Multimedia Tools and Applications
https://doi.org/10.1007/s11042-024-19409-z

1245: APPLICATION OF MULTI-SENSOR FUSION TECHNOLOGY FOR AUTONOMOUS DRIVING

Employing cross-domain modelings for robust object detection in dynamic environment of autonomous vehicles

Oshin Rawlley[1] · Shashank Gupta[1] · Hardik Kathera[1] · Siddharth Katyal[1] · Yashvardhan Batwara[1]

Received: 14 July 2023 / Revised: 18 March 2024 / Accepted: 9 May 2024

Abstract
Object detection (OD) in Advanced Driver Assistant Systems (ADAS) has been a fundamental problem especially when complex unseen cross-domain adaptations occur in real driving scenarios of autonomous Vehicles (AVs). During the sensory perception of autonomous Vehicles (AV) in the driving environment, the Deep Neural Networks (DNNs) trained on the existing large datasets fail to detect the vehicular instances in the real-world driving scenes having sophisticated dynamics. Recent advances in Generative Adversarial Networks (GAN) have been effective in generating different domain adaptations under various operational conditions of AVs, however, it lacks key-object preservation during the image-to-image translation process. Moreover, high translation discrepancy has been observed with many existing GAN frameworks when encountered with large and complex domain shifts such as night, rain, fog, etc. resulting in an increased number of false positives during vehicle detection. Motivated by the above challenges, we propose COPGAN, a cycle-object preserving cross-domain GAN framework that generates diverse variations of cross-domain mappings by translating the driving conditions of AV to a desired target domain while preserving the key objects. We fine-tune the COPGAN training with an initial step of key-feature selection so that we realize the instance-aware image translation model. It introduces a cycle-consistency loss to produce instance specific translated images in various domains. As compared to the baseline models that needed a pixel-level identification for preserving the object features, COPGAN requires instance-level annotations that are easier to acquire. We test the robustness of the object detectors SSD, Detectron, and YOLOv5 *(SDY)* against the synthetically-generated COPGAN images, along with AdaIN images, stylized renderings, and augmented images. The robustness of COPGAN is measured in mean performance degradation for the distorted test set (at IoU threshold = 50) and relative performance degradation under corruption (rPD). Our empirical outcomes prove a strong generalization capability of the object detectors under the introduced augmentations, weather translations, and AdaIN mix. The experiments and findings at various phases intend to the applicability and scalability of the domain adaptive DNNs to ADAS for attaining a safe environment without human intervention.

Keywords Object detection · Deep neural networks · Generative adversarial networks · Cross domain adaptation

Extended author information available on the last page of the article

Published online: 31 May 2024

1 Introduction

Cyber-physical systems (CPS), which connect the physical and cyber worlds using intelligent devices, are now omnipresent in the digital realm [1]. The latest commercial cars are equipped with advanced driver assistance systems (ADAS) to improve safety in driving and adaptive cruise control for enhanced driving comfort [2]. Numerous autonomous vehicle competitions have led to many pioneering advancements in AV technology for urban driving situations. The champions of these competitions, for example, the autonomous prototype car named Boss possess abilities in perception, localization, route planning, and motion control [3]. For the implementation of these functionalities, deep-learning techniques have progressed as the vital driving forces for the sensor-captured data to make guided decisions [4]. Figure 1 depicts the challenges the AVs face during the adverse scenarios such as rainy, winter etc. We have taken the adverse environments as operational environments in our paper (Tables 1 and 2).

Further, the safety of autonomous driving is of utmost importance. However, deep learning algorithms are typically not 100% accurate and can be influenced by many factors such as temperature, weather conditions (e.g., fog and rain), and driving environment (for instance, ambient traffic flow) [5]. Figure 2 depicts the object detection in a cloud-edge environment. Object detection takes place with the help of RoadSide Units (RSUs) equipped with edge servers (ES). Hence, it is vital to thoroughly validate the perception system across various operational scenarios in order to ensure its resilience in unforeseen environments. Although numerous automotive companies have been conducting on-road tests of their AVs for a considerable time, it can be stated that such testing is restricted due to a scarcity of data, particularly in rare extreme operational conditions. To address this issue, various validation approaches are suggested and are largely divided into three classes: model-based techniques, simulation-based techniques, and data-driven techniques. *Model-based approaches*: The goal of model-based approaches is the creation of image data which is done by employing analytical physical models whose task is to replicate various weather conditions similar to summer, rain, and others. However, these approaches frequently struggle to capture substandard

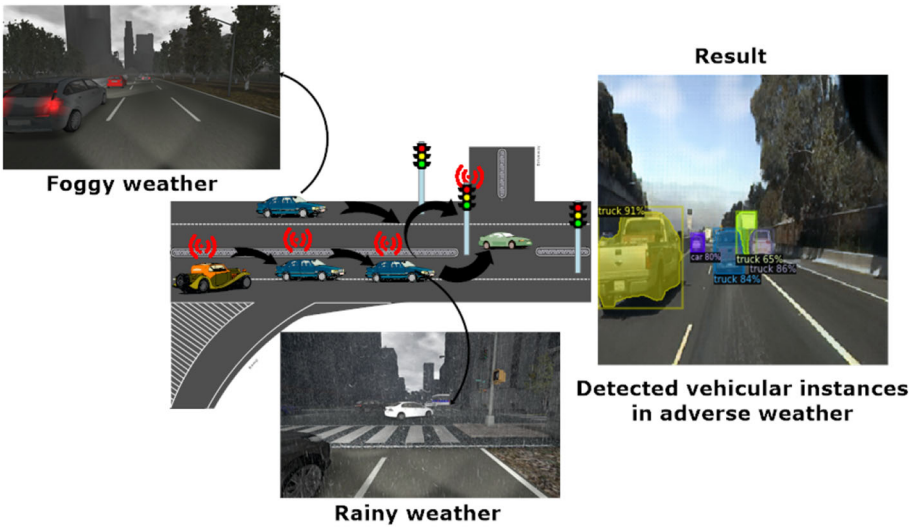

Fig. 1 Adverities in the autonomous diving enevironment

Table 1 Notational meanings

Notation	Meaning
O_{BB}	Bounding boxes and labels
Gen and Disc	Parameterized generator and discriminator respectively
z	Shared latent space
$\boldsymbol{X_1}$	Image domain 1
$\boldsymbol{X_2}$	Image domain 2
$\boldsymbol{P_{X_1\ X_2}(x_1,\ x_2)}$	Joint distribution space
$\boldsymbol{(x_1,\ x_2)}$	pair of images
$\boldsymbol{P_{X_1}(x_1)}$	marginal distributions of domain 1
$, P_{X_2}(x_2)$	marginal distributions of domain 2
$E_1 and\ E_2$	Encoding functions
$G_1 and G_2$	Generative functions to map latent codes to images
$\boldsymbol{p = [p_1,\ p_2,\ \ldots p_k]^T}$	set of representative scenic parameters
$\boldsymbol{O_I}$	object of interest
$\boldsymbol{Obj_{cent}(x_{cent},\ y_{cent})}$	Center point of an object in an image I
$\boldsymbol{C(x_c,\ y_c)}$	Center point of a complete image
w	width
h	Height
$\boldsymbol{Imp_{score}}$	importance score of an object
$\boldsymbol{Key_{val}}$	key score proportion
$\boldsymbol{F_m(p)}$	number of feature maps over continuous pixels
Wt	weight matrix
$\boldsymbol{B_v}$	bias vector
$\boldsymbol{\lambda}$	index number of convolution kernel
$\boldsymbol{F_R}$	reconstructed feature maps
B	arbitrary small batch
$\boldsymbol{B_s}$	maximum batch size
C	Channel
$\boldsymbol{\alpha}$	user-defined factor
$\boldsymbol{o_p}$	user-defined parameter
$\boldsymbol{X^m_{syn}}$	synthetic image
$\boldsymbol{X^m_{idt}}$	Identity image
$\boldsymbol{X^m_{rec}}$	Reconstructed image
$\boldsymbol{Loss_{adv}}$	adversarial loss
$\boldsymbol{Loss_{cyc}}$	Cycle consistency loss
$\boldsymbol{Loss_{idt}}$	Identity loss
$\boldsymbol{D^{ideal}}$	Ideal discriminator
$\boldsymbol{G^{ideal}}$	Ideal generator
$\boldsymbol{\omega_{adv}}$	Weight controlling parameter for adversarial loss

Table 1 continued

Notation	Meaning
ω_{cyc}	Weight controlling parameter for cyclic loss objective
ω_{idt}	Weight controlling parameter for identity loss
E^{m}_{adv}	output error
$E^{m,l}_{adv}$, E^{m}_{cyc}, E^{m}_{idt}	pixel-wise recovery error
f_s, f_t	feature space representations of source and target
σ, μ	Mean and variance

Table 2 Main abbreviations

Abbreviation	Explanation
DA	Domain Adaptation
OD	Object Detection
GAN	Generative Adversarial Network
ML/DL	Machine Learning/ Deep Learning
ES	Edge Server
SDY	SSD, Detectron and YOLO
CPS	cyber-physical systems
ADAS	Advanced Driver Assistant Systems
COPGAN	operational GAN
DNN	Deep Neural Networks
UNIT	Unsupervised Image-to-Image Translation
VAE	Variational Autoencoders
NN	Neural Networks
AdaIN	adaptive instance normalization
TB	Tera Byte
CNN	Convolutional Neural Network
LQI	Low-quality Image
MLE	maximum likelihood estimation
GB	Gausian Blur
GS	Gray Scale
IT	Invert Transform
ST	Solarise Transform
JT	Jitter Transform
mAP	Mean average precision
AP	Average precision
MOS	Mean opinion scoere
IoU	Intersection over Union
mPD	mean performance degradation
rPD	relative performance degradation
S2W	Summer to Winter translation
S2N	Summer to Night translation

Table 2 continued

Abbreviation	Explanation
S2R	Summer to Rain translation
S2Aug	Summer to Augmented translation
S2AdaIN	Summer to AdaIN translation
P2W	PASCAL VOC to Winter translation
P2N	PASCAL VOC to Night translation
P2R	PASCAL VOC to Rain translation
P2Aug	PASCAL VOC to Augmented translation
P2AdaIN	PASCAL VOC to AdaIN translation
CUDA	Compute Unified Device Architecture

effects, resulting in a significant loss of accuracy [6]. *Simulation-based methods*: For the production of virtual image data, the numerical algorithms-driven compound physical models in simulation engines are applied by simulation-based methods. Similar to the methods based on models, numerous simulations rely on less complex models and, as a result, fail to capture the intricate details of the real-time scenes. *Data-oriented methods*: On the other hand, use generative adversarial networks (GANs) to produce images synthetically for different operational conditions. CycleGAN has been utilized to regularize the process of training on the basis of cycle-consistency loss. Nevertheless, methods based on GAN require a sufficient amount of data for training from two distinct operational scenarios [7].

Moreover, as per the statistical survey conducted by the National Highway Traffic Safety Administration, the overwhelming majority of accidents, specifically 95%, are the result of errors made by human beings [8]. Nevertheless, in ADAS, we tend to ignore most of these errors leading to issues related to the inaccurate detection of objects by object detection models [9].

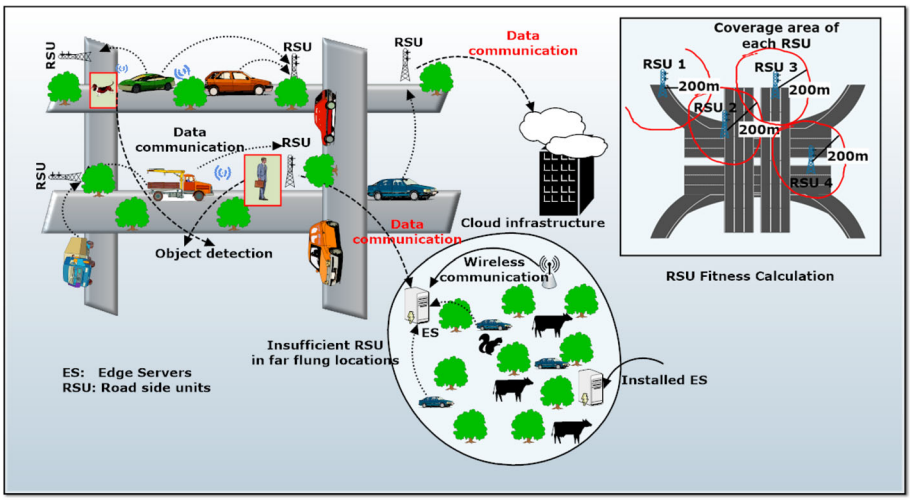

Fig. 2 A cloud-edge computing framework toward Internet of Vehciles

There are many object detection models in autonomous driving that have made substantial breakthroughs in locating vehicles on the road. These models are in two categories: one-stage and two-staged detectors. We present a state-of-the-art comparison of the features of both categories keeping in mind two parameters. One is the speed and high accuracy that are required by the AVs, which are delivered by the one-stage detectors. Secondly, the authors try to address the data imbalance problem in the two-staged detectors. Moreover, it also caters huge quantity of ground truths across the testing set of images along with the large number of poor-quality RPN proposals. However, the existing works have not investigated the accuracy and efficiency trade-off in the existing object detectors under unprecedented environments [10]. For this purpose, the existing ML/DL data-hungry models demand diversity in their representative data during their learning process [11]. There might be scenarios that the AVs must have never seen in their training regime. Consequently, facing a lot of unpredictable challenges contribute to the failure of many object detectors to generalize competently enough to the unobserved testing scenarios. These challenges are affected by the change in the settings of the training and testing data [12]. One of the examples is the camera used in the training of one AV which can be way different from the ones already installed in the vehicles. One of the other primary factors in this context concerns the varied weather conditions. For instance, datasets like KITTI, Cityscapes, are designed in clear weather conditions, ignoring real-world problems like fog, rain, smog, etc.

We explain this concept mathematically as follows: *Statement:* x is some training instance where $x \in X$; X is the dataset used for training and O_{BB} depicts the result containing anchor boxes and its respective classification labels for x image. *Conjecture:* The testing images are also taken from the data distribution X. Although, in real go-implementation, the scope of the testing data will not be limited to X, owing to the different factors such as camera shake, blurring, or defocus. Therefore, in these scenarios, the trained DNNs on data distribution X cause inaccurate detection of objects. These blunders restrict the detection process and can result in many concerns, such as AVs failing to detect pedestrians or persons misclassified as animals. Another point of concern is the budget and the effort essential for categorizing the data in which the highly reliable simulators such as CARLA, LGSVL, etc who are generally suffering from domain shift. This causes the failure in generation of the data to coordinate exactly with the physics of real-world sensors. *Emergence of GAN*:To address this domain shift, the recent propagation of GANs [4] has facilitated the generation and augmentation of synthetic instances, which appear quite realistic. The GANs are also termed two-player mini-max game [13] and are employed to generate fake images along with different data augmentation techniques related to the image-processing methods. This is intended to improve the generalizability of object detection models when deployed in novel environments, unlike some training scenarios. The objective function of this two-player mini-max game is formulated with the value function $V(Disc, Gen)$ and is shown in (1). We must maximize the probability of designating accurate labels to the training samples and the examples from Generator Gen for which we train Discriminator $Disc$.

$$\begin{aligned} \underset{Gen,Disc}{minimax}(Disc, Gen) = \underset{Gen,Disc}{minimax} E_{x \approx samp_t(x)}[log\, Disc(x)] \\ + E_{z \approx samp_z(x)}[log(1 - Disc(Gen(z)))] \end{aligned} \tag{1}$$

In ADAS, there are many complex domain shifts that the object detectors witness. Even after successfully adopting the GAN approaches, they fail to preserve the key objects and adapt sufficiently to the target domain. In addition, there prevails class imbalance where the accuracy is scaled down on specific or limited classes. *Key Object Preserving Strategy*: Inspired by the disadvantages discussed above, we design a key object preservation strategy

which emphasizes on retaining the important objects that appear in the video. Generally, to detect the foreground instances in a video, the conjecture holds that longer the object lingers in front of the eyes and, larger the cross-sectional area it occupies it gives a subjective feeling of the object being centrally located and an important foreground object. In this manuscript, we use the area size of the object and central point of the video frame as two determining factors. The closer the vehicle is to the center point, the more importance the vehicle holds. We calculate Imp_{score} and subsequently, Key_{value} for each vehicular instance present in the video.

