defined range, as opposed to an excessively broad one, enhances the efficacy of the attack.

To elucidate the disparity in attack effectiveness between utilizing a broad versus a narrow object size range, we conducted experiments comparing the attack success rates with small and large range of the STOP sign size. We follow a similar evaluation setup as in §5.3.2 but use a pure simulation-based setup for RP$_2$ attack. Specifically, the small range for the STOP sign spans from 30 px to 100 px, whereas the large range extends from 30 px to 416 px, representing the maximum range at which the benign STOP sign is detectable. Results presented in Table 5.4.1 reveal a superior average attack success rate for the small range over the large range. Although the large range demonstrates promising convergence at close distances, its performance diminishes between 5 to 30 m. This suggests that simply opting for a larger range does not guarantee enhanced performance.

Our system-driven solution (S2). We introduce our Solution (S2) to ascertain the system-critical range from the *vehicle plant model* and the *AD system model*. With these, we directly deduce the d_{min} and d_{max} values as shown in Fig. 5.2.1. Then, we convert these distances to the corresponding object sizes in pixels ($\mathcal{S}$). From the system model (Fig.5.2.1), it's evident that the minimum braking distance can be used as d_{min}. Within this distance, detection results have a negligible impact on system-level effects. As for the d_{max}, several tasks in the AD system, such as object detection and tracking, can influence its determination. For object detection, the maximum distance can be the furthest benign distance where an object is detected. For object tracking, we select a conservative tracking [150] since attackers might not always access the precise tracking parameters of the targeted AD system and a conservative tracking provides a broader system-critical range generally. To achieve system-level effects, the object should not be tracked when the vehicle reaches the d_{min}. Due to conservative tracking, such tracking distance (i.e., if within this distance, the object can never be detected, the tracker will be deleted) usually exceeds the distance where the object detector can detect the object. Thus, simplifying this, we select the distance where the benign object can be detected with a small detection rate as d_{max}. Having deduced the d_{min} and d_{max}, the next step involves translating these distances into pixel object sizes ($\mathcal{S}$) and determining the appropriate object location in the background (function $M(.)$ in §5.4.1). We suggest two methodologies to solve address it: 1) camera-based rendering [301, 66, 70] and 2) manual annotation [305]. Employing these methods allows us to acquire precise specifications about position and pixel size range (in §5.5.1 and §5.5.3).

5.5 Evaluation

We adopt the same evaluation methodology and setup as §5.3.2. The printed STOP signs with the newly generated patches are in Fig. 5.5.1. We evaluate some attacks on one-stage object detectors, i.e. Y2, Y3, and Y5 due to their better real-time performance compared to two-stage ones [338]. RP_2 and FTE are selected as the evaluated attacks. Attack generality is evaluated in §5.5.2 and §5.5.3. The combination for attacks and object detectors are RP_2, FTE-Y3, and FTE-Y5.

5.5.1 System-level Attack Effectiveness Evaluation

Attack generation. We adopt the attack methodology in §5.4. We employ a camera-based rendering method and utilize the nuScenes dataset [66] to translate the system-critical range from the physical world to the pixel range in images. Notably, nuScenes offers APIs that

Table 5.4.2: Attack success rates (ASRs) of RP_2, FTE-Y3, and FTE-Y5 on STOP sign-evasion attack (§5.5) and ADV-Tshirt on pedestrian-evasion attack (§5.5.3) for our attack design evaluation with perception results modeling from physical world. + S1: with S1 only; + S2: with S2 only; + S1 + S2: with S1 and S2; + S1 + S2 (TV): + S1 + S2 with TV loss (§5.5.1). **Bolded** numbers indicate the cases where our design outperforms the original baseline attack ("Original") within the system-critical range.

			Dis. (m): Gray: ASR within the system-critical range								
OD		Attack design	4 - 5	5 - 10	10 - 15	15 - 20	20 - 25	25 - 30	30 - 35	35 - 40	40 - 45
Y2	RP_2	Original [110]	41.8%	10.0%	23.8%	65.4%	99.9%	100%	100%	100%	100%
		+ S1	4.4%	**13.7%**	**51.2%**	**99.3%**	100%	100%	100%	100%	100%
		+ S2	5.6%	**44.9%**	**57.8%**	**98.7%**	100%	100%	100%	100%	100%
		+ S1 + S2	36.1%	**65.8%**	**88.0%**	**100%**	100%	100%	100%	100%	100%
Y3	FTE-Y3	Original [148]	10.1%	0%	0%	12.7%	57.1%	99.4%	100%	100%	100%
		+ S1	0%	0%	0%	**14.0%**	**72.2%**	95.9%	100%	100%	100%
		+ S2	0%	0%	0%	**13.4%**	**81.4%**	94.4%	97.2%	100%	100%
		+ S1 + S2	5.3%	0%	**34.7%**	**94.0%**	**99.4%**	**100%**	100%	100%	100%
Y5	FTE-Y5	Original [148]	8.8%	0%	0%	0.3%	11.8%	51.6%	96.1%	100%	100%
		+ S1	0.3%	0%	0%	**1.3%**	**13.9%**	**69.3%**	94.1%	99.0%	100%
		+ S2	1.5%	0%	**0.1%**	**1.7%**	**32.7%**	**81.9%**	**99.0%**	100%	100%
		+ S1 + S2	16.5%	0%	**4.3%**	**47.2%**	**93.4%**	**99.7%**	**100%**	100%	100%
		+ S1 + S2 (TV)	43.6%	**51.7%**	**42.1%**	**26.3%**	**23.8%**	**66.1%**	**97.7%**	99.7%	100%
Y2		Original [305]	13.5%	0%	31.3%	86.1%	96.1%	90.7%	86.0%	100%	100%
		ADV-Tshirt + S1 + S2	0%	0%	**34.2%**	**89.0%**	91.7%	83.5%	78.5%	98.7%	100%
Y3		Original [305]	3.8%	0%	3.8%	32.2%	75.7%	89.8%	90.5%	91.5%	95.1%
		ADV-Tshirt + S1 + S2	0%	0%	**33.6%**	**88.2%**	**91.3%**	**92.4%**	89.7%	90.7%	87.7%
Y5		Original [305]	35.9%	6.8%	17.1%	36.7%	37.4%	72.0%	88.6%	92.3%	91.5%
		ADV-Tshirt + S1 + S2	2.6%	1.0%	**61.6%**	**74.7%**	**58.3%**	**89.6%**	**90.5%**	64.1%	61.1%

facilitate rendering objects within images. Specially, we render the four corners of the STOP sign and obtain its size in pixels by measuring the distance between these four corner points in the image. With S1 and S2, we can embed the system-model property into the attack generation process to improve system-level effects. To further validate the effects of S1 and S2, we perform ablation studies by generating the attack with S1 only and S2 only, and comparing them to the attack generated with/without both S1 and S2. Details of attacks without S1 and S2 (i.e., original attacks) are in §5.3.

Table 5.5.1: System-level violation rate tested in simulation and component-level ASR evaluation including baseline comparison (i.e., Original and ablation studies). Each cell contains 10 runs with different initial positions of the AD vehicle. S1: with S1 only; S2: with S2 only; S1 + S2: with S1 and S2; SCR: System-critical range (§5.4.3). * with special improvements (§5.5.1).

Eval. level	Speed (mph)	RP_2				FTE-Y3				FTE-Y5			
		[110]	S1	S2	S1 + S2	[148]	S1	S2	S1 + S2	[148]	S1	S2	S1 + S2
System (violation rate)	25	0%	90%	100%	100%	0%	0%	0%	40%	0%	0%	0%	10%*
	30	-	-	-	-	0%	0%	30%	100%	0%	0%	0%	80%
	35	-	-	-	-	-	-	-	-	0%	30%	40%	100%
p-value		-	0.00	0.00	0.00	-	-	0.08	0.00	-	0.08	0.04	0.00
Component (ASR)	Overall	71.2%	74.2%	78.6%	87.8%	53.3%	53.6%	54.0%	70.4%	41.0%	42.0%	46.3%	62.3%
	SCR	33.1%	54.7%	67.1%	84.6%	33.8%	36.4%	37.8%	65.6%	26.6%	29.8%	35.9%	57.4%

Results. The STOP sign attack images are in Fig. 5.5.1, which are printed in physical world and the perception modeling results from the physical world are in Table 5.4.2. From the results in Table 5.4.2, almost all the results (bolded in the table) with our system-driven attack improvement can outperform the original attack. As shown in Table 5.5.1, with our system-driven attack designs, the system-level violation rate can increase by around 70% on average, where we only include the results where the benign cases have a 0% system-level violation rate. The p-value (Table 5.5.1) is generally at the statistically significant level (e.g., generally < 0.05 or at a similar magnitude, especially for S1+S2). With S1 + S2, the overall component attack success rate can increase by around 33% on average. Especially, in the system critical range, the attack success rate can increase by 122%, which can significantly improve the system-level effects. Taking FTE-Y5 at 35 mph as an example, the brake distance of 35 mph is around 20 m and the attack success rate from 20 - 35 m shown in Table 5.4.2 is around 98%, which shows a high chance to make the STOP sign not tracked before the brake distance, which leads to the 100% violation rate (Table 5.5.1).

For FTE-Y5 at 25 mph, due to the low effectiveness (i.e., around 4%) from 10 m to 15 m, the tracker cannot be deleted, which leads to 0% system violation. Thus, we provide

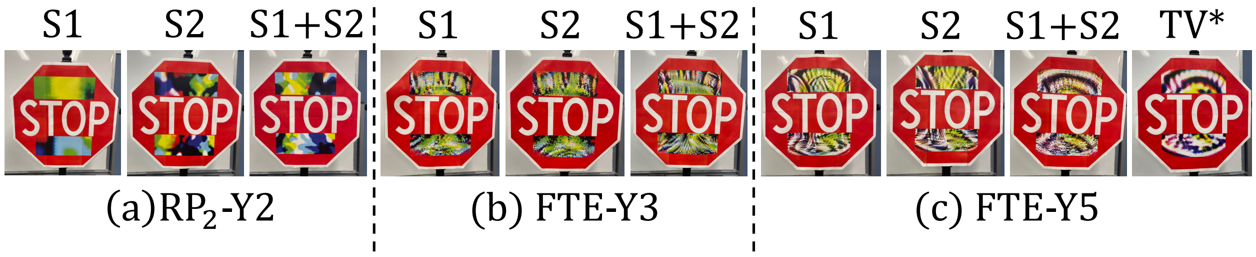


Figure 5.5.1: Visualization of STOP sign attacks with system-driven design. S1: with S1 only; S2: with S2 only; S1 + S2: with both S1 and S2; TV*: S1 + S2 with TV loss (§5.5.1).

Table 5.5.2: System-level violation rate tested in simulation on different AD parameter settings which are highly critical to the system-level effects. The perception modeling results from physical world are in Table 5.3.3 and Table 5.5.1.

Tracking param (H, R)	(4, 6) [54]		(3, 5) [160]		(4, 40) [334]	
Brake (m/s^2)	-3.4	-6.0	-3.4	-6.0	-3.4	-6.0
Original [148]	20%	0%	50%	0%	40%	0%
Ours	**100%**	**100%**	**100%**	**90%**	**100%**	**100%**

a *special improvement* by applying the total variation (TV) loss as prior works [110, 301] which benefits the attack effectiveness. The perception modeling results from the physical world are in Table 5.4.2 and the attack visualization is shown in Fig. 5.5.1. The system violation rate increases to 10% after improvement as shown in Table 5.5.1 with *. Based on results in Table 5.4.2, the attack success rate in a near distance is generally lower, which aligns well with the results of prior work [330]. This leaves space for future works: improving component attack success rate in the near distance.

The results of the ablation study are also summarized in Table 5.5.1. Although in the majority of cases, S1 cannot significantly improve the system-level effects (20% on average), the component attack success rate in the system-critical range is improved. Compared to S1, S2 has better results (around 28% on average). Only combining S1 and S2 can further benefit the system-level effects (around 70% on average), which shows the necessity of both S1 and S2.

Table 5.5.3: Pedestrian collision rate tested in simulation with ADV-Tshirt attack on different object detectors. 10 runs for each cell with different initial AD vehicle position.

Speed (mph)	YOLO v2 (Y2)		YOLO v3 (Y3)		YOLO v5 (Y5)	
	Original [305]	**Ours**	Original [305]	**Ours**	Original [305]	**Ours**
25	20%	**50%**	0%	**50%**	0%	**70%**
30	100%	**100%**	50%	**100%**	10%	**80%**
35	100%	**100%**	80%	**100%**	60%	**90%**

5.5.2 Generality on Different AD System Parameters

Methodology and setup. We select the most safety-critical parameters on system-level effects in AD systems for this evaluation including the tracking parameters (H, R), where the tracking creates a tracker for an object only when it is continuously detected for H frames, and deletes its tracker only when the object continuously disappears for R frames [336, 54, 160, 150], and the brake deceleration where we use the safe vehicle deceleration and max vehicle deceleration [93]. We select the tracking parameters from Baidu Apollo [54] and Autoware.AI [160], and the representative research paper [334]. All the parameter details are in Table 5.5.2 and for others, we follow the same setup in §5.5.1. We select the FTE-Y5 since it is the most representative attack so far and 35 mph as the target speed due to its high safety impact.

Results. The system-level attack effect results (violation rate) are summarized in Table 5.5.2, where we compared our attack with the original naive attacks (§5.3). The results show that our attack can outperform the original attack in all the different AD parameter settings on the system-level effect. On average, we have around 98% system violation rate (5 times larger than the original one) while the original naive attack only has 18%. The results further point out that our attack is general to different critical AD system parameters.

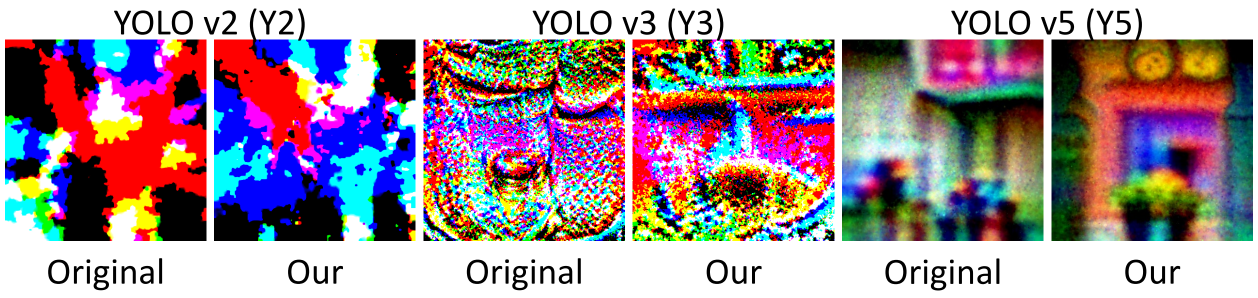


Figure 5.5.2: Visualization of ADV-Tshirt attack with and without system-driven design.

5.5.3 Generality on a Different Object Types

Methodology and setup. We select the "pedestrian" as our target object type since making the pedestrian vanish will cause a significant impact on AD. We select the most representative patch attack – adversarial T-shirt [305], which is called ADV-Tshirt in our paper. For object detectors, we select Y2, Y3, and Y5 and follow the same setup in ADV-Tshirt paper [305], and we collect the videos from the real world (similar methodology in §5.3) for attack generation and manually annotate the four corner points for placing the patch (obtaining the size and position §5.4.3). Each video segment has around 200 frames. We perform digital perception result modeling with real-world data we collected.

Results. The perception results modeling results for ADV-Tshirt are shown in Table 5.4.2 and the generated patches are visualized in Fig. 5.5.2. We define the system-level effect metric as pedestrian collision rate: $\frac{K_{\text{collision}}}{K_{\text{total}}}$, in which $K_{\text{collision}}$ means the number of runs where the AD vehicle crash into the pedestrian, and K_{total} is the number of total runs. The system-level evaluation with the comparison with the original attack [305] is shown in Table 5.5.3. In average, our attack designs can achieve around 82% pedestrian collision rate while the original attack can only achieve around 47% pedestrian collision rate. Especially, for the most advanced object detector such as Y3 and Y5, our pedestrian collision rate has significant improvement compared to the original attack. Y2 is more fragile than others which makes the original attack have very high attack effectiveness in the component level and leads to pedestrian collision rates at the similar level as ours. The results show the generality of our attack designs to different object types which further shows the generality to different system models (§5.2).

5.6 Discussion

Potential mitigation. The ongoing tug-of-war between adversarial attacks and their defenses has yielded a range of mitigation strategies, such as adversarial training [199]. Since several object-evasion attacks in AD context have been identified [330, 148], there is an immediate need for defense exploration. Before pursuing novel mitigation strategies, it is imperative to first measure how existing defenses affects system-level attack effectiveness in AD, especially the ones with theoretical guarantees [297, 295], which should be a future work. Another promising direction involves cross-checking with alternate perception sources. For example, AD systems might verify camera-based pedestrian detection with LiDAR perception. Despite not offering a fundamental defense strategy [70], they may make system-level attack effects more difficult to achieve. Thus, we leave a systematic exploration of these defenses to future work.

Limitation and future work. First, although we leverage the perception results that modeling from the physical world and demonstrate the system-level effects in AD system with LGSVL, the feasibility of the attack effects on real AD systems in physical world remains unclear. Thus, the exploration of the attack practicality is a valuable future work. Second, our attack is within white-box threat model, which is less practical compared to black-box one. Thereby, the development of a novel attack with a practical threat model is a potential future work. Third, although we explore the generality on different AD system parameters in §5.5.2, our evaluation results and findings are limited by the current AD system setups introduced in §5.3.2. Therefore, the system-level effect measurement on commercial AD systems such as Tesla is an important future direction.

5.7 Conclusion

In this paper, we ask whether previous works can achieve system-level effects (e.g., vehicle collisions, traffic rule violations) under real AD settings. Then, we perform the first measurement study to answer this research question. Our evaluation results show that all representative prior works cannot achieve any system-level effects in a closed-loop AD setup due to the lack of the system model. With our newly proposed system-driven designs, i.e., SysAdv, the system-level effects can be significantly improved. We hope that the concept of the system model could guide future security analysis/testing for real/practical AD systems.

Chapter 6

SlowTrack: Increasing the Latency of Camera-Based Perception in Autonomous Driving Using Adversarial Examples

6.1 Introduction

Autonomous Driving (AD) vehicles, manufactured by various companies, have become ubiquitous in our daily lives. For instance, numerous Tesla vehicles are equipped with the Autopilot feature [158, 265] running in the real world. For these vehicles, camera-based *perception* is pivotal, enabling them to detect real-time environmental objects such as pedestrians to ensure safety. Given its significance for safety and security, various prior works [70, 252, 239, 283] have studied its security, especially on integrity such as making the object vanished or changing the label of the objects to cause traffic rule violations or safety hazards. We refer to these as *system-level effects* throughout this paper.

Nevertheless, the *availability* aspect (real-time performance) of the system, which is crucial for safety (e.g., causing vehicle collision [275]) has been relatively underexplored, especially for the complete camera-based AD perception pipeline. While some existing AD security analysis has studied availability in object detection [249, 80], they do not encompass the entire AD perception since usually, object detection is a part of the AD perception [150]. In addition, in the Cyber-Physical System area, it is widely recognized that small component level errors do not necessarily lead to system-level effects [252, 283]. Thus, these studies leave a critical research gap: *their proposed attack strategies may not be effective enough to conduct system-level effects in end-to-end AD systems.* As we demonstrate later, existing attacks targeting only object detection do not consistently produce highly potent system-level effects due to lack of entire AD perception consideration.

To fill in this critical research gap, in this paper, we are the first to study availability-based adversarial attacks across the entire camera-based AD perception including both object detection and tracking. Our proposed novel attack framework, SlowTrack, is designed to increase the latency of camera-based AD perception. Instead of solely targeting object detection, which might not yield potent system-level effects due to the limited increase of the latency, we realize the untapped potential of object tracking response time to generate a much more effective latency attack. To illustrate, an attacker focusing only on object detection might attempt to dramatically increase the number of proposed bounding boxes [80]. Object tracking might filter out a majority of these boxes and in common object detection post-processing [155, 327], the maximum number of detection is provided to ensure performance. Thus, the effectiveness of these attack is limited. Due to the importance of object tracking, we first perform availability attack surface analysis by theoretically analyzing the time complexity of the state-of-the-art representative tracking algorithms. Then, we propose a two-stage attack strategy and formulate the attack as an optimization problem, shown in

Fig. 6.3.1. Additionally, our novel loss function designs, encompassing score loss, bounding box area loss, and feature match loss, fully leverage the entire tracking-by-detection pipeline to generate effective latency-based attack.

Our experimental evaluation of SlowTrack targets four state-of-the-art camera-based AD perception pipelines. We find that SlowTrack, when compared with existing object detection latency attacks, provides significant improvements in latency under comparable perturbation levels. For instance, SlowTrack on average induces latency 2.9 times more than that for existing approaches [80, 249]. We also demonstrate the system-level effects of our SlowTrack using Baidu Apollo [52] LGSVL [234] AD simulator. The results show that SlowTrack induces a 70% vehicle crash rate in two representative AD scenarios while existing methods achieve only a 30% rate. Demo videos are at the project website: https://sites.google.com /view/cav-sec/slowtrack

To sum up, our contributions are as follows:

- We are the first to study availability-based adversarial attacks considering the entire AD perception pipeline and find that previous object detection-based latency attack strategies may not induce potent system-level effects.
- We propose a novel attack framework SlowTrack to systematically generate the latency adversarial attacks on camera-based AD perception by designing a two-stage attack strategy and proposing three novel loss functions.
- SlowTrack is tested on four popular camera-based AD perception pipelines across different hardware, showing increase in latency and boost in system-level effects.

6.2 Background and Related Work

Camera-based AD perception. In the AD system, camera-based perception is primarily constituted by object detection and multi-object tracking (MOT) [52, 160]. This process aims to identify objects in each image frame and track their movement over time [150]. *Tracking-by-detection* has become the dominant MOT paradigm [326] and is widely used in industry-grade full-stack AD systems such as Baidu Apollo [52] and Autoware.AI [160]. It incorporates a detection module, a data association module, and a tracker management module. The detection module identifies objects in an image, noting their location, confidence, class scores, as well as other features for later data association. Data association then compares these detection with existing trackers based on features such as location and appearance, matching them based on similarity.

Tracking-by-detection, despite varying in matching strategies, shares a similar tracking management [150] to build and delete the moving trajectories, called trackers, and mark trackers and detection boxes as different states. Specifically, unmatched detection boxes are marked as unconfirmed and will be deleted unless they are continuously detected for H frames. Matched trackers are marked as re-find or remain activated depending on the trackers' previous states, while unmatched trackers are marked as lost, and will be deleted if no objects are associated with them for R frames. All of these trackers involved in matching constitute tracker_pool and trackers with activated states are outputs.

Prior works [150, 192, 252] show that MOT poses a general challenge to cause AD system-level attack impact for existing attacks that target object detection since MOT is designed to be robust against errors in object detection. Given this challenge, our work introduces an innovative latency attack against the most representative and popular MOT (tracking-by-detection) and demonstrates the heightened system-level attack effects in AD.

Adversarial attack in AD. DNNs are vulnerable to adversarial attacks [75, 243, 191], which are maliciously crafted samples to force DNNs to misbehave. Various prior works have explored the adversarial attacks in AD [69, 240, 242, 241, 211, 97]. While a majority of these attacks target integrity, our research concentrates on availability, which is another critical problem in AD [275]. Although some attack works study availability [182, 80, 249, 278], none of them consider the whole AD perception pipelines, which leads to suboptimal system-level effects in AD [150].

Availability-based latency attack. Availability-based latency attack can induce delays in the outputting function. Such adversarial methods, when applied to DNN, have been investigated recently [249, 80, 182]. However, their oversight of MOT within AD perception restricts their potential to achieve potent system-level effects. Thus, in this work, we perform the first availability-based latency attack on the whole AD perception to significantly boost system-level effects.

6.3 Methodology

6.3.1 Availability Attack Surface Analysis

To understand the vulnerability of the tracking-by-detection paradigm to latency attacks, we analyze the time complexity of the main three key steps (detection, data association, and tracker management) presented in Algorithm 6.4. The time consumption of the image preprocessing and backbone network of the detector hinges upon the computational dimension [255] and the number of computations [139, 130]. Given that the dimensions of the input images are unchanged and the majority of the detection model activation values are inherently non-zero for the most of images [249], we do not prioritize the time complexity of this segment.

Algorithm 6.4: Tracking-by-detection

```
   Input: Video sequence V; detector Det; detection filter $F$
   Output: Activated trackers $\mathcal{T}$ of the video
 1 Initialization: $\mathcal{T} \leftarrow \emptyset$
 2 for frame $f_k$ in V do
       /* detection module */
 3     $\mathcal{D}_k \leftarrow \texttt{Det}(f_k)$
 4     $\downarrow O(n^2) + O(n)$
 5     $\mathcal{D}_{reserved} \leftarrow F(D_k)$
 6     $\downarrow O(m)$
       /* predict locations of trackers */
 7     for $t$ in $\mathcal{T}$ do
 8         $t \leftarrow \texttt{KalmanFilter}(t)$
 9     end
10     $\downarrow O(mn')$
       /* trackers management */
11     Details in [193]
12 end
13 Return: $\mathcal{T}$
```

The boxes obtained by the detection module usually need to be filtered before being passed to subsequent modules. Prevailing filtration techniques encompass non-maximum suppression (NMS) and score filtering with time complexity of $O(n^2)$ and $O(n)$ respectively, where n denotes the number of outputs from the detection network. However, since most tracking algorithms [327] confine the maximum number of detection to $|\mathcal{D}|_{max}$, the maximum time complexities of these are $O(|\mathcal{D}|^2_{max})$ and $O(|\mathcal{D}|_{max})$. Then, data association matches the reserved detection boxes and tracker_pool with features, and the time complexity of this process is $O(mn')$, where m represents the number of trackers in tracker_pool and n' represents the number of reserved detection boxes. In the tracker management module, trackers are created and deleted according to the matching results and are marked with different states with a time complexity of $O(m)$. Meanwhile, the velocity and location of the trackers are also updated with a time complexity of $O(m)$.

