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

Chen Ma*
ershang@stu.xjtu.edu.cn
Xi'an Jiaotong University
Xi'an, China

Ningfei Wang*
ningfei.wang@uci.edu
University of California, Irvine
Irvine, USA

Zhengyu Zhao
zhengyu.zhao@xjtu.edu.cn
Xi'an Jiaotong University
Xi'an, China

Qian Wang
qianwang@whu.edu.cn
Wuhan University
Wuhan, China

Qi Alfred Chen
alfchen@uci.edu
University of California, Irvine
Irvine, USA

Chao Shen†
chaoshen@mail.xjtu.edu.cn
Xi'an Jiaotong University
Xi'an, China

Abstract

Recent research shows that adversarial patches can attack object detectors in camera-based perception for Autonomous Driving (AD). However, camera-based perception includes more than object detection; it also involves Multiple Object Tracking (MOT), which enhances robustness by requiring consistent detection across multiple frames before affecting tracking and thus, driving decisions. This makes attacks on object detection alone less effective. To attack such robust systems, a digital hijacking attack has been proposed, aiming to induce dangerous scenarios such as collisions. However, this attack has limited effectiveness, especially in the physical world.

In this paper, we introduce a novel physical-world adversarial patch attack, ControlLoc, which exploits hijacking vulnerabilities in entire AD camera-based perception. ControlLoc utilizes a two-stage process: 1) identifying the optimal patch location and 2) generating the patch to modify the perceived location and shape of objects at that optimal location. Extensive experiments demonstrate the superior performance of ControlLoc, with an average attack success rate of around 98.1% across various AD camera-based perception and datasets, four times higher than that of the best existing method. Furthermore, the physical-world effectiveness of ControlLoc is validated in real vehicle tests under different conditions, such as outdoor lighting, angle, and background, achieving an average ASR of 79%. We also assess AD system-level impact with a production-grade AD simulator. ControlLoc yields a vehicle collision rate of 72.5% and an unnecessary emergency stop rate of 96.3%.

CCS Concepts

• **Security and privacy → Systems security**.

*Co-first author
†Corresponding author

CCS '25, Taipei, Taiwan

ACM ISBN 979-8-4007-1525-9/2025/10
https://doi.org/10.1145/3719027.3744842

Keywords

Adversarial Attacks, Autonomous Driving

ACM Reference Format:
Chen Ma, Ningfei Wang, Zhengyu Zhao, Qian Wang, Qi Alfred Chen, and Chao Shen. 2025. ControlLoc: Physical-World Hijacking Attack on Camera-based Perception in Autonomous Driving. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25), October 13–17, 2025, Taipei, Taiwan.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3719027.3744842

1 Introduction

Autonomous Driving (AD) vehicles, also known as self-driving cars, are increasingly becoming an integral part of our daily lives [23, 43]. Various companies [28], 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) [2, 27, 45] such as object detection, owing to the superior performance of DNNs.

Given that failing to detect objects can lead to violent crashes [51, 67], 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 [17, 24, 54, 56, 73], 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) [2, 25, 27, 49]. 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 [25], which is infeasible for previous attacks on object detection [24, 54, 56, 73].

Therefore, a digital adversarial hijacking attack [25] to fool the entire AD perception has been proposed with adversarial patches as the attack vector. By nature, this attack is 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, this prior attack [25] has shown limited effectiveness even in the digital space and is ineffective in the physical world fundamentally, since it manually specifies BBOX locations rather than relying on outputs from the object detector. Reproducing their patch generation method, described in their appendix, revealed low attack success rates with a 0% success rate in physical-world experiments. The detailed analysis and experiment results of the ineffectiveness for this adversarial hijacking attack [25] in both digital and physical domain are demonstrated in §5, §6.3, and §6.4.

In this paper, we propose the first effective 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. 3 (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. 1.

ControlLoc adopts a 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 §5.5, 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 §5.5, which demonstrates superior performance compared to existing methods in prior works [24, 25, 54, 73]. Due to these, ControlLoc can significantly outperform the existing hijacking attack [25].

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. On two driving datasets, ControlLoc achieves an impressive average attack success rate (ASR) of 98.1%. Furthermore, when compared with a baseline attack [25], 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.

Figure 1: Examples of monitor as attack vectors in real world.

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 [25] 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 [2], an industry-grade full-stack AD system with the LGSVL simulator [47], a production AD simulator with an average effectiveness of 84.4%. We also evaluate various state-of-the-art domain-specific defenses and existing directly adaptable DNN-level defenses on ControlLoc. For attack demos, code, and supplementary material, please check out our project website at **https://sites.google.com/view/av-ioat-sec/controlloc**.

To sum up, we make the following contributions:

- We propose the first practical and effective 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 designs, and etc.
- We evaluate ControlLoc on multiple AD perception systems and 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 such as vehicle collisions in a production AD simulator.

2 Motivation

Our primary motivation arises from the limitations of existing research [25], which lacks an effective method for generating adversarial patches capable of precisely controlling an object's location to achieve the object hijacking attack. Although a novel attack strategy is proposed, the evaluation in [25] relied on manually specifying BBOX locations rather than using adversarial examples to influence the outputs of the object detector. Furthermore, when we reproduced the adversarial patch generation method described in their appendix, we achieved only a 20.69% attack success rate in the digital domain (as shown in Fig. 6), and a 0% success rate

in physical-world experiments as shown in Table 3. These results suggest that while the proposed strategy may be conceptually interesting, it lacks a concrete implementation method, rendering tracker hijacking attacks against AD perception largely theoretical. Consequently, their work falls short of demonstrating the practical, real-world threats posed by adversarial vulnerabilities in MOT.

Furthermore, we find that existing adversarial attacks against AD systems mainly focus on making objects disappear, fabricating new ones, or causing objects to be misclassified, but lack an effective method to precisely control an object's location. Direct adaptation of these methods, as well as the method in [25], proves ineffective for this purpose. This ineffectiveness is due to limitations in their loss function design, optimization method, random patch location, and lack of precise BBOX filtering. Specifically, their loss function and optimization method struggle to handle the coupled conflicts between fabricating the new target BBox and erasing original ones, as well as conflicts among the position, shape, and confidence scores of the fabricated BBox. More detailed theoretical analysis and comparisons can be found in §6.3. Additionally, their BBOX filtering method is crude, preventing them from obtaining BBoxes with the potential to align attacker-specified locations and shapes. Successfully executing a tracker hijacking attack requires the attacker to fabricate a BBOX that both associates with the original tracker and produces the maximum possible hijacking speed. This means the BBOX must strike a delicate balance—it cannot deviate too far from the original tracker's position, nor can it be placed too close. This demands fine-grained BBOX selection during optimization, for which we propose a new precise BBOX filter based on the universal grid-based architecture of object detectors.

3 Background and Related Work

3.1 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. Such object detection plus MOT designs are commonly used in AD perception systems [33, 37, 38, 49] and are even presented in production-grade AD systems such as ZOOX [76]. As depicted in Fig. 2, the process initiates with a series of images. The AD perception algorithm employs an object detector [77] 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 [9]. 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, such as the planning [66], which facilitates decision-making processes. Since only the detection results with sufficient consistency and stability across multiple frames can be included in the tracking, the MOT can generally 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 [8]. State-of-the-art DNN-based object detectors are divided into two main categories: anchor-based and anchor-free approaches [68]. Anchor-based detection methods leverage a large number of preset anchors, then predict the category and refine the coordinates of these anchors,

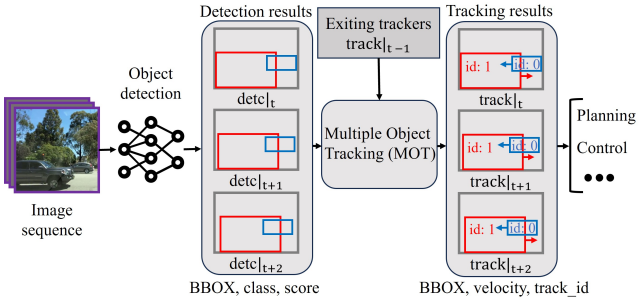


Figure 2: AD system pipeline: The camera captures images for object detection and multiple object tracking (MOT). Results are sent to the planning and then control modules.

and finally output these refined anchors as detection results. Conversely, anchor-free detection [52], directly predicts the bounding boxes of objects, offering a more generalizable solution.

