

Can You Trust What You See? Alpha Channel No-Box Attacks on Video Object Detection

Ariana Yi¹ Ce Zhou² Liyang Xiao³ Qiben Yan³

¹Mission San Jose High School ²Missouri University of Science and Technology

³Michigan State University

yiariana7@gmail.com, cezhou@mst.edu, {xiaoliya, qyan}@msu.edu

Abstract—As object detection models are increasingly deployed in cyber-physical systems such as autonomous vehicles (AVs) and surveillance platforms, ensuring their security against adversarial threats is essential. While prior work has explored adversarial attacks in the image domain, those attacks in the video domain remain largely unexamined, especially in the no-box setting. In this paper, we present α -Cloak, the first no-box adversarial attack on object detectors that operates entirely through the alpha channel of RGBA videos. α -Cloak exploits the alpha channel to fuse a malicious target video with a benign video, resulting in a fused video that appears innocuous to human viewers but consistently fools object detectors. Our attack requires no access to model architecture, parameters, or outputs, and introduces no perceptible artifacts. We systematically study the support for alpha channels across common video formats and playback applications, and design a fusion algorithm that ensures visual stealth and compatibility. We evaluate α -Cloak on five state-of-the-art object detectors, a vision-language model, and a multi-modal large language model (Gemini-2.0-Flash), demonstrating a 100% attack success rate across all scenarios. Our findings reveal a previously unexplored vulnerability in video-based perception systems, highlighting the urgent need for defenses that account for the alpha channel in adversarial settings.

Index Terms—Video attack, No-box attack, Object detection, LLM security

I. INTRODUCTION

Artificial intelligence (AI) models are increasingly integrated into cyber-physical systems, empowering tasks such as obstacle avoidance in autonomous driving, environmental sensing in smart homes, and intelligent motion control in robotics. With the rapid advancement of AI, large language models (LLMs) are also being adopted in these domains [27]. Due to their low cost and portability, camera sensors have become one of the most widely used sensing modalities in such systems. As a result, computer vision tasks involving both images and videos play a critical role in system functionality.

Despite these advancements, security vulnerabilities persist due to the inherent weaknesses of AI models. To address practical and generalizable threats, black-box adversarial attacks have been widely studied [2, 4, 16, 24]. However, existing black-box attacks often suffer from key limitations, including excessive query requirements, low efficiency, and reduced success rates and confidence levels [26]. While some black-box attacks have been extended to the physical world [28, 29], they remain constrained by real-world challenges such as physical access and continuous control. Recently, Xia et al. [26]

proposed AlphaDog, a no-box universal attack, which exploits the previously overlooked alpha channel in images to achieve 100% success rate and confidence with high stealth. However, their work focuses solely on the image domain, leaving the video domain unexplored.

Notably, various video formats, such as Apple ProRes 4444 (.mov), HEVC (.hevc), WebM (.webm), OpenEXR (.exr), and Animated PNG (.apng), also support alpha channels. In the video domain, the alpha channel functions similarly to that in images. It works as a transparent layer enabling seamless blending of visual elements. It plays a critical role in video editing, web development, and graphic design. In this paper, inspired by AlphaDog [26], we propose the first no-box adversarial attack in the video domain, called α -Cloak, which targets object detection systems commonly deployed in cyber-physical environments.

Unlike the image domain, the video domain presents two unique challenges. First, most video players use black or gray backgrounds by default, rather than white, and not all video formats support alpha channels. To address this, we conduct extensive preliminary experiments to identify compatible video formats and analyze the background colors used by popular video players. Second, embedding an adversarial image into a video is non-trivial because videos consist of multiple frames, not a single static image. To overcome this, we design a novel fusion algorithm that combines a benign video and a malicious adversarial video. By carefully tuning key parameters, our method ensures the malicious content remains completely invisible to human observers while still being detected by AI models.

We evaluate α -Cloak on five widely used object detection models, one vision-language model (VLM), and extend our analysis to an LLM with visual capabilities. Because the adversarial content is embedded structurally within the video format, the attack remains robust across diverse models and video players, achieving a 100% success rate in all experiments.

Our contributions are summarized as follows:

- We present α -Cloak, the first no-box adversarial attack on object detection models processing video inputs. It requires no model queries, architecture knowledge, parameter access, or output feedback during generation.
- We demonstrate that adversarially perturbed videos can cause object detection models to consistently perceive a

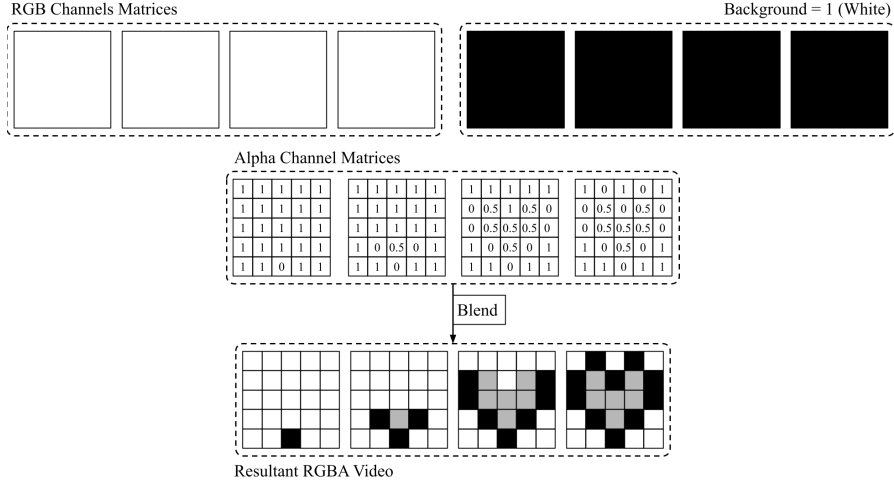


Fig. 1: An example of alpha channel blending illustrates how the alpha value interacts with the RGB channels and the background color to control pixel transparency and the final color seen in video display applications.

target malicious video while human viewers see only the original benign content. Unlike perturbation-based methods, *α -Cloak* introduces no visible noise for humans or detectors.