The design of our new COPGAN model is inspired by the drawbacks discussed above and its goal is to address the issue of cross-domain shifts for time-critical applications similar to perception tasks in AV. COPGAN is not tailored specifically to any application and can generalize to all existing GAN architectures [14–19] by providing different inter-domain transformations for improving the robustness of the OD frameworks [20]. Moreover, in this manuscript, we aim to assess the generalization capability of the various OD models on our unfamiliar COPGAN synthetic cross-domain data and other variations. Unlike some previous works [16, 17], we have also grouped all the other variations such as AdaIN fast style transfer, stylized renderings and, augmented data and, call it a combined dataset. *Cycle-object preserving*: Moreover, the COPGAN introduces novel cycle-object preserving to retain the instance-level features all through the process of image-to-image translation. It means the vehicle detector is directly employed in generator training process directing the vehicular instances in the converted images retain the realistic appearances of the target domain through different domains. This helps in not including the object detector to be not included at the test time. COPGAN uses the object detector in the target-domain rather than an off-the-shelf object detector. Further, we have considered OD techniques of two categories namely: two-stage detector (Detectron) and one-stage detector (single-shot multi-box detector and YOLOv5). These models are fine-tuned separately using the original PASCAL VOC dataset, nuScenes [21], and with certain data augmentations. We have also taken the SYNTHetic collection of Imagery and Annotations (SYNTHIA) dataset as a source dataset for producing weather variations using the proposed COPGAN framework. The evaluation of the generalization and robustness of these trained algorithms is carried out separately for each type of variation produced. Contrasting to prevailing state-of-the-art datasets, we have taken into account both time diversity and adverse weather diversity for constructing a classified cross-domain dataset having diverse images of fog, rain, and night. We list out our major objectives and contributions as follows:-

- We suggest a novel cycle-object preserving cross-domain framework (also referred as COPGAN) for the synthetic generation of cross-domain images. COPGAN incorporates a single generator and a single discriminator, which cooperatively works in an adversarial fashion during training. This system enhances the robustness of the object detectors making it suitable for the AV applications by enabling them to learn the adverse variations in the wild.
- A combined dataset is generated and utilized comprising five different augmentation techniques, four approaches of AdaIN fast style transfer, and three operational weather domain shifts (generated by COPGAN) rendered to AdaIN to satiate and train the data-hungry OD models for generalizing well in real world scenarios. This combined dataset handles the problem of adverse data insufficiency in inter-domain vehicular detection.
- We also conduct extensive performance evaluations and statistical assessments to validate the suitability of the COPGAN-generated synthetic images and our combined dataset in the inter-domain sphere.

- Lastly, we investigate the generalizability of different categories of object detection models on varying quality of images based on average precision, total loss, and mean opinion score (MOS). Furthermore, the exhaustive and robustness evaluations in different scenarios are conducted to keep the fairness of considering maximum unseen domains. These experiments could be useful in the AV domain to employ OD models in all-around-the-clock variations.

The paper is divided into sections: Section 2 elaborates the Related Work and the limitations of the existing GANs, Section 3-4 describes the proposed COPGAN architecture, Section 5 discusses its loss function, Section 6 elucidates about various scenario settings during the experiment, Section 8 conducts robustness and exhaustive evaluation over the various scenarios discussed in the previous section. Further, Section 9 comprehends of the individual loss functions in the COPGAN framework. Summarizing the complete paper, we present Section 10 for a discussion and analysis. Finally, Section XI reads the conclusion of the paper.

2 Related work

This section discusses the DA in the AV domain, the existing datasets used by the OD models in different domains, GAN-generated synthetic instances, and the underlying concept of UNIT. Lastly, the limitations of GANs are discussed in a tabular form.

There is a wide acknowledgement of the susceptibility of deep neural networks (DNNs) to various discrepancies in the quality of the images across various image classification tasks. The recent literature has revealed that the DNN performance constantly drops on lower-quality images. For e.g., existing OD models in ADAS often fail to recognize vehicular instances in unclear weather. However, gathering varied operational instances can be costly and laborious, but they are helpful for the DA. In addition the OD models failed to recognize vehicles on the road which is authenticated by the state-of-the-art datsets that are constructed in perfect daylight conditions. Furthermore, in the field of AVs, the goal of the OD models is to localize vehicular components and distinguish between them. There have been significant advancements in OD methods recently, primarily due to the application of neural networks. Table 3 reviews the key characteristics of the recent OD methods. Additionally, the GANs have utilized UNIT. Nevertheless, they are successful only in small domain shifts. Since the DNN-based OD models presume that the two working domains (training and testing) are the same, deep-domain adaptive methods have come into play to reduce the domain discrepancy by fine-tuning them with *LQI*. The goal of the object detectors is to localize the vehicular instances and distinguish between them. In recent times, significant progress has been made in OD methods through the application of neural networks, specifically in the two categories of detectors: one-stage and two-stage detectors. These advancements have brought about substantial quantum leaps in the field. We have listed the notational meanings and the main abbreviations realized throughout the manuscript in Tables 1 and 2.

2.1 Domain adaptation

Recently, a subset of transfer learning known as domain adaptive learning has grown in popularity [28]. To address the variations between datasets originating from different domains, domain adaptation utilizes the additional training information provided by unlabeled data in the target domain. Several domain adaptation techniques, such as maximum

Table 3 Comparison of recent OD models with their advantages and disadvantages

Model	Backbone	Input size	FPS	VOC	RP	Pros	Cons
RCNN [22]	AlexNet	227 × 227	< 0.1	53.3% (12)	SS	Records higher accuracy over existing object detectors	Computation intensive and expensive training which slows them in real-world deployments
Fast RCNN [23]	VGG16, AlexNet	Random	< 1	68.4% (12)	SS	RoI pooling layer is proposed	Less speedy in real-world deployments
Faster RCNN [24]	VGG16, ZFnet	Random	< 5	70.4% (12)	RPN	Introducing RP+ Fast-RCNN	Complex training process
Mask RCNN [25]	ResNet101	Random	< 5	50.3% (7)	RPN	Predicts object masks by extending Faster-RCNN; Proposing FPN	Cannot achieve real time speed
YOLO [26]	GoogLeNet	448 × 448	< 50	66.4% (12)	–	First fast detector (one-stage) with no RP phase	Recorded low accuracy as compared to the recent detectors
SSD [27]	VGG16	300 × 300, 512 × 512	< 60	74.9% (12)	–	High speed and accuracy as compared to YOLO	Fails in detecting smaller objects

mean discrepancy, that aim to reduce the shift in the domain by decreasing the numerical differences between the depth features were successful in the classification of images and semantic segmentation [29]. Wang et al. [30] introduced the simple transfer learning (EasyTL) approach, which aims to acquire non-parametric transfer features and image classification classifiers.

2.2 Existing autonomous driving datasets

The two categories of object detectors i.e., one-stage and two-stage are implemented over a variety of latest datasets for AVs [31, 32]. In general, the datasets are built over daytime images. LISA 2010 dataset comprises three sequences of AV's video. Urban Traffic Surveillance (UTS) dataset offers images of vehicles taken by surveillance cameras [33]. Compcars dataset is composed of numerous samples of classified cars with anchor boxes, five attributes, and viewpoints [34, 35]. The Cars dataset comprises more than 100 classes of cars [36]. PASCAL VOC also provides annotated buses and cars in even more challenging and diverse scenarios in many respects, such as different viewing angles, varied distances, sizes, and aspect ratios [37]. However, these datasets rarely have images of the frontal angle or real-time scenarios of driving. KITTI dataset is specifically premeditated for AV driving, containing each image captured in a real driving scenario; however, all images are of the same daytime domain [38]. ITRI datasets comprising ITRI-Day and ITRI-Night have practical driving scenarios collected at night and daytime. SYNTHIA dataset is built from fake driving scenes having domains such as evening, night, morning, and dusk. For more realistic stylized images, the GTA dataset contains images in the snow, day, night, rain, and sunset conditions [39, 40]. While autonomous vehicle (AV) datasets have helped to advance the development of AV technology, there are some potential drawbacks to existing AV datasets, such as limited diversity, lack of real-world complexity, annotation quality etc.

2.3 GAN-generated synthetic data

In ADAS, OD models often a witness a large discrepancy between the training time and the actual scenes, i.e., the testing phase. This inconsistency results in a significant drop in performance in AV detection and can frequently occur in different forms, such as weather scenarios, occlusion, background clutter, illumination conditions, quality of the data, etc. The construction of diverse datasets can only, to some extent, cater to the crucial factor of object detection since, it is difficult to capture some extreme climatic changes. Furthermore, gathering and building big databases of images are generally considered to be complex tasks and expensive. Elimination of this problem has recently emerged into the picture through unsupervised and supervised domain adaptation methods. Authors in [41] built dataset which contained rainy image samples from the current large datasets by drenching a glass pane with sprays and capturing through a camera. Georg Volk et al. [42] solved such issues by combining the input data with these synthetically generated rainy images. In addition, Cycle GAN was used to transform the domain from day to night for training the model using a synthetic dataset and calculating its performance [43]. However, the real-world scenarios are different as such test dataset may not be of the same domain as the training set. Furthermore, distribution mismatch also prevails between the actual dataset built under varied setups or biases. Hence, we must also pay attention to this unseen environment mismatch.

Moreover, in the recent times, Pix2Pix have done significant advancements in the paired image-to-image translation [44, 45]. In this view, the generator transforms the input images

into the expected synthetic image. Nevertheless, retrieving the labeled image pairs in different scenes, such as snow, day, night etc., is quite challenging [18]. Recently, DiscoGAN [46], CycleGAN [10], and DualGAN [47], also called the unpaired image-to-image translation approaches, have facilitated the GAN training probable without paired data and introduced the cycle-consistency constraint [47].

2.4 Unsupervised image-to-image translation (UNIT)

The objective of UNIT is the comprehension of the shared space between two image distributions consisting of different domains.This is achieved by obtaining images from the two distinct domains' marginal distributions [48]. Computer vision challenges are being framed as problems of translating images from one domain to another, also known as image-to-image translation. This challenge involves mapping a single image from one domain to a corresponding image in a different domain. Such challenges exist in both unsupervised and supervised learning scenarios. In the unsupervised case, we have two sets of images that are unpaired, where one set has the image samples of one domain and the other set consists of image samples of another domain. However, we cannot guarantee the translation of an image from one domain to another when there is a lack of paired examples. Therefore, it is condemned as a harder problem to implement, but the training data collection is easier [49]. Hence, we propose the concept of a shared latent space that comprises pairs of corresponding images from distinct domains. This pair can be mapped to a common representation within the shared latent space. On the basis of the assumption, authors have proposed a UNIT framework grounded on Variational Autoencoders (VAEs) and GANs [50]. UNIT is self-possessed by the variational auto-encoders (VAE) and the GAN model. Each image is sampled using a VAE-GAN. An interaction between the objective of adversarial training persists with the constraint of weight sharing. This also administers a mutual latent space that generates corresponding samples of images in two different domains. The VAEs correspond and compare the converted images with the input images in their respective domains.The same has been implemented to the domain adaptation challenge. The average precisions on the conventional datasets PASCAL VOC, SYNTHIA, and nuScenes was realized. The authors [15] elaborate on the process of learning the translation of images wherein there is a lack of paired examples from an input domain X to an output domain Y. The objective is to acquire the mapping function using an adversarial loss, ensuring that the distribution of input data X closely matches the distribution of output data Y [44]. The mapping function can be depicted as follows in (2):

$$Gen : X \rightarrow Y \tag{2}$$

As this mapping is subject to significant constraints, we integrate it with a reverse mapping function, represented by (3), in the following manner:

$$F : Y \rightarrow X \tag{3}$$

Moreover, the cycle consistency loss function helps to accelerate $F(Gen)(X)) \approx X$ and also conversely. To retain the usual structure of the sample image and its related geometrical features, unsupervised image-to-image translation methods have sufficed for it in the recent state-of-the-art [51]. This results in cumulating instance occurrences and overcoming the inter-domain shift issue in novel environments by producing high-fidelity sophisticated synthetic samples in the training segment [26, 52]. This proves to be very useful in AV, for which the concept of UNIT is leveraged in our proposed work to carry out the inter-domain shift [53, 54].

Assumptions Let there be two domains of images X_1 and X_2. In the UNIT framework of supervised learning setting, we have sample pairs (i_1, i_2) taken from the collaborative space $C_{I_1,I_2}(i_1, i_2)$. While in unsupervised setting, the image samples are taken from the marginal distributions say $C_{I_1}(i_1)$ and $C_{I_2}(i_2)$. As there can be an unbounded set of probable shared distributions, which can produce peripheral distributions, hence, without any supplementary assumptions we could deduce nothing regarding the joint distribution from the marginal samples [55].

We assume the existence of a collaborative latent space denoted as z. In Fig. 3, it is demonstrated that for a given pair of images i_1 and i_2, within this collaborative space, there exists a common latent code z from which both image samples can be reconstructed. This code can also be figured out from the two samples respectively. In simpler terms, we hypothesize that there are functions En_1^*, En_2^*, Gen_1^*, Gen_2^* and there is an assumption of pair of corresponding image samples i_1 and i_2 from the collaborative space: $z = En_1^*(i_1) = En_2^*(i_2)$ and contrariwise, $i_1 = Gen_1^*(z)$ and $i_2 = Gen_2^*(z)$. Table 4 shows the interpretation of the various networks in the proposed model. To train the UNIT, we extract day and night image samples from the SYNTHIA dataset. Figure 4 depicts some of the translated samples from the dataset.

2.5 Limitations of existing GAN frameworks

While, the existing GAN frameworks have shown a substantial amount of success but it has their own limitations in generating the images in conditional/unconditional settings in AV driving environments. These frameworks are potentially capable of generating images of other domains, but it is challenging to generate training data for detecting objects with pre-determined anchor boxes in the target domain scenario from random noise [56]. Transformation of a labeled image sample to the target domain seems more plausible [57]. In the recent state-of-the-art [58, 59], there have been numerous GAN frameworks that have produced inter-domain mappings. Nevertheless, they have fallen short of delivering a comprehensive performance. Various performance factors such as network architecture, techniques

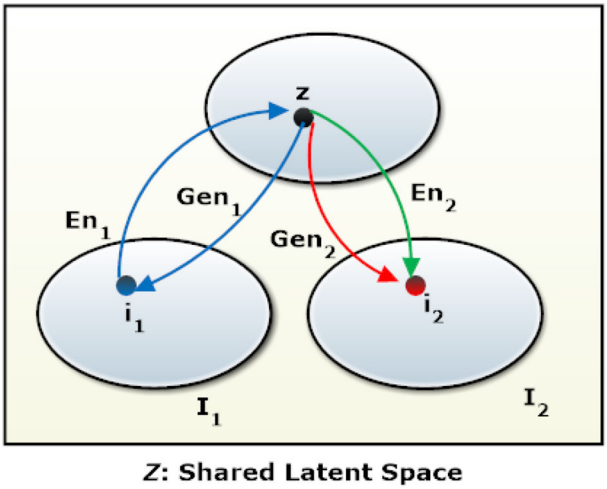

Fig. 3 Assumed shared latent space Z

Table 4 Notation Interpretation

Terms	Meaning
$\boldsymbol{En_1}, \boldsymbol{Gen_1}$	VAE for X_1
$\boldsymbol{En_1}, \boldsymbol{Gen_2}$	Image Translator $X_1 \rightarrow X_2$
$\boldsymbol{Gen_1}, \boldsymbol{Disc_1}$	GAN for X_1
$\boldsymbol{En_1}, \boldsymbol{Gen_1}, \boldsymbol{Disc_1}$	VAE-GAN
$\boldsymbol{Gen_1}, \boldsymbol{Gen_2}, \boldsymbol{Disc_1}, \boldsymbol{Disc_2}$	CoGAN

like adaptive learning rate, weight decay, etc., have a contribution to the efficient implementation and working of GANs. They can improve the convergence rate and stability of training in GANs. Moreover, other factors such as noise injection, regularization, standard, and size of the training dataset contribute to the GAN's efficacy. However, the existing GANs lie in a deficit of these factors. For example, CycleGAN cannot learn many-to-many mapping, i.e., one image having many variations in the output. Therefore, there is a need for invertible inter-domain mapping to cross-check the accuracy of the produced image with the original one [60]. Further, DualGAN faces distortion in the visual quality of the sample and is problematic to train. The existing literature lacks stability in the training process of the images and has a lesser amount of sample diversity. In addition, the prevailing GAN frameworks have produced sparse weather variations, low-resolution images, and distorted, unevenly illuminated images.

