Industrial Artificial Intelligence (2023) 1:1
https://doi.org/10.1007/s44244-023-00005-3

RESEARCH

On the robustness of vision transformers for in-flight monocular depth estimation

Simone Ercolino[1] · Alessio Devoto[1] · Luca Monorchio[2] · Matteo Santini[2] · Silvio Mazzaro[2] · Simone Scardapane[1]

Received: 26 September 2022 / Accepted: 29 January 2023

Abstract
Monocular depth estimation (MDE) has shown impressive performance recently, even in zero-shot or few-shot scenarios. In this paper, we consider the use of MDE on board low-altitude drone flights, which is required in a number of safety-critical and monitoring operations. In particular, we evaluate a state-of-the-art vision transformer (ViT) variant, pre-trained on a massive MDE dataset. We test it both in a zero-shot scenario and after fine-tuning on a dataset of flight records, and compare its performance to that of a classical fully convolutional network. In addition, we evaluate for the first time whether these models are susceptible to adversarial attacks, by optimizing a small adversarial patch that generalizes across scenarios. We investigate several variants of losses for this task, including weighted error losses in which we can customize the design of the patch to selectively decrease the performance of the model on a desired depth range. Overall, our results highlight that (a) ViTs can outperform convolutive models in this context after a proper fine-tuning, and (b) they appear to be more robust to adversarial attacks designed in the form of patches, which is a crucial property for this family of tasks.

Keywords Depth estimation · Vision transformer · Dense prediction · Adversarial attacks

1 Introduction

Given an RGB monocular image, monocular depth estimation (MDE) is the task of estimating the distance of each pixel of that image from the camera. We often refer to this distance as depth of the pixel and to the output of an MDE model as a depth map. The depth prediction task has traditionally been solved by leveraging deep convolutional neural networks (CNNs), both in a supervised [1–3] and self-supervised [4, 5] fashion. Popular datasets for this task are NYU-v2 [6] and KITTI [7], containing mainly indoor images, and Make3D [8] and Mid-Air [9], containing mainly outdoor images. Recently, large meta-datasets for self-supervised MDE learning have also been proposed by combining appropriately different datasets [10].

MDE finds application in a number of different use cases, ranging from autonomous vehicles and robotic systems to 3D architectural modeling and terrestrial surveys. Furthermore, it turns out to be of key importance to the flight of unmanned aerial vehicles (UAVs) [11–13], which often can only rely on a single front camera as the main sensing device [11]. Such cameras are used not only for collision avoidance—as the primary task—but also for other outdoor tasks such as object detection, 3D reconstruction and digital terrain model (DTM) generation [12]. As such, efficient depth estimation and drone flight has become essential in several industrial applications, including visual navigation in industrial platforms [14], aerial surveillance and safety [15], and aerial object detection and crowd counting [4]. Unfortunately, due to cost and design constraints, it is often impractical to equip UAVs with devices modern enough, able to directly provide in-depth maps. Installed cameras cannot sense distances, which is crucial to a large number

✉ Simone Scardapane
simone.scardapane@uniroma1.it

Alessio Devoto
alessio.devoto@uniroma1.it

Luca Monorchio
luca.monorchio@leonardo.com

Matteo Santini
matteo.santini@leonardo.com

Silvio Mazzaro
silvio.mazzaro@leonardo.com

[1] Sapienza University of Rome, Rome, Italy

[2] Leonardo S.p.A., Rome, Italy

Published online: 14 March 2023

of tasks, hence the need for estimating depth of objects from a single monocular camera.

Recent advances in the field of image processing [16] have paved the way to brand-new approaches to MDE, that make use of the vision transformer (ViT) architecture instead of plain CNNs to estimate the depth of a single pixel. Differently from a CNN, a ViT model makes use of multi-head attention (MHA) layers as its core component. In particular, in a ViT layer the original image is decomposed into a series of non-overlapping patches, and each block of the model alternates between in-patch operations and between-patch operations via the MHA mechanism, with the benefit of providing global receptive fields at every layer of the network thanks to the cross-attention mechanism. Due to this and to their scalability to ever-larger datasets, ViT are quickly becoming the state-of-the-art in computer vision when pre-trained on sufficiently diverse datasets [16], and they have been exploited successfully also for dense prediction tasks such as MDE.

A prominent example in this category is the dense prediction transformer (DPT) [17], an architecture which yields substantial improvements on dense prediction tasks. Just like other transformer-based architectures, it delivers optimal results especially when a large amount of training data is available, and can be fine-tuned on smaller datasets after an initial training. DPT is of particular interest for monocular depth estimation due to its strong pre-training stage executed on a dataset significantly larger than the previous state-of-the-art [17], that allows to obtain high zero-shot performance on smaller benchmarks. However, most of the results obtained up-to-now [17] refer to classical indoor datasets or outdoor datasets obtained by leveraging ground cameras (e.g., mounted on top of a moving car). In this paper, we evaluate instead whether ViT-based architectures for MDE can benefit also flight drone scenarios, by fine-tuning DPT on the Mid-Air dataset [9], a multi-purpose synthetic dataset for low-altitude drone flights, achieving state-of-the-art performances after the fine-tuning.

In a further analysis, we assess for the first time whether such transformer-based models for dense prediction are susceptible to adversarial attacks, which are already known to affect CNNs. Several works [18–20] bring evidence that CNN-based models are vulnerable to adversarial examples attacks can alter significantly the output of the model. Adversarial attacks can be obtained in two different ways: either as small perturbations added to each input image, in some cases not even visible to the human eye [18]; or in the form of more visible patches of various size and shape, that can be applied to several images at different locations, affecting the output of a model in a near-universal fashion [21]. An example of the latter case is found in [22], where the authors perform an attack against an image classifier, training and obtaining a small patch, which can then be placed anywhere within the field of view of the classifier, and causes the classifier to output a targeted class. An attack akin to this one, but targeting a convolutional model which performs MDE, is described in [23]. In general, a number of analogous attacks, coming in different flavors, are described in the literature. However, the impact of such patch-based attacks on transformers has not yet been extensively researched and remains mainly unexplored.

1.1 Contributions of this work

We provide two major contributions in this paper. First, we validate that fine-tuning DPT models can provide state-of-the-art performance in depth estimation from a drone flight, reaching or surpassing the accuracy of video models but starting from a single static frame. Second, we probe the hypothesis, rather widespread in the literature, that vision transformers are less vulnerable to adversarial attacks with respect to CNN-based models. To this end, we conduct a series of experiments, devising patch attacks against DPT and CNNs for depth estimation. Our results suggest that such attacks are still possible against DPT, yet less effective than against CNNs. We show that adversarial patches, trained in the same fashion against DPT and CNNs for MDE, yield higher error rates on the latter. Together, the combination of these two results point to the fact that ViT-based models such as the DPT are vastly superior to convolutional architectures, both in terms of accuracy and robustness, when considering UAVs and general flight scenarios.

1.2 Organization of the paper

The rest of the paper is organized as follows. In Sect. 2, we provide a brief overview of related works, concerning adversarial attacks for dense targets, and neural architectures for MDE. Next, in Sect. 3, we describe the DPT architecture, the Mid-Air dataset, and the design of our patch adversarial attack. We provide extensive experimental results in Sect. 4, before concluding and highlighting possible future works in Sect. 5.

2 Related works

2.1 Adversarial attacks for structured targets

In this section we focus on adversarial attacks and defenses for structured computer vision targets, i.e., object detection, semantic segmentation, and depth estimation. For a broader overview of the field, we refer the reader to a number of comprehensive surveys, including [24–26].

