TUM School of Computation, Information and Technology
Technische Universität München

Synthetic Data for Perception in Autonomous Driving.

Artem Savkin

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz:

Prof. Dr. Christian Mendl

Prüfer der Dissertation:

1. Priv.-Doz. Dr.-Ing. habil. Federico Tombari
2. Prof. Dr. Vasileios Belagiannis,
   Friedrich-Alexander-Universität Erlangen-Nürnberg
3. Prof. Dr.-Ing. Darius Burschka

Die Dissertation wurde am 17.02.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 18.09.2023 angenommen.

Abstract

The autonomous driving field poses an immense demand for reliable training data for the neural networks of its perception systems. A training dataset is required to be exact, comprehensive, and diverse. Acquisition of data that fulfills these requirements can be a challenging task for a multitude of reasons. For instance, manual annotation of samples can be a cost-inefficient process. Another example can be capturing near-accident scenarios, which can be ethically compromised due to endangering road users.

Synthetic data offers a solution to the challenges of manual data acquisition. Nevertheless, the domain gap between simulated and real images limits their applicability in the real world. The problem of domain gap is addressed in many recent works through the framework known as generative adversarial network (GAN) employed in sim-to-real image transfer. These methods take rendered images as input and map them into a closer representation of the real data domain through realistic style transfer. Often this procedure results in a realistic appearance of translated images but corrupted macro structure of its content. Image perturbations make them inconsistent with corresponding annotations and thus unsafe for critical recognition tasks in autonomous driving.

This work explores data synthesis techniques that can alleviate the gap between the simulated and real domains. It first proposes a data augmentation pipeline for geometrically correct, collision-free placement of virtual models of pedestrians into real traffic scenes. This procedure improves the quality of blending the computer-aided design (CAD) models by learning appearance from reference data via generative adversarial network (GAN). To achieve that, the method proposed a class-specific discriminator which counter-acts the content corruption exemplified by vanishing out-of-distribution 3D objects.

The problem of content hallucination in a sim-to-real setting caused by GANs is further analyzed in work through the lens of global class statistics. It identifies that adversarial transfer is affected by dataset bias, which leads to inconsistent generation results. As a solution, the method proposes the combination of adversarial learning and density pre-matching through an importance estimation procedure so that the model can learn from the most informative samples. The proposed combination effectively mitigated the semantic inconsistency problem; it required, though, privileged knowledge about target data statistics which is not available in the preferred setting.

Further, this works investigates semantically consistent sim-to-real image transfer where reference data annotations are not available. To address this problem, a content disentanglement strategy is utilized. It aims to learn domain-agnostic content features separately from domain-specific appearance features. Separation of both vectors constrains the appearance component, so cross-domain image transfer merely changes the sample on low-level without modifying its macro-structure. On the other hand, fixed style disentangling reduces the generative capacity of the model by learning of the entire content vectors manifold.

The final part addresses the compromise between the degree of generative perturbations and semantic consistency. It suggests a procedural scene generation accompanied by appearance learning as an alternative to arduous high-fidelity virtual environments. The proposed pipeline produces traffic scenes without visual attributes, whereas the appearance learning part lets them attain a realistic look. To achieve that, the approach employs synthetic scene graphs extended to an unsupervised setting, allowing users to manipulate the scene according to a required scenario and encode spatial and global parameters.

Kurzfassung

Im Bereich des autonomen Fahrens besteht ein immenser Bedarf an zuverlässigen Trainingsdaten für die Wahrnehmungssysteme. Ein Trainingsdatensatz muss genau, umfassend und vielfältig sein. Die Beschaffung von Daten, die diese Anforderungen erfüllen, kann aus einer Vielzahl von Gründen eine Herausforderung darstellen. So kann beispielsweise die manuelle Beschriftung von Proben ein kostenineffizienter Prozess sein. Ein weiteres Beispiel ist die Erfassung von unfallnahen Szenarien, die aufgrund der Gefährdung von Verkehrsteilnehmern ethisch bedenklich sein kann.

Synthetische Daten bieten eine Lösung für die Herausforderungen der manuellen Datenerfassung. Allerdings schränkt die Domänenlücke zwischen simulierten und realen Bildern ihre Anwendbarkeit in der realen Welt ein. Das Problem der Domänenlücke wird in vielen neueren Methoden durch den Rahmen, der als generative adversarial network (GAN) bekannt ist und bei der Übertragung von simulierten zu realen Bildern verwendet wird, angegangen. Diese Methoden nehmen gerenderte Bilder als Eingabe und bilden sie durch eine realistische Stilübertragung in eine genauere Darstellung des realen Datenbereichs ab. Oft führt dieses Verfahren zu einem realistischen Aussehen der übersetzten Bilder, aber zu einer verfälschten Makrostruktur ihres Inhalts. Die Bildmanipulationen führen dazu, dass die Bilder nicht mit den entsprechenden Anmerkungen übereinstimmen und somit für kritische Erkennungsaufgaben beim Autonomen Fahren unbrauchbar sind.

In dieser Arbeit werden Datensynthesemethoden untersucht, die die Diskrepanz zwischen simulierter und realer Domäne abmildern können. Zunächst wird eine Methode zur Datenagmentierung für die geometrisch korrekte, kollisionsfreie Platzierung von virtuellen Fußgängermodellen in realen Szenen vorgeschlagen. Dieses Verfahren verbessert die Qualität der Zusammenstellung der CAD-Modelle durch Lernen ihres Aussehens aus Referenzdaten über GAN. Um dies zu erreichen, schlägt die Methode einen klassenspezifischen Diskriminator vor, der der Inhaltsverderbung entgegenwirkt, die durch das Verschwinden von out-of-distribution 3D-Objekten entsteht.

