A Little Fog for a Large Turn

Harshitha Machiraju, Vineeth N Balasubramanian
Indian Institute of Technology, Hyderabad, India
{ee14btech11011, vineethnb}@iith.ac.in

Abstract

Small, carefully crafted perturbations called adversarial perturbations can easily fool neural networks. However, these perturbations are largely additive and not naturally found. We turn our attention to the field of Autonomous navigation wherein adverse weather conditions such as fog have a drastic effect on the predictions of these systems. These weather conditions are capable of acting like natural adversaries that can help in testing models. To this end, we introduce a general notion of adversarial perturbations, which can be created using generative models and provide a methodology inspired by Cycle-Consistent Generative Adversarial Networks to generate adversarial weather conditions for a given image. Our formulation and results show that these images provide a suitable testbed for steering models used in Autonomous navigation models. Our work also presents a more natural and general definition of Adversarial perturbations based on Perceptual Similarity. [1]

1. Introduction

Autonomous navigation has occupied a central position in the efforts of computer vision researchers in recent years. Autonomous vehicles can not only aid navigation in urban areas but also provide critical support in disaster-affected areas, places with unknown topography (such as Mars), and many more. The vast potential of the applications thereof and the feasibility of the solutions in contemporary times has led to the growth of several organizations across industry, academia, and government institutions that are investing significant efforts on self-driving vehicles. Computer vision has been studied and shown to play an important role in the development of autonomous navigation technologies over the years [17][14][19]. Vision tasks such as image-level classification, object detection, semantic segmentation, as well as steering angle prediction, play critical roles in the development of autonomous vehicles. The increasing emphasis of this problem domain has also led to the creation of different vision datasets that are necessary to develop solutions across different geographies [10][23][37].

Figure 1: Steering Angle(radians) deviation seen in the same scene due to Fog, for the AutoPilot model[8]. Lower image was generated by our method given the left image

It is common knowledge now that deep neural networks have achieved state-of-the-art performance in many computer vision tasks [16, 25]. With great leaps in performance, deep learning models have been deployed in many physical systems, and efforts have been afoot to develop robust deep neural network models for autonomous vehicles too [5][8][33]. However, the recent mishaps involving self-driving vehicles has necessitated the requirement for testing such deep learning models with data in various conditions. Existing efforts largely rely on copious amounts of collected data from the real world, data augmentation using simple affine transformations [35] or the use of data from synthetic/virtual environments [30] for various conditions. There is an impending need for methods that can provide data with a wider variety of conditions that can validate the robustness of the learned models for vision tasks in such settings, in order to save lives and property in the future. One such important dimension is the variability of a given

[1] Accepted to WACV 2020

environment under various weather conditions. Real-world studies such as [1] have shown that bad weather can alone manage to crash navigation systems. It is hence important to train deep learning models with as many weather conditions of a given environment as possible to obtain robust systems in deployment. Our efforts in this work are towards addressing this need.

From a different perspective, many recent efforts have also been made to study the robustness of deep neural network models, by showing how vulnerable they can be towards adversarial perturbations [2], which involve adding a small amount of noise to the input data in order to fool the model. Attempts have been made to show that systems performing essential day-to-day tasks such as face and speech recognition [34][7], as well as physical systems [20] can be attacked using such adversarial perturbations. Similar efforts have also been attempt to attack autonomous navigation models such as in [13][9]. These efforts use adversarial patches [6] physically placed in the field of view of the model to fool it. However, as shown in a very recent work [18], natural adversarial examples are sufficient to fool them, and do not need any explicit image manipulation or hacking with an intention to fool such systems. Weather-based changes in the environment fall into such a category, which we focus on in this work. For example, as shown in Figure 1, when fog is added to the scene, the steering angle predicted by deep learning models deviates by a significant amount from the original, making these models vulnerable to such 'weather-adversarial' data.

In this work, we bring together two perspectives: the data augmentation one and the adversarial one, to explicitly generate weather-adversarial images that can fool deep neural networks for autonomous vehicles (steering angle prediction, in particular). To the best of our knowledge, this is the first effort in this direction. Such an effort is important to provide essential data from different conditions that are likely to affect (or attack) such models. Our work can be utilized to test steering angle prediction models and their robustness against adverse weather conditions. (We focus on fog in this work, due to the availability of relevant data, but our framework is generalizable and can be easily extended to other weather conditions.) Our work also demonstrates that current steering angle models used in self-driving cars are inadequate to handle weather variations in real-world settings. The key contributions of this work are as follows:

- We bring together data augmentation and adversarial perspectives to introduce a methodology that can generate foggy images that are intended to 'fool' models for steering angle prediction in autonomous vehicles. Our methodology integrates an adversarial loss term in an unpaired image-to-image translation framework [42][4] towards the aforementioned objective. To the best of our knowledge, this is the first such effort, in particular, for autonomous navigation applications.
- Existing adversarial attack models are largely focused on classification tasks; we provide an extension of such attacks to regression tasks such as steering angle prediction models.
- We validate the proposed methods using qualitative and quantitative analysis on a well-known autonomous navigation dataset to showcase its promise. We also show that a model that is adversarially trained using the images generated by our method provides significant robustness to the model.