- We validate our approach across a broad range of vision and language models, including various types of object detection models, a VLM, and a multimodal LLM (Gemini-2.0-Flash). We achieve a 100% attack success rate across all cases. This highlights the broad applicability and robustness of our attack across diverse architectures and modalities.

II. BACKGROUND

In this section, we present background information on the RGBA video format and the role of the alpha channel. We then describe how such videos are rendered by video playback applications and processed by models.

A. RGBA Video Format and Alpha Channel

Digital videos have pixel formats that define the color and transparency information for each frame. Two pixel formats of the RGB color model are RGB and RGBA. The RGB format stores red, green, and blue channels that determine the color of each pixel. In an RGB video, each pixel stores values for these three channels, enabling the rendering of a wide spectrum of colors. RGBA extends this format by adding a fourth channel, the alpha channel.

The alpha channel represents the transparency level of each pixel and typically ranges from 0 to 255 in 8-bit encoding. A value of 255 indicates full opacity, while a value of 0 denotes complete transparency. Intermediate values result in varying degrees of partial transparency. In this paper, we normalize the alpha channel values to a range between 0 and 1 for consistency in the blending calculations. When an RGBA video is rendered, pixels with alpha values less than 1 reveal the underlying background of the video player, with the RGB values blended as an overlay. The final color of each pixel

displayed to viewers is computed through an element-wise alpha compositing operation, combining the original RGB values with the background color of the video player.

Formally, for each pixel in the rendered video, the resulting 3-element vector P is calculated as a weighted combination of the pixel's original RGB values C_{RGB} and the background color C_{BG} , using the normalized alpha value $\alpha \in [0, 1]$ as the blending factor. The compositing formula is:

$$P = \alpha \cdot C_{\text{RGB}} + (1 - \alpha) \cdot C_{\text{BG}}. \quad (1)$$

This operation is performed independently for each pixel in the frame, therefore applying the formula in an element-wise manner across the entire image. When this equation is applied to all pixels in a frame, this operation produces a 3-D output matrix of size $H \times W \times 3$, where H and W denote the height and width of the video frame, respectively.

An example of this process is shown in Fig. 1, where four RGBA video frames are composited over a black background to produce the final visible result. The appearance of the same RGBA video can vary depending on the alpha channel values and the background color rendered by the video player. In this example, each 5×5 square represents a single frame from the video. A group of four such frames together forms a segment of the overall video, demonstrating how alpha blending operates consistently across consecutive frames.

B. Background Colors of Video Player Applications

Digital video content can be rendered through various playback environments, including standalone video player applications and embedded viewers within web browsers. These players differ in their support for alpha channels. While some video players can interpret and render the alpha channel, they typically default to displaying the video over a solid background color. As a result, semi-transparent regions in the video may blend with the background, making portions of the background color visible to human viewers. This inconsistency

TABLE I: Background colors of video players.

Background Color	Thumbnail (Reduced-Size Video Display)	Viewer (Full-Size Video Display)
Black Background	VLC Media Player, macOS Finder, Apple TV, Adobe Premiere Pro, Capcut	VLC Media Player, QuickTime Player, Apple TV, Microsoft ClipChamp, Adobe Premiere Pro, Capcut, Vimeo Player
Grey Background	YouTube Player, Google Drive Video Player, OneDrive Player, Amazon Drive, iPhone Photos	YouTube Player, Google Drive Video Player, OneDrive Player, Amazon Drive
White Background	Vimeo Player	iPhone Photos

TABLE II: Alpha channel support in various video file formats.

	Supports Alpha Channel	Does Not Support Alpha Channel
Video File Format (Media Type)	Apple ProRes 4444 (.mov), HEVC (.hevc), WebM (.webm), OpenEXR (.exr), Animated PNG (.apng)	MPEG-4 (.mp4), Audio Video Interleave (.avi), Windows Media Video (.wmv)

in background handling leads to visual differences in how the same RGBA video appears across different platforms.

In addition to full video playback, most systems also generate video thumbnails, which are small preview images commonly shown in file explorers or gallery applications. These thumbnails are often rendered using a different default background color than the one used during full playback. Table I summarizes the background colors applied by popular video player applications during video playback and thumbnail rendering. The results indicate that *most video players default to a black background during playback, while thumbnail backgrounds tend to alternate between black and grey*.

C. Alpha Channel in Video File Types

Many video formats, such as .mp4, are designed for RGB videos and do not support alpha channels. Attempting to store transparency in these formats will result in the alpha channel data being discarded, and the video will default to being fully opaque. To retain the alpha channel, the video must be encoded using a file type that explicitly supports the RGBA format. Table II outlines some of the most widely used video file formats and their ability to support an alpha channel when displaying videos.

D. RGBA Video-Based Object Detectors Processing Pipeline

Most modern object detectors accept only three-channel RGB inputs, since they are trained on datasets of standard RGB images or videos [30]. When an RGBA video is provided, the alpha channel is removed during preprocessing, either explicitly removed or by converting the input to RGB format, so any transparency information is lost before inference [20, 26].