Despite several works which first explored the possibility of fooling a classifier applied to spam detection [27–29], the

renewed interest for adversarial attacks dates back to [30], where Szegedy et al. showed that CNNs for classification could be attacked by applying perturbations to input images and thus fooled into a misclassification. Over the past years, since this weakness was discovered, a wide range of different attacks have been introduced in the literature, targeting deep networks designed not only for classification, but also for object detection [31–34], and semantic segmentation [35].

Adversarial attacks are usually performed by computing an additive perturbation to the input image, obtained by solving a problem of loss maximization [24]. The resulting perturbed images can fool deep learning models into performing wrong predictions, even with a high level of accuracy, and in some cases can even be 'universal' [36], meaning they are able to fool a network on any image and they can transfer to different architectures, irrespective of their composition or their weights.

Perturbations can come in many forms, from simple noise added to the sample, to a single pixel [37] or a patch [20–22]. Whereas the former two are dangerous in that the perturbed image can look very similar to the original one (and changes might be imperceptible to the human eye), patch-based attacks represent a more critical threat because a trained patch can be printed and applied to real world images, thus allowing physical attacks [18, 19].

Even though the robustness of ViTs and its variants against adversarial attacks is still far from being widely investigated, recent works [38] show that transformers are slightly less vulnerable to adversarial attacks when compared to CNNs on classical classification benchmarks, although their robustness to universal attacks and attacks for dense prediction tasks is still unexplored. Our results seem to substantiate this hypothesis, bringing evidence that CNNs are actually more vulnerable to patch attacks when compared to ViTs for dense prediction.

2.2 Monocular depth estimation

In this section, we introduce and briefly discuss the MDE task. For a deeper analysis of the subject we refer the reader to more comprehensive surveys like [39–41].

MDE is the task of estimating the distance of every pixel of an image from a monocular camera. Recovering a depth map, i.e., the depth information for each pixel, looks to be a trivial task when done by humans or when leveraging stereo images, where we can infer depth from additional information, thanks to the perception from multiple angles or scale relative to known objects (for humans). In the case of monocular images, the task is far from trivial as we can only leverage the intensity of pixels, and has been tackled using deep learning methods which yielded more promising results than hand crafted ones [42]. In particular, MDE has traditionally been solved using CNNs and, more recently, architectures based on the ViT family [17]. These models have been trained in both supervised and self-supervised fashion. In particular, the DPT variant we consider [17] is pre-trained on MIX-6, a large dataset consisting of more than 1.5 million annotated images.

In the case of absence of ground truth data, self-supervised approaches have achieved state-of-the-art results by adopting different methods, like replacing the use of explicit depth data with easier-to-obtain binocular stereo footage [4], devising alternative loss functions [5], or building up image reconstruction networks or generative adversarial networks (GANs) [43].

3 Methods

In this section, we review the main components of our experimental evaluation: MDE and the DPT architecture in Sects. 3.1 and 3.2, the Mid-Air dataset in Sect. 3.3, and our proposed procedure for generating adversarial patches in Sect. 3.4.

3.1 Setup of the problem

In the case of supervised learning, the problem of predicting a depth map from a single RGB image can be viewed as that of estimating a non-linear mapping $\Psi : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is the domain of RGB images $x \in \mathbb{R}^{H \times W \times 3}$ and $\mathcal{Y}$ is the set of depth maps. To this end, we are given a training set $S = \{(x, y)\}_i^N$, where each sample is composed by an RGB image x and the corresponding depth map y. In our scenario, the input image will be a frame captured by a camera mounted on top of a flying drone, like described later on in Sect. 3.3.

Such problem is solved by minimizing an objective loss function. Common criteria for this task are the absolute relative error (AbsRel) as in (1) and the root mean square error (RMSE) as in (2), where $\hat{y} = f(x)$ is the MDE model, and the summation is done over all the indices i of pixels for which a ground-truth is available:

$$\text{AbsRel}(y, \hat{y}) = \frac{1}{N} \sum_i \frac{|y_i - \hat{y}_i|}{y_i}, \tag{1}$$

$$\text{RMSE}(y, \hat{y}) = \sqrt{\frac{1}{N} \sum_i |y_i - \hat{y}_i|^2}. \tag{2}$$

As stated in Sect. 2, the distance in the ground truth dataset depends on the type of measurement that was taken when building the dataset (e.g., metric vs. relative distance, dense vs. sparse depth maps). In our experimental evaluation, we mostly consider metric depth maps with no missing values.

3.2 Dense prediction transformer

The DPT model [17] is an encoder–decoder model which can be pre-trained for the MDE task. Differently from standard convolutive models, it employs transformer blocks [16] both in the encoder and the decoder. Since transformer blocks do not modify the spatial resolution of the image, it also adds specialized components to reassemble the final depth maps at varying sizes, as described next. In this section, we summarize the major components using the same nomenclature as [17], and we refer to [17] for a fuller overview of each module. For simplicity, a schematic overview is also provided in Fig. 1. Because our aim is to evaluate the pre-trained DPT performance on a UAV dataset, we do not modify the architecture of the model with respect to [17].

The first step of the DPT model is to encode the input image into a sequence of tokens, which have a similar meaning to character or words in natural language processing applications. To this end, the image is split into non-overlapping patches of size 16×16, which are then collected in row-major order and vectorized. Denoting by x_i the vector describing the i-th patch, each patch is linearly projected to the final embeddings as:

$$h_i = \left[g(x_i) \;\|\; p_i \right], \tag{3}$$

where $g(\cdot)$ can be either a trainable linear projection (standard DPT model) or a pre-trained ResNet-34 encoder (hybrid DPT model), p_i is a trainable positional embedding, and $\|$ denotes concatenation. An additional trainable token h_{aux}, inspired to the class token of ViTs, is concatenated at the beginning of the patch tokens to provide an additional degree of freedom to the architecture. The final set of tokens h becomes the input to the encoder part of the DPT architecture.

The encoder itself is composed by a stack of transformer blocks, each of which is composed by several transformer layers (details on the hyper-parameter are provided later on in Sect. 4.1). A generic transformer layer is built as follows [16]:

$$z = \text{MHA}(\text{LN}(h)) + h, \tag{4}$$

$$o = \text{MLP}(\text{LN}(z)) + z, \tag{5}$$

where LN denotes layer normalization, MHA multi-head attention, and MLP a generic feed-forward network applied on each token independently, like the two residual connections.

The decoder of the DPT model is instead built from 4 separate blocks (called Reassemble blocks), taking as input the output of three intermediate layers of the encoder and its final output (see Fig. 1). Each block outputs an image of resolution $\frac{H}{s} \times \frac{W}{s}$, where s is 32 for the block acting on the encoder output, and then progressively decreases to 16, 8, and 4 (i.e., blocks closer to the original image are decoded to higher resolutions). These outputs are then progressively aggregated from the lowest-resolution one using so-called Fusion blocks, which apply two residual convolutional layers, followed by an upsampling by a factor of 2, so that the final output of the decoder has shape $\frac{H}{2} \times \frac{W}{2}$.

The Reassemble block is instead composed of two operations: Read and Resample. Denote by t the input to the block, where t_i is the ith token, and there are $N_p + 1$ tokens, corresponding to the original N_p patches and to the readout token. The Read operation is used to merge the readout token inside the other N_p tokens, by a linear projection:

$$t_i = \text{GELU}\left(W \left[t_i \| t_0 \right] \right), \tag{6}$$

where t_0 is the readout token, W is a trainable matrix, and GELU is the Gaussian Error Linear Unit activation function. The resulting N_p tokens are then rearranged into a image-like shape, which is propagated to a strided transposed convolution to achieve the final dimensionality (Resample). To obtain the final depth map estimation, a final head is appended to the output of the decoder, composed of a 3×3 convolutional layer, a 1×1 convolutional layer to project to a single channel, and a final 2×2 upsampling operation.