The existing GAN frameworks [58, 59] also fail to adaptively learn the special characteristics of style renderings of new domain images. We have shown a detailed comparison between the different types of existing GANs based on the factors affecting the implementation of GANs in Table 5. The phraseology used in the table is defined as follows for the readers to comprehend the challenges faced by the existing GAN frameworks in faster convergence and stable training. *Stability*- Stability refers to the ability of the GAN to consistently produce

Fig. 4 Sample images of Day-night transformations

Table 5 Comparative analysis of different types of GAN with our proposed model

Existing GAN Model	Pros	Cons	Training Stability	Quality of Generated Samples	Speed	Complexity of Implementation
Conditional GAN (CGAN) [56]	Produces diverse synthetic samples to balance class distributions	Requires labeled data. Requires additional information as conditional inputs. Accuracy of OD on specific classes. Class Imbalance	Less Stable	Limited by the quantity of the conditioning information	Fast to train and infer, but the speed can depend on the complexity of the conditioning information	Simple to implement, but the complexity can increase with more complex conditioning information
Cycle GAN [15]	Can handle variations in lighting and camera angles	Prone to mode collapse	Unstable	High-quality samples, but can also suffer from mode collapse or lack of diversity	Low due to the cycle consistency loss	Complex, due to the cycle consistency loss
Adversarial Over sampling (SMOTE GAN) [16]	Oversamples the minority class to balance the class distribution	Can generate unrealistic samples. Imbalanced classification tasks	Good training stability	High-quality samples with improved class balance in datasets	Fast to train and infer, depending on the dataset size	Simple to implement, but may require additional pre-processing steps for class balancing
Boundary Equilibrium GAN (BEGAN) [17]	Balances the foreground background class distribution using the boundary-seeking loss term	Requires tuning of hyperparameters and optimization methods. OD tasks with foreground background imbalance	Less Stable	High-quality samples with good diversity and stability	Slower to train than traditional GANs due to the balance factor loss, but inference can be fast	Balance factor loss adds complexity
COPGAN (Proposed)	Lightweight as the discriminator is eliminated at the end	Restricted to less number scenario generation	Relatively has better stability owing to less intervention of discriminator	Generates a distinct visual of an operational parameter	Generator is trained with the adversarial function of the discriminator, therefore after eliminating the discriminator generator starts generating images with better speed and less errors	Simpler in architecture

high-quality generated samples while maintaining the convergence of the training process. *Paired and Unpaired Images*- These images refer to where each image in one set is paired with a corresponding image in another set. While, unpaired samples refer where there is no explicit correspondence between the images in one set and the images in another set. *Class Imbalance*- Class imbalance GANs occur when the training data contains significantly more samples from one class or category than from another. *Regularization in the training process of GAN*- The regularization term in a GAN implementation is used to control the complexity of the generator and discriminator networks. Regularization discourages the model from fitting the training data too closely. *Convergence*- Faster convergence in GANs refers to the GAN's ability to reach a satisfactory level of fulfilment (such as generating high-quality samples) in a shorter amount of training time or with fewer training iterations.

Against the recent approaches, our proposed COPGAN framework doesn't have a strict similarity function underlying between the input domain and the output domain (Unpaired images). Furthermore, we do not impose limitations on our image dataset to be confined within the boundaries of an intra-low-dimensional space. Moreover, it provides realistic diverse variations, its accuracy of OD can be scaled to all classes of vehicular instances, and there is a balanced class distribution. Furthermore, it is more stable in training, produces high-quality samples, and has relatively faster convergence (Convergence) to a stable equilibrium (Stability). Hence, this framework can generalize to various perception tasks, such as detecting objects in unseen scenarios. In this manuscript, we have considered the prevailing issue of unseen environment mismatch, and therefore, the style and aesthetics of our image dataset are amended to form the most latent weather adversities. We have performed our evaluations on a combined dataset (COPGAN-generated images, augmented and stylized) having different domains. Lastly, the role of the synthetic images, on an inter-domain dataset of images is also judged for detecting the vehicles (Class Imbalance) [61, 62].

As per the discussion in the prior section, the existing GAN frameworks fail to preserve the image structure of the transformed image. Hence, the motive is to study the achieving of an object-preserving image translation model using the latent space representation of two image domains. The proposed COPGAN framework generates synthetic images of the target visual domain which, are consistent with its original counterpart in the source domain by incorporating the key feature selection step. We further elaborate on the training procedure and the network architectures of the proposed COPGAN method. We introduce two primary components of the proposed COPGAN: (i) We utilize a ***Unity Generator***, referred to as "Gen," to generate a synthetic image under a specific environmental condition denoted as p, (ii) a ***Unity discriminator*** *Disc* to assess if the rendered image at a particular operational state p is synthetic or real.

3 Proposed COPGAN framework

3.1 Unity generator

In contrast to the conventional approach of CycleGAN that employs dual generators for the forward and reverse mappings, our proposed COPGAN utilizes a unified generator, which means it utilizes a single parametric generator. Under a set of representative scenic parameters $\left(p = [p_1 p_2 p_k]^T\right)$ for an input image, the proposed COPGAN generates a synthetic image at the scenic condition p which describes the given operational state (for instance, summer, rain, winter, night). The consideration of the operational parameters is done, keeping in mind

the real-time scenarios and variations in the environment faced by the AVs, and are also assessed using several pieces of equipment installed within the vehicle. A small visibility sensor put on the car, for instance, can assess the visibility in the fog [34]. An input image can be quickly plotted to the resultant image constrained by p using such a unity generator. Figure 5 shows the three components i.e., encoder, decoder and transformation layer of the unity generator parameterized with an operational condition. Since the traffic data volume is very large (1 TB+ / h) to process, which overall burdens the computational processing of neural networks, we propose a key feature selection algorithm. With a key feature selection algorithm (Algorithm 1) and CNN made up of convolutional layers, normalization layers, and Leaky ReLU layers, an encoder extracts the significant features from a given image to perform object detection.

We design a key feature selection algorithm that carries dominant information to be processed further. We take two factors for claiming the importance of an object of interest O_I in an image I i.e., center point $Obj_{cent}(x_{cent}, y_{cent})$ of an object in an image I and the image center point $C(x_C, y_C)$ of the complete image. The dimensions of the image are measured in width and height respectively $(\boldsymbol{w_d}, \boldsymbol{h_t})$. The importance score of an object is $\boldsymbol{Imp_{score}}$. We emphasize the presence of an inverse relationship between the center point of the image and the center point of the object within it. Generally, to detect the foreground instances in a video, the conjecture holds that the longer the object lingers in front of the eyes and, the larger the cross-sectional area it occupies it gives a subjective feeling of the object being centrally located and an important foreground object. In this manuscript, we use the area size of the

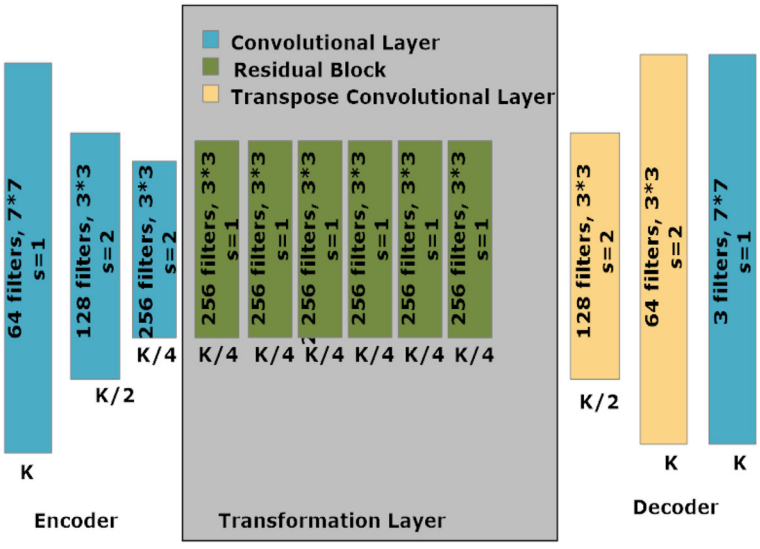

Fig. 5 Unity Generator composed of three components: Encoder, decoder, and Transformation. There is a reduction of the image representation in the encoder stage i.e. down sampling takes place. During the transformation phase, the encoder maps the input image to a feature space having low dimensions and, it can be processed by the generator network to produce the translated image. The generator network commonly includes a sequence of deconvolutional layers that systematically boost the spatial dimensions of the feature representation, while reducing its depth, until it produces the final translated image. Above, it is a list of the representation size for each layer, expressed in terms of the size of the input image, k. The amount of filters, their sizes, and the stride are provided for each layer. Instance normalisation and Leaky ReLU activation come after each layer

object and the central point of the video frame as two determining factors. The closer the vehicle is to the center point, the more importance the vehicle holds. We calculate Imp_{score} and subsequently, Key_{value} for each vehicular instance present in the video.

The $\boldsymbol{Imp_{score}}$ is calculated by the formula in (4) below:

$$Imp_{score} = 1 - \frac{\sqrt{(x_{cent} - x_C)^2 + (y_{cent} - y_C)^2}}{\sqrt{(w_d^2 + h_t^2)/2)}} \tag{4}$$

The range of $\boldsymbol{Imp_{score}}$ is **[0, 1]**. The image center is denoted by 1 and four corners of the picture is denoted by limit 0. To determine the key features, we need to have a key score proportion which we calculate using the below (5):

$$Key_{val} = \sum_{r=0}^{N} \sum_{s=0}^{M_r} [Imp_{score}(r, s) * w_d(r, s) * h_t(r, s)] \tag{5}$$

The parameters in the above equation are defined as follows:

-

N: total number of image features

M_r: total objects identified in $\boldsymbol{l}$ images

Imp_{score}: importance score

$w_d(r, s)$: width of the object (r,s)

$h_t(r, s)$: height of the object (r,s)

Out of $\boldsymbol{q}$ number of objects, the percentage of every detected object can be formulated in the below (6) as follows:

$$Per = \frac{Key_{val}}{\sum_{i=0}^{q} Key_{val}(i)} \tag{6}$$

The algorithm for detecting the key objects in an image is explained in the Algorithm 1. After the feature extraction process, we obtain key features and then we rebuild our original image by retaining these key features and transforming the dynamics of the image into another domain using the proposed COPGAN framework. This outputs a synthetic image of a newly adopted domain retaining the original features. For example, suppose we have two set of images $\boldsymbol{S}$ and $\boldsymbol{T}$ where $S = \{F_m^R(p)\}_{m=1}^n$ set belongs to the input domain that is the data captured in real-world using sensors, cameras etc. $T = \{F_m^F(p)\}_{m=1}^n$ set belongs to the target domain or the synthetic domain, which is obtained from the input domain over a rigorous training process of this adversarial model. Here, $\boldsymbol{F_m}$ is the number of feature maps over continuous pixels p, where R *and* F denotes real and fake images in the input domain and the output domain, respectively. The input source domain set S has n number of images $\boldsymbol{S = \{I_1, I_2, ...I_n\}}$ and each image has a number of feature maps of continuous pixels say in (7),

$$I = \{fea_1(p), fea_2(p), fea_3(p), ...fea_n(p)\} \tag{7}$$

3.2 Synthetic image reconstruction

After the key feature extraction process in an image, the proposed COPGAN model takes key features of an image as input, and the encoder in the COPGAN converts image $\boldsymbol{I}$ to codes $\boldsymbol{C}$ and the decoder rebuilds the images from the codes as shown in (8) below: -

$$X : I \rightarrow C, Y : C \rightarrow I' \tag{8}$$

Algorithm 1: Key image feature selection.

Input: sum of all images N
Output: set of key features based on Percent value
Procedure `CalPercent`**:**
  $TotalKey_{value}, Key_{value} \leftarrow CalKey_{value}(S)$
, $T \leftarrow$ Threshold value for percentage $KeyFeatures$
$Key_{value} \leftarrow S$
for ($i = 0;\ i<S-1;\ i = i+1$) {
  Using (6)
}
for ($i = 0;\ i < S-1;\ i = i+1$) {
  if $PercentageScore_k > T$
  Add Feature k to KeyFeatures; **then**
  return KeyFeatures
}
Procedure `CalKeyvalue(`S`)`**:**
  $TotalKey_{value} \leftarrow 0, Key_{value} \leftarrow S$
  for ($k = 0;\ k < S-1;\ k = k+1$) {
    $M_k \leftarrow$ sum of all objects identified in k image
    $C \leftarrow$ center point of image k;
    for ($l = 0;\ l < M_k;\ l = l+1$) {
      $Obj_{cent}(x_{cent}, y_{cent}) \leftarrow$ center point of O_I of object l
      $Imp_{score} \leftarrow CalImpScore(Obj_{cent}(x_{cent},$
      $y_{cent}), C, w, h)$
      Using (5)
      $TotalKey_{value} \leftarrow TotalKey_{value}+$
      $Key_{value}(l)$
    }
  }
  return $TotalKey_{value}, ImgKey_{value}$ to $CalPercent$
Procedure `CalImpscore(`*Point* $Obj_{cent}(x_{cent}, y_{cent})$,
Point $C(x_C, y_C)$,*width* w_d, *height* h_t)**:**
  $x_{cent} \leftarrow Obj_{cent}.x_{coord}$
  $y_{cent} \leftarrow Obj_{cent}.y_{coord}$
  $x_C \leftarrow C.x_{coord}$
  $y_C \leftarrow C.y_{coord}$
  Using (4)
  return Imp_{score} to $CalKeyvalue$

Table 6 describes the notational interpretation of the terminologies used in the description of the framework. $\boldsymbol{Y}$ is a non-linear mapping between the codes and the reconstructed image sample. Further, Algorithm 2 shows the training methodology of the proposed novel COPGAN framework.

For a HQI dataset X^m_{real} where image $x \in X^m_{real}$, we create a LQI dataset $\bar{X}^m_{real}$. This dataset is built using image quality distortion kernel $\rho(.)$ such that $\bar{x} = \rho_n(x)\rho(.)$ is a level

Table 6 Notation Interpretation

Terms	Meaning
HQI	High quality image
LQI	Low quality image
AD	Augmented Data
R	Reference model
$\boldsymbol{\rho(.)}$	Distortion Kernel

Algorithm 2: COPGAN training.

Input: Real image dataset of $X^m_{real}; m = 1, 2..M$, respective operational conditions $p^m_{real}; m = 1, 2..M$, Generator model Gen and discrimination model $Disc$ initialized with pre-trained model weights.
Output: Augmented images having synthesized variations
Assumptions: Small-batch size B_s for obtaining global optimization
Initialize Gen in Fig. 4 and $Disc$ in Fig. 10
for ($k = 1...K$) {
1. Arbitrary batch selection $x_B \in X^m_{real}$
2. Augmented batch samples: $\bar{x}_B \in (\bar{X}^m_{real})_B$
3. Utilize $R(x_B)$ to train $Disc$; predict $x_B \in X^m_{real}$. (31) with $\eta < 0$.
4. Utilize $Gen\ (\bar{x_B})$ to train corresponding $Disc$ to predict $\bar{x}_{B_s} \in (\bar{X}^m_{real})_B$. Update weights of corresponding $Disc$ to minimize (31) with $\eta < 0$.
5. Train Gen to predict $Gen\ (\bar{x}_B)$ such that $Disc\ (Gen\ (\bar{x}_B))$ predicts $x_B \in X^m_{real}$. Update weights of Gen to minimize (31) with $\eta > 0$.

}
Discard $Disc$. Utilize Gen to detect objects with better robustness to image distortions.

of an arbitrary selection of LQI out a collection of N distortion levels. We construct an augmented dataset (AD) $\bar{X}$ that pools the original image sample dataset X^m_{real} and the low-quality $\bar{X}^m_{real}$. This augmented dataset $\bar{X}$ undergoes training in the proposed architecture.