Multiple Object Tracking (MOT). The state-of-the-art MOT can be broadly classified into two main approaches [13, 36]: 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 [36]. Tracking-by-detection has emerged as the predominant technique in MOT, particularly within the context of AD [13, 36, 71]. 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 [13]. These conditions render tracking-by-detection algorithms especially well-suited for integration into AD systems [13]. In this paper, we concentrate on the tracking-by-detection paradigm. As illustrated in Fig. 2, 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 [2, 25, 27, 75]. Thus, merely compromising the object detection component may not sufficiently disrupt the AD perception [25, 49]. Therefore, this paper introduces a novel physical-world adversarial hijacking attack strategy targeting the entire AD perception.

3.2 Previous Attacks and Comparisons

Attacks on Object Detection. Recent studies have highlighted the vulnerability of DNN models to adversarial examples or attacks [5, 7, 20, 41, 59, 69]. To improve their practicality especially in AD settings, further investigations have extended to the physical-world object detection adversarial attacks [17, 24, 34, 54, 56, 60, 73]. However, the entire AD perception encompasses both object detection and MOT. Given the 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 [25, 39, 56]. This

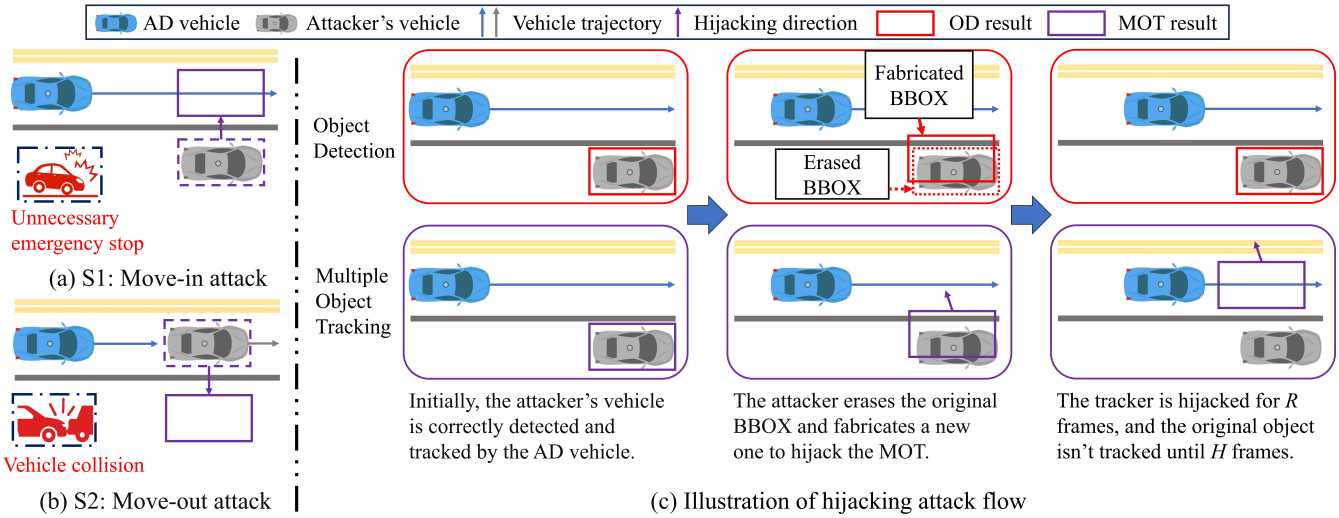


Figure 3: Attack goals: (a) move-in attack and (b) move-out attack. (c) shows hijacking attack flow. Rather than aiming for a stable and continuous attack success, ControlLoc achieves a sustained attack effect by successfully attack just a few frames. As shown in the figure, this brief success can have a lasting impact on the MOT, even if the OD recovers to benign detection.

paper proposes a novel, effective, and practical the physical-world adversarial hijacking attack on entire AD perception.

Attacks on Object Tracking. Various attacks targeting object tracking have been proposed, spanning both the digital [11, 63] and physical domains [14, 39]. Among them, AttrackZone [39] represents one of the most related physical domain hijacking attack against siamese-based tracking [30, 31], which is a single object tracking (SOT) [72], employing a projector to introduce adversarial perturbations. However, contemporary AD systems employ MOT rather than SOT [2, 10, 27, 29, 49] due to the requirement to identify and track multiple objects simultaneously [29]. Additionally, AttrackZone cannot be fundamentally extended to MOT for the following reasons: (1) The SOT in AttrackZone uses a DNN-based binary classification to label each pixel as either background or target, then inverts these binary labels. However, MOT requires tracking multiple objects simultaneously without background detection, rendering the AttrackZone methodology inapplicable; (2) AttrackZone relies on gradient information to generate attacks. However, in MOT, gradient information is unavailable due to non-differentiable processes such as tracker management introduced in §3.1; (3) In SOT, there are no strict constraints on the location of target BBOXes across consecutive frames. But, MOT enforces IOU constraints between BBOXes in consecutive frames, making it significantly more challenging to identify and generate the desired target BBOX accurately. Our paper addresses these challenges with new designs detailed in §5; (4) In SOT, an attack succeeds if the target BBOX achieves the highest confidence score (top-1). However, in MOT, success requires a high confidence score for the target BBOX and a reduction of the original BBOX's confidence score below a threshold. Without this, the tracker cannot be hijacked. This necessitates new designs detailed in §5.5. In this paper, we introduce a novel attack against the entire AD perception, i.e., object detection plus MOT, leveraging adversarial patch effective in different light conditions, angles, and backgrounds.

4 Attack Goal, Threat Model, and Vulnerability

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 [53]. We specifically explore physical-world attack vectors within the AD landscape, employing the adversarial patch due to its high practicality and realism [17, 38, 54, 73, 74]. Our research outlines two main hijacking attack goals: the move-in attack and the move-out attack, shown in Fig. 3 (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 [5, 17, 24, 25, 39, 48, 73]. 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 [1, 2, 15, 27, 71]. 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 [5, 48] for attack preparation. To effectively attack AD perception, we employ a monitor as an attack vector [38], a method with significant potential for dynamically displaying

adversarial patches, despite being rarely explored in the context of physical-world adversarial attacks.

MOT Vulnerability. The MOT vulnerability lies in its fundamental algorithm design, where attackers can generate tracker hijacking attacks as shown in Fig. 3 (c). The targeted MOT tracks objects that appear consistently for H consecutive frames and when tracked objects are missing, MOT retains trackers for R frames. This allows attackers to attack object detection for a small number of frames, but create a lasting hijacking effect. For instance, the tracker remains hijacked for R frames after the attack, and even if object detection recovers, the tracker remains hijacked for H frames. During the attack period, it is generally super dangerous especially when the vehicles with high speed. This vulnerability is general due to generality on this fundamental MOT design. Our survey of the MOT17 benchmark [42] found that 87% of the top 15 algorithms with code share this design and thus such a vulnerability is general.

5 ControlLoc Attack Methodology

5.1 Attack Design Overview

We provide a detailed overview of our ControlLoc. This hijacking attack flow is illustrated in Fig. 3 (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. 3 (c) and introduced in §3. To achieve the above attack strategy, we propose a dual-stage attack method, of which overview is in Fig. 4.