Modern object detection models are typically categorized into one-stage and two-stage architectures. One-stage detectors, such as YOLOv5 and its successors, perform object

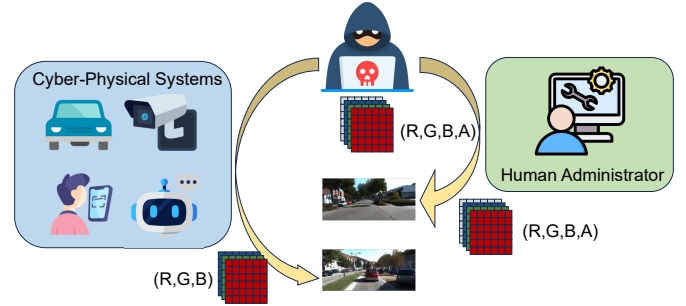


Fig. 2: Attack scenario

classification and localization in a single network pass [13]. In contrast, two-stage detectors such as Faster R-CNN [17] first generate region proposals and then classify and refine these proposals in a second network stage.

Recent advances in VLMs and LLMs have broadened the scope of visual recognition. Models such as Open-VCLIP extend the CLIP framework to videos, learning aligned embeddings between video inputs and textual class labels [25]. Gemini-2.0-Flash is a variant of Google DeepMind’s multimodal LLM, which processes both text and visual inputs through multimodal embeddings [21]. Both models operate on RGB inputs, with alpha channels ignored or discarded during preprocessing.

III. THREAT MODEL

As shown in Fig. 2, we consider a threat model grounded in cyber-physical systems where visual perception plays a critical role in system behavior. These systems, including autonomous vehicles (AVs), surveillance systems, face recognition systems and smart home robots, such as Tesla’s Full Self-Driving System (FSD) [23], mobile robots, such as Starship Technologies’ autonomy robots [22], and AI-driven sensing platforms, highly rely on object detectors to interpret the surrounding environment and later make real-time decisions. These models often process video inputs under the assumption that the video input is benign and clean. However, our attack exposes a possible vulnerability for the model when given an RGBA α -Cloak video.

Attack Goal. The attacker’s goal is to hide information in the alpha channel of an RGBA video, and then take advantage of the detector’s preprocessing step that drops this channel. Once the alpha channel is removed, the detector only sees the RGB image, which shows a scene picked by the attacker. This change is invisible to a person but can cause serious safety problems in real systems. For example, an AV using such an attacked video might miss important objects that are not in the attacker’s scene, leading to wrong and potentially dangerous driving decisions.

Attacker’s Capabilities. α -Cloak is a no-box attack that requires no knowledge of the detection model’s parameters or architecture. In our threat model, the attacker aims at compromising the object detector of a cyber-physical system. To launch the attack, they craft an adversarial RGBA α -Cloak video by merging two streams: a malicious “target” video

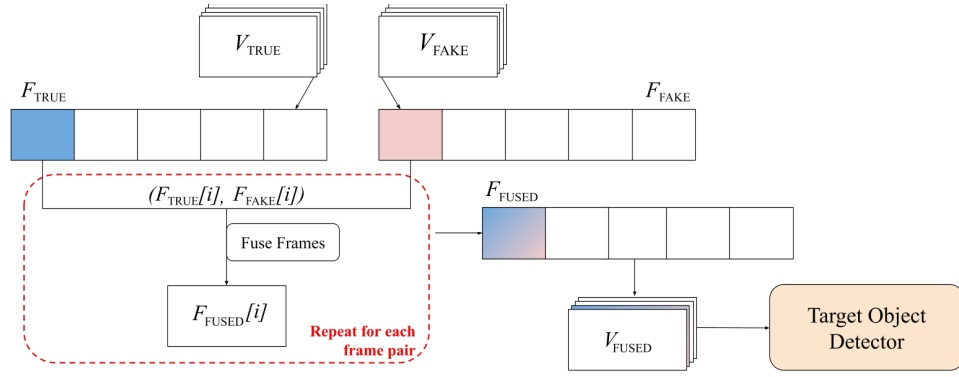


Fig. 3: Overview of the α -Cloak pipeline

meant for the detector and a benign video meant for human viewers. Because the detector accepts only RGB inputs and drops the alpha channel, it processes only the attacker's chosen scene. In addition, the attacker never needs physical access to the device; the attacked video can be delivered digitally (for example, via standard media uploads).

Attack Scenarios. Fig. 2 illustrates the attack workflow in a cyber-physical system. The adversary first creates an RGBA video by embedding a malicious target RGB stream into the alpha channel of a benign RGB video. This single tampered file is then uploaded through the cloud-based media interface of the system to the human administrator and the vehicle perception module. Because the perception module drops the alpha channel, it processes only the attacker's chosen frames, while the human operator sees the benign original sequence. Consequently, an AV may interpret a busy road as empty, leading to unsafe actions such as unintended lane changes or sudden acceleration.

IV. α -Cloak ATTACK DESIGN

In this section, we present the attack design of α -Cloak. We first present an overview of the attack, and then we detail how the attack is conducted on each frame of the video.

A. Attack Overview

As shown in Fig. 3, we define three video streams in α -Cloak: V_{TRUE} , V_{FAKE} , and V_{FUSED} . V_{TRUE} represents the benign, human-visible video, while V_{FAKE} denotes the malicious target video intended to deceive AI perception systems. The final adversarial output, V_{FUSED} , is generated by fusing V_{TRUE} and V_{FAKE} in a manner that preserves visual normality to human viewers but induces incorrect outputs in object detection models.

