Simulating Content Consistent Vehicle Datasets with Attribute Descent

Yue Yao[1] Liang Zheng[1] Xiaodong Yang[2]
Milind Naphade[2] Tom Gedeon[1]

[1]Australian National University
{yue.yao, liang.zheng, tom.gedeon}@anu.edu.au
[2]NVIDIA
yangxd.hust@gmail.com, mnaphade@nvidia.com

Abstract. This paper uses a graphic engine to simulate a large amount of training data with free annotations. Between synthetic and real data, there is a two-level domain gap, *i.e.,* content level and appearance level. While the latter has been widely studied, we focus on reducing the content gap in attributes like illumination and viewpoint. To reduce the problem complexity, we choose a smaller and more controllable application, vehicle re-identification (re-ID). We introduce a large-scale synthetic dataset VehicleX. Created in Unity, it contains 1,362 vehicles of various 3D models with fully editable attributes. We propose an attribute descent approach to let VehicleX approximate the attributes in real-world datasets. Specifically, we manipulate each attribute in VehicleX, aiming to minimize the discrepancy between VehicleX and real data in terms of the Fréchet Inception Distance (FID). This attribute descent algorithm allows content domain adaptation (DA) orthogonal to existing appearance DA methods. We mix the optimized VehicleX data with real-world vehicle re-ID datasets, and observe consistent improvement. With the augmented datasets, we report competitive accuracy. We make the dataset, engine and our codes available at https://github.com/yorkeyao/VehicleX.

Keywords: vehicle retrieval, domain adaptation, synthetic data

1 Introduction

Data synthesis, as can be conveniently performed in graphic engines, provides valuable convenience and flexibility for the computer vision area [25,28,27,35,30]. One can synthesize a large amount of training data under various combinations of environmental factors even from a small number of 3D object/scene models. However, there exists a huge domain gap between synthetic data and real-world data [14,27]. In order to effectively alleviate such a domain gap, it should be addressed from two levels: **content level** and **appearance level** [14]. While much existing work focuses on appearance level domain adaptation [7,11,44], we focus on the content level, *i.e.,* learning to synthesise data with similar content to the real data, as different computer vision tasks require different image contents.

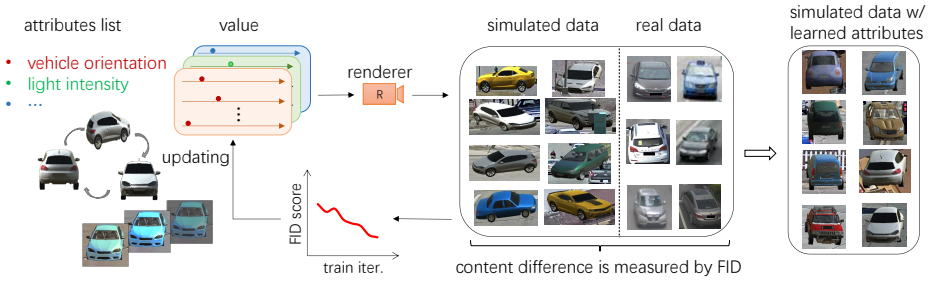

Fig. 1. System workflow. (**Left:**) given a list of attributes and their values, we use a renderer (*i.e.,* Unity) for vehicle simulation. We compute the Fréchet Inception Distance (FID) between the synthetic and real vehicles to indicate their distribution difference. By updating the values of attributes using the proposed attribute descent algorithm, we can minimize FID along the training iterations. (**Right:**) we use the learned attributes values that minimize FID to generate synthetic data to be used for re-ID model training.

Our system is designed based on the following considerations. It is expensive to collect large-scale real-world datasets for muti-camera system like re-ID. During annotation, one needs to associate an object across different cameras, which is a difficult and laborious process as objects might exhibit very different appearances in different cameras. In addition, there also has been an increasing concern over privacy and data security, which makes collection of large real datasets difficult [26,40]. On the other hand, we can see that datasets can be very different in their content. Here content means the object layout, illumination, and background in the image. For example, the VehicleID dataset [18] consists mostly of car rears and car fronts, while vehicle viewpoints in the VeRi-776 dataset [20] cover a very diverse range. Though the VehicleID dataset has a large number of identities which is useful for model training, this content-level domain gap might cause a model trained on VehicleID to have poor performance on VeRi. Most existing domain adaptation methods work on the pixel level or the feature level so as to allow the source and target domains to have similar appearance or feature distributions. However, these approaches are not capable of handling content differences, as can often be encountered when training on synthetic data and testing on real data.

Based on above considerations, we aim to utilize flexible 3D graphic engine to 1) scale up the real-world training data without labeling and privacy concerns, and 2) build synthetic data with *less content domain gap* to real-world data. To this end, we make contributions from two aspects. First, we introduce a large-scale synthetic dataset named VehicleX, which lays the foundation of our work. It contains 272 backbone models, with different colored textures, and creates 1,362 different vehicles. Similar to many existing 3D synthetic datasets such as PersonX [30] and ShapeNet [5], VehicleX has editable attributes and is able to generate a large training set by varying object and environment attributes. Second, based on the VehicleX, we propose an attribute descent method which

automatically configures the platform attributes, such that the synthetic data shares similar content distributions with the real data of interest. As shown in Figure 1, specifically, we manipulate the range of five key attributes closely related to the real dataset content. To measure the distribution discrepancy between the synthetic and real data, we use the FID score and aim to minimize it. In each epoch, we optimize the values of attributes in a specific sequence.