Das Problem der Inhaltshalluzination in einer Sim-zu-Real-Umgebung wird in der Arbeit durch die Linse der globalen Klassenstatistik weiter analysiert und es wird festgestellt, dass Transfermethoden durch Datensatz-Bias beeinträchtigt werden, was zu inkonsistenten Generierungsergebnissen führt. Als Lösung schlägt die Methode die Kombination von adversarialem Lernen und Density Matching durch ein Wichtigkeitsschätzungsverfahren vor, so dass das Modell aus den informativsten Proben lernen kann. Die vorgeschlagene Kombination entschärft wirksam das Problem der semantischen Inkonsistenz; sie erfordert privilegiertes Wissen über die Statistik der Zieldaten, das in der bevorzugten Umgebung nicht verfügbar ist.

Darüber hinaus wird in dieser Arbeit die semantisch konsistente Übertragung von Simulations- zu Realbildern untersucht, wenn keine realen Annotationen

verfügbar sind. Um dieses Problem zu lösen, wird eine Methode zur Disentanglement von Inhalten vorgeschlagen. Sie zielt darauf ab, domänenagnostische Inhaltsmerkmale getrennt von domänenspezifischen Erscheinungsmerkmalen zu lernen. Durch die Trennung beider Aspekte wird die Erscheinungsbildkomponente eingeschränkt, so dass bei der domänenübergreifenden Bildübertragung lediglich die Low-level-Aspekte des Beispiels geändert wird, ohne seine Makrostruktur zu modifizieren. Darüber hinaus reduziert die feste Stil-Disentanglement die generative Kapazität des Übersetzungsmodells, indem sie das Lernen des gesamten Inhaltsvektor-Mannigfaltigkeit ermöglicht.

Der letzte Teil befasst sich mit dem Kompromiss zwischen dem Ausmaß der generativen Änderungen und der semantischen Konsistenz. Er schlägt eine prozedurale Szenengenerierung vor, die mit dem Lernen von Erscheinungsbildern einhergeht und eine Alternative zu mühsamen High-Fidelity-3D-Umgebungen darstellt. Die Pipeline erzeugt Verkehrsszenen ohne visuelle Attribute, während der Teil des Erscheinungsbild-Lernens ihnen ein realistisches Aussehen verleiht. Um dies zu erreichen, verwendet die Methode synthetische Szenegraphen, die auf eine nicht überwachte Art und Weise erweitert werden und es dem Benutzer ermöglichen, die Szene entsprechend einem gewünschten Szenario zu manipulieren und räumliche und globale Parameter zu kodieren.

Acknowledgements

This thesis is the results of my research at BMW Group and Technical University of Munich. I extend my appreciation to both organizations for granting me the opportunity to delve into this captivating subject.

I would like to convey my deepest gratitude to Federico Tombari and say *grazie* and *spasibo*. His unwavering guidance, invaluable knowledge, and being always there have been indispensable. I extend my thanks to Prof. Dr. Nassir Navab for fostering an exceptional research group, of which I am honored to be a part. Special appreciation is also due to Mario Tokarz and Cornelia Denk for providing me with this initial opportunity.

Throughout my research journey, I had the privilege of collaborating with exceptionally talented fellow students and colleagues. While it's impossible to name everyone, I want to express my sincere gratitude to Niko, Thomas B., Luca, Branka, Julian, Alexander F., Lennart, Egon, Chen, Jakob, Oliver, Manoj, Stefano, Yida, Mahdi, Shun-Cheng, Markus, Alexander L., Monika, Thomas L., Kevin, Uzair, Mert, Rachid, Balaji, Mohab, Andrew, and Sherif. I am equally thankful to Sebastian Wirkert, Johannes Niedermayer, Loren Schwarz, Pilar Garcia, Fridolin Bauer, Tom Stone, Elvira Shishenina, Thomas Stauner, and Hans-Jörg Vögel. Your feedback, enriching discussions, knowledge sharing, casual conversations, and, of course, the shared pizzas have been invaluable.

A special mention goes to my friends Artem, Egor, Max, and Misha for their support in coping with challenging moments.

I am very thankful to my mom, Natalia, and my dad, Alexander, for their love, continuous support, and encouragement. No words can adequately convey the depth of this gratitude.

I reserve special thanks for my wife, Olya, who has consistently been a bright and cheerful light in my life, making even the most challenging moments easier.

Contents

List of Figures

List of Tables

Acronyms

iid	independent and identically distributed.
AdaIN	Adaptive Instance Normalization.
AMT	Amazon Mechanical Turk.
AP	average precision.
AR	autoregressive.
BDD	*Berkeley DeepDrive*.
BRDF	bidirectional reflectance distribution function.
CAD	computer-aided design.
CNN	convolutional neural network.
DLL	dynamic-link library.
DRN	*Dilated Residual Network*.
EBM	energy-based model.
ELBO	evidence lower bound.
FCN	*Fully Convolutional Network*.
FID	Fréchet inception distance.
GAN	generative adversarial network.
GPU	graphics processing unit.
GSN	generative stochastic network.
HOG	histogram of oriented gradients.
IN	instance normalization.
IoU	intersection over union.
IS	Inception Score.
JSD	Jensen–Shannon divergence.
KLIEP	Kullback–Leibler Importance Estimation Procedure.

LID	local intrinsic dimensionality.
LPIPS	learned perceptual image patch similarity.
ML	machine learning.
MLP	multilayer perceptron.
MSE	mean squared error.
MTS	Mesh-Texture-Shader.
NeRF	neural radiance field.
PfD	*Playing for Data*.
PPL	perceptual path length.
RBF	radial basis function.
RCNN	Region-based Convolutional Neural Networks.
SIR	Sampling Importance Resampling.
SSIM	structural similarity index measure.
SVM	support vector machines.
TL	transfer learning.
VAE	variational autoencoder.
VQA	visual question-answering.
VRU	vulnerable road users.