The remainder of this paper is organized as follows. In Section 2, we review previous related efforts in the areas of autonomous navigation and adversarial attacks. We describe our methodology in Section 3, with a perspective of how this provides a more general notion of natural adversaries. Our implementation details, experiments and results are shown in Sections 4 and 5 respectively, followed by conclusions and future directions in Section 7.

2. Related Work

Considering the focus of this work, we present an overview of related earlier efforts from perspectives of both testing the robustness of vision models in autonomous navigation, as well as adversarial attacks in general. In each of these discussions, we present the limitations of the existing efforts and the scope of improvement when the proposed method is used.

2.1. Adversarial Attacks

The conventional notion of an adversarial attack is generally formalized as: given a classification model f, and input image $\mathbf{x}$, we define a perturbation δ as a quantity that is added to $\mathbf{x}$ such that:

$$\arg\max f(\mathbf{x}+\delta) \neq \arg\max f(\mathbf{x}) \;\text{ and }\; ||\delta|| \leq \epsilon \qquad (1)$$

The δ defined above has usually been specific to the input and found through methods like FGSM [15], JSMA [27], and the more recent PGD [24]. The δ may also be common to the entire dataset and may be found iteratively like in the case of UAP [26] or may be generated by a GAN [28]. For the interested reader, a detailed survey of these methods is provided by Akhtar et al. in [2]. All these methods attempt to add a perturbation to the image, which can fool the trained model. Our proposed work is different in two ways from these efforts: (i) we extend the concept of adversaries to go beyond additive perturbations to natural perturbations such as induced by weather changes in disturbing the model (similar to [18], can be considered implicitly adversarial); (ii) most existing efforts focus on adversarial attacks in classification settings, and there has not been much

effort to define an adversary in a regression-based setting. We propose an adversarial loss for regression models (steering angle prediction, in particular) in this work.

Attacking Autonomous Navigation Models: There have been a few explicit efforts in the recent past to develop adversarial attacks for vision models in self-driving cars. Works such as that of Eykholt et al. [13] utilize adversarial patches [6] which are physically added to objects like traffic lights to fool the model. Similarly, Zhang et al. [41] resort to physically camouflaging cars to fool object detector models used in autonomous navigation systems, to test the robustness of the model. We observe that these efforts rely on physical changes in the environment to attack the black box models, for which human effort in case of large scale testing may be prohibitive. On the other hand, it has been shown that weather-induced environment changes naturally result in implicit adversarial circumstances for the model involved [29], and robustness to such weather-adversarial samples is also critical for models in autonomous navigation. We hence focus on generating natural-looking images of existing scenes affected by weather conditions (in particular, fog, in this work).

2.2. Testing Autonomous Navigation Models with Weather Changes

Since the advent of Generative Adversarial Networks (GANs), there have been limited efforts that have explicitly attempted to analyze the effect of different weather conditions on steering angle prediction models in self-driving cars. DeepTest [35] used synthetic images generated using Photoshop to study the impact of rain and fog on the predicted steering angle. The manpower required for a large-scale deployment is prohibitive for such an approach. DeepRoad [40] tried to automate the same using generative models instead. DeepRoad learns the translation to different weather conditions; however, the sole goal for this work is to perform image-to-image translation with no explicit goal to 'fool' the model or obtain any minimum deviation of steering angle from the ground truth. In this work, we bring together adversarial and unpaired image-image translation perspectives to produce a minimum deviation in steering angle prediction in the produced images. We also show in Section 5 that our method to generate fog adversaries causes an average perturbation of nearly 1 radian($\sim$ 60 degrees), which can serve as a rigorous platform to test vision models in autonomous navigation.

2.3. Fog Generation

Adverse weather conditions such as fog are common in day-to-day life. There have been very few efforts based on image processing and filters to generate fog in images. We note that most related work in this direction based on image processing focus on defogging [22, 39], whereas our work is focused on the generation of fog. Li, et al [21] and Sakaridis et al[32] attempt to generate fog using image processing methods with a strong prior. A prior is usually carefully constructed based on handcrafted features, such as texture and brightness, from the image. Unfortunately, using such priors automatically restricts the approach to constraints on expected intensity, texture, and other features present in the image. Besides, in settings of autonomous navigation, handcrafted priors do not scale to the significant variations in scenes and environments, as well as significant variations in a single scene due to effects induced by factors such as light-and-shadow and fast motion. In this work, we attempt to provide an automated method to generate fog images that are intended to distort steering angle prediction models, to improve robustness of such models. We show later in this paper that performing adversarial training using the images generated by our method provides significant robustness to the model.

