Adversarial AI image perturbation attack invariant to object scale and type

Michel van Lier, Richard J.M. den Hollander, Hugo J. Kuijf
TNO, the Hague, the Netherlands

ABSTRACT

Adversarial AI technologies can be used to make AI-based object detection in images malfunction. Evasion attacks make perturbations to the input images that can be unnoticeable to the human eye and exploit weaknesses in object detectors to prevent detection. However, evasion attacks have weaknesses themselves and can be sensitive to any apparent object type, orientation, positioning, and scale. This work will evaluate the performance of a white-box evasion attack and its robustness for these factors.

Video data from the ATR Algorithm Development Image Database is used, containing military and civilian vehicles at different ranges (1000-5000 m). A white-box evasion attack (adversarial objectness gradient) was trained to disrupt a YOLOv3 vehicles detector previously trained on this dataset. Several experiments were performed to assess whether the attack successfully prevented vehicle detection at different ranges. Results show that for an evasion attack trained on object at only 1500 m range and applied to all other ranges, the median mAP reduction is >95%. Similarly, when trained only on two vehicles and applied on all seven remaining vehicles, the median mAP reduction is >95%.

This means that evasion attacks can succeed with limited training data for multiple ranges and vehicles. Although a (perfect-knowledge) white-box evasion attack is a worst-case scenario in which a system is fully compromised, and its inner workings are known to an adversary, this work may serve as a basis for research into robustness and designing AI-based object detectors resilient to these attacks.

Keywords: adversarial AI, object detection, objectness gradient attack, evasion attack, universal adversarial perturbation, cyber attack

1. INTRODUCTION

Deep learning-based object detection methods are increasingly deployed in military systems to enhance situational awareness and operational efficiency. They can automate tasks or offer support for humans in the loop [1], because continuous and manual searching for possible targets in camera streams is a tedious and exhausting task for the operator. Some threats can even be barely noticeable to the human eye, because of their small size or camouflage [2]. Furthermore, there is a limited number of human operators, multiple concurrent tasks should be executed, while multiple camera streams are continuously interpreted.

However, the use of deep learning or artificial intelligence (AI)-based detection methods brings potential risks. Adversarial attacks can be used to manipulate AI-based object detectors, so they overlook or misclassify critical objects or generate ghost attacks, potentially endangering crew and the success of the mission. Given the presence of human operators in the loop, these adversarial attacks must be effective while being visually imperceptible, ensuring that the operator remains unaware of any attack.

Attacks using adversarial AI exploit the well-known neural networks' weakness where (small) alterations to an input image can result in highly confident incorrect answers. [3] [4]. Common examples are specifically generated patterns that are added to the images in order to mislead image classification methods [5] or patches placed onto objects to thwart detection methods [6] [7]. These additive patches mislead AI-based methods, but are oftentimes clearly visible to the naked eye. On the contrary, so-called universal adversarial perturbation methods generate a pattern that is overlayed on the full image and is nearly invisible to the naked eye [8] [9] [10].

Adversarial attacks can be targeted or untargeted [11]. In targeted attacks, the attacker aims to force the model to predict a specific incorrect prediction. Some examples of adversarial object detection make objects vanish, fabricate more objects, mis-localize objects, or mislabel some or all objects [12]. In untargeted attacks, the goal is to cause the model to make any

Artificial Intelligence for Security and Defence Applications II, edited by Henri Bouma, Radhakrishna Prabhu, Yitzhak Yitzhaky, Hugo J. Kuijf, Proc. of SPIE Vol. 13206, 132060Y · © 2024 SPIE
0277-786X · doi: 10.1117/12.3031545

incorrect prediction without controlling what the incorrect output is. These attacks on deep learning object detectors can be achieved in multiple ways [13]. For example, modifying the input data, such as the frames used at inference time, often through (subtle) perturbations in the input frames. Poisoning attacks [14] [15] occur during the training phase, where malicious data is injected into the training set to corrupt the model's learning process and degrade its performance. Physical attacks [16] [7] involve creating adversarial perturbations on real-world objects, such as altering a stop sign or packaging label, to fool models when they encounter these objects in physical environments. Attacks can be created in an one-shot or iterative way. One-shot adversarial attacks involve creating adversarial attacks in a single attempt. They are fast and require less computational power, but may be less effective against robust models. Iterative adversarial attacks refine the adversarial perturbation over multiple steps, leading to more successful attacks. They require more computational resources and time but are often more effective, especially when the attacker has detailed knowledge of the model.

Based on the attacker's knowledge of the target model, adversarial attacks can be classified into white, black, and grey-box attacks. White-box attacks assume full access to the model's architecture, parameters, and gradients, allowing effective adversarial attacks by exploiting detailed internal knowledge. Black-box attacks operate without access to the model's internals, relying on input-output queries to infer the model's behaviour, making them more challenging and usually less effect than white-box attacks. Grey-box attacks are categorized in between these two, where the attacker has partial knowledge of the model.

This paper focuses on performance and sensitivity evaluation of a white-box iterative adversarial evasion targeted attack [12]. To perform such an attack, the attacker should have full access to the model's architecture, parameters, and gradients. This level of access enables the attacker to generate attacks to exploit the well-known weakness in neural networks by (small) alterations to an input image that can effectively fool the model. For example, such access can be gained through an adversary insider during partner collaborations, a man-in-the-middle attack during distributed/federated machine learning, cyber theft, or by reverse engineering from obtained hardware. Physical access cyber hacks are not uncommon [17] and could potentially happen unnoticed. Exploitation of open-source models can result in access to the full model. While exploiting open source models lowers the bar for attack creation, the effectiveness is unknown because of the many types of open source models. While gaining access to a full model poses a challenge, the deployment of the attack poses a subsequent challenge. To deploy such created attacks, the attacker does not need access to the full model, but only to the input data of the model (e.g. the camera hardware or camera output stream). The attack should be deployed unnoticeably during the development or deployment [1] phase of the model, by a cyber-attack during these phases (e.g. during over-the-air updates), exploiting supply chain vulnerabilities, or by physical access to the system (e.g. during maintenance).