6.3.2 Threat Model, Formulation, and Attack Overview

Threat model. Our attack method assumes white-box settings for the detection model, wherein both its architecture and parameters are known. For the tracking, we do not force the attackers to know the specific parameters and implementation details, which is a similar threat model as prior latency attack works [249, 80].

Formulation. Detectors often restrict the maximum number of detection boxes. This constraint results in a worst-case time complexity for the detection module of $O(|\mathcal{D}|^2_{max})$, effectively reducing the impact of previous latency attacks on object tracking [80, 249]. Thus, we propose an attack methodology that focuses on increasing the latency of the subsequent tracking stage under the constraint of limiting the maximum number of detection boxes. For a camera-based perception pipeline P, given the original image x, the attack goal is to craft an adversarial example x^* to maximize the tracking pipeline latency, while keeping the added adversarial perturbations imperceptible. We formulate it as the following optimization problem:

$$\arg\max_{x^*} T(P(x^*)) \quad \text{s.t.} \begin{cases} |\mathcal{D}|_{max} = N \\ \Delta(x^*, x) \leq \epsilon \end{cases} \tag{6.1}$$

, where T indicates the time function. Our attack strategy is designed to create detection boxes that exploit vulnerabilities in the tracker management module of AD. Specifically, for avoiding the temporary loss of objects in consecutive video frames due to occlusion, etc., the lost tracker will not be deleted immediately until the object is lost for R consecutive frames. Leveraging this mechanism, we are able to inject more and more trackers into the tracking module by strategically creating detection boxes through carefully designed perturbations. The detection boxes that appear in each frame are not associated with existing trackers and cause new tracking boxes to be created. As a result, the worst-case time complexity of the tracking module under our attack method is $O(R|\mathcal{D}|_{max}^2)$, where $m = R|\mathcal{D}|_{max}$ and $n' = |D|_{max}$. Thus, it can lead to more computation cost than detection attacks in prior works.

Overview. In this paper, we propose SlowTrack, the first adversarial attack maximizing the latency of the whole camera-based perception pipeline, leveraging object tracking, which can significantly increase the latency of AD perception under the constraint shown in Eq (6.1). The two attack stages of SlowTrack are: 1) in the attack initialization stage, make detection boxes created as many new trackers as possible, which requires detection boxes not to be associated with existing trackers, 2) make the lost trackers re-found before they are deleted, and kept in track_pool, which requires detection boxes to be associated with corresponding trackers. Thus, we need to construct sets of detection boxes with the same matching features and make these sets of boxes appear or disappear in the corresponding video frames using adversarial attacks. To our best knowledge, representative tracking algorithms always use motion-based features for association. Thus, we divide the images into different regions and use the candidates in these regions as sets, which facilitates the inter-association of detection boxes within sets and the disassociation of detection boxes between sets.

The overview of SlowTrack is in Fig. 6.3.1. Assuming that we divide the image into 3 regions, in frame 0, we select the top region and make the detection boxes appear in it, which are initialized as new track boxes. Similarly, in frame $H = 1$ we make the fake detection boxes appear in the middle region, not associated with the existing track boxes, and make the tracker management module create new track boxes, while the track boxes in frame 0 do not disappear. These frames are not associated with existing frames, making the tracker management module create new trackers, and the trackers initialized in frame 0 are not deleted, allowing to inject $|\mathcal{D}|_{max}$ trackers into the track_pool. Meanwhile, we make the detection boxes of the corresponding region reappear before the R frames to keep the trackers. Since the tracker management module only initializes the detection boxes

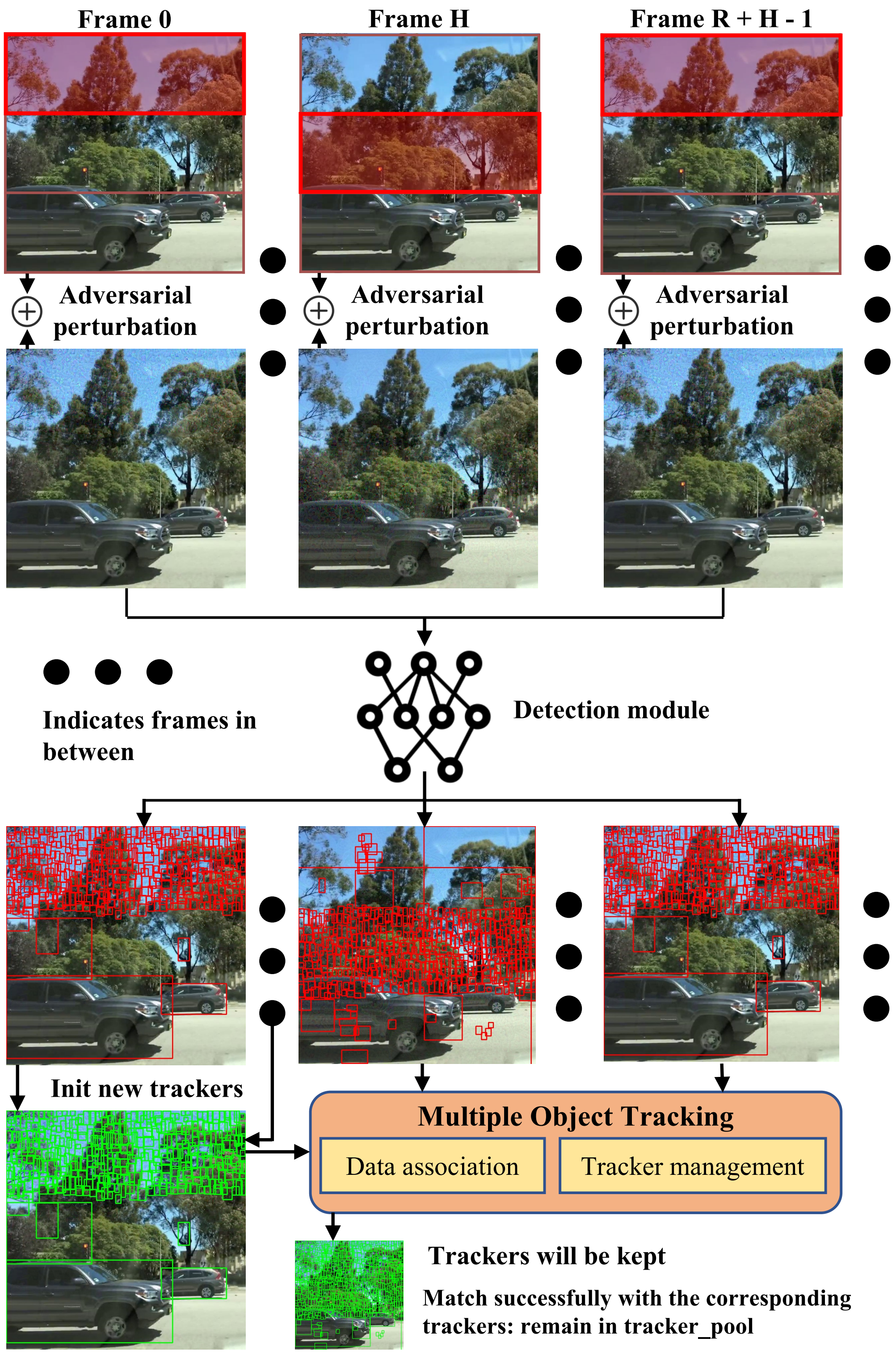


Figure 6.3.1: Overview of our SlowTrack attack.

that detected H frames consecutively, our attack strategy can be formulated: divide the image into K regions, each of which needs to be selected for H frames consecutively in the

Algorithm 6.5: Generate Attack Strategy

```
Input: A video sequence V = [$v_0, v_1, ..., v_{K-1}$]; reserved age $R$; hit count $H$
Output: attack strategy $\mathcal{S}$
/* $region\_idx$ is selected image region */
1  Initialization: $\mathcal{S} \leftarrow \emptyset$; $region\_idx = 0$; $n = 0$
2  while $n < K$ do
       /* $Re$ is the next time each region needs to be reactivated */
3      if $n == 0$ then
4          $\mathcal{S} \leftarrow \mathcal{S} \cup \{region\_idx\}$
5          $Re_{region\_idx} = n + R + 1$
6          $region\_idx = region\_idx + 1$
7      end
8      else
9          $Re_{min}, idx \leftarrow$ minimum value and index in $Re$
10         if $Re_{min} - n < H$ then
11             $\mathcal{S} \leftarrow \mathcal{S} \cup \{idx\}$
12             $Re_{idx} = n + R + 1$
13         end
14         else
15             for $i = 1$ to $H$ do
16                 $\mathcal{S} \leftarrow \mathcal{S} \cup \{region\_idx\}$
17             end
18             $Re_{region\_idx} = n + R + 1$
19             $region\_idx = region\_idx + 1$
20             $n = n + H - 1$
21         end
22     end
23     $n = n + 1$
24 end
25 Return: $\mathcal{S}$
```

initialization stage, and then be selected once more before R frames. We use the greedy algorithm with the maximum value of K: $R - H + 1$ (when $R - K = H - 1$, no region can be added).

Attack Strategy Generating. Algorithm 6.4 outlines the full process of Tracking-by-detection. We analyze the representative tracker management module, which is also used in some joint-tracking algorithms and propose the attack strategy generation algorithm shown in Algorithm 6.5. Specifically, we use a greedy algorithm to generate attack strategy to continuously inject different regions of detection boxes into the tracker management module. Such an attack strategy can also be generalized to different joint-tracking algorithm.

6.3.3 Loss Function Design

Score loss. To boost the number of selected boxes, it is necessary to raise the number of prediction candidates that bypass the detection filter, which selects candidates based on their confidence scores. Thus, to increase confidence score of selected candidates $\mathcal{C}_{sel}$, we propose a novel score loss:

$$\mathcal{L}_{score} = \frac{1}{|\mathcal{C}_{sel}|} \sum_{c \in \mathcal{C}_{sel}} \max((T_{conf} - c_{conf}), \lambda)$$

where T_{conf} represents the filtering confidence threshold set by the detection model, c_{conf} represents the confidence scores of the object detector, and λ is a hyper-parameter.

Bounding box area loss. To make more candidates to be reserved in the NMS employed by some detection filters, we need to compress the dimensions of the boxes to reduce the IOU between the candidates. This is expressed as:

$$\mathcal{L}_{area} = \frac{1}{|\mathcal{C}_{sel}|} \sum_{c \in \mathcal{C}_{sel}} \left(\frac{b_c^w \cdot b_c^h}{S_W \times S_H}\right)^2$$

where the bounding box is b, with b^w and b^h being its width and height. S_W and S_H are the width and height of the input image. This loss is added only when the filter contains NMS.

Feature matching loss. To successfully match the selected detection box with corresponding lost tracker so that the lost tracker is re-found before being deleted, the feature distance for the data association module needs minimizing.

$$\mathcal{L}_{match} = \Psi(\mathcal{T}_i, F'(\mathcal{C}_{sel}))$$

where Ψ is the feature distance function in data association, $\mathcal{T}_i$ represents i-th set of trackers, and F' represents detection filters without score threshold filtering.

Pairwise computation for obtaining feature distances could be expensive if feature extraction is complex or if the number of detection boxes is too large. Therefore, we propose a less computationally intensive method to make the corresponding trackers and detection frames match successfully. The images that need to appear with the same batch of detection boxes use the same perturbation. However, this makes the perturbation accumulation more obvious and the result of the attack decreases. Therefore, for balance, we use the universal perturbation method in the attack initialization stage and feature matching loss after that. Finally, the adversarial loss is represented by:

$$\mathcal{L}_{adv} = \lambda_1 \mathcal{L}_{score} + \lambda_2 \mathcal{L}_{area} + \lambda_3 \mathcal{L}_{match} \tag{6.2}$$

Similar to existing works [75], to make the perturbation invisible, we constrain the $\mathcal{L}_2$ norm:

$$\min_{x^*} \mathcal{L}_{adv} + \mu \|x^* - x\|_2$$

where μ is the hyper-parameter and x is the original image.

6.4 Experiments

6.4.1 Experimental Setup

Datasets and models. We use the BDD [246] and MOT17DET (MOT17) [207] datasets to evaluate SlowTrack. MOT17 includes 14 videos with more than 10K images and BDD contains 100K images with various attributes such as weather, scene, and time of day, resulting in a diverse dataset. For MOT17, we use all the data while for BDD, we randomly select 10 videos. As for the models, we select the most representative perception models: SORT (Y5) [61] (a Kalman filtering-based MOT generally used in AD [52, 252], with YOLO

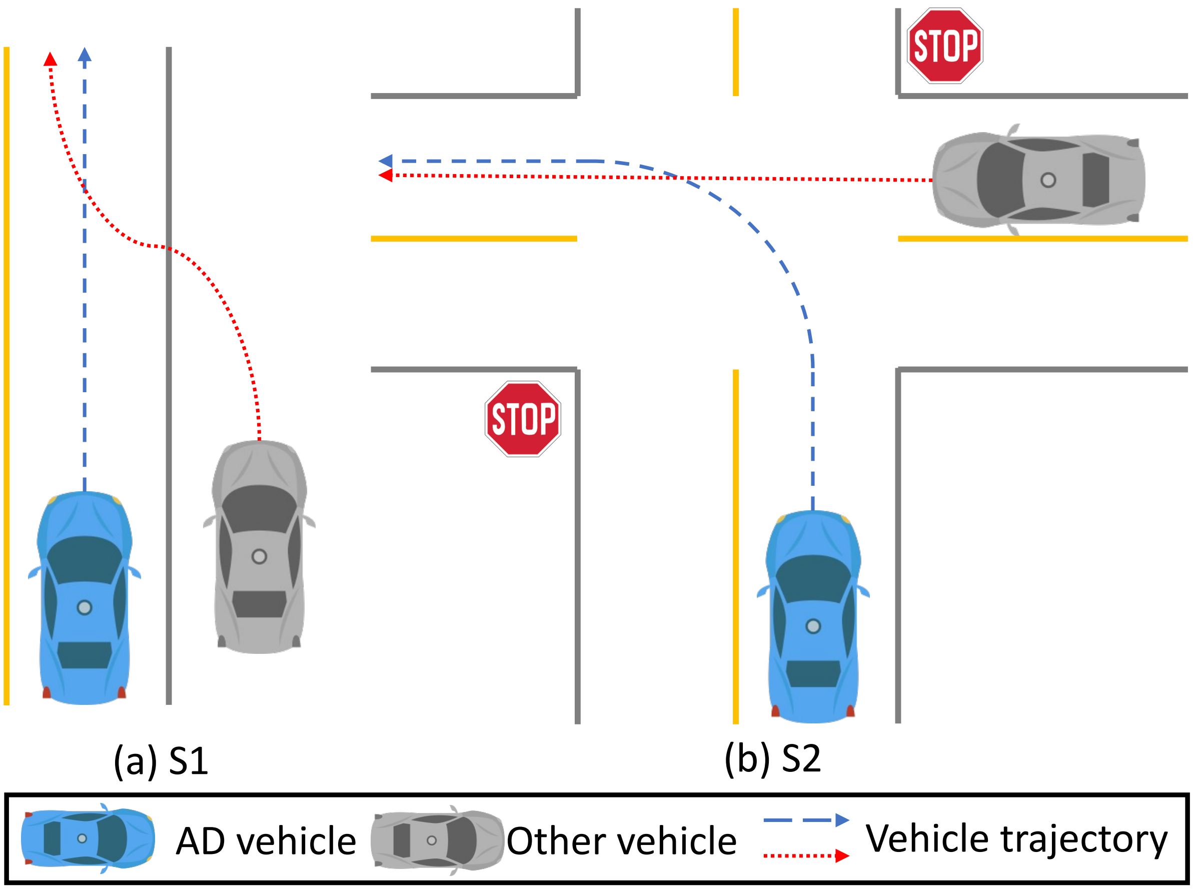


Figure 6.4.1: Two scenarios (S1 and S2) for our simulation evaluation setup on the system-level effects.

Model	Attack	Titan V	2080 Ti	3090
SORT (Y5)	PS	211	199	160
	Overload	406	411	337
	SlowTrack	**847**	**1082**	**1018**
FairMOT	PS	417	359	259
	Overload	970	914	685
	SlowTrack	**1848**	**1731**	**1726**
ByteTrack	PS	174	188	166
	Overload	379	427	330
	SlowTrack	**584**	**621**	**555**
BoT-SORT	PS	2395	2460	2372
	Overload	3093	3195	2485
	SlowTrack	**3768**	**4101**	**3245**

Table 6.4.1: Latency Time (ms) on MOT 17 dataset.

	w/o $\mathcal{L}_{score}$	w/o $\mathcal{L}_{area}$	w/o $\mathcal{L}_{match}$	SlowTrack
R-Track	0.0	153.2	171.4	225.5
R-Lat	0.0	34.0	38.3	50.9
#Track	0.1	63.9	66.9	73.5

Table 6.4.2: Ablation study for loss designs ($\mathcal{L}_{score}$, $\mathcal{L}_{area}$, and $\mathcal{L}_{match}$ in Eq. (6.2)) on R-Track, R-Lat, and #Track with SORT (Y5) and MOT17 using 2080 Ti. w/o: without

Model	S1			S2		
	PS	Overload	SlowTrack	PS	Overload	SlowTrack
SORT(Y5)	10%	30%	**100%**	20%	40%	**90%**
FairMOT	20%	40%	**100%**	20%	40%	**100%**
ByteTrack	0%	40%	**80%**	0%	40%	**90%**
BoT-SORT	40%	50%	**100%**	50%	50%	**100%**

Table 6.4.3: System-level evaluation (vehicle crash rate) with Baidu Apollo and LGSVL simulator. 10 runs for each cell.

v5 [155] as detector), FairMOT [327], ByteTrack [326], and BoT-SORT [46]. We use the default parameter for each model.

Evaluation metrics. We design the metrics as follows:

$$\text{R-Track} = \frac{\text{Track-Lat}(x^*) - \text{Track-Lat}(x)}{\text{Track-Lat}(x)}$$

$$\text{R-Lat} = \frac{\text{Total-Lat}(x^*) - \text{Total-Lat}(x)}{\text{Total-Lat}(x)}$$

$$\#\text{Track} = \frac{\text{Tracker\#}(x^*) - \text{Tracker\#}(x)}{\text{Tracker\#}(x)}$$

where R-Track and R-Lat represent the rate of increase for the tracking latency and whole perception latency and #Track represents the rate of increase for the number of tracker. To measure the imperceptibility, we use average $\mathcal{L}_2$ norm [75, 323]. We define system-level effects metric as vehicle crash rate: $\frac{N_{\text{crash}}}{N_{\text{total}}}$, where N_{crash} denotes the number of runs causing vehicle crashes and N_{total} is the number of total runs.

Testing hardware. Given that latency is intrinsically tied to the hardware device, we test

SlowTrack on multiple hardware: TiTAN V, GeForce RTX 2080 Ti (shown to be used in real AD [252]), and GeForce RTX 3090.

Baselines comparison. To our best knowledge, we are the first to propose a latency attack against whole camera-based AD perception pipeline, while the existing attacks focus on object detection alone. Thus, we select two representative latency attacks on object detection as our baselines: PS [249] and Overload [80].

Simulation evaluation. To study the system-level effects, we perform an end-to-end attack evaluation on Baidu Apollo [52], an industry-grade full-stack AD system, with LGSVL simulator [234], a production-grade AD simulator. Our experiments are conducted on the Borregas Ave map and the Lincoln2017MKZ AD vehicle with default configuration. To simulate our attack impact, we model the latency of the camera-based AD perception and inject it into the AD system. Due to the representativeness of SORT (Y5), we use its latency results tested on 2080 Ti GPU (used in genuine AD vehicles [252]), as our latency modeling results. Our evaluation focuses on two representative scenarios as shown in Fig. 6.4.1, where the blue vehicle is the victim AD vehicle and the blue and red lines are the trajectories of the two vehicles. S1 (Fig. 6.4.1 (a)) is a common driving scenario for other vehicles to change the lane line and S2 (Fig. 6.4.1 (b)) is another common driving scenario for the STOP sign-controlled intersection. We perform 10 runs on each scenario and compare SlowTrack with the two baselines.

6.4.2 Experimental Results

Effectiveness. As shown in Table 6.4.4, we compare our SlowTrack with the baselines. SlowTrack can increase the number of tracker up to 1334.9 on FairMOT, which is much better than existing works: at most 939.9 on FairMOT. Especially, for the tracking stage, SlowTrack provides 453.8 times slowing on average compared to the existing works which

Table 6.4.4: Effectiveness results of tracking-stage and whole perception latency with number of trackers and average $\mathcal{L}_2$ norm in different models and hardware. Bold denotes the best results (i.e., highest R-Track, R-Lat, #Track, and lowest $\mathcal{L}_2$) in each row. M.: Model; D.: Dataset; H.: Hardware; SY: SORT (Y5); FM: FairMOT; BT: ByteTrack; BS: BoT-SORT.

M.	D.	H.	PS [249]				Overload [80]				SlowTrack			
			R-Track	R-Lat	#Track	$\mathcal{L}_2$	R-Track	R-Lat	#Track	$\mathcal{L}_2$	R-Track	R-Lat	#Track	$\mathcal{L}_2$
SY	BDD	Titan V	73.1	6.2	69.1	0.042	156.1	12.1	94.6	0.011	**247.0**	**17.5**	**141.8**	**0.010**
		2080 Ti	64.9	11.7			205.1	34.7			**310.3**	**49.4**		
		3090	76.4	6.2			186.3	13.8			**338.3**	**23.0**		
	MOT17	Titan V	37.5	3.8	32.4	0.041	93.0	8.7	47.0	**0.013**	**208.4**	**18.5**	**73.5**	**0.013**
		2080 Ti	33.2	8.6			82.6	19.3			**225.5**	**50.9**		
		3090	39.8	3.7			111.1	9.8			**283.9**	**25.7**		
FM	BDD	Titan V	517.7	10.4	401.1	0.036	1505.9	29.1	939.9	0.029	**2647.6**	**51.3**	**1334.9**	**0.028**
		2080 Ti	390.1	8.7			1258.7	26.7			**2260.7**	**48.1**		
		3090	236.0	3.8			836.4	13.3			**1572.9**	**25.2**		
	MOT17	Titan V	67.4	8.6	48.7	0.036	181.7	21.5	106.2	**0.026**	**341.0**	**41.5**	**159.0**	**0.026**
		2080 Ti	49.3	7.1			149.0	19.9			**279.5**	**38.4**		
		3090	32.4	3.2			108.5	10.1			**220.7**	**21.3**		
BT	BDD	Titan V	40.2	2.5	79.2	0.032	173.0	9.2	224.7	0.027	**290.0**	**14.7**	**307.0**	**0.022**
		2080 Ti	39.5	2.3			184.0	9.0			**266.4**	**12.5**		
		3090	57.8	3.0			217.7	10.1			**341.4**	**15.1**		
	MOT17	Titan V	29.0	1.9	38.0	0.030	95.0	5.4	78.3	0.025	**173.0**	**9.8**	**130.1**	**0.022**
		2080 Ti	26.9	1.8			97.7	5.4			**168.7**	**9.5**		
		3090	45.2	2.4			115.2	5.9			**204.0**	**10.5**		
BS	BDD	Titan V	53.2	19.4	70.0	0.033	79.0	28.9	221.9	0.029	**92.1**	**33.6**	**280.8**	**0.025**
		2080 Ti	57.6	19.6			89.1	30.6			**103.6**	**35.5**		
		3090	61.0	22.1			93.9	34.1			**135.3**	**49.7**		
	MOT17	Titan V	31.7	14.0	40.7	0.034	42.1	18.5	83.9	**0.025**	**52.2**	**23.0**	**127.6**	**0.025**
		2080 Ti	34.2	14.6			45.6	19.5			**59.1**	**25.3**		
		3090	44.0	19.9			47.2	21.1			**69.4**	**30.8**		

only have 256.4 times for Overload and 89.1 times for PS. As for the latency of the whole camera-based AD perception, we find that SlowTrack can provide 28.4 times slowing down on average, which is 2.9 times more than the two existing works. Especially, for the practical SORT (Y5) on 2080 Ti, we observe 50.9 times slowing down, while existing works can only have 19.3 times at most. For the imperceptibility, our average $\mathcal{L}_2$ norm (around 0.021 on average) is very small compared to the baselines and existing adversarial attacks on integrity [323].

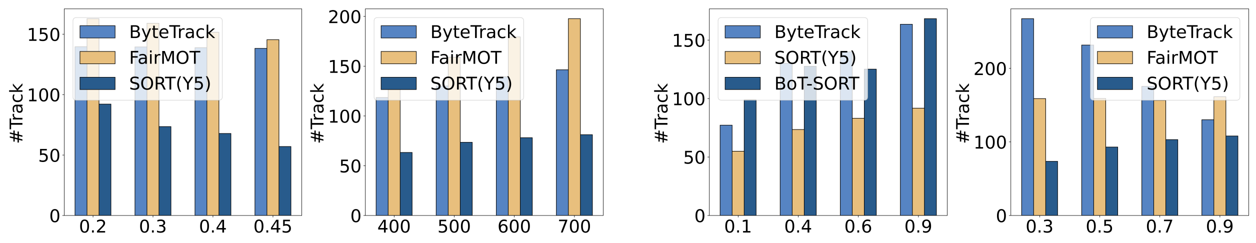
We measure the latency time in Table 6.4.1. For instance, while existing representative attack Overload [80] can trigger 411 ms latency on SORT (Y5) and MOT17 dataset with 2080 Ti, SlowTrack can provide 1,082 ms, which is 1.6 times more than Overload. Thus, SlowTrack can significantly outperform the existing baselines on camera-based AD perception, with similar levels of imperceptibility.