Part I

1 Introduction

1.1 Autonomous Driving

History. Autonomous driving is a field of applied research and technology about the systems capable of navigating the environment without human control. Such systems are expected to significantly impact society in various areas, from road safety [16] to environmental pollution [17] to urban space planning [18]. Early research in autonomous driving began in the 1950s, but the first autonomous vehicle, ALVINN [1], did not appear until the 1980s. ALVINN pioneered employing neural networks for road detection and using synthesized images for their training. Other notable vehicles developed in the following years include VaMP [19] and ARGO [20]. Several influential milestones include ProLab2 of Prometheus Program [21], BART [22], and REMI [23], which also employed neural networks in their designs. In these works, researchers analyzed which types of networks were best suited for an autonomous vehicle. For instance, Kornhauser [24] used a computer visual simulation system known as *Road Machine*. Later Rosenblum et al. [25] used *radial basis function networks* as an alternative to ALVINN's multilayer perceptron (MLP), and Yu et al. [26] proposed to use reinforcement learning to eliminate supervision. In the 2000s, a series of DARPA Grand Challenges further advanced research in the field of autonomous driving, where some participants, including the winning team Stanley [27], relied on the learning algorithms as a part of the perception stack. Learning helped to deal with the variance of possible driving scenarios, and nowadays, it is almost impossible to imagine an autonomous vehicle without neural networks.

Data. The success of deep neural networks [28] in image classification led to the subsequent application of this technology to a broader range of computer vision tasks. The transition of deep learning from a mere research topic to an application in a broad spectrum of real-world tasks made the accessibility of comprehensive and reliable training data exceptionally crucial. The training dataset must be exact – maintain error-free annotations; comprehensive – provide abundant training samples; diverse – comprise high variance of scenarios. However, large amounts of annotated training data required for the effective use of neural networks may not be cost-efficient to obtain with manual labelling. Thus more economical solutions are required for productive use.

The transition from the research to the application imposes an obligation for perception systems of autonomous vehicles to *understand* the surrounding environments and to deal with a wide variety of traffic situations in the real world. These environments and traffic scenarios include myriad variations caused by

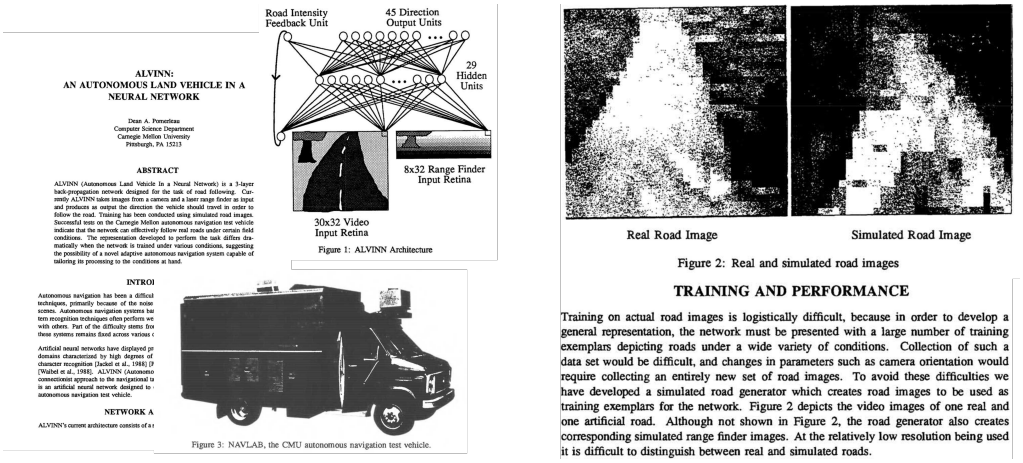


Figure 1.1: Autonomous vehicle ALVINN used a neural network trained on synthesized data for road perception [1].

factors such as lighting, weather conditions, locations, sensor setups, novel traffic users, near-accident and rare events. The ability to perform well in these unseen traffic situations determines if a vehicle can navigate fully autonomously and represents one of the main challenges of developing autonomous systems. To meet this criterion, the perception algorithms of an autonomous vehicle must generalize well in various traffic situations that it may confront.

While neural networks have significantly advanced the recognition algorithms of autonomous vehicles, their generalization capacity is highly dependent on the availability of a substantial collection of reliable and well-annotated traffic scenarios. This property significantly influences the development and quality assurance of perception systems. It is necessary to include specific scenarios in the training dataset beforehand to ensure that the system will handle them correctly. Engineers continually adjust their deep models by acquiring new traffic scenarios during the field drives and retraining, a process that requires annotating every newly captured scene.

A collection of sufficient training data that covers various traffic scenarios in real-world environments for autonomous driving is often not feasible, which makes the incremental strategy of capturing new scenarios during field drives impractical. Moreover, such data acquisition typically implies manual annotation, which is a laborious and time-consuming process [29], especially in the tasks such as semantic segmentation, as it requires dense per-pixel labelling. Although the data sample annotation routine seems relatively straightforward, real-world data at scale differs. Additionally, annotating long-tail scenarios that occur infrequently and are challenging to capture can be non-trivial. Moreover, capturing near-accident events can be ethically constrained, as it may put vulnerable road users in unsafe traffic situations. As a result, productive use of deep neural networks requires efficient acquisition and annotation of training samples.