Stage I: This stage shown in Fig. 4 is an optimization-based approach to preselect the patch location. The details of this part will be introduced in §5.2. 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 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, 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 attack direction. The details are outlined in §5.3.

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 and is elaborated upon in §5.4.

Step 3: Loss Function Design and Optimization Method. This step introduces novel loss functions and a new optimization method detailed in §5.5. 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 [54, 73].

The novelty of our attack lies in several key aspects: we propose novel loss functions and overcome optimization challenges, becoming the first to jointly manipulate BBOX's position, shape, and confidence score during adversarial attacks (these three have coupling conflict effect challenge); whereas prior work typically targets only one or two of these aspects. By addressing the coupling conflict among all three, our method significantly advances the state of the art. We also design a novel BBOX-Filter to precisely filter BBOXes that meet attack criteria. This filter leverages a fundamental property of visual detectors—arising from the trade-off between CNN downsampling and localization precision. In addition, we propose Patch Location Preselection to guide where adversarial patches should be placed on the target object. This guidance is approximate rather than exact, enabling robust performance without pixel-level precision. Compared to random placement, our method boosts attack success rate by over 4 times as shown in Table 2.

5.2 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 ideal region for patch placement [12]. The detailed process is illustrated in Fig. 4. The objective function, denoted as $\mathcal{L}_{\text{mask}}$, is as follows:

$$\arg\min_{p,m} \mathcal{L}_{\text{adv}}(x') + \alpha \cdot \mathcal{L}_{M'}(m, \Delta_h, \Delta_w) \tag{1}$$

$$\textbf{where } \mathcal{L}_{M'} = \| \max(M') - \frac{1}{hw} \sum_{j=1}^{h} \sum_{i=1}^{w} M'[i,j] \|_1 \tag{2}$$

$$M'[i,j] = \sum_{z_1=0}^{\Delta_h - 1} \sum_{z_2=0}^{\Delta_w - 1} M[i+z_1, j+z_2] W[z_1, z_2] \tag{3}$$

$$M[i,j] = \frac{1}{2} \times \tanh(\gamma \cdot m[\lfloor \frac{i}{s} \rfloor, \lfloor \frac{j}{s} \rfloor]) + \frac{1}{2} \tag{4}$$

$$x' = x \odot (1 - M) + p \odot M \tag{5}$$

Equation (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 (4). The transformation using the tanh function in Equation (4) constrains the mask M

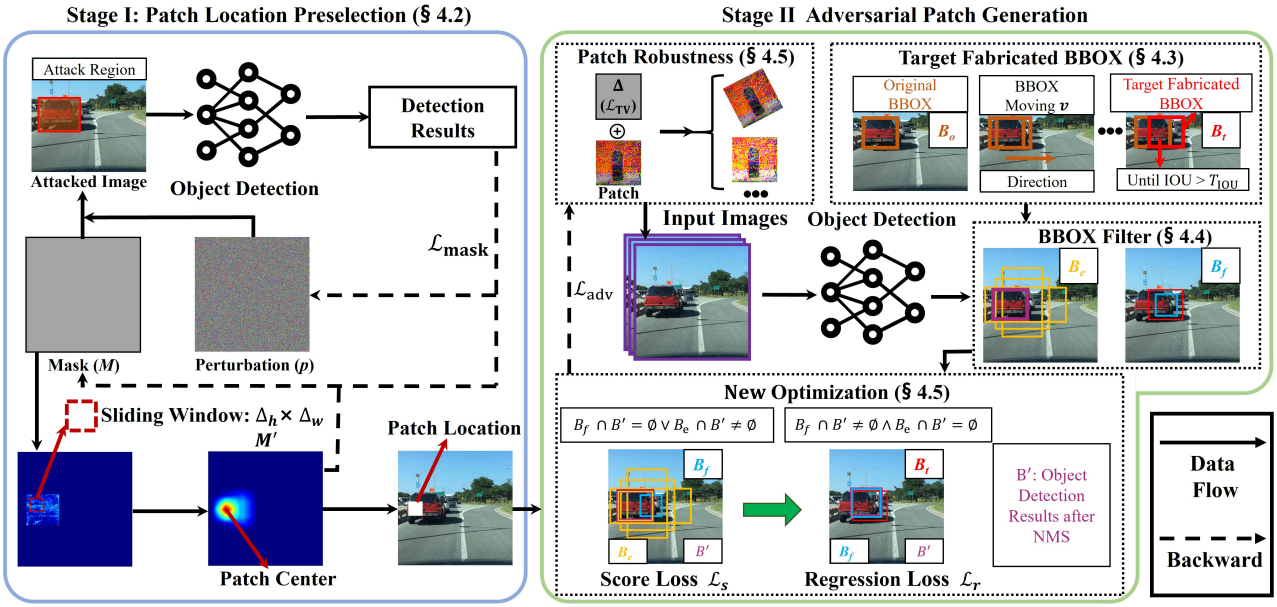


Figure 4: Overview of our ControlLoc, a two-stage hijacking attack via adversarial patches.

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 (5) to obtain the input x' for $\mathcal{L}_{\text{adv}}$. The details of $\mathcal{L}_{\text{adv}}$ will be introduced in §5.5. 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 (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.

Moreover, recognizing physical constraints on the capabilities of the attackers, only designated areas are considered viable for patch placement. Therefore, consistent with prior works [38, 40], 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: only 20 iterations in our experiments to determine the optimal location shown in §6.3.

5.3 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 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 [25], 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.

Algorithm 1 Find target fabricated BBOX location

Require: B_o: Original object BBOX; $\vec{v}$: Attacker desired directional vector; T_{IOU}: IOU threshold for data association between trackers and detection results.
Ensure: B_t: Target fabricated BBOX location.

```
k ← 1
B_t ← B_o
while IOU(B_t, B_o) > T_IOU do
    B_t ← B_o + v⃗ · k
    k = k + 1
end while
B_t ← B_o + v⃗ · (k − 1)
return B_t
```

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 [2, 15, 25, 27, 71], 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 in Algorithm 1 to find the desired deviation.

5.4 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 (§3.1).

The mainstream object detectors, including one-stage detectors such as the YOLO series, or two-stage detectors such as the RCNN series, introduced in §3, can adopt grid-based designs [22, 26, 35, 44–46]. 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, 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 §3. The details of the C-BBOX process are in Algorithm 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 (§5.3) 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 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.

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 [24], 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 [6], is used to distinguish between BBOXes for fabrication and those for erasure. This approach ensures our method's applicability across various object detection designs. The filter in Equation (6) help to extract the BBOXes needed to be fabricated B_f and erased B_e.

$$B_f, B_e = F(O_{\text{bbox}}, B_t, B_o) \tag{6}$$

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

Algorithm 2 C-BBOX

Require: B_t: Target BBOX; size$_f$: Feature map size; size$_x$: Image size; $\vec{v}$: Attack directional vector, k_s: Step size.
Ensure: B_f: BBOX needed to be fabricated.

```
(c_x, c_y) ← center point of B_t
scale = size_x / size_f
id_grid = int(c_x/scale, c_y/scale)
grid ← grid cell corresponding to id_grid
if detector is anchor_based then
    anchors ← all anchors of grid
    index_anchor = top(IOU(B_t, anchors));
    B_f = anchors[index_anchor]
else if detector is anchor_free then
    c_x = c_x + k_s · v/|v|
    id_grid = int(c_x/scale, c_y/scale)
    B_f ← the anchor of grid corresponding to id_grid
end if
return B_f
```

5.5 Loss Design and Optimization Method

As detailed in Algorithm 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 [54, 73], 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 (7), aiming to minimize Δ subject to conditions that B_f is encompassed within B' and B_e is excluded from B', where B' is all BBOXes after NMS.

$$\begin{aligned} &\arg\min_{\Delta} \mathcal{L}\{F[D(x,\Delta), B_t, B_o]\} \\ &\text{s.t. } B_f \in B' \text{ and } B_e \cap B' = \varnothing \end{aligned} \tag{7}$$

where Δ is adversarial patch and F is from Equation (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 in Equation (8).

$$\mathcal{L}_s = \underbrace{\frac{1}{|B_e|}\sum_{c\in B_e} \mathbb{1}^c \cdot c_{conf}^2}_{\mathcal{L}_e} + \underbrace{\mu_1 \cdot \frac{1}{|B_f|}\sum_{c\in B_f} (1-c_{conf})^2}_{\mathcal{L}_f} \tag{8}$$

$$c_{conf} = c_{obj} \cdot \max\{c_{class_i}\},\ i \in [1, N_c] \tag{9}$$

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. μ_1 is a hyperparameter.