We show the effectiveness of attribute descent by training with VehicleX only and joint training with real-world datasets. The synthetic training data with optimized attributes can improve re-ID accuracy under both settings. Furthermore, under our joint training scheme, with VehicleX data, we achieve competitive re-ID accuracy with the state-of-the-art approaches, validating the effectiveness of learning from synthetic data. A subset of VehicleX has been used in the 4th AICITY challenge [23].[1]

2 Related Work

Vehicle re-identification has received increasing attention in the past few years, and many effective systems have been proposed [15,36,33,46], generally with specially designed or fine-tuned architectures. In this paper, our baseline system is built with commonly used loss functions [41,9,32] with no bells and whistles. Depending on the camera conditions, location and environment, existing vehicle re-ID datasets usually have their own distinct characteristics. For example, images in the VehicleID [18] are either captured from the car front or the back. In comparison, the VeRi-776 [20] includes a wider range of viewpoints. The recently introduced CityFlow [34] has distinct camera heights and backgrounds. Apart from dataset differences, there also exists huge differences between cameras in a single dataset [45]. For example, a camera filming a crossroad naturally has more vehicles orientation than a camera on a straight road. Because of these characteristics, within a specific dataset, we learn attributes for each camera and simulate that filming environment in a 3D engine. As a result, our proposed data simulation approach will make synthetic data more similar to the real-world in key attributes, and thus can effectively augment re-ID datasets due to its strong ability in content adaptation.

Appearance(style)-level domain adaptation. Domain adaptation is often used to reduce the domain gaps between the distributions of two datasets. Till now, the majority of work in this field focuses on discrepancies in image style, such as real vs. synthetic [2] and real vs. sketch [24]. For example, some use the cycle generative adversarial network (CycleGAN) to reduce the style gap between two domains [11,29,7], as well as various constraints being exerted on the generative model such that useful properties are preserved. While these works have been shown to be effective in reducing the style domain gap, a fundamental problem remains to be solved, *i.e.*, the content difference.

Content-level domain adaptation, to our knowledge, has been discussed by only a few existing works [14,27]. For [27], their main contribution is clever

[1] https://www.aicitychallenge.org/

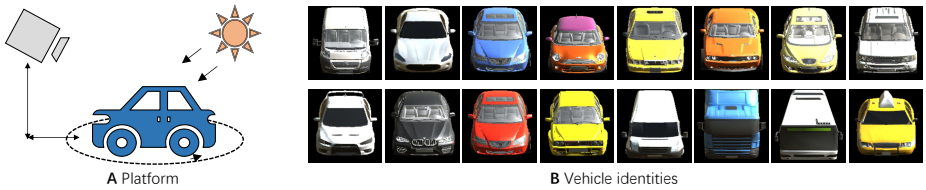

Fig. 2. The VehicleX engine. (A) An illustration of the rendering platform. We adjust the vehicle orientation, light direction and intensity, camera height, and the distance between the camera and the vehicle. (B) 16 different vehicle identities are shown.

usage of the task loss to guide the domain adaptation procedure. But for the re-ID task, we will search attributes for each camera but get task loss across camera systems. That is, task loss can only be gotten when all camera attributes are set. As a result, it is hard to optimise attributes for a single camera using loss from a cross-camera system. For [14], they use Graph Convolution Neural Network (GCN) to optimise the probability grammar for scene generation (*e.g.,* detection task). Their target is to solve the relationship between multiple objects. But in re-ID settings, we only have one object (car) to optimize. As their method cannot be directly used for the re-ID task, we adopt their advantages and make new contributions. On the one hand, we adopt the idea of Ruiz *et al.* [27] that represents attributes using predefined distributions. We are also motivated by Kar *et al.* [14], who suggest that some GAN evaluation metrics (*e.g.,* KID [4]) are potentially useful to measure content differences. In practice, we propose attribute descent, which does not involve random variables and has easy-to-configure step sizes.

Learning from 3D simulation. Due to low data acquisition costs, learning from 3D world is an attractive way to increase training set scale. But unlike other synthetic data (*e.g.,* images generated by GAN [43]), 3D simulation provides more accurate data labeling, flexibility in content generation and scalability in resolution, as GAN generated image may suffers from these problems. In the 3D simulation area, many applications exist in areas such as semantic segmentation [11,8,38], navigation [16], detection [14,12], object re-identification [30,33], *etc.* Usually, prior knowledge is utilized during data synthesis since we will inevitably need to determine the distribution of attributes in our defined environment. Tremblay *et al.* suggest that attribute randomness in a reasonable range is beneficial [35]. Even if it is random, we need to specify the range of random variables in advance. Our work investigates and learns these attribute distributions for vehicle re-ID.

3 VehicleX Engine

We introduce a large-scale synthetic dataset generator named VehicleX that includes three components: (1) vehicles rendered using the graphics engine Unity,

Table 1. Comparison of some real-world and synthetic vehicle re-ID datasets. "Attr" denotes whether the dataset has attribute labels (*e.g.,* orientation). Our identities are different 3D models, thus can potentially render an unlimited number of images under different environment and camera settings. VehicleX is released open source and can be used to generate (possess) an unlimited number of images (cameras).

Datasets		#IDs	#Images	#Cameras	# Attr
real	VehicleID [18]	26,328	222,629	2	✗
	CompCar [39]	4,701	136,726	-	✗
	VeRi-776 [20]	776	49,357	20	✓
	CityFlow [34]	666	56,277	40	✗
synthetic	PAMTRI [33]	402	41,000	-	✓
	VehicleX	1,362	∞	∞	✓

(2) a Python API that interacts with the Unity 3D engine, and (3) detailed labels including car type and color.