Ablation study. The ablation study evaluating the designs of Eq. (6.2) is presented in Table 6.4.2. In this study, each loss component is sequentially removed, and the attack is tested on the most practical SORT (Y5) tracking method and the MOT17 dataset, utilizing a 2080 Ti. The findings in Table 6.4.2 highlight the indispensability of all three loss designs in achieving high attack effectiveness. Notably, the $\mathcal{L}_{score}$ performs as the most pivotal element in enhancing attack effects. In the absence of $\mathcal{L}_{area}$ and $\mathcal{L}_{match}$, the R-Lat experiences reductions of approximately 33% and 25%, respectively. Thereby, these results indicate the importance of integrating all three loss designs together.

Generality to different thresholds. The effectiveness of SlowTrack is most impacted by different thresholds: confidence score threshold in object detection, maximum number of detection boxes, IOU threshold for NMS, and IOU threshold for data association. To evaluate their impact, we vary the thresholds with different values and measure the #Track on the MOT17 dataset with 2080 Ti. The results are shown in Fig. 6.4.2. SlowTrack can generally generate stable results across different threshold parameters, which indicates that setting better thresholds cannot fully defend against SlowTrack. For instance, setting the confidence score threshold to 0.2, the #Track of SORT (Y5) is around 92.1 while setting the confidence score threshold to 0.4, the #Track is around 67.8. Additionally, finetuning the thresholds usually will significantly reduce the benign performance. Thus, the results show the generality of SlowTrack to different thresholds.

Table 6.4.5: Variances of R-Track and R-Lat in repeated experiments (5 times) with different hardware on MOT17 dataset.

Model	Titan V		2080 Ti		3090	
	R-Track	R-Lat	R-Track	R-Lat	R-Track	R-Lat
SORT(Y5)	0.032	0.001	0.106	0.007	0.459	0.007
FairMOT	0.402	0.004	0.963	0.010	0.835	0.005
ByteTrack	0.156	0.002	0.075	0.003	0.050	0.001
BoT-SORT	0.006	0.001	0.158	0.032	0.192	0.054



(a) Confidence Threshold (b) Max. Number of Detections (c) IOU Threshold for NMS (d) IOU Threshold for Match

Figure 6.4.2: Attack effectiveness under different thresholds.

End-to-end simulation evaluation results. As shown in Table 6.4.3, SlowTrack can achieve 95% vehicle crash rate on average while for other attacks, they can only have around 30%. Note that the vehicle crash rate in benign cases is always 0%. The results demonstrate that the existing latency attacks on object detection alone cannot trigger sufficient latency to highly effectively cause vehicle crashes, which motivates our new attack design on entire perception. SlowTrack significantly improves the system-level effects.

Running time variance. Recognizing the potential variability of system latency due to many factors, we repeat our experiments 5 times [236] and measure the variance of R-Track and R-Lat. As shown in Table 6.4.5, the variance is negligible compared to the original values [182].

6.5 Discussion

6.5.1 Physical-World Attack Realizability

Our study investigates the runtime robustness of camera-based AD perception by adversarial attacks. Such attack is physically realizable as demonstrated in AttrackZone attack [211], which leverages the projector to project the noise-level adversarial attack in physical world at night. To improve our attack realizability, we take a small step forward to generate a patch-based adversarial attack which is generally demonstrated as a physical-world realizable attack [283, 282].

Adversarial patch generation. To formulate the patch into a patch δ, we design the following method:

$$\min_{\delta} E_{x \sim \mathcal{X}} \mathcal{L}_{adv}(x + \delta)$$

where the $\mathcal{L}_{adv}$ is introduced in Eq. (6.2), $\mathcal{X}$ denotes EoT [55] distribution for robustness such as different pre-processing method.

Evaluation setup and preliminary results. We use a similar setup in the experiment section, where we select the most practical model: SORT (Y5) and MOT17 dataset with 2080 Ti. The results values for R-Lat, R-Track, and #Track are 82.3, 336.1, and 80.9, respectively, aligning closely with the findings in Table 6.4.4. These results suggest that the patch attack can potentially induce substantial system-level effects with practicality. Note that the patch results are slightly better than the noise attack. Since the perturbation strength of the patch is much larger, it enables the attacker to generate bounding boxes with high confidence and to precisely dictate their locations. Thus, the tracking can be easily controlled by the attacker and the attack effects can be improved. However, the patch attack requires more complex designs such as patch size and patch location, which are correlated with the practicality and effectiveness. In this paper, we only provide a preliminary evaluation demonstrating the potential to transfer the noise attack to the patch attack. We leave the study on patch-based attack as our future work.

6.5.2 Limitations and Future Work

First, although we have some physical-world realizability improvements and some existing works demonstrate that such attack is realizable, it is still unclear whether SlowTrack can indeed work well in physical world, which can be a potential future direction to explore. Second, exploring SlowTrack under a black-box threat model, a more practical one, is another potential future work. Third, while several availability-based latency attacks in AD have been identified, the exploration of defense directions in this context remains limited. Consequently, we consider the investigation of defenses as a part of our future work.

6.6 Conclusion

This paper presents a first study on availability-based latency adversarial attacks considering the entire camera-based AD perception pipeline, i.e., both object detection and object tracking. We design a novel attack framework, SlowTrack, with a two-stage attack strategy and three novel loss functions. Our results show that SlowTrack can outperform all the existing latency attacks on camera-based object detection and significantly improve system-level effects, i.e., 95% vehicle crash rate. Due to the critical role of perception, we hope that our findings and insights can inspire more future research into this largely overlooked research perspective.

Chapter 7

Revisiting Physical-World Adversarial Attack on Traffic Sign Recognition: A Commercial Systems Perspective

7.1 Introduction

In the rapidly evolving landscape of Autonomous Driving (AD) technology, AD vehicles, such as the millions of Tesla cars [158] running on the public road, are becoming an integral part of our daily lives. Compliance with traffic signs is essential for all vehicles, no matter if they are high-autonomy AD vehicles (e.g., those for robo-taxi [286]), semi-autonomous AD vehicles (e.g., those with Tesla Autopilot [264]), or conventional human-driven vehicles. Failure to obey these rules can lead to accidents, posing a threat to human life.

Due to the importance of traffic sign detection, a natural question is whether AD vehicles are truly as secure as we hope. To answer this critical question, recent research in security analysis of Traffic Sign Recognition (TSR) systems has highlighted vulnerabilities to a wide range of physical adversarial attacks that can significantly impair the traffic sign detection accuracy [187, 330, 283, 148, 213, 110, 83, 243, 186, 332]. Among them, the most representative and also the most widely-exploited attack vectors are *physical patches or posters* [187, 283, 330, 110, 148, 83, 243], which are low-cost, highly deployable, and demonstrated capable of causing various highly severe attack effects. For instance, they can make critical legitimate traffic signs undetectable, or *hiding attacks*, and trigger false detection at any attacker-chosen positions, or *appearing attacks*. Such attacks can cause various potential safety hazards such as traffic sign violations, unexpected emergency braking, speeding, etc. Due to such a high potential for practical impacts, these physical-world adversarial attacks on TSR have drawn wide attention across not only the technology community [135, 108, 256, 216, 45] but also the general public [100, 162, 121, 210, 114].

Despite such high practical impact potentials, so far existing works generally only considered evaluating the attack effects on academic TSR models, leaving the impacts of these attacks on real-world commercial TSR systems largely unclear. A few recent works tried to understand such commercial TSR system-level impacts, but their evaluation is all limited to one particular vehicle model [148, 243], sometimes even an unknown one [148], making both the generalizability and representativeness of these evaluation results questionable. This thus raises a critical research question: *Can any of the existing physical-world TSR adversarial attacks achieve a general impact on commercial TSR systems today?*

To answer this critical research question, in this paper, we perform the first large-scale measurement of physical-world adversarial attacks against commercial TSR systems. In this measurement, we focus on hiding attacks as they can most directly impair the function of a commercial TSR product (i.e., by nullifying the TSR function), and test their black-box transfer attack effectiveness against commercial TSR systems, which is the most practical threat model against commercial systems and also the exact setup used by prior works to claim their attack effects on commercial systems [148, 243]. In total, we were able to include four different commercial vehicle models in this testing, all of which are from the top 15 best-selling vehicle brands in the US to ensure high representativeness (§7.2.1). As estimated later in §7.3.2, this setup can be generalizable to at least 33.2% of the commercial TSR systems sold in the U.S. in 2023, which is significantly improved over prior works (at most 3.8% or unknown). We tested all prior works that demonstrated black-box attack transferability in the physical world, which are presumably those having the highest potential to successfully attack commercial systems.

Our testing results reveal that it is actually possible for existing attack works from academia to have a highly-reliable 100% attack success rate against certain commercial TSR system functionality, which is much higher than expected if compared to the transfer attack success rates reported by the corresponding original papers (e.g., for one such case the reported was <20%). Meanwhile, we do not see generalizability of such attack capabilities over different commercial vehicle models and sign types. Over the entire 30 attack test combinations of different sign types, attack methods, surrogate models, and vehicle models, the vast majority (28/30) do not show any successful attack effects, leading to a 6.67% overall transfer attack success rate against commercial TSR systems, which is almost a magnitude lower than those reported in the original papers (51.6% on average).

The much lower-than-expected black-box transfer attack success rate on commercial systems suggests the potential existence of deeper challenges for such attacks to take effect at the TSR system level. Through our investigations, we find that one major factor might be an unexpected *spatial memorization design* that commonly exists in commercial TSR systems today. Specifically, this design exhibits an effect that once a sign is detected, both the detected sign type and the detected location are persistently memorized until the sign's reaction task is finished. Different from methods like object tracking that can only temporally memorize a detection result for a very short time (typically < 3 sec), the spatial memorization we observed will only forget/clear a detection result after the sign's reaction need in the spatial domain is met (e.g., when the vehicle passes the detected sign), regardless of time.

Such a spatial memorization design can significantly impact the success of existing adversarial attacks at the TSR system level. For example, for hiding attacks, to achieve a system-level success in which the TSR system is unable to show the sign display at the sign's reaction task period, the attack has to be continuously successful at *all* possible detection moments that can trigger such memorization before the vehicle passes the sign. For appearing attacks, such an impact on the TSR system-level attack success is the *opposite*: as long as the attack can succeed in *any* of such detection moments, the TSR system-level attack effect can be achieved.

Since such a spatial memorization design commonly existing in commercial TSR systems today may create a significant discrepancy between the TSR model-level attack effect and that at the TSR system level, we further mathematically model its impact on the TSR system-level attack success for both hiding and appearing attacks, which results in new attack success metric designs that can systematically consider the spatial memorization effect. Through both theoretical proof and numerical analysis using these new metric designs, we find that due to spatial memorization, hiding attacks are theoretically harder (if not equally hard) than appearing attacks in achieving TSR system-level attack success. Such an attack hardness gap can be huge ($\geqslant$93.8% absolute differences in attack success rate values). Meanwhile, due to the lack of consideration of spatial memorization, existing TSR model-level attack success metrics can be highly misleading in judging the TSR system-level attack success, with a potential of having $\sim$50% absolute attack success rate value differences.

Due to such potential huge differences in judging the TSR system-level attack success, we then use the new attack success metrics to revisit the evaluations, designs, and capabilities of existing attacks in this problem space. These efforts lead to various new findings compared to existing knowledge in this problem space, some of which directly challenge the observations or claims in prior works due to the introduction of the new attack success metrics. For example, we find that while some prior works in this problem space can be claimed as effective using prior success metrics (e.g., with $\sim$50% to 90% success rates), when spatial memorization is considered, their success rates can drop significantly to $\leqslant$6.6% even in white-box attack settings, making it no longer approximate to claim them as effective at the TSR system level. As another example, also due to spatial memorization, we find that the benefits of certain prior attack designs can be seemingly high (e.g., >20% attack success rate increase) using prior metrics, but are actually nearly negligible (e.g., only 1% increase) at the TSR system level after spatial memorization is considered. The code and data will be made available at our website: **https://sites.google.com/view/av-ioat-sec/commercial-tsr-test**.

To sum up, this paper makes the following contributions:

- *First large-scale commercial system measurements:* We conduct the first large-scale measurement of physical-world adversarial attacks against commercial TSR systems. Our testing results reveal that although it is possible for existing attack works from academia to have highly reliable (100%) attack success against certain commercial TSR system functionality, such black-box commercial system attack capabilities are not generalizable, leading to a much lower-than-expected overall black-box transfer attack success rates.
- *Discovery and analysis of spatial memorization:* We discover a spatial memorization design that commonly exists in today's commercial TSR systems, which can keep memorizing a sign detection result until the sign's reaction need in the spatial domain is met (e.g., when the vehicle passes the detected sign's position). This discovery is crucial as it is shown to be capable of creating a significant discrepancy between the TSR model-level attack effect and that at the TSR system level.
- *New attack success metric designs*: We mathematically model the impact of this design on the TSR system-level attack success for both hiding and appearing attacks, resulting in new attack success metric designs that can systematically consider the spatial mem-

Table 7.1.1: Top 15 leading car brands in the United States based on vehicle sales in 2023 [260]. The ones with direct evidence of equipping front windshield cameras for TSR from their official websites or vehicle manuals are marked with check-marks. The vehicle models from 4 out of the 5 highlighted brands below (with bold and underline) are tested in our study (we choose to not directly reveal which four for anonymity purpose).

Car brand	Sales number	TSR
Ford	1,904,038	✓
Toyota	1,888,941	✓
Chevrolet	1,702,700	
Honda	1,156,591	✓
Nissan	834,091	✓
Hyundai	796,506	✓
Kia	782,468	✓
Jeep	641,166	✓
Subaru	632,083	
GMC	563,692	✓
Ram	539,477	✓
Tesla	498,000	✓
Mazda	365,044	✓
BMW	361,654	✓
Volkswagen	329,025	✓

orization effect. We then use them to revisit the evaluations, designs, and capabilities of existing attacks in this problem space.

- *New observations:* Through the commercial TSR system measurements, new metric designs and analysis, and the revisiting of existing attacks, we uncover a total of 7 novel observations compared to existing knowledge in this problem space, some of which directly challenge the observations or claims in prior works due to the introduction of the new attack success metrics.

Table 7.1.2: Existing works with successfully demonstrated traffic sign hiding or appearing attack effects in the physical world using sticker patches or whole-sign posters. HA: Hiding Attack. AA: Appearing Attack. As shown, the commercial system testing aspect is currently under-studied. Highlighted in gray rows are the works that have demonstrated black-box attack transferability in the physical world and thus have the highest potential to successfully attack commercial systems, which are thus the targets of our study later in §7.3 and §7.4.

Existing works	Year	HA	AA	Demonstrated transferability?	Commercial system testing?
AEFD [187]	2017	✓		No	None
RP_2 [110]	2018	✓	✓	Yes	None
SIB [330]	2019	✓	✓	Yes	None
FTE [148]	2022	✓	✓	Yes	1 unknown vehicle model
SysAdv [283]	2023	✓		No	None
DM [243]	2024		✓	Yes	1 Tesla model

7.2 Background and Related Work

In the section, we introduce the background for Traffic Sign Recognition (TSR) systems, physical-world adversarial attacks against TSR system, security of autonomous driving (AD) systems, and the threat model for this study.

7.2.1 Traffic Sign Recognition (TSR) System

As a key component of Advanced Driver Assistance Systems (ADAS), Traffic Sign Recognition (TSR) system is defined as a system that employs camera sensors to detect road signs, including but not limited to speed limit and STOP signs [47, 290, 137]. Today, this technology is highly prevalent across various vehicle brands to enhance both safety and driving comfort. Table 7.1.1 shows the top 15 leading car brands in the United States based on vehicle sales in 2023 [260]. We surveyed their official websites and vehicle manuals, and were able to find direct evidence of equipping front windshield cameras for TSR for at least 13 out of these 15 brands.

Recent advancements in Deep Neural Networks (DNNs) have propelled significant progress in various domains, including TSR systems, which now increasingly rely on DNN-based methodologies for real-time object detection [237, 148, 330, 50]. These systems process camera sensor data through DNN-based object detectors to identify road signs efficiently. Current state-of-the-art object detection models are categorized into two primary types: one-stage and two-stage detectors [338]. One-stage detectors, such as YOLO [231], are celebrated for their rapid detection capabilities. Conversely, two-stage detectors, exemplified by Faster R-CNN [232], are noted for their exceptional accuracy. Prior research [330, 283, 237] focusing on the security aspects of TSR systems has examined models from both categories for evaluation comprehensiveness. In line with these research, our analysis also encompasses object detectors from both categories, aiming to provide a comprehensive assessment of TSR systems security.

7.2.2 Physical-World Adversarial Attacks against TSR

DNN models today are shown to be generally vulnerable to adversarial examples (or *adversarial attacks*) [262, 123, 76, 220, 200, 221, 323, 71]. Such vulnerabilities are especially extensively studied and demonstrated in the vision domain [262, 123, 76, 220, 200, 221, 323, 150, 193, 242, 151, 283, 239]. Due to the increasing real-world penetration of TSR systems and their fundamental reliance on the camera inputs, TSR models soon became a natural target of adversarial attack research, including many that were able to achieve particularly high realism with successfully demonstrated attack effect in the physical world [110, 187, 330, 83, 213, 335, 283, 243, 237, 186, 104, 332, 195].

Among them, the most representative and also the most widely-exploited attack vectors are *physical patches/posters*, e.g., by physically printing attack patterns on *sticker patches* and attaching them to the legitimate traffic sign surface [110, 283, 330, 132], or on *whole-*

sign posters that replace or spoof the entire traffic sign surface [187, 148, 330, 110, 243]. These attacks are low-cost, highly deployable, and demonstrated capable of causing various severe attack effects, most notably (1) making critical legitimate traffic signs undetectable, most representatively *hiding attack*, or HA [110, 330, 283, 148]; and (2) triggering false detection at any attacker-chosen positions, most representatively *appearing attack*, or AA [110, 330, 148, 243]. For drivers who are relying on such a driver assistance function, or higher-autonomy AD systems that try to automatically react to real-time TSR results (e.g., in Tesla [26]), such attacks, especially the hiding ones, can directly impair the TSR function and cause various potential safety hazards such as traffic sign violations, unexpected emergency braking, speeding, etc. Due to such a high potential for practical impacts, these physical-world adversarial attacks on TSR have drawn wide attention in not only the technology community [135, 108, 256, 216, 45] but also the general public [100, 162, 121, 210, 114].

Despite such a high practical impact potential, so far these works generally only considered evaluating the attack effects on academic TSR models, leaving the impacts of these attacks on real-world commercial TSR systems largely unclear. Table 7.1.2 summarized all the prior works so far that were able to successfully demonstrate using physical patches/posters to trigger HA or AA attack effects in the physical world. As shown, although many of them have demonstrated the attack transferability across academic models, which could be viewed as an indicator of high potential black-box attack capability against commercial systems, very few have actually evaluated the attacks against real commercial vehicle systems. For the only two works that have done so, the evaluation is limited to one particular vehicle model (for one of them it is even an unknown vehicle model) [148, 243], making both the generalizability and representativeness of these evaluation results questionable.

The observations above thus raise a critical research question: *Can any of these existing physical-world TSR adversarial attacks achieve a general impact on commercial TSR systems today?* In this work, we thus aim to systematically answer this critical research question by performing the first large-scale testing of representative existing attacks against commercial TSR systems of top popularity on the consumer market today. Leveraging the observations and insights from the testing, we also further systematically revisit existing evaluation metrics and attack designs in this problem space.

7.2.3 Security of Autonomous Driving (AD) systems

Due to the fundamental reliance of AD systems on environmental sensing, prior works have extensively investigated sensor attacks within the AD context. These include spoofing or jamming attacks targeting cameras [310, 213, 244, 161], LiDAR [72, 254, 68, 152], RADAR [310], etc., highlighting vulnerabilities at the sensor level. In contrast, our study focuses on the vulnerabilities at the autonomous AI algorithm level, specifically targeting the TSR functions in AD systems, which is highly crucial for safe and correct driving automation. While the existing body of literature covers security aspects of various components such as camera object detection [110, 83, 330, 148, 71, 283], object tracking [150, 193], lane detection [239], and end-to-end AD systems [221, 268], there is a notable gap in large-scale studies on their effectiveness in real-world commercial AD systems. In this work, we thus aim to bridge this critical research gap by conducting the first large-scale measurements of physical-world adversarial attacks on commercial TSR systems, which not only expands the understanding of such security vulnerabilities in real-world AD systems but also provides various new insights into the design and evaluation of existing works in this problem space.

7.2.4 Threat Model

To understand the impacts of these existing physical-world TSR attack works from the commercial systems perspective, we consider the most realistic *transferability-based black-box* threat model, i.e., generating the adversarial attack pattern using publicly-accessible surrogate TSR models and then applying it to the attack-targeted TSR systems with unknown model parameters and architectures. We choose this because (1) as with most commercial products, commercial TSR systems, especially the most popular ones today in Table 7.1.1, are by default closed-source to the public and so far there is no effective approach to generally reverse-engineer them; and (2) this is also the setup used by existing physical-world TSR attack works to claim their attack effects on commercial systems [148, 243]. For attack goals, we consider both traffic sign hiding and appearing attacks as highlighted in §7.2.2, with a more specific focus on the hiding attack side as it can most severely impair a commercial TSR product by completely nullifying a TSR system's functionality. For attack vectors, we follow the most representative and practical physical patch/poster attack vectors (§7.2.2). Specifically, we use sticker patches for hiding attacks (HA) and whole-sign posters for appearing attacks (AA), which are the most practically-deployable attack vectors on both sides [110, 330, 148, 283, 243].

7.3 Large-Scale Commercial TSR Systems Testing and Observations

In this section, we report our efforts on the first large-scale testing of existing physical-world TSR model attacks on commercial TSR systems. In this testing, we specifically focus on the hiding attacks (HA) as they can make critical traffic signs undetectable and thus most directly impair the function of a commercial TSR product (§7.2.4). We first detail the experimental setup and then report our key observations.

Table 7.3.1: Attack success rates for our reproduced RP_2, SIB, and FTE attacks compared to those in the original papers following the same surrogate model and attack distance setups.

Attack success rate	RP_2 [110]	SIB [330]	FTE [148]
Reported by the papers	63.5%	60.5%	98.8%
Our reproduced attacks	60.5%	84.5%	98.7%

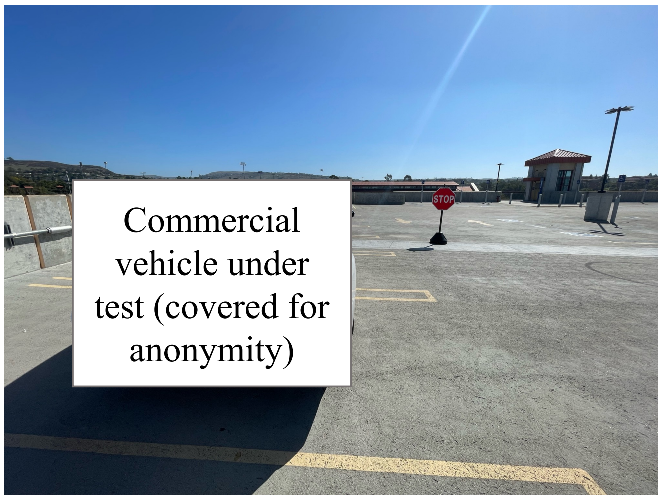
7.3.1 Attack Setup

Traffic Sign Selection. Our study considers two types of traffic signs: the STOP sign and the 25 mph speed limit sign. We choose these two types of signs since (1) STOP and speed limit signs are the most popular targets in prior works for demonstrating the adversarial attack effects in the physical world [188, 148, 110, 330, 309, 187, 83, 243, 283, 237]; and (2) both are highly safety-critical as missing STOP signs can lead to intersection collisions and 25 mph speed limit sign is usually for residential or school districts where children can be outside or crossing the street [215]. Meanwhile, these two sign types are also the only two types of signs with demonstrated physical-world black-box attack transferability in prior works [110, 330, 148].

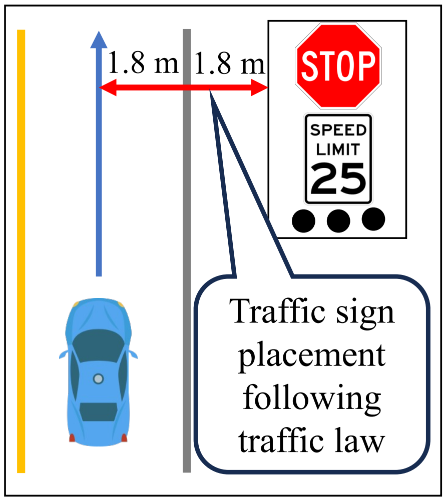
Selected Attacks. As shown in Table 7.1.2, there are three prior works so far that were able to demonstrate black-box attack transferability for the hiding attack effect in the physical world: RP_2[110], SIB [330], and FTE [148]. Specifically, FTE was able to demonstrate this against a commercial vehicle model, despite an unknown one [148]. These three works have the highest potential to successfully attack commercial systems; thus, we aim to include all of them in our testing. Unfortunately, at this point, none of these three works have open-sourced their methods. Thus, we have to reproduce them. We tried our best to reproduce them, which included both following their papers closely and consulting with the original paper authors for all three, and the reproduced attack success rates are shown in Table 7.3.1. Note that there are some performance discrepancies between our reproduced version and the original one. These discrepancies can be due to various factors, most likely the differences in the experimental setups such as the hyperparameters used in attack generation, the real-world data collected for physical-world robustness, and realizability optimization, etc. Despite our best efforts to replicate their original experimental setup, including consulting the original paper authors, it is fundamentally impossible for us to exactly replicate their setup due to the lack of open-sourcing in these prior works.