3. Methodology

When a carefully crafted perturbation is added to the image such that it causes the network to misclassify, we call it an adversarial perturbation (as in Sec 2.1). For a classifier network f and an input image $\mathbf{x}$; the perturbation, ϕ applied on it may be defined as:

$$\arg\max\ f(\phi(\mathbf{x})) \neq \arg\max\ f(\mathbf{x}) \tag{2}$$

To ensure visual similarity, the adversarial image should be within ϵ (very small) distance of the original input. This constraint is generally include to the above problem as:

$$\text{s.t } ||\phi(\mathbf{x}) - \mathbf{x}|| \leq \epsilon \tag{3}$$

In the case of an additive perturbation, we define ϕ as:

$$\phi(\mathbf{x}) = \mathbf{x} + \delta$$

We observe that this results in the same set of equations as earlier in Eqn 1.

We now use this definition to extend the typical form of adversarial perturbations, by allowing ϕ to be more than simple additive perturbations. We can define ϕ as a multiplicative noise or a filter, or a neural network itself can model it. We expect any transformation created by ϕ to be valid in such a setting as long it guarantees task-perceptual similarity, discussed below.

Task-Perceptual Similarity. In the case of images, the adversarial attacks performed are such that both the adversarial image $\phi(\mathbf{x})$ and original image $\mathbf{x}$ are perceived to be similar. This usually includes visual similarity like in the case of many popular attacks like FGSM [15] and JSMA

[27]. Recent works have shown that simple image transformations like rotation, scaling, and the translation are sufficient to fool the network [12][3]. These efforts do not emphasize complete visual similarity, yet we humans find them to be perceptually similar, i.e., they are interpreted in the same way by the human visual system. Following this, we can extend the definition of adversaries beyond visual similarity to something more intuitive: *task-perceptual similarity*.

To ensure the same, in Eqn 2, we expect the transformations created by ϕ to be such that the result predicted by humans for the given task is the same for both the original input and the adversary. The transformation need not necessarily guarantee visual similarity, but the task at hand needs to be perceived in the same way by the human. Examples of such transformations for images include contrast and brightness changes, blurring, rotation, scaling, sharpening, whitening, the addition of noise, etc. In all of these cases or a combination of these, humans are capable of perceiving the image similarly; this is, however, not the case with neural network models. One should note that visual similarity generally guarantees task-perceptual Similarity, but the converse need not hold. In Eqn 2, in order to guarantee visual similarity, one may simply add a constraint like that of Eqn 3.

3.1. Adversarial Attack on Regression Models

From the previous definition of adversarial perturbations, we may redefine it for a regression network, N as:

$$||N(\phi(\mathbf{x})) - N(\mathbf{x})|| \geq \theta \tag{4}$$

This implies we want the perturbation applied to the image $\mathbf{x}$ to cause a minimum deviation of θ. As stated earlier in Section 2.1, ϕ can be purely additive or another function captured by a neural network. We may add additional constraints on ϕ for visual similarity or sparsity. We can then simplify Eqn 4 to:

$$\begin{aligned} &||N(\phi(\mathbf{x})) - N(\mathbf{x})|| - \theta \geq 0 \\ \implies\ &\theta - ||N(\phi(\mathbf{x})) - N(\mathbf{x})|| \leq 0 \end{aligned} \tag{5}$$

Hence, we define regression loss, $L_{regress}$ required for creating an adversarial sample for input $\mathbf{x}$ as:

$$\min_{\phi} L_{regress} = \min_{\phi}(\theta - ||N(\phi(\mathbf{x})) - N(\mathbf{x})||) \tag{6}$$

By minimizing $L_{regress}$, we can find the perturbation, ϕ which ensures that a minimum deviation is caused for every input sample, $\mathbf{x}$.

3.2. Proposed Idea

Steering angle prediction models are fundamentally designed to be regression models, and take the input scene to predict the angle of the steering wheel in the range of $[-\pi, \pi]$. We have previously seen in Fig 1 that adverse weather conditions like fog affect the steering angle predicted by a model. Fog may be represented as a mixture of blurring and whitening on an image. We also observe that a foggy image is perceived in a similar manner as the normal one by a human, especially when it comes to tasks like steering angle prediction. To elaborate further, in Fig.1, we as humans expect the same steering angle for both the sets of images but the network does not. We leverage this fact to design a testbed for steering angle models using adverse weather conditions. Hence, combining the previous two sections, we can utilize weather conditions like fog to attack steering angle prediction models adversarially.

To achieve this goal, we define N as our steering angle predictor in Eqn 6 and ϕ as our adversarial weather generator for a given sample $\mathbf{x}$. We train a generator, ϕ to learn the transformation (Sec 3.3) from normal weather to foggy. The weights of ϕ will be obtained by minimizing Eqn 6.

We now have ϕ, a generator being modeled by a neural network and a continuous-valued discriminator N. The θ in the equation naturally becomes the minimum steering angle deviation we desire.