VehicleX has **a diverse range of realistic backbone models and textures**, allowing it to be able to adapt to the variance of real-world datasets. It has 272 backbones that are hand-crafted by professional 3D modelers. The backbones include ten mainstream vehicle types including sedan, SUV, van, hatchback, MPV, pickup, bus, truck, estate, sportscar and RV. Each backbone represents a real-world model. From these backbones, we obtain 1,362 variances (*i.e.,* identities) by adding various colored textures or accessories. A comparison of VehicleX with some existing vehicle re-ID datasets is presented in Table 1. VehicleX is three times larger than the synthetic PAMTRI dataset [33] in identities, and can potentially render an unlimited number of images from various attributes. In experiments, we will show that our VehicleX benefits real-world testing either when used alone or in conjunction with a real-world training set.

In this work, VehicleX can be set to training mode and testing mode. In training mode, VehicleX will render images with black background and these images will be used for attribute descent (see Section 4); in comparison, the testing mode uses random images (*e.g.,* from CityFlow [34]) as backgrounds, and generates attribute-adjusted images. In addition, to increase randomness and diversity, the testing mode contains random street objects such as lamp posts, billboards and trash cans. Figure 2 shows the simulation platform, and some sample vehicle identities.

We build the **Unity-Python interface** using the Unity ML-Agents toolkit [13]. It allows Python to modify the attributes of the environment and vehicles, and obtain the rendered images. With this API, given the attributes needed, users can easily obtain rendered images without expert knowledge about Unity. The code of this API is released together with VehicleX.

VehicleX is a large scale public 3D vehicle dataset, with real-world vehicle types. We focus on vehicle re-ID task in this paper but our proposed 3D vehicle models also has potential benefits for many other tasks, such as semantic segmentation, object detection, fine-grained classification, 3D generation or

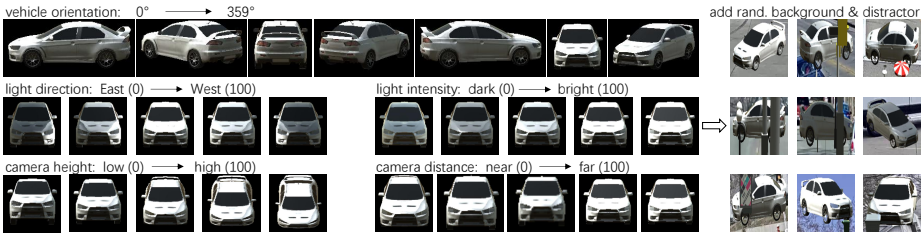

Fig. 3. (**Left:**) Attribute editing. We rotate the vehicle, edit light direction and intensity, or change the camera height and distance. Numbers in the bracket correspond to the attribute values in Unity. (**Right:**) We further add random backgrounds and distractors to the attribute-adjusted vehicles when they are used in the re-ID model.

reconstruction. It gives flexibility to computer vision systems to freely edit the content of the object, thus enabling new research in content-level image analysis.

4 Proposed Method

4.1 Attribute Distribution Modeling

Important attributes. For vehicle re-ID, we consider the following attributes to be potentially influential on the training set simulation and testing accuracy. Figure 3 shows examples of the attribute editing process.

- **Vehicle orientation** is the horizontal viewpoint of a vehicle and takes a value between 0°and 359°. In the real world, this attribute is important because the camera position is usually fixed and vehicles usually move along predefined trajectories. Therefore, the distribute of vehicle orientation of real world dataset is usually multimodal and tend to exhibit certain patterns under a certain camera view.
- **Light direction** simulates daylight as cars are generally presented in outdoor scenes. Here, we assume directional parallel light, and the light direction is modeled from east to west, which is the movement trajectory of the sun.
- **Light intensity** is usually considered a critical factor for re-ID tasks. Factors include glass refection and shadows will seriously influence the results. We manually defined a reasonable range for intensity from dark to light.
- **Camera height** describes the vertical distance from the ground, and significantly influences viewpoints.
- **Camera distance** determines the horizontal distance from vehicles. This factor has a strong effect on the vehicle resolution since the resolution of the entire image is predefined as 1920×1080. Additionally, the distance has slight impacts on viewpoints.

Distribution modeling. We model the aforementioned attributes with single Gaussian distributions or Gaussian Mixture Model (GMM). This modeling strategy is also used in Ruiz *et al.*'s work [27]. We denote the attribute list

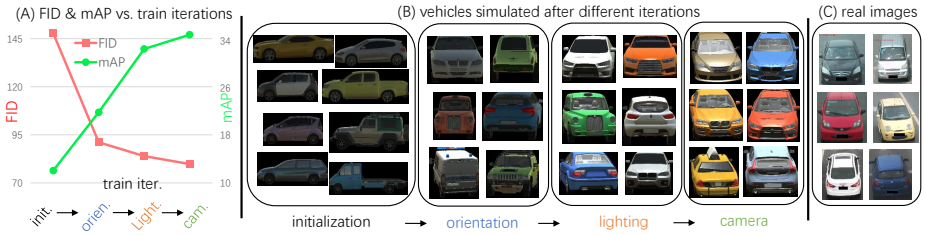

Fig. 4. Attribute descent visualization on the VehicleID [18]. (A) The FID-mAP curve through training iterations. The FID successively drops (lower is better) and domain adaptation mAP successively increases (higher is better) during attribute descent. For illustration simplicity, we use "light" to denote light direction and intensity, and use "cam." to denote camera height and distance. (B) We show the synthetic vehicles in each iteration. We initialize the attributes by setting orientation to right, the light intensity to dark, light direction to west, camera height to being equal to the vehicle, and camera distance to medium. The content of those images become more and more similar to (C) the target real images through the optimization procedure.

as: $\mathcal{A} = (a_1, a_2, ..., a_N)$, where N is the number of attributes considered in the system, and $a_i, i = 1, ..., N$ is the random variable representing the ith attribute.