Military systems are designed with layered security, including encryption, access controls, and physical security measures, making it difficult for attackers to gain the necessary access for crafting and deploying attacks. While the probability of successfully carrying out a white-box attack is generally low because of the stringent security measures, the consequences of such an attack could be severe, making it an attractive target.

The previously introduced perturbation-based evasion attacks exploit weaknesses so that neural networks provide incorrect results [18], but they have some weaknesses of their own. They are sensitive to the apparent object type, its orientation, position, and scale within the image. In addition, the perturbation-based methods are commonly applied to image classification, and are understudied for object detection.

This work will evaluate the performance and sensitivity of a universal white-box perturbation-based evasion vanishing attack for object detection. The aim is to demonstrate invariance to object type, orientation, position, scale, time of day and sensors with one universal patch that is not visible to the human eye, such that an attacked object detector will detect no object. We will specifically look into an adversarial objectness gradient attack [12] to evade detections of military and civilian vehicles in both visual and mid-wave infrared (MWIR). To the best of our knowledge, evaluating the performance of a universal patch that is invisible to the human eye while being invariant to object type, orientation, position, scale, time of day, and sensors in a military scenario has so far not been explored. Understanding the possibilities, performance and sensitivity of such an attack is important. It can help design attack detection techniques, object detectors robust to adversarial attacks, and reduce the security risks of object detectors in military systems and operations.

2. METHODS

A set of experiments is performed to test if an unnoticeable universal adversarial perturbation patch (to employ an evasion vanishing attack) can be invariant to object type, orientation, position, scale, time of day and sensors in a military setting. To test the trade-off between visibility and effectiveness, patches with increasing L_∞ norms are trained and evaluated

against each other to determine the trade-off between visibility and effectiveness. Then, the resulting L_∞ norm is used in the subsequent experiments. To test the invariance against object distance and orientation, the patch is trained on one distance while evaluated on the others. Furthermore, to test invariance across object type and orientation, the patch is trained using two vehicle types and evaluated on the others. This was performed for daytime and nighttime imaging, with two sensor types (visible, MWIR), resulting in three experiments (visible day, MWIR day, MWIR night). To test for invariance to time of day and sensor types, each patch is trained on one time of day and sensor combination (visible day, MWIR day, MWIR night) and evaluated based on the other two.

We will use the public ATR (automatic target recognition) Algorithm Development Image Database [19], containing videos of military and civilian vehicles that are manually annotated. The adversarial AI technology is based on the published framework of the adversarial objectness gradient attack [12]. It first trains an object detector based on YOLOv3 [20] model to perform vehicle detection. Next, based on the trained YOLOv3 object detector, a universal adversarial perturbation patch is trained that will be overlayed on the video frames in the dataset to create the desired evasion attack. This approach is summarised in Figure 1. Finally, multiple experiments and evaluations are performed to assess its final performance.