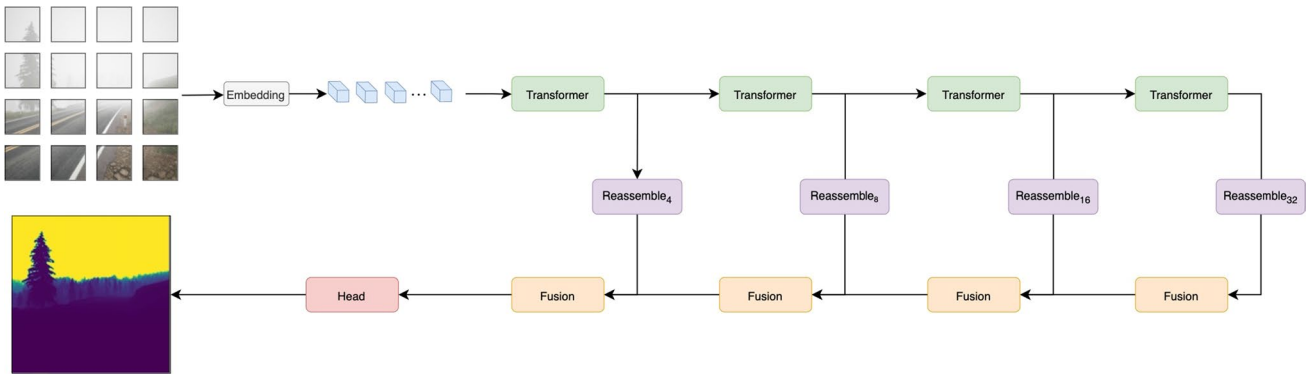


Fig. 1 Schematic representation of a DPT model applied to the task of monocular depth estimation (adapted and modified from [17])

Because the DPT is pre-trained on a mixture of datasets whose depth maps have different semantics, a customized scale-invariant loss [17] is adopted, where two scale and shift values are optimized separately for each dataset and are applied to the (unscaled) output of the DPT model, allowing it to generalize even to scenarios that were not considered in the training set.

3.3 Dataset

As mentioned in Sect. 1, the Mid-Air dataset is used for the fine-tuning of the model in order to evaluate its performance on in-flight operations. Mid-Air (Montefiore Institute Dataset of Aerial Images and Records) [9] is a multi-purpose synthetic dataset containing outdoor videos captured by manually flying a drone in a virtual environment. Because it is comprised of images captured in low-altitude drone flights, it is particularly suitable to our needs. Mid-Air contains synchronized data of multiple sensors, for a total of more than 420k video frames with resolution 1024×1024 simulated in various climate conditions (4 weather setups and 3 different seasons). In order to reduce memory usage the images were down-scaled to a resolution of 512×512 during fine-tuning. Two examples of images and corresponding ground truths taken from the dataset are shown in Fig. 2.

3.4 Adversarial patch attack for MDE

Apart from evaluating the performance of DPT on the Mid-Air dataset (Sect. 3.3), we want to test its robustness to a particular type of adversarial attacks, namely, patch attacks. A patch [22] is a small region which is trained to fool a classifier or other model whenever it is applied on top of an image, before the image itself is sent to the model for prediction.

Our method for generating adversarial attacks is inspired to the one proposed in [23], where a patch is randomly initialized as a tensor of size $R \times R \times C$, where $R \times R$ is the patch resolution (R being smaller than the smallest size of the image to be attacked) and $C = 3$ is the number of channels (RGB). The patch is then reduced to a circle with radius $R/2$ by applying a circular mask to each channel of the patch. The number of trainable parameters in the patch is then equal to $\pi \times \frac{R^2}{4} \times C$. Finally, the patch is transformed before being applied to an image by superposition, and it is trained so that is provides the highest perturbation to any image on the training set, irrespective of the transformation. This is shown visually in Fig. 3.

More in detail, we train the patch by maximizing a perturbation loss over a training set (which in our case is taken as a subset of the test data in order to avoid data leakage from the fine-tuning phase, see Sect. 4), in order to maximize discrepancies between attacked images' depths and original depths:

$$P = \arg\max_{P} \quad \mathbb{E}_{T \in \mathcal{T}} \left[\sum_i L(f(x_i + T(P)),\ y_i) \right], \tag{7}$$

where $f(x)$ is the MDE model, $\mathcal{T}$ is a family of transformations over the original patch (see below), $+$ is intended as the application of the transformed patch over the original image (which we do by elementwise summation during training), L is an error metric used to compute discrepancies between predictions and ground truths, and the sum $\sum_i$ is over all images in the training set of the adversarial patch. In practice, during the training, before the application on each image (both in training and test), the following transformations $\mathcal{T}$ are applied to the patch to make the patch as robust as possible:

- Re-scaling with a random scale factor s^2, $s \in [0.45, 0.55]$ meaning that the actual resolution of the applied patch is $R/4$;
- Rotation with random angle $\alpha \in [-20, 20]$ (degrees);
- Horizontal and vertical translation from the center of the image, with random horizontal offset $x \in [-0.3 \times W, 0.3 \times W]$ and random vertical offset $y \in [-0.05 \times H, 0.3 \times H]$, where W, H are the width and height of the attacked image and the values are negative for left and up offsets, and positive for right and down offsets;

Fig. 2 Examples of images and corresponding ground-truth depths from the dataset Mid-Air in two different climate settings