3.3. Loss Formulation

We train a CycleGAN [42] to learn the translation between the normal (sunny) weather (*Domain A*) and the adverse weather condition images (*Domain B*). ϕ_{AB} is the Generator responsible for the translation from Domain A to B and ϕ_{BA} for the reverse. Similarly, D_A is the Discriminator responsible for Domain A and D_B for Domain B. CycleGAN utilizes adversarial losses across its generated images. It also uses a cycle-consistency loss which ensures that when an image $\mathbf{x}$ belonging to Domain A, is translated from Domain A to B and back to Domain A, remains the same. i.e.,

$$\mathbf{x} \approx \phi_{BA}(\phi_{AB}(\mathbf{x}))$$

This also applies in a similar fashion for images from Domain B. Hence, CycleGAN loss in total consists of: [2]

$$\begin{aligned} L_{CycleGAN} = L_{cycle}(\phi_{AB}, \phi_{BA}) + \\ L_{adversarial}(\phi_{AB}, \phi_{BA}, D_A, D_B) \end{aligned}$$

We augment the CycleGAN losses with the regression loss (Eqn 6) and train this combined loss.

Hence the net loss that the CycleGAN is trained on:

$$L_{total} = (1-\alpha)L_{CycleGAN} + \alpha L_{regress} \tag{7}$$

$$\text{where, } L_{regress} = \theta - ||N(\phi_{AB}(\mathbf{x})) - N(\mathbf{x})|| \tag{8}$$

In the above equation α is a multiplier lying between $(0, 1)$. We summarize the working of the network in Figure 2 and also in Algorithm 1.

[2] For complete details of the CycleGAN loss, please refer Zhu et al[42].

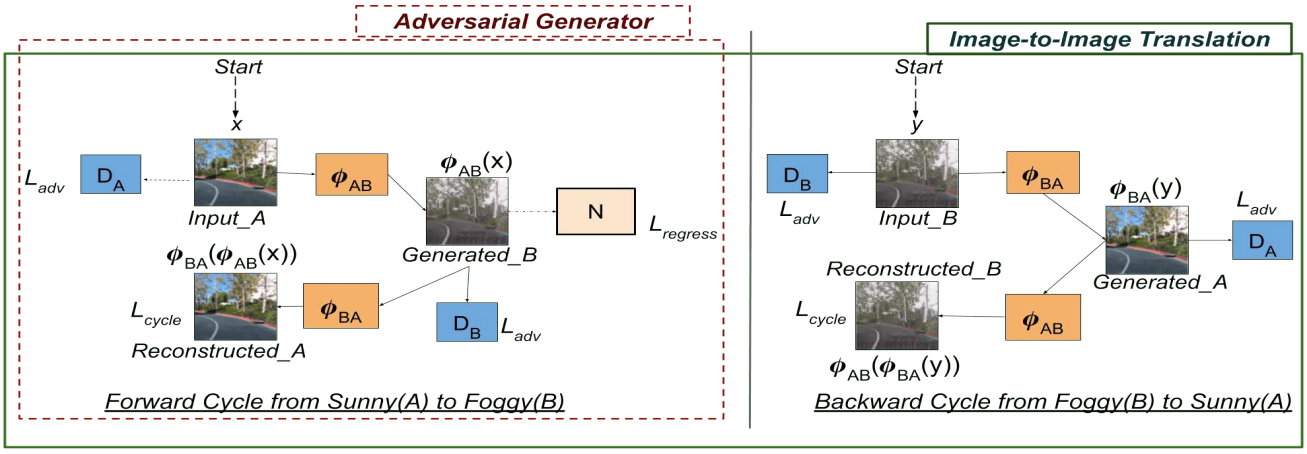

Figure 2: Summary of the framework followed to train the CycleGAN network to fool the steering angle predictor N. The image-to image translation part of the system learns the translation between Normal, Sunny weather(Domain A) and Foggy weather(Domain B). The adversarial generator part helps ϕ_{AB} generate foggy images which can cause a minimum deviation(θ) in the predictions of N.

One may note that we can use any other domain translation model instead of CycleGAN to achieve the same goal. To demonstrate the efficiency of our concept, we run our experiments on another recently popular model: DistanceGAN [4]. This model utilizes CycleGAN losses along with a distance loss to ensure that the distance between a pair of samples is maintained across both the domains.

Algorithm 1 CycleGAN training pseudo code

Input $X \leftarrow$ Training samples from Domain A
Input $Y \leftarrow$ Training samples from Domain B
Input $N \leftarrow$ Steering Angle Model to be Fooled
Input $T \leftarrow$ Number of epochs to train
Input $\theta \leftarrow$ Minimum deviation desired
Output $\{\phi_{AB}, \phi_{BA}, D_A, D_B\}$

procedure GENERATE()
  for t in $\{1...T\}$ **do**
    Draw m training samples $\{x_1, ..x_m\}$ from X
    Draw m training samples $\{y_1, ..y_m\}$ from Y
    for i in $\{1, ...m\}$ **do**
      $x \leftarrow x_i$; $y \leftarrow y_i$
      Compute: $\hat{y} \leftarrow \phi_{AB}(x)$ ▷ Forward Cycle
      Compute: $\hat{x} \leftarrow \phi_{BA}(y)$ ▷ Backward Cycle
      Compute the losses:
        $L_{regress}(\phi_{AB}, x)$ ▷ Eq. 8
        $L_{cycle}(\phi_{AB}, \phi_{BA}, x, y, \hat{x}, \hat{y})$
      Update Generators