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 in Equation (10).

$$\mathcal{L}_r = \overbrace{\frac{1}{|B_f|}\sum_{c\in B_f} -\log(\mathrm{IOU}(c, B_t))}^{\mathcal{L}_{\mathrm{IOU}}} + \underbrace{\beta \cdot \frac{1}{|B_f|}\sum_{c\in B_f} (\mathrm{center}(c) - \mathrm{center}(B_t))^2}_{\mathcal{L}_{\mathrm{center}}} \tag{10}$$

where the regression loss $\mathcal{L}_r$ comprises two components: $\mathcal{L}_{\mathrm{IOU}}$ and $\mathcal{L}_{\mathrm{center}}$. 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 (11) is used to reduce the color changes between the adjacent pixels.

$$\mathcal{L}_{\mathrm{TV}} = \sum_{i,j} \sqrt{\left|\Delta_{i+1,j} - \Delta_{i,j}\right|^2 + \left|\Delta_{i,j+1} - \Delta_{i,j}\right|^2} \tag{11}$$

Optimization Method. Simultaneously optimizing multiple loss functions, particularly $\mathcal{L}_s$ and $\mathcal{L}_r$, requires a sophisticated strategy. Existing literature [25] 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 with gradient descent.

In our case, this method is fundamentally ineffective. Notably, it does not perform well across various coefficient configurations, as detailed in §6.3. 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 for hijacking attack generation to address the limitations mentioned above:

$$\begin{aligned} &\arg\min_{\Delta} \mathcal{L}_{\mathrm{adv}} + \mu_2 \cdot \mathcal{L}_{\mathrm{TV}} \\ \textbf{where } \mathcal{L}_{\mathrm{adv}} &= \mathbb{1}[B_f \cap B' \neq \varnothing \text{ and } B_e \cap B' = \varnothing] \cdot \mathcal{L}_r \\ &+ \mathbb{1}[B_f \cap B' = \varnothing \text{ or } B_e \cap B' \neq \varnothing] \cdot \mathcal{L}_s \end{aligned} \tag{12}$$

Algorithm 3 Generating Adversarial Patch

Require: 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.
Ensure: Δ: Adversarial patch.

```
Initial Δ ← Δ_0
for n = 1 to N do
    O_bbox = D(x + Δ);
    B_f, B_e = F(O_bbox, B_t, B_o)
    B' = NMS(O_bbox)
    if B_f ∩ B' ≠ ∅ and B_e ∩ B' = ∅ then
        L_adv = L_r(B_f, B_t)
    else
        L_adv = L_s(B_f, B_e, T_conf)
    end if
    L = L_adv + μ_2 · L_TV
    Δ = Adam(Δ, L)
end for
return Δ
```

Our method optimizes either $\mathcal{L}_s$ or $\mathcal{L}_r$ based on the condition specified shown in Equation (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 [4]. 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 3. μ_2 is a hyperparameter.

Attack Robustness Enhancement. To enhance attack robustness, particularly in physical world, we incorporate the Expectation over Transformation (EoT) [3, 17, 24, 54] illustrated in Fig. 4. 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 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 patch generation method for disappearing attacks is in Appendix.

6 Evaluation

6.1 Evaluation Methodology and Setup

AD Perception. We include different AD perceptions, i.e., different object detection models and MOT. For object detection, we encompass both anchor-based and anchor-free detectors. Our examination mostly leverages algorithms in open-source full-stack AD systems

Table 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 #).

Attack scenario	OD\MOT		BDD dataset [64]				KITTI dataset [19]				Average
			ApoT	BoT-SORT	ByteTrack	StrongSORT	ApoT	BoT-SORT	ByteTrack	StrongSORT	
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
Average		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

to affirm the practicality and representativeness of our findings. We select a variety of object detection models, including the Baidu Apollo Object Detection (ApoD) [2]; YOLO v3 (Y3) [45] as incorporated in Autoware.AI [27]; YOLO v5 (Y5) [26] which is highlighted in recent security research on AD [24]; and YOLOX (YX) [18], 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: the MOT used in Baidu Apollo [2](ApoT); BoT-SORT [1]; ByteTrack [71], and StrongSORT [15]. We all use their default configurations.

Datasets. We select two widely recognized datasets in the AD research [5, 19, 25, 64]: the Berkeley Deep Drive (BDD) dataset [64] and the KITTI dataset [19]. 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.

6.2 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.* This metric is widely used in the security analysis of tracking [25, 39]. 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 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. 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 succeed than move-out attacks, the latter tend to reach attack goals faster within successful cases. From Table 1, StrongSORT exhibits greater robustness compared to others, except ApoT. This is likely due to a Noise Scale Adaptive Kalman filter [15] design, which adjusts measurement noise covariance based on confidence scores of detection results. Note that in this evaluation, we only generate the adversarial attack based on the object detection and evaluate attack effectiveness on entire AD perception. Thus, ControlLoc can potentially have very high attack transferability between different MOT algorithms.

6.3 Comparison with Baselines

6.3.1 Comparison with Prior Attack [25]. For the camera-based perception, we select different object detectors coupled with ApoT due to its robustness. The evaluation utilizes the BDD dataset in §6.1. Following the methodology of baseline research [25], we employ λ to denote the weighting factor between two loss functions in

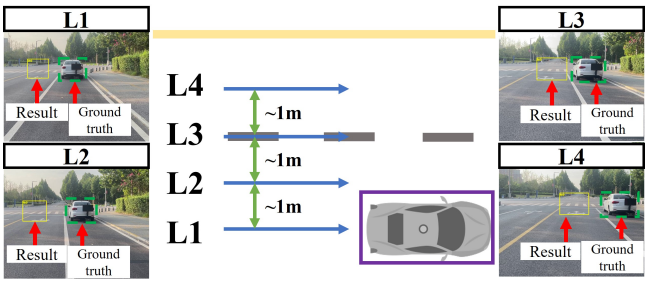


Figure 5: Physical-World attack evaluation setups.

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 [25], following the same setup as the baseline [25].

The results, as depicted in Fig. 6, 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 [25], 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 [25] 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.

6.3.2 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 (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 (8). To facilitate a detailed comparison, we use $\mathcal{L}_f$ and $\mathcal{L}_e$ for $\mathcal{L}_s$. Previous research leveraging SLRM [25] 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. 7 (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. 7 (a) to (c), where $\mathcal{L}_f$ barely decreases unless η is substantially increased, for example, to around 1000, as shown in Fig. 7 (d). However, elevating the η introduces a new problem: the regression loss ($\mathcal{L}_r$) fails to be well optimized, shown in Fig. 7 (d). This damages the attack's effectiveness, preventing the fabricated BBOX from associating with the target tracker. However, our approach successfully mitigates these issues in Fig. 7 (e).