To construct V_{FUSED} , both V_{TRUE} and V_{FAKE} are first decomposed into their constituent frames. Let F_{TRUE} and F_{FAKE} denote the respective arrays of frames. For each corresponding pair $(F_{\text{TRUE}}[i], F_{\text{FAKE}}[i])$, we apply the FUSEFRAMES method to produce a fused frame, and further resulting in the array F_{FUSED} . The fused frames are then reassembled into a video at a consistent frame rate to produce the final adversarial video V_{FUSED} , which appears visually identical to V_{TRUE} to human

observers but is interpreted by detection models as V_{FAKE} . The full fusion pipeline is described in Algorithm 1.

Algorithm 1: Generate α -Cloak Fused Video

Input : Benign Video V_{TRUE} , Malicious Video V_{FAKE} , Frame Width l , Frame Height h

Output: The generated attack video V_{FUSED}

```

1 function GENERATEFUSEDVIDEO( $V_{\text{TRUE}}, V_{\text{FAKE}}, l, h$ ):
2    $F_{\text{TRUE}} \leftarrow \text{FRAMEPREP}(V_{\text{TRUE}}, l, h)$ ;
3    $F_{\text{FAKE}} \leftarrow \text{FRAMEPREP}(V_{\text{FAKE}}, l, h)$ ;
4   for  $i \leftarrow 0$  to  $\min(\text{len}(F_{\text{TRUE}}), \text{len}(F_{\text{FAKE}}))$  do
5      $F_{\text{FUSED}}[i] \leftarrow \text{FUSEFRAMES}(F_{\text{TRUE}}[i], F_{\text{FAKE}}[i])$ ;
6    $V_{\text{FUSED}} \leftarrow \text{GENERATE\_VIDEO}(F_{\text{FUSED}})$ ;
7   return  $V_{\text{FUSED}}$ ;

8 function FRAMEPREP( $V, l, h$ ):
9    $V \leftarrow \text{RESIZE}(V, l, h)$ ;
10   $F \leftarrow \text{SPLIT\_INTO\_FRAMES}(V) \rightarrow \{F_1, F_2, \dots, F_n\}$ ;
11  return  $F$ ;

```

B. Video Frame Preprocessing

To ensure successful fusion between the two input videos V_{TRUE} and V_{FAKE} , we first preprocess the two videos using the FRAMEPREP function outlined in Algorithm 1. The purpose of this step is to standardize spatial and temporal properties between inputs and to allow frame-level access for subsequent functions. The input videos are rescaled to a uniform frame size $l \times h$, ensuring compatibility for pixel-wise fusion. Finally, we split each video into an array of its frames, which enables direct access and manipulation during the fusion process.

C. Video Frames Combination

We perform frame fusion between both arrays of frames to generate a single composite video that embeds information from both V_{TRUE} and V_{FAKE} . After splitting the input videos V_{TRUE} and V_{FAKE} into frame arrays F_{TRUE} and F_{FAKE} , we apply the FUSEFRAMES function to each corresponding frame pair $(F_{\text{TRUE}}[i], F_{\text{FAKE}}[i])$. This per-frame fusion step is shown in lines 5 and 6 of Algorithm 2.

Algorithm 2: Generate Fused Frame

Input : Benign Frame F_{TRUE} , Malicious Frame F_{FAKE} .

Output: The generated fused attack frame V_{FUSED} .

```
1 function FUSEFRAMES( $F_{\text{TRUE}}$ ,  $F_{\text{FAKE}}$ ):  
2    $F_{\text{TRUE}} = \text{PREPROCESS}(F_{\text{TRUE}}) \times 0.4$ ;  
3    $F_{\text{FAKE}} = \text{PREPROCESS}(F_{\text{FAKE}}) \times 0.6 + 0.4$ ;  
4    $A_{\text{FUSED}} = \frac{F_{\text{TRUE}}}{F_{\text{FAKE}}}$   
5    $F_{\text{FUSED}} = \text{concatenate}(A_{\text{FUSED}}, F_{\text{FAKE}})$ ;  
6    $F_{\text{FUSED}} = F_{\text{FUSED}} \times 255.0$ ;  
7   return  $F_{\text{FUSED}}$ ;  
8 function PREPROCESS( $F$ ):  
9    $F = \text{grayscale}(F)$   
10   $F = F \div 255.0$ ;  
11  return  $F$ ;
```

To prepare the frames for fusion, we convert them both to grayscale and normalize their pixel intensities. We normalize pixel values to the range [0,1] by dividing each pixel by its maximum intensity value, i.e., 255 in 8-bit images.

To ensure that the content of $F_{\text{TRUE}}[i]$ and $F_{\text{FAKE}}[i]$ each remain perceptible in the fused output to their intended targets, while remaining imperceptible to the unintended side, we constrain the intensity ranges of both input frames. We adjust the frames such that $F_{\text{TRUE}} \leq F_{\text{FAKE}}$, ensuring that the alpha channel remains within the normalized bounds [0,1]. We calculate the alpha channel matrix $A_{\text{FUSED}}[i]$ using the following formula:

$$A_{\text{FUSED}}[i] = \frac{F_{\text{TRUE}}[i]}{F_{\text{FAKE}}[i]}. \quad (2)$$

Substituting Equation (2) into the inequality, we have the following:

$$0 \leq \frac{F_{\text{TRUE}}[i]}{F_{\text{FAKE}}[i]} \leq 1, \quad (3)$$

which directly implies the constraint $F_{\text{TRUE}} \leq F_{\text{FAKE}}$ across all pixels.