Figure 6 displays the schematic diagram of the proposed COPGAN architecture. This framework comprises a $\{Gen\}$ and a $Disc$ deceiving each other, forming an adversarial system. In the training regime, the object detector outputs $o \in HQI$ for each augmented image $\bar{x} \epsilon \bar{X}^m_{real}$ in the training domain X^m_{real} using the past-trained model on HQI which

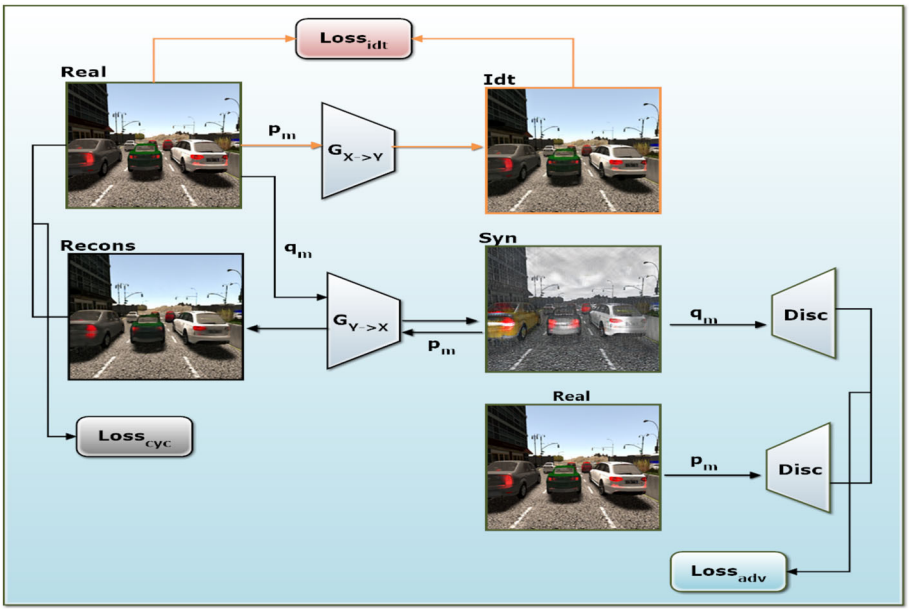

Fig. 6 The proposed COPGAN model comprising two primary parts: (i) the generator Gen and (ii) the discriminator Disc

is, therefore, referred as a reference model $R(x)$. Correspondingly, the OD model outputs $\bar{o} \in AD$ for every augmented image $\bar{x} \epsilon \bar{X}_{real}^{m}$ whose quality is variable in the augmented samples $\bar{X}_{real}^{m}$ using the Gen. The discriminator $Disc$ fulfills its goal of correctly categorizing the output o from $\bar{O}_{Gen}$ to be coming from reference model $R(x)$ and generator $Gen(\bar{x})$. The aim of the generator Gen is to outperform the discriminator $Disc$ to classify the augmented output $\bar{o}$ in $R(x)$.

To put forth differently, the discriminator $Disc$ is trained to learn this adversary system that differentiates the reference model $R(x)$ from generators $Gen(\bar{x})$ instead of forming an additional factor of stability loss. On the basis of this adversary system in which $Disc$ is trained, the generators Gen learn to diminish the distance between the output o from $\bar{O}_{Gen}$ such that the output yield of the generator due to $\bar{x}$ is similar to o. The discriminator $Disc$ is discarded upon the successful completion of the training and the generators Gen are used in place of the reference framework. This will boost the robustness of the framework to the quality distinctions in dataset samples. The weights of the generator Gen are configured with the pre-trained weights of the reference models $R(x)$. The reference models are *SDY*. The HQI images are taken in a balanced ratio of 1:1.

3.2.1 Encoder

Encoder E basically comprises a number of convolutional layers $(C_1, C_2, ...C_n)$ to extract the key representational features called feature maps. These feature maps F_m are of diverse semantics coming from the input samples. After the encoding process, the input image can be viewed as in (1). To obtain the feature map m in the ith filtering, the simulation of the convolution layer is expressed as in (9):

$$m^i = A\left[\sum_{k=1}^{D} Wt^{i\lambda} \otimes Y + B^{i\lambda}\right] \tag{9}$$

Wt is the matrix for calculating weight, and B denotes the bias vector. These parameters are initialized randomly in the encoder. We employed Leaky ReLu as an activation function A for the encoder and decoder, and λ signifies the convolution kernel index number.

3.2.2 Decoder

In this stage, the generator Gen upsamples m^i to the reconstructed feature maps F_R as shown in (10) below:

$$F_R^i = A\left[\sum_{k=1}^{D} Wt^{'i\lambda} m^i \otimes m^i + B^{'i\lambda}\right] \tag{10}$$

Gradually, after reaching to the last convolution layer, the generator Gen yields an output image $I^{'}$ of the same semantics as that of I which is shown in (11) below:

$$I^{'} = \left\{f_1^{'}(p), f_2^{'}(p), ...f_n^{'}(p))\right\} \tag{11}$$

Further, these convolutional layers are utilized for down sampling and feature extraction. After passing through the convolutional layer, as depicted in Fig. 7, a feature map of size *5*5* is reduced to a smaller size of *2*2*. The initial convolutional layer directly extracts local features from the image. Subsequently, each subsequent convolutional layer extracts feature

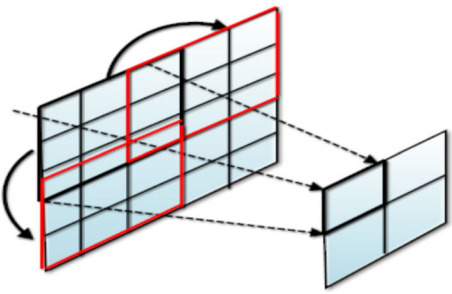

Fig. 7 Encoder outputs N image feature maps that are transformed with K auxiliary feature maps which are representative of the K operational conditions

maps from the output of the previous layer, progressively capturing higher-level features present in the image.

The application of convolutional layers is done in this case as they can be utilised for the detection and identification of objects with two unique characteristics: spatial invariance and local correlation. For instance, raindrops can appear everywhere in the image and cause local distortion patterns. As a result, an encoder's convolutional layers can aid in extracting both characteristics (low-level and high-level) from a sample. The extracted features from the input image data will be utilized in subsequent steps below, to generate a synthesized image data that represents the complete image in an object-centric manner.

- Following a convolutional layer, a normalization layer is employed to normalize the output feature map. This normalization aids in achieving faster learning and higher accuracy during the training process [28].
- Further, Leaky ReLU is employed as an activation function for non-linear transformation.
- Therefore, Leaky ReLU is taken as a correction to the drawback posed by ReLU that avoids the dying neuron problem. Leaky ReLU returns a small positive value for negative inputs instead of zero.
- Hence, in the generator network of COPGAN, leaky ReLU is used in the hidden layers to introduce non-linearity and prevent the gradients from vanishing.
- In the discriminator network, leaky ReLU is the activation function in the intermediate layers to allow for negative values and avoid the saturation of the sigmoid function. Additionally, leaky ReLU also aids in preventing the problem of mode collapse in COPGAN by allowing the discriminator to have more capacity to distinguish between different input domains.
- Following the encoder stage, the transformation layer receives the extracted features of image along with the operational scene parameter p, which specifies the desired rendering for the image.

As illustrated in Fig. 8, let us consider that there are N feature maps obtained from the encoder layer. Additionally, the scenic parameter p_k, where k belongs to the set $1,\ 2,\ \ldots K$, represents an auxiliary feature map in which all elements are equal to p_k. Therefore, the total of $N + K$ feature maps have the same size. The goal of the translation is to produce the scenario under a given operational scene p. To obtain precise refinement for feature variation, we utilize deep ResNets in the implementation. As depicted in Fig. 9, each residual block in ResNets is made up of numerous convolutional and ReLU layers as well as a direct bypass from the input to the output. Relevant image features, such as the colour and texture of the scene's objects, are suitably adjusted as they pass through a number of residual blocks. After the transformation, the features are updated, and a decoder uses these characteristics to create the corresponding image. Deconvolution is a crucial process carried out by the decoder.

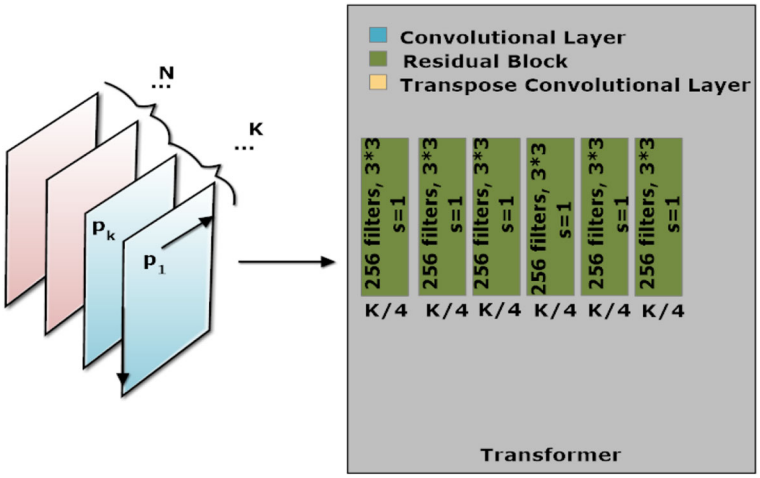

Fig. 8 Feature map of *5*5* is down sampled to a *2*2* feature map after running a conv layer with a *3*3* kernel and the stride of 2

Unlike the convolutional layer depicted in Fig. 7, the deconvolutional layer enhances the quality of the feature maps as illustrated in Fig. 10.The modified features are then converted back into an image with the original resolution by the decoder, which may theoretically be thought of as an inverse function of the down-sampling process carried out by the encoder.

The encoder captures the essential elements of the input image, which are then transformed by the image translation process in our proposed parameterized unity generator. This transformation aligns the image with the desired scenario specified by the given operational parameters. Next, using these updated features, the decoder creates a new image. The identical scene from the original photograph should remain in the synthetic sample, albeit in a different operational state. Such a generator constrained with a parameter needs to be instructed properly, and the training process needs to be supported by a parameterized discriminator.

3.3 Unity discriminator

To accommodate the scene parameters in the conventional Cycle GAN there is a competition of the two discriminators against the two generators. COPGAN takes up a single discriminator with an operational condition *p*. When the discriminator is given an image as input

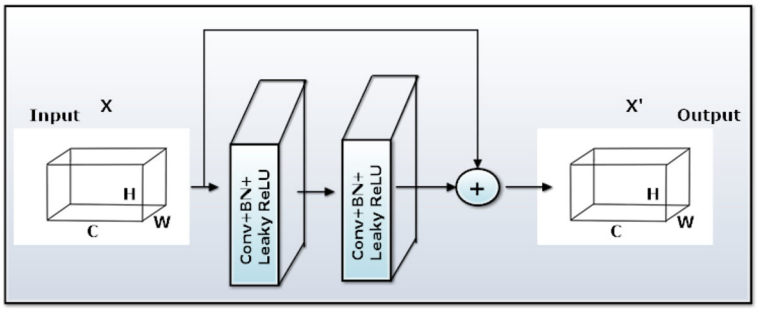

Fig. 9 Residual block with ReLU and conv layers

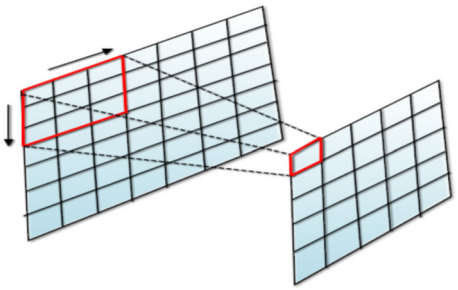

Fig. 10 After passing the deconvolutional layer, up sampling of a 2*2 feature map to a 5*5 feature map is done

with the operational parameter p, it produces a scalar that indicates the likelihood of that image to be real under the specified condition p. In other words, at the specified condition p, the parameterized discriminator is intending to discriminate between real and the synthetic images. Figure 11 shows four convolutional, normalisation, and ReLU layers in its architecture which is followed by a fully connected layer. The first layer receives the input image and the specified operational parameter, p. We presume that there are three RGB colour channels in the image. For each parameter p_k, where k belongs to the set $1, 2, ...K$, there exists an additional feature map that corresponds to p_k, similar to the generator. Downsampling occurs as the input sample and additional feature maps are passed through the convolutional layers. The discriminator produces one scalar after the fully connected layer, which represents the likelihood that an input picture will be real under the specified operational condition stipulated by p. The resulting number should approach 1 if the input image is highly probable to be real (or synthetic), or alternatively approach 0 if it is unlikely to be real (or synthetic).

To provide a smooth gradient and achieve a quick convergence during training, we avoid a sigmoid function during the output, in contrast to conventional classifiers. Sigmoid functions soon saturate to 0 or 1 and the gradient value disappears, leading to sluggish convergence. Our proposed parametrized unity discriminator can be exploited for the evaluation of the probability that under the given operational condition p, an image generated by the generator is reliable. The generator network can then be effectively updated to produce synthetic images with a higher probability. It is difficult to train the generator and discriminator, as it has to undergo thorough adversarial training.

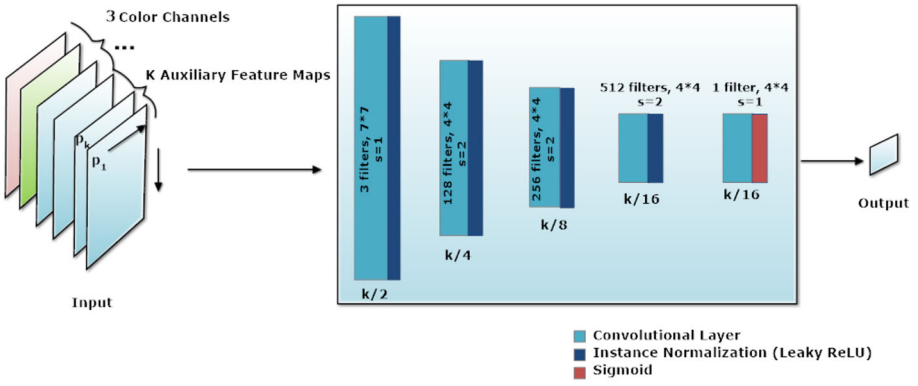

Fig. 11 The RGB image with three feature maps is given to the discriminator, K additional feature maps denotes the K operational states/parameters.The output is a solitary scalar value representing the probability of the input image being genuine under the provided parametric condition

4 Operational COPGAN

We propose the development of novel COPGAN framework for generating cross-domain variations faced by the AVs in the real-time driving scenarios. This will aid in equipping DNN-based OD models to boost their inference speed and robustness. Figure 6 illustrates the proposed COPGAN framework consisting of two primary adversarial components: *Unity Generator* and *Unity Discriminator*. Algorithm 2 shows the training procedure of the proposed COPGAN. Consider a real image set $X^m_{real},\ m = 1,\ 2,\ 3 \ldots M$ where every X^m_{real} is captured under the operational parameter p_m. Further, to achieve an ideal generator, the proposed COPGAN should be able to complete the three training targets. Firstly, given the defining operational parameter q_m, for the input image X^m_{real}, the *unity generator* must be able to produce a corresponding synthetic image $X^m_{syn} = Gen\left(X^m_{real},\ q_m\right)$ which is homogenous to the real image (highlighted in green color) shown in the Fig. 6.

Secondly, the obtained synthetic image X^m_{syn}, when fed under the operational parameter p_m, should be reconstructed by the *unity generator*, which must be indistinguishable from the original image X^m_{real} (highlighted in black color). Thirdly, the unity generator should also produce a synthetic image $X^m_{idt} = Gen\left(X^m_{real},\ p_m\right)$ that is indistinguishable from the X^m_{real} (highlighted in orange color) in Fig. 6. Further, for an ideal generator, since both X^m_{rec} and X^m_{idt} should be the same to the X^m_{real}, in practicality, they are likely to be different from X^m_{real} as it is difficult to achieve an ideal generator scenario. In order to satisfy the requirement of an ideal generator, both the produced synthetic image, i.e., X^m_{idt} and the reconstructed image X^m_{rec}, should look alike. However, in practical scenarios, they are likely to be different from X^m_{real}. Hence, it is infeasible to achieve an ideal generator. Since X^m_{idt} is processed only once by the generator for a given image X^m_{real} without altering its operational condition, X^m_{idt} often resembles X^m_{real} more than X^m_{rec}. In contrast, X^m_{rec} undergoes two processing stages by the generator, and additionally, the operational condition is modified. To realize the aforementioned objectives for training the generator Gen, we suggest three complementary loss functions-the adversarial loss $Loss_{adv}$ combinedly composed of the $Disc$, the object-preserving cycle consistency loss $Loss_{cyc}$, and the identity loss $Loss_{idt}$. We will elaborate on the specifics of each of the loss functions in the subsequent sections that follow.