For the vehicle orientation, we use a GMM to capture its distribution. This is based on our prior knowledge that the field-of-view of a camera covers either a road or an intersection. If we do not consider vehicle turning, there are rarely more than four major directions at a crossroad. In this work, we set a GMM with 6 components. For lighting conditions and camera attributes, we use four independent Gaussian distributions. Therefore, given N attributes, we optimize M mean values of the Gaussians, where $M \geq N$.

We speculate that the means of the Gaussian distributions or components are more important than the standard deviations because means reflect how the majority of the vehicles look. Although our method has the ability to handle variances, this would significantly increase the search space. As such, we predefine the values of standard deviations and only optimize the means of all the Gaussians $\boldsymbol{\mu} = (\mu_1, \mu_2, ..., \mu_M)$, where $\mu_i \in \mathbb{R}, i = 1, ..., M$ is the mean of the mth Gaussian. As a result, given the means $\boldsymbol{\mu}$ of the Gaussians, we can sample an attribute list as $\mathcal{A} \sim G(\boldsymbol{\mu})$, where G is a function that generates a set of attributes given means of Gaussian.

4.2 Optimization

The objective of our optimization is to train a model to generate a dataset that has a similar content distribution with respect to a target real dataset.

Measuring distribution difference. We need to precisely define the distribution difference before we apply any optimization algorithm. There potentially exists two directions: using the appearance difference, and the task loss on the

validation set. But as re-ID is a cross-camera task, it is indirect and difficult for us to optimise attributes for a single camera using loss from a cross-camera system. So we focus on the appearance difference. For the appearance difference, we use the Fréchet Inception Distance (FID) [10] to quantitatively measure the distribution difference between two datasets. Adversarial loss is not used as the measurement directly since there exists a huge appearance difference between synthetic and real data, and the discriminator would easily detect the specific detailed differences between real and generated, and yet not be useful.

Formally, we denote the sets of synthetic data and real data as X_s and X_r respectively, where $X_s = \{\mathcal{R}(\mathcal{A}_1), \cdots, \mathcal{R}(\mathcal{A}_K) | \mathcal{A}_k \sim G(\boldsymbol{\mu})\}$, and $\mathcal{R}$ is our rendering function through the 3D graphics engine working on a given attribute list $\mathcal{A}$ that controls the environment. K is the number of images in the synthetic dataset. For the FID calculation, we employ the Inception-V3 network [32] to map an image into its feature space. We view the feature as a multivariate real-valued random variable and assume that it follows a Gaussian distribution. To measure the distribution difference between two Gaussians, we resort to their means and covariance matrices. Under FID, the distribution difference between synthetic data and real data is written as,

$$\mathrm{FID}(X_s, X_r) = \|\boldsymbol{\mu}_s - \boldsymbol{\mu}_r\|_2^2 + Tr(\boldsymbol{\Sigma}_s + \boldsymbol{\Sigma}_r - 2(\boldsymbol{\Sigma}_s \boldsymbol{\Sigma}_r)^{\frac{1}{2}}), \tag{1}$$

where $\boldsymbol{\mu}_s$ and $\boldsymbol{\Sigma}_s$ denote the mean and covariance matrix of the feature distribution of the synthetic data, and $\boldsymbol{\mu}_r$ and $\boldsymbol{\Sigma}_r$ are from the real data.

Attribute descent. An important difficulty for attribute optimization is that the rendering function (through the 3D engine Unity) is not differentiable, so the widely used gradient-descent based methods cannot be readily used. Under this situation, there exist several methods for gradient estimation, such as finite-difference [14] and reinforcement learning [27]. However, these methods are developed in scenarios where there are many parameters to optimize. In comparison, our system only contains a few parameters, allowing us to design a more stable and efficient approach that is sufficiently effective in finding a close to global minimum.

We are motivated by coordinate descent, an optimization algorithm that can work in derivative-free contexts [37]. The most commonly known algorithm that uses coordinate descent is k-means [21]. Coordinate descent successively minimizes along coordinate directions to find a minimum of a function. The algorithm selects a coordinate to perform the search at each iteration. Compared with grid search, coordinate descent significantly reduces the search time, based on the hypothesis that each parameter is relatively independent. For our designed attributes, we study their independence in subsection 5.3.

Using Eq. 1 as the objective function, we propose attribute descent to optimize each single attribute in the attribute list. Specifically, we view each attribute as a coordinate in the coordinate descent algorithm. In each iteration, we successively change the value of an attribute to search for the minimum value of the objective function. Formally, for our defined parameters $\boldsymbol{\mu}$ for attributes list

$\mathcal{A}$, the objective is to find

$$\begin{aligned} \boldsymbol{\mu} &= \arg\min_{\boldsymbol{\mu}} \mathrm{FID}(X_s, X_r), \\ X_s &= \{\mathcal{R}(\mathcal{A}_1), \cdots, \mathcal{R}(\mathcal{A}_K) | \mathcal{A}_k \sim G(\boldsymbol{\mu})\}. \end{aligned} \tag{2}$$

We achieve this objective iteratively. Initially, we have

$$\boldsymbol{\mu}^0 = (\mu_1^0, \cdots, \mu_M^0), \tag{3}$$

At epoch j, we optimize a single variable μ_i^j in $\boldsymbol{\mu}$,

$$\begin{aligned} \mu_i^j &= \arg\min_{z \in S_i} \mathrm{FID}(X_s, X_r), \\ X_s &= \{\mathcal{R}(\mathcal{A}_1), \cdots, \mathcal{R}(\mathcal{A}_K) | \mathcal{A}_k \sim G(\mu_1^j, \\ &\quad \cdots, \mu_{i-1}^j, z, \mu_{i+1}^{j-1}, \cdots, \mu_M^{j-1})\}, \end{aligned} \tag{4}$$

where the $S_i, i = 1, ..., M$ define a specific search space for mean variable μ_i. For example, the search space for vehicle orientation is from 0° to 330° by 30° degree increments; the search space for camera height is the equally divided editable range with 9 segments. $j = 1, \cdots, J$ are the training epochs. One epoch is defined as all attributes being updated once. In this algorithm, we perform greedy search for the optimized value of an attribute in each iteration, and achieve a local minimum for each attribute when fixing the rest.