Surrogate Model and Dataset Selection. As detailed in §7.2.4, in this testing, we adopt the most realistic transferability-based black-box threat model. To achieve this, we carefully select surrogate models and datasets, based on the selection by previous research in this field [237, 283, 148]. Specifically, we utilize the Microsoft COCO dataset [181] to study adversarial attacks on STOP signs and the ARTS dataset [50] for speed limit signs. Our surrogate model selection covers both one-stage and two-stage TSR model designs (S7.2.1) to increase the chance of successful transfer attacks. Specifically, we choose YOLO v5 (denoted as *Y5*) for the one-stage model and Faster RCNN (denoted as *FR*) as the two-stage one, both of which are from the most widely-used model families in prior works [148, 330, 110, 83, 335]. In particular, Y5 is also the one that has been used to demonstrate a successful transfer attack to a commercial vehicle model [148]. For Faster RCNN, we adopt the latest official PyTorch implementation that uses the ResNet-50-FPN-V2 backbone [177]. For the Microsoft COCO dataset, the models are obtained directly from the Y5 official website [155] and the PyTorch models [226]. For the ARTS dataset, we conduct our own model training. The benign performances of these models have an mAP (mean Average Precision) of 0.831 for Y5 and an mAP of 0.871 for FR, which are consistent with those reported in prior research [237].

Test Environment Setups. Our experiments are performed outdoors during sunny afternoons between 1 pm and 4 pm, to simulate the most common real-world attack scenarios. To maintain consistent testing conditions, we measure the ambient light levels using a light meter, ensuring that all tests are conducted within a light range of 25,000 to 30,000 lux. Visual representations of the real-world environment and its bird's-eye view illustration are



(a) Real-world view



(b) Bird's-eye view

Figure 7.3.1: Experiment setup for commercial TSR system testing. We cover the vehicle in the photo for anonymity purpose.

Table 7.3.2: Four of the five commercial vehicle models below are tested in our study (denoted as C1 to C4 in Table 7.3.3). Each model has functions to detect a STOP sign, speed limit signs, or both. We choose not to directly reveal the exact models for C1 to C4 for anonymity purpose. Note that this is the least anonymization to protect the affected companies (i.e., only 1 confusing vehicle model), and this is also part of the agreement with the companies during our responsible vulnerability disclosure.

Tesla Model 3 2023	Toyota Camry 2023	Nissan Sentra 2023	Mazda CX-30 2023	Hyundai Tucson 2024

provided in Fig. 7.3.1. We select a spacious rooftop parking structure as the location for these experiments ensuring no presence of other vehicles or humans to maintain safety. The placement of traffic signs is carefully designed to follow traffic laws in the U.S. as outlined in previous studies [283]. For the testing distance, we carefully set the start point to be farther than the detection distance in the benign case for each vehicle model (~50 meters). For the testing speed, we test at the maximum-allowed speed limit of our rooftop parking structure (5 mph) to ensure safety.

Table 7.3.3: TSR functions of the four vehicle models tested in our measurement study. These four models are among the five in Table 7.3.2. We choose to not directly reveal the exact vehicle brands and models for anonymity purpose.

	TSR functionality	
Vehicle model	STOP sign	Speed limit sign
Car 1 (denote as C1)	✓	✗
Car 2 (denote as C2)	✓	✓
Car 3 (denote as C3)	✗	✓
Car 4 (denote as C4)	✗	✓

7.3.2 Commercial Systems Under Test and Metric

Commercial Systems Under Test. We were able to include 4 different vehicle models in this testing through borrowing or renting. These four vehicle models are among the five in Table 7.3.2. As shown, all of them are among the top 15 popular vehicle brands in the US based on vehicle sales (Table 7.1.1) and all of them are from the most recent model years, either 2023 or 2024. Note that we choose to not directly reveal the exact vehicle brands and models for anonymity purpose; this is already the least anonymization to protect the affected companies (i.e., only 1 confusing vehicle model), and this is also part of the agreement with the companies during our responsible vulnerability disclosure. In the paper, we denote the four vehicle models we tested as C1 to C4. Table 7.3.3 shows the TSR functionality support we found for C1 to C4 using benign STOP and speed limit signs. As shown, two of them can support STOP sign detection, while three of them can support speed limit sign detection. In particular, C2 can support both.

Generalizability of the systems under test. While the rankings of the vehicle sales in Table 7.1.1 have quantifiably shown the representativeness of the vehicle models under test, it is better if we can further quantify the generalizability of this tested system setup. To achieve this, we use the market share of the tested vehicle models as an estimate. Specifically, in 2023 the U.S. automotive industry sold around 15.6 million vehicles [204]. Based on the data in Table 7.1.1, at most 13.3 million of these 15.6 million vehicles are TSR-equipped. The total sales of the 5 possible vehicle brands in our test account for 33.2% of such upper-bound number of TSR-equipped vehicles sold in 2023. Thus, our current testing results can be estimated as generalizable to at least 33.2% of the commercial TSR systems sold in the U.S. in 2023. Using this estimation, our commercial TSR systems testing results can also be shown to be much more generalizable than prior works in Table 7.1.2 (i.e., at most 3.8% for [243] and unknown for [148]).

TSR System-Level Attack Success Metric. In prior works, the TSR adversarial attack success rates are generally calculated by TSR model-level metrics, i.e., first determining the attack success at TSR model output level per frame and then aggregating the per-frame results over one or multiple distance ranges [330, 110, 252, 186, 335, 283, 237] or a certain number of consecutive frames [330, 252]. However, we find that the TSR systems in commercial vehicle models today do not generally show real-time traffic sign detection results to end users; instead, the duration and timing of the detection result display are more generally based on the *system-level needs* for different sign types. For example, for speed limit signs, we find that all the vehicle models supporting them (C2 to C4) do not immediately show the speed limit sign detection results when the sign is actually detected; instead, the detection results will only be on display *after* the vehicle passes the sign (more precisely, always when the vehicle body is halfway past the sign in our experiments). This TSR system-level design aligns with system-level needs for such traffic sign detection functionality, as a newly-detected speed limit should be applied *after* the vehicle drives past the physical location of the sign [257]. This design can also be viewed as reasonable if we consider the design of the speed limit sign display is to indicate the speed limit of the current road segment, so it is indeed correct to only show the new speed limit after it enters the corresponding road segment. If it shows the new speed before that, the car can be speeding before it enters a road segment with a higher speed limit.

Table 7.3.4: Commercial TSR systems testing results against vehicle model C1 to C4 with comparisons to the black-box transfer attack success rates reported by the original papers. The testings for each benign or attack setup are repeated 3 times. Ave.: Ave. over all attacks.

	Original paper transferability	Surrogate model	C1	C2		C3	C4	Ave.
			STOP	STOP	Speed limit	Speed limit	Speed limit	
Benign traffic sign			**100% (3/3)**	**100% (3/3)**	**100% (3/3)**	**100% (3/3)**	**100% (3/3)**	**100%**
RP_2	18.9%	Y5	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0%
		FR	0% (0/3)	**100% (3/3)**	0% (0/3)	0% (0/3)	0% (0/3)	**20%**
SIB	46.1%	Y5	0% (0/3)	**100% (3/3)**	0% (0/3)	0% (0/3)	0% (0/3)	**20%**
		FR	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0%
FTE	89.8%	Y5	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0%
		FR	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0%
Ave	51.6%		0%	**33.3%**	0%	0%	0%	**6.67%**

For the STOP sign, we have a similar observation: although it is different from speed limit signs in that it should be displayed before the vehicle passes the sign, we find that in C2 once the sign is detected, the TSR system will keep having the sign display instead of showing the real-time detection results until it passes the sign (this is also the spatial memorization effect that we will investigate more later in §7.3.3). This again aligns well with the system-level needs, as a detected STOP sign should take effect until it is passed.

Due to these observations, we need to use an attack success metric defined at the TSR system level to most generally and practically-meaningfully capture the impacts of adversarial attacks on commercial TSR systems. To this end, we thus define the TSR system-level attack success per each *traffic sign reaction task* on the TSR system user side, i.e., when the TSR system user needs to react to a sign, if the TSR system is able to correctly display the sign, the attack fails; otherwise, the attack succeeds. For example, for speed limit sign, the attack success is judged by whether the system can have the sign displayed at the time when the vehicle passes the sign, while for STOP sign it is judged by whether the system can have the sign displayed before the vehicle passes the sign.

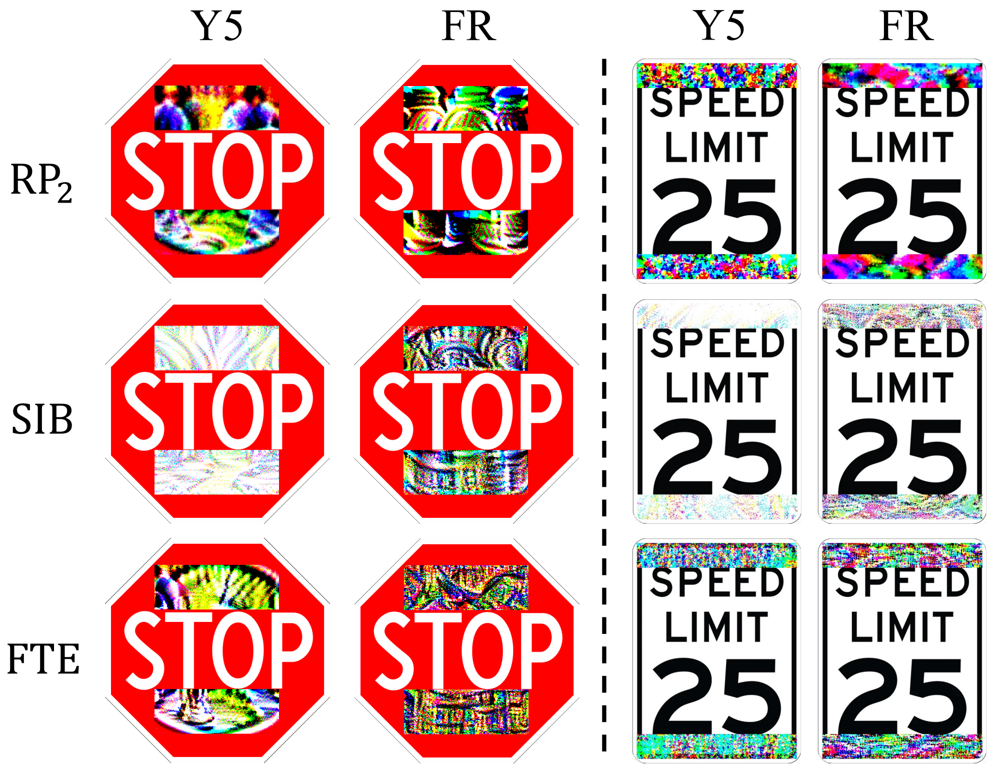


Figure 7.3.2: Visualisation of the hiding attacks (HA) generated for STOP and speed limit signs, which are used in our commercial TSR systems testing. They are generated by the three most promising prior works (RP$_2$ [110], SIB [330], FTE [148]) using surrogate models of both representative one-stage and two-stage TSR model designs.

7.3.3 Testing Results and Observations

Overall Testing Results. Table 7.3.4 summarizes the overall testing results and the reproduced attack visualization is in Fig. 7.3.2. As shown, there are 30 attack test combinations in total, each for one combination of the 2 sign types, 3 attack methods, 2 surrogate models, and 4 vehicle models. For each benign and attack setup, we repeat the testing three times. As shown, we are indeed able to find attack setups in which existing physical-world adversarial attack works from academia can reliably work on a certain commercial TSR system, more specifically the RP$_2$ attack using FR as the surrogate model, and the SIB attack using Y5 as the surrogate model. In these setups, the attack can *always* succeed over the three runs, leading to a 100% success rate. Such a high black-box transfer attack effectiveness is even higher than expected as the transfer attack success rates are actually less than 50% for both and even less than 20% for RP_2. This suggests that for certain commercial TSR systems, although from top brands in the US (Table 7.1.1), their TSR functionality can actually be much more vulnerable than academic TSR models under black-box transfer attacks. Interestingly, both successful attack setups are against C2, the only vehicle model among the four that can support both STOP and speed limit signs. We have already performed responsible vulnerability disclosure to the C2 manufacturer to report these.

While there do exist successful attack cases, we do not see generalizability of such attack effects over the entire testing results. Over the entire 30 attack test combinations, the vast majority (28/30) do not show any successful attack effects, leading to a 6.67% overall transfer attack success rate against commercial systems. As shown in the table, this is almost *a magnitude lower* than those reported in the original papers [110, 330, 148] (51.6% on average). Even for these two successful attack setups, the attack effect is limited to the STOP sign detection and cannot even generalize to the speed limit sign detection for the same vehicle model (C2). This could be due to the need to customize the detection accuracy for certain sign types because of real-world deployment or customer needs. Interestingly, although FTE was able to demonstrate a successful transfer attack against a commercial vehicle model in its paper, we were not able to find any successful attack results against any of the four commercial vehicle models in our tests. This further reveals the lack of generalizability of the reported commercial TSR system attack success in the original FTE paper, which cannot be revealed without the large-scale commercial system testing efforts in this paper.

Note that we have performed statistical testing for the results above to understand their statistical significance. While the overall transfer attack success rate against commercial systems over all 30 attack test combinations (6.67%) is statistically significant, the statistical significance for the result of each test combination cannot be calculated since the variance

Table 7.3.5: TSR detection result memorization rates when we hide the sign for a long time (20-60 seconds) after a short sign display time (1-3 seconds). The experiment setup is described in Fig. 7.3.3. As shown, three out of the four vehicle models exhibit a spatial memorization design, i.e., keeping memorizing a sign detection result until the sign's reaction need in the spatial domain is met (e.g., when the vehicle passes the detected sign), regardless of time. Notice the observed much longer memorization time (60 sec) than that from typical temporal memorization designs such as object tracking (typically <3 sec [334, 252, 283, 150]).

Vehicle model	Sign display time	Sign disappearing time after the short sign display					
		STOP sign			Speed limit sign		
		20 sec	40 sec	60 sec	20 sec	40 sec	60 sec
C1	1 sec	0% (0/3)	0% (0/3)	0% (0/3)	-	-	-
	3 sec	0% (0/3)	0% (0/3)	0% (0/3)	-	-	-
C2	1 sec	100% (3/3)	100% (3/3)	100% (3/3)	100% (3/3)	100% (3/3)	100% (3/3)
	3 sec	100% (3/3)	100% (3/3)	100% (3/3)	100% (3/3)	100% (3/3)	100% (3/3)
C3, C4	1 sec	-	-	-	100% (3/3)	100% (3/3)	100% (3/3)
	3 sec	-	-	-	100% (3/3)	100% (3/3)	100% (3/3)

for each is 0 (all failure or all success as shown in Table 7.3.4). While it may be possible that some variance can appear if we significantly increase the number of attempts per test combination (e.g., to over 30) to make statistical significance calculable, we cannot afford this due to inherent limitation for any outdoor vehicle testing setups. Note that our current setup is already scientifically more rigorous than all prior works in this as they only tried once for each attack test. More detailed discussions are in §7.5.2.

Observation 1: It is in fact possible for existing physical-world adversarial attack works from academia to have highly reliable (100%) attack success against certain commercial TSR system function in practice. However, such black-box commercial system attack capability is currently not generalizable over different representative commercial system models and sign types. Overall, the black-box transfer attack success rate on commercial systems (at least on our setup that can account for at least 33.2% of commercial TSR systems sold in the U.S. in 2023, as estimated in §7.3.2) is much lower than that on academic models in prior works.

Spatial Memorization of TSR Results. The much lower-than-expected black-box transfer attack success rate on representative commercial systems suggests the potential existence of deeper challenges for such attacks to take effect at the TSR system level. Through our investigations, one major factor might be an unexpected *spatial memorization design* that commonly exists in commercial TSR systems today. Specifically, this design exhibits an effect that once a traffic sign is detected, both the detected sign type and the detected location are persistently memorized until the sign's reaction task is finished. Different from simple object tracking that can only temporarily memorize a detection result for a very short time (typically at most 3 seconds [334, 252, 283, 150]), the spatial memorization we observed will only forget/clear a detection result after the sign's reaction need in the spatial domain is met (e.g., when the vehicle spatially passes the position of a detection STOP sign or speed limit sign), regardless of time.

Table 7.3.5 shows our experimental investigation of this design in the four commercial vehicle models. Fig. 7.3.3 illustrates the experimental setup. As shown, in the experiments we first keep the tested vehicle stationary and show the traffic sign on the roadside in front of the vehicle for 1 to 3 seconds (*sign display time*. Then, we hide the traffic sign and wait for 20 to 60 seconds (*sign disappearing time*). Then, we test whether the sign detection result triggered at the sign display time is still memorized by the TSR system after the sign has disappeared for a certain time by driving the vehicle past the original sign-display position. For the STOP sign, the memorization is judged by whether the sign display disappears after driving past the original sign-display position, and for the speed limit sign, this is judged by whether the sign display appears after driving past the original sign-display position. As shown, for three out of the four vehicle models (C2 to C4), the sign detection result can retain even after the sign has already disappeared for 60 seconds, which is way longer than the typical temporal memorization time from object tracking (3 seconds [334, 150]), and will only be cleared/forgotten when the vehicle passes the position of the detected sign.

Such a spatial memorization design can significantly impact how we judge the adversarial attack effect at the TSR system level. For example, for hiding attacks, to achieve a system-level success in which the TSR system is unable to show the sign display at the sign's reaction task period, the attack has to be continuously successful at all possible detection moments that can trigger such memorization before the vehicle passes the sign. As shown in our experiments in Table 7.3.5, such a detection moment can be as short as 1 second. In most recent prior works, the attack success is most commonly judged by first separating the entire sign detection distance range into small distance segments and claiming high attack effectiveness as long as the majority of the distance segments have high success rates [330, 148, 186]. However, due to such spatial memorization, the TSR system-level hiding attack success can only be achieved when *all* these distance segments have high success rates, instead of just the majority, which thus may make the TSR system-level hiding attack success harder than expected. For appearing attacks, such an impact on the system-level attack success is the *opposite*, as it does not really need the majority of the segments to have high success rates; as long as one of them can have a high success rate, the system-level attack effect is achieved.

Observation 2: We discover a spatial memorization design that commonly exists in today's commercial TSR systems, which can keep memorizing a sign detection result until the sign's reaction need in the spatial domain is met (e.g., when the vehicle passes the detected sign's position). This design may create a significant discrepancy between the TSR model-level attack effect and that at the TSR system level.

7.4 Revisiting Existing Metric and Attacks

As discussed above, the newly-discovered spatial memorization design in commercial TSR systems today may create a significant discrepancy between the TSR model-level attack effect

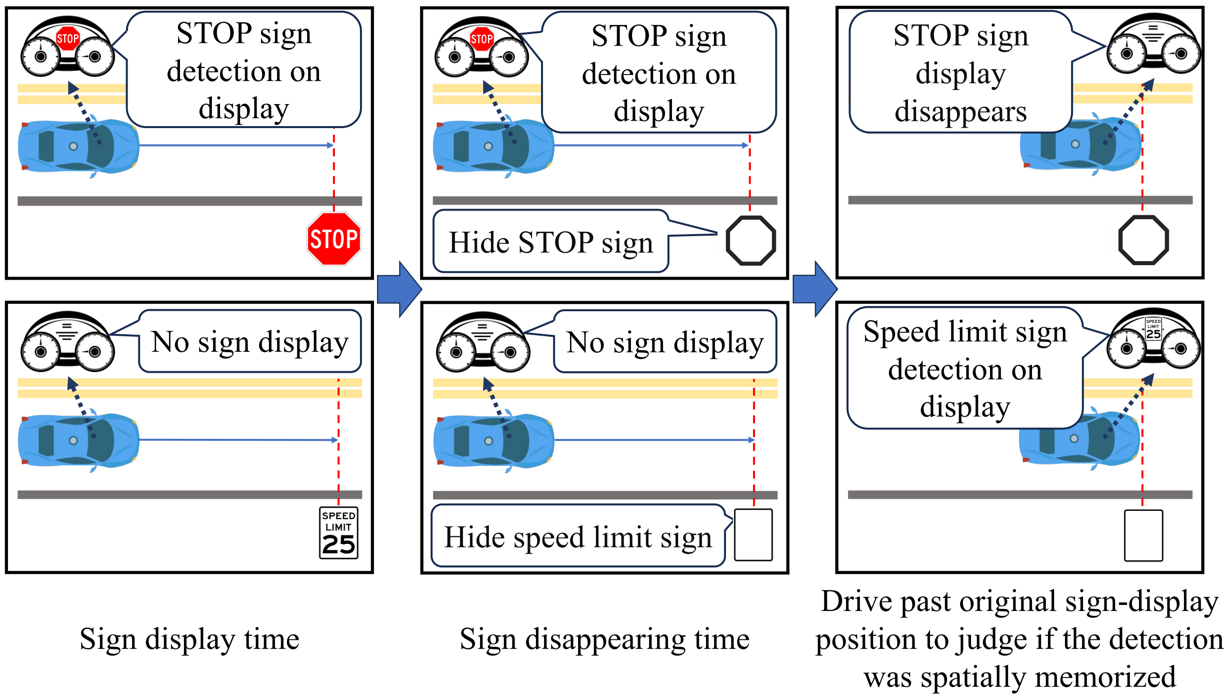

STOP sign detection on display
STOP
STOP sign detection on display
Hide STOP sign
STOP sign display disappears
No sign display
SPEED LIMIT 25
No sign display
Hide speed limit sign
Speed limit sign detection on display
Sign display time
Sign disappearing time
Drive past original sign-display position to judge if the detection was spatially memorized

Figure 7.3.3: Experimental setup for our investigation into the spatial memorization design in commercial TSR systems. As shown, we first show the sign to the vehicle for a short time (*sign display time*), and hide the sign and wait for a certain time duration (*sign disappearing time*). After that, we drive the vehicle past the original sign-display position to measure whether the sign detection result is spatially memorized.

and that at the commercial TSR system level. Thus, in this section we aim to mathematically model the impact of this design on the TSR system-level attack success on both hiding and appearing attack sides, and then use the resulting new TSR system-level metrics to revisit the evaluations, designs, and capabilities of existing attacks in this problem space.

7.4.1 Revisiting Existing Attack Success Metrics

Limitation of Existing Model-Level Attack Success Metrics. In prior works, the TSR model-level attack success metric is generally measured by averaging the per-frame attack success among a set of frames sampled in the sign detection distance range [148, 283, 330, 110]. In this paper, we denote such metrics for object hiding and appearing attacks as f_{HA} and f_{AA}, respectively. However, such averaged per-frame attack success rates do not

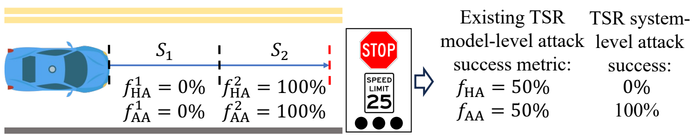


Figure 7.4.1: Illustration of the potentially misleading effect of existing TSR model-level metrics with respect to the TSR system-level attack success. As shown, although f_{HA} and f_{AA} are both 50% for this scenario, the TSR system-level attack success rates are in fact 0% for hiding attack (HA) and 100% for appearing attack (AA) due to spatial memorization.

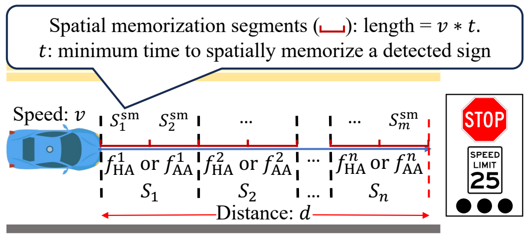


Figure 7.4.2: Setup for calculating the proposed surrogate TSR system-level attack success metric designs (SysHA and SysAA, detailed design in §7.4.1).

take the distribution of the attack effects within a targeted distance range into consideration. Due to the spatial memorization design, a certain distance range segment with an especially high or low attack success rate can directly lead to overall TSR system-level attack success or failure (as found in §7.3.3, the detection result can be memorized within 1 second), which can thus make such existing metrics highly misleading with respect to the TSR system-level attack effect.

Fig. 7.4.1 shows an illustrative example of such a misleading effect. As shown, there are two distance range segments, S_1 and S_2, and the attack success rates are both $f^1_{\text{HA}} = f^1_{\text{AA}} = 0\%$ for S_1 and $f^2_{\text{HA}} = f^2_{\text{AA}} = 100\%$ for S_2. As shown, using the existing f_{HA} and f_{AA} metrics, the attack success rates in this entire distance range will be the average of the attack success rates in S_1 and S_2, and thus are $f_{\text{HA}} = f_{\text{AA}} = 50\%$. However, due to spatial memorization, on the hiding attack side the 100% f^2_{HA} cannot directly lead to the overall system-level attack success, as the TSR system can still 100% detect the sign in S_1, memorize it, and correctly display it at the sign's reaction task period, leading to actually a 0% TSR system-level attack success rate. On the appearing attack side, although f^1_{HA} is 0%, the attack can always (100%) trigger a fake sign detection in S_2 that will be memorized and displayed, and thus the end-to-end TSR system-level attack success rate is actually 100%. As can be seen, due to spatial memorization, the TSR system-level attack success rate can be completely different from that from f_{HA} and f_{AA}, which can lead to highly problematic judgments of an attack's capability at the TSR system level, e.g., concluding that an attack is reasonably effective (50%) when it actually cannot work at all (0%).