Table 2: ASR between our patch location preselection (§5.2) and a random location preselection on rear of the vehicle.

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

6.3.3 Baseline Evaluation for Stage I in §5.2. 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 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.

6.4 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. 5, 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. 5 with around 1 m per segment. Among them, L1 and L2 correspond to the move-out attack, while L3 and L4 are used for the move-in attack. Our attack goals align with the angles, as introduced in §4. 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 [38]. 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 [2]. For the

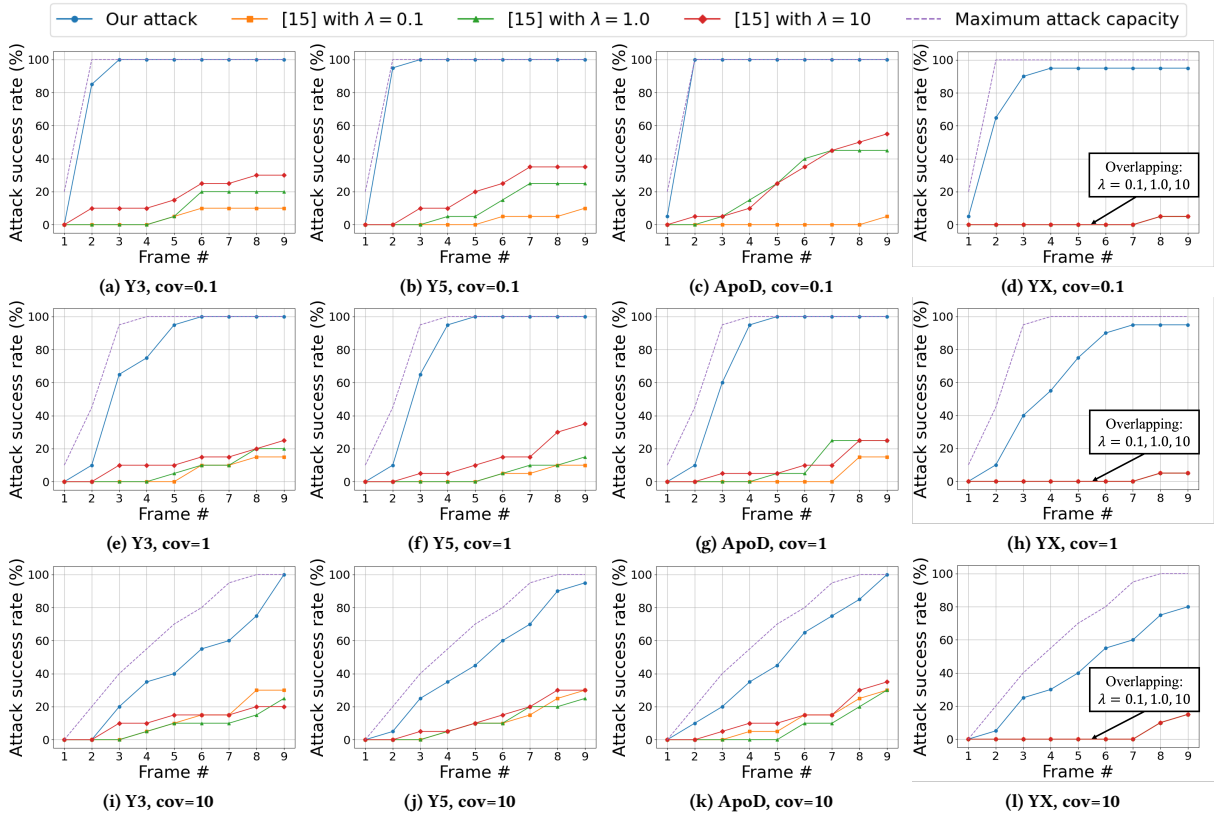


Figure 6: Comparison between our attack and the baseline attack [25] under four different object detection models (Y3, Y5, ApoD, and YX) with three different parameter values of ApoT (cov = 0.1, 1, 10). λ is the hyperparameter in [25]. Maximum attack capability assumes the attacker can arbitrarily control BBOX locations.

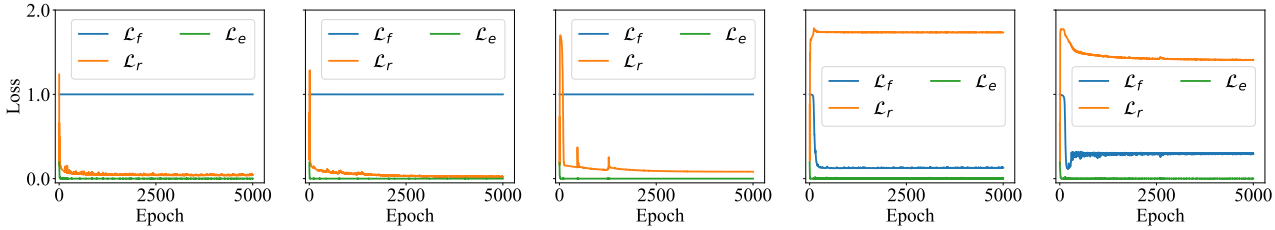


Figure 7: Comparison of loss value between our optimization method in ControlLoc and the standard Lagrangian relaxation method (SLRM) with different hyperparameter values η. The detailed loss designs are in §5.5: Equation (8) and Equation (10).

baseline [25], the hyperparameter λ is set to 10, as this value yields the best performance in §6.3, 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 compared with baseline attack [25], under variations in angle, light conditions, and background, is presented in Table 3. Our ControlLoc achieves a 79% average attack success rate, while the baseline [25] 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

Table 3: Physical-world attack evaluation regarding ASR for ControlLoc and baseline attack [25] 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. 5, 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	
	Baseline [25]	Ours	Baseline [25]	Ours	Baseline [25]	Ours	Baseline [25]	Ours	Baseline [25]	Ours	Baseline [25]	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%

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 (§6.2).

6.5 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 [2], an open-source full-stack AD system released by the commercial AD system provider Baidu, using LGSVL simulator [47], a production-grade AD simulator. Our experiments are conducted on the Borregas Ave map and the Lincoln2017MKZ AD vehicle. To better reflect real-world driving conditions, we incorporate random vehicles and pedestrians as surrounding obstacles, as well as environmental variability (including rain and fog conditions) into our simulation setup. To enhance the perception fidelity of simulators [54], we comprehensively model the hijacked tracker's displacement by characterizing its movement speed, duration, and location, and inject it into the AD system based on our physical-world attack evaluation results in §6.4 including the four different drive trajectories with different angles/lateral distances as shown in Fig. 5. Our evaluation focuses on two representative scenarios as shown in Fig. 3: move-in attack (Fig. 3 (a)), a common scenario for other vehicles to park on the side of the road, and move-out attack (Fig. 3 (b)), another common driving scenario. To evaluate the effectiveness of tracker hijacking attacks across different driving speeds, we conducted comprehensive simulations focusing on: 1) attack success rates under various speeds and 2) minimum tracker displacement requirements for successful attacks across speed ranges. Additionally, we evaluate the performance of the AD system in benign scenarios to verify its ability to function properly. We perform 10 runs on each scenario across a range of vehicle speeds.