Discussion. In Section 5.3 we show that attribute descent (non-gradient solution) is superior to our implementation of reinforcement learning (gradient-based solution). Attribute descent, inherited from the coordinate descent algorithm, is simple to implement and steadily leads to convergence. It is a new optimization tool in the learning to synthesize literature and avoids drawbacks such as difficulty in optimization and sensitivity to hyper-parameters. That being said, we note that our method is effective in small-scale environments like vehicle bounding boxes where only a small number of attributes need to be optimized. In more complex environments, we suspect that reinforcement learning algorithms should also be effective.

5 Experiment

5.1 Datasets and Evaluation Protocol

We use three real-world datasets for evaluation. **VehicleID** [18] is at a large scale, containing 222,629 images of 26,328 identities. Half of the identities are used for training, and the other half for testing. Officially there are 3 test splits. The **VeRi-776** dataset [20] contains 49,357 images of 776 vehicles captured by 20 cameras. The vehicle viewpoints and illumination cover a diverse range. The training set has 37,778 images, corresponding to 576 identities; the test set has 11,579 images of 200 identities. There are 1,678 query images. The train / test

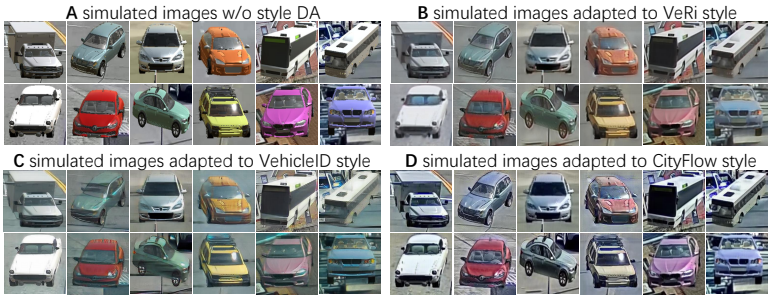

Fig. 5. Images w/ and w/o style domain adaptation. (A) Synthetic images without style domain adaptation. (B)(C)(D) We translate images in (A) to the style of VeRi, VehicleID and CityFlow, respectively, using SPGAN [7].

sets share the same 20 cameras. **CityFlow** [34] has more complex environments, and it has 40 cameras in a diverse environment where 34 are used in the training set. The dataset has in total 666 IDs where half are used for training and the rest for testing. We use mean average precision (mAP) and Rank-1 accuracy to measure the re-ID performance.

5.2 Implementation Details

Data generation. For the VehicleID dataset, we only optimize a single attribute list targeting the VehicleID training set. But most re-ID datasets like VeRi-776 and CityFlow are naturally divided according to multiple camera views. Since a specific camera view usually has stable attribute features (*e.g.,* viewpoint), we perform the proposed attribute descent algorithm on each individual camera, so as to simulate images with similar content to images from each camera. For example, we optimize 20 attribute lists using the VeRi-776 training set, which has 20 cameras. Attribute descent is performed for two epochs. One epoch is defined as all attributes in the list being updated once.

Image style transformation. We apply SPGAN [7] for image style transformation, which is a state-of-the-art algorithm in style level domain adaptive re-ID. Sample results are shown in Figure 5 and influence is shown in Table 2. Image translation models are trained using 112,042 images with random attributes as source domain and the training set in three vehicle datasets as target domain separately. When performing SPGAN for learned attributes data, we directly inference the learned attributes images, based on the fact that our learned attributes are a subset of the random range.

Table 2. Re-ID accuracy (mAP) w/ and w/o style DA when training with synthetic data only. We clearly observe style DA brings significant improvement and thus is necessary.

StyleDA	VehicleID	VeRi
✗	24.36	12.35
✓	**35.33**	**21.29**

Table 3. Method comparison on VehicleID in data augmentation. Our method is built on IDE [41] with the cross-entropy (CE) loss. Attribute descent consistently improves over both the baseline and random attributes, and is competitive compared with the state-of-the-art. "R" means training use real data only. "R+S" denotes that both synthetic data and real data are used in training. "Small", "Medium" and "Large" refers to the number of vehicles on the VehicleID test set [18].

Method	Data	Small			Medium			Large		
		Rank-1	Rank-5	mAP	Rank-1	Rank-5	mAP	Rank-1	Rank-5	mAP
RAM [19]	R	75.2	91.5	-	72.3	87.0	-	67.7	84.5	-
AAVER [15]	R	74.69	93.82	-	68.62	89.95	-	63.54	85.64	-
GSTE [1]	R	75.9	84.2	75.4	74.8	83.6	74.3	74.0	82.7	72.4
IDE (CE loss)	R	77.35	90.28	83.10	75.24	87.45	80.73	72.78	85.56	78.51
Ran. Attr.	R+S	80.2	93.98	85.95	76.94	90.84	82.67	73.45	88.66	80.55
Attr. Desc.	R+S	**81.50**	**94.85**	**87.33**	**77.62**	**92.20**	**83.88**	**74.87**	**89.90**	**81.35**

Table 4. FID values between the generated data and VehicleID after Epoch I and II (attribute descent is performed for two epochs). Different orders of attributes are tested. 'C', 'O' and 'L' refer to camera, orientation and lighting, respectively. After Epoch II, the FID values are generally similar, suggesting that the correlation among attributes is weak, and so they are mostly independent.