New Metric Design: Surrogate TSR System-Level Attack Success Metrics. To avoid such misleading effects of existing TSR model-level attack success metrics, the most direct solution is to perform system-level evaluations on commercial TSR systems like our efforts in §7.3. However, it is highly difficult and also too costly for the academic community to always acquire a substantial number of commercial vehicles for experiments. And also since these commercial TSR systems are black-boxes, it is difficult to perform model design-level security research such as vulnerability cause analysis and defense evaluations. To address this, we thus propose to design *surrogate* TSR system-level attack success metrics that model the spatial memorization effect on top of the existing model-level metrics, which can make them directly calculable using the more readily-accessible academic TSR model-based setups.

We start with hiding attacks. The symbols for calculating the metric are illustrated in Fig. 7.4.2. As shown, d denotes the benign-case detection distance of the targeted TSR model. In alignment with existing model-level metric calculation, we divide d into n *measurement segments*, denoted as $S_i, i \in \{1, \dots, n\}$, and for each segment the averaged per-frame attack success rates can be calculated, denoted as $f_{\text{HA}}^1, f_{\text{HA}}^2, \dots, f_{\text{HA}}^n$. To incorporate the spatial memorization effect, we consider the minimum time to spatially memorize a detected sign as t. To map this spatially memorizable detection time to the detection distance range segments, we calculate the distance traveled by the vehicle during t as $v * t$, with v being the vehicle speed. In this paper, we call each such distance segment of $v * t$ as *spatial memorization segments*, denoted as $S_j^{\text{sm}}, j \in \{1, \dots, m\}, m = \frac{d}{vt}$.

Due to spatial memorization, to achieve the TSR system-level attack success, a hiding attack needs to achieve attack success at every spatial memorization segment in d; otherwise, the sign can be detected and memorized, making the attack fail at the system level. Thus, the system-level attack success rate should be the product of f_{HA} for all S_j^{sm}, i.e., $\prod_{j=1}^{m} f_{\mathrm{HA}}^{j}$. However, to allow flexibility of the f_{HA} success rate measurements, which can incur heavy-weight physical-world experiments, we do not prefer to require the measurement of f_{HA} exactly at the $v * t$ granularity; instead, we aim to design the metric to be calculable for any distance choices for S_i. To achieve this, we thus use the f_{HA}^{i} to approximate the f_{HA}^{j} for all S_j^{sm} inside a measurement segment S_i; the resulting surrogate TSR system-level metric, which we call *SysHA*, is shown in Eq. equation 7.1. Note that for cases when a S_j^{sm} spans multiple measurement segments S_i, we use the average f_{HA} of these segments f_{HA}^{i} as the f_{HA} for S_j^{sm}.

$$\mathrm{SysHA} = \prod_{i=1}^{n} (f_{\mathrm{HA}}^{i})^{\frac{m}{n}} = \prod_{i=1}^{n} (f_{\mathrm{HA}}^{i})^{\frac{d}{nvt}} \tag{7.1}$$

On the appearing attack side, we can use a similar design to model the impacts of spatial memorization on TSR system-level attack success. Here, the difference is that the system-level attack success can be achieved as long as the appearing attack can succeed (and thus spatially memorized) in one of the spatial memorization segments S_j^{sm}. Thus, the probability to achieve an eventual sign appearing at the TSR system level is the negation of the probability that the appearing attack *cannot* succeed (and thus memorized) in *any* of the S_j^{sm}, i.e., $1 - \prod_{j=1}^{m}(1 - f_{\mathrm{AA}}^{j})$. Following the same design in SysHA to allow measurement flexibility of f_{AA}, we use f_{AA}^{i} to approximate f_{AA}^{j} for each S_j^{sm}. The resulting surrogate TSR system-level metric for appearing attack, which we call *SysAA*, is thus:

$$\mathrm{SysAA} = 1 - \prod_{i=1}^{n} (1 - f_{\mathrm{AA}}^{i})^{\frac{d}{nvt}} \tag{7.2}$$

Implications to TSR System-Level Attack Hardness. As informally discussed in §7.3.3, the spatial memorization design may make hiding attacks harder than expected and appearing attacks easier than expected. Now with the mathematical modeling of the spatial memorization's impacts on the TSR system-level attack success above, we can both theoretically and numerically analyze such impacts of the spatial memorization on the TSR system-level attack hardness.

Theorem: When $f^i_{\text{HA}} = f^i_{\text{AA}}$, where $i \in \{1, \ldots, n\}$, SysAA $\geqslant$ SysHA always holds.

Proof. To prove this theorem, Eq. equation 7.2 can be reformulated to the sum of the probabilities of all possible attack result scenarios that can lead to a successful system-level appearing attack. To derive that, we denote the *power set* of all the measurement segments as $\mathcal{P}(S)$, where $S = \{S_i | i \in \{1, \ldots, n\}\}$. Each possible attack result scenario can be described in the form of two subsets of S: A and $S \setminus A$, where A is a subset of $\mathcal{P}(S)$, i.e., $A \in \mathcal{P}(S)$. Among them, the appearing attack for all $S_i \in A$ can succeed, and that for all $S_j \in (S \setminus A)$ fails. Due to spatial memorization, as long as $A \neq \emptyset$, the TSR system-level appearing attack effect can be achieved. Thus, SysAA can be represented as:

$$\text{SysAA} = \sum_{A \in (\mathcal{P}(S) \setminus \emptyset)} \left(\prod_{S_i \in A} (f^i_{\text{AA}})^{\frac{d}{nvt}} \prod_{S_j \in (S \setminus A)} (1 - f^j_{\text{AA}})^{\frac{d}{nvt}} \right) \tag{7.3}$$

When $f^i_{\text{HA}} = f^i_{\text{AA}}$, SysHA $= \prod_{i=1}^{n} (f^i_{\text{HA}})^{\frac{d}{nvt}} = \prod_{i=1}^{n} (f^i_{\text{AA}})^{\frac{d}{nvt}}$, which is actually one instance of $A \in (\mathcal{P}(S) \setminus A)$, i.e., $A = S$. Thus, we can calculate SysAA $-$ SysHA:

$$\begin{aligned} \text{SysAA} - \text{SysHA} &= \\ &\sum_{A \in \mathcal{P}(S) \setminus \{\emptyset, S\}} \left(\prod_{S_i \in A} (f^i_{\text{AA}})^{\frac{d}{nvt}} \prod_{S_j \in (S \setminus A)} (1 - f^j_{\text{AA}})^{\frac{d}{nvt}} \right) \end{aligned} \tag{7.4}$$

For $\forall A \in \mathcal{P}(S) \setminus \{\emptyset, S\}$, $f^i_{\text{AA}} \geqslant 0$ and $(1 - f^j_{\text{AA}}) \geqslant 0$, where $S_i \in A$ and $S_j \in (S \setminus A)$, we can have $\prod (f^i_{\text{AA}})^{\frac{d}{nvt}} \prod (1 - f^j_{\text{AA}})^{\frac{d}{nvt}} \geqslant 0$, thus SysAA $-$ SysHA $\geqslant 0$, and consequently, SysAA $\geqslant$ SysHA. $\square$

Numerical Analysis. The theoretical analysis above can mathematically reveal that SysAA can always be equal or larger than SysHA when f_{HA} and f_{AA} are equal. However, it cannot reveal how large the gap between SysHA and SysAA can be. Thus, we further perform a numerical analysis of SysHA and SysAA. In this analysis, we set $n = m$, and $x = f^i_{\text{HA}} = f^i_{\text{AA}}$ for all $i \in \{1, \ldots, n\}$. Thus, SysHA $= x^m$ and SysAA $= 1 - (1 - x)^m$. In Fig. 7.4.3, we plot the values of SysHA, SysAA, f_{HA}, and f_{AA} when $m = 2, \ldots, 5$. This is a realistic approximation of m since as found in §7.3.3, commercial TSR systems can spatially memorize a detection result within one second. At normal driving speed, the traffic sign can appear for at least 2-5 seconds, and thus there at least exist 2-5 spatial memorization segments. As shown, SysAA is always greater than SysHA when x is between 0 and 1, and the difference (SysAA $-$ SysHA) is at least 50% (when $m = 2$) and can be as high as 93.8% when $m = 5$. This means that even when hiding attacks and appearing attacks seem to have similar model-level attack effectiveness, due to the spatial memorization design the TSR system-level attack effectiveness can have huge differences ($\geqslant$93.8% differences in absolute attack success rate values).

Meanwhile, we can also observe that SysHA and SysAA can both differ significantly from f_{HA} and f_{AA} results. As shown, when $m = 5$, SysAA can be much higher than f_{AA} (46.9% difference in absolute attack success rate values), and SysHA can be much lower than f_{HA} (also 46.9% difference in absolute attack success rate values). This thus numerically proves the potentially misleading effect of existing TSR model-level metrics with respect to the TSR system-level attack effect, which can lead to a misjudgment of the attack effectiveness to the extent of nearly 50% in absolute attack success rate values.

About the novelty and importance of the new metric design. Note that the metric and the concept of spatial memorization might seem straightforward, but no prior research has discovered and analyzed the impact of this important TSR system-level design from the security perspective, nor formulated it mathematically. Before this paper, it was unknown that such a design is commonly used in commercial TSR systems. Additionally, as discussed above, mathematically modelling and quantifying its impacts on adversarial attack success rates as a metric is crucial for the research community to systematically understand real-world system vulnerabilities, especially from the commercial TSR systems perspective. Meanwhile, our theoretical and empirical analyses above further provide mathematically provable insights about the design-level implications and the magnitude of such impacts from the spatial memorization design. These thus all make our newly-proposed surrogate TSR system-level attack success metrics valid and significant scientific contributions.

Meanwhile, we would also like to clarify that we do not intend to claim that the existing TSR model-level metrics do not have important value when compared with our proposed surrogate TSR system-level metric. As explained above, the design of this new metric and the later experimental comparisons of the results from these two metrics are only for the purpose of scientifically understanding how much the newly-observed spatial memorization design in commercial TSR systems today can affect the judgment and understanding of the capability of a certain attack design at the TSR system level.

Observation 3: Due to spatial memorization, hiding attacks are theoretically harder (if not equally hard) than appearing attacks in achieving TSR system-level attack success. Such an attack hardness gap can be huge (e.g., $\geqslant$93.8% absolute differences in the attack success rate values). Meanwhile, due to the lack of consideration of spatial memorization, existing TSR model-level attack success metrics can be highly misleading in judging the TSR system-level attack effect, with a potential of having $\sim$50% absolute attack success rate value differences.

7.4.2 Revisiting Existing Attacks

In this section, We use the SysHA and SysAA metrics to revisit the evaluations, designs, and attack capabilities of the important prior works in this problem space.

Specifically, we revisit (1) both hiding and appearing attacks, with RP_2 [110], SIB [330], FTE [148] as representative examples on the hiding side, and SIB [330] and DM [243] as representative examples on the appearing side; (2) both white-box and black-box transfer attack setups, with the original white-box attack setups evaluated in the original papers (e.g., YOLO v2 (Y2) for RP_2, YOLO v3 (Y3) for SIB, and YOLO v5 (Y5) for FTE) on the white-box side, and a set of representative transfer target models on the black-box transfer attack setup side. Specifically, this transfer target model set includes 6 models in total, 3 from the one-stage model design family (YOLO v8 (Y8) [157], YOLOS (YS) [112], DETR [73]) and 3 from the two-stage model design family (two versions of Faster RCNN with different backbones [226], and Mask RCNN [226]).

In all the experiments, we focus on STOP sign as the representative attack target since it is used the most commonly in prior works in their physical-world evaluation (in Table 7.1.2, five out of the six prior works only used STOP sign in their physical-world experiments for TSR), which can thus best suit our need in this section to revisit the evaluation of prior works. All the attacks are physically printed out and measured outdoor. The distance range measured is 0 to 30m following the setups in prior works [110, 330, 148]. For each attack setup, we calculate the SysHA or SysAA attack success rate by averaging the SysHA or SysAA values across the full combinations of a set of common speeds (25 mph, 30 mph, and 35 mph, the most common speed limits for STOP sign-controlled roads [283]) for v and all possible minimum spatial memorization time t (0.05 seconds to 1 second with a step of 0.05 seconds considering the common camera frame rate 20Hz [53] and the 1-second upper-bound of t found from our experiments in §7.3.3).

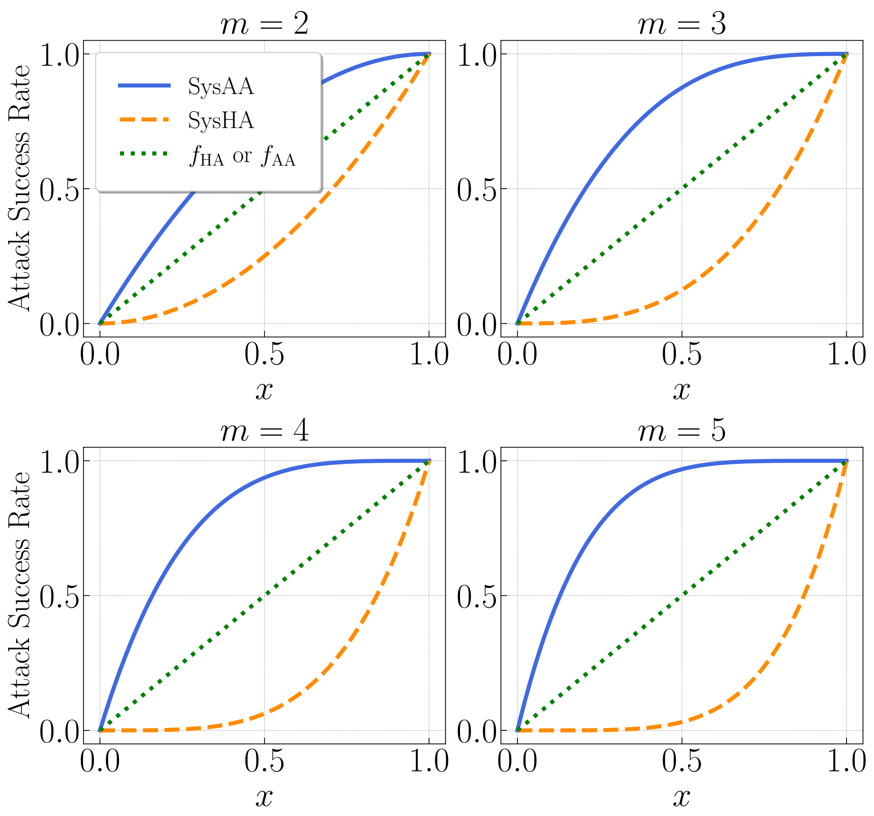


Figure 7.4.3: Numerical analysis of the attack success rate values from SysHA, SysAA, f_{HA}, and f_{AA} when $m = 2, \ldots, 5$. Here, we set $n = m$ and $x = f^i_{\text{HA}} = f^i_{\text{AA}}$ for all $i \in \{1, \ldots, n\}$.

In the following, we report the most notable findings our these revisiting experiments.

White-Box Attack Effectiveness. We start by revisiting the prior works in white-box attack setups. The impact of the spatial memorization design on the TSR system-level attack success is orthogonal to the attacker's knowledge level (e.g., white- or black-box) of the targeted TSR model, and thus it is of interest to use our new metrics to revisit existing attacks even in the white-box attack setting. Table 7.4.1 shows the white-box attack effectiveness of representative existing hiding attacks measured by both f_{HA} and SysHA. When using f_{HA}, the attack success rates are similar to those reported in the original papers [110, 330, 148] and can be claimed as effective (from ∼50% to 90%). However, as numerically analyzed in §7.4.1, due to spatial memorization the TSR system-level attack success rate can be much lower. As shown, when using SysHA, the attack success rates decrease significantly by 2 to 9 times, with the absolute success rate value drops of at least 40%. Most notably, the

Table 7.4.1: White-box attack effectiveness for representative prior works on hiding attacks measured by f_{HA} and SysHA. As shown, while the f_{HA} results can indeed be claimed as effective, it may not be appropriate to claim so with SysHA. This suggests a potential lack of TSR system-level attack effectiveness even in the white-box setting for existing works.

	f_{HA}							SysHA
	Distance ranges (meters)						Ave.	
	0-5	5-10	10-15	15-20	20-25	25-30		
RP_2	41.8%	10.0%	23.8%	65.4%	99.9%	100%	56.8%	6.6%
SIB	84.6%	56.6%	82.0%	99.2%	100%	100%	87.1%	45.1%
FTE	88.9%	57.1%	13.6%	3.1%	47.8%	74.5%	47.5%	5.2%

attack success rates for RP_2 and FTE are dropped to at most 6.6%. Thus, it may not be appropriate to claim that these attacks can be effective at the TSR system level. This suggests a potential lack of TSR system-level attack effectiveness for existing works even in the white-box settings.

Observation 4: When spatial memorization is considered, prior works in this problem space may not be claimed as effective at the TSR system level even in white-box attack settings. Our newly-proposed surrogate TSR system-level metrics can help improve this in the future as they can be leveraged to better approximate the impact of spatial memorization on the TSR system-level attack success.

Black-Box Transfer Attack Effectiveness. We next revisit the black-box transfer attack effectiveness of existing works. Table 7.4.2 shows the results for representative prior works on hiding attacks measured by both f_{HA} and SysHA. As shown, when using f_{HA}, all prior works show a descent transfer attack success rates at ∼40%, which is very similar to the reported numbers from the original papers on average (∼50%). However, when using SysHA, the success rates becomes much lower, which are generally decreased by around 3 times for all three attacks. This is in fact quite similar to the observations from the commercial TSR testing in §7.3.3 and Table 7.3.4, which found that the black-box transfer attack success rates against commercial TSR systems are almost a magnitude lower than those reported

Table 7.4.2: Black-box transfer attack effectiveness for representative prior works on hiding attacks measured by f_{HA} and SysHA. Each success rate value is an average over the results from 6 representative transfer target models: 3 one-stage models (Y8, YS, DETR) and 3 two-stage models (two Faster RCNN with different backbones, MaskRCNN), which are detailed more in §7.4.2. As shown, although the f_{HA} values show descent transfer attack success rates similar to those reported by the original paper, those calculated by SysHA are much lower, similar to the observations from the commercial TSR testing in §7.3.3 and Table 7.3.4.

	Original paper transferability	Transfer attack success rates (averaged over a set of six transfer target models (§7.4.2)							
		f_{HA}							SysHA
		0-5m	5-10m	10-15m	15-20m	20-25m	25-30m	Ave.	
RP_2	18.9%	36.4%	32.0%	29.6%	46.0%	61.3%	50.0%	42.6%	14.5%
SIB	46.1%	20.7%	26.5%	37.2%	42.6%	54.9%	51.2%	38.9%	12.4%
FTE	89.8%	29.2%	36.4%	29.3%	34.0%	45.5%	40.1%	35.7%	11.0%
Ave.	51.6%	28.8%	31.6%	32.0%	40.9%	53.9%	47.1%	39.1%	12.6%

by the original papers on average. This suggests that spatial memorization may indeed be a major factor for the observed much lower-than-expected black-box transfer attack success rates against commercial TSR systems.

Observation 5: When spatial memorization is considered, the black-box transfer attack success rates of prior works at TSR system level can be much lower than expected (only ∼13%) for hiding attack. This suggests that existing hiding attack works are unlikely to have direct impacts on real-world commercial TSR systems in general, which is consistent with our observations in our large-scale commercial TSR systems testing. This suggests that future work in this problem space should focus more on black-box attack settings, which can be more generally enabled by our newly-proposed surrogate TSR system-level attack success metrics.

Hiding vs. Appearing Attack Effectiveness. As numerically analyzed in §7.4.1, it can be much harder to achieve hiding attack success than to achieve appearing attack success at the TSR system level due to spatial memorization. Now with concrete attack examples from prior works, we can more directly study such attack hardness gaps. Table 7.4.3 shows

Table 7.4.3: Attack capability comparison between the hiding and appearing attacks proposed from the same prior work (SIB [330]) measured by f_{HA}, SysHA, f_{AA}, and SysAA. As shown, if using the prior TSR model-level attack success metrics (f_{HA}, f_{AA}), the judgment of the attack capability differences between the proposed hiding and appearing attacks can be the *completely opposite* to those using SysHA and SysAA due to the consideration of spatial memorization.

	Hiding attack		Appearing attack	
SIB [330]	f_{HA}	SysHA	f_{AA}	SysAA
White-box attack	87.1%	45.1%	29.1%	87.6%
Black-box transfer attacks	38.9%	12.4%	31.7%	64.2%

the comparison of the attack success rates for the hiding and appearing attacks from the same prior work SIB [330]. The success rates are calculated for both prior TSR model-level metrics (f_{HA}, f_{AA}) and our newly-proposed surrogate TSR system-level metrics (SysHA, SysAA) under both white-box and black-box transfer attack settings. Here, the black-box transfer attack success rates are calculated by averaging results over a set of 6 representative target models (§7.4.2).

As shown, in both white-box and black-box transfer attack settings, SysHA is lower than f_{HA} and SysAA is higher than f_{AA}, which is consistent with our numerical analysis results (§7.4.1) and is caused by the spatial memorization effect. Interestingly, if we only use prior TSR model-level metrics (f_{HA}, f_{AA}) to judge the attack capabilities between the proposed hiding and appearing attacks, the conclusion will be that the proposed hiding attack is more effective (if not much more effective) than the proposed appearing one, as the f_{HA} results are always higher than the f_{AA} results in both white-box and black-box transfer attack settings; in particular, in the white-box setting, the f_{HA} success rate is ∼3 times of the f_{AA} one. However, when using SysHA and SysAA, the conclusion is the *completely opposite*: the appearing attack success rate is always higher than the hiding attack one in both white-box and black-box transfer attack settings, and the former can be ∼2-5 times higher. This suggests that if not considering spatial memorization, the judgment of the TSR system-level attack capabilities across hiding and appearing attacks can be completely

Table 7.4.4: The white-box appearing attack effectiveness of RP_2 [110] with and without the Nested AE (NAE) design measured by f_{AA} and SysAA. As shown, when using f_{AA}, the NAE design can indeed "significantly improve the robustness of adversarial attack in various distances" according to the original paper that proposed NAE as the key new design contribution [330]: it improves f_{AA} in every distance range segments, leading to a 22% improvements on average). However, when using SysAA with spatial memorization considered, the success rate improvement is almost negligible (∼1%).

RP_2 [110]	f_{AA}						SysAA
	0-5m	5-10m	10-15m	15-20m	20-25m	Ave.	
w/o NAE	86.8%	100%	64.7%	66.9%	19.5%	67.6%	98.2%
w/ NAE	100%	100%	100%	88.3%	25.8%	82.8%	100%

wrong. This further highlights the importance of systematically modelling the impacts of spatial memorization on the TSR system-level attack success, which is exactly what we aim to achieve in the design of SysHA and SysAA.

Observation 6: Using the hiding and appearing attacks proposed from the same prior work, the hiding one can indeed be much harder (2-5 times) than the appearing one in both white-box and black-box transfer attack settings after spatial memorization is considered. However, if using the prior TSR model-level attack success metrics, the judgment of such relative attack hardness differences can be the completely opposite, which thus highlights the necessity of the design of the new SysHA and SysAA metrics.

Judgement of the Value of New Attack Designs. As numerically analyzed in §7.4.1, spatial memorization can significantly impact the attack success at the TSR system level, with a potential of having ∼50% absolute attack success rate value difference. Due to this, it is possible that certain new attack designs proposed in prior works can be seemingly highly beneficial to the attack success when judged using the prior TSR model-level metrics, but when judged with the consideration of spatial memorization, the benefits to the attack success are actually very minimal. This thus may significantly change how we judge the practical value of certain prior attack designs at the TSR system level. Table 7.4.4 shows our investigation into one such example. As shown, in this experiment we compare the white-

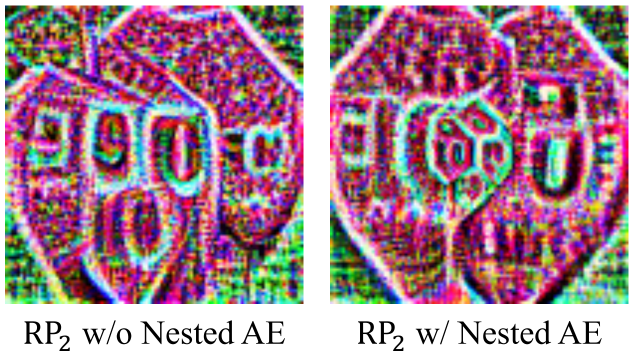


Figure 7.4.4: Visualisation of the adversarial patterns with and without the Nested AE (NAE) design reproduced by us using RP$_2$ (the original design has not been open-sourced). As shown, the "nested" adversarial pattern feature is highly consistent with that reported in the original paper (e.g., Fig. 5 in [330]).

box appearing attack effectiveness of RP$_2$ [110] with and without a specific attack design called Nested AE, which we denote as *NAE*. This design is proposed by Zhao et al. [330] as a key new appearing attack design contribution. In this design, the key idea is to decouple the task of varying distance attack into two pieces: long-distance and short-distance attacks, and distribute them to different regions on the adversarial pattern printed on a whote-sign poster, with the goal of systematically increasing the distance of an appearing attack [330].