4.1 Evaluation of adversarial loss function

To correctly classify the image data as real or synthetic at a specified condition, both the real X^m_{real} and synthetic X^m_{syn} are recorded by $Disc$. Further, the $Disc$ outputs a score near 1 (high) in case of a real image X^m_{real} with parameter p_m and, 0 in case of X^m_{syn} with parameter q_m. Therefore, an ideal discriminator $Disc^{ideal}$ must possess the characteristics in (12)-(13) as follows:

$$Disc^{ideal}\left(X^m_{real},\ p_m\right) = 1 \tag{12}$$

$$Disc^{ideal}\left(X^m_{syn}\ ,\ q_m\right) = 0 \tag{13}$$

Contrary to this, the goal of the Gen is to produce false images at a given scenic condition, indistinguishable from the real ones. As a result, even an ideal $Disc$ cannot recognize the produced synthetic instances. Hence, an ideal generator Gen^{ideal} must produce synthetic

images with higher values equivalent to 1 shown in (14).

$$Disc^{ideal}\left(X_{syn}^{m},\ q_m\right) = Disc^{ideal}\left(Gen^{ideal}\left(X_{real}^{m},\ q_m\right), q_m = 1\right) \tag{14}$$

The above-stated facts highlight an important observation that an adversarial nature of the system is exhibited in which Gen and $Disc$ try to outperform each other during the learning process. This is also called a two-player mini-max game, and it can be mathematically represented using the adversarial behavior as follows:

$$Loss_{adv,\ Disc}\left(Disc; Gen^{ideal}\right) = \frac{1}{M}\sum_{m=1}^{M}\left[1 - Disc\left(X_{real}^{m},\ p_m\right)\right]^2 + \frac{1}{M}\sum_{m=1}^{M}\left[0 - Disc\left(Gen^{ideal}\left(X_{real}^{m},\ q_m\right), q_m\right)\right]^2 \tag{15}$$

$$Loss_{adv,\ Gen}\left(Gen; Disc^{ideal}\right) = \frac{1}{M}\sum_{m=1}^{M}\left[1 - Disc^{ideal}\left(Gen\left(X_{real}^{m},\ q_m\right) q_m\right)\right]^2 \tag{16}$$

where, $Loss_{adv,\ Disc}\left(Disc; Gen^{ideal}\right)$ and $Loss_{adv,\ Gen}\left(Gen; Disc^{ideal}\right)$ defines the loss functions of the adversarial system. (15)-(16) shows the mean squared error calculated amongst the outputs of the discriminator. Further, the loss function is defined by the target value. If the $Disc$ is ideal, the $Loss_{adv,\ Disc}\left(Disc; Gen^{ideal}\right)$ becomes 0 i.e., to the minimal value. Moreover, the (12)-(13) should attain their optimal values i.e $Disc^{ideal}\left(X_{real}^{m},\ p_m\right) = 1$ and $Disc\left(X_{syn}^{m},\ q_m\right)$=0. Therefore, it means that the $Loss_{adv,\ Disc}\left(Disc; Gen^{ideal}\right)$ should be minimized during the learning process of $Disc^{ideal}$ shown in (17).

$$Disc^{ideal} = argmin_{Disc} Loss_{adv,\ Disc}\left(Disc;\ Gen^{ideal}\right) \tag{17}$$

In addition, $Loss_{adv,\ Gen}\left(Gen; Disc^{ideal}\right) = 0$, in case of an ideal generator. Hence, the Gen^{ideal} should reduce the loss function $Loss_{adv,\ Gen}\left(Gen; Disc^{ideal}\right)$ shown in (18).

$$Gen^{ideal} = argmin_{Gen} Loss_{adv,\ Gen}\left(Gen;\ Disc^{ideal}\right) \tag{18}$$

Nevertheless, the Gen and $Disc$ battle with each other in their learning objectives. The $Disc^{ideal}\left(X_{syn}^{m}\ q_m\right)$ should be equivalent to 1 with the given $Disc^{ideal}$ as depicted in (14). For an optimized $Disc$, $Disc\left(X_{syn}^{m},\ q_m\right)$ should be 0; on the other hand, an optimal Gen should be 1. Iterative training should be conducted for Gen and $Disc$ by resolving the following two optimization problems:

$$Disc^{(t+1)} = argmin_{Disc}\ Loss_{adv,\ Disc}\left(Disc;\ Gen^{(t)}\right) \tag{19}$$

$$Gen^{(t+1)} = argmin_{Gen}\ Loss_{adv,Gen}\left(Gen;\ Disc^{(t+1)}\right) \tag{20}$$

where t denotes the iteration index. It is observed that there is a high correlation between the X^m_{real} and X^m_{syn}, although Gen and $Disc$ can be trained using the optimization in (19)-(20). However, there is no guarantee that it will converge to the right answer. Mode collapse is the extreme scenario in which Gen may map various input images to the same output image subjected to a specific operational condition p. Therefore, in this situation, the synthetic image drops certain crucial elements of the original image and is declared useless. Nevertheless, the (19)-(20) depicting the loss function cannot stop the mode collapse from occurring. To cater to this problem, we suggest using object-preserving cycle-consistency loss $Loss_{cyc}$ to normalize the process of training.

Figure 6 depicts that, X^m_{syn} is converted into a reconstructed image called X^m_{rec}, with the parameter p_m indicating the initial operating state. Here, the object-preserving cycle consistency should be met i.e., $X^m_{rec} = Gen\left(Gen\left(X^m_{real},\ q_m\right), p_m\right) \approx X^m_{real}$ by both the original image X^m_{real} and the rebuilt image X^m_{rec}. We compute the differences between the pixels between the two images X^m_{real} and X^m_{rec} and then normalize by L-2 norm in (21) as follows:-

$$Loss_{cyc}(Gen) = \frac{1}{I \bullet D} \sum_{m=1}^{I} ||X^m_{real} - X^m_{rec}||_2^2 \tag{21}$$

where D denotes the density of the pixels in an image. $||\bullet||$ means the L-2 norm of a vector. The synthetic picture X^m_{syn} from Gen should keep all of the key elements of the original image X^m_{real} by minimizing the cycle consistency $Loss_{cyc}(Gen)$. Conversely, in such a case, the reconstructed image X^m_{rec} from X^m_{syn} cannot approximate X^m_{real} precisely. Therefore, mode collapse can be effectively avoided by object-preserving cycle consistency loss. The combining and rebuilding processes is constrained by the object-preserving cycle consistency loss though, we cannot be certain about the accuracy of each procedure. Inaccurate synthetic images can result from malfunctioning generators that incorrectly retain the key aspects of the original image. For instance, a flawed color-inversion generator can produce a false image with inverted colors, and by inverting the colors once more, it might precisely rebuild the original image from the false image. Such a flawed generator cannot be successfully rejected by the object-preserving cycle consistency loss in (21). Therefore, to cater to this issue, we further present identity loss $Loss_{idt}$. Under a certain operational condition p_m, X^m_{idt} is produced from the original image X^m_{real}. Here, both the original image X^m_{real} and the synthetic image X^m_{idt} should be almost indistinguishable. $X^m_{idt} = Gen(X^m_{real},\ p_m) \approx X^m_{real}$. L2 norm is utilized to measure the variation between X^m_{real} and X^m_{idt}:

$$Loss_{idt}(Gen) = \frac{1}{I \bullet D} \sum_{m=1}^{I} ||X^m_{real} - X^m_{idt}||_2^2 \tag{22}$$

If we minimize the identity loss, a similarity should be achieved between the synthetic image and the real image. Henceforth, the important features of the real image should be properly maintained by the Gen. Observing (19), (21), and (22), we can express the improvement in the problem for training Gen and $Disc$ in (23) and (24):

$$Disc^{(t+1)} = argmin_{\ Disc} Loss_{adv,\ Disc}\left(Disc,\ Gen^{(t)}\right) \tag{23}$$

$$\begin{aligned} Gen^{(t+1)} = argmin_{\ Gen}\Big(&\omega_{adv}, Loss_{adv,Gen}\left(Gen; D^{(t+1)}\right) \\ &+\omega_{cyc}\ Loss_{cyc}\left(Gen\right) + \omega_{idt} Loss_{idt}(Gen)\end{aligned} \tag{24}$$

where, ω_{adv}, ω_{cyc}, ω_{idt} are the adversarial, cyclic and identity parameters regulating the weights for their respective objectives.

5 Weighted loss function

Careful estimation of the weights ω_{adv}, ω_{cyc}, and ω_{idt} mentioned in (24) is crucial as they have a substantial impact on the optimization problem's resolution. Numerous conventional techniques in [15, 63, 64] have approximated these weights experimentally; nevertheless, looking for the best weights non-mechanically is time-consuming. In this study, we adopt the concept from [65] and introduce a new approach based on maximum likelihood estimation (MLE) for estimating the unknown weights. The probability function for each objective in (24) will be derived first, and the proposed MLE formulation will be presented subsequently. The first goal in (24) depicts the adversarial loss in (16).For every input pair of X^m_{real} and q_m, the discriminator generates a scalar value $Disc^{ideal}\left(Gen\left(X^m_{real},\ q_m\right)q_m\right)$, and the anticipated output value is 1, indicating the score of a real image. Hence, the error for the output is depicted in (25) below:

$$ERR^m_{adv} = Disc^*\left(Gen\left(X^m_{real},\ q_m\right), q_m\right) - 1 \tag{25}$$

A reconstructed image $X^{rec}_m = Gen\left(Gen\left(X^m_{real},\ q_m\right), p_m\right)$ is given as an output for every input sample $\left(X^m_{real},\ q_m\right), p_m$. The objective is to restore the original image X^m_{real}. Hence, the error of recovering each pixel can be denoted in (26) as follows:

$$ERR^{m,\rho}_{cyc} = X^{m,\rho}_{rec} - X^{m,\rho}_{real} \tag{26}$$

Identity loss in (22) is depicted by the last objective in (24). Without altering the operational condition p_m, the false image $X^m_{idt}\left(Gen\left(X^m_{real},\ p_m\right)p_m\right)$ is given as an output by the generator for a given input image $\left(X^m_{real},\ p_m\right)$. An effective generator should ensure that X^m_{idt} is indistinguishable from X^m_{real}. Consequently, the error of pixel recovery can be expressed in (27) as:

$$ERR^{m,\rho}_{idt} = X^{m,\rho}_{idt} - X^{m,\rho}_{real} \tag{27}$$

5.1 Data generation

For the comprehensive and robust testing of the object detection models *SDY*, we implement adverse climate changes to both synthetic and real images which are trained in different scenarios by the proposed COPGAN model, and various meteorological conditions of the real samples. *SDY* acts as a standard baseline models. SSD and YOLOv5 represent widely used architectures earlier and well-studied benchmarks for assessing the effectiveness of detection capability. Moreover, they are well-known for their speed and efficiency making them suitable for a large-scale testing on adverse environments, synthetic datasets. We simulate these extreme weather scenarios with SYNTHIA dataset, AdaIN and other data augmentation styles.

5.1.1 Varying weather modeling

We consider domain bias produced by different weathers such as ($summer \rightarrow winter$, $summer \rightarrow rainy$, and $summer \rightarrow night$) on which *SDY* are trained to identify the

different vehicular instances. We adopt season transfers as one of the robustness benchmarks owing to measure the resilience of the trained models *SDY* against real-time misrepresentations. Compared to the diverse synthetic falsifications, we also utilize following three different approaches in contrast to the above-mentioned season transfers for mitigating the insubstantiality of our object detection models. The adopted three approaches are described as follows:

Approach 1 and approach 2 (A1 and A2) We integrate the real images with different style renderings namely: Cezanne, Monet, Ukiyoe, and Vangogh. This can lessen the vulnerability of *SDY* to various distortions and has validated that our trained models are stronger against different types of noise, moderating the biases of the texture.

Approach 3 (A3) The AdaIN styles are also integrated into the weather-simulated environments along with the augmented images. The three approaches employed in our investigations come under AdaIN fast style transfer [66] technique. This technique casts the source sample to Adain's source sample representation.

5.1.2 AdaIN styles

To perform the transfer of style, we employed the AST-AdaIN model. This model is tuned using unpaired images from both the input(source) and output(target) domains, similar to the approach used in CycleGAN. Unpaired images from the input and output domains are fed into an encoder, which produces some feature maps $Enc(s) = f_s$ and $Enc(t) = f_t$, where s and t represent the input and output images, respectively, and f_s and f_t denote their respective representations in the feature space. The fake-output feature maps f_t are generated by aligning the mean and variance of the input feature maps with those of the output feature maps using the Adaptive Instance Normalization layer (AdaIN).

$$f^t = AdaIN\left(f_s,\ f^t\right) = \sigma\left(f_t\right)\left(\frac{f_s - \mu\left(f_s\right)}{\sigma\left(f_s\right)}\right) + \mu\left(f^t\right) \tag{28}$$

5.1.3 Stylized weather modeling

The AdaIN styles are integrated into the weather-simulated environments and the augmented images discussed in the next subsequent sub-section. The AdaIN style approaches implemented in our investigations are elucidated as follows:

(1) *Cezanne style* contains 526 paintings that transform an input sample in the Cezanne sample style. We have kept our training dataset 50% stylized to accentuate the features based on the shape in our trained model. (2) *Monet style* contains 1073 samples, (3) *Ukiyoe style* contains 563 paintings, and (4) *Vangogh style* contains 400 paintings. The samples of the different styles are shown in Fig. 12 that shows our input dataset rendered to these styles.

5.1.4 Basic data augmentation

One of the principal approaches for satisfying the invariance in the unseen test domains is to introduce basic data augmentation strategies for data-hungry models to persist better. We explore the effect of augmentating data using few techniques in this manuiscript that are well-defined as follows:

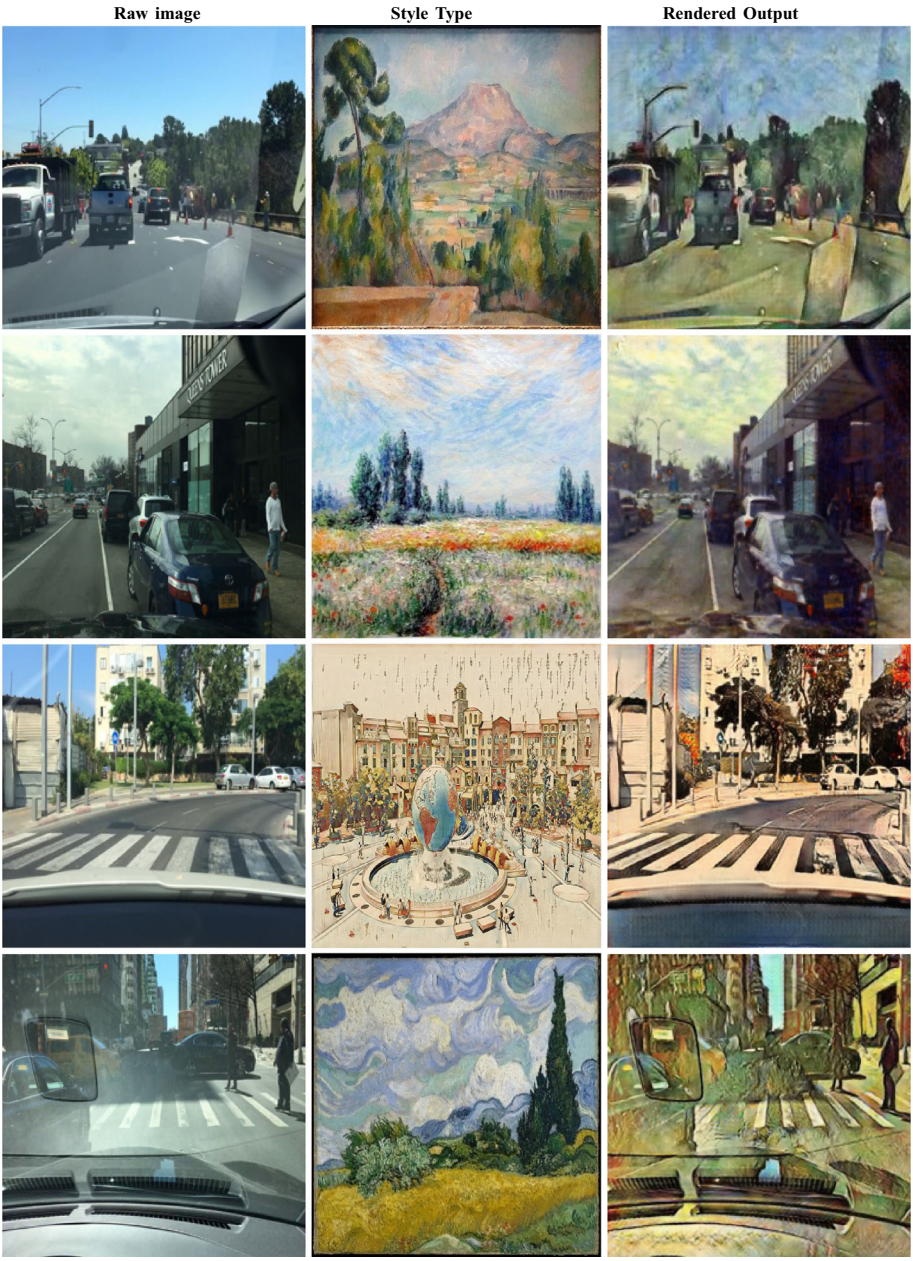

Fig. 12 Different style renderings of four varied paintings

(a) *GaussianBlurTransformed*: This type of effect is also called Gaussian smoothing image processing. The effect results from blurring an image using a Gaussian function. The term "Gaussian function" is derived from the name of Carl Friedrich Gauss, a renowned scientist and mathematician, which is a successfully accepted variation in graphics, which lessens the noise effect and the image details. One can observe a smooth blur as a visual effect of this

variation, when a scene is viewed through a translucent screen. It is noticeably dissimilar from the object shadow under usual illumination or a bokeh effect generated by an out-of-focus lens.