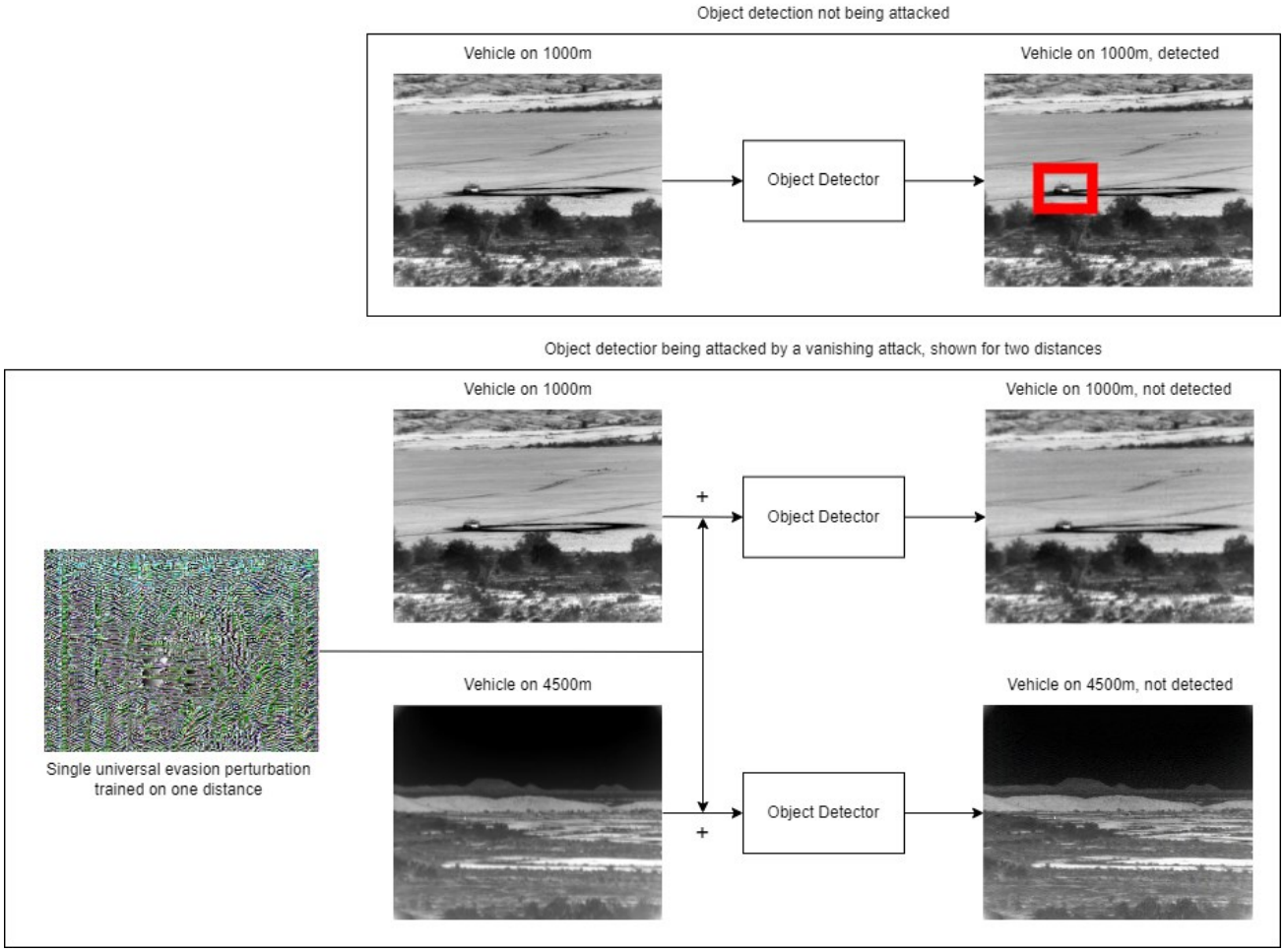


Figure 1. Overview of the universal evasion attack. Each row contains a YOLOv3 object detector that has been trained on a subset of the data to perform vehicle detection. Top row: A YOLOv3 object detector is not attacked and is able to detect a vehicle. Left: Based on the trained object detector, a single universal adversarial perturbation patch is trained (for visualization purposes, its visibility is increased). Bottom row: An adversarial evasion vanishing attack is performed, where the universal adversarial perturbation patch is overlayed on the video frames, the previously successful object detector (top row) no longer detects the vehicles regardless of distance.

Data

We used data from the ATR (automatic target recognition) Algorithm Development Image Database [19], to create and evaluate our method. This dataset contains both visual and MWIR videos of military and civilian vehicles, namely:

- a 2S3 (a Soviet self-propelled gun),
- a BMP-2 (Boyevaya Mashina Pekhoty, a Soviet amphibious infantry fighting vehicle),
- a BTR-70 (bronetransportyor, a Soviet eight-wheeled armoured personnel carrier),
- a BRDM-2 (Boyevaya Razvedyvatelnaya Dozornaya Mashina, a Soviet amphibious armoured scout car),
- a MT-LB (Mnogotselevoy tyagach legky bronirovanny, a Soviet multi-purpose, fully amphibious, tracked armoured fighting vehicle) that was towing a D-20 (a Soviet 152 mm gun-howitzer artillery piece),
- a T-72 (a Soviet main battle tank),
- a ZSU-23-4 (Zenitnaya Samokhodnaya Ustanovka, a Soviet lightly armoured, self-propelled, radar-guided anti-aircraft weapon system),
- a pickup truck, and
- an SUV (sport utility vehicle).

Each vehicle is recorded driving a circle with a diameter of 100 meters, at approximately 16 km/h, at different distances from the cameras: from 1000 m to 5000 m, with 500 m range steps (distances 1000 – 4500 m are used in this work). This was performed during daylight conditions (visible light and MWIR imaging) and night (MWIR imaging). Each video lasts about one minute at 30 frames per second. Some example images are shown in Figure 2.