Fig. 7.4.4 shows a visualization of the adversarial patterns with and without the NAE design reproduced by us (like what we reported in §7.3.1 for prior works on appearing attacks, this prior attack design also has not been open-sourced so far). As shown, we are able to reproduce the designed adversarial pattern features, with an inner STOP sign-like adversarial pattern "nested" at the center of an outer adversarial pattern, with the inner one for short-distance attack effect and the outer one for long-distance attack effect, which is consistent with the reported adversarial pattern features in the original paper (can be seen by Figure 5 in [330]). As shown by our results in Table 7.4.4, when using f_{AA}, such a design can indeed "significantly improve the robustness of adversarial attack in various distances" as claimed in the original paper [330]: it improves f_{AA} in every distance range segments from 0 to 30m, which is able to increase the average f_{AA} success rates by 22%, from lower than 70% to higher than 80%. However, when using SysAA with spatial memorization considered, the success rate improvement is *almost negligible* (∼2%), from 98.2% to 100%. This is mainly because due to spatial memorization, it can be much easier to achieve a TSR system-level appearing attack success compared to the attack capabilities that can be reflected by the direct TSR model-level metrics like f_{AA}. Thus, even with a naive attack design, the attack success rates at the TSR system level can already be very high, which can thus make the necessity and practical value of more sophisticated appearing attack designs very low.

Observation 7: Due to spatial memorization, the benefits of certain attack designs can be seemingly high (e.g., >20% attack success rate increase) using prior TSR model-level success metrics, but actually nearly negligible (e.g., only 1% increase) at the TSR system level. This thus highlights the necessity of attack success metrics that can incorporate the impact of spatial memorization, such as SysHA and SysAA, with regard to the judgment of the necessity and practical value of any given attack designs in this problem space, for both the designs in the past or those in the future.

7.5 Discussions

In this section, we provide detailed discussion on the ethics, limitations, and future work.

7.5.1 Ethics

In discussing the ethical considerations of our measurement study, it is crucial to highlight the measures taken to ensure safety and responsibility. Our experiments with commercial vehicles are conducted on the roof of a parking structure, a controlled environment where we can ensure the absolute absence of other vehicles and people when performing the experiments. Additionally, we take precautions to make sure that the adversarial attacks are not visible anywhere from public roads, thereby eliminating any risk to others.

Recognizing the direct impacts of our study on the security of commercial vehicles, we have performed responsible vulnerability disclosure to all the potentially affected vehicle manufacturers following the ethical standard in the security community. Specifically, this involved informing the vehicle manufacturers for all the vehicle models we tested (i.e., C1 to C4) about our testing results, especially those with vehicle models that were tested vulnerable such as the the vehicle manufacturer for C2; we have already done all these. This proactive approach allows the manufacturers to address and mitigate any potential risks posed by these attacks, ensuring the safety and security of their vehicles. Additionally, even in this submission version, we do not directly reveal the exact vehicle brands and models for C1 to C4 to protect the potentially-affected companies (§7.3).

7.5.2 Limitations and Future Work

Threats to Validity for the Statistical Significance in Commercial Systems Testing Results. To scientifically interpret our commercial system testing results, it is important to understand their statistical significance. First, for the overall transfer attack success rates against commercial systems over all 30 attack test combinations, the result (6.67% in Table 7.3.4) is statistically significant with $p < 0.02$ using Z-Test [167], Binomial Test [274], One-Sample T-Test [235], and Wilcoxon Signed-Rank Test [292]. Second, for the result of each test combination, we technically cannot compute the statistical significance values since the variance for each is 0 (all failure or all success as shown in Table 7.3.4) and statistical testing methods are designed for data samples with variance (and thus their calculation generally requires division by the standard deviation [273, 274, 167, 292, 235, 259, 245, 197, 140, 62], which is 0 in our case). For our case, this may not be addressable by simply increasing the number of attempts, since the root cause is the lack of variation in the results, which is likely to continue since the output of each test run for us is binary (failure or success). In hypothesis testing, if all observed values are exactly the same as the hypothesized value (i.e., there is no variation in the data), it means that the data perfectly matches the null hypothesis [251] and there is no need to conduct further statistical tests to determine significance, as the data inherently confirms the hypothesis [116, 117]. In our experiments, this is indeed the case as (1) we observe no variations during the 3 attempts and (2) considering the spatial memorization design, in each attempt there are also multiple TSR model-level tests when the vehicle approaches the sign, but still no variations were observed, which made us believe that the results are unlikely to change even if we try a few more times (we cannot afford a significant increase in the number of attempts as we explain below). Note that this is also already statistically more rigorous than all prior works in this: they only tried once for each attack test [243, 148] (we have confirmed this with the authors), which is not enough to even calculate variance.

Nevertheless, it is indeed possible that if we significantly increase the number of attempts, some variations will appear and thus allow us to calculate statistical significance. However, since we need to manually drive the car past the sign and circle back to the same starting point for each attempt, and also need to manually take down and put up new adversarial patterns, at 3 attempts and 14 different tested signs (2 benign, 12 attacks) we need to spend over 2 hours per vehicle model. If we largely increase the attempt number, it may not be possible to control the lighting conditions to be comparable across these tests, which is one of the most critical factors for the effectiveness of physical-world adversarial attacks [330, 148, 309]. For example, if we conduct at least 30 attempts per test as commonly suggested in statistical testing [218], it will cost at least 10 hours in total and the lighting conditions will be completely changed. Note that this is an inherent limitation for any outdoor experimental setup on commercial vehicles. To address this, one possibility is to use an indoor vehicle testing facility with controlled lighting conditions. However, we are not aware of any prior works in AD security that can have such a setup, and also it is unclear whether such a setup can accurately simulate outdoor conditions, as the targeted physical environment for TSR adversarial attacks is outdoor. We thus leave the solution exploration of this to future work.

More Root Cause Analysis for Commercial TSR System Testing Results. In this work, we were able to perform the first large-scale measurement of existing physical-world adversarial attacks on commercial TSR systems by performing black-box transfer attack testing on the commercial systems. Although this was able to fill the critical research gap in achieving a more general understanding of existing attacks' impacts on real-world commercial TSR systems, certain aspects remain partially understood , for example the reason why the TSR function of certain commercial systems can actually be much more vulnerable than academic TSR models, and also why the two successful attack setups are limited to the STOP sign detection. There are multiple possible causes we can speculate, e.g., for the former due to the need to reduce false alarm rates in real-world deployment, or due to the long deployment cycle to integrate latest academic model designs; and for the latter due to the need to customize the detection accuracy for certain sign types because of real-world deployment or customer needs. However, due to the black-box nature of these commercial systems, it requires more systematic follow-up studies to scientifically answer these questions, which we believe is one of the most highly desired future works.

Defense-Side Explorations. In this work, we mainly focus on measuring and understanding the potential gap between existing academic research and real-world commercial systems on the attack side. In the future, it is also of interest to explore such a potential gap on the defense side. Such a gap may indeed exist, for example, due to spatial memorization, hiding attacks can be easier to be defended against at the TSR system level than expected, since as long as the defense method can prevent the hiding attack success in *any* of the spatially memorizable sign detection periods, it can prevent the hiding attack success at the TSR system level. For appearing attacks, this will be the opposite since to prevent the appearing attack success at the TSR system level, the defense method has to prevent the appearing attack success in *all* of the spatially memorizable sign detection periods. We thus leave a systematic investigation of these aspects to the future work.

About misclassification attacks. In this work, we focus on the two most representative TSR adversarial attack types so far with highly demonstrated physical-world realism, hiding and appearing attacks (§7.2.2). Meanwhile, there are also prior works studying misclassification attacks on TSR models (i.e., changing the detection from one sign to another) [148, 111], which is less studied in the security community (potentially due to their relatively indirect attack consequences compared to direct sign hiding or appearing) but can also be of interest to be studied from the commercial systems perspective, especially if we consider the potential impact from spatial memorization. Specifically, for such attacks, if the attack doesn't always succeed/fail in every spatial-memorization segment, the detected sign class will change across spatial-memorization segments. The impact of spatial memorization will depend on whether a later-detected different sign class will override a previously-memorized sign class. For example, if the design is to not override, the TSR system-level success depends on the model-level success of the first/farthest-to-the-sign spatial-memorization segment. If the design is to override, for speed limit signs the TSR system-level success depends on the last/nearest-to-the-sign spatial-memorization segment, while for STOP sign the driver will see alternating/simultaneous display of correct and incorrect signs and thus it may require to newly define the TSR system-level success from the driver's perspective. To effectively answer these new questions, a separate follow-up study is required since it will require new research methodology designs starting from the commercial system testing stage, which we thus leave to future work.

Impact from Other Possible Data Sources for TSR. Our current commercial TSR system testing results are unlikely to be influenced by other possible sources for TSR such as GPS/map. For example, since all the tested signs are temporarily placed at rooftop of a parking structure, no existing maps can have these sign information. Thus, the failed attack attempts cannot be due to possible sign information retrieved from GPS/map. Meanwhile, out of the 13 car brands providing TSR in Table 1, for 12 of them (including all the brands we tested) we cannot find any evidence that fusion is used; specifically, for them their official websites or vehicle manuals only mention the use of camera without any mention of the use of other sources. Nevertheless, systematically understanding whether fusion can be an impact factor on the current results (and if so, how much) can be an interesting follow-up research direction, especially those on designing new analysis methods to systematically understand the root causes of many current observations given the black-box nature of these systems as discussed above. We thus leave this to future work.

7.6 Conclusion

In this paper, we conduct the first large-scale measurement of physical-world adversarial attacks against commercial TSR systems. Our testing results reveal that although it is possible for existing attack works from academia to have highly reliable (100%) attack success against certain commercial TSR system functionality, such black-box commercial system attack capabilities are not generalizable, leading to a much lower-than-expected black-box transfer attack success rates overall. We find that one potential major factor is the spatial memorization design that commonly exists in today's commercial TSR systems. We design new attack success metrics that can mathematically model the impacts of this design on the TSR system-level attack success, and use them to revisit existing attacks. Through these efforts, we uncover 7 novel observations, some of which can directly challenge the observations or claims in prior works due to the use of our new metrics. We hope that the results and new insights from this work can help inspire and facilitate more practically meaningful and impactful research in this critical problem space.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

This dissertation aims to bridge the two general yet crucial limitations of prior security analysis in autonomous systems: the limited practicality in real-world autonomous system setups, either due to 1) their sole focus on the AI component alone, which makes it non-trivial to transfer their component-only attack effects to the system level, or 2) their research scope limited to academic prototypes instead of real-world systems. This dissertation focuses on novel vulnerability discovery, measurement, and attack designs for safety-critical autonomous systems from practicality perspectives. The research contributions can be summarized as follows:

Gap between AD system level and AI component level:

- *MSF-ADV (Chapter 3):* I am the first to study security issues of MSF-based AD perception and the first to challenge the basic MSF design assumption in the AD context. I successfully design and engineer a physical-world adversarial attack aiming at generating adversarial 3D object to mislead a victim AD vehicle to fail in detecting it and thus crash into it. I adopt an optimization-based approach that addresses two main design challenges: non-differentiable target camera and LiDAR sensing systems, and non-differentiable cell-level aggregated features used by LiDAR. I also design strategies to enhance the attack robustness, stealthiness, and physical-world realizability. I evaluate the attack on MSF algorithms in representative production-grade AD systems in real-world driving scenarios. The attack is shown to achieve more than 91% success rates across different object types and MSF algorithms. Such high effectiveness can also be achieved with (1) high stealthiness, (2) high robustness to victim positions, (3) high transferability across MSF algorithms, and (4) high physical-world realizability after being 3D-printed and captured by LiDAR and camera devices. To understand the end-to-end safety impact, I further evaluate the proposed attack on a production-grade simulator, and show that the attack can cause a 100% vehicle collision rate to a production-grade AD system. I also evaluate and discuss defense strategies.

- *ControlLoc (Chapter 4):* I propose the first practical hijacking attack on AD perception using the monitor as the attack vector, to alter the location and shape of objects. This attack can cause vehicle collisions or unnecessary emergency stops. I introduce a novel attack framework, ControlLoc, to generate physical-world adversarial patches. This includes patch location preselection, BBOX filters, loss function designs, and a novel optimization method. I evaluate ControlLoc on multiple AD perception systems including industry-grade ones. ControlLoc is effective in the real world with a real vehicle across different backgrounds, outdoor light conditions, hijacking directions, and angles. It causes AD system-level effects like collisions in a production AD simulator.

- *SysAdv (Chapter 5):* I conduct the first measurement study on the system-level effect of the representative prior object-evasion attacks with my proposed novel evaluation framework (i.e., system model) including 4 popular object detectors and 3 state-of-the-art object-evasion attacks. I identify the limitations of prior works which hinder them in potently achieving system-level effects and propose SysAdv, a system-driven adversarial object-evasion attack with the system model in AD context. I further evaluate SysAdv and show that the system-level effect of SysAdv can be significantly improved, i.e., the system violation rate increases by around 70%.

- *SlowTrack (Chapter 6):* I am the first to study availability-based adversarial attacks considering the entire AD perception pipeline and find that previous object detection-based latency attack strategies may not induce potent system-level effects. I propose a novel attack framework SlowTrack to systematically generate the latency adversarial attacks on camera-based AD perception by designing a two-stage attack strategy and proposing three novel loss functions. SlowTrack is tested on four popular camera-based AD perception pipelines across different hardware, showing increase in latency and boost in system-level effects.

Gap between academic prototypes and real-world systems: In Chapter 7, I conduct the first large-scale measurement of physical-world adversarial attacks against commercial TSR systems. Our testing results reveal that although it is possible for existing attack works from academia to have highly reliable (100%) attack success against certain commercial TSR system functionality, such black-box commercial system attack capabilities are not generalizable, leading to a much lower-than-expected overall black-box transfer attack success rates. I discover a spatial memorization design that commonly exists in today's commercial TSR systems, which can keep memorizing a sign detection result until the sign's reaction need in the spatial domain is met (e.g., when the vehicle passes the detected sign's position). This discovery is crucial as it is shown to be capable of creating a significant discrepancy between the TSR model-level attack effect and that at the TSR system level. I mathematically model the impact of this design on the TSR system-level attack success for both hiding and appearing attacks, resulting in new attack success metric designs that can systematically consider the spatial memorization effect. I then use them to revisit the evaluations, designs, and capabilities of existing attacks in this problem space. Through the commercial TSR system measurements, new metric designs and analysis, and the revisiting of existing attacks, I uncover a total of 7 novel observations compared to existing knowledge in this problem space, some of which directly challenge the observations or claims in prior works due to the introduction of the new attack success metrics.

8.2 Future Work

Following my current research, I plan to keep exploring, analyzing and solving security problems in the safety-critical autonomous systems. I am dedicated to improving the practicality of the current security research such as delving deeper into the system and AI component gaps by providing general solution to measure the difference between AI component level effect and system level effect and improve their effectiveness. This includes developing general/unified solutions to quantify and bridge the discrepancies between AI component level effects and system level effects. This will also enhance our understanding of the actual impact of these vulnerabilities, leading to more effective and practical security solutions. My research goal is to develop a set of general security solution frameworks for autonomous systems that can be applied across application domains. Below, I elaborate on the specific research paths I intend to pursue along my vision.

- **Extending beyond integrity and availability to other security properties such as authenticity and confidentiality:** This dissertation primarily addresses the integrity of autonomous systems and conducts some of the initial exploratory studies on their availability. Although availability has received less attention, other crucial security properties, such as authenticity and confidentiality, remain under-explored in this research field. Building on the methods developed for integrity and availability, my future research aims to expand into these additional security areas. For example, as autonomous systems increasingly rely on high-resolution sensors, such as cameras and LiDAR, sensitive or private data may be inadvertently collected during vehicle operation. This raises potential confidentiality concerns, particularly when this data is transmitted to external servers. Addressing such security vulnerabilities presents a significant scientific gap, which my research intends to explore further.

- **Extensive AI component-level and system-level security research:** It is a widely held belief that AI component-level errors do not inherently result in system-level consequences. My initial explorations, especially those emphasizing integrity and availability in AD perception [283, 70, 194, 195], illuminate some facets of this issue. Yet, the vast expanse of this field is largely uncharted. Numerous studies [252] have a narrow focus on single-component security, often presupposing that theoretical insights will seamlessly apply to real-world autonomous system contexts. This perspective overlooks the imperative of empirically validating how such vulnerabilities affect operations in tangible settings. Therefore, this necessitates a synthesis of research and extensive engineering undertakings—possibly providing general/unified platform as well as revisiting and fine-tuning methodologies from past research and rigorously testing them in an real-world autonomous systems. Thus, novel security analysis on both AI component level and system level can be proposed.

- **Physical-world realizable security analysis with realistic threat model:** Contemporary AI-based research primarily navigates within digital confines, often overlooking the real-world implications of such theoretical threats, especially in autonomous systems. This gap necessitates a shift towards assessing these digital vulnerabilities' practical impacts and risks in physical scenarios. My research aims to bridge this divide, focusing on the tangible aspects of theoretical threats. By employing empirical analyses and real-world testing based on previous studies [71, 239, 283], the initiative will evaluate the robustness of current autonomous systems against actual physical-world adversarial tactics. This approach seeks not only to bring depth to our understanding of autonomous system security but also to enhance real-world safety protocols by grounding digital threats in reality. In addition, a substantial portion of AI-related autonomous system security research relies on threat models that do not always align with real-world conditions, such as assuming direct access to system internals or compromised training data. These unrealistic bases may diminish the practicality of security strategies. My goal is to cultivate more practical threat models, anchoring security assessments in feasible attack scenarios to foster defenses that are genuinely equipped to counteract realistic threats in AI-related autonomous systems.

- **Systematical commercial system security analysis:** Building on practical security research in autonomous systems, it is imperative to delve into the nuances of commercial systems. While these systems are inherently robust, handling real-world intricacies, the exploration of their potential vulnerabilities remains superficial. My research [284] provides an initial step on this direction. I aim to keep going beyond theoretical understanding, pioneering in-depth offensive security analyses specifically for these sophisticated environments. My research endeavors to enhance the resilience of safety-critical autonomous systems, guiding their evolution in both security and safety realms.

- **Defensive research:** Currently, while a plethora of security vulnerabilities in autonomous systems has been identified, the landscape of effective defensive solutions remains sparse, particularly in the realm of attack prevention.

  - **Model-Level defense.** Certain safety-critical vulnerable AI components in autonomous systems—such as lane detection [239] and MSF perception [71]—lack robust defenses. While generic AI defenses, such as input transformation, do exist [71], they often fail to defend against the attacks on autonomous systems [71, 239]. Consequently, more advanced defense mechanisms, such as certified robustness [175], have been proposed for various tasks, including classification and object detection, due to their ability to provide strong theoretical guarantees. However, applying certified robustness in current autonomous systems poses significant challenges: low efficiency that damages real-time applicability especially in AD systems; vulnerability to adaptive attacks [132]; and difficulties in defending against adversarial attacks in the 3D domain [71]. These limitations motivate my research to leverage invariable factors such as physics invariance to narrow the scope of potential perturbations, enabling the development of robust and efficient defenses for real-world autonomous systems.

– **System-Level defense.** To mitigate these vulnerabilities, a fundamental approach is to perform the system-level defense. This can be achieved through various strategies such as sensor fusion: enhancing robustness by integrating more sensor sources and infrastructure-based defense: combining results from infrastructure [191]. Currently, these technologies are still in their early stages of development. Moving forward, my plan is to continue developing innovative methodology to generally enhance the system-level robustness of autonomous systems.

Bibliography

[1] 3D Printing Online. https://formlabs.com/software/.

[2] 40+ Corporations Working On Autonomous Vehicles. https://www.cbinsights.com/research/autonomous-driverless-vehicles-corporations-list.

[3] Adding Gaussian Noise. https://pytorch.org/docs/stable/tensors.html.

[4] Amazon Mechanical Turk. https://www.mturk.com.

[5] Apollo Models. https://github.com/ApolloAuto/apollo/tree/r5.5.0/modules/perception/production/data/perception/lidar/models/cnnseg.

[6] Autoware Self-driving Vehicle on a Highway. https://www.youtube.com/watch?v=npQMzH3j_d8.

[7] Autoware.AI. https://www.autoware.ai//.

[8] Avis will Service Waymo's Self-driving Minivans. https://www.theverge.com/2017/6/26/15873236/avis-waymo-google-self-driving-cars-vans.

[9] Baidu Launches Public Robotaxi Trial Operation. https://www.globenewswire.com/news-release/2019/09/26/1921380/0/en/Baidu-Launches-Public-Robotaxi-Trial-Operation.html.

[10] Baidu launches their open platform for autonomous cars–and we got to test it. https://technode.com/2017/07/05/baidu-apollo-1-0-autonomous-cars-we-test-it/.

[11] Brake Distance. http://www.csgnetwork.com/stopdistcalc.html.

[12] Carma Platform. https://github.com/usdot-fhwa-stol/carma-platform.

[13] COCO Dataset. http://cocodataset.org/.

[14] Curvature. https://en.wikipedia.org/wiki/Gaussian_curvature.

[15] Experimental Security Research of Tesla Autopilot. https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research_of_Tesla_Autopilot.pdf.

[16] FormLabs. https://formlabs.com/software/.

[17] Genetic Algorithm. https://pypi.org/project/geneticalgorithm/.

[18] Inside Waymo's Secret World for Training Self-Driving Cars. https://www.theatlantic.com/technology/archive/2017/08/inside-waymos-secret-testing-and-simulation-facilities/537648/.

[19] Intro to Rendering, Ray Casting. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-837-computer-graphics-fall-2012/lecture-notes/MIT6_837F12_Lec11.pdf.

[20] LARGE-FORMAT 3D PRINTER FOR INDUSTRIAL APPLICATIONS. https://bigrep.com/bigrep-one/.

[21] Mesh Simplification. http://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/08_Simplification.pdf.

[22] Navigant Research Names Waymo, Ford Autonomous Vehicles, Cruise, and Baidu the Leading Developers of Automated Driving Systems. https://www.businesswire.com/news/home/20200407005119/en/Navigant-Research-Names-Waymo-Ford-Autonomous-Vehicles.

[23] Our Project Website. https://sites.google.com/view/cav-sec/msf-adv.

[24] Pillow (PIL Fork). https://pillow.readthedocs.io/en/stable/.

[25] Pony.ai Tech. https://www.pony.ai/en/tech.html.

[26] Tesla AutoPilot: Traffic Light and Stop Sign Control.

[27] Traffic Cone. https://en.wikipedia.org/wiki/Traffic_cone.

[28] Tri. Interpolation. en.wikipedia.org/wiki/Trilinear_interpolation.

[29] Understanding Accuracy, Precision, and Tolerance in 3D Printing. https://formlabs.com/blog/understanding-accuracy-precision-tolerance-in-3d-printing/.

[30] UPS joins race for future of delivery services by investing in self-driving trucks. https://abcnews.go.com/Business/ups-joins-race-future-delivery-services-investing-driving/story?id=65014414.

[31] User Study: Anomalous Traffic Cone Survey. https://drive.google.com/file/d/1EqtQL6m1ZPNOQGs6pbAM25WFT8D58EC2/view.

[32] Velodyne Alpha Prime. https://autonomoustuff.com/product/velodyne-vls-128/.

[33] Waymo has launched its commercial self-driving service in Phoenix - and it's called 'Waymo One'. https://www.businessinsider.com/waymo-one-driverless-car-service-launches-in-phoenix-arizona-2018-12.

[34] Waymo Tech. https://waymo.com/tech/.

[35] YOLOv3 Darknet. https://pjreddie.com/darknet/yolo/.

[36] Does your car have automated emergency braking? It's a big fail for pedestrians. https://www.zdnet.com/article/does-your-car-have-automated-emergency-braking-its-a-big-fail-for-pedestrians/, 2019.

[37] Model Hacking ADAS to Pave Safer Roads for Autonomous Vehicles. https://www.mcafee.com/blogs/other-blogs/mcafee-labs/model-hacking-adas-to-pave-safer-roads-for-autonomous-vehicles/, 2020.

[38] LGSVL Simulator. https://www.lgsvlsimulator.com/, 2022.

[39] AASHTO. *Policy on Geometric Design of Highways and Streets (7th Edition)*. 2018.

[40] M. Abdulhamid and O. Amondi. Collision Avoidance System using Ultrasonic Sensor. *Land Forces Academy Review*, 25(3):259–266, 2020.

[41] A. Abed Abud, B. Abi, R. Acciarri, M. Acero, M. R. Adames, G. Adamov, M. Adamowski, D. Adams, M. Adinolfi, A. Aduszkiewicz, et al. Separation of track- and shower-like energy deposits in protodune-sp using a convolutional neural network. *The European Physical Journal C*, 82(10):903, 2022.

[42] B. Abi, R. Acciarri, M. Acero, G. Adamov, D. Adams, M. Adinolfi, Z. Ahmad, J. Ahmed, T. Alion, S. Alonso Monsalve, et al. Neutrino interaction classification with a convolutional neural network in the dune far detector. *Physical Review D*, 102(9):092003, 2020.

[43] A. A. Abud, B. Abi, R. Acciarri, M. Acero, M. Adames, G. Adamov, M. Adamowski, D. Adams, M. Adinolfi, C. Adriano, et al. Reconstruction of interactions in the protodune-sp detector with pandora. *The European Physical Journal C*, 83(7):618, 2023.