	C → O → L	C → L → O	O → L → C	O → C → L	L → C → O	L → O → C
FID (Epoch I)	98.38	99.57	78.67	80.94	104.84	81.20
FID (Epoch II)	78.42	77.18	77.96	79.54	78.48	77.06

Baseline configuration. For VeRi and VehicleID, we use the ID-discriminative embedding (IDE) [41]. We adopt the strategy from [22] which adds batch normalization and removes ReLU after the final feature layer. We also use the part-based convolution baseline (PCB) [31] on VeRi for improved accuracy. In PCB, we horizontally divide the picture into six equal parts and perform classification on each part. For CityFlow training, we use the setting from [22] using a combination of the cross-entropy loss and the triplet loss.

Experiment protocol. We evaluate our method on both vehicleX training and joint training settings. Under vehicleX training, we train our model on VehicleX and test on real world data. Under joint training, we combine the VehicleX data and real world data and perform two-stage training; testing is on the same real-world data.

Two-stage training is conducted in joint training with three real-world datasets [42]. We mix synthetic dataset and a real-world dataset in the first stage and finetune on the real-world dataset only in the second stage. Taking CityFlow for example, in the first stage, we train on both real and synthetic data. We classify vehicle images into one of the 1,695 (333 from real + 1,362 from synthetic) identities. In the second stage, we replace the classification

Table 5. Comparison of the Re-ID accuracy (mAP) of two stage training when performing joint training. We can see a significant performance boost from Stage I to Stage II.

	VehicleID	VeRi	CityFlow
Stage I	77.54	69.39	33.54
Stage II	**81.35**	**70.62**	**37.16**

Table 6. Re-ID test accuracy (mAP) on VehicleID test set (large) using various training datasets with [22]. The first four training sets are generated by random attributes, random search, LTS and attribute descent, respectively. The last two training sets are real-world ones. FID measures domain gap between the training sets and VehicleID.

	Ran. Attr.	Ran. Sear.	LTS	Attr. Desc.	VeRi	Cityflow
FID	134.75	109.94	95.27	77.96	-	-
mAP	22.00	26.35	32.21	35.33	38.59	45.57

Table 7. Method comparison when testing on VeRi-776. Both VehicleX training and joint training results are included. "R" means training with real data only, "S" represents training use synthetic data only and "R+S" denotes the joint training. VID→VeRi shows the result trained on VehicleID, test on VeRi and Cityflow→VeRi means the result trained on Cityflow, test on VeRi. In addition to some state-of-the-art methods, we summarize the results on top of two baselines, *i.e.,* IDE [41] and PCB [31].

Experiment	Method	Data	Rank-1	Rank-5	mAP
VehicleX Training	ImageNet	R	30.57	47.85	8.19
	VID →VeRi	R	59.24	71.16	20.32
	Cityflow → VeRi	R	69.96	81.35	26.71
	Ran. Attr.	S	43.56	61.98	18.36
	Attr. Desc.	S	51.25	67.70	21.29
Joint Training	VANet [6]	R	89.78	95.99	66.34
	AAVER [15]	R	90.17	94.34	66.35
	PAMTRI [33]	R+S	92.86	96.97	71.88
	IDE	R	92.73	96.78	66.54
	Ran. Attr.	R+S	93.21	96.20	69.28
	Attr. Desc.	R+S	93.44	97.26	70.62
	Attr. Desc. (PCB)	R+S	**94.34**	**97.91**	**74.51**

layer with a new classifier that will be trained on the real dataset (recognizing 333 classes). Table 5 shows significant improvements with this method.

5.3 Evaluation

Analysis of attribute descent process. Figure 4 shows how the re-ID accuracy and FID change during along the training iterations. We observe that attributes are successively optimized when FID decreases and mAP increases. Furthermore, from the slope of the FID curve we can see that orientation has the largest impact on the distribution difference and mAP, with a huge FID drop from 147.85 to 91.14 and large mAP increase from 12.1% to 21.94%. Lighting is the second most impactful (-7.2 FID, +10.7% mAP), and camera attributes are the third (-4.11 FID, +2.4% mAP).

Effectiveness of learned synthetic data. Learned synthetic data can be used as a training set alone, or in conjunction with real training data for data augmentation. We show the results of both cases on the three datasets in Table 3 (VehicleID), Table 7 (VeRi) and Table 8 (CityFlow). From these results we observe that when used as training data alone, learned attributes achieve much

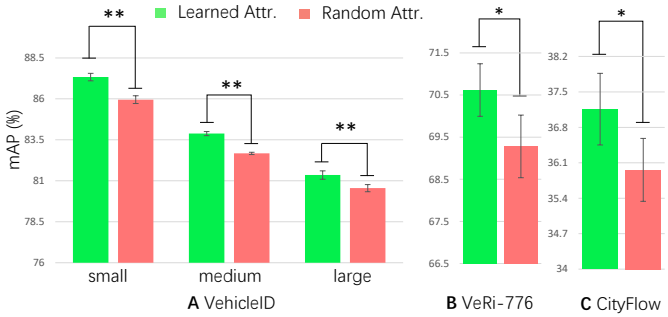

Fig. 6. Performance comparison between learned attributes and random attributes in joint training. We present mAP on three datasets and use statistical significance analysis to show the training stability. $*$ means statistically significant (*i.e.*, $0.01 < p$-value < 0.05) and $**$ denotes statistically very significant (*i.e.*, $0.001 < p$-value < 0.01).

higher re-ID accuracy than random attributes. For example, on the VeRi-776 dataset, attribute descent has a +7.69% improvement in Rank-1 accuracy over random attributes. Moreover, attribute learning also benefits the data augmentation setting. For example, on CityFlow and VeRi, the improvement of learned attributes over random attributes is +1.49% and +3.87% in Rank-1 accuracy, respectively. Although this improvement looks small in number, we show that the improvement is statistical significant (Figure 6). We note that the improvement of using synthetic data as a training set is more significant than for data augmentation. When the training set consists of only the synthetic data, a higher quality of attributes will have a more direct impact on the re-ID results.