      Compute Discriminator losses:
      $L_{adversarial}(\phi_{AB}, \phi_{BA}, D_A, D_B, x, y, \hat{x}, \hat{y})$
      Update Discriminators
    end for
  end for
end procedure

4. Implementation

Network Architecture: For the domain translation between normal to foggy weather, we use the CycleGAN model. The generators in the model use the same architecture involving Resnet-9 blocks as described in [42]. We also utilize similar architecture for the DistanceGAN model[4].[3] For the steering angle models, we use AutoPilot[8] which is an improvement on NVIDIA PilotNet model[5]. The model is adapted with an additional convolution layer for 128×128 images. We also perform similar experiments on the architecture developed by Comma AI [33].

Datasets used: There are many weather conditions which are capable of causing a large deviation in steering angle predictions. However, due to unavailability of datasets with heavy snow or rainy weather conditions, we restrict ourselves to foggy conditions [31]. We, however, note that our framework is generic and can be extended to other weather conditions.

We train the CycleGAN model[42] to learn the domain translation from normal, sunny weather to foggy weather conditions. The sunny weather condition images are obtained from the widely used SullyChen's dataset [8]. We chose this dataset over the Udacity dataset [36] due to its better quality. The Udacity dataset has many images with blank bright spots or hazy regions, making it difficult for the Generator to learn the translation to its equivalent foggy condition. Most other steering angle based datasets are either virtual [11] or of poor quality with minimal variations [33], and hence not suitable for this work.

Our foggy weather images (to train the CycleGAN) are taken from the Foggy Zurich dataset [31], which consists

[3]Project Webpage: https://code-assasin.github.io/little_fog/

of 3.7k high-quality images collected during the occurrence of fog in and around Zurich. The steering angle prediction model is trained on the Sullychen dataset [8], and the MSE is shown in Table 1.

Preprocessing: In order to train the CycleGAN, the normal weather images from Sully Chen's dataset [8] are sampled since many of the frames consist of similar scenes. After sampling (Systematic sampling with an offset of 12 frames), we obtain nearly 3.7k images for training the CycleGAN. The foggy images from [31] consist of a wiper and a dashboard in their scenes, something absent in the normal images dataset. Hence we crop the bottom few pixels of the foggy images to train the CycleGAN. Both the dataset images are resized to 128×128 pixels and then used to train the CycleGAN. For the steering angle models, however, we use the entire Sully Chen dataset [8], split into a train and test set. The images are resized to 128×128 and normalized between $[-1, +1]$.

Training details: We train the CycleGAN model along with the regression loss to learn the translation between normal and foggy weather conditions (Eqn 6). The parameters used in case of the CycleGAN losses are mostly the same as those mentioned in [42] except for the identity loss having a multiplier ($\lambda_{identity}$) of 3 . The model is then trained with both losses. The value of α is chosen to be 0.2 and θ is 0.5 radians (~ 30 degrees).

DistanceGAN model: We also train the DistanceGAN [4] model with similar preprocessing, as mentioned above. We used the hyperparameters as used in the code provided by [4], with the exception of the identity loss. We choose the value of α to be 0.07 and θ as 0.5 radians.

Model	Train Error	Test Error
AutoPilot[8]	0.0185	0.0448
Comma AI[33]	0.0198	0.0551

Table 1: Test + Train error for diff steering angle models

5. Results

We showcase the results of our trained model in Fig.3 along with the steering angle, predicted for each image. We observe that the second image has nearly a 180 degree change in the predicted angle, indicating how dangerous these adverse weather conditions can be, further encouraging the need for testbeds provided by methods like ours.

5.1. Subjective Image Quality Assessment

We assess the realism of the foggy images generated by the CycleGAN after being trained with the Regression Loss. We asked 10 participants to assess the quality of the generated foggy images. These participants had never seen the normal, sunny version of the image before and were asked to judge the realism of the foggy image provided to them. We use the foggy images from the Foggy Zurich dataset as the control for these experiments. We ran this experiment for different variations of the CycleGAN model used. We observed that on an average (for both steering prediction models), nearly 48% (44% for DistanceGAN) of the participants found the fog generated by utilizing Regression loss to be real enough compared to Foggy images from the Zurich dataset [31]. Additionally, we asked participants to choose the between the foggy images produced by CycleGAN (or DistanceGAN) and those produced when regression loss was added. We asked them to compare theses images based on their quality. We found that nearly 43% (48% for DistanceGAN model) of the people found that CycleGAN with regression loss produces better images. Considering this is nearly half the participants, we find that our method produces images of comparable quality as the original foggy images.