(b) *InvertTransformed*: The name itself suggests that the colors of the image are inverted to some other color values. These color values are of the opposite value in the color palette on the color wheel. This technique is also called the Random Invert transform.

(c) *GrayScaleTransformed*: The pixels in the grayscale image hold information about the amount of light it emanates. It consists of the intensity value. The visual effect of the grayscale images is gray monochrome or black and white type. This effect comes from the different shades of gray color depicted in a single image. The range of contrast is from the black (low intensity color) to the strongest white (high intensity color).

(d) *JitterTransformed*: This variation alters the saturation, brightness, and other relevant image properties.

(e) *SolariseTransformed*: In the solarized transformed sample, the pixel values are obtained by inverting the pixel values that exceed a certain threshold.

6 Model training of object detection models

A fundamental role for autonomous driving is vision-based object detection. In supervised scenarios, CNN models perform great whereas they underachieve in mutable environments. These models also require the learning of new parameters under the domain shifts that appear subtle to the human observer. For instance, training the images in one weather condition/or geographical area may result in an insignificant performance due to a shift in the pixel-level distribution of scenes. Therefore, we train our object-detection models with a diverse set of images having various class annotations.

6.1 Experimental settings

In this manuscript, a system configuration having *Intel® Xeon® CPU @ 2.20GHz* with Core i7 9th Gen and a *Nvidia GeForce GTX 1080 Ti* GPU is utilized. We leverage the designed COPGAN-generated synthetic image dataset of urban scenes, keeping SYNTHIA as a base dataset. We report our experimental results on three varied domain adaptations mentioned in the data generation subsection. A common variation of gradient descent, the Adam optimizer is used to make the process of training more efficient and stable. We have taken the following parameters for *hyper-parameter tuning* as follows: *batch-size=1*, *n-epochs=100*, *n-epochs-decay=100*, *beta-1=0.5*, *lr=0.0002*, *gan-mode=lsgan*, *pool-size=50*, *lr-policy=linear*, *lr-decay-iters=50* as parameters to fine-tune our model for wide-ranging results.

Data split: The original dataset PASCAL VOC is partitioned into different subsets of data according to the customary ratio of *70:20:10* for training, validation, and testing purposes.

6.2 Scenarios for training

The promptness and accuracy of the object detectors are evaluated. These are the trade-off factors for them to generalize over the various types of disparities in the wild. Therefore, we

craft such diverse dataset variations to achieve our goal. Besides this, we craft an auxiliary dataset by introducing stylized/ natural variations in PASCAL VOC dataset, which enriches our small dataset quantitatively in the residual training. The false images are produced by the COPGAN framework, which is trained using the SYNTHIA dataset for obtaining scalable results. Moreover, the two-stage detectors and one-stage detector are also applied to the images generated by COPGAN to assess the robustness of the fine-tuned models. Initially, we assess the balance between the accuracy and speed of both one-stage and two-stage object detectors on image samples that are not part of the training domain. For this purpose, the models are fed with the images given by the SYNTHIA dataset's training class which is considered as *# setup 1* base experiment. We also test these models in unseen and new environments which have other adverse weather conditions depicted in the image samples in Fig. 13. These images are collected from the Internet source (*# setup 2)* for providing diversity to the trained models as shown in Fig. 14. The *# setup 3* investigates the variational influence of the augmented images on the object detectors. This again improves the diversity of our training set. For this purpose, we concatenate the COPGAN-generated fake images with the SYNTHIA dataset. Lastly, the *# setup 4* incorporates all three weather modelings

Fig. 13 Object Detection in (a) summer to winter (b) summer to night (c) summer to rainy image translations

Fig. 14 Samples of internet collected images

i.e., $summer \rightarrow winter$, $summer \rightarrow rainy$, and $summer \rightarrow night$. The above-stated scenarios for fine-tuning our object detectors in the exhaustive evaluation are briefed in the Table 7. In the further sections, we analyze the robustness of these models against the new unseen environments by incorporating yet another extensive irregular variation in the wild into the training set.

7 Experimental results

We evaluate the generalization efficiency of *SDY* by taking three approaches and four setups discussed in the robustness evaluation section and exhaustive evaluation section respectively.

7.1 Detection results in combined dataset

Firstly, PASCAL VOC dataset is used for training the DNN models *SDY* and the nuScenes dataset is utlized to fine-tune YOLOv5. Secondly, the combined dataset i.e., *PASCAL VOC+ augmented+ stylized images + COPGAN generated synthetic images using SYNTHIA dataset*

Table 7 Varied sets of training approaches and setups for robustness and exhaustive investigations of object detectors

Types of Datasets		Robustness Assessment			Exhaustive Evaluation			
		A1	A2	A3	S1	S2	S3	S4
BDD		✓	✓	✓	✓	✓	✓	✓
Basic Augmentation	**Gaussian Blur Transformed**	✓	✓	✓	✓	✓	✓	✓
	Gray Scale Transformed	✓	✓	✓	✓	✓	✓	✓
	Invert Transformed	✓	✓	✓	✓	✓	✓	✓
	Jitter Transformed	✓	✓	✓	✓	✓	✓	✓
	Solarise Transformed	✓	✓	✓	✓	✓	✓	✓
Internet source						✓		
Summer->winter(s2w)		✓	✓	✓			✓	✓
Summer->rainy(s2r)		✓	✓	✓			✓	✓
Summer->night(s2n)		✓	✓	✓			✓	✓
Cezanne style		✓						
Monet style			✓					
Ukiyoe style				✓				
Vangogh style		✓						

is used for testing. Moreover, to test YOLOv5, we fine-tuned the model with the combined dataset of *nuScenes+augmented+COPGAN generated synthetic images* and tested the model on real-world images captured in BITS campus. As shown in Fig. 13, the weather modeling introduced in these images shows good performance of object detectors on these unseen variations in the wild. Figure 15 shows the tested results on real-world images. We have also tested the model on the night images to check the resilience of the model trained in a particular operational condition. Table 9-11 compute the average precision (AP) of the *SDY* over different vehicular instances.

7.2 Stylized images

The three approaches employed in our investigations comes under AdaIN fast style transfer technique, which casts an input sample under the source sample representation. We have kept our training dataset 50% stylized to accentuate the features based on the shape in our trained model. The AdaIN styles are described in detail in the stylized weather modeling section above. Figure 16 describes the testing results of the SSD and Detectron when the stylized images are given as an input to these trained object detectors.

Fig. 15 YOLOv5 object detection in the real-world dataset captured in our university campus (BITS Pilani, India)

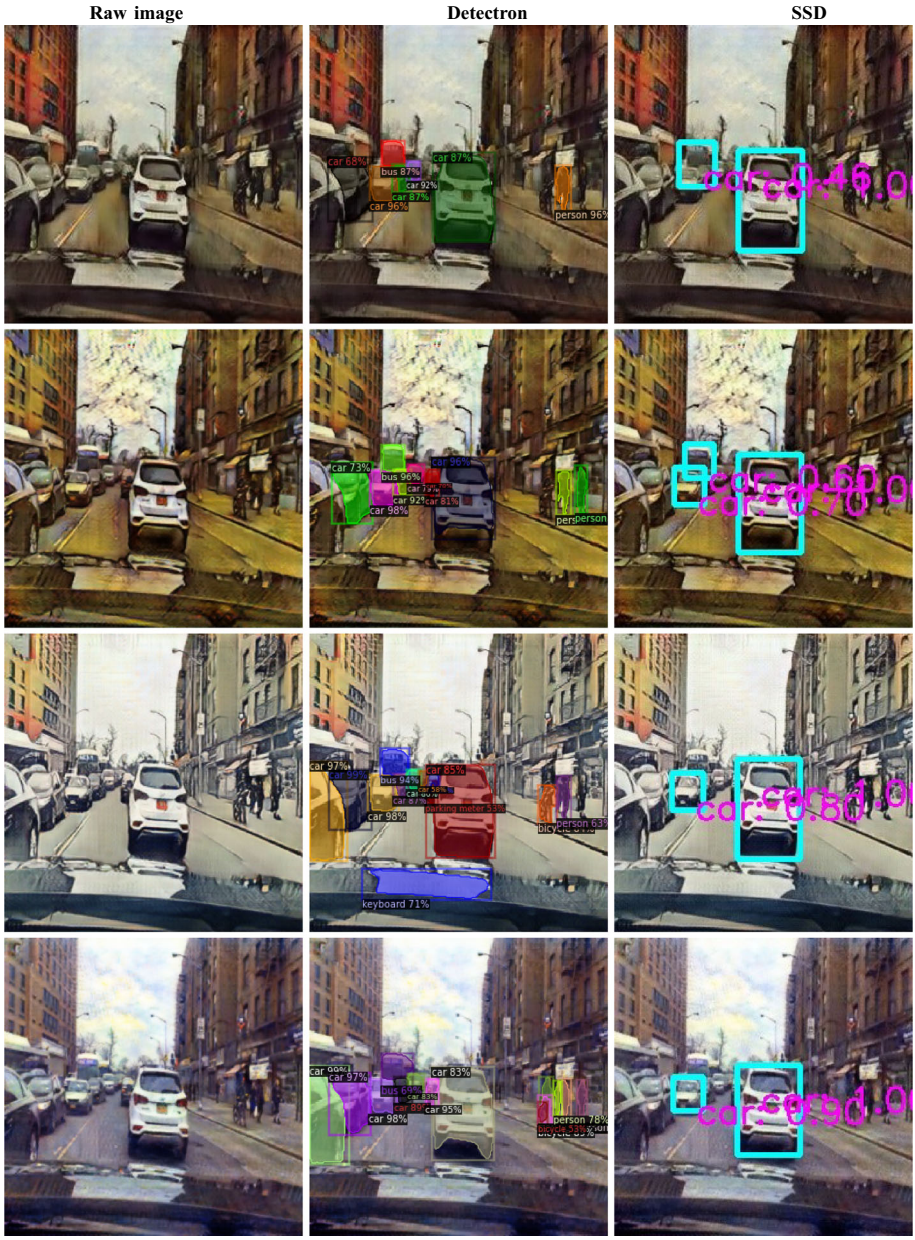

Fig. 16 Testing results of detection by methods Detectron and SSD on style transferred images. The stylized images chosen are of 4 different types: (a) Cezanne (b) Vangogh (c) Ukiyoe (d) Monet

7.3 Testing results of augmented images

For a fair comparison, the training set does not include these augmentations. These input samples are rendered to these augmentations and produce the results as shown in the Figs. 17 and 18.

Fig. 17 Testing results of Detectron and SSD on augmented images with five augmented styles: GB- Gaussian Blur, GS-Gray Scale, IT-Invert Transform, ST- Solarise Transform, JT- Jitter Transform

8 Robustness evaluation

Once the object detection models are trained with the above-described parameters and settings, we conduct a rigorous evaluation to claim the performance of the object detectors in varied scenarios. This evaluation is vital for the robust functioning of the object detection models to perform better in real-world conditions. Henceforth, our testing set is distorted to discover the impact of the discussed approaches, which entails five dissimilar corruptions. These distortions include all the augmented styles. The stated approaches in assessing the object detector's robustness are shown in the Table 7. For the evaluation of the robustness of the suggested strategies, an assumption of the mean performance degradation (mPD) *(in* (29)*)* is taken for the distorted set of the testing database at a particular threshold of 50 IoU [48]. Furthermore, the relative performance degradation under corruption (rPD) is measured in (30) as follows:

$$mPD = \frac{1}{L_s} \sum_{s \in S} AP_{50}^{s} \tag{29}$$

Fig. 18 Testing results of YOLOv5 on augmented images with five augmented styles: GB- Gaussian Blur, GS-Gray Scale, IT-Invert Transform, ST- Solarise Transform, JT- Jitter Transform

$$rPD = \frac{rPD}{AP_{50}^{ns}} \tag{30}$$

where L_s is the total number of severity levels, AP_{50}^{s} denotes mean average precision under distortion with s level of severity, AP_{50}^{ns} denotes mAP of our model on natural data with no distortions.

Approach 1 and approach 2 (A1 and A2) The statistical results in Table 8 authenticate that the integration of stylized images with different style renderings can lessen the vulnerability of *SDY* to various distortions. This happens because of the reduced texture bias by the produced stylized images and mends the detection frameworks to learn the illustrations centered on the shape of various objects. This makes our trained model stronger against different distortions because of the rendered stylized images, which moderate the biases of the texture.

Table 8 Calculated results of SSD (model #1) and Detectron (model #2) in the robustness evaluation for two severity levels

Models	S=1	S=2	AP_{50}^{ns}	mPD	rPD
SSD	68	63	77	65.5	85.06
Detectron	72	66	82	69	84.14

Approach 3 (A3) As compared to *A1 and A2* particularly, integrating the AdaIN styles into the weather simulated environments along with the augmented images (in Fig. 17) have shown a better performance in the robustness which can be witnessed in Fig. 16. The performance of the object detectors in *A3* is quantitatively evidenced in the Tables 9, 10 and 11.

8.1 Exhaustive evaluation

To assess our fine-tuned object detectors, we have adopted the Average Precision (AP) metric at different IoU thresholds (50,75) as shown in Tables 9-11. We have also calculated mAP of all the three models in Tables 12, 13, 14 and 15. It provides a performance summary of the model in terms of precision and recall for every class. *Setup 1 (S1)* SSD highly underperforms as compared to the Detectron to the detriment of a significant fall in computational speed. YOLOv5 also performs better all the varied scenarios but slightly less than SSD and Detectron. *Setup 2 (S2)* In this scenario, we have aggregated image samples gathered from the Internet source to the PASCAL VOC and nuScenes dataset utilized in the fine-tuning our models. *Setup 3 (S3)* In all the cases the average precision of both the models is incremented by the augmented images in the third scenario. *Setup 4 (S4)* Tables 9-11 authenticate that in varied weather scenarios or incorporating weather image translations enhances the results of all three models.