We empirically determine an optimal intensity range for both input videos. Through experimentation with 6,680 generated α -Cloak videos, we find that scaling F_{TRUE} to 40% of its original intensity, while maintaining F_{FAKE} values above 0.4 achieves the highest performing fusion quality. This results in the following bound:

$$0 \leq F_{\text{TRUE}}[i] \leq 0.4 \leq F_{\text{FAKE}}[i] \leq 1. \quad (4)$$

We finalize each fused frame by combining the computed alpha channel matrix with its RGB channel intensity matrix. Because systems typically remove the alpha channel matrix when rendering the frame, we assign the RGB channel intensity matrix equal to F_{FAKE} , as this will be the only image that the computer will see. Thus, the resulting fused frame is: $F_{\text{FUSED}}[i] = A_{\text{FUSED}}[i] + V_{\text{FAKE}}[i]$.

V. EVALUATION

To evaluate the proposed attack, we input adversarially fused videos into multiple object detection models and measure how closely their predictions align with the content of either the benign or malicious source video.

A. Experimental Setup

1) *Experimental Procedure.*: We design this experiment to evaluate the extent to which object detection models can identify and localize objects within adversarially fused videos. Each target model receives a list of attacked videos along with the ground truth bounding box labels corresponding to each original, unaltered input video used to construct those attacked videos.

For each frame in a given attacked video, the detection model performs inference and outputs its predicted bounding boxes. We then compute a frame-level similarity score (*FLS*) by comparing the model prediction to all ground truth boxes. This process is repeated for each frame, and the resulting *FLS* values are averaged to obtain a video-level similarity score (*VLS*) between the attacked video and each candidate source video. The candidate source video with the highest average similarity is identified as the most likely source video, indicating that the model's predictions most closely resemble that video's object layout. This experiment allows us to quantify how closely the fused content influences model perception and how effectively the attack obscures source attribution.

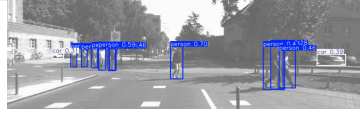
2) *Target Attack Models.*: We evaluate our attack using five widely adopted object detection architectures, selected to cover a diverse range of model structures. Specifically, we test three versions of YOLO, including YOLOv5n [10], YOLOv8n [11], and YOLOv11n [12], using their official pre-trained weights. Additionally, we include RetinaNet [14] and Faster R-CNN in our evaluation, both of which utilize a ResNet-50 backbone with a Feature Pyramid Network (FPN) to enhance multi-scale feature extraction.

While our evaluation focuses on standard object detection benchmarks using widely adopted architectures (YOLOv5/8/11, Faster R-CNN, RetinaNet), these models constitute the core perception modules in many modern vision-based systems, including autonomous vehicles, surveillance platforms, and robotics pipelines. Evaluating at this level allows us to precisely measure the model-level effects of our attack, which directly influence downstream system behavior.

All models generate bounding boxes along with associated class confidence scores for each input video. We apply a fixed confidence threshold of 0.25 across all models to increase object recall, prioritizing detection coverage over precision. This choice ensures that our similarity metric is sensitive to all detectable object instances. This diverse selection of models provides a robust basis for evaluating the generalizability and effectiveness of our attack strategy.

3) *Evaluation Metrics.*: To assess the similarity between each attacked video and its potential source videos, we introduce a two-level similarity metric framework: *FLS* and *VLS*.

TABLE III: Object detectors attack performance on three videos.

Target frame seen by humans (V_{TRUE})	Target frame seen by object detectors (V_{FAKE})	Object Detectors Output	Object Detector
			YOLOv5
			YOLOv11
			RetinaNet

These metrics rely on spatial overlap between predicted and ground truth boxes, measured using Intersection over Union (IoU), how much two bounding boxes overlap.

For each predicted box p in a given attacked frame, we compute the IoU against each ground truth box g across all candidate videos and retain the maximum value:

$$\text{IoU}(A, B) = \frac{\text{area}(A \cap B)}{\text{area}(A \cup B)}. \quad (5)$$

$$FLS = \frac{1}{n} \sum_{i=1}^n \max_{j=1}^m (\text{IoU}(p_i, g_j)). \quad (6)$$

We repeat this process for every frame in the attacked video and then compute the average FLS across all T frames to compute the VLS for each candidate video:

$$VLS = \frac{1}{T} \sum_{t=1}^T FLS_t. \quad (7)$$

This resulting VLS for every candidate video captures how closely each candidate video matches the video detected by the object detector. A higher VLS indicates stronger alignment between the candidate's ground-truth content and the detector's predictions on the attacked video. These metrics allow us to quantify and assess how convincingly our attack blends the benign and malicious videos.

4) *Dataset.*: We conduct our evaluations using the KITTI tracking dataset [6], which provides annotated video sequences for real-world urban driving. We convert the individual KITTI tracking sequences into full-length videos, yielding a set of 21 complete candidate videos. We split the first 20 videos into two equal subsets: the first 10 videos serve as V_{TRUE} videos and the next 10 videos as V_{FAKE} videos. These fused videos, along with the original KITTI training labels for all 21 videos, are provided as input to the object detection models during evaluation. This experiment design allows us to evaluate attack performance in a multi-object urban context with dynamic backgrounds. Table III showcases examples of attacked frames alongside bounding box outputs from three models. The first and second columns are frames that humans and AI should see, respectively. The third and fourth columns show the bounding