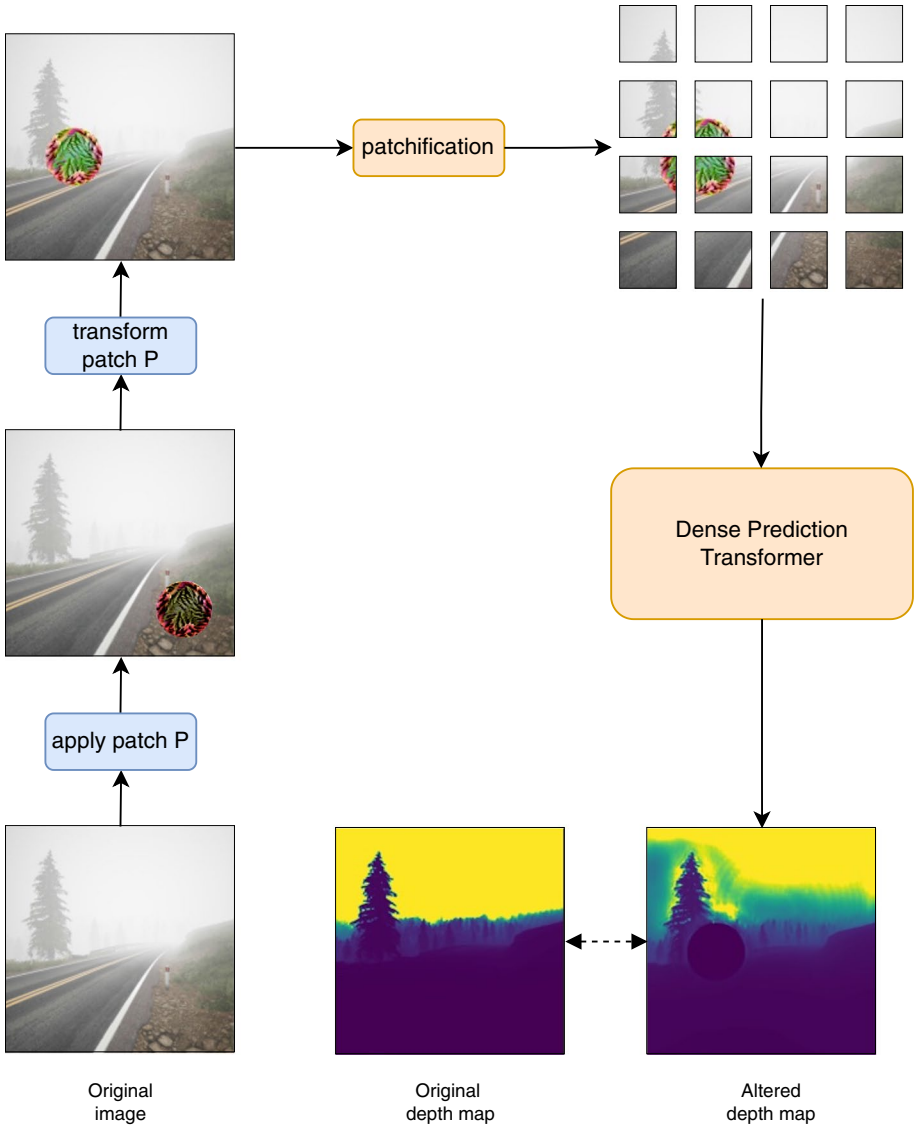


Fig. 3 Schematic description of our patch attack for MDE

- Perspective transformation with random independent translations of the four corners with a distortion factor $d \in [-0.1, 0.1]$ for each coordinate of each of the four corners of the patch.

A set of losses $L(\cdot, \cdot)$ has been tested during hyper-parameter tuning in order to maximize the overall error induced by applying the attacks to the validation set. Using the same notation as Sect. 3.1, denote by $\hat{y} = f(x)$ the output of the MDE model, and assume all summations below are made over all pixels for which a depth ground truth is available (the total number of such pixels is denoted by P, i.e., for dense depth maps like in Mid-Air we will have $P = HW$). Apart from the AbsRel (Eq. (1)) and the RMSE (Eq. (2)), we also consider:

- A mean absolute error (MAE) loss, where the ℓ_2 norm is replaced with its absolute value:

$$L(y, \hat{y}) = \frac{1}{P} \sum_i |y_i - \hat{y}_i| \tag{8}$$

- The Scale Invariant Error (SIE, [1]) loss:

$$L(y, \hat{y}) = \frac{1}{P} \sum_i d_i^2 - \frac{\lambda}{P^2} \left(\sum_i d_i \right)^2, \tag{9}$$

where $d_i = \log y_i - \log \hat{y}_i$ is the per-pixel difference between predicted and ground-truth log-depth maps, and λ is a hyper-parameter.

In addition, we note during the fine-tuning phase (see Sect. 4.1) that high-accuracy of the model is obtained by learning to predict accurately short-term distances (<

200 m) which are prevalent in the dataset. Based on this observation, we also test a variant of the AbsRel loss function, where each pixel is weighted based on its ground truth depth, which we call distance-weighted error (DWE):

$$L(y, \hat{y}) = \frac{1}{P} \sum_i W(\hat{y}_i) \|y_i - \hat{y}_i\|. \tag{10}$$

The per-pixel weights are computed as a function of the ground truth depth using a truncated Normal density function of the depth, $W(a) = \mathcal{N}(a; \mu, \sigma)$, and the parameters of the truncated Normal μ and σ are computed w.r.t. the extreme values of the interval in which the attack is focused as $\mu = \frac{1}{2}(\max(I) + \min(I))$ and $\sigma = (\max(I) - \min(I))/2$. In particular, we set $\min(I) = 0$ and $\max(I) = 300$ based on the above reasoning.

The loss that produced the overall best attacking performances was the AbsRel, and while DWE showed some ability in focusing the attack on a depth interval of interest and excluding depths outside of the interval (by properly tuning the weighting parameters), the error induced by these localized attacks was smaller than the one induced by a global attack on the whole depth map with the AbsRel. Further experiments may bring better results in depth-localized attacks, and we leave these for future work.

After training, the trained patch is evaluated by randomly applying it on top of the remaining part of the test set, and computing its average effect on the predictions. Note that, even if the patch is highly localized, due to the effect it has on the non-linear processing of the neural networks, its perturbation is expected to have a global receptive field.

4 Experiments

4.1 Fine-tuning over Mid-Air

For our experiments we use the pre-trained DPT hybrid model [17], and we fine-tune it on the training part of the Mid-Air dataset. As baseline, we compare the results of the attacks on DPT with the results obtained attacking a state-of-the-art Fully Convolutional Residual Network (FCRN) model having a ResNet50 encoder pre-trained on ImageNet [47] and trained on the same train-set used for fine-tuning DPT. The pre-trained DPT model was trained by its authors with the procedure described in [17] using the MIX-6 Dataset designed by the same authors. In particular, the hybrid variant of DPT includes a ResNet50 encoder pre-trained on ImageNet [47]. The pre-trained weights of the DPT model and the implementation of the model architecture are available online at https://github.com/isl-org/DPT.

The FCRN model was built from pre-trained ResNet50 blocks forming the encoder part of the architecture and with a final decoding part bringing back the intermediate feature maps to the original input size, consisting of unpooling and convolutional (trainable) layers initialized randomly and a final bilinear upsampling to align the output shape to the input shape. For our experiments, the model was trained on a subset consisting of 50,000 images coming from the Mid-Air Dataset for 6 epochs using the scale-invariant loss (9), with different learning rates for the encoder and decoder parts of the network. For the ResNet50 encoder (pre-trained on ImageNet), the learning rate was set to 10^{-3}, i.e., 1/10 of the learning rate set to 10^{-2} for the de-convolution block.

The fine-tuning procedure of DPT used for our experiments follows the indications provided by the authors, i.e., we use the validation set for finding the optimal scale and shift parameters for the scale-invariant loss on the Mid-Air dataset in order to align the predictions to the average scale and shift of the dataset ground-truth depths. The tuned scale-invariant loss was then used for fine-tuning DPT for 10 epochs with a learning rate $lr = 10^{-7}$ over the same train-set used for training the FCRN model. The procedure to compute an estimate of the correct scale and shift parameters is proposed by the authors of DPT [17], and consists in finding the values of scale and shift that minimize the AbsRel over the training set, and then using these values both in training and testing to align the outputs to the ground-truths by applying the optimal parameters to the predicted depths as $d^* = s * d + r$, denoting with s, r the scale and shift, and with d, d^* the unaligned and aligned output depths, respectively.

This alignment is necessary in order to make use of the unscaled spatial information coming from the pre-training of DPT, otherwise the unscaled depth-maps produced by DPT without the alignment could be so far from the ground-truth depths that the training would be basically starting from scratch, given the possibly high values of the losses.

The 50,000 training images were selected in order to reflect the variability introduced by the different simulated seasons, so that most of the different climate and lighting variations of the environment could be seen during training, with the exception of the summer season which was not simulated by the authors of Mid-Air.

We show the results of this experimental section in Table 1, and we plot the AbsRel over the test set as a function of the pixel's distance in Fig. 4.

In order to more easily read the table, we give a quick description of the metrics reported in it.