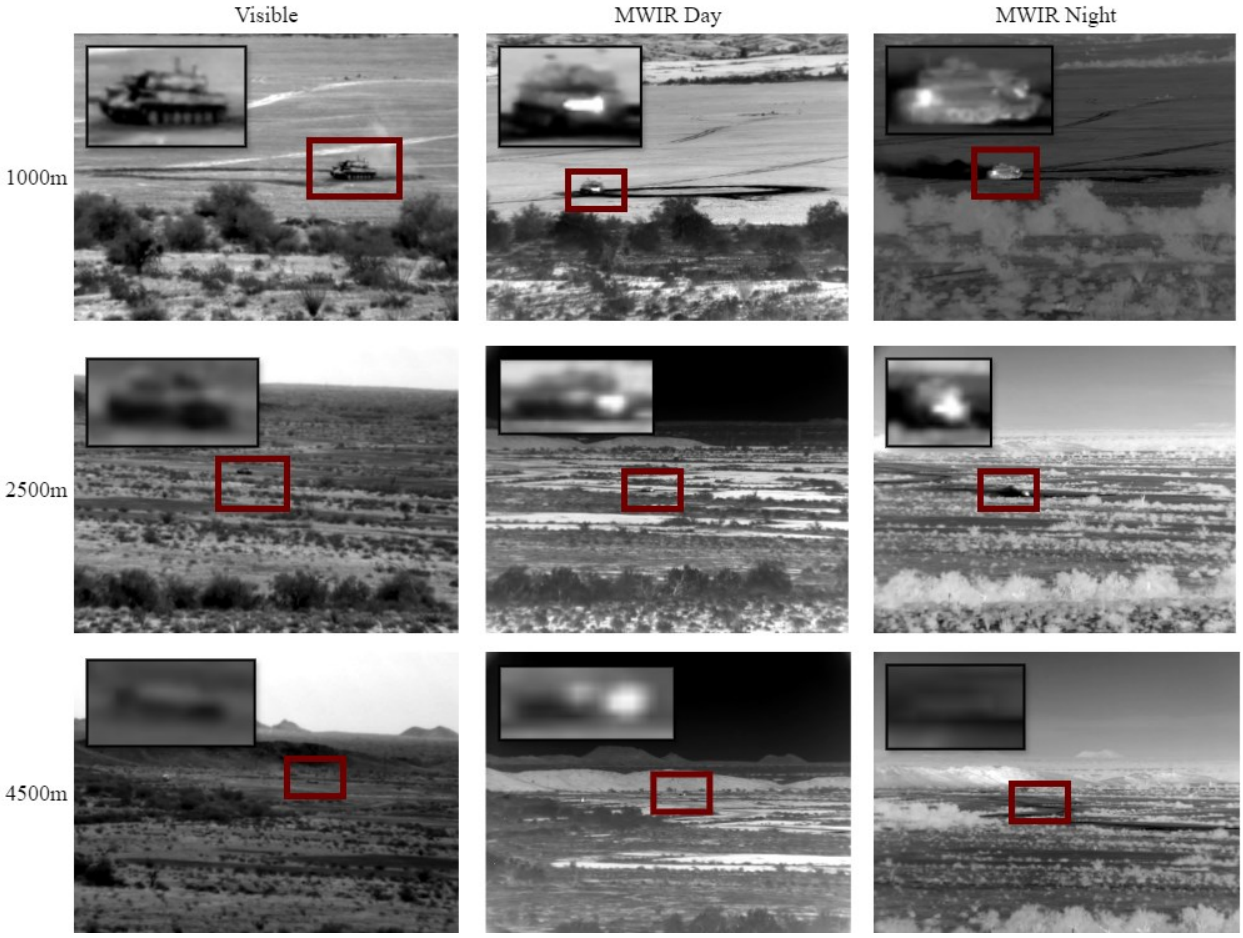


Figure 2: Example video frames from the ATR dataset showing the ZSU-23-4 vehicle at three different distances of 1000, 2500, and 4500 meters (row) in three different recordings: visible light, MWIR daytime, and MWIR nighttime (columns).

Manual annotations of the vehicles were provided as a single point in the centre of the vehicle (X_{obj}, Y_{obj}) in pixel coordinates for each video frame. In addition, for the MWIR imaging, pixel coordinates of manually drawn bounding boxes were provided as the upper left coordinate and width and height $(X_{bbox}, Y_{bbox}, W_{bbox}, H_{bbox})$ in pixels.

Bounding boxes for the visual imaging were computed using additional parameters provided in the dataset. For every video frame, the distance d_{obj} between the vehicle and the camera is provided in metres, as well as the aspect α in degrees that define the angle between the object and the sensor head with clockwise rotation. Using these values, the known physical vehicle dimensions $(W_{obj}, H_{obj}, L_{obj})$, in metres, the camera field-of-view (FoV), and the image size (X_{img}, Y_{img}),in pixels, the bounding boxes for visual imaging can be estimated as follows:

$$W_{bbox} = \frac{f_x * \left(|\sin\alpha| * L_{obj} + |\cos\alpha| * W_{obj}\right)}{d_{obj}} \tag{1}$$

$$H_{bbox} = \frac{f_y * H_{obj}}{d_{obj}} \tag{2}$$

$$X_{bbox} = X_{obj} - \frac{1}{2} * W_{bbox} \tag{3}$$

$$Y_{bbox} = Y_{obj} - \frac{1}{2} * H_{bbox} \tag{4}$$

Where focal length f_x, f_y can be estimated as follows:

$$f_x = \frac{\frac{1}{2} * IMG_w}{\tan\left(FoV_w * \frac{1}{2}\right)} \tag{5}$$

$$f_y = \frac{\frac{1}{2} * IMG_h}{\tan\left(FoV_h * \frac{1}{2}\right)} \tag{6}$$

Object detection

The YOLOv3 object detector with the Darknet-53 backbone was used to detect the vehicles in the dataset [20], and is the detector that will be attacked. Depending on the type of experiments (see below), different subsets of the dataset were selected to train the YOLOv3 object detector. The weights pre-trained on MSCOCO [21], were used as a starting point for fine-tuning on a subset of the dataset, with a resolution of 800×800 pixels, the grey images duplicated over the three channels of YOLOv3 and a learning rate of 1×10^{-3}. Fine-tuning was performed in two steps: 50 epochs with a frozen backbone and a batch size of 16, followed by 50 epochs with an unfrozen backbone and a batch size of two. The selected subset was split into 90% for training and 10% for validation. Individual frames from the videos were sampled linearly at every 15th frame (i.e. every 0.5 seconds) because (directly) neighbouring frames only show minor differences and do not contain new information for the YOLOv3 object detector. Ultimately, the selected training dataset equally represented every vehicle class and distance to the camera. Only two default YOLOv3 data augmentation operations were used during training: random left-right flips (50%) and random shift (50%, shift range 0 – 10%). Other colour augmentations were discharged since the ART dataset is a grey-scale dataset. Random brightness and contract modifications (15%) and Gaussian noise (5%, with $\sigma^2 = 15$ and $\mu = 0$) were added to accommodate the brightness, contrast and noise differences due to zooming and scene differences. Perspective (10%) and affine (10%) transforms were used to accommodate a greater variety of visual presentations, which can be limited due to sampling. Additionally, random zooming was added by randomly resizing the image between 75% – 150% to accommodate for the different distances in the dataset [22].