5.2. Objective Image Quality Assessment

We use standard Image quality assessment metrics like MSE, PSNR, SSIM[38] to compare the quality of the normal sunny image, w.r.t its foggy counterpart for different flavors of the models used. We report the results for the same in Table 3. In addition, we compare the image quality of foggy images produced using produced by CycleGAN alone and those produced when Regression loss was added (see Table 4).

We see from Tables 3 and 4 that the regression loss indeed causes a visible change in the image. Visibly also the quality of the images without regression loss seems to be better than those produced with it. To study the efficacy of regression loss, we compute the difference in steering angle predicted for foggy images produced by the CycleGAN model with and without regression loss, i.e., we compute:

$$||N(\phi_{foggy}(x)) - N(\hat{\phi}_{foggy}(x))||$$

ϕ_{foggy} is the CycleGAN Generator responsible for the translation from normal to foggy domain and $\hat{\phi}_{foggy}$ is the same generator trained with the Regression Loss. From Table 2, we can clearly see that the deviation produced with and without our loss is clearly very significant for both the CycleGAN and DistanceGAN model.

Method	Deviation Caused
Cycle vs Cycle+Regress (AutoPilot)	1.09 ± 0.9
Cycle vs Cycle+Regress (Comma AI)	0.76 ± 0.7
Distance vs Distance+Regress(AutoPilot)	1.31 ± 0.5
Distance vs Distance+Regress(Comma AI)	0.58 ± 0.54

Table 2: Deviation caused with regress loss in comparison to the original models (Calculated using Eq.8)

6. Ablation Studies

We notice in Eq.8 that there is a subtle balance between the CycleGAN and regression loss. While $L_{CycleGAN}$ is

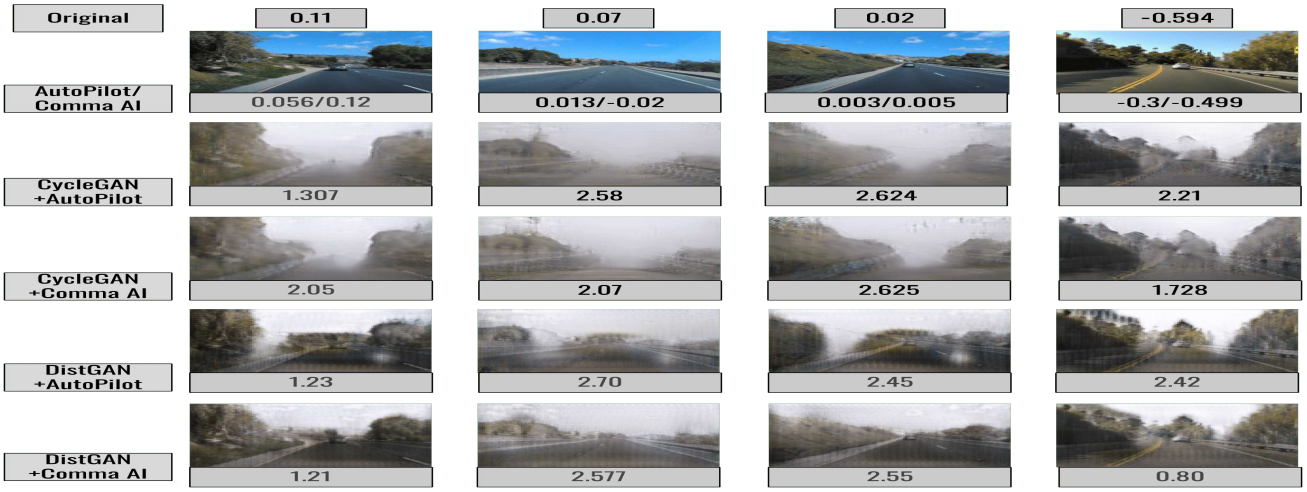

Figure 3: Fooling Models: Ground truth Steering Angle (in radians) for each of the original test samples. The angles right below indicate the ordered pair of predicted steering angle by AutoPilot and Comma AI respectively. From the second row onward, we indicate the image translation model used and respective steering model it was trained on. The angle below each of those images indicates the prediction by the steering model for the generated foggy image.

Figure 4: Variation in image quality of CycleGAN for different α values

Method	MSE	PSNR	SSIM
Cycle	3172 ± 1172.4	13.4 ± 1.8	0.51 ± 0.08
+Regress(AutoPilot)	3024 ± 1076	13.7 ± 1.9	0.52 ± 0.07
+Regress(Comma AI)	3111.2 ± 1086	13.5 ± 1.9	0.52 ± 0.08
Distance	2073.2 ± 663.4	15.18 ± 1.4	0.59 ± 0.05
+Regress(AutoPilot)	3292 ± 1085.8	13.2 ± 1.56	0.54 ± 0.07
+Regress(Comma AI)	2327 ± 811	14.7 ± 1.6	0.61 ± 0.058

Table 3: Objective IQA normal w.r.t respective foggy counterpart using different methods

responsible for the quality of the foggy images generated, $L_{regress}$ is responsible for causing the minimum deviation in the steering angle model. We can see that either loss overpowering the other is undesirable. Hence, a subtle balance must be created between them to achieve the end goal. To gain a deeper understanding of this, we explore variations in the hyperparameters α and θ. For all of the experiments, we use a CycleGAN[42] model along with the AutoPilot[8] model as the steering angle predictor.