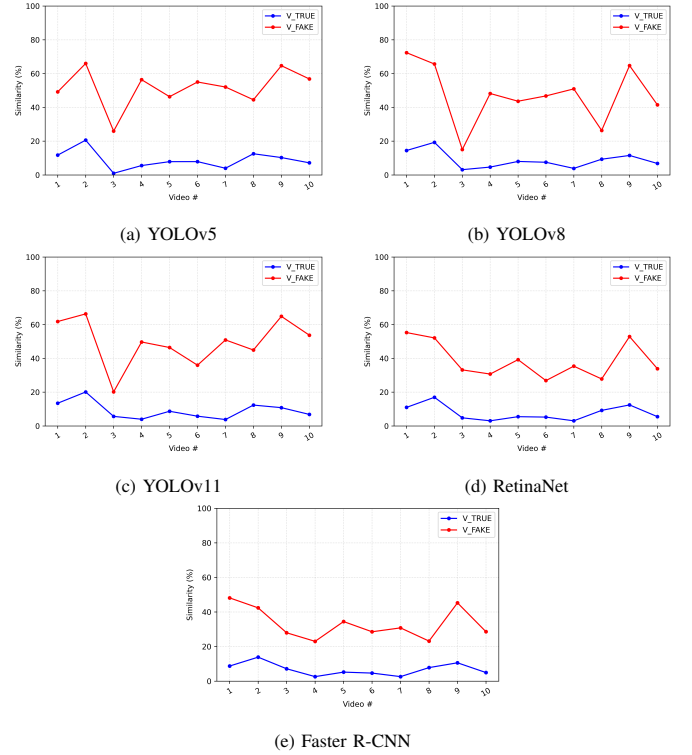


Fig. 4: Evaluation results across different detection models. The x-axis represents each attacked video; the y-axis is the VLS (%). Red: V_{FAKE} ; blue: V_{TRUE} .

boxes predicted for each frame, and which model ran the image. It can be seen that the object detector views the V_{FAKE} image and runs its object detection on the malicious target video.

B. Attack Results on KITTI

We evaluate the effectiveness of our attack by measuring how closely the fused videos resemble the source components across five object detection models. For each attacked video, we compute the VLS relative to its corresponding V_{TRUE} and V_{FAKE} videos, and plot the results in Fig. 4.

Across all models and videos, we observe that the red line, representing the VLS of V_{FAKE} , consistently lies above the blue

TABLE IV: Summary of VLS evaluation results across the different object detection models.

Model	Avg. VLS to V_{FAKE} (%)	Avg. VLS to V_{TRUE} (%)	V_{FAKE} Top-1 (%)	V_{TRUE} Top-1 (%)
YOLOv5	51.689	8.852	100	0
YOLOv8	47.424	8.852	100	0
YOLOv11	49.487	9.138	100	0
RetinaNet	38.733	7.665	100	0
Faster R-CNN	33.237	6.715	100	0
OVERALL AVG	44.114	8.244	100	0

line, the VSL of V_{TRUE} . This consistent pattern demonstrates that the object detection models perceived the fused videos as more similar to V_{FAKE} , indicating the success of the attack in misleading the models.

While the difference between the VLS scores of V_{FAKE} and V_{TRUE} are not extreme in magnitude, the direction of the shift is consistent across all models and examples. We attribute the limited gap to the intrinsic visual similarity among the KITTI videos, as all scenes are captured from a vehicle through similar environments. This inherent similarity likely increases the challenge of precise disambiguation.

We summarize quantitative results in Table IV, which reports the average VLS for each model with respect to both source videos, as well as the percentage of cases where V_{FAKE} or V_{TRUE} was identified as the top-1 most similar candidate. In all models, V_{FAKE} leads as the top-1 prediction. This result confirms that our attack effectively redirects model recognition from the benign video and towards the malicious target video, regardless of model architecture.

C. Attack Results on Open-VCLIP

To evaluate the transferability of our attack beyond object detection, we test it on a VLM, Open-VCLIP, using the UCF-101 dataset [19]. Open-VCLIP is a contrastively trained extension of CLIP designed for video inputs, and it is pre-trained on UCF-101 itself, making it highly familiar with the dataset’s class distribution.

Given a video clip and a predefined set of class labels, Open-VCLIP computes embeddings for both the video input and each class label, ranking all class labels based on cosine similarity, and returns the top-1 and top-5 predicted classes. This allows us to test whether our attack changes the model’s prediction to favor the malicious adversarial label.

We generate 6,660 adversarially fused videos by splitting UCF-101 into two disjoint halves. Videos in the first subset serve as V_{TRUE} , while the second subset serves as V_{FAKE} . For each index i , we construct a fused video $V_{FUSED}^{(i)}$ by applying fusion (Algorithm 1) to the pair $(V_{TRUE}^{(i)}, V_{FAKE}^{(i)})$.

We submit each V_{FUSED} video to Open-VCLIP twice: once with the class label of V_{TRUE} , and once with the class label of V_{FAKE} . For both runs, we check whether the submitted label appears in the model’s top-1 or top-5 predictions. Table V

TABLE V: Top-1 and Top-5 classification accuracy of Open VCLIP.

Label Given	% of labels in Top-1 list	% of labels in Top-5 list
V_{TRUE} Labels	0.06	1.83
V_{FAKE} Labels	71.56	90.68

TABLE VI: Attack performance on Gemini-2.0-Flash.

Target frame seen by humans (V_{TRUE})	Target frame seen by object detectors (V_{FAKE})	Gemini-2.0-Flash Output
		The vehicle is traveling on a street with parked cars on the right side. There are sidewalks and grassy areas adjacent to the road. There are pedestrian crossings with signage. There are other vehicles moving in the same direction and parked.
		The video captures a street scene with a cyclist initially. The cyclist rides across the scene from right to left. As the video progresses the cyclist exits the frame, and then a person walking on the sidewalk on the left side of the road becomes visible.
		The video shows a close-up view of someone knitting. The primary focus is on the hands and knitting needles as they manipulate yarn to create stitches. The actions appear to be those of a standard knitting technique. Due to the visual quality, it’s difficult to pinpoint the specific stitch being made, but it looks like a basic knit stitch.

summarizes the classification accuracy under both conditions across all 6,660 attacked videos. The middle row displays the detection accuracy of V_{FUSED} using the class label corresponding to the human-visible video V_{TRUE} . The bottom row displays the detection accuracy of V_{FUSED} when using the class label corresponding to the AI-targeted video V_{FAKE} .