Adversarial attack

The Targeted adversarial Objectness Gradient (TOG) framework was used to create universal adversarial perturbation patches [12] for the experiments to perform a targeted vanishing attack. This is a targeted white-box evasion vanishing attack, where the patch is trained against the YOLOv3 object detector to create a vanishing evasion attack. A vanishing attack is a specifically targeted attack with the goal of fooling the object detector in such a way that no objects are detected. Object detectors such as YOLOv3 are producing final detection result $\hat{O}(x)$ after processing input image x, which constitute a set of bounding boxes. Bounding boxes include the coordinates of the object (x, y, width, height) were pixel coordinates refer to the centre of the object with their objectness score (probability of being an object 0–1) and the probability assigned to a class. A vanishing patch n is generated by perturbing an input x sent to the detector, which results in a perturbed input image $\acute{x}$. The generation process of the adversarial example can be formulated as follows:

$$min \, \|\acute{x} - x\|_{L_\infty} \, \text{s.t.} \, \hat{O}(\acute{x}) = \emptyset \tag{7}$$

Where the L_∞ norm is denoting the maximum change to any pixel and $\emptyset$ denotes the empty target detections. L_∞ norm can be used to control the visibility of the patch (lower values result in a less visible patch) and higher effectiveness (higher values result in a more effective patch). The input image $\acute{x}$ can be formulated as follows:

$$\acute{x} = x + n \tag{8}$$

Training a deep neural network often starts with random initialization of model weights, which will be updated slowly by taking the derivative of the loss function L, regarding the learnable model weights W and batch of input-output pairs $\{(x, O)\}$. Intuitively speaking, the TOG reverts the training process of YOLOv3. During the training of the patch, the model weights of YOLOv3 are fixed while iteratively updating each iteration t, the input image $\acute{x}$ towards the goal in equation 8. This can be formulated as follows:

$$\acute{x}_{t+1} = \prod_{x,\epsilon} \left[\acute{x}_t - \alpha \Gamma \left(\frac{\partial L(\acute{x}_t; \, \emptyset, W)}{\partial \acute{x}_t} \right) \right] \tag{9}$$

where $\prod_{x,e}[...]$ is the projection onto a hypersphere with a radius ε centred at x in L_∞ norm, α is the learning rate, controlling the step size of the patch update, Γ is a sign function, and L defines the YOLOv3 loss function that is optimized during the attack. The training process starts at $t = 0$, where patch n_0 is randomly initialized in the range of $[-\epsilon, \epsilon]$. During patch training x is sampled from a set of training images D.

Depending on the type of experiment (see below), different subsets D of the dataset were used to train the patch n. The selected subset was split into 90% for training and 10% for validation. Individual video frames were sampled linearly at every 5th frame, resulting in approximately 360 sampled frames per video. For every experiment, the patch was trained for 24 epochs with an L_∞ norm of 8/255 on a [0,1] intensity range and a learning rate of 1×10^{-4}. The L_∞ norm controls the degree to which pixel intensity values can be altered, which is a trade-off between possible visibility of the patch versus attack performance. The trained patch included three channels since YOLOv3 has three channels as input. The same augmentations were applied to training YOLOv3.

Experiments

All experiments will be evaluated using the mean average precision (mAP) metric for object detection [21], with an intersection-over-union (IoU) of 50%. For evaluation, all 1800 frames of every video were used. Ultimately, the goal of the universal adversarial perturbation patch for the YOLOv3 object detector is to create a single perturbation that, when applied to various images, consistently reduces mAP by impairing its ability to detect and classify objects accurately.

First, a suitable L_∞ norm was empirically determined by evaluating values in the 4/255 – 11/255 range. The YOLOv3 object detector was trained on the MWIR daytime images, and next, a patch was trained for the various L_∞ norm values. Quantitative measures of recall, precision, and F1 were computed, and a qualitative visual assessment was performed to determine the visibility of the patches. One L_∞ norm value was selected for the subsequent experiments.

Next, the baseline performance of the YOLOv3 method for detecting vehicles in MWIR daytime videos was assessed. Two universal adversarial perturbation patches were trained: (i) one that was trained only on vehicles at 1500 m distance and next applied to all remaining distances, and (ii) one that was trained on two vehicle types (BMP2 and SUV) and next applied to all the remaining vehicles types. This was to confirm that the patches are indeed universal and can be applied to

all distances/vehicles whilst only trained on a selected subset. The performance of the attacks was assessed by measuring the reduction in mAP.

To verify that the attack was caused by the patch itself and not just any noise pattern, it was compared to two random patches: one generated by uniform noise with an L_∞ norm of 8/255 and another by random shuffling the original patch.

Next, the experiment with a universal adversarial perturbation patch trained on MWIR daytime imaging of two vehicles was extended to the other two sensors in the dataset: visible light and MWIR nighttime imaging. A separate YOLOv3 object detector was trained for both sensors, keeping all hyperparameters the same, excluding the 4500 m distance. For each trained YOLOv3 object detector, a patch was trained on two vehicles (BMP2 and SUV) and next evaluated on all remaining vehicles. Again, the reduction in mAP was assessed.

Finally, to assess whether the trained adversarial perturbation patches were truly universal, the patch trained on one dataset (e.g. MWIR daytime) was applied to the two other datasets (visible light and MWIR nighttime). This was repeated for the other combinations, and the reduction in mAP was evaluated.

3. RESULTS

An L_∞ norm value of 8/255 was selected based on the results in Table 1 and visual inspection of the trained adversarial perturbation patch (see Figure 3). Table 1 shows the baseline results of the YOLOv3 object detector on the MWIR daytime images (bottom row), which achieves a mAP@0.5 of 0.59 and an F1 of 0.71, where lower is better since it denotes the patch's effectiveness. L_∞ norm values of 8/255 and larger can reduce the mAP to below 0.05 and the F1 below 0.10. Since mAP is a holistic measure, precision and recall are evaluated to provide a more fine-grained performance insight. As an example, a patch with L_∞ norm values of 6/255 have low mAP@0.5 of 0.16, but on average, 37% of the objects are detected in the dataset with a precision of 67%, which is relatively high. The reduction in detection accuracy was then balanced against the visibility of the patch, which becomes more visible for L_∞ norm values of 9/255 and larger.