The Logarithmic RMSE, reported as RMSE_{10}, is defined as

$$\text{RMSE}_{10}(y, \hat{y}) = \sqrt{\frac{1}{P} \sum_i \left[\log_{10}(y_i) - \log_{10}(\hat{y}_i)\right]^2}$$

Table 1 Results of the fine-tuning procedure on the Mid-Air dataset

Method	Distance	RMSE	AbsRel	$RMSE_{10}$	δ_1	δ_2	δ_3
FCRN [44]	80 m	8.31	0.16	0.06	0.83	0.94	0.67
	500 m	53.07	0.29	0.08	0.81	**0.92**	**0.96**
	1000 m	104.8	0.39	0.08	0.81	0.92	0.96
DPT (zero-shot)	80 m	13.20	0.33	0.13	0.53	0.73	0.87
	500 m	95.26	0.49	0.18	0.47	0.67	0.80
	1000 m	204.74	0.62	0.20	0.47	0.66	0.79
DPT (fine-tuned)	80 m	**7.48**	**0.14**	**0.05**	**0.88**	**0.96**	**0.98**
	500 m	**46.10**	**0.19**	**0.07**	**0.86**	0.85	**0.97**
	1000 m	**98.43**	**0.21**	**0.07**	**0.85**	**0.94**	**0.96**

Best results for each distance threshold are in bold

and unlike the RMSE, it is less influenced by the higher values of the absolute difference found in the distant parts of the image given the use of the log-depths as arguments. The threshold accuracies, reported in the table as δ_p, $p \in 1, 2, 3$, are defined as the ratios of pixels for which the ratio between predicted and true depth and its inverse are both smaller than the thresholds $\delta_p = 1.25^p$, $p \in \{1, 2, 3\}$. These ratios are expressed by the formula:

$$\delta_p(y, \hat{y}) = \frac{1}{P} \sum_i \max\left(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i}\right) \leq \delta_p, \quad p \in \{1, 2, 3\}$$

and higher values for these ratios indicate more accurate models.

For each experiment, we evaluate three different variants by clamping the maximum predicted distance to a given threshold (respectively, of 80 m, 500 m, and 1 km). We note that for all scenarios, the zero-shot DPT has good performances which, however, are not on-par with a standard FCRN architecture. However, after fine-tuning, the DPT model obtains state-of-the-art results in all scenarios. For example, for the case where the maximum distance is 500 m, we obtain a 51% relative improvement in RMSE over the zero-shot version, and a 13% improvement over the FCRN model. Other metrics have a similar behavior, e.g., we obtain a 61% improvement in AbsRel over the zero-shot DPT, and a 29% relative improvement in AbsRel over the FCRN.

It is interesting to observe from Fig. 4 that the poor results of the zero-shot DPT variant stems from a high AbsRel over

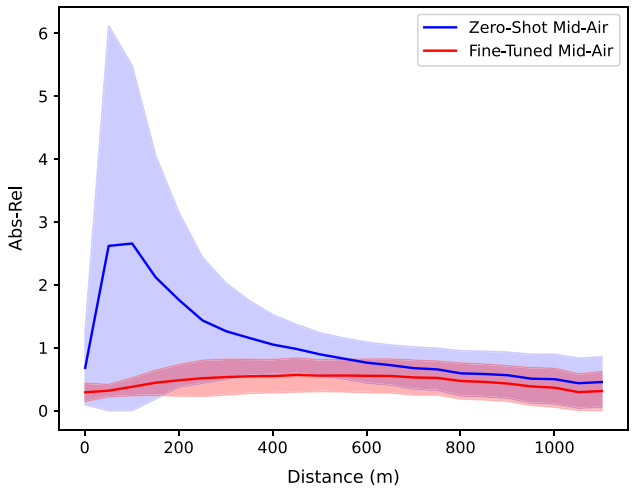


Fig. 4 Results of the fine-tuning procedure, aggregated with respect to the true distance of each pixel

short (< 300 m) distances, which are quite common in the Mid-Air dataset, and which are impacted significantly by the fine-tuning procedure.

In Table 2, we provide an additional benchmark comparison of the architecture with recent state-of-the-art models for MDE, including models that make use of auxiliary data for the training. In the case of M4Depth [46], in particular, the training is performed leveraging a captured input sequence together with the motion information about the sequence itself. DPT is remarkably competitive with

Table 2 Additional performance comparisons with other state-of-the-art models, considering a maximum distance of 80 m from the camera

Method	Data	RMSE	AbsRel	$RMSE_{10}$	δ_1	δ_2	δ_3
FCRN [44]	Image	8.31	0.16	0.06	0.83	0.94	0.67
Monodepth2 [5]	Image	12.351	0.394	0.462	0.610	0.751	0.833
ManyDepth [45]	Image sequence	10.919	0.203	0.327	0.723	0.876	0.933
M4Depth-d6 [46]	Image sequence and motion	7.043	0.105	0.186	0.919	0.953	0.969
DPT (zero-shot)	Image	13.20	0.33	0.13	0.53	0.73	0.87
DPT (fine-tuned)	Image	7.48	0.14	0.05	0.88	0.96	0.98

this architecture as well, which is the current state-of-the-art on the MidAir dataset, while being able to predict the depth map starting from a single frame. The fine-tuned version is also significantly better than alternative state-of-the-art models including Monodepth2 [5] and ManyDepth [45].

The excellent scores attained by DPT come at the cost of a higher computational complexity, as shown in Table 3, where we provide a comparison in terms of Multiply-and-Accumulate (MAC) operations and millions of parameters. Nevertheless, DPT is still able to predict accurate depth maps in extraordinary short times (few tens of milliseconds), even without resorting to last generation hardware, due to a high degree of parallelism through its wide and rather shallow structure [17].

4.2 Adversarial attacks to transformers for MDE

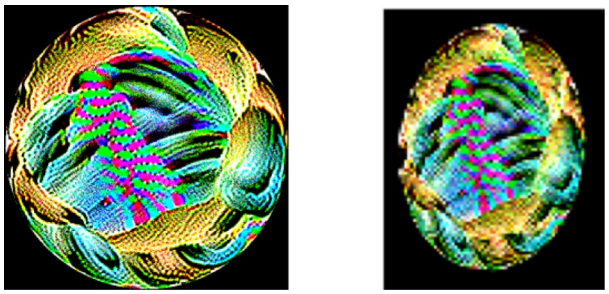
Different losses and parameter configurations were evaluated through a random-search hyper-parameter tuning before finding the optimal loss and configuration used in our experiments. The best results on the validation set used for the tuning were obtained by training with the AbsRel as training loss using the Adam optimization algorithm for 50 epochs with a learning rate of 500. The trainable patch was initialized randomly as a parameters tensor of shape $256 \times 256 \times 3$, and a circular mask is applied to all channels in order to obtain an approximately circular RGB patch of diameter 256 pixels. We show an example of trained patch for the fine-tuned DPT model in Fig. 5a, an example of transformed patch in Fig. 5b, and an example of application on the test set in Fig. 6.

In Tables 4 and 5, we report the average performances of DPT and the FCRN models on the test set (obtained after removing the 400 images used to train the adversarial patch), both with and without using the patch itself. By comparing these results with the performances on the unattacked test set we observe that in average the error induced by the adversarial patch is unsatisfactory for DPT, especially comparing these results with the ones obtained on the model FCRN by training a new patch against this model with the same configuration used against DPT.

Table 3 Complexity of benchmark models, in terms of MAC operations and millions of parameters. In this table we exclude sequence-based models

Model	WxH	Million Params	MACs
FCRN	345×460	62.5	90
Monodepth2	640×192	14	8
DPT hybrid	384×384	123	110
DPT large	384×384	343	280

The best configuration yielded results that, although visible, can't be interpreted as successful attacks, since the predictions on most input images remain quite accurate (except for the portion of the images covered by the patch) after the application of the adversarial patch. The inputs for which the patch is actually causing some visible error are the ones for which accurate estimation is already harder, such as inputs in which fog or snow covers most of the scene. This can be seen in Fig. 7, while in Fig. 8 an example of a failed attack can be seen for a more common Mid-Air setting, where the highest errors are the ones regarding the portion of the image covered by the patch itself. By comparisons, the attacks on the FCRN model (Figs. 9 and 10) are significantly more powerful, as can also be seen by the relative errors in Table 4.