9 Comprehending loss function

For each image $x \in X''$, the image translation cycle should be capable of generating x back to its real form, expressed as $x \rightarrow Gen(x) \rightarrow F(Gen(x)) \approx x$. This property is referred

Table 9 AP of Detectron on the BDD Test Dataset using Varied Quality of Images

Scenarios	Car	Bus	Pedestrian	Bicycle	Signboard	Traffic Light	Auto	Bike
Data Augmentations								
Gaussian Blur	98	89	99	85	95	70	87	85
Gray Scale	99	91	96	98	96	70	83	82
Invert Transform	94	79	86	93	97	71	83	89
Jitter	99	70	98	98	97	69	85	88
Solarise	98	52	98	97	97	68	87	88
AdaIN Artistic Style Renderings								
Monet	98	69	78	53	90	67	40	75
Cezzane	96	87	96	68	89	63	43	71
Ukiyoe	98	94	63	97	89	63	41	69
Vangogh	98	96	80	58	90	66	43	77
COPGAN generated weather translations								
Summer → winter	**93**	**92**	**80**	**94**	**99**	**95**	**88**	**80**
Summer → rain	**94**	**95**	**71**	**91**	**96**	**98**	**89**	**81**
Summer → night	**98**	**97**	**84**	**89**	**97**	**98**	**89**	**82**

Table 10 AP of SSD on the BDD Test Dataset using Varied Quality of Images

Scenarios	Car	Bus	Pedestrian	Bicycle	Signboard	Traffic Light	Auto	Bike
Data Augmentations								
Gaussian Blur	93	55	68	86	98	65	83	50
Gray Scale	98	57	60	80	95	70	79	49
Invert Transform	95	40	70	75	96	71	80	55
Jitter	99	40	62	42	99	66	90	56
Solarise	96	49	65	44	99	63	90	60
AdaIN Artistic Style Renderings								
Monet	98	40	42	42	61	67	38	46
Cezzane	96	87	68	60	61	60	44	53
Ukiyoe	80	79	70	65	63	60	41	57
Vangogh	70	60	55	62	66	66	36	40
COPGAN-generated Weather Translations								
Summer → winter	**98**	**85**	**55**	**88**	**99**	**89**	**98**	**77**
Summer →rain	**80**	**48**	**88**	**89**	**95**	**88**	**99**	**72**
Summer →night	**93**	**70**	**83**	**85**	**96**	**86**	**98**	**70**

to as forward cycle consistency. A similar phenomenon takes place for image $y \in Y$. Gen and F should also fulfill backward cycle consistency $y \rightarrow F(y) \rightarrow Gen(F(y)) \approx y$. GAN architectures perform efficiently when they take task-oriented loss into consideration. Hence,

Table 11 AP of YOLOv5 trained on nuScenes dataset and tested on different datasets having the varying quality of images

Scenarios	Auto	Car	Bus	Pedestrian	Bicycle	Signboard	Truck	Bike
Data Augmentations								
Gaussian Blur	88	98	70	83	86	93	96	83
Gray Scale	85	98	35	80	98	89	94	89
Invert Transform	82	97	77	70	70	88	96	85
Jitter	82	97	60	80	70	89	96	96
Solarise	89	97	54	86	44	91	96	90
AdaIN Artistic Style Renderings								
Monet	98	97	42	94	61	67	38	46
Cezzane	96	96	68	89	61	60	44	53
Ukiyoe	80	96	70	90	63	60	41	57
Vangogh	70	97	55	92	66	66	36	40
COPGAN-generated Weather Translations								
Summer → winter	**98**	**98**	**89**	**87**	**99**	**83**	**82**	**80**
Summer →rain	**80**	**98**	**89**	**20**	**95**	**88**	**88**	**78**
Summer →night	**93**	**98**	**85**	**68**	**96**	**93**	**88**	**80**

Table 12 mAP results of YOLOv5 in augmented images

Scenarios	Gaussian Blur	Gray Scale	Invert Transform	Jitter	Solarise
mAP	93	79	86	91	85

the $Total_{loss}$ incurred in the proposed architecture is depicted in the following (31):

$$Total_{loss} = OD_{loss} + \omega Loss_{adv} \tag{31}$$

where OD_{loss}: object detectors loss and $Loss_{adv}$: adversarial loss of the GAN system. For the sake of regularization without losing the model generality, we intend to minimize the $Total_{loss}$ by a weighting factor ω. The terms OD_{loss} and $Loss_{adv}$ are elaborated further. The OD_{loss} is expressed as follows in (32):

$$OD_{loss} = \frac{1}{N}(Classification_{loss} + \tau BB_{Reg_{loss}}) \tag{32}$$

where N: total no. of anticipated anchor boxes

$Classification_{loss}$: loss in classification, $BB_{Reg_{loss}}$: bounding box regression loss, τ: hyper-parameter $Classification_{loss}$ and $BB_{Reg_{loss}}$ are model specific losses. To make our network framework more robust to the image quality distortions/variations, we augment OD_{loss} with an adversarial objective $Loss_{adv}$ in the training regime. The entire value function of such a framework of COPGAN is formulated as in (33):

$$\begin{aligned} Loss_{adv} = E_{x\sim p_x(x)}(logDisc(R(x))) + E_{\bar{x}\sim p_{\bar{x}}(\bar{x})} \\ (log(1 - Disc(Gen(\bar{x})))) \end{aligned} \tag{33}$$

The generator Gen yields output $Gen(\bar{x})$ by taking an input $\bar{x}$. Discriminator $Disc$ differentiates the "original" image sample x with the "synthetic" augmented sample $\bar{x}$ by taking the outputs from the detector models, $R(x)$ and $Gen(\bar{x})$. Moreover, the result of the discriminator $Disc(.)$ denotes the probable certainty of the input source sample associated with the original distribution of data X. The goal of the generator to minimize (33) which is conflicting with the discriminator's objective to maximize (33).

9.1 Generator loss

The loss of the generator is determined by reducing the logarithmic factor of the inverted probability predicted by the discriminator for the synthetic instances. This calculation is averaged across each mini-batch of samples. In Fig. 19(a), the graph shows the recorded generator loss for a specific number of steps. However, the distribution of the graph is inconsistent and does not provide any conclusive information. This situation is not effective when the generator is not performing well and the discriminator is capable of confidently identifying counterfeits.

Table 13 mAP results of YOLOv5 in artistic renderings

Scenarios	Monet	Cezzane	Ukiyoe	Vangogh
mAp	92	88	81	90

Table 14 mAP results of YOLOv5 in COPGAN-generated weather translations

Scenarios	Summer → winter	Summer → rain	Summer → night
mAP	93	89	90

9.2 Discriminator loss

The training of the proposed COPGAN involves the reduction of the discriminator's log probability by the generator of correctly categorizing real and fake images. Figure 19(b) also shows an inconsistent behavior of the discriminator in all-weather translations (summer to rain and summer to night) when the training is performed. In this game of minimax, the goal of the generator is to maximize the likelihood of the discriminator being incorrect, which in turn maximizes the logarithmic probability of the discriminator. The loss function indicator is then reversed to create a familiar minimal loss function for the training of the generator.

9.3 Cycle consistency loss

The cycle consistency helps in transferring uncommon style elements between the two COPGANs, while maintaining the common content. Figure 19(c) depicts the cycle consistency loss on different steps of training of the proposed COPGAN. It adds an extra loss term to each generator to softly encourage cycle consistency and is used in both directions.

9.4 Identity loss

Identity loss takes the real image in domain B and inputs it into the generator. An identity mapping means that the output is the same as the input. The aim of the generator is to replicate the existing styles present within the input image. The pixel distance is used to determine this, and ideally, there is no difference between its input and its related outcome. The identity loss is zero. Identity loss in Fig. 19(d) is optionally added with the idea to enforce COPGAN to preserve the complete temperature or color structure of the image. This is achieved by providing the images present in domain A to the generator from domain B to domain A.

9.5 GPU utilization

Figure 19 captures the GPU utilization, which represents the percentage of time during a specific period in which one or more kernels were actively running on the GPU. We want the GPU to be 100% busy all the time performing data crunching. The GPU utilization graphs

Table 15 Mean average precision results of SSD and Detectron model

Scenarios	SSD AP_{50}	Mean AP (50,75)	Detectron AP_{50}	Mean AP(50,75)
PascalVOC	77	64.5	82	67.5
Augmented	68	54	72	57.5
Adain Style Transfer	63	47.5	66	51
s-w Cycle GAN Translated	70	57	77	62.5
s-n Cycle GAN Translated	71	57.5	78	63.5
s-r Cycle GAN Translated	70	56.5	76	61.5

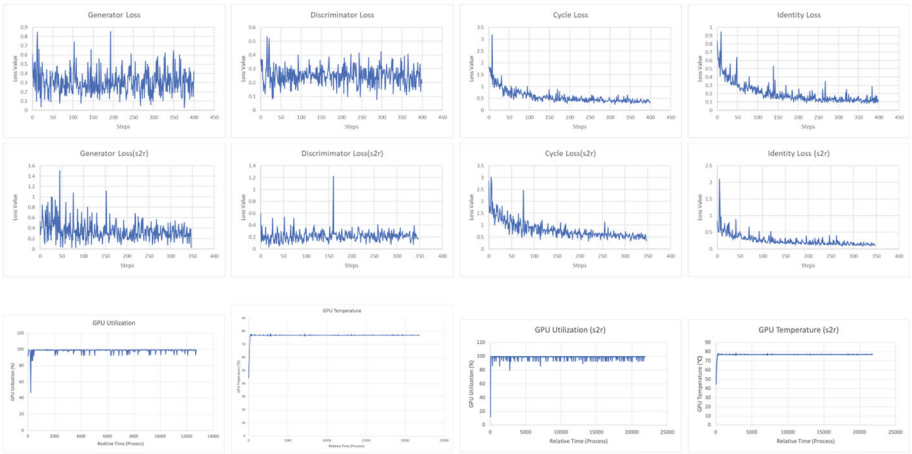

Fig. 19 Loss functions of (a) Generator (b) Discriminator (c) Cycle consistency (d) Identity, 1st row shows summer to winter translation results, 2nd row shows summer to rain translation results. The third row depicts the (a) GPU Utilization and (b) GPU Temperature graphs, first two shows summer to winter translation inferences and the last two shows summer-to-rain translation inferences

tell that they are about 90% busy. Well, the cause for this sub-optimal usage is because of the smaller batch size taken in our experiments. The GPU fetches a small portion of the data from its memory frequently and further cannot saturate the CUDA cores or the memory bus. Moreover, the graphs show the erratic distribution of GPU utilization because parallel training of the GAN network takes place. It is although not possible to utilize the GPU strength fully, but we can vary a few parameters to optimize the training of complex applications such as GAN. Two threads are concurrently being operated by the GAN and a lot of calculations are performed by each thread to get some output. In our experiment, one generator doesn't deliver the output at the exact same time another generator finishes calculating the same output. Therefore, we cannot claim a 100% GPU utilization in our experiments. Consequently, we found that it is virtually very difficult to orchestrate the GAN to remove the bottlenecks completely.

9.6 GPU temperature

We have considered another metric for measuring the system/resource performance during the experiments. This metric is vital in keeping the track of the GPU temperature in the course of executing experiments. To keep a record of the speed of the training, we ensure that there should not be any overheating which may damage the hardware. Figure 19 show that the busy GPU goes up to 75 degree Celsius and an idle GPU remains around 45 degrees Celsius.

9.7 Discussion and analysis

To study the effect of various augmentations and corruptions on the performance of three adopted object detection models, we tested these models on the augmented and corrupted datasets to observe the resiliency of our object detectors. We have gathered a diverse data and rendered it to different image quality variations such as winter, rain, night. The gray scale variations have also been taken into account. The reduced quality of images is characterized

by the stylized variations such as Monet, Cezzane, Ukiyoe, and Vangogh. The image quality is based on the subjective evaluation based on the sharpness, blurriness, overall appearance and sharp edges of the objects in the images. Figure 20 shows that we have taken three different schemes for training: one is the *standard dataset* i.e., the PASCAL VOC and SYNTHIA respectively, the second is the *stylized data* with different weather variations as well as some artistic renderings and the third one is the *combined dataset* which includes the original images along with weather, artistic, augmented, corrupted variations. Figure 20 shows that even after introducing the changes in the original dataset, the object detection models perform better than the standard dataset. This validates the resiliency of our models on the unseen variations of the environment. We have observed a similar behavior in the case of the SYNTHIA dataset for object detection, where the model is given training on COPGAN-generated synthetic images and then trained on the artistic and augmented images. We witness that the combined data suffers less from corruption than the standard data. Subsequently, the combination of the real data and synthetic/augmented data results in the best performance with decreased corruption severity and improved mAP. We can infer from Table 8 that these datasets improve the rPD, which shows better results for both SSD and Detectron models. We can validate that this behavior is fairly consistent. There are also some classes like pedestrian and bus in Table 11 that have not shown a significant amount of improvement and have less accuracy. It may be because of the inefficiency of the object detector to recognize certain shape formations of a particular vehicular instance. The baseline models may perform well in recognizing these instances without any augmentations but have underperformed in some cases like bus or pedestrian in a few of the image quality variations.

Additionally, we decide upon the mean opinion score (MOS) for every setup we have opted for in our work. Table 16 demonstrates the consistent outperformance of COPGAN against CycelGAN and UNIT. The vehicular instances and the style renderings are retained in the domain change of the image under the transformation process. This happens because the network of parsing (generator Gen) instructs the transformation process not to change the structural characteristics of the image. UNIT can deliver relatively good results compared to the CycleGAN as it attempts to retain the sophisticated-level semantics of an image.

9.8 Testing results of arbitrarily corrupted images

We have also listed the results of the two corruption types i.e., Gaussian Blur and rainy weather variation type. We test whether the robustness of our object detection models is generalizing better to real-world distortions such as motion blur, invert transforms, and different weather variations. We conducted our experiments under three different training schemes. For our first scheme, we trained our two models with clear pictures of PASCAL VOC dataset. For the second experiment we trained the models with SYNTHIA + COPGAN-generated synthetic images and for the last scheme, we combined all the synthetically generated images along

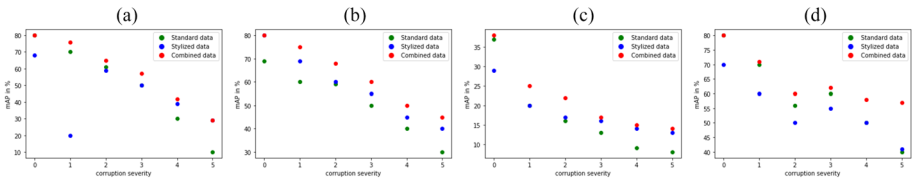

Fig. 20 Resiliency test of object detection models on three types of training schemes for(a) PASCAL VOC (b) SYNTHIA (c) Gaussian Blur (d) Rainy weather

Table 16 Degree of Style-render quality and vehicular instances preservation after weather transformations in augmented, Adain, winter translated, night translated, rainy translated setups using COPGAN from SYNTHIA and PASCAL VOC dataset

SYNTHIA			PASCAL VOC		
Dataset	GAN Architecture	MOS	Dataset	GAN Architecture	MOS
S2Aug	Cycle GAN	2.79	P2Aug	Cycle GAN	2.70
S2Aug	UNIT	2.76	P2Aug	UNIT	2.61
S2Aug	COPGAN	3.81	P2Aug	COPGAN	3.80
S2Adain	Cycle GAN	0.90	P2Adain	Cycle GAN	1.90
S2Adain	UNIT	1.84	P2Adain	UNIT	2.15
S2Adain	COPGAN	2.86	P2Adain	COPGAN	2.62
S2W	Cycle GAN	1.40	P2W	Cycle GAN	1.89
S2W	UNIT	2.10	P2W	UNIT	2.19
S2W	COPGAN	3.86	P2W	COPGAN	3.85
S2N	Cycle GAN	2.81	P2N	Cycle GAN	2.10
S2N	UNIT	2.89	P2N	UNIT	2.90
S2N	COPGAN	3.19	P2N	COPGAN	3.81
S2R	Cycle GAN	0.85	P2R	Cycle GAN	0.99
S2R	UNIT	2.0	P2R	UNIT	2.56
S2R	COPGAN	2.98	P2R	COPGAN	3.31

with the clear images. We find that if we train the models solely on the stylized images or on the weather variations, they have a have slight effect on the *SDY* performance. On the other hand, if we equip the models using the combined dataset, including all stylizations and variations, it improves the detection of the *SDY*. The models trained on the combined dataset are tested on the augmented images or any weather-transformed image, which attests to be a stronger evaluation of generalization. Figure 20(c) and (d) show the combined data performance by displaying higher *mAP* with lower values of corruption. We discover that the object detectors trained over a combined database of images yield better performance.