The label of V_{TRUE} appears in the top-1 prediction less than 2% of cases, while the label of V_{FAKE} appears in over 90% of top-1 predictions. This strong skew in prediction distribution indicates that Open-VCLIP consistently aligns with the adversarial target V_{FAKE} , even in the presence of benign content V_{TRUE} , highlighting the generalizability and strength of our attack across modalities.

D. α -Cloak Attack Example on Gemini-2.0

We extend our investigation on a commercial multimodal LLM, Gemini-2.0-Flash. As shown in the Table VI, the first column presents video frames that have been covertly manipulated via the alpha channel, which appear entirely normal to human observers. The second column shows the corresponding attack samples after the large language model strips away the alpha channel. It can be seen that from the LLM’s point of view, these frames convey a completely different scene. The third column reports the model’s analysis of each attack sample. From the results, we can see that the LLM reads the adversarial videos V_{FAKE} instead of V_{TRUE} and demonstrates the effectiveness of α -Cloak on Gemini-2.0-Flash.

VI. RELATED WORK

1) *Security of Image Preprocessing Pipelines.*: In the domain of black-box adversarial attacks, a large body of work has proposed query-based iterative methods to improve attack efficiency and reduce reliance on substitute models, such as Square Attack [2], Boundary Attack [3], HopSkipJumpAttack [4], GenAttack [1], the triangle attack [16], bandit-based

approaches [8], and SimBA [7]. Although these methods enhance query efficiency, they still require hundreds to thousands of queries and are generally tailored to specific models. At the same time, image-scaling attacks exploit preprocessing-stage resizing algorithms to conceal malicious payloads within benign images. Most closely related to our work, AlphaDog [26] is a ‘no-box’ camouflage attack to exploit the alpha channel of RGBA images; it embeds the adversarial target into the RGB channels to mislead AI classifiers while crafting the alpha channel so that human observers see only innocuous content, achieving zero queries and model-agnostic applicability. In this paper, we move from the image/classification setting to the video/object-detection setting, where temporal consistency and region-level localization (rather than global labels) make attacks substantially harder.

2) *Security of Vision-based Perception Systems.*: Vision is central to cyber-physical systems (e.g., AVs), but has proven vulnerable to both perturbation and patch attacks that directly corrupt input images.

Both perturbation and patch attacks fall under the umbrella of adversarial attacks. The fundamental idea of adversarial attacks is to induce significant errors in a deep learning model’s output through minimal modifications to the input. Perturbation attacks typically affect all pixels in the input image with slight value changes, whereas patch attacks modify only a small region but with relatively large alterations in pixel values. For example, V-Phantom [9] introduces adversarial perturbations in captured images by adjusting the camera’s supply voltage, thus disrupting the downstream image recognition process. GhostShot [18], on the contrary, achieves the injection of adversarial patterns into CCD cameras through externally applied electromagnetic interference. Cheng et al. [5] demonstrate that the image stabilization mechanism used in autonomous driving camera sensors can be disrupted by malicious ultrasonic signals, inducing abnormal jitter and dynamic blur in acquired images. L-Hawk [15] adopts a similar attack concept against autonomous driving platforms, but replaces the injected signal with a laser beam precisely aimed at the camera lens. These attacks require physical access/proximity, environmental control, or specialized hardware, and often produce conditions (e.g., jitter, blur, overexposure) that can be perceptible or operationally constraining. α -Cloak attack does not interact with the sensor or the physical environment. Instead, it targets the downstream video handling and model-input stack by exploiting how RGBA videos are decoded and consumed by perception models.

VII. DISCUSSION

In this section, we discuss the practicality, limitations, and the potential defense method of the α -Cloak attack.

A. Attack Feasibility and Practicality

Our work demonstrates that no-box alpha channel-based attacks can be both simple and effective. Unlike existing video attacks that rely on temporal perturbations and require knowledge of the model architecture or parameters, α -Cloak

exploits a fundamental inconsistency in video input handling. It embeds an adversarial payload into the alpha channel, which is commonly ignored or discarded by object detectors. As a result, α -Cloak remains lightweight, broadly applicable, and agnostic to detector architectures and media playback environments.

B. Limitations

Although α -Cloak is effective against a wide range of object detectors, it depends on two key assumptions: detectors discard the alpha channel while standard video players preserve it, conditions met by most systems trained solely on RGB inputs. Furthermore, α -Cloak is inherently limited to grayscale content, since the alpha channel controls only pixel transparency without altering the relative intensities of the RGB channels. Consequently, the fusion mechanism cannot reproduce full-color scenes, restricting the attack to monochrome videos or regions.

C. Defense

α -Cloak exploits the mismatch in video pipelines in how video content is presented to human viewers compared to detection models. It leverages the removal of the alpha channel by models to hide adversarial content in plain sight. To defend against this attack, model designers can implement alpha channel profiling techniques tailored for video input during the preprocessing stage. After decoding each video frame, but before model inference, the system can conduct per-frame alpha channel analysis and compute per-pixel intensity histograms to detect any unnatural transparency distributions. Frames exhibiting nonuniform transparency in regions where transparency is not expected, such as in the center of the video in high-traffic videos, will be flagged by the detector.