1.2 Synthetic Data

Synthetic data is a potential solution for the challenges of data acquisition and annotation described in Section 1.1. The straightforward approach involves rendering engines for generating data that computer vision models could use. It is a cost-effective method that can reveal potentially unbounded variance of training data at a reduced cost, decrease manual labelling effort, and cover potential rare or near-accident scenarios. Many researchers have utilized images rendered with computer graphics in their works on vision tasks [30] [31], [32], [33]. Similarly, autonomous driving and deep learning researchers utilized data rendering as an alternative to data acquisition and annotation. Indeed, simulation has a long history of use in autonomous driving [1], Figure 1.1 demonstrates one of the first attempts to utilize synthesized road images to train a recognition neural network of an autonomous vehicle in 1988. In recent years, with the renaissance of deep learning, the research community has intensified efforts in this direction, and many synthetic datasets [3, 5, 6], and simulation systems [34, 35] emerged.

Although simulation can generate a large amount of training data, it still exhibits limited applicability in a real-world environment. Recent research results [3, 36, 37] indicate that training deep models on synthetic data leads to inferior performance on real data. Multiple works have demonstrated that models trained merely on synthetic data show a drastic performance drop on real-world data compared to the models trained on real data. Specifically, Richter et al. [4] reported a decrease of almost 20% mean intersection over union (IoU) for a semantic segmentation model [38], which has been trained on CamVid train set [39], and synthetic dataset [4] but evaluated on CamVid validation set.

In machine learning, training and validation data are assumed to be independent and identically distributed (*iid*), which may nevertheless be not an accurate assumption in a sim-to-real setting. The underlying rendered domain can cause specific bias in the data even though the rendering engine approximates light transport physically correctly and generates realistic images. The reasons mainly lie in rendering specifics, including the correctness of light transport simulation or low-level realism, and discrepancy in the overall structure of the virtual environment, such as the building shape or the occurrence of different objects. While computer graphics images are often seen as a cost-effective way to obtain training samples, algorithms that are trained using this data do not tend to achieve comparable results on real-world data due to the *domain gap* between artificial and acquired samples and that contradicts the *iid* assumption common to most models.

The direct way to solve the *iid* contradiction and address the domain discrepancy is to elevate the quality of rendered images and make the generated and real data distributions similar. A more precise approximation of the rendering process, which minimizes artefacts resulting from image generation, can help to achieve that goal. However, it is crucial to consider the variation in the macrostructure of rendered images, which can vary due to changes in the environment,

or the sensor suite of an autonomous system [36]. Thus, despite being a valuable research tool, computer graphics can still exhibit a significant domain gap compared to target data. The phenomenon, which is believed to be the main cause of degraded performance mentioned previously [40], is known as *covariate shift* [41]. It describes a setting when marginal distributions in both domains diverge. The phenomenon highlights that the assumption of *iid* does not hold for a synthetic-and-real setting where domains differ in content and appearance. Typically, domain adaptation methods such as sim-to-real image transfer address the mismatch. They improve the performance of vision models compared to models trained exclusively synthetically.

1.3 Domain Adaptation

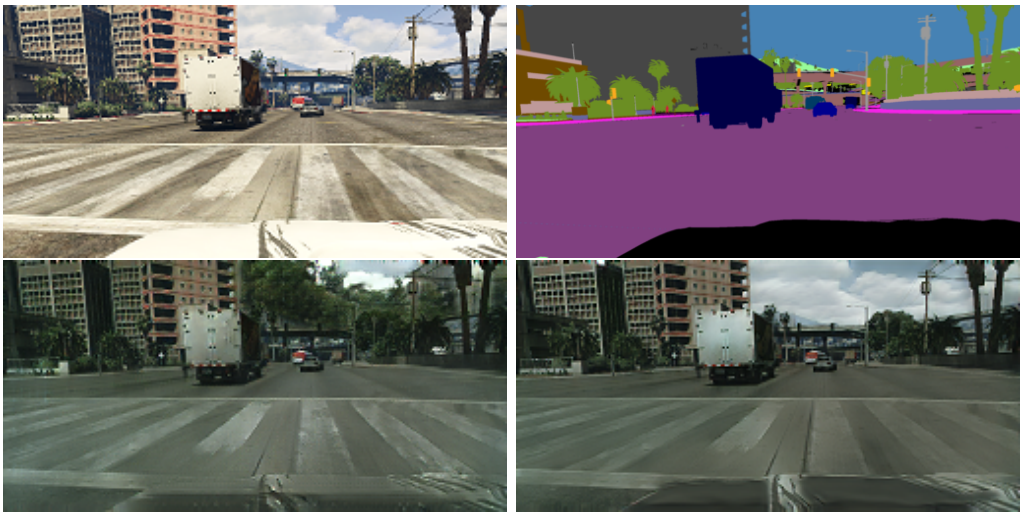


Figure 1.2: Example of class mismatch introduced by domain adaptation in class-imbalanced setup. Original PfD image (topmost), original image translated to Cityscapes by CycleGAN (second from the top), original image translated to Cityscapes with proposed approach (bottommost). © 2019 IEEE

Multiple works proposed improving the realism in synthetic images using domain adaptation techniques in order to address the domain gap problem discussed in Section 1.2. These works include, for example, DTN [42], *FCNs in the Wild* [43], CycleGAN [10] or DualGAN [13]. Typical adaptation mechanism employs generative adversarial network (GAN) [8], which aims to minimize the gap between two domains by translating the synthetic data into a representation that is closer to the real data [44, 45]. A generative adversarial network (GAN) attempts to fit distribution over target observations x from target data D_t by training two neural networks: a generator and a discriminator, which is a binary classifier. The generator performs permutations of source input samples $x_i^s \in D_s$ from the

source dataset so that they are indistinguishable for the discriminator network from target samples $x_j^t \in D_t$. Thus, generative adversarial network indirectly imposes target distribution over the generated distribution [8]. Convergence of the networks' training in the Nash equilibrium leads to minimizing the distance between both distributions but cannot ensure the immutability of the samples. As a result, generative adversarial network deliver visually realistic data but causes semantic mismatch and embeds visual artefacts into synthesized images. Adversarial training is a powerful tool for adaptation tasks, but it is also affected by covariate shift, as the discriminator inflicts the regularities learned from the target data on the generated samples.