The baseline performance of YOLOv3 for detecting vehicles in MWIR daytime imaging has a mAP@0.50 of 0.55 with σ = 0.26 (mean σ across the different distances), which is provided in Table 2. When training a universal adversarial perturbation patch on the vehicles at 1500 m and applying it to the videos, the detection performance decreases by 89% to an mAP of 0.07±0.07 across all distances. When applying a shuffled or random noise patch, there is no considerable decrease in performance, suggesting that the trained patch is indeed effective. These results are summarised in Table 2, and performance metrics per distance are provided in Table 3. The patches are shown in Figure 4.

Table 1. Quantitative results for determining a suitable L_∞ norm value for creating a universal adversarial perturbation patch. The bottom row (no patch) shows the detection results of YOLOv3 trained on MWIR daytime videos. L_∞ norm values range from 4/255 – 11/255 (less to more visible).

L_∞ norm patch	mAP (@0.5)	F1	Precision	Recall
11/255	0.00 +/- 0.01	0.01 +/- 0.03	0.02 +/- 0.07	0.02 +/- 0.02
10/255	0.01 +/- 0.03	0.06 +/- 0.06	0.34 +/- 0.34	0.03 +/- 0.04
9/255	0.03 +/- 0.03	0.09 +/- 0.08	0.27 +/- 0.21	0.06 +/- 0.06
8/255	0.03 +/- 0.02	0.05 +/- 0.05	0.09 +/- 0.08	0.04 +/- 0.04
7/255	0.04 +/- 0.04	0.26 +/- 0.11	0.40 +/- 0.14	0.21 +/- 0.09
6/255	0.16 +/- 0.12	0.46 +/- 0.17	0.67 +/- 0.19	0.37 +/- 0.14
5/255	0.18 +/- 0.11	0.15 +/- 0.16	0.17 +/- 0.16	0.14 +/- 0.14
4/255	0.39 +/- 0.21	0.77 +/- 0.20	0.85 +/- 0.18	0.72 +/- 0.21
No patch	0.59 +/- 0.27	0.71 +/- 0.22	0.72 +/- 0.18	0.70 +/- 0.25

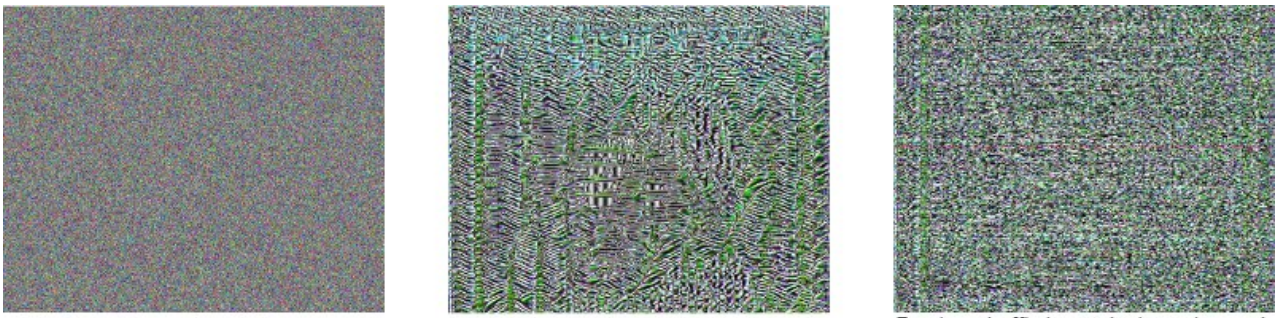
Figure 3. An example of a BRM-2 on the MWIR daytime imaging at a distance of 1500 m. Left: the original image is without any adversarial perturbation patch applied. Middle: patch with L_∞ norm of $\frac{8}{255}$ applied, which is visually not very noticeable. Right: patch with L_∞ norm of $\frac{11}{255}$ applied, which is somewhat noticeable visually. The raw patch itself is provided in Figure 1. Images are best viewed on a computer screen after zooming in.

Table 2. The performance of a YOLOv3 object detector trained for detecting vehicles in MWIR daytime imaging. When applying a universal adversarial perturbation patch, object detection accuracy decreases considerably. Applying shuffled or random noise patches does not show a considerable decrease in object detection accuracy.

Distance	**No patch**	**With patch**	**Shuffled patch**	**Random patch**
Mean mAP (@0.50) σ	0.55 σ = 0.26	0.07 σ = 0.07	0.46 σ = 0.25	0.55 σ = 0.24
Reduction (%) σ		88.60 σ = 8.27	16.14 σ = 3.34	0.85 σ = 6.03

Table 3. Detailed performance results of a YOLOv3 object detector trained for detecting vehicles in the MWIR daytime imaging. The bottom row shows the mean results, which are also presented in Table 2. The distance of 1500 m was used for training and thus not shown.

Distance (m)	**mAP (@0.5)**	**mAP (@0.5), with patch**	**Reduction (%)**
1000	0.86	0.09	89.94
2000	0.81	0.05	93.56
2500	0.69	0.15	77.82
3000	0.48	0.08	83.85
3500	0.50	0.08	83.77
4000	0.40	0.01	98.60
4500	0.10	0.00	100.00
Mean σ	**0.55** σ = 0.26	**0.07** σ = 0.07	**88.60** σ = 8.27



Random noise patch with L∞ norm of 8/255

Perturbation patch with L∞ norm of 8/255

Random shuffled perturbation noise patch with L∞ norm of 8/255

Figure 4. The universal adversarial perturbation patch is shown in the middle. On the left, a patch with uniform random noise with the same L_∞ norm. On the right, the values from the trained patch are randomly shuffled to create the shuffled patch. All the patches have an L_∞ norm, but for visibility, they are normalized between [0, 255].