Few dependencies between attributes. We proceed study on dependency by testing whether the order of attributes matters. From Table 4 it is clear that attribute orders do not affect the downward trend, the only clear dependency is the relationship between orientation and camera. If we learn camera attributes before orientation, the accuracy will be influenced. But such influence will be eliminated by performing the attribute descent twice. Based on the few dependencies between attributes, we make it possible to use attribute descent rather than grid search, saving computation time.

Attribute descent performs better than multiple methods: 1) random attribute 2) random search 3) LTS [27]. For LTS, we follow their ideas but we replace the task loss with FID score, since task loss is not generalised to a re-ID task. Our reproduced LTS uses the same distribution definition and initialization as attribute descent. In order to make a fair comparison, we report values from 200 iterations of training (*i.e.,* compute FID score 200 times). Random search is a strong baseline in hyper-parameter optimization [3]. In practice, we randomly sample attribute values 200 times and choose an attribute list with the best FID score. The result comparison is shown in Table 6. First, under the same task network, all learned attributes (*i.e., random search, LTS and attribute descend*) perform better than random attributes, in both FID and

Method	Data	R-1	R-20	mAP
BA [17]	R	49.62	80.04	25.61
BS [17]	R	49.05	78.80	25.57
PAMTRI [33]	R+S	59.7	80.13	33.81
IDE(CE+Tri.)	R	56.75	72.24	30.21
Ran. Attr.	R+S	63.59	82.60	35.96
Attr. Desc.	R+S	**64.07**	**83.27**	**37.16**

Table 8. Method comparison on CityFlow with joint training. Our baseline is built with a combination of the CE loss and the triplet loss [22]. Rank-1, Rank-20 and the mAP are calculated by the online server.

mAP, showing that learned attributes significantly improves the result, and that content differences matter. Second, random search does not perform well in a limited search time. It has been shown that random search performs well when there exists many less important parameters [3]. But in our search space, all attributes contribute to the distribution differences as shown in Figure 4, thus random search has no advantage in helping find important attributes. Third, LTS works but it does not find a better FID score than attribute descent. LTS seems to fall into a local optimum and does not reach a global one. A example of local optima is LTS outputs are either outputs of car front or rear, whereas the VehicleID contains both car front and rear. With a more hand-crafted design, we will definitely reach a better performing LTS framework. But at this stage, attribute descent is a simple realized method that finds a better solution with few iterations. It deserves to be a strong baseline in this field.

Comparison with the state-of-the-art. When the synthetic data is used in conjunction with real-world training set, we achieve very competitive accuracy compared with the state-of-the-art (Table 3, Table 7 and Table 8). For example on VehicleID (Small), our method is +5.6% higher than [1] in Rank-1 accuracy. On CityFlow, our method is higher than [33] by +7.32% in Rank-1 accuracy.

6 Conclusion

This paper study the domain gap problem between synthetic data and real data from the content level. That is, we automatically edit the source domain image content in a graphic engine so as to reduce the content gap between the synthetic images and the real images. We use this idea to study the vehicle re-ID task, where the usage of vehicle bounding boxes decreases the set of attributes to be optimized. Fewer attributes-of-interest and low dependencies between them allow us to optimize them one by one using our proposed attribute descent approach. We show that the learned attributes bring about improvement in re-ID accuracy with statistical significance. Moreover, our experiment reveals some important insights regarding the usage of synthetic data, *e.g.,* style DA brings significant improvement and two stage training is beneficial for joint training.

Acknowledgement

Dr. Liang Zheng is the recipient of Australian Research Council Discovery Early Career Award (DE200101283) funded by the Australian Government.

References

1. Bai, Y., Lou, Y., Gao, F., Wang, S., Wu, Y., Duan, L.Y.: Group-sensitive triplet embedding for vehicle reidentification. IEEE Transactions on Multimedia **20**(9), 2385–2399 (2018) 11, 14
2. Bak, S., Carr, P., Lalonde, J.F.: Domain adaptation through synthesis for unsupervised person re-identification. In: ECCV (2018) 3
3. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of machine learning research **13**(Feb), 281–305 (2012) 13, 14
4. Binkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying mmd gans. In: ICLR (2018) 4
5. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015) 2
6. Chu, R., Sun, Y., Li, Y., Liu, Z., Zhang, C., Wei, Y.: Vehicle re-identification with viewpoint-aware metric learning. In: ICCV (2019) 12
7. Deng, W., Zheng, L., Ye, Q., Kang, G., Yang, Y., Jiao, J.: Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification. In: CVPR (2018) 1, 3, 10
8. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: CVPR (2016) 4
9. Hermans, A., Beyer, L., Leibe, B.: In defense of the triplet loss for person re-identification. arXiv preprint arXiv:1703.07737 (2017) 3
10. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: NeurIPS (2017) 8
11. Hoffman, J., Tzeng, E., Park, T., Zhu, J.Y., Isola, P., Saenko, K., Efros, A., Darrell, T.: Cycada: Cycle-consistent adversarial domain adaptation. In: ICML (2018) 1, 3, 4
12. Hou, Y., Zheng, L., Gould, S.: Multiview detection with feature perspective transformation. arXiv preprint arXiv:2007.07247 (2020) 4
13. Juliani, A., Berges, V.P., Vckay, E., Gao, Y., Henry, H., Mattar, M., Lange, D.: Unity: A general platform for intelligent agents. arXiv preprint arXiv:1809.02627 (2018) 5
14. Kar, A., Prakash, A., Liu, M.Y., Cameracci, E., Yuan, J., Rusiniak, M., Acuna, D., Torralba, A., Fidler, S.: Meta-sim: Learning to generate synthetic datasets (2019) 1, 3, 4, 8
15. Khorramshahi, P., Kumar, A., Peri, N., Rambhatla, S.S., Chen, J.C., Chellappa, R.: A dual path modelwith adaptive attention for vehicle re-identification. In: ICCV (2019) 3, 11, 12
16. Kolve, E., Mottaghi, R., Gordon, D., Zhu, Y., Gupta, A., Farhadi, A.: Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474 (2017) 4
17. Kumar, R., Weill, E., Aghdasi, F., Sriram, P.: Vehicle re-identification: an efficient baseline using triplet embedding. arXiv preprint arXiv:1901.01015 (2019) 14
18. Liu, H., Tian, Y., Yang, Y., Pang, L., Huang, T.: Deep relative distance learning: Tell the difference between similar vehicles. In: CVPR (2016) 2, 3, 5, 7, 9, 11
19. Liu, X., Zhang, S., Huang, Q., Gao, W.: Ram: a region-aware deep model for vehicle re-identification. In: ICME (2018) 11
20. Liu, X., Liu, W., Ma, H., Fu, H.: Large-scale vehicle re-identification in urban surveillance videos. In: ICME (2016) 2, 3, 5, 9