Therefore, to measure the effect of different image augmentations, corruptions, and training schemes, we evaluated the object detection models *SDY* on different types of datasets. Both the models show *rPD* values of 85.06% and 85.14%, respectively, on the corrupted datasets. We investigated the impact of these corruptions on the models, and as a result, the object detectors demonstrated increased robustness to these variations introduced in the images. This stems from the underlying fact that variations such as fog, night, rain, and snow change the image scene globally, but the local instances present in the image remain unaltered.

10 Conclusion

This manuscript addresses the issue of detecting objects in distinct real-world operational conditions which are generated by the proposed COPGAN model. To alleviate the problem of data insufficiency for domain adaptations in autonomous driving we also utilized AdaIN style-transfer models, and augmented images to train the object detectors. Further, we evaluated the performance of the existing OD models SDY, on COPGAN-generated sce-

narios as well as on other target domains that were not seen during training. Our empirical results improved the generalization of these contemporary OD models to our unseen and unstructured dataset. Specifically, we achieved rPD values of 85.06% and 85.14% for SSD and Detectron, respectively, on corrupted versions of the datasets. Integrating synthetic corruptions into our dataset resulted in good performance of the object detectors in real-world scenarios as it enriched the inter-domain database for the models.

Our main focus was to enable the object detectors employed in the ADAS/AVs to learn the domain adaptations occurring in the real world, thereby, facilitating seamless detection of vehicles in challenging situations (COPGAN-generated weather translations, artistic images, augmented images). All three models (*SDY*) exhibited higher values in the AP_{50} threshold for detecting different vehicular instances. Therefore, we conclude that these findings can be utilized in Advanced Driver-Assistance Systems (ADAS) to effectively adapt to inter-domain changes and accurately locate vehicular instances on the road. Our proposed COPGAN model can be utilized for synthetic image generation in giving varied environments for testing the working of ADAS in AVs without actually exposing the AVs to full-fledged on field-testing. Our future work will try to expand the database with yet more variations, the AVs encounter especially in the Indian scenarios. We envision it to make suitable for heterogeneous multitask learning. Our future goals include covering various vision such as semantic segmentation, object tracking, and lane detection. We wish to establish an autonomous driving testbed on which the vehicle navigates through the adversities of the environment with very less loss of accuracy. As a part of limitations, we have restricted ourselves to only certain variations which does not cover Indian scenarios and simulated environments. The simulated environments required a complex setup for the software-in-loop testing. Also, the inclusion of almost all the stochastic Indian scenarios was challenging. Therefore, we wish to cover the on-road and off-road case scenarios in the testing database for the SOTA object detection models.

Acknowledgements This work is carried out in the Disruptive Technologies lab, which is supported by the Department of Science and Technology (DST), Govt. of India, in the form of FIST Level-1 grant to the Department of CSIS, BITS Pilani. This work is also supported by CHANAKYA Fellowships of IITI DRISHTI CPS Foundation under the National Mission on Interdisciplinary Cyber-Physical System (NM-ICPS) of the Department of Science and Technology, Government of India. This work is also supported by I-DAPT HUB FOUNDATION, IIT (BHU), Varanasi for the project ref. no. I-DAPT/IIT (BHU)/2023-24/ Project Sanction/44. dated 19-09-2023.

Author Contributions **Oshin Rawlley:** Conceptualization, Methodology, Software, Investigation, Formal Analysis, Writing - original draft. **Shashank Gupta:** Conceptualization, Methodology, Validation, Writing - review and editing, Supervision. **Hardik Katehara:** Software. **Siddharth Katyal:** Software. **Yashvardhan Batwara:** Software

Data Availability The data that support the findings of this study are available within the article. The raw data that supports the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflicts of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Nie C, Zhou S, Zhang H, Sun Z (2022) Monocular vision based perception system for nighttime driving. In 2022 8th International conference on control, automation and robotics (ICCAR), pp 258–263. IEEE

2. SAE Taxonomy (2018) Definitions for terms related to driving automation systems for on-road motor vehicles. SAE: Warrendale, PA, USA, 3016
3. Urmson C, Anhalt J, Bagnell D, Baker C, Bittner R, Clark MN, Dolan J, Duggins D, Galatali T, Geyer C et al (2008) Autonomous driving in urban environments: Boss and the urban challenge. J Field Robot 25(8):425–466
4. Bhat A, Aoki S, Rajkumar R (2018) Tools and methodologies for autonomous driving systems. Proceedings of the IEEE 106(9):1700–1716
5. Li W, Pan CW, Zhang R, Ren JP, Ma YX, Fang J, Yan FL, Geng QC, Huang XY, Gong HJ et al (2019) Aads: Augmented autonomous driving simulation using data-driven algorithms. Sci Robot 4(28):eaaw0863
6. Lee EA (2016) Fundamental limits of cyber-physical systems modeling. ACM Trans Cyber-Phys Syst 1(1):1–26
7. Yu H, Li X (2023) Data-driven parameterized corner synthesis for efficient validation of perception systems for autonomous driving. ACM Trans Cyber-Phys Syst
8. Wang R, Zhao H, Xu Z, Ding Y, Li G, Zhang Y, Li H (2023) Real-time vehicle target detection in inclement weather conditions based on yolov4. Front Neurorobot
9. Lin CT, Kew JL, Chan CS, Lai SH, Zach C (2023) Cycle-object consistency for image-to-image domain adaptation. Pattern Recognit 138:109416
10. Essich M, Rehmann M, Curio C (2023) Auxiliary task-guided cyclegan for black-box model domain adaptation. In Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 541–550,
11. Lee Suhyeon, Lee Sangyong, Seong Hongje, Hyun Junhyuk, Kim Euntai (2023) Fallen person detection for autonomous driving. Expert Syst Appl 213:119242
12. Khosravian Amir, Amirkhani Abdollah, Kashiani Hossein, Masih-Tehrani Masoud (2021) Generalizing state-of-the-art object detectors for autonomous vehicles in unseen environments. Expert Syst Appl 183:115417
13. Hsu CC, Kang LW, Chen SY, Wang IS, Hong CH, Chang CY (2023) Deep learning-based vehicle trajectory prediction based on generative adversarial network for autonomous driving applications. Multimed Tools Appl 82(7):10763–10780
14. Mullick K, Jain H, Gupta S, Kale AA (2023) Domain adaptation of synthetic driving datasets for real-world autonomous driving. arXiv:2302.04149
15. Zhu JY, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision, pp 2223–2232
16. Chawla NV, Bowyer KW, LO Hall, Kegelmeyer WP (2002) Smote synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
17. Berthelot D, Schumm T, Metz L (2017) Began: Boundary equilibrium generative adversarial networks. arXiv:1703.10717
18. Lin CT, Huang SW, Wu YY, Lai SH (2020) Gan-based day-to-night image style transfer for nighttime vehicle detection. IEEE Trans Intell Trans Syst 22(2):951–963
19. Liang D, Wang R, Tian X, Zou C (2019) Pcgan: Partition-controlled human image generation. In Proceedings of the AAAI conference on artificial intelligence vol 33 pp 8698–8705
20. Arruda VF, Berriel RF, Paixão TM, Badue C, De Souza AF, Sebe N, Oliveira-Santos T (2022) Cross-domain object detection using unsupervised image translation. Expert Syst Appl 192:116334
21. Caesar H, Bankiti V, Lang AH, Vora S, Liong VE, Xu Q, Krishnan A, Pan Y, Baldan G, Beijbom O (2020) Nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 11621–11631
22. Cheng B, Wei Y, Shi H, Feris R, Xiong J, Huang T (2018) Revisiting rcnn: On awakening the classification power of faster rcnn. In Proceedings of the European conference on computer vision (ECCV), pp 453–468
23. Girshick R (2015) Fast r-cnn. In Proceedings of the IEEE international conference on computer vision, pp 1440–1448
24. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. Adv Neural Inf Process Syst, 28
25. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In Proceedings of the IEEE international conference on computer vision, pp 2961–2969
26. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
27. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) Ssd: Single shot multibox detector. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14, pp 21–37. Springer

28. Hu A, Russell L, Yeo H, Murez Z, Fedoseev G, Kendall A, Shotton J, Corrado G (2023) Gaia-1: A generative world model for autonomous driving. arXiv:2309.17080
29. Han T, Liu C, Yang W, Jiang D (2020) Deep transfer network with joint distribution adaptation: A new intelligent fault diagnosis framework for industry application. ISA Trans 97:269–281
30. Wang J, Chen Y, Yu H, Huang M, Yang Q (2019) Easy transfer learning by exploiting intra-domain structures. In 2019 IEEE international conference on multimedia and expo (ICME), pp 1210–1215. IEEE
31. Lin H, Liu Y, Li S, Qu X (2023) How generative adversarial networks promote the development of intelligent transportation systems: A survey. IEEE/CAA journal of automatica sinica
32. Camara F, Bellotto N, Cosar S, Weber F, Nathanael D, Althoff M, Wu J, Ruenz J, Dietrich A, Markkula G et al (2020) Pedestrian models for autonomous driving part ii: high-level models of human behavior. IEEE Trans Intell Trans Syst 22(9):5453–5472
33. Prakash CD, Karam LJ (2021) It gan do better: Gan-based detection of objects on images with varying quality. IEEE Trans Image Process 30:9220–9230
34. Bi R, Xiong J, Tian Y, Li Q, Choo KK (2022) Achieving lightweight and privacy-preserving object detection for connected autonomous vehicles. IEEE Int Things J
35. Xia Y, Monica J, Chao WL, Hariharan B, Weinberger KQ, Campbell M (2023) Image-to-image translation for autonomous driving from coarsely-aligned image pairs. In 2023 IEEE international conference on robotics and automation (ICRA), pp 7756–7762. IEEE
36. Wang X, Zhu Z, Huang G, Chen X, Lu J (2023) Drivedreamer: Towards real-world-driven world models for autonomous driving. arXiv:2309.09777
37. Guo Y, Liang RL, Cui YK, Zhao XM, Meng Q (2022) A domain-adaptive method with cycle perceptual consistency adversarial networks for vehicle target detection in foggy weather. IET Intell Trans Syst
38. Zhang H, Zhou L, Wang R, Knoll A (2023) Attention mechanism for contrastive learning in gan-based image-to-image translation. arXiv:2302.12052
39. Zareapoor M, Zhou H, Yang J (2020) Perceptual image quality using dual generative adversarial network. Neural Comput Appl 32(18):14521–14531
40. Couto GC, Antonelo EA (2023) Hierarchical generative adversarial imitation learning with mid-level input generation for autonomous driving on urban environments. arXiv:2302.04823
41. Porav H, Musat VN, Bruls T, Newman P (2020) Rainy screens: Collecting rainy datasets, indoors. arXiv:2003.04742
42. Volk G, Müller S, Von Bernuth A, Hospach D, Bringmann O (2019) Towards robust cnn-based object detection through augmentation with synthetic rain variations. In 2019 IEEE intelligent transportation systems conference (ITSC), pp 285–292. IEEE
43. Singh PK, Nandi SK, Nandi S (2019) A tutorial survey on vehicular communication state of the art, and future research directions. Veh Commun 18:100164
44. Isola P, Zhu JY, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1125–1134
45. Shan Y, Lu WF, Chew CM (2019) Pixel and feature level based domain adaptation for object detection in autonomous driving. Neurocomputing 367:31–38
46. Kim T, Cha M, Kim H, Lee JK, Kim J (2017) Learning to discover cross-domain relations with generative adversarial networks. In International conference on machine learning, pp 1857–1865. PMLR
47. Yi Z, Zhang H, Tan P, Gong M (2017) Dualgan: Unsupervised dual learning for image-to-image translation. In Proceedings of the IEEE international conference on computer vision, pp 2849–2857
48. Michaelis C, Mitzkus B, Geirhos R, Rusak E, Bringmann O, Ecker AS, Bethge M, Brendel W (2019) Benchmarking robustness in object detection: Autonomous driving when winter is coming. arXiv:1907.07484
49. Cai Zhaowei, Vasconcelos Nuno (2019) Cascade r-cnn: high quality object detection and instance segmentation. IEEE Trans Pattern Anal Mach Intell 43(5):1483–1498
50. Guan D, Huang J, Xiao A, Lu S, Cao Y (2021) Uncertainty-aware unsupervised domain adaptation in object detection. IEEE Trans Multimed 24:2502–2514
51. Kim JH, Batchuluun G, Park KR (2018) Pedestrian detection based on faster r-cnn in nighttime by fusing deep convolutional features of successive images. Expert Syst Appl 114:15–33
52. Duan Kaiwen, Dawei Du, Qi Honggang, Huang Qingming (2019) Detecting small objects using a channel-aware deconvolutional network. IEEE Trans Circ Syst Vid Technol 30(6):1639–1652
53. Girshick R, Donahue J, Darrell T, Malik J (2015) Region-based convolutional networks for accurate object detection and segmentation. IEEE Trans Pattern Anal Mach Intell 38(1):142–158
54. Prabhakar G, Kailath B, Natarajan S, Kumar R (2017) Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving. In 2017 IEEE region 10 symposium (TENSYMP), pp 1–6. IEEE

55. Chen Y, Li W, Sakaridis C, Dai D, Van Gool L (2018) Domain adaptive faster r-cnn for object detection in the wild. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3339–3348
56. Mirza M, Osindero S (2014) Conditional generative adversarial nets. arXiv:1411.1784
57. Liu MY, Breuel T, Kautz J (2017) Unsupervised image-to-image translation networks. Adv Neural Inf Process Syst, 30
58. Cao J, Mo L, Zhang Y, Jia K, Shen C, Tan M (2019) Multi-marginal wasserstein gan. Adv Neural Inf Process Syst, 32
59. Oprea S, Karvounas G, Martinez-Gonzalez P, Kyriazis N, Orts-Escolano S, Oikonomidis I, Garcia-Garcia A, Tsoli A, Garcia-Rodriguez J, Argyros A (2021) H-gan: the power of gans in your hands. In 2021 International joint conference on neural networks (IJCNN), pp 1–8. IEEE
60. Rangesh A, Zhang B, Trivedi MM (2020) Driver gaze estimation in the real world: Overcoming the eyeglass challenge. In 2020 IEEE Intelligent vehicles symposium (IV), pp 1054–1059. IEEE
61. Song Z, He Z, Li X, Ma Q, Ming R, Mao Z, Pei H, Peng L, Hu J, Yao D, et al (2023) Synthetic datasets for autonomous driving: A survey. arXiv:2304.12205
62. Chow Tsz-Yeung, Lee King-Hung, Chan Kwok-Leung (2023) Detection of targets in road scene images enhanced using conditional gan-based dehazing model. Appl Sci 13(9):5326
63. Zhang M, Zhang Y, Zhang L, Liu C, Khurshid S (2018) Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems. In Proceedings of the 33rd ACM/IEEE international conference on automated software engineering, pp 132–142
64. Yu H, Li X (2018) Intelligent corner synthesis via cycle-consistent generative adversarial networks for efficient validation of autonomous driving systems. In 2018 23rd Asia and South Pacific design automation conference (ASP-DAC), pp 9–15. IEEE
65. Kendall A, Gal Y, Cipolla R (2018) Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7482–7491
66. Huang X, Belongie S (2017) Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the IEEE international conference on computer vision, pp 1501–1510

Authors and Affiliations

Oshin Rawlley[1] · Shashank Gupta[1] · Hardik Kathera[1] · Siddharth Katyal[1] · Yashvardhan Batwara[1]

✉ Shashank Gupta
shashank.gupta@pilani.bits-pilani.ac.in

Oshin Rawlley
p20200063@pilani.bits-pilani.ac.in

Hardik Kathera
f20190089@pilani.bits-pilani.ac.in

Siddharth Katyal
f20190066@pilani.bits-pilani.ac.in

Yashvardhan Batwara
f20212224@pilani.bits-pilani.ac.in

[1] Department of Computer Science & Information Systems, Birla Institute of Technology and Science, Pilani, Rajasthan, India