Figure 1.2 demonstrates a failure of a GAN model to correctly translate rendered images into the real domain. Specifically, the generator introduces semantically mismatching artefacts into refined images. For these two datasets, the network generates an inconsistency between the vegetation and sky classes in an attempt to restore the target data distribution [46]. The example with augmented meshes of pedestrians in Figure 4.4 visualizes reinstating the target distribution more clearly. In this case, the discriminator recovers the original distribution by removing the out-of-distribution objects. This behaviour can be undesirable for several reasons. Two main issues can impact the realism of images generated using a generative adversarial network. First, the GAN's capacity or perceptive field may not be sufficient to generate a realistic class instance, leading to lower-quality images. Second, to use the generated images for a downstream task, it is necessary to have a ground truth in accordance with the image's semantic structure. The artefacts shown in Figure 1.2 can make it challenging to use the generated samples for training computer vision models.

More samples from various datasets where a generative adversarial network introduces semantic inconsistencies during the translation process are demonstrated in Figure 5.5. For example, image transfer imposes the vegetation patches in sky regions in *Playing for Data* (PfD) [4] and Cityscapes [29] setting. It also removes road users from the traffic scene in SYNTHIA and Cityscapes. The domain adaptation process, where a network introduces semantically mismatching artefacts to restore the reference data distribution, can be especially critical in traffic scene understanding as it results in unreliable training data. Indeed, inconsistent permutations of the image significantly diminish the applicability of data synthesis for computer vision tasks.

Several works addressed the problem of inconsistent transfer in order to guarantee invariance of the image macro-structure during domain adaptation. The main idea of such methods is to leverage available knowledge about the scene by using semantic maps [47, 48, 49]. Constraining the adaptation process on semantic maps is effective but depends on manual labels. Other works introduce constraints such as self-regularization [9], semantic consistency [10], regularization by enforcing bijectivity [47], modeling a shared latent space [50], [51], or semantic aware discriminator [52].

Many works proposed to generate the images in a consistent way by doing that from scratch via conditioning on alternative content representations. For example, the approach Pix2Pix [53] introduced a supervised generator and discriminator conditioning on semantic maps. Another strategy proposed by Qi et al. [54] involves using predefined patches from the database to combine an image canvas. Wang et al. [55] analyzed the reasons for lacking high-quality textures and details in the results produced by Conditional GANs and proposed a multi-scale GAN architecture to mitigate them. Other works, such as those presented in [51] and [15], combined cycle loss and adversarial loss to disentangle appearance and content by learning the corresponding latent space.

1.4 Objectives

Previous sections highlighted the challenges of employing synthetic data for training deep perception models. This thesis aims to explore the methodology of data synthesis in autonomous driving and to advance the field. Given the initial condition where the discrepancy between synthetic and real data distributions restricts the application of the generated data in the real world, this thesis pursues to find synthesis techniques to alleviate these limitations. They intend to produce data that entirely suffice for training perception models employed in real-world scenarios.

First, the thesis examines the applicability of synthesized data in a simplified setting. It proposes an augmentation technique for placing synthetic objects of interest in a real-world traffic scene and visually adjusting them to the surrounding environment. The augmentation only focuses on the scenarios with vulnerable road users so that any real scene can be manipulated to configure a scenario of interest. Additionally, it explores the enforcing of visual characteristics from real data onto in-painted objects in lieu of translating complete synthetic scenes to the real domain. Furthermore, it studies the ability of the translation to overcome semantic inconsistency inflicted during image transfer and resulted in the disappearance of augmented objects.

Next, the thesis analyzes the weaknesses of the standard image transfer approaches, which minimize the discrepancy between the synthetic and real data. In particular, it investigates the origin of semantic inconsistency occurring during the transfer between two domains. A method proposed in this work explores direct manipulation of the datasets for preventing class confusion and later integration of a density estimation technique as a solution for semantically consistent sim-to-real domain adaptation.

Further, the thesis explores a way to perform sim-to-real domain adaptation in an unsupervised fashion without privileged information about global statistics. It studies a technique of disentangling the content and style representation for implicit sifting out the apparent discrepancies between both domains. By disentangling both features, the model aims to learn the domain and content-

agnostic image transfer model, which is intended to apply visual characteristics of any particular domain to individual content vectors from the underlying content manifold.

Finally, the thesis rounds up by merging initial intuition with the obtained findings. It studies the potential of newly introduced data synthesis concepts with procedural content generations and learned appearance. Procedural content generation is dedicated to the controlled and explicit content definition, and appearance learning synthesises realistic imagery based on it. The thesis aims to explore data synthesis mechanisms with accordance to several principles:

- Scalable data generation with high variance of content.
- Similarity of the generated data to reference data.
- Consistency of the generated data with ground truth.
- Straightforward manipulation of the synthesized data

1.5 Contribution

Consequent chapters propose several methods to fulfil the objectives outlined in Section 1.4 whose contribution can be summarized as follows:

Augmentation. Chapter 4 focuses on pedestrian detection and proposes an augmentation pipeline that enhances real urban traffic datasets with virtual pedestrians in different scenarios. This pipeline performs geometrically consistent placement of the pedestrian CAD models into real traffic scenes. Since mere augmentation does not consider the appearance of the image of a traffic scene, the pipeline requires a mechanism for realistic pedestrian depiction. This work introduces a domain adaptation model which learns the realistic appearance and makes the pedestrian CAD models visually integrated into the scene. The adaptation network employs multiple discriminators similar to those used in [56] for various image resolutions. However, the use of a masking approach makes it resistant to differences in distribution between real and synthetic data. Mask discriminators enable the production of consistent synthetic imagery with realistic illumination and the appearance of the original dataset. Figure 4.1 shows augmented and translated images, along with semantic and instance segmentation labels. The corresponding publication is:

- Artem Savkin, Thomas Lapotre, Kevin Strauss, Uzair Akbar, Federico Tombari, "Adversarial Appearance Learning in Augmented Cityscapes for Pedestrian Recognition in Autonomous Driving", IEEE Int. Conf. on Robotics and Automation (ICRA), 2020

Density Matching. Chapter 5 introduces an approach which tackles the semantic inconsistency issue in sim-to-real domain transfer as a class imbalance

problem. In a class imbalanced setting, it is assumed that marginal distributions for source and target samples are different: $P_s(X) \neq P_t(X)$ [57, 41]. If labels of the target domain are known, then the resampling technique can be employed to tackle the class imbalance [58, 59, 60, 61]. These approaches typically manipulate the training set of data and include several directions over-sampling, under-sampling, and hybrid approach [58].

This work proposes a sim-to-real translation mechanism based on matching target distribution via Sampling Importance Resampling (SIR) [62] strategy in an ensemble with effective cycle-consistent loss [10]. The proposed dataset resampling mitigates the semantic perturbations in translated images, which leads to improved performance of the downstream semantic segmentation models. Chapter 5, further, proposes the use of density ratio-based distribution pre-matching in conjunction with the cyclic-consistency loss for direct sim-to-real adaptation without dataset manipulation. Contrary to other works utilizing importance weights [63] or kernel density [64] to improve GAN training the proposed method employs a technique named KLIEP [65] to pre-match distribution densities alongside adversarial and cycle consistency loss. Density ratio estimation maintains the semantic consistency of generated images, and thus betters their visual characteristics. Evaluation of semantic segmentation showed improvement in dedicated computer vision models. Moreover, the proposed method positively impacts the stability of underlying architecture, as it does not introduce additional constraints. The corresponding publications are:

- Artem Savkin, Federico Tombari, "KLIEP-based Density Ratio Estimation for Semantically Consistent Synthetic to Real Images Adaptation in Urban Traffic Scenes," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2020
- Artem Savkin, Monika Kasperek, Federico Tombari, "Sampling/Importance Resampling for Semantically Consistent Synthetic to Real Image Domain Adaptation in Urban Traffic Scenes," IEEE Intelligent Vehicles Symposium (IV), 2019

Disentanglement. Chapter 6 introduces a novel, end-to-end learning architecture that aims at semantically consistent unsupervised sim-to-real domain adaptation. One limiting requirement is that the architecture refrains from privileged information, such as semantic segmentation maps or pre-trained networks, in the form of perception loss. Contrary to the methods introduced in Chapter 5, this approach does not rely on global statistics about the source and target domains. The main principle is that the generator learns features separately by using a single content encoder and decoder. The decoder, in turn, operates on learned content features with fixed style codes. Furthermore, the proposed network is lightweight since it includes only one encoder, a decoder for both domains. The generator network is constrained through repeating intra-domain and cross-domain reconstruction. The corresponding publication is:

- Mert Keser, Artem Savkin, Federico Tombari, "Content Disentanglement for Semantically Consistent Synthetic-to-Real Domain Adaptation", IEEE Int. Conf. on Intelligent Robots and Systems (IROS), 2021

Scenegraph. Chapter 7 describes a method for scalable, realistic, and consistent image synthesis with traffic scene manipulation capabilities. The core idea is to entirely avoid the usage of the rendering process. The key idea comes from the fact that rendered images carry a bias originating from rendering. These images require then adaptation to the real domain in order to abolish the bias. The proposed idea consists in replacing the rendered images with generic representations such as scene graphs and directly creating images from those representations. Scene graphs encode the scene's objects as nodes and relations between them as edges [66]. In addition, they can also integrate specific characteristics of the objects or entire scene as dedicated parameters [67]. A procedural content generation pipeline randomly produces the synthetic scene graphs from a virtual scene in a domain-agnostic way. In addition, the graphs are reasonably straightforward to automatically derive from the virtual scene and manipulate manually when required, this contributes to data synthesis of possibly arbitrary extent and variability, thus increasing scalability.

Furthermore, this work introduces an end-to-end learnable model that extends synthetic graph processing in an unsupervised fashion to facilitate realistic image generation. The proposed model improves realism by directly synthesizing the images from scene descriptions. Synthetic scene graphs obtained from a simulated environment include typical traffic objects such as car, person, and others, and relations like *left to*, *right to*. Procedural traffic scene simulation lets extend this setting with other classes required for automotive data road, sidewalk, building, and vegetation. Additionally, the virtual environment yields spatial information, which results in spatial attributes and object relations. Finally, the virtual environment provides accurate pixel-dense annotations at no additional cost.

- Artem Savkin, Rachid Ellouze, Nassir Navab, Federico Tombari, "Unsupervised Traffic Scene Generation with Synthetic 3D Scene Graphs", IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2021

Further work explores domain adaptation methods for other data modalities common in autonomous driving - Lidar point cloud, but will not be discussed in this dissertation.