Alternatively, instead of discarding the alpha channel in its entirety, a model can first composite each incoming RGBA frame onto a black background before passing it to the object detection model. This approach emulates how standard video players render transparent regions, ensuring consistency between human and model perception. Therefore, the opportunity to exploit rendering mismatches is eliminated.

VIII. CONCLUSION

In this paper, we introduce α -Cloak, the first no-box adversarial attack targeting object detection systems in the video domain. By leveraging the alpha channel, we demonstrate that adversarial content can be stealthily embedded within videos without any perceptible distortion to human viewers. Our approach addresses the unique challenges of video processing by proposing multi-frame fusion. Extensive evaluations across a range of object detectors, a vision-language model, and an LLM confirm the attack’s robustness and universality, achieving a 100% success rate in all scenarios. Our findings highlight a critical and previously overlooked threat vector in cyber-physical systems, emphasizing the urgent need for new defense mechanisms to protect video-based AI applications from invisible adversarial manipulation.

REFERENCES

- [1] 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 *Proceedings of the genetic and evolutionary computation conference*, 2019, pp. 1111–1119.
- [2] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: a query-efficient black-box adversarial attack via random search," in *European conference on computer vision*. Springer, 2020, pp. 484–501.
- [3] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," *arXiv preprint arXiv:1712.04248*, 2017.
- [4] J. Chen, M. I. Jordan, and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 1277–1294.
- [5] Y. Cheng, X. Ji, W. Zhu, S. Zhang, K. Fu, and W. Xu, "Adversarial computer vision via acoustic manipulation of camera sensors," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 3734–3750, 2023.
- [6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The international journal of robotics research*, vol. 32, no. 11, pp. 1231–1237, 2013.
- [7] C. Guo, J. Gardner, Y. You, A. G. Wilson, and K. Weinberger, "Simple black-box adversarial attacks," in *International conference on machine learning*. PMLR, 2019, pp. 2484–2493.
- [8] A. Ilyas, L. Engstrom, and A. Madry, "Prior convictions: Black-box adversarial attacks with bandits and priors," *arXiv preprint arXiv:1807.07978*, 2018.
- [9] Y. Jiang, R. Li, Y. Cheng, X. Ji, and W. Xu, "V-phantom: voltage-based physically-triggered backdoor attack against facial recognition," in *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025, pp. 1–5.
- [10] G. Jocher, "YOLOv5 by Ultralytics," May 2020. [Online]. Available: <https://github.com/ultralytics/yolov5>
- [11] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>
- [12] G. Jocher and J. Qiu, "Ultralytics yolo11," 2024. [Online]. Available: <https://github.com/ultralytics/ultralytics>
- [13] R. Khanam and M. Hussain, "What is yolov5: A deep look into the internal features of the popular object detector," 2024. [Online]. Available: <https://arxiv.org/abs/2407.20892>
- [14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018. [Online]. Available: <https://arxiv.org/abs/1708.02002>
- [15] T. Liu, Y. Liu, Z. Ma, T. Yang, X. Liu, T. Li, and J. Ma, "L-hawk: A controllable physical adversarial patch against a long-distance target," in *NDSS*, 2025.
- [16] S. Moon, G. An, and H. O. Song, "Parsimonious black-box adversarial attacks via efficient combinatorial optimization," in *International conference on machine learning*. PMLR, 2019, pp. 4636–4645.
- [17] 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*, vol. 39, no. 6, pp. 1137–1149, 2016.
- [18] Y. Ren, Q. Jiang, C. Yan, X. Ji, and W. Xu, "Ghostshot: Manipulating the image of ccd cameras with electromagnetic interference."
- [19] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.
- [20] R. Suri, "Is it okay to convert an image with an alpha channel to an image without an alpha channel?" <https://stackoverflow.com/questions/37877020/is-it-okay-to-convert-an-image-with-an-alpha-channel-to-an-image-without-an-alpha>, Jun. 2016, [Online; accessed 30-July-2025].
- [21] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican *et al.*, "Gemini: a family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023.
- [22] S. Technologies, "Starship: Our robots," <https://www.starship.xyz/our-robots/>, 2014, accessed: 2025-07-25.
- [23] Tesla, "Autopilot and full self-driving (supervised)," <https://www.tesla.com/support/autopilot>, 2022, accessed: 2025-07-25.
- [24] G. Wang, C. Zhou, Y. Wang, B. Chen, H. Guo, and Q. Yan, "Beyond boundaries: A comprehensive survey of transferable attacks on ai systems," *arXiv preprint arXiv:2311.11796*, 2023.
- [25] Z. Weng, X. Yang, A. Li, Z. Wu, and Y.-G. Jiang, "Open-vclip: Transforming clip to an open-vocabulary video model via interpolated weight optimization," in *International conference on machine learning*. PMLR, 2023, pp. 36978–36989.
- [26] Q. Xia and Q. Chen, "Alphadog: No-box camouflage attacks via alpha channel oversight," in *NDSS*, 2025.
- [27] C. Zhou, Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He *et al.*, "A comprehensive survey on pretrained foundation models: A history from bert to chatgpt," *International Journal of Machine Learning and Cybernetics*, pp. 1–65, 2024.
- [28] C. Zhou, Q. Yan, D. Kent, G. Wang, Z. Zhang, and H. Radha, "Optical lens attack on deep learning based monocular depth estimation," *arXiv preprint arXiv:2409.17376*, 2024.
- [29] C. Zhou, Q. Yan, Y. Shi, and L. Sun, "Doublestar: Long-range attack towards depth estimation based obstacle avoidance in autonomous systems," in *31st USENIX security symposium (USENIX Security 22)*, 2022, pp. 1885–1902.
- [30] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.