(a) Trained patch (b) Transformed patch

Fig. 5 **a** Trained patch for DPT; **b** the same trained patch after a random transformation from $\mathcal{T}$ is applied on top of it

Fig. 6 Example of a patch trained attacking DPT on Mid-Air images, transformed and applied on an image of the test-set

Table 4 Results of the attacks on the fine-tuned DPT model and the FCRN model trained on Mid-Air on a test set of 400 images coming from Mid-Air

Method	Distance	RMSE	AbsRel	$RMSE_{10}$	δ_1	δ_2	δ_3
FCRN	80 m	8.83	0.18	0.13	0.78	0.91	0.96
	500 m	55.36	0.33	0.19	0.76	0.89	0.95
	1000 m	109.04	0.44	0.22	0.76	0.89	0.94
FCRN (attacked)	80 m	15.62	0.54	0.23	0.66	0.79	0.86
	500 m	96.54	1.41	0.34	0.63	0.77	0.84
	1000 m	182.37	2.08	0.38	0.63	0.76	0.83
DPT (unattacked)	80 m	7.92	0.15	0.11	0.83	0.95	0.98
	500 m	48.98	0.21	0.15	0.81	0.93	0.96
	1000 m	106.10	0.23	0.18	0.80	0.93	0.96
DPT (attacked)	80 m	8.78	0.16	0.13	0.79	0.93	0.96
	500 m	52.03	0.23	0.18	0.76	0.91	0.95
	1000 m	112.89	0.25	0.20	0.75	0.90	0.94

Table 5 Percentage variations of the metrics after the attacks for the two models

Method	Distance	RMSE	AbsRel	$RMSE_{10}$	δ_1	δ_2	δ_3
FCRN	80 m	76.95%	196.34%	70.85%	−15.66%	−12.31%	−9.48%
	500 m	74.37%	329.00%	75.08%	−16.88%	−13.60%	−10.94%
	1000 m	67.25%	372.71%	73.79%	−17.08%	−13.72%	−11.05%
DPT	80 m	10.92%	9.07%	24.02%	−5.24%	−2.23%	−1.34%
	500 m	5.82%	7.07%	15.60%	−5.92%	−2.64%	−1.55%
	1000 m	5.31%	5.96%	13.40%	−5.70%	−2.86%	−1.74%

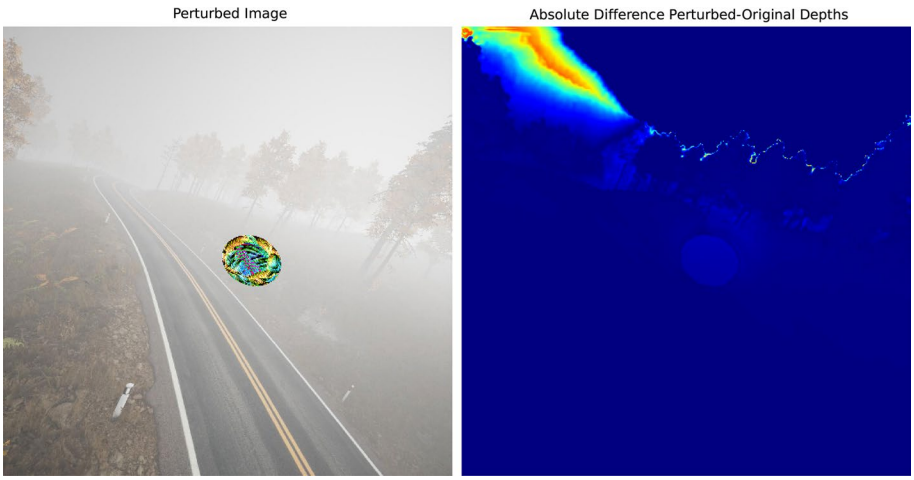


Fig. 7 Example of successful attack on DPT with a hard input

4.3 Transfer experiments

An interesting research question is to determine if the patch trained for attacking one model can be used to successfully attack a different model. This would configure as a black-box adversarial patch attack, where the architecture of the attacked model is not known and possibly very different from the one for which the patches are trained. In our setting, the two model architectures are indeed very different (transformer/CNN), although they both share a pre-trained ResNet50 encoder, so if the attack is influencing the predictions at the embedding extraction stage we might expect to see some influence on the predictions because of these shared embedding blocks.

The results reported in Table 6 show that FCRN suffers more in terms of AbsRel error with the patch trained for DPT, while DPT suffers more in terms of all the other metrics, but predictions for both models are actually almost