Table 4. Detailed performance results of a YOLOv3 object detector trained for detecting vehicles in the MWIR daytime imaging. The bottom row shows the mean detection performance. The vehicles BMP2 and SUV were used for training and thus not shown.

Class	**mAP (@0.5)**	**mAP (@0.5) with patch**	**Reduction (%)**
2S3	0.83	0.00	99.97
BRDM2	0.78	0.04	94.66
BTR70	0.76	0.05	93.56
MTLB	0.11	0.00	100.00
PICKUP	0.37	0.01	96.76
T72	0.79	0.00	100.00
ZSU23-4	0.71	0.02	97.13
Mean σ	**0.62** $\sigma = 0.28$	**0.02** $\sigma = 0.28$	**97.44** $\sigma = 0.28$

The universal adversarial perturbation patch was also trained on two types of vehicles, BMP2 and SUV, to cover both a military with a barrel and civilian vehicle, and applied to all remaining vehicles. These results are shown in Table 4, which indicates that the mAP can be reduced by more than 90% for all other vehicle types.

The experiments for Table 4 were repeated for the visible light and MWIR nighttime datasets. These results are shown in Tables 5 and 6. A similar trend can be observed for the MWIR daytime imaging, although the reduction in mAP is somewhat less. For visible light imaging, the baseline performance of YOLOv3 (without a patch applied) is somewhat higher.

Table 5. Detailed performance results of a YOLOv3 object detector trained for detecting vehicles in the MWIR nighttime imaging. The bottom row shows the mean detection performance. The vehicles BMP2 and SUV were used for training and thus not shown.

Class	mAP (@0.5)	mAP (@0.5), with patch	Reduction (%)
2S3	0.99	0.19	80.44
BRDM2	0.32	0.00	98.92
BTR70	0.82	0.10	88.03
MTLB	0.25	0.01	95.33
PICKUP	0.51	0.03	94.65
T72	0.79	0.09	89.29
ZSU23-4	0.74	0.16	78.18
Mean σ	**0.63** $\sigma = 0.28$	**0.08** $\sigma = 0.07$	**86.88** $\sigma = 7.76$

Table 6. Detailed performance results of a YOLOv3 object detector trained for detecting vehicles in visible light imaging. The bottom row shows the mean detection performance. The vehicles BMP2 and SUV were used for training and thus not shown.

Class	mAP	mAP (@0.5), w patch	Reduction (%)
2S3	0.95	0.33	65.19
BRDM2	0.76	0.24	68.40
BTR70	0.92	0.41	54.90
MTLB	0.07	0.01	79.18
PICKUP	0.52	0.02	95.90
T72	0.93	0.20	78.14
ZSU23-4	0.75	0.11	84.94
Mean σ	**0.70** $\sigma = 0.32$	**0.19** $\sigma = 0.15$	**75.23** $\sigma = 13.58$

The adversarial perturbation patch trained on either MWIR daytime, was applied to visible light and MWIR nighttime, while MWIR nighttime was applied to visible light and MWIR daytime, and the performances were evaluated against the patches that were trained and evaluated on the same sensor and time of day combination. When the patch trained on MWIR daytime was applied to the MWIR nighttime dataset, its effectiveness was considerably less, and the mean mAP reduction was only 39% across all vehicle types, compared to applying on the same detector and daytime as the patch was trained on (where the patch reduces the mAP up to 97%). Similarly, when the MWIR daytime patch was applied to the visible light dataset, the mean mAP reduction was only 29% across all vehicle types. Almost similar results are seen when the MWIR nighttime patch is applied to MWIR daytime a mean mAP reduction of 42% or visible light with mean mAP reduction of 18%. Finally, the patch trained on visible light imaging is even less effective on the other two datasets: MWIR daytime reduction is 7%, and MWIR nighttime reduction is 2%.

4. DISCUSSION AND CONCLUSION

A white-box evasion vanishing attack for vehicle detection was evaluated in order to test its invariance for object type, orientation, position, scale, time of day and sensor type. A YOLOv3 model trained on the ART dataset was attacked by creating patches with the TOG framework. To increase the robustness of its invariance to object type, orientation, position, scale and time of day, the same data augmentation was used for training the object detector. A subset of the ART dataset was used to evaluate the level of invariance.

To create an unnoticeable patch to the human eye, but still effective enough to vanish objects, a range of L_∞ norms were explored, and the 8/255 threshold seemed a good trade-off between visibility and effectiveness. While unnoticeable to untrained people, it could be possible that trained operators will notice the patch because of the subtle changes. In addition,

this patch introduces a high-frequency component into the image frame. This could be absent in normal recording conditions because of camera optics and potentially allows detection by the operators. The high-frequency component can result in a larger file size when applying (lossless) compression or can be noticeable when inspecting the Fourier domain, which could be used to detect an attack.