Results. The outcomes are summarized in Fig. 9. Our ControlLoc achieves an average AD system-level attack effectiveness rate of 77.5% 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 (L3, L4) at 82.9% is notably superior to that of the move-out attack, which has a 72.1% rate. ControlLoc achieves high attack success rates across common driving speed ranges. Only in some low-speed scenarios (10 km/h in L1, L2, and L3) is the success rate relatively low. The average tracker displacement ControlLoc achieves (shown as red dashed line in Fig. 9 (b)) in physical-world experiments effectively covers almost all speeds in L1, L2, and L3 scenarios, only falling short of the displacement requirements for high-speed L4 scenarios (70km/h). For move-out attacks (L1, L2), higher speeds generally lead to higher success rates. This is because at lower speeds, the attack effect may expire before the AD vehicle collides with the attacker's vehicle, thus requiring larger tracker displacements to quickly move the attacker's vehicle tracker away, making the AD vehicle perceive the path ahead as obstacle-free sooner. Conversely, for move-in attacks (L3, L4), high-speed scenarios pose a different challenge: the AD vehicle may pass by before the attacker's vehicle has moved far enough ahead to trigger a braking response. As a result, attack success rates are lower at higher speeds, and greater tracker displacement is required to achieve effectiveness.

7 Discussion

7.1 Defenses

Domain-Specific Defenses. Two representative types of domain-specific defenses are involved: trajectory validation: PercepGuard [38], and motion consistency checks: VOGUES [40] and PhySense [65].

Following the official implementation and design of PercepGuard [38], we apply the AD perception results to the classifier to predict their classes and check the consistency. BDD dataset and four AD perception combinations introduced in §6.1 are used. ControlLoc achieved a 98.8% attack success rate in bypassing PercepGuard. We believe this high attack success rate stems from PercepGuard's assumption regarding consistency between object labels and trajectories: an object cannot exhibit motion patterns

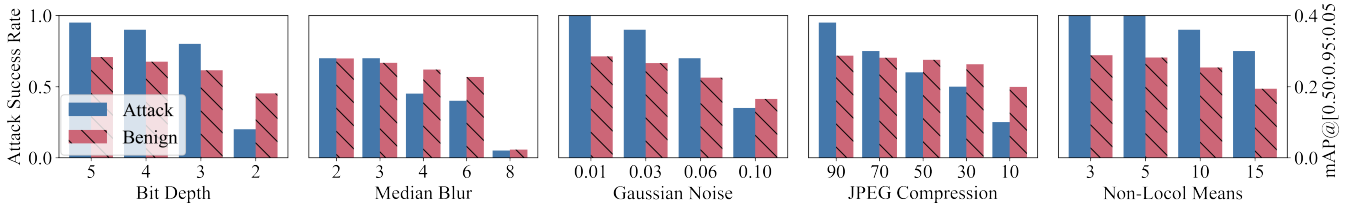
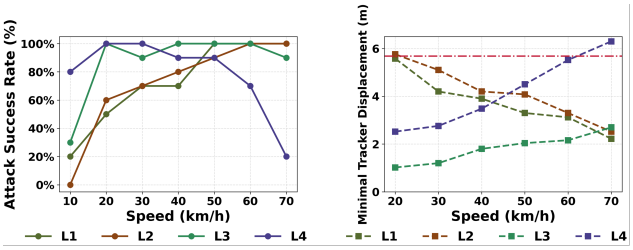


Figure 8: Attack effectiveness regarding attack success rate and model utility regarding mAP (mean Average Precision) of five common input transformation-based defenses. The x-axis represents the strength of each defense.



(a) Attack Success Rates under Various Speeds

(b) Minimum Tracker Displacement Requirements

Figure 9: AD system-level evaluation under different driving speeds using Baidu Apollo and LGSVL. Attack success rate is defined as the rate of vehicle collisions or unnecessary emergency stops. 10 runs for each scenario.

typical of another object class. For example, for an object to be classified as a car, it has to not only look like a car but also move like a car. This defense fails since the adversarial tracker hijacking movement follows normal vehicle motion patterns.

PhySense [65] defends against physical adversarial attacks leveraging static object attributes, dynamic behaviors, and inter-object interactions to detect and correct wrong perception results through multi-faceted consistency reasoning. We apply the physical adversarial patch generated by ControlLoc and input video into PhySense for prediction with the official implementation in the original paper. The patch sizes are consistent with the original settings: the larger patch measures 400 × 400 within a 1024 × 1024 mask, while the smaller patch is 200 × 200 within the same mask. ControlLoc achieves an 89.16% attack success rate in bypassing PhySense. This high success rate is attributed to the fact that our patch does not significantly alter the object's attributes, and any changes in the object's behaviors or interactions remain plausible and consistent with the expected characteristics of the object class.

We further perform an analysis of ControlLoc on VOGUES [40]. However, its official evaluation is limited to SOT attacks, which is not the focus of this paper. Our analysis further reveals that VOGUES is not applicable to MOT settings due to its high false positive rate (FPR) even in benign cases: 95.0% average FPR across different score thresholds (0.01–0.5) in the benign. This is due to the much higher probability of benign-case matching errors between pose estimation and object tracking in MOT scenarios. We believe that consistency check methods hold promise for defending against this type of attack. However, their design requires improvement on false positives caused by mismatches in the consistency matching.

Our evaluation shows that state-of-the-art domain-specific defenses proposed for AD perception in this problem space cannot defend or be applicable to ControlLoc. Thus, it still needs to develop more effective and targeted defenses for the MOT vulnerability.

General DNN-based Defenses. Prior research has focused on enhancing the robustness of DNNs against adversarial attacks. Such efforts fall into two broad categories: certified defenses [32, 57, 58] and non-certified defenses [16, 61, 70]. 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 [16], bit depth reduction [61], Gaussian noise [70], median blur [61], and non-local means [61, 69]. Due to their easily adaptable nature, these methods have been assessed in recent security studies [5, 48, 69, 74]. 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. 8, we observe that median blur can partially mitigate the attacks, particularly with large kernel sizes. However, it remains possible for the attack to succeed, and this harms the model, which may cause serious consequences in safety-critical applications [74].

Sensor Fusion Based Defense. Employing multi-sensor fusion (MSF) for improving perception robustness, such as integrating LiDAR, represents a strategic defensive approach.

State-of-the-art MSF can be broadly categorized into late fusion and early fusion approaches. In late fusion, our evaluation using Apollo's MSF demonstrates that attacks are effective when the main sensor is a camera, achieving a 76.8% ASR under the same setup as in §6.5. However, if LiDAR serves as the main sensor, it can often correct the errors introduced by camera-based attacks. This indicates that MSF can function as a practical defense mechanism, although certain MSF designs remain susceptible to attack. Importantly, many commercial AD systems such as Tesla and OpenPilot, primarily rely on camera-based perception. Given Tesla's widespread adoption, our attack poses substantial real-world implications. In early fusion, prior research [12] has shown that even camera-only attacks can significantly degrade MSF performance. Given that the 3D object detections they evaluated are also grid-based structures, our method can inform future attacks on feature-fusion MSF.

Other Defenses. VisionGuard [21] detects physical adversarial examples by exploiting inconsistencies across consecutive frames. However, because tracker-hijacking attacks introduce persistent, continuous trajectory manipulations, VisionGuard's defense effectiveness is inherently limited. PhyScout [62] detects sensor spoofing

attacks by formalizing and checking spatio-temporal consistency conflicts. However, the manipulated object trajectories caused by tracker hijacking attacks are crafted in a smooth and coherent fashion, maintaining both spatial and temporal consistency across frames. As a result, PhyScout is not fundamentally capable of detecting such attacks due to the inherent lack of spatial-temporal consistency conflicts.

Promising Defense Suggestion. We propose that stricter management of unmatched trackers — for example, through hierarchical frameworks or by integrating tracker confidence scores — can be an effective mitigation strategy. Nevertheless, effectively addressing false positives and false negatives under benign conditions remains an important avenue for future work in defense design.

7.2 Ethics

Our physical-world evaluation is taken to ensure both safety and responsibility. The experimental setup is located within our institute, under controlled conditions to ensure minimal traffic. All equipment is 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. We have performed vulnerability disclosure during the paper submission period to AD companies. We will closely follow up with them to mitigate this attack and avoid potential negative impacts.