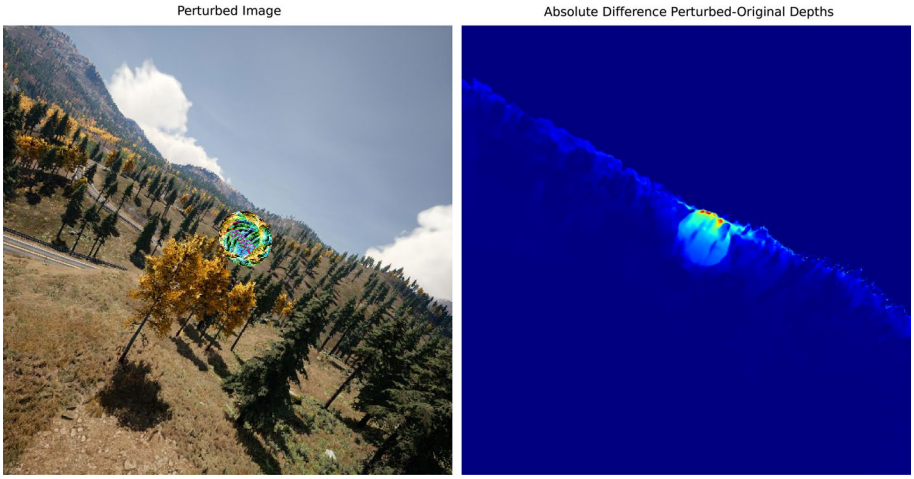


Fig. 8 Example of failed attack on DPT with a normal input

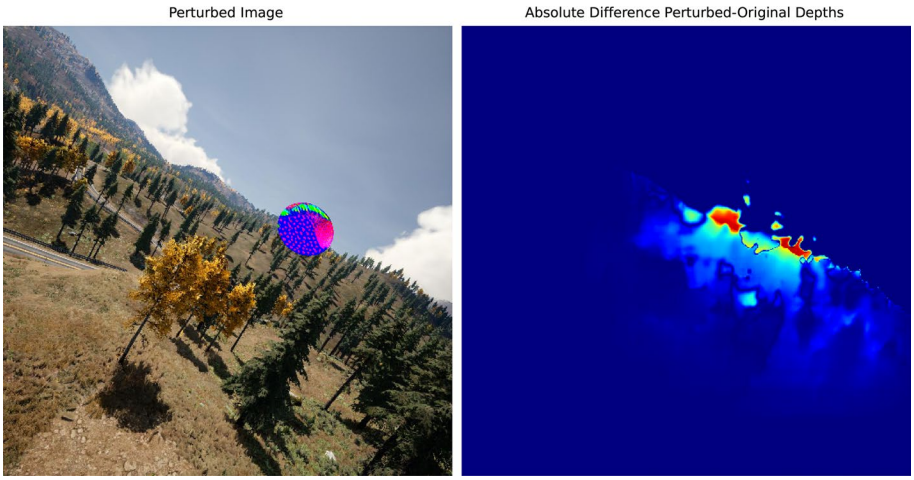


Fig. 9 Example of attack on FCRN with a normal input

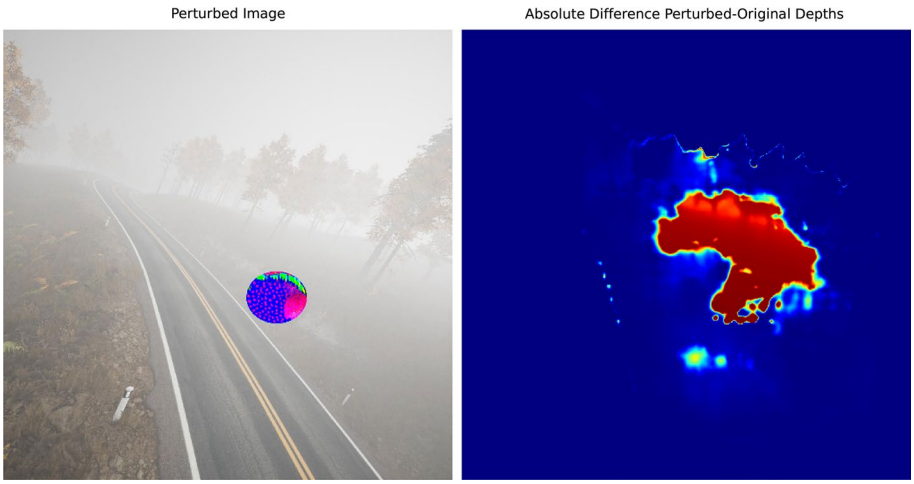


Fig. 10 Example of attack on FCRN with a hard input

unmodified when the patch trained on a different model is applied.

Examples in Figs. 11 and 12 show that the effect on hard inputs is the same already observed for the attacks on DPT with its own trained patch, possibly indicating that on these inputs the effect that the patch induces on the encoder part of both architectures could be the most influential on predictions for these kinds of inputs, although this claim is beyond the scope of this experiment and requires further investigation to be verified or refuted.

Table 6 Percentage variations of the metrics after attacking each model with the patch trained for attacking the other

Method	Distance	RMSE	AbsRel	$RMSE_{10}$	δ_1	δ_2	δ_3
FCRN	80 m	6.31%	8.13%	4.42%	−2.25%	−1.12%	−0.48%
	500 m	4.77%	8.14%	4.41%	−2.50%	−1.35%	−0.67%
	1000 m	4.42%	8.02%	4.28%	−2.54%	−1.37%	−0.69%
DPT	80 m	7.38%	2.58%	16.95%	−3.74%	−1.36%	−0.90%
	500 m	4.92%	2.08%	11.06%	−4.25%	−1.65%	−1.06%
	1000 m	4.56%	1.55%	9.70%	−4.08%	−1.83%	−1.21%

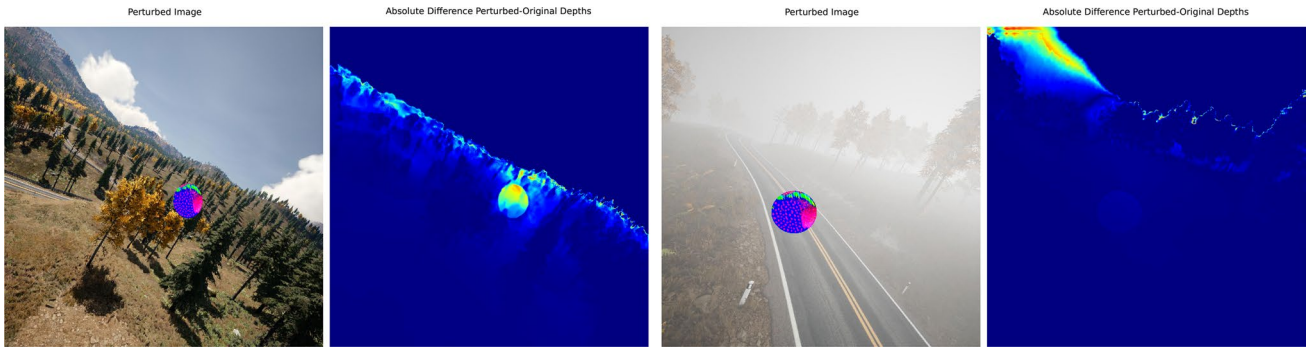


Fig. 11 Example of transfer attack on DPT with regular (left) and hard (right) input, using a patch trained to attack FCRN

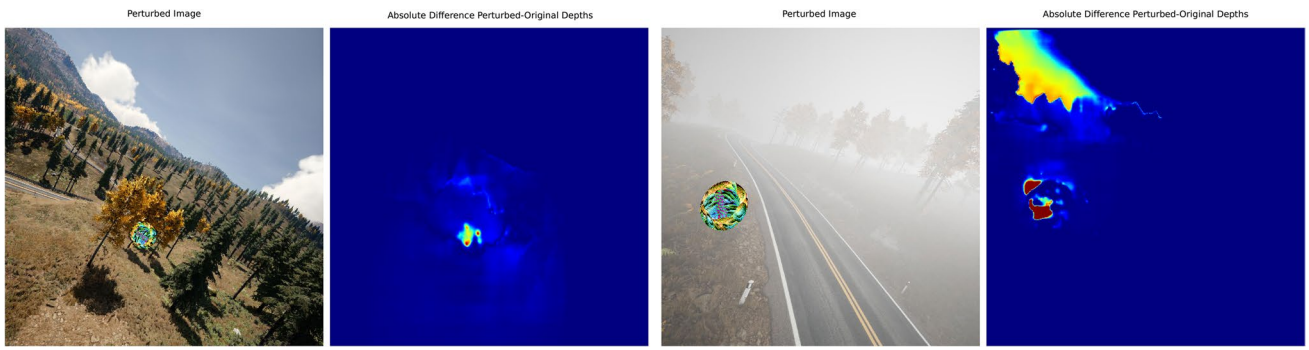


Fig. 12 Example of transfer attack on FCRN with regular (left) and hard (right) input, using a patch trained to attack DPT

5 Conclusions and future works

In this paper, we evaluated for the first time the performance of a ViT model on a MDE task performed from a flying drone. We show that the model is capable of good performances even in a zero-shot context, while it achieves state-of-the-art results after a brief fine-tuning phase. Next, we evaluated its robustness to an adversarial attack (in the form of a pre-trained patch), showing it possesses a strong degree of robustness against this kind of attacks, while competitive fully convolutional models can be fooled with a high degree of precision. Overall, our results highlight the strong potential of this class of models for performing MDE in scenarios involving low-altitude flying drones, where robustness is also important. Future work will study the performance of the model on a real setup, with the collection of a large dataset of low-altitude flight scenarios, and an analysis of the robustness of DPT against physical versions of our patch attack and more general corruptions and attacks, as well as a study of the transferability of the attacks to a wider range of scenarios, datasets, and architectures.

Author contributions S.S., S.M., and L.M. elaborated the original idea and the experimental setup. A.D. and S.E. provided the main implementation and tested the algorithms. L.M. and S.M. provided some test benchmarks and contributed to the evaluation of the results. S.S., S.M., A.D., L.M., S.E., and M.S. participated in writing and reviewing the manuscript. All authors read and approved the final manuscript.

Funding No funding was received to assist with the preparation of this manuscript.

Availability of data and materials The dataset used for this article is available at https://midair.ulg.ac.be/ under CC BY-NC-SA 4.0 license.

Declarations

Competing interests The authors do not have competing interests to declare.