To test the invariant against vehicle distances, orientation and scale, a patch was trained on one distance but applied to multiple distances. We found that one universal unnoticeable patch can be made and reduces the mAP by 88% on average, which demonstrates its invariance between 1000 – 4500 m distance. While not evaluated on the distances below 1000 m or above 4500 m, we expect that the patch is still effective on these distances. Furthermore, when testing for invariance to object type, orientation and scale; a patch was trained on two vehicle types and tested on the remaining seven. We found that such a universal patch can be effective on visible daytime, MWIR daytime, and MWIR nighttime recordings. It reduces the mAP on average by 90%, which shows that such a patch can be invariant to object type orientation and scale. The dataset does not contain all times of the day or weather conditions, such as rain or haze, and the patch cannot guaranteed to be effective in these conditions. While this patch is effective, it is unknown how effective such a universal patch is on state-of-the-art object detectors that achieve higher mAPs or spatiotemporal detectors that achieve high mAPs for small objects. Furthermore, it should be noted that such a patch cannot be applied to systems that are not deep learning-based, such as moving object detectors.

The vanishing attack means that no objects are detected at all. If, for example, a detector is trained to also detect common objects, such as people or vehicles, not detecting them could easily raise suspicion. Therefore, creating attacks on only specific object types could be important and a future step.

The results indicate that an effective, unnoticeable vanishing evasion attack can be performed by training the patch with a limited amount of data: only from 1 out of 8 distances or 2 out of 9 vehicle types. This suggests that the resulting patches are indeed universally applicable. However, when a patch trained for a specific camera (e.g., visible light) is applied to data from another camera, the evasion attack is no longer successful. Since cyber hacks are not uncommon and can be performed unnoticed, such effective universal white-box attacks can be created. When deployed, they can be used to fool automated object detection and let objects vanish, even for novel object types and distances that have not been seen by the patch training. This can result in various security risks in military systems. Detecting such attacks or making object detectors robust to such attacks is therefore important for future work.

REFERENCES

[1] A. Munir, A. Aved and #. Blasch, “Situational Awareness: Techniques, Challenges, and Prospects,” *AI,* vol. 3, no. https://doi.org/10.3390/ai3010005, pp. 55-77, 2022.

[2] J. Johnson, “Automating the OODA loop in the age of intelligent machines: reaffirming the role of humans in command-and-control decision-making in the digital age,” *Defence Studies,* vol. 23, no. 1, pp. 43-67, 2020.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow and R. Fergus, “Intriguing properties of neural networks,” 2014.

[4] I. Goodfellow, J. Shlens and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” 2015.

[5] T. B. Brown, D. Mané, A. Roy, M. Abadi and J. Gilmer, “Adversarial Patch,” 2017.

[6] M. Lee and Z. Kolter, “On Physical Adversarial Patches for Object Detection,” 2019.

[7] R. den Hollander, A. Adhikari, I. Tolios, M. van Bekkum, A. Bal, S. Hendriks, M. Kruithof, D. Gross, N. Jansen, G. Perez, K. Buurman and S. Raaijmakers, “Adversarial patch camouflage against aerial detection,” in *Proc. SPIE 11543, Artificial Intelligence and Machine Learning in Defense Applications II*, 2020.

[8] S.-M. Moosavi-Dezfooli, A. Fawzi and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[9] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi and P. Frossard, “Universal adversarial perturbations,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[10] H. Hirano and K. Takemoto, “Simple iterative method for generating targeted universal adversarial perturbations,” *Algorithms,* vol. 13, no. 11, p. 268, 2020.

[11] K. N. T. Nguyen, W. Zhang, K. Lu, Y. Wu, X. Zheng, H. L. Tan and L. Zhen, “A Survey and Evaluation of Adversarial Attacks for Object Detection,” 2024.

[12] K.-H. Chow, L. Liu, M. Loper, J. Bae, M. E. Gursoy, S. Truex, W. Wei and Y. Wu, “Adversarial Objectness Gradient Attacks in Real-time Object Detection Systems,” in *Second IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, 2020.

[13] B. Biggio and F. Roli, “Wild patterns: Ten years after the rise of adversarial machine learning,” *Pattern Recognition,* vol. 84, pp. 317-331, 2018.

[14] A. Schwarzschild, M. Goldblum, A. Gupta, J. P. Dickerson and T. Goldstein, “Just How Toxic is Data Poisoning? A Unified Benchmark for Backdoor and Data Poisoning Attacks,” *CoRR,* vol. abs/2006.12557, 2020.

[15] A. E. Cinà, K. Grosse, A. Demontis, B. Battista, R. Fabio and M. Pelillo, “Machine Learning Security against Data Poisoning: Are We There Yet?,” *Institute of Electrical and Electronics Engineers (IEEE),* vol. 57, no. 10.1109/mc.2023.3299572, pp. 26-34, 2024.

[16] Z. W, S.-N. Lim , L. Davis and T. Goldstein, “Making an Invisibility Cloak: Real World Adversarial Attacks on Object Detectors,” *European Computer Vision Association,* 2020.

[17] “Significant Cyber Incidents,” Center for strategic & international studies, July 2024. [Online]. Available: https://www.csis.org/programs/strategic-technologies-program/significant-cyber-incidents. [Accessed 28 08 2024].

[18] B. Biggio and F. Roli, “Wild patterns: Ten years after the rise of adversarial machine learning,” *Pattern Recognition,* vol. 84, pp. 317-331, 2018.

[19] Defense Systems Information Analysis Center (DSIAC), “ATR Algorithm Development Image Database,” [Online]. Available: https://dsiac.org/databases/atr-algorithm-development-image-database/.

[20] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” 2018.

[21] T.-Y. Lin, “Microsoft COCO: Common Objects in Context,” 2014.

[22] B. v. d. Hoogen, W. Uijens, R. den Hollander, W. Huizinga, J. Dijk and K. Schutte, “Long range person and vehicle detection,” in *roceedings of SPIE - The International Society for Optical Engineering, Artificial Intelligence and Machine Learning in Defense Applications II*, 2020.