21. Lloyd, S.: Least squares quantization in pcm. IEEE transactions on information theory **28**(2), 129–137 (1982) 8
22. Luo, H., Gu, Y., Liao, X., Lai, S., Jiang, W.: Bag of tricks and a strong baseline for deep person re-identification. In: CVPR Workshops (2019) 11, 12, 14
23. Naphade, M., Wang, S., Anastasiu, D.C., Tang, Z., Chang, M.C., Yang, X., Zheng, L., Sharma, A., Chellappa, R., Chakraborty, P.: The 4th ai city challenge. In: CVPR Workshops. pp. 626–627 (2020) 3
24. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: ICCV (2019) 3
25. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: Ground truth from computer games. In: ECCV (2016) 1
26. Ristani, E., Solera, F., Zou, R., Cucchiara, R., Tomasi, C.: Performance measures and a data set for multi-target, multi-camera tracking. In: ECCV Workshops (2016) 2
27. Ruiz, N., Schulter, S., Chandraker, M.: Learning to simulate. In: ICLR (2019) 1, 3, 4, 6, 8, 13
28. Sakaridis, C., Dai, D., Van Gool, L.: Semantic foggy scene understanding with synthetic data. International Journal of Computer Vision pp. 1–20 1
29. Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R.: Learning from simulated and unsupervised images through adversarial training. In: CVPR (2017) 3
30. Sun, X., Zheng, L.: Dissecting person re-identification from the viewpoint of viewpoint. In: CVPR (2019) 1, 2, 4
31. Sun, Y., Zheng, L., Yang, Y., Tian, Q., Wang, S.: Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In: ECCV (2018) 11, 12
32. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR (2016) 3, 8
33. Tang, Z., Naphade, M., Birchfield, S., Tremblay, J., Hodge, W., Kumar, R., Wang, S., Yang, X.: Pamtri: Pose-aware multi-task learning for vehicle re-identification using highly randomized synthetic data. In: ICCV (2019) 3, 4, 5, 12, 14
34. Tang, Z., Naphade, M., Liu, M.Y., Yang, X., Birchfield, S., Wang, S., Kumar, R., Anastasiu, D., Hwang, J.N.: Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In: CVPR (2019) 3, 5, 10
35. Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., Birchfield, S.: Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In: CVPR Workshops (2018) 1, 4
36. Wang, Z., Tang, L., Liu, X., Yao, Z., Yi, S., Shao, J., Yan, J., Wang, S., Li, H., Wang, X.: Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In: ICCV (2017) 3
37. Wright, S.J.: Coordinate descent algorithms. Mathematical Programming **151**(1), 3–34 (2015) 8
38. Xue, Z., Mao, W., Zheng, L.: Learning to simulate complex scenes. arXiv preprint arXiv:2006.14611 (2020) 4
39. Yang, L., Luo, P., Change Loy, C., Tang, X.: A large-scale car dataset for fine-grained categorization and verification. In: CVPR (2015) 5
40. Yao, Y., Plested, J., Gedeon, T.: Information-preserving feature filter for short-term eeg signals. Neurocomputing (2020) 2
41. Zheng, L., Bie, Z., Sun, Y., Wang, J., Su, C., Wang, S., Tian, Q.: Mars: A video benchmark for large-scale person re-identification. In: ECCV (2016) 3, 11, 12

42. Zheng, Z., Ruan, T., Wei, Y., Yang, Y.: Vehiclenet: Learning robust feature representation for vehicle re-identification. In: CVPR Workshops (2019) 11
43. Zheng, Z., Zheng, L., Yang, Y.: Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In: CVPR. pp. 3754–3762 (2017) 4
44. Zhong, Z., Zheng, L., Zheng, Z., Li, S., Yang, Y.: Camera style adaptation for person re-identification. In: CVPR (2018) 1
45. Zhong, Z., Zheng, L., Zheng, Z., Li, S., Yang, Y.: Camstyle: A novel data augmentation method for person re-identification. IEEE Transactions on Image Processing **28**(3), 1176–1190 (2018) 3
46. Zhou, Y., Shao, L.: Aware attentive multi-view inference for vehicle re-identification. In: CVPR (2018) 3