[44] A. A. Abud, B. Abi, R. Acciarri, M. A. Acero, M. R. Adames, G. Adamov, M. Adamowski, D. Adams, M. Adinolfi, C. Adriano, et al. Highly-parallelized simulation of a pixelated lartpc on a gpu. *Journal of instrumentation*, 18(04):P04034, 2023.

[45] E. ACKERMAN. Slight Street Sign Modifications Can Completely Fool Machine Learning Algorithms. https://spectrum.ieee.org/slight-street-sign-modifications-can-fool-machine-learning-algorithms, 2017.

[46] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky. BoT-SORT: Robust Associations Multi-Pedestrian Tracking. *arXiv preprint arXiv:2206.14651*, 2022.

[47] H. Akatsuka and S. Imai. Road Signposts Recognition System. Technical report, SAE Technical Paper, 1987.

[48] A. Alexos, J. Liu, A. Tiwari, K. Bhardwaj, S. Hayes, P. Baldi, S. Bukkapatnam, and S. Bhandarkar. Machine learning-enhanced prediction of surface smoothness for inertial confinement fusion target polishing using limited data. *arXiv preprint arXiv:2312.10553*, 2023.

[49] R. Alika, E. M. Mellouli, and E. H. Tissir. Optimization of Higher-Order Sliding Mode Control Parameter using Particle Swarm Optimization for Lateral Dynamics of Autonomous Vehicles. In *IRASET*, pages 1–6. IEEE, 2020.

[50] F. Almutairy, T. Alshaabi, J. Nelson, and S. Wshah. Arts: Automotive repository of traffic signs for the united states. *IEEE Transactions on Intelligent Transportation Systems*, 22(1):457–465, 2019.

[51] M. Alzantot, Y. Sharma, S. Chakraborty, H. Zhang, C.-J. Hsieh, and M. B. Srivastava. Genattack: Practical Black-box Attacks with Gradient-free Optimization. In *GECCO*, 2019.

[52] apollo. Baidu Apollo team (2017), Apollo: Open Source Autonomous Driving. https://github.com/ApolloAuto/apollo, 2023.

[53] B. Apollo. Baidu Apollo. http://apollo.auto, 2022.

[54] B. Apollo. Baidu Apollo. https://www.apollo.auto/, 2022.

[55] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. Synthesizing Robust Adversarial Examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018.

[56] A. Athalye and I. Sutskever. Synthesizing Robust Adversarial Examples. In *International Conference on Machine Learning (ICML)*, 2018.

[57] D. Atlanta. 25 MPH Speed Limit - Frequently Asked Questions, 2020.

[58] P. F. Baldi, S. Abdelkarim, J. Liu, J. K. To, M. D. Ibarra, and A. W. Browne. Vitreoretinal surgical instrument tracking in three dimensions using deep learning. *Translational vision science & technology*, 12(1):20–20, 2023.

[59] D. Belayneh, F. Carminati, A. Farbin, B. Hooberman, G. Khattak, M. Liu, J. Liu, D. Olivito, V. B. Pacela, M. Pierini, et al. Calorimetry with deep learning: particle simulation and reconstruction for collider physics. *The European Physical Journal C*, 80(7):1–31, 2020.

[60] J. Beltran, C. Guindel, F. M. Moreno, D. Cruzado, F. Garcia, and A. De La Escalera. Birdnet: a 3D Object Detection Framework from Lidar Information. In *ITSC*, pages 3517–3523. IEEE, 2018.

[61] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple Online and Realtime Tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[62] K. A. Bollen and K. H. Barb. Pearson's r and coarsely categorized measures. *American Sociological Review*, pages 232–239, 1981.

[63] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial Patch. *arXiv preprint arXiv:1712.09665*, 2017.

[64] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial Patch. In *arXiv:1712.09665*, 2017.

[65] O. Bryniarski, N. Hingun, P. Pachuca, V. Wang, and N. Carlini. Evading adversarial example detection defenses with orthogonal projected gradient descent. *arXiv preprint arXiv:2106.15023*, 2021.

[66] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[67] V. California. Driving in California. https://www.visitcalifornia.com/experience/driving-california/, 2022.

[68] Y. Cao, S. H. Bhupathiraju, P. Naghavi, T. Sugawara, Z. M. Mao, and S. Rampazzi. You Can't See Me: Physical Removal Attacks on LiDAR-based Autonomous Vehicles Driving Frameworks. In *USENIX Security Symposium*, 2023.

[69] Y. Cao, N. Wang, C. Xiao, D. Yang, J. Fang, R. Yang, Q. A. Chen, M. Liu, and B. Li. 3D Adversarial Object against MSF-based Perception in Autonomous Driving. In *MLSys*, 2020.

[70] Y. Cao, N. Wang, C. Xiao, D. Yang, J. Fang, R. Yang, Q. A. Chen, M. Liu, and B. Li. Invisible for both Camera and LiDAR: Security of Multi-Sensor Fusion based Perception in Autonomous Driving Under Physical World Attacks. In *IEEE S&P 2021*, May 2021.

[71] Y. Cao, N. Wang, C. Xiao, D. Yang, J. Fang, R. Yang, Q. A. Chen, M. Liu, and B. Li. Invisible for both Camera and LiDAR: Security of Multi-Sensor Fusion based Perception in Autonomous Driving Under Physical-World Attacks. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 176–194. IEEE, 2021.

[72] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao. Adversarial Sensor Attack on LiDAR-based Perception in Autonomous Driving. In *ACM CCS*, 2019.

[73] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-End Object Detection with Transformers. In *ECCV*, 2020.

[74] N. Carlini and D. Wagner. MagNet and "Efficient Defenses against Adversarial Attacks" are not Robust to Adversarial Examples. *arXiv*, 2017.

[75] N. Carlini and D. Wagner. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE symposium on Security and Privacy*, pages 39–57. IEEE, 2017.

[76] N. Carlini and D. A. Wagner. Towards Evaluating the Robustness of Neural Networks. In *IEEE S&P*, 2017.

[77] M. Carranza-García, J. Torres-Mateo, P. Lara-Benítez, and J. García-Gutiérrez. On the Performance of One-stage and Two-stage Object Detectors in Autonomous Vehicles using Camera Data. *Remote Sensing*, 13(1):89, 2020.

[78] S. Challa. *Fundamentals of Object Tracking.* Cambridge University Press, 2011.

[79] V. Chandrasekaran, B. Tang, N. Papernot, K. Fawaz, S. Jha, and X. Wu. Rearchitecting Classification Frameworks For Increased Robustness. In *arXiv:1905.10900*, 2019.

[80] E.-C. Chen, P.-Y. Chen, I. Chung, C.-r. Lee, et al. Overload: Latency Attacks on Object Detection for Edge Devices. *arXiv preprint arXiv:2304.05370*, 2023.

[81] J. Chen, F. Wang, C. Li, Y. Zhang, Y. Ai, and W. Zhang. Online Multiple Object Tracking Using a Novel Discriminative Module for Autonomous Driving. *Electronics*, 10(20):2479, 2021.

[82] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin. MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[83] S.-T. Chen, C. Cornelius, J. Martin, and D. H. P. Chau. ShapeShifter: Robust Physical Adversarial Attack on Faster R-CNN Object Detector. In *ECML PKDD*, pages 52–68. Springer, 2018.

[84] T. Chen, S. Liu, S. Chang, Y. Cheng, L. Amini, and Z. Wang. Adversarial Robustness: From Self-Supervised Pre-Training to Fine-Tuning. In *CVPR*, pages 699–708, 2020.

[85] X. Chen, C. Fu, F. Zheng, Y. Zhao, H. Li, P. Luo, and G.-J. Qi. A Unified Multi-Scenario Attacking Network for Visual Object Tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1097–1104, 2021.

[86] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-View 3D Object Detection Network for Autonomous Driving. In *CVPR*, 2017.

[87] Z. Cheng, H. Choi, S. Feng, J. C. Liang, G. Tao, D. Liu, M. Zuzak, and X. Zhang. Fusion is Not Enough: Single Modal Attack on Fusion Models for 3D Object Detection. In *The Twelfth International Conference on Learning Representations*, 2024.

[88] L. Chi and Y. Mu. Deep Steering: Learning End-to-End Driving Model from Spatial and Temporal Visual Cues. *arXiv*, 2017.

[89] V. Cockayne. Waymo self-driving taxis deliver for Uber Eats in Arizona. https://www.washingtontimes.com/news/2024/apr/4/waymo-self-driving-taxis-deliver-uber-eats-arizona/, 2024.

[90] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter. Certified Adversarial Robustness via Randomized Smoothing. *ICML*, 2019.

[91] Comma.AI. OpenPilot. https://github.com/commaai/openpilot.

[92] S. O.-R. A. V. S. Committee et al. Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. *SAE International: Warrendale, PA, USA*, 2018.

[93] CopRadar.com. Vehicle Acceleration and Braking. https://copradar.com/chapts/references/acceleration.html, 2017.

[94] J. Dagoon, P. F. Baldi, S. Abdelkarim, J. Liu, M. E. Riazi, M. Andrade, A. Azzam, P. R. Esfahani, S. Chang, and A. Browne. Annotation of surgical tool depth in vitreoretinal surgical videos: Agreement and performance between vitreoretinal surgeons vs. non-surgeon graders. *Investigative Ophthalmology & Visual Science*, 65(7):3762–3762, 2024.

[95] A. Dhar. *Object Tracking for Autonomous Driving Systems*. PhD thesis, Master's thesis, EECS Department, University of California, Berkeley, 2020.

[96] L. Ding, Y. Wang, K. Yuan, M. Jiang, P. Wang, H. Huang, and Z. J. Wang. Towards Universal Physical Attacks on Single Object Tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1236–1245, 2021.

[97] C. DiPalma, N. Wang, T. Sato, and Q. A. Chen. Security of Camera-based Perception for Autonomous Driving under Adversarial Attack. In *2021 IEEE Security and Privacy Workshops (SPW)*, pages 243–243. IEEE, 2021.

[98] S. Dominguez, A. Ali, G. Garcia, and P. Martinet. Comparison of Lateral Controllers for Autonomous Vehicle: Experimental Results. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1418–1423. IEEE, 2016.

[99] T. Dreossi, D. J. Fremont, S. Ghosh, E. Kim, H. Ravanbakhsh, M. Vazquez-Chanlatte, and S. A. Seshia. VerifAI: A toolkit for the formal design and analysis of artificial intelligence-based systems. In *31st International Conference on Computer Aided Verification (CAV)*, July 2019.

[100] E. DREYFUSS. Security News This Week: A Whole New Way to Confuse Self-Driving Cars. https://www.wired.com/story/security-news-august-5-2017/, 2017.

[101] X. Du, M. H. Ang, S. Karaman, and D. Rus. A General Pipeline for 3D Detection of Vehicles. In *ICRA 2018*, pages 3194–3200. IEEE, 2018.

[102] X. Du, M. H. Ang, and D. Rus. Car Detection for Autonomous Vehicle: LIDAR and Vision Fusion Approach Through Deep Learning Framework. In *IROS*, 2017.

[103] Y. Du, Z. Zhao, Y. Song, Y. Zhao, F. Su, T. Gong, and H. Meng. StrongSORT: Make DeepSORT Great Again. *IEEE Transactions on Multimedia*, 2023.

[104] R. Duan, X. Mao, A. K. Qin, Y. Chen, S. Ye, Y. He, and Y. Yang. Adversarial Laser Beam: Effective Physical-World Attack to DNNs in a Blink. In *CVPR*, 2021.

[105] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy. A Study of the Effect of JPG Compression on Adversarial Images. *arXiv*, 2016.

[106] C. Elliott. Do Safety Features In Cars Actually Reduce Car Accidents? https://www.forbes.com/advisor/car-insurance/vehicle-safety-features-accidents/, 2024.

[107] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast Object Detection in 3D Point Clouds using Efficient Convolutional Neural Networks. In *ICRA*, 2017.

[108] I. Evtimov, K. Eykholt, E. Fernandes, and B. Li. Physical Adversarial Examples Against Deep Neural Networks. https://bair.berkeley.edu/blog/2017/12/30/yolo-attack/, 2017.

[109] K. Eykholt. *Designing and Evaluating Physical Adversarial Attacks and Defenses for Machine Learning Algorithms*. PhD thesis, 2019.

[110] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, F. Tramer, A. Prakash, T. Kohno, and D. Song. Physical Adversarial Examples for Object Detectors. In *WOOT*, 2018.

[111] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust Physical-World Attacks on Deep Learning Visual Classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.

[112] Y. Fang, B. Liao, X. Wang, J. Fang, J. Qi, R. Wu, J. Niu, and W. Liu. You Only Look at One Sequence: Rethinking Transformer in Vision through Object Detection. *NeurIPS*, 2021.

[113] Y. Feng, B. Wu, Y. Fan, L. Liu, Z. Li, and S. Xia. CG-ATTACK: Modeling the Conditional Distribution of Adversarial Perturbations to Boost Black-Box Attack. 2020.

[114] J. Fingas. You can confuse self-driving cars by altering street signs. https://www.engadget.com/2017-08-06-altered-street-signs-confuse-self-driving-cars.html, 2017.

[115] D. Frossard and R. Urtasun. End-to-end Learning of Multi-sensor 3D Tracking by Detection. In *ICRA 2018*, pages 635–642. IEEE, 2018.

[116] J. Frost. Hypothesis testing: An intuitive guide for making data driven decisions. 2020.

[117] J. Frost. Null Hypothesis: Definition, Rejecting & Examples. https://statisticsbyjim.com/hypothesis-testing/null-hypothesis/, 2022.

[118] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021.

[119] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision Meets Robotics: The KiTTi Dataset. *IJRR*, 2013.

[120] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[121] J. M. GITLIN. Hacking street signs with stickers could confuse self-driving cars. https://arstechnica.com/cars/2017/09/hacking-street-signs-with-stickers-could-confuse-self-driving-cars/?amp=1, 2017.

[122] I. Gog, S. Kalra, P. Schafhalter, M. A. Wright, J. E. Gonzalez, and I. Stoica. Pylot: A modular platform for exploring latency-accuracy tradeoffs in autonomous vehicles. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8806–8813. IEEE, 2021.

[123] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015.

[124] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015.

[125] M. Greiffenhagen, D. Comaniciu, H. Niemann, and V. Ramesh. Design, Analysis, and Engineering of Video Monitoring Systems: An Approach and a Case Study. *Proceedings of the IEEE*, 89(10):1498–1517, 2001.

[126] M. Greiffenhagen, V. Ramesh, D. Comaniciu, and H. Niemann. Statistical modeling and performance characterization of a real-time dual camera surveillance system. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 2, pages 335–342. IEEE, 2000.

[127] M. Greiffenhagen, V. Ramesh, and H. Niemann. The systematic design and analysis cycle of a vision system: A case study in video surveillance. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–II. IEEE, 2001.

[128] P. Guo, H. Kim, N. Virani, J. Xu, M. Zhu, and P. Liu. RoboADS: Anomaly Detection against Sensor and Actuator Misbehaviors in Mobile Robots. In *DSN*, pages 574–585. IEEE, 2018.

[129] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun. Deep Learning for 3D Point Clouds: A Survey, 2019.

[130] M. Haque, A. Chauhan, C. Liu, and W. Yang. Ilfo: Adversarial attack on adaptive neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14264–14273, 2020.

[131] R. M. Haralick. Performance characterization in image analysis: thinning, a case in point. *Pattern Recognition Letters*, 13(1):5–12, 1992.

[132] C. He, X. Ma, B. B. Zhu, Y. Zeng, H. Hu, X. Bai, H. Jin, and D. Zhang. DorPatch: Distributed and Occlusion-Robust Adversarial Patch to Evade Certifiable Defenses. In *NDSS*, 2024.

[133] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[134] W. He, J. Wei, X. Chen, N. Carlini, and D. Song. Adversarial Example Defense: Ensembles of Weak Defenses are not Strong. In *USENIX WOOT*, 2017.

[135] D. Heaven. Why deep-learning AIs are so easy to fool. https://www.nature.com/articles/d41586-019-03013-5, 2019.

[136] D. Hendrycks, K. Lee, and M. Mazeika. Using Pre-Training can Improve Model Robustness and Uncertainty. *ICML*, 2019.

[137] S. Hinderer. What Is Traffic-Sign Recognition? https://www.kbb.com/what-is/traffic-sign-recognition/, 2023.

[138] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun. Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing. In *2007 American control conference*, pages 2296–2301. IEEE, 2007.

[139] S. Hong, Y. Kaya, I.-V. Modoranu, and T. Dumitraş. A panda? no, it's a sloth: Slowdown attacks on adaptive multi-exit neural network inference. *arXiv preprint arXiv:2010.02432*, 2020.

[140] H. Hsu and P. A. Lachenbruch. Paired t test. *Wiley StatsRef: statistics reference online*, 2014.

[141] B. Huang and H. Ling. SPAA: Stealthy Projector-based Adversarial Attacks on Deep Image Classifiers. *ArXiv*, 2020.

[142] M. D. Ibarra, J. K. To, J. Liu, S. Abdelkarim, A. Herekar, B. D. Kuppermann, P. Baldi, and A. Browne. Automated detection of the spatial location of vitreoretinal instruments from retinal images using deep learning methods. *Investigative Ophthalmology & Visual Science*, 63(7):210–F0057, 2022.

[143] D. Ingram. Waymo will launch paid robotaxi service in Los Angeles on Wednesday. https://www.nbcnews.com/tech/innovation/waymo-will-launch-paid-robotaxi-service-los-angeles-wednesday-rcna147101, 2024.

[144] R. Ivanov, M. Pajic, and I. Lee. Attack-resilient Sensor Fusion. In *DATE*, pages 1–6. IEEE, 2014.

[145] R. C. Jain and T. O. Binford. Ignorance, myopia, and naivete in computer vision systems. *CVGIP: Image Understanding*, 53(1):112–117, 1991.

[146] V. L. Jessie Smith. Scaling Simulation. https://aurora.tech/blog/scaling-simulation/, 2021.

[147] Q. Ji and R. M. Haralick. Error propagation for computer vision performance characterization. In *International Conference on Imaging Science, Systems, and Technology, Las Vegas*, 1999.

[148] W. Jia, Z. Lu, H. Zhang, Z. Liu, J. Wang, and G. Qu. Fooling the Eyes of Autonomous Vehicles: Robust Physical Adversarial Examples Against Traffic Sign Recognition Systems. In *NDSS*, 2022.

[149] Y. Jia, Y. Lu, J. Shen, Q. A. Chen, H. Chen, Z. Zhong, and T. Wei. Fooling Detection Alone is Not Enough: Adversarial Attack Against Multiple Object Tracking. In *ICLR*, 2019.

[150] Y. J. Jia, Y. Lu, J. Shen, Q. A. Chen, H. Chen, Z. Zhong, and T. W. Wei. Fooling Detection Alone is Not Enough: Adversarial Attack against Multiple Object Tracking. In *International Conference on Learning Representations (ICLR'20)*, 2020.

[151] L. Jiang, X. Ma, S. Chen, J. Bailey, and Y.-G. Jiang. Black-box Adversarial Attacks on Video Recognition Models. In *ACM International Conference on Multimedia*, 2019.

[152] Z. Jin, X. Ji, Y. Cheng, B. Yang, C. Yan, and W. Xu. PLA-LiDAR: Physical Laser Attacks against LiDAR-based 3D Object Detection in Autonomous Vehicle. In *IEEE S&P*, 2023.

[153] P. Jing, Q. Tang, Y. Du, L. Xue, X. Luo, T. Wang, S. Nie, and S. Wu. Too Good to Be Safe: Tricking Lane Detection in Autonomous Driving with Crafted Perturbations. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3237–3254, 2021.

[154] Y. Jo, S. Yang, and S. Joo Kim. Investigating Loss Functions for Extreme Super-Resolution. In *CVPR Workshops*, pages 424–425, 2020.

[155] G. Jocher. ultralytics/yolov5. https://github.com/ultralytics/yolov5, 2020.

[156] G. Jocher. YOLOv5. https://github.com/ultralytics/yolov5, 2022.

[157] G. Jocher, A. Chaurasia, and J. Qiu. Ultralytics yolo (version 8.0.0) [computer software]. https://github.com/ultralytics/ultralytics, 2023.

[158] M. Kane. Tesla Sold 2 Million Electric Cars: First Automaker To Reach Milestone, 2021.

[159] H. Kato, Y. Ushiku, and T. Harada. Neural 3D Mesh Renderer. In *CVPR*, June 2018.

[160] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi. Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, pages 287–296. IEEE, 2018.

[161] S. Köhler, G. Lovisotto, S. Birnbach, R. Baker, and I. Martinovic. They See Me Rollin': Inherent Vulnerability of the Rolling Shutter in CMOS Image Sensors. In *ACSAC*, 2021.

[162] A. Krok. It is surprisingly easy to bamboozle a self-driving car. https://www.cnet.com/roadshow/news/it-is-surprisingly-easy-to-bamboozle-a-self-driving-car/, 2017.

[163] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. Joint 3D Proposal Generation and Object Detection from View Aggregation. In *IROS*, pages 1–8, 2018.

[164] Y. K. Kwag, M. S. Choi, C. H. Jung, and K. Y. Hwang. Collision Avoidance Radar for UAV. In *2006 CIE International Conference on Radar*, pages 1–4. IEEE, 2006.

[165] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast Encoders for Object Detection from Point Clouds. In *CVPR*, pages 12697–12705, 2019.

[166] Y. Law. 5 TOP SELF-DRIVING CAR MANUFACTURERS. https://yoshalawfirm.com/blog/5-top-self-driving-car-manufacturers/.

[167] D. N. Lawley. A generalization of fisher's z test. *Biometrika*, 30(1/2):180–187, 1938.

[168] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified Robustness to Adversarial Examples with Differential Privacy. In *IEEE S&P*.

[169] K. Lee, Z. Chen, X. Yan, R. Urtasun, and E. Yumer. ShapeAdv: Generating Shape-Aware Adversarial 3D Point Clouds. *ArXiv*, 2020.

[170] F. Leon and M. Gavrilescu. A Review of Tracking and Trajectory Prediction Methods for Autonomous Driving. *Mathematics*, 9(6), 2021.

[171] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4282–4291, 2019.

[172] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High Performance Visual Tracking With Siamese Region Proposal Network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8971–8980, 2018.

[173] J. Li, F. Schmidt, and Z. Kolter. Adversarial camera stickers: A physical camera-based attack on deep learning systems. In *International Conference on Machine Learning*, pages 3896–3904. PMLR, 2019.

[174] L. Li, X. Qi, T. Xie, and B. Li. SoK: Certified Robustness for Deep Neural Networks. *arXiv preprint arXiv:2009.04131*, 2020.

[175] L. Li, T. Xie, and B. Li. SoK: Certified Robustness for Deep Neural Networks. In *2023 IEEE symposium on security and privacy (SP)*, pages 1289–1310. IEEE, 2023.

[176] S. Li, X. Xu, L. Nie, and T.-S. Chua. Laplacian-Steered Neural Style Transfer. In *ICMR*, pages 1716–1724, 2017.

[177] Y. Li, S. Xie, X. Chen, P. Dollar, K. He, and R. Girshick. Benchmarking Detection Transfer Learning with Vision Transformers. *arXiv preprint arXiv:2111.11429*, 2021.

[178] H. Liang, R. Jiao, T. Sato, J. Shen, Q. A. Chen, and Q. Zhu. End-to-End Analysis of Adversarial Attacks to Automated Lane Centering Systems. In *AutoSec Workshop at NDSS*, 2021.

[179] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun. Multi-Task Multi-Sensor Fusion for 3D Object Detection. In *CVPR*, 2019.

[180] M. Liang, B. Yang, S. Wang, and R. Urtasun. Deep Continuous Fusion for Multi-Sensor 3D Object Detection. In *ECCV*, 2018.

[181] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

[182] H. Liu, Y. Wu, Z. Yu, Y. Vorobeychik, and N. Zhang. SlowLiDAR: Increasing the Latency of LiDAR-Based Detection Using Adversarial Examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5146–5155, 2023.

[183] J. Liu, A. Ghosh, D. Smith, P. Baldi, and D. Whiteson. Geometry-aware autoregressive models for calorimeter shower simulations. *arXiv preprint arXiv:2212.08233*, 2022.

[184] J. Liu, A. Ghosh, D. Smith, P. Baldi, and D. Whiteson. Generalizing to new geometries with geometry-aware autoregressive models (gaams) for fast calorimeter simulation. *Journal of Instrumentation*, 18(11):P11003, 2023.

[185] J. Liu, J. Ott, J. Collado, B. Jargowsky, W. Wu, J. Bian, and P. Baldi. Deep-learning-based kinematic reconstruction for dune (2020). *arXiv preprint arXiv:2012.0618*, 2012.

[186] G. Lovisotto, H. Turner, I. Sluganovic, M. Strohmeier, and I. Martinovic. SLAP: Improving Physical Adversarial Examples with Short-Lived Adversarial Perturbations. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1865–1882, 2021.

[187] J. Lu, H. Sibai, and E. Fabry. Adversarial Examples that Fool Detectors. *arXiv preprint arXiv:1712.02494*, 2017.

[188] J. Lu, H. Sibai, E. Fabry, and D. Forsyth. No Need to Worry about Adversarial Examples in Object Detection in Autonomous Vehicles. *arXiv preprint arXiv:1707.03501*, 2017.

[189] X. Lu, B. Li, Y. Yue, Q. Li, and J. Yan. Grid R-CNN. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7363–7372, 2019.

[190] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim. Multiple object tracking: A literature review. *Artificial intelligence*, 293:103448, 2021.