- Artem Savkin, Yida Wang, Sebastian Wirkert, Nassir Navab, Federico Tombari, "Lidar Upsampling with Sliced Wasserstein Distance", IEEE Robotics and Automation Letters (RAL), 2022

2 Fundamentals

2.1 The Rendering Equation

Rendering, also known as image synthesis, is the problem of generating images from virtual scenes. Global illumination algorithms aim to solve the synthesis problem by simulating the physical travel of light from its source to the virtual sensor [2]. However, computer graphics typically rely on the simplest light propagation model, called geometric optics, which, in essence, can simulate only limited types of phenomena such as emission, reflection, and transmission. In accordance with the geometric model, light travels on straight lines at infinite speed. As a result, a straightforward rendering approach involves simulating the emission of light ray from the light sources, their interaction with the environment, and tracking the light that reaches the virtual camera sensor. However, this naive rendering technique is very computationally intensive, as only a fraction of the emitted rays reaches the sensor, requiring the simulation of numerous beams to capture enough of them for image creation on the sensor side.

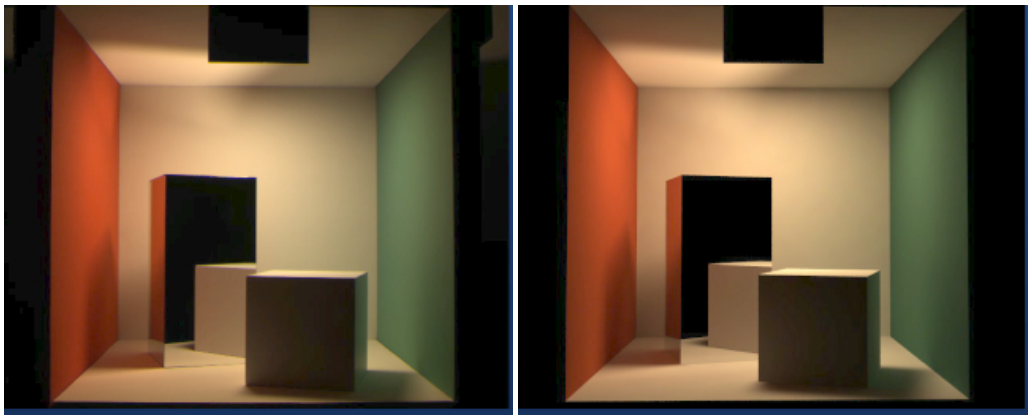


Figure 2.1: Cornell box experiment with the photographed (left) and rendered (right) scene [2].

The global illumination algorithm which neglects the rays that do not reach the sensor is known as ray tracing [68]. This algorithm, proposed by Whitted [68], relies on reciprocity - a property that allows reversing the light trace from the sensor to the source without violating physical simulation. It helps capturing essential effects of direct illumination such as shadows, reflections, and refraction [69]. Simulating the effects of indirect illumination, such as inter-reflection,

commonly known as color bleeding or caustics, requires a notion of radiance. Radiance describes the amount of power transmitted or reflected by a surface that an optical system viewing the surface from a certain angle will receive.

$$L(x,\omega) = \frac{\partial\Phi(x,\omega)}{\partial\omega A(x)} \tag{2.1}$$

Equation 2.1 expresses the amount of light that reaches a hypothetical area A perpendicular to a direction of its origin ω. Φ represents the radiant power, or flux, which is the amount of energy passing through a surface. The amount of incident power on a unit of surface is known as irradiance and can be expressed as:

$$E(x) = \frac{\partial\Phi(x)}{\partial A(x)} \tag{2.2}$$

The relation between irradiance and reflected radiance is described by the bidirectional reflectance distribution function (BRDF) [70] as:

$$f_r(x,\hat{\omega},\omega) = \frac{\partial L(x,\omega)}{\partial E(x,\hat{\omega})} \tag{2.3}$$

The bidirectional reflectance distribution function describes the interaction between the light and a surface. Essentially, it specifies the perceived brightness of the surface when viewed from the particular direction ω and illuminated from the other direction $\hat{\omega}$.

The equilibrium radiance leaving a point x in a particular direction ω can therefore be expressed as the sum of emitted radiance and reflected radiance as:

$$L(x,\omega) = L_e(x,\omega) + \int_\omega f_r(x,\hat{\omega},\omega)L(x,-\hat{\omega})(\hat{\omega}\cdot\mathbf{n})d\hat{\omega} \tag{2.4}$$

Equation 2.4, also known as the rendering equation, was introduced independently by [71] and [72]. Various mechanisms, including Monte Carlo methods, path tracing, or photon mapping, seek to solve this equation to achieve realistic rendering. Photo-realistic or unbiased rendering involves minimizing statistical bias in the approximation of radiance, thus accurately reproducing optical effects such as those described previously.

2.2 Synthetic Datasets

Advances in realistic rendering achieved by computer graphics engines offered access to a potentially unbounded training data source for deep learning-based perception methods. Furthermore, image rendering extended the variation of the covered traffic scenarios and extensively increased the number of produced training samples. Recently several large-scale automotive datasets emerged in the field and strengthened research in the direction of synthetic data for perception.