Method	MSE	PSNR	SSIM
Cycle vs (AutoPilot)	802.9 ± 446.6	19.6 ± 2.04	0.64 ± 0.06
Cycle vs (Comma AI)	868.4 ± 491	19.3 ± 2.17	0.64 ± 0.07
Distance vs (AutoPilot)	2025.2 ± 1253	15.8 ± 2.56	0.52 ± 0.07
Distance vs (Comma AI)	$1362. \pm 786$	17.43 ± 2.36	0.57 ± 0.068

Table 4: Results using different I.Q.A methods. For each row we compare, the model and its counterpart trained with regression loss on the respective steering model.

Variation in α. We vary the hyper-parameter α (defined in Eq.8) to study its effect on the images generated by the CycleGAN. We choose a constant value of $\theta = 0.5$ radians for these experiments.

- $\alpha = 0.2$: With a decent value of α, as seen in Section 5, it produces fairly good results. The image quality of the CycleGAN remains pretty clear while causing the minimum required deviation.
- $\alpha = 0.5$: With a moderate value of α, although we expect to average results, the image quality of the Cy-

cleGAN is completely wrecked.

- $\alpha = 0.8$: With very high value of α, $L_{regress}$ is given more importance. This causes the output foggy images to have lower-quality than those obtained in Section 5.

While for $\alpha = 0.2$ it only takes 150 epochs to reach the desired goal, for higher values of α it seems to take more epochs to converge. For $\alpha = 0.5$ it took nearly 200 additional epochs to settle while the losses kept oscillating for 0.8. We can also see the variation in their quality in Fig 4.

Variation in θ. We vary the minimum deviation θ in the Eq. 8 to see its effect on the generated images. For all experiments with changes in θ, we fix our α value to 0.2.

- $\theta = 0$ radians : In this case, we would ideally expect the $L_{regress}$ to be dominated by $L_{CycleGAN}$ but we observe that in a matter of few epochs the $L_{regress}$ values start becoming highly negative hence the net loss function start shifting its attention towards minimizing the regression loss without paying heed to the CycleGAN losses. This causes a lot of deterioration in the image quality of the foggy images.
- $\theta = 0.5$ radians : As seen in Section 5, this value of θ relayed good results both in terms of image quality and deviation produced.
- $\theta = 1$ radians: With a higher value of θ the number of epochs taken for convergence was much higher with nearly similar quality as those obtained with $\theta = 0.5$. Hence, we prefer to reduce our costs by utilizing $\theta = 0.5$. In addition, $\theta = 0.5$ itself is causing an average deviation of 1 radian (Table 2) hence reaping the benefits without additional costs.

Similar trends, for θ and α, are seen using the DistanceGAN [4] model.

Regression loss on Backward Generator ϕ_{BA}: From Eq. 8 we can observe that the Regression loss is applied only to the forward cycle of the CycleGAN. Our goal is only to create adverse weather conditioned images and not vice-versa. Hence the Regression loss is only applied on ϕ_{AB}. To test if adding a regression loss on backward cycle might help, we add the following term to Eq. 8:

$$L_{Bregress} = \theta - ||N(\phi_{BA}(y)) - N(y)||$$

Where y belongs to Domain B(foggy weather). This loss is then multiplied with α, same as in Eq. 8. We train the CycleGAN model in conjunction with these losses. As seen in Fig. 5, the results of this model seems to be distorted, and of poor quality. We also observed that the loss seemed to oscillate. We believe that this occurs because backward regression loss forces the normal images generated to cause distortion. There is a good chance (as in Sec 6) that the regression loss dominates over the CycleGAN loss and hence, might even generate blank images which are perfectly capable of causing the desired deviation in the steering angle. To prevent this from happening, we ensure that the generated image is perfectly capable of being returned to the original (Eq. 3.3). We hence do not include the adversarial loss term in ϕ_{BA}.

Figure 5: Original images (left) and foggy counterparts (right) generated using regression loss in both directions

Defending against Fog: We train the AutoPilot and Comma AI models with the foggy counterparts of the training dataset [8], generated using the proposed method. We then test the model on the foggy counterpart of the test set. From Table 5, there is a clear improvement in the deviation caused once the model is adversarially trained using the foggy images generated by our method.

Method	Trained on	Deviation Caused
AutoPilot	Normal	1.81 ± 1.03
AutoPilot	Foggy	0.17 ± 0.4
Comma AI	Normal	1.88 ± 0.84
Comma AI	Foggy	0.2 ± 0.7

Table 5: Deviation caused with regress loss in comparison to the original models

7. Conclusions

As part of our work, we introduced a more generic definition of adversarial perturbations. Our definition makes use of the more understandable Perceptual Similarity rather than Visual Similarity. We have also introduced a manner in which adversarial perturbations may be used to fool regression-based networks such that they cause a minimum deviation. We then showed how this could be applied to steering angle models to generate sufficient Fog to produce a minimum deviation in the scene. In the future, with the help of better datasets, we would like to construct more stringent testbeds to evaluate the credibility of autonomous navigation.