7.3 Limitation and Future Work

This paper does not directly evaluate commercial AD systems. Prior research [55] indicates that adversarial attacks generally have limited impact on commercial AD systems, revealing a significant gap between academic prototypes and commercial systems. While bridging this gap requires significant effort, our findings still raise caution. For example, Tesla's AD system employs mechanisms similar to MOT—featuring short-term memory for occluded objects—and our findings reveal vulnerabilities in these "memory" buffer mechanisms [50]. Exploring these threats in commercial systems is an important direction for future research. Our research examines one-stage detectors. However, a systematic investigation on two-stage detection can be a future direction.

8 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 evaluated using a production AD simulator with 84.4% attack effectiveness.

Acknowledgments

We thank Weiwei Yang, and the anonymous reviewers for their valuable feedback. This research is supported by the National Key Research and Development Program of China (2023YFB3107400), the National Natural Science Foundation of China (U24B20185, T2442014, 62161160337, 62132011, U21B2018), the Shaanxi Province Key Industry Innovation Program (2023-ZDLGY-38, 2021ZDLGY01-02). Thanks to the New Cornerstone Science Foundation and the Xplorer Prize. Chao Shen is the corresponding author.

References

[1] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. 2022. BoT-SORT: Robust Associations Multi-Pedestrian Tracking. *arXiv preprint arXiv:2206.14651* (2022).
[2] Baidu Apollo. 2022. Baidu Apollo. http://apollo.auto.
[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. 2018. Synthesizing Robust Adversarial Examples. In *International conference on machine learning*. PMLR, 284–293.
[4] Oliver Bryniarski, Nabeel Hingun, Pedro Pachuca, Vincent Wang, and Nicholas Carlini. 2021. Evading adversarial example detection defenses with orthogonal projected gradient descent. *arXiv preprint arXiv:2106.15023* (2021).
[5] Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, and Bo Li. 2021. Invisible for both Camera and LiDAR: Security of Multi-Sensor Fusion based Perception in Autonomous Driving Under Physical-World Attacks. In *IEEE S&P*. IEEE, 176–194.
[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-End Object Detection with Transformers. In *European conference on computer vision*. Springer, 213–229.
[7] Nicholas Carlini and David Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE symposium on Security and Privacy*. IEEE, 39–57.
[8] Manuel Carranza-García, Jesús Torres-Mateo, Pedro Lara-Benítez, and Jorge García-Gutiérrez. 2020. On the Performance of One-Stage and Two-Stage Object Detectors in Autonomous Vehicles Using Camera Data. *Remote Sensing* (2020).
[9] Sudha Challa. 2011. *Fundamentals of Object Tracking*. Cambridge University Press.
[10] Jia Chen, Fan Wang, Chunjiang Li, Yingjie Zhang, Yibo Ai, and Weidong Zhang. 2021. Online Multiple Object Tracking Using a Novel Discriminative Module for Autonomous Driving. *Electronics* 10, 20 (2021), 2479.
[11] Xuesong Chen, Canmiao Fu, Feng Zheng, Yong Zhao, Hongsheng Li, Ping Luo, and Guo-Jun Qi. 2021. A Unified Multi-Scenario Attacking Network for Visual Object Tracking. In *AAAI*, Vol. 35. 1097–1104.
[12] Zhiyuan Cheng, Hongjun Choi, Shiwei Feng, James Chenhao Liang, Guanhong Tao, Dongfang Liu, Michael Zuzak, and Xiangyu Zhang. 2024. Fusion is Not Enough: Single Modal Attack on Fusion Models for 3D Object Detection. In *The Twelfth International Conference on Learning Representations*.
[13] Aman Dhar. 2020. *Object Tracking for Autonomous Driving Systems*. Ph. D. Dissertation. Master's thesis, EECS Department, University of California, Berkeley.
[14] Li Ding, Yongwei Wang, Kaiwen Yuan, Minyang Jiang, Ping Wang, Hua Huang, and Z Jane Wang. 2021. Towards Universal Physical Attacks on Single Object Tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
[15] Yunhao Du, Zhicheng Zhao, Yang Song, Yanyun Zhao, Fei Su, Tao Gong, and Hongying Meng. 2023. StrongSORT: Make DeepSORT Great Again. *IEEE Transactions on Multimedia* (2023).
[16] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. 2016. A study of the effect of JPG compression on adversarial images. *arXiv preprint arXiv:1608.00853* (2016).
[17] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. 2018. Robust Physical-World Attacks on Deep Learning Visual Classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1625–1634.
[18] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. 2021. YOLOX: Exceeding YOLO Series in 2021. *arXiv preprint arXiv:2107.08430* (2021).
[19] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. 2013. Vision meets Robotics: The KITTI Dataset. *IJRR* (2013).
[20] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *ICLR*.
[21] Xingshuo Han, Haozhao Wang, Kangqiao Zhao, Gelei Deng, Yuan Xu, Hangcheng Liu, Han Qiu, and Tianwei Zhang. 2024. VisionGuard: Secure and Robust Visual Perception of Autonomous Vehicles in Practice. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 1864–1878.
[22] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask R-CNN. In *ICCV*. 2961–2969.
[23] David Ingram. 2024. 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.
[24] Wei Jia, Zhaojun Lu, Haichun Zhang, Zhenglin Liu, Jie Wang, and Gang Qu. 2022. Fooling the Eyes of Autonomous Vehicles: Robust Physical Adversarial Examples Against Traffic Sign Recognition Systems. In *NDSS*.
[25] Yunhan Jia Jia, Yantao Lu, Junjie Shen, Qi Alfred Chen, Hao Chen, Zhenyu Zhong, and Tao Wei Wei. 2020. Fooling Detection Alone is Not Enough: Adversarial Attack against Multiple Object Tracking. In *ICLR'20*.
[26] Glenn Jocher. 2022. YOLOv5. https://github.com/ultralytics/yolov5.

[27] Shinpei Kato, Shota Tokunaga, Yuya Maruyama, Seiya Maeda, Manato Hirabayashi, Yuki Kitsukawa, Abraham Monrroy, Tomohito Ando, Yusuke Fujii, and Takuya Azumi. 2018. Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems. In *ICCPS*. IEEE, 287–296.

[28] Yosha Law. [n. d.]. 5 TOP SELF-DRIVING CAR MANUFACTURERS. https://yoshalawfirm.com/blog/5-top-self-driving-car-manufacturers/.

[29] Florin Leon and Marius Gavrilescu. 2021. A Review of Tracking and Trajectory Prediction Methods for Autonomous Driving. *Mathematics* 9, 6 (2021). doi:10.3390/math9060660

[30] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. 2019. SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks. In *CVPR*. 4282–4291.

[31] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. 2018. High Performance Visual Tracking With Siamese Region Proposal Network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 8971–8980.

[32] Linyi Li, Tao Xie, and Bo Li. 2023. SoK: Certified Robustness for Deep Neural Networks. In *2023 IEEE symposium on security and privacy (SP)*. IEEE, 1289–1310.

[33] Shaoshan Liu, Jie Tang, Zhe Zhang, and Jean-Luc Gaudiot. 2017. Computer Architectures for Autonomous Driving. *Computer* 50, 8 (2017), 18–25. doi:10.1109/MC.2017.3001256

[34] Giulio Lovisotto, Henry Turner, Ivo Sluganovic, Martin Strohmeier, and Ivan Martinovic. 2021. SLAP: Improving Physical Adversarial Examples with Short-Lived Adversarial Perturbations. In *USENIX Security*. 1865–1882.

[35] Xin Lu, Buyu Li, Yuxin Yue, Quanquan Li, and Junjie Yan. 2019. Grid R-CNN. In *CVPR*. 7363–7372.