References

1. Eigen D, Puhrsch C, Fergus R (2014) Depth map prediction from a single image using a multi-scale deep network. Adv Neural Inf Process Syst 27. https://dl.acm.org/doi/10.5555/2969033.2969091
2. Eigen D, Fergus R (2015) Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Proceedings of the IEEE international conference on computer vision, pp 2650–2658
3. Liu F, Shen C, Lin G (2015) Deep convolutional neural fields for depth estimation from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5162–5170
4. Godard C, Mac Aodha O, Brostow GJ (2017) Unsupervised monocular depth estimation with left-right consistency. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 270–279
5. Godard C, Mac Aodha O, Firman M, Brostow GJ (2019) Digging into self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 3828–3838
6. Nathan Silberman, PK Derek Hoiem, Fergus R (2012) Indoor segmentation and support inference from rgbd images. In: ECCV
7. Silberman N, Hoiem D, Kohli P, Fergus R (2012) Indoor segmentation and support inference from rgbd images. In: Computer Vision, ECCV 2012-12th European conference on computer vision, proceedings. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp 746–760
8. Saxena A, Sun M, Ng AY (2009) Make3d: Learning 3d scene structure from a single still image. IEEE Trans Pattern Anal Mach Intell 31(5):824–840
9. Fonder M, Van Droogenbroeck M (2019) Mid-air: a multi-modal dataset for extremely low altitude drone flights. In: 2019 IEEE/CVF Conference on computer vision and pattern recognition workshops (CVPRW), pp 553–562
10. Ranftl R, Lasinger K, Hafner D, Schindler K, Koltun V (2020) Towards robust monocular depth estimation: mixing datasets for zero-shot cross-dataset transfer. IEEE Trans Pattern Anal Mach Intell. https://ieeexplore.ieee.org/document/9178977
11. Zhang Z, Xiong M, Xiong H (2019) Monocular depth estimation for uav obstacle avoidance. In: 2019 4th International conference on cloud computing and internet of things (CCIOT), pp 43–47. IEEE
12. Madhuanand L, Nex F, Yang MY (2021) Self-supervised monocular depth estimation from oblique uav videos. ISPRS J Photogramm Remote Sens 176:1–14
13. Shimada T, Nishikawa H, Kong X, Tomiyama H (2022) Pix2pix-based monocular depth estimation for drones with optical flow on airsim. Sensors 22(6):2097
14. Djenouri Y, Hatleskog J, Hjelmervik J, Bjorne E, Utstumo T, Mobarhan M (2022) Deep learning based decomposition for visual navigation in industrial platforms. Appl Intell 52(7):8101–8117
15. Ajakwe SO, Ihekoronye VU, Kim D-S, Lee JM (2022) Dronet: multi-tasking framework for real-time industrial facility aerial surveillance and safety. Drones 6(2):46
16. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S (2020) An image is worth 16x16 words: transformers for image recognition at scale. In: International conference on learning representations
17. Ranftl R, Bochkovskiy A, Koltun V (2021) Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 12179–12188
18. Eykholt K, Evtimov I, Fernandes E, Li B, Rahmati A, Xiao C, Prakash A, Kohno T, Song D (2018) Robust physical-world attacks on deep learning visual classification. In: Proc. IEEE conference on computer vision and pattern recognition, pp 1625–1634
19. Huang L, Gao C, Zhou Y, Xie C, Yuille AL, Zou C, Liu N (2020) Universal physical camouflage attacks on object detectors. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 720–729
20. Chiang P-Y, Ni R, Abdelkader A, Zhu C, Studer C, Goldstein T (2020) Certified defenses for adversarial patches. arXiv preprint arXiv:2003.06693
21. Liu X, Yang H, Liu Z, Song L, Li H, Chen Y (2018) Dpatch: an adversarial patch attack on object detectors. arXiv preprint arXiv:1806.02299
22. Brown TB, Mané D, Roy A, Abadi M, Gilmer J (2017) Adversarial patch. arXiv preprint arXiv:1712.09665
23. Yamanaka K, Matsumoto R, Takahashi K, Fujii T (2020) Adversarial patch attacks on monocular depth estimation networks. IEEE Access 8:179094–179104
24. Akhtar N, Mian A (2018) Threat of adversarial attacks on deep learning in computer vision: a survey. IEEE Access 6:14410–14430
25. Biggio B, Roli F (2018) Wild patterns: ten years after the rise of adversarial machine learning. Pattern Recogn 84:317–331
26. Yuan X, He P, Zhu Q, Li X (2019) Adversarial examples: attacks and defenses for deep learning. IEEE Trans Neural Netw Learn Syst 30(9):2805–2824
27. Dalvi NN, Domingos PM, Mausam Sanghai SK, Verma D (2004) Adversarial classification. Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining
28. Lowd D, Meek C (2005) Adversarial learning. In: Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery in data mining, pp 641–647
29. Zhou Y, Jorgensen Z, Inge M (2008) Countering good word attacks on statistical spam filters with instance differentiation and multiple instance learning. In: Tools in artificial intelligence. IntechOpen

30. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R (2013) Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199
31. Xie C, Wang J, Zhang Z, Zhou Y, Xie L, Yuille A (2017) Adversarial examples for semantic segmentation and object detection. In: Proceedings of IEEE international conference on computer vision (ICCV), pp 1369–1378
32. Song D, Eykholt K, Evtimov I, Fernandes E, Li B, Rahmati A, Tramer F, Prakash A, Kohno T (2018) Physical adversarial examples for object detectors. In: Proceedings of 12th USENIX Workshop on Offensive Technologies (WOOT)
33. Cisse M, Adi Y, Neverova N, Keshet J (2017) Houdini: fooling deep structured prediction models. arXiv preprint arXiv:1707.05373
34. Wu Z, Lim S-N, Davis LS, Goldstein T (2020) Making an invisibility cloak: real world adversarial attacks on object detectors. In: Proceedings of European conference on computer vision (ECCV), pp 1–17. Springer
35. Arnab A, Miksik O, Torr PH (2018) On the robustness of semantic segmentation models to adversarial attacks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 888–897
36. Moosavi-Dezfooli S-M, Fawzi A, Fawzi O, Frossard P (2017) Universal adversarial perturbations. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1765–1773
37. Su J, Vargas DV, Sakurai K (2019) One pixel attack for fooling deep neural networks. IEEE Trans Evol Comput 23(5):828–841
38. Mahmood K, Mahmood R, Van Dijk, M (2021) On the robustness of vision transformers to adversarial examples. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 7838–7847
39. Bhoi A (2019) Monocular depth estimation: a survey. arXiv preprint arXiv:1901.09402
40. Xiaogang R, Wenjing Y, Jing H, Peiyuan G, Wei G (2020) Monocular depth estimation based on deep learning: a survey. In: 2020 Chinese Automation Congress (CAC), pp 2436–2440. IEEE
41. Ming Y, Meng X, Fan C, Yu H (2021) Deep learning for monocular depth estimation: a review. Neurocomputing 438:14–33
42. Saxena A, Chung S, Ng A (2005) Learning depth from single monocular images. Adv Neural Inf Process Syst 18. https://dl.acm.org/doi/10.5555/2976248.2976394
43. Aleotti F, Tosi F, Poggi M, Mattoccia S (2018) Generative adversarial networks for unsupervised monocular depth prediction. In: Proceedings of the European conference on computer vision (ECCV) workshops
44. Laina I, Rupprecht C, Belagiannis V, Tombari F, Navab N (2016) Deeper depth prediction with fully convolutional residual networks. In: 2016 Fourth international conference on 3D vision (3DV), pp 239–248. IEEE
45. Watson J, Mac Aodha O, Prisacariu V, Brostow G, Firman M (2021) The temporal opportunist: self-supervised multi-frame monocular depth. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 1164–1174
46. Fonder M, Ernst D, Van Droogenbroeck M (2021) M4depth: a motion-based approach for monocular depth estimation on video sequences. arXiv preprint arXiv:2105.09847
47. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on computer vision and pattern recognition, pp 248–255