References

[1] E. Ackerman. *Korean Competition Shows Weather Still a Challenge for Autonomous Cars*, 2014 (accessed July 25, 2019). 2

[2] N. Akhtar and A. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018. 2

[3] A. Azulay and Y. Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv preprint arXiv:1805.12177*, 2018. 4

[4] S. Benaim and L. Wolf. One-sided unsupervised domain mapping. In *NIPS*, 2017. 2, 5, 6, 8

[5] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. 1, 5

[6] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017. 2, 3

[7] N. Carlini and D. Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 1–7. IEEE, 2018. 2

[8] S. Chen. Autopilot-tensorflow, 2016. *URL https://github.com/SullyChen/Autopilot-TensorFlow*, 2016. 1, 5, 6, 7, 8

[9] S.-T. Chen, C. Cornelius, J. Martin, and D. H. P. Chau. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 52–68. Springer, 2018. 2

[10] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[11] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017. 5

[12] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017. 4

[13] I. Evtimov, K. Eykholt, E. Fernandes, T. Kohno, B. Li, A. Prakash, A. Rahmati, and D. Song. Robust physical-world attacks on deep learning models. *arXiv preprint arXiv:1707.08945*, 1:1, 2017. 2, 3

[14] G. Garibotto, P. Bassino, M. Ilic, and S. Masciangelo. C.1 - computer vision for autonomous navigation: From research to applications. In V. Cappellini, editor, *Time-Varying Image Processing and Moving Object Recognition, 4*, pages 51 – 56. Elsevier Science B.V., Amsterdam, 1997. 1

[15] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2, 3

[16] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. 1

[17] M. Hebert. Computer vision for autonomous navigation. Technical report, Carnegie Mellon University, June 1988. 1

[18] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song. Natural adversarial examples, 2019. 2

[19] J. Janai, F. Güney, A. Behl, and A. Geiger. Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art. *CoRR*, abs/1704.05519, 2017. 1

[20] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016. 2

[21] B. Li, W. Ren, D. Fu, D. Tao, D. Feng, W. Zeng, and Z. Wang. Benchmarking single-image dehazing and beyond. *IEEE Transactions on Image Processing*, 28(1):492–505, 2018. 3

[22] Y. Li, S. You, M. S. Brown, and R. T. Tan. Haze visibility enhancement: A survey and quantitative benchmarking. *Computer Vision and Image Understanding*, 165:1–16, 2017. 3

[23] W. Maddern, G. Pascoe, C. Linegar, and P. Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017. 1

[24] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 2

[25] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018. 1

[26] 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*, pages 1765–1773, 2017. 2

[27] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016. 2, 4

[28] K. Reddy Mopuri, U. Ojha, U. Garg, and R. Venkatesh Babu. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 742–751, 2018. 2

[29] G. Report. Korean competition shows weather still a challenge for autonomous cars, 2016. https://spectrum.ieee.org/cars-that-think/transportation/advanced-cars/japan-competition-shows-weather-still-a-challenge-for-autonomous-cars. 3

[30] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016. 1

[31] C. Sakaridis, D. Dai, S. Hecker, and L. Van Gool. Model adaptation with synthetic and real data for semantic dense foggy scene understanding. In *European Conference on Computer Vision (ECCV)*, pages 707–724, 2018. 5, 6

[32] C. Sakaridis, D. Dai, and L. Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126(9):973–992, 2018. 3

[33] E. Santana and G. Hotz. Learning a driving simulator. *arXiv preprint arXiv:1608.01230*, 2016. 1, 5, 6

[34] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540. ACM, 2016. 2

[35] Y. Tian, K. Pei, S. Jana, and B. Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*, pages 303–314. ACM, 2018. 1, 3

[36] Udacity, 2016. https://github.com/udacity/self-driving-car. 5

[37] G. Varma, A. Subramanian, A. Namboodiri, M. Chandraker, and C. Jawahar. Idd: A dataset for exploring problems of autonomous navigation in unconstrained environments. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1743–1751. IEEE, 2019. 1

[38] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 6

[39] Y. Xu, J. Wen, L. Fei, and Z. Zhang. Review of video and image defogging algorithms and related studies on image restoration and enhancement. *IEEE Access*, 4:165–188, 2016. 3

[40] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid. Deeproad: Gan-based metamorphic autonomous driving system testing. *arXiv preprint arXiv:1802.02295*, 2018. 3

[41] Y. Zhang, H. Foroosh, P. David, and B. Gong. CAMOU: Learning physical vehicle camouflages to adversarially attack detectors in the wild. In *International Conference on Learning Representations*, 2019. 3

[42] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017. 2, 4, 5, 6, 7