[191] Y. Luo, N. Wang, B. Yu, S. Liu, and Q. A. Chen. Infrastructure-aided defense for autonomous driving systems: Opportunities and challenges. In *NDSS Workshop on Automotive and Autonomous Vehicle Security (AutoSec)*, 2022.

[192] C. Ma, N. Wang, Q. A. Chen, and C. Shen. WIP: Towards the Practicality of the Adversarial Attack on Object Tracking in Autonomous Driving. In *ISOC Symposium on Vehicle Security and Privacy (VehicleSec)*, 2023.

[193] C. Ma, N. Wang, Q. A. Chen, and C. Shen. SlowTrack: Increasing the Latency of Camera-based Perception in Autonomous Driving Using Adversarial Examples. In *AAAI*, 2024.

[194] C. Ma, N. Wang, Q. A. Chen, and C. Shen. SlowTrack: Increasing the Latency of Camera-Based Perception in Autonomous Driving Using Adversarial Examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4062–4070, 2024.

[195] C. Ma, N. Wang, Z. Zhao, Q. Wang, Q. A. Chen, and C. Shen. Controlloc: Physical-world hijacking attack on visual perception in autonomous driving. *arXiv preprint arXiv:2406.05810*, 2024.

[196] X. Ma, Z. Wang, H. Li, P. Zhang, W. Ouyang, and X. Fan. Accurate Monocular 3D Object Detection via Color-Embedded 3D Reconstruction for Autonomous Driving. In *CVPR*, pages 6851–6860, 2019.

[197] T. W. MacFarland, J. M. Yates, T. W. MacFarland, and J. M. Yates. Kruskal–wallis h-test for oneway analysis of variance (anova) by ranks. *Introduction to nonparametric statistics for the biological sciences using R*, pages 177–211, 2016.

[198] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *stat*, 1050:9, 2017.

[199] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[200] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*, 2018.

[201] A. V. Malawade, T. Mortlock, and M. A. Al Faruque. HydraFusion: Context-Aware Selective Sensor Fusion for Robust and Efficient Autonomous Vehicle Perception. In *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*, pages 68–79. IEEE, 2022.

[202] Y. Man, M. Li, and R. Gerdes. GhostImage: Perception Domain Attacks against Vision-based Object Classification Systems. *arXiv preprint arXiv:2001.07792*, 2020.

[203] Y. Man, R. Muller, M. Li, Z. B. Celik, and R. Gerdes. That Person Moves Like A Car: Misclassification Attack Detection for Autonomous Systems Using Spatiotemporal Consistency. In *USENIX Security*, 2023.

[204] marklines.com. USA-Automotive Sales Volume. https://www.marklines.com/en/statistics/flash_sales/automotive-sales-in-usa-by-month-2023, 2023.

[205] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, pages 922–928, 2015.

[206] D. Meng and H. Chen. MagNet: a Two-Pronged Defense against Adversarial Examples. In *ACM CCS*, pages 135–147, 2017.

[207] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A Benchmark for Multi-Object Tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[208] M. Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[209] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. DeepFool: a Simple and Accurate Method to Fool Deep Neural Networks. In *CVPR*, pages 2574–2582, 2016.

[210] D. Z. MORRIS. Researchers Show How Simple Stickers Could Trick Self-Driving Cars. https://fortune.com/2017/09/02/researchers-show-how-simple-stickers-could-trick-self-driving-cars/, 2017.

[211] R. Muller, Y. Man, Z. B. Celik, M. Li, and R. Gerdes. Physical Hijacking Attacks against Object Trackers. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2309–2322, 2022.

[212] R. Muller, Y. Man, M. Li, R. Gerdes, J. Petit, Z. B. Celik, and O. D. OD. Vogues: Validation of object guise using estimated components. 2024.

[213] B. Nassi, Y. Mirsky, D. Nassi, R. Ben-Netanel, O. Drokin, and Y. Elovici. Phantom of the ADAS: Securing Advanced Driver-Assistance Systems from Split-Second Phantom Attacks. In *ACM CCS*, 2020.

[214] R. Nese, N. K. Pandey, and S. Thimmalapura. A systematic approach of vehicle plant model development for vehicle virtual testing & calibration. In *2015 IEEE International Transportation Electrification Conference (ITEC)*, pages 1–10. IEEE, 2015.

[215] U. S. D. of Transportation. Speed Limit Basics. https://highways.dot.gov/safety/speed-management/speed-limit-basics, 2017.

[216] S. on Security. Confusing Self-Driving Cars by Altering Road Signs. https://www.schneier.com/blog/archives/2017/08/confusing_self-.html, 2017.

[217] OpenPilot. OpenPilot. https://github.com/commaai/openpilot, 2022.

[218] R. Pannell. The Importance of Identifying the Right Sample Size for Business Improvement. https://leanscape.io/the-importance-of-identifying-the-right-sample-size-for-business-improvement, 2023.

[219] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The Limitations of Deep Learning in Adversarial Settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[220] N. Papernot, P. D. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The Limitations of Deep Learning in Adversarial Settings. In *Euro S&P*, 2016.

[221] K. Pei, Y. Cao, J. Yang, and S. Jana. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In *SOSP*, 2017.

[222] J. Petit and S. E. Shladover. Potential Cyberattacks on Automated Vehicles. *IEEE ITS*, 16(2):546–556, 2014.

[223] J. Philion, A. Kar, and S. Fidler. Learning to Evaluate Perception Models using Planner-Centric Metrics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14055–14064, 2020.

[224] J. Phillips, J. Martinez, I. A. Bârsan, S. Casas, A. Sadat, and R. Urtasun. Deep multi-task learning for joint localization, perception, and prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4679–4689, 2021.

[225] PYMNTS. Amazon-Owned Zoox Begins Operating Driverless Robotaxi in Las Vegas. https://www.pymnts.com/news/ridesharing/2023/amazon-owned-zoox-begins-operating-driverless-robotaxi-las-vegas/, 2023.

[226] PyTorch. Models and pre-trained weights. https://pytorch.org/vision/main/models.html.

[227] H. Qiu, C. Xiao, L. Yang, X. Yan, H. Lee, and B. Li. SemanticAdv: Generating Adversarial Examples via Attribute-conditioned Image Editing. In *ECCV*, pages 19–37. Springer, 2020.

[228] R. Quinonez, J. Giraldo, L. Salazar, and E. Bauman. SAVIOR: Securing Autonomous Vehicles with Robust Physical Invariants. In *USENIX Security*, 2020.

[229] V. Ramesh, R. Haralick, A. Bedekar, X. Liu, D. Nadadur, K. Thornton, and X. Zhang. Computer vision performance characterization. *RADIUS: Image Understanding for Imagery Intelligence*, pages 241–282, 1997.

[230] J. Redmon and A. Farhadi. YOLO9000: Better, Faster, Stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[231] J. Redmon and A. Farhadi. YOLOv3: An Incremental Improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[232] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in neural information processing systems*, 28, 2015.

[233] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[234] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta, et al. LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving. In *2020 IEEE 23rd International conference on intelligent transportation systems (ITSC)*, pages 1–6. IEEE, 2020.

[235] A. Ross, V. L. Willson, A. Ross, and V. L. Willson. One-sample t-test. *Basic and Advanced Statistical Tests: Writing Results Sections and Creating Tables and Figures*, pages 9–12, 2017.

[236] N. D. Rybel. How to obtain standard deviation of replicate experiments with repeated measurements? 2017.

[237] T. Sato, S. H. V. Bhupathiraju, M. Clifford, T. Sugawara, Q. A. Chen, and S. Rampazzi. Invisible Reflections: Leveraging Infrared Laser Reflections to Target Traffic Sign Perception. In *NDSS*, 2024.

[238] T. Sato, J. Shen, N. Wang, Y. Jia, X. Lin, and Q. A. Chen. Dirty Road Can Attack: Security of Deep Learning based Automated Lane Centering under Physical-World Attack. In *USENIX Security*, 2021.

[239] T. Sato, J. Shen, N. Wang, Y. Jia, X. Lin, and Q. A. Chen. Dirty Road Can Attack: Security of Deep Learning based Automated LaneCentering under Physical-World Attack. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3309–3326, 2021.

[240] T. Sato, J. Shen, N. Wang, Y. J. Jia, X. Lin, and Q. A. Chen. Hold Tight and Never Let Go: Security of Deep Learning based Automated Lane Centering under Physical-World Attack. *ArXiv*, 2020.

[241] T. Sato, J. Shen, N. Wang, Y. J. Jia, X. Lin, and Q. A. Chen. Security of deep learning based lane keeping system under physical-world adversarial attack. *arXiv preprint arXiv:2003.01782*, 2020.

[242] T. Sato, J. Shen, N. Wang, Y. J. Jia, X. Lin, and Q. A. Chen. Deployability Improvement, Stealthiness User Study, and Safety Impact Assessment on Real Vehicle for Dirty Road Patch Attack. In *AutoSec Workshop at NDSS*, 2021.

[243] T. Sato, J. Yue, N. Chen, N. Wang, and Q. A. Chen. Intriguing Properties of Diffusion Models: A Large-Scale Dataset for Evaluating Natural Attack Capability in Text-to-Image Generative Models. In *CVPR*, 2024.

[244] A. Sayles, A. Hooda, M. Gupta, R. Chatterjee, and E. Fernandes. Invisible Perturbations: Physical Adversarial Examples Exploiting the Rolling Shutter Effect. In *CVPR*, 2021.

[245] R. Schumacker, S. Tomek, R. Schumacker, and S. Tomek. F-test. *Understanding statistics using R*, pages 197–207, 2013.

[246] D. Seita. BDD100k: A large-scale diverse driving video database. *The Berkeley Artificial Intelligence Research Blog. Version*, 511:41, 2018.

[247] S. A. Seshia, D. Sadigh, and S. S. Sastry. Towards Verified Artificial Intelligence. *Communications of the ACM*, 65(7):46–55, 2022.

[248] A. Shafahi, M. Najibi, M. A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein. Adversarial Training for Free! In *NIPS*, pages 3358–3369, 2019.

[249] A. Shapira, A. Zolfi, L. Demetrio, B. Biggio, and A. Shabtai. Phantom Sponges: Exploiting Non-Maximum Suppression to Attack Deep Object Detectors. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4571–4580, 2023.

[250] Y. Sharma and P.-Y. Chen. Bypassing Feature Squeezing by Increasing Adversary Strength. *ICLR Workshop*, 2018.

[251] J. P. Shaver. What statistical significance testing is, and what it is not. *The Journal of Experimental Education*, 61(4):293–316, 1993.

[252] J. Shen, N. Wang, Z. Wan, Y. Luo, T. Sato, Z. Hu, X. Zhang, S. Guo, Z. Zhong, K. Li, et al. SoK: On the Semantic AI Security in Autonomous Driving. *arXiv preprint arXiv:2203.05314*, 2022.

[253] J. Shen, J. Y. Won, and Q. A. Chen. Drift with Devil: Security of Multi-Sensor Fusion based Localization in High-Level Autonomous Driving under GPS Spoofing. In *Usenix Security*, 2020.

[254] H. Shin, D. Kim, Y. Kwon, and Y. Kim. Illusion and Dazzle: Adversarial Optical Channel Exploits Against Lidars for Automotive Applications. In *CHES*, 2017.

[255] I. Shumailov, Y. Zhao, D. Bates, N. Papernot, R. Mullins, and R. Anderson. Sponge examples: Energy-latency attacks on neural networks. In *2021 IEEE European symposium on security and privacy (EuroS&P)*, pages 212–231. IEEE, 2021.

[256] D. Silver. Adversarial Traffic Signs. https://medium.com/self-driving-cars/adversarial-traffic-signs-fd16b7171906, 2017.

[257] K. Smallwood. WHEN DOES A SPEED LIMIT COME INTO EFFECT? https://www.todayifoundout.com/index.php/2015/03/exactly-speed-limit-come-effect/, 2015.

[258] L. SMILEY. The Legal Saga of Uber's Fatal Self-Driving Car Crash Is Over. https://www.wired.com/story/ubers-fatal-self-driving-car-crash-saga-over-operator-avoids-prison/, 2023.

[259] L. St, S. Wold, et al. Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4):259–272, 1989.

[260] statista.com. Leading car brands in the United States in 2023, based on vehicle sales. https://www.statista.com/statistics/264362/leading-car-brands-in-the-us-based-on-vehicle-sales/, 2023.

[261] J. Sun, Y. Cao, Q. A. Chen, and Z. M. Mao. Towards Robust LiDAR-based Perception in Autonomous Driving: General Black-box Adversarial Sensor Attack and Countermeasures. In *Usenix Security*, 2020.

[262] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing Properties of Neural Networks. In *ICLR*, 2014.

[263] K. Tang, J. Shen, and Q. A. Chen. Fooling Perception via Location: A Case of Region-of-Interest Attacks on Traffic Light Detection in Autonomous Driving. In *AutoSec Workshop at NDSS*, 2021.

[264] Tesla. Future of Driving. https://www.tesla.com/autopilot.

[265] Tesla. Future of Driving. https://www.tesla.com/autopilot, 2022.

[266] N. A. Thacker, A. F. Clark, J. L. Barron, J. R. Beveridge, P. Courtney, W. R. Crum, V. Ramesh, and C. Clark. Performance characterization in computer vision: A guide to best practices. *Computer vision and image understanding*, 109(3):305–334, 2008.

[267] S. Thys, W. Van Ranst, and T. Goedemé. Fooling Automated Surveillance Cameras: Adversarial Patches to Attack Person Detection. In *CVPR Workshops*, pages 0–0, 2019.

[268] Y. Tian, K. Pei, S. Jana, and B. Ray. Deeptest: Automated Testing of Deep-Neural-Network-Driven Autonomous Cars. In *ICSE*, 2018.

[269] Z. Tian, C. Shen, H. Chen, and T. He. FCOS: A Simple and Strong Anchor-Free Object Detector. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):1922–1933, 2020.

[270] S. Topan, K. Leung, Y. Chen, P. Tupekar, E. Schmerling, J. Nilsson, M. Cox, and M. Pavone. Interaction-dynamics-aware perception zones for obstacle detection safety evaluation. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 1201–1210. IEEE, 2022.

[271] T. Tsai, K. Yang, T.-Y. Ho, and Y. Jin. Robust Adversarial Objects against Deep Learning Models. In *AAAI*, 2020.

[272] Y. Tu, Z. Lin, I. Lee, and X. Hei. Injected and Delivered: Fabricating Implicit Control over Actuation Systems by Spoofing Inertial Sensors. In *USENIX Security*, pages 1545–1562, 2018.

[273] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[274] M. M. Wagner-Menghin. Binomial test. *Encyclopedia of statistics in behavioral science*, 2005.

[275] Z. Wan, J. Shen, J. Chuang, X. Xia, J. Garcia, J. Ma, and Q. A. Chen. Too Afraid to Drive: Systematic Discovery of Semantic DoS Vulnerability in Autonomous Driving Planning under Physical-World Attacks. In *Network and Distributed System Security (NDSS) Symposium, 2022*, April 2022.

[276] Z. Wan, J. Shen, J. Chuang, X. Xia, J. Garcia, J. Ma, and Q. A. Chen. Too Afraid to Drive: Systematic Discovery of Semantic DoS Vulnerability in Autonomous Driving Planning under Physical-World Attacks. In *NDSS*, 2022.

[277] D. Wang, T. Jiang, J. Sun, W. Zhou, Z. Gong, X. Zhang, W. Yao, and X. Chen. Fca: Learning a 3d full-coverage vehicle camouflage for multi-view physical adversarial attack. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2414–2422, 2022.

[278] D. Wang, C. Li, S. Wen, Q.-L. Han, S. Nepal, X. Zhang, and Y. Xiang. Daedalus: Breaking Nonmaximum Suppression in Object Detection via Adversarial Examples. *IEEE Transactions on Cybernetics*, 52(8):7427–7440, 2021.

[279] D. Z. Wang and I. Posner. Voting for Voting in Online Point Cloud Object Detection. In *Robotics: Science and Systems*, 2015.

[280] L. Wang, J. Tang, and Q. Liao. A Study on Radar Target Detection based on Deep Neural Networks. *IEEE Sensors Letters*, pages 1–4, 2019.

[281] N. Wang, S. Ji, and T. Wang. Integration of Static and Dynamic Code Stylometry Analysis for Programmer De-anonymization. In *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*, pages 74–84, 2018.

[282] N. Wang, Y. Luo, T. Sato, K. Xu, and Q. A. Chen. Poster: On the System-Level Effectiveness of Physical Object-Hiding Adversarial Attack in Autonomous Driving. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 3479–3481, 2022.

[283] N. Wang, Y. Luo, T. Sato, K. Xu, and Q. A. Chen. Does Physical Adversarial Example Really Matter to Autonomous Driving? Towards System-Level Effect of Adversarial Object Evasion Attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4412–4423, October 2023.

[284] N. Wang, S. Xie, T. Sato, Y. Luo, K. Xu, and Q. A. Chen. Revisiting Physical-World Adversarial Attack on Traffic Sign Recognition: A Commercial Systems Perspective. In *2025 The Network and Distributed System Security Symposium (NDSS)*, 2025.

[285] W. Wang, Y. Yao, X. Liu, X. Li, P. Hao, and T. Zhu. I Can See the Light: Attacks on Autonomous Vehicles Using Invisible Lights. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1930–1944, 2021.

[286] Waymo. The World's Most Experienced Driver. https://waymo.com/intl/es/.

[287] Waymo. Waymo Safety Report. https://storage.googleapis.com/waymo-uploads/files /documents/safety/2021-12-waymo-safety-report.pdf, 2021.

[288] Waymo. WAYMO ONE: The future of transportation is here. https://waymo.com/w aymo-one/, 2024.

[289] H. Wen, S. Chang, L. Zhou, W. Liu, and H. Zhu. OptiCloak: Blinding Vision-Based Autonomous Driving Systems Through Adversarial Optical Projection. *IEEE Internet of Things Journal*, 2024.

[290] WIKI. Traffic-sign recognition. https://en.wikipedia.org/wiki/Traffic-sign_recogniti on, 2023.

[291] E. Wong, L. Rice, and J. Z. Kolter. Fast is Better than Free: Revisiting Adversarial Training. *ICLR*, 2020.

[292] R. F. Woolson. Wilcoxon signed-rank test. *Encyclopedia of Biostatistics*, 8, 2005.

[293] Z. Wu, S.-N. Lim, L. S. Davis, and T. Goldstein. Making an invisibility cloak: Real world adversarial attacks on object detectors. In *European Conference on Computer Vision*, pages 1–17. Springer, 2020.

[294] C. Xiang, A. N. Bhagoji, V. Sehwag, and P. Mittal. PatchGuard: A Provably Robust Defense against Adversarial Patches via Small Receptive Fields and Masking. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2237–2254, 2021.

[295] C. Xiang and P. Mittal. DetectorGuard: Provably Securing Object Detectors against Localized Patch Hiding Attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3177–3196, 2021.

[296] C. Xiang, C. R. Qi, and B. Li. Generating 3D Adversarial Point Clouds. In *CVPR*, pages 9136–9144, 2019.

[297] C. Xiang, A. Valtchanov, S. Mahloujifar, and P. Mittal. ObjectSeeker: Certifiably Robust Object Detection against Patch Hiding Attacks via Patch-agnostic Masking. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1329–1347. IEEE, 2023.

[298] C. Xiao, R. Deng, B. Li, F. Yu, M. Liu, and D. Song. Characterizing Adversarial Examples based on Spatial Consistency Information for Semantic Segmentation. In *ECCV*, pages 217–234, 2018.

[299] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song. Generating Adversarial Examples with Adversarial Networks. *ArXiv*, 2018.

[300] C. Xiao, X. Pan, W. He, J. Peng, M. Sun, J. Yi, M. Liu, B. Li, and D. Song. Characterizing Attacks on Deep Reinforcement Learning. *arXiv*, 2019.

[301] C. Xiao, D. Yang, B. Li, J. Deng, and M. Liu. MeshAdv: Adversarial Meshes for Visual Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6898–6907, 2019.

[302] C. Xiao, J.-Y. Zhu, B. Li, W. He, M. Liu, and D. Song. Spatially Transformed Adversarial Examples. *ICLR*, 2018.

[303] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille. Adversarial Examples for Semantic Segmentation and Object Detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1369–1378, 2017.

[304] D. Xu, D. Anguelov, and A. Jain. PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation. In *CVPR*, pages 244–253, 2018.

[305] K. Xu, G. Zhang, S. Liu, Q. Fan, M. Sun, H. Chen, P.-Y. Chen, Y. Wang, and X. Lin. Adversarial T-Shirt! Evading Person Detectors in a Physical World. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 665–681. Springer, 2020.

[306] W. Xu, D. Evans, and Y. Qi. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *NDSS*, 2018.

[307] W. Xu, D. Evans, and Y. Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *NDSS*, 2018.

[308] W. Xu, C. Yan, W. Jia, X. Ji, and J. Liu. Analyzing and Enhancing the Security of Ultrasonic Sensors for Autonomous Vehicles. *IEEE Internet of Things Journal*, 5(6):5015–5029, 2018.

[309] M. Xue, C. Yuan, C. He, J. Wang, and W. Liu. NaturalAE: Natural and Robust Physical Adversarial Examples for Object Detectors. *Journal of Information Security and Applications*, 57:102694, 2021.

[310] C. Yan, W. Xu, and J. Liu. Can You Trust Autonomous Vehicles: Contactless Attacks against Sensors of Self-driving Vehicle. *DEF CON*, 24(8):109, 2016.

[311] X. Yan, X. Chen, Y. Jiang, S.-T. Xia, Y. Zhao, and F. Zheng. Hijacking Tracker: A Powerful Adversarial Attack on Visual Tracking. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2897–2901. IEEE, 2020.

[312] B. Yang, W. Luo, and R. Urtasun. Pixor: Real-time 3d object detection from point clouds. In *CVPR 2018*, pages 7652–7660, 2018.

[313] G. Yang, T. Duan, E. Hu, H. Salman, I. Razenshteyn, and J. Li. Randomized Smoothing of All Shapes and Sizes. *ICML*, 2020.

[314] J. Young. Pedestrian crash avoidance systems cut crashes — but not in the dark. https://www.iihs.org/news/detail/pedestrian-crash-avoidance-systems-cut-crashes--but-not-in-the-dark, 2022.

[315] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell. BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020.

[316] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. *IEEE Access*, 8:58443–58469, 2020.

[317] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. *IEEE access*, 8:58443–58469, 2020.

[318] H. Zhang and J. Wang. Towards Adversarially Robust Object Detection. In *ICCV*, pages 421–430, 2019.

[319] P. Zhang. Taiwanese star Jimmy Lin injured in Tesla crash. https://cnevpost.com/2022/07/22/taiwanese-star-jimmy-lin-injured-in-tesla-crash-report-says/, 2022.

[320] Q.-s. Zhang and S.-C. Zhu. Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.

[321] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*, 2018.

[322] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li. Bridging the Gap Between Anchor-based and Anchor-free Detection via Adaptive Training Sample Selection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9759–9768, 2020.

[323] X. Zhang, N. Wang, H. Shen, S. Ji, X. Luo, and T. Wang. Interpretable Deep Learning under Fire. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, 2020.

[324] Y. Zhang, P. D. Hassan Foroosh, and B. Gong. CAMOU: Learning A Vehicle Camouflage For Physical Adversarial Attack On Object Detections In The Wild. In *ICLR*, 2019.

[325] Y. Zhang and P. Liang. Defending against Whitebox Adversarial Attacks via Randomized Discretization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 684–693. PMLR, 2019.

[326] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang. ByteTrack: Multi-object Tracking by Associating Every Detection Box. In *European Conference on Computer Vision*, pages 1–21. Springer, 2022.

[327] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu. FairMOT: On the Fairness of Detection and Re-Identification in Multiple Object Tracking. *International Journal of Computer Vision*, 129:3069–3087, 2021.

[328] Y. Zhang, T. Wang, K. Liu, B. Zhang, and L. Chen. Recent advances of single-object tracking methods: A brief survey. *Neurocomputing*, 455:1–11, 2021.

[329] Y. Zhao, H. Zhu, R. Liang, Q. Shen, S. Zhang, and K. Chen. Seeing isn't Believing: Practical Adversarial Attack Against Object Detectors. *ACM CCS*, 2019.

[330] Y. Zhao, H. Zhu, R. Liang, Q. Shen, S. Zhang, and K. Chen. Seeing isn't Believing: Towards More Robust Adversarial Attack Against Real World Object Detectors. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 1989–2004, 2019.

[331] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu. Object Detection with Deep Learning: A Review. *IEEE NNLS*, pages 3212–3232, 2019.

[332] Y. Zhong, X. Liu, D. Zhai, J. Jiang, and X. Ji. Shadows Can Be Dangerous: Stealthy and Effective Physical-World Adversarial Attack by Natural Phenomenon. In *CVPR*, 2022.

[333] Y. Zhou and O. Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, pages 4490–4499, 2018.

[334] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M.-H. Yang. Online Multi-Object Tracking with Dual Matching Attention Networks. In *ECCV*, pages 366–382, 2018.

[335] W. Zhu, X. Ji, Y. Cheng, S. Zhang, and W. Xu. TPatch: A Triggered Physical Adversarial Patch. In *USENIX Security*, 2023.

[336] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*, 2018.

[337] A. Zolfi, M. Kravchik, Y. Elovici, and A. Shabtai. The Translucent Patch: A Physical and Universal Attack on Object Detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15232–15241, 2021.

[338] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye. Object Detection in 20 Years: A Survey. *Proceedings of the IEEE*, 2023.