[36] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. 2021. Multiple object tracking: A literature review. *AI* (2021).

[37] Chen Ma, Ningfei Wang, Qi Alfred Chen, and Chao Shen. 2024. Slowtrack: Increasing the latency of camera-based perception in autonomous driving using adversarial examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 4062–4070.

[38] Yanmao Man, Raymond Muller, Ming Li, Z Berkay Celik, and Ryan Gerdes. 2023. That Person Moves Like A Car: Misclassification Attack Detection for Autonomous Systems Using Spatiotemporal Consistency. In *USENIX Security*.

[39] Raymond Muller, Yanmao Man, Z Berkay Celik, Ming Li, and Ryan Gerdes. 2022. Physical Hijacking Attacks against Object Trackers. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 2309–2322.

[40] Raymond Muller, Yanmao Man, Ming Li, Ryan Gerdes, Jonathan Petit, and Z Berkay Celik. 2024. VOGUES: Validation of Object Guise using Estimated Components. *USENIX Security* (2024).

[41] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The Limitations of Deep Learning in Adversarial Settings. In *IEEE EuroS&P*. IEEE, 372–387.

[42] Paperswithcode. 2025. Multi-Object Tracking on MOT17. https://paperswithcode.com/sota/multi-object-tracking-on-mot17.

[43] PYMNTS. 2023. 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/.

[44] Joseph Redmon and Ali Farhadi. 2017. YOLO9000: Better, Faster, Stronger. In *CVPR*. 7263–7271.

[45] Joseph Redmon and Ali Farhadi. 2018. YOLOv3: An Incremental Improvement. *arXiv preprint arXiv:1804.02767* (2018).

[46] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE transactions on pattern analysis and machine intelligence* 39, 6 (2016), 1137–1149.

[47] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, et al. 2020. LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving. In *ITSC*. IEEE, 1–6.

[48] Takami Sato, Junjie Shen, Ningfei Wang, Yunhan Jia, Xue Lin, and Qi Alfred Chen. 2021. Dirty Road Can Attack: Security of Deep Learning based Automated LaneCentering under Physical-World Attack. In *USENIX Security*. 3309–3326.

[49] Junjie Shen, Ningfei Wang, Ziwen Wan, Yunpeng Luo, Takami Sato, Zhisheng Hu, Xinyang Zhang, Shengjian Guo, Zhenyu Zhong, Kang Li, et al. 2022. SoK: On the Semantic AI Security in Autonomous Driving. *arXiv:2203.05314* (2022).

[50] Karan Singh. 2024. Inside Tesla's FSD: Patent Explains How FSD Works. https://www.notateslaapp.com/news/2362/inside-teslas-fsd-patent-explains-how-fsd-works.

[51] LAUREN SMILEY. 2023. 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/.

[52] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. 2020. FCOS: A simple and strong anchor-free object detector. *IEEE transactions on pattern analysis and machine intelligence* 44, 4 (2020), 1922–1933.

[53] Ziwen Wan, Junjie Shen, Jalen Chuang, Xin Xia, Joshua Garcia, Jiaqi Ma, and Qi Alfred Chen. 2022. Too Afraid to Drive: Systematic Discovery of Semantic DoS Vulnerability in Autonomous Driving Planning under Physical-World Attacks. In *NDSS*.

[54] Ningfei Wang, Yunpeng Luo, Takami Sato, Kaidi Xu, and Qi Alfred Chen. 2023. 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)*. 4412–4423.

[55] Ningfei Wang, Shaoyuan Xie, Takami Sato, Yunpeng Luo, Kaidi Xu, and Qi Alfred Chen. 2025. Revisiting Physical-World Adversarial Attack on Traffic Sign Recognition: A Commercial Systems Perspective. In *NDSS*.

[56] Huixiang Wen, Shan Chang, Luo Zhou, Wei Liu, and Hongzi Zhu. 2024. OptiCloak: Blinding Vision-Based Autonomous Driving Systems Through Adversarial Optical Projection. *IEEE Internet of Things Journal* (2024).

[57] Chong Xiang, Arjun Nitin Bhagoji, Vikash Sehwag, and Prateek Mittal. 2021. PatchGuard: A Provably Robust Defense against Adversarial Patches via Small Receptive Fields and Masking. In *USENIX Security*. 2237–2254.

[58] Chong Xiang, Alexander Valtchanov, Saeed Mahloujifar, and Prateek Mittal. 2023. ObjectSeeker: Certifiably Robust Object Detection against Patch Hiding Attacks via Patch-agnostic Masking. In *IEEE S&P*. IEEE, 1329–1347.

[59] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. 2017. Adversarial Examples for Semantic Segmentation and Object Detection. In *ICCV*. 1369–1378.

[60] Kaidi Xu, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. 2020. Adversarial T-Shirt! Evading Person Detectors in a Physical World. In *ECCV*. Springer, 665–681.

[61] Weilin Xu, David Evans, and Yanjun Qi. 2018. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *NDSS*.

[62] Yuan Xu, Gelei Deng, Xingshuo Han, Guanlin Li, Han Qiu, and Tianwei Zhang. 2024. PhyScout: Detecting Sensor Spoofing Attacks via Spatio-temporal Consistency. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 1879–1893.

[63] Xiyu Yan, Xuesong Chen, Yong Jiang, Shu-Tao Xia, Yong Zhao, and Feng Zheng. 2020. Hijacking Tracker: A Powerful Adversarial Attack on Visual Tracking. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2897–2901.

[64] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. 2020. BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2636–2645.

[65] Zhiyuan Yu, Ao Li, Ruoyao Wen, Yijia Chen, and Ning Zhang. 2024. Physense: Defending physically realizable attacks for autonomous systems via consistency reasoning. In *ACM CCS*. 3853–3867.

[66] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. 2020. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. *IEEE access* 8 (2020), 58443–58469.

[67] Phate Zhang. 2022. Taiwanese star Jimmy Lin injured in Tesla crash. https://cnevpost.com/2022/07/22/taiwanese-star-jimmy-lin-injured-in-tesla-crash-report-says/.

[68] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. 2020. Bridging the Gap Between Anchor-based and Anchor-free Detection via Adaptive Training Sample Selection. In *CVPR*. 9759–9768.

[69] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. 2020. Interpretable Deep Learning under Fire. In *USENIX Security*.

[70] Yuchen Zhang and Percy Liang. 2019. Defending against Whitebox Adversarial Attacks via Randomized Discretization. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 684–693.

[71] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. 2022. ByteTrack: Multi-object Tracking by Associating Every Detection Box. In *ECCV*. Springer, 1–21.

[72] Yucheng Zhang, Tian Wang, Kexin Liu, Baochang Zhang, and Lei Chen. 2021. Recent advances of single-object tracking methods: A brief survey. *Neurocomputing* 455 (2021), 1–11.

[73] Yue Zhao, Hong Zhu, Ruigang Liang, Qintao Shen, Shengzhi Zhang, and Kai Chen. 2019. Seeing isn't Believing: Towards More Robust Adversarial Attack Against Real World Object Detectors. In *ACM CCS*. 1989–2004.

[74] Wenjun Zhu, Xiaoyu Ji, Yushi Cheng, Shibo Zhang, and Wenyuan Xu. 2023. TPatch: A Triggered Physical Adversarial Patch. In *USENIX Security*.

[75] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. 2018. Distractor-aware Siamese Networks for Visual Object Tracking. In *ECCV*.

[76] ZOOX. 2023. Going beyond seeing to perceiving the world around us. https://zoox.com/journal/perception/.

[77] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2023. Object detection in 20 years: A survey. *Proc. IEEE* 111, 3 (2023), 257–276.