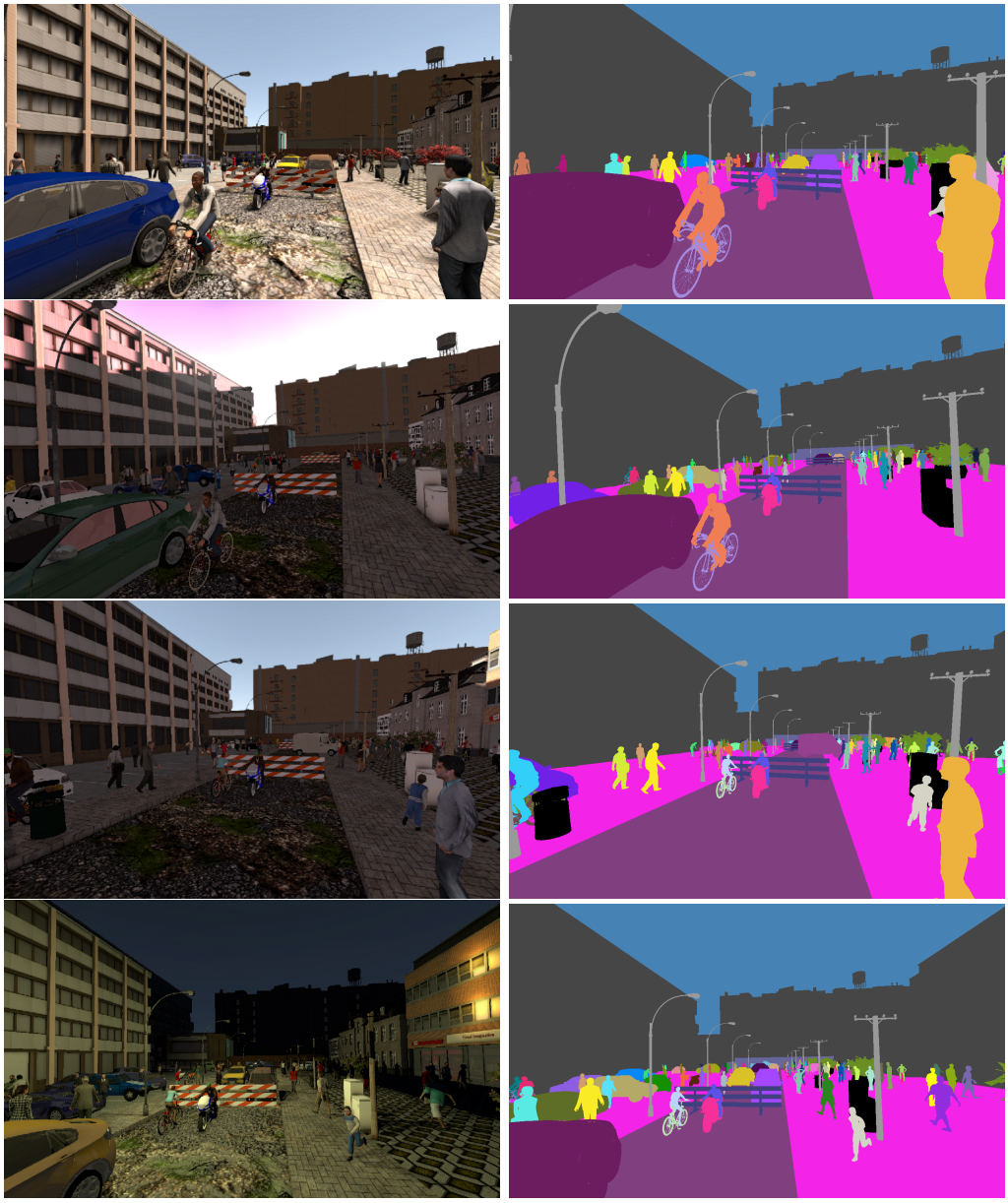


Figure 2.2: Examples of SYNTHIA dataset and corresponding semantic labels [3].

2.2.1 SYNTHIA

Ros et al. [3] introduced a method for realistically generated images with pixel-dense annotations using 3D scenes and demonstrated the SYNTHIA dataset. The primary application of SYNTHIA is training the data-driven algorithms for semantic segmentation for autonomous driving scenarios, where it offered multiple variations of individual scenes and accurate annotations. The SYNTHIA dataset

consists of photo-realistic frames rendered from various viewpoints and dense semantic annotations covering 13 classes such as sky, building, road, sidewalk, fence, vegetation, lane-marking, pole, car, traffic signs, pedestrians, cyclists, and miscellaneous. In addition to semantic maps, it provides corresponding dense depth maps.

The authors obtained the traffic scenes in the SYNTHIA dataset using a virtual city constructed from dedicated basic blocks that contain elements typically found in the traffic environment, like streets, sidewalks, cars, or pedestrians. The virtual environment includes various seasons, weather conditions, and dynamic illumination, which enables the simulation of diverse daylight conditions for the same scene. The simulated multi-cameras with a baseline of 0.8 meters capture the frames of the virtual environment. Each multi-camera system consists of four monocular cameras with a common centre but different orientations spaced by 90 degrees apart. All cameras have a field of view of 100 degrees. Figure 2.2 shows an example frame and accompanying semantic map.

The SYNTHIA dataset consists of the frames of a resolution 960×720 pixels divided into two subsets: SYNTHIA-Rand and SYNTHIA-Seqs, which include $13,400$ and $200,000$ examples, respectively. The SYNTHIA-Rand subset was obtained by randomly moving the camera within a scene at a height between 1.5 and 2 meters, ensuring that individual camera positions were at least 10 meters apart to achieve high variability. The SYNTHIA-Seqs subset simulates four video sequences, each containing $50,000$ frames.

2.2.2 VIPER

The VIPER benchmark suite introduced by Richter et al. [5] is an extension to the previous work *Playing for Data* (PfD) [4] of the authors, who explore the use of commercial video games as a source of large-scale, pixel-accurate ground truth data for semantic segmentation task. The previous work, PfD, exploits the realism of the GTA video game, which is characterized by its high-fidelity textures and materials, and realistic light transport simulation, to generate superior renders. On the other hand, high-level realism on a large scale is primarily defined by factors such as the content, motion, and interactions between the objects or objects and the environment within the game.

The authors of *Playing for Data* were able to intercept the rendering commands and reproduce the game frames by introducing a wrapper between the game engine and the operating system. The key idea lies in intercepting the calls between the game and the graphics hardware typically accessed through encapsulating the dedicated dynamic-link library (DLL). The wrapper can be injected by mimicking such a library during the loading process so that all consequent communication with the hardware will be intercepted, a technique known as detouring [73]. During the gameplay, all necessary information for reconstructing the frame was stored and post-processed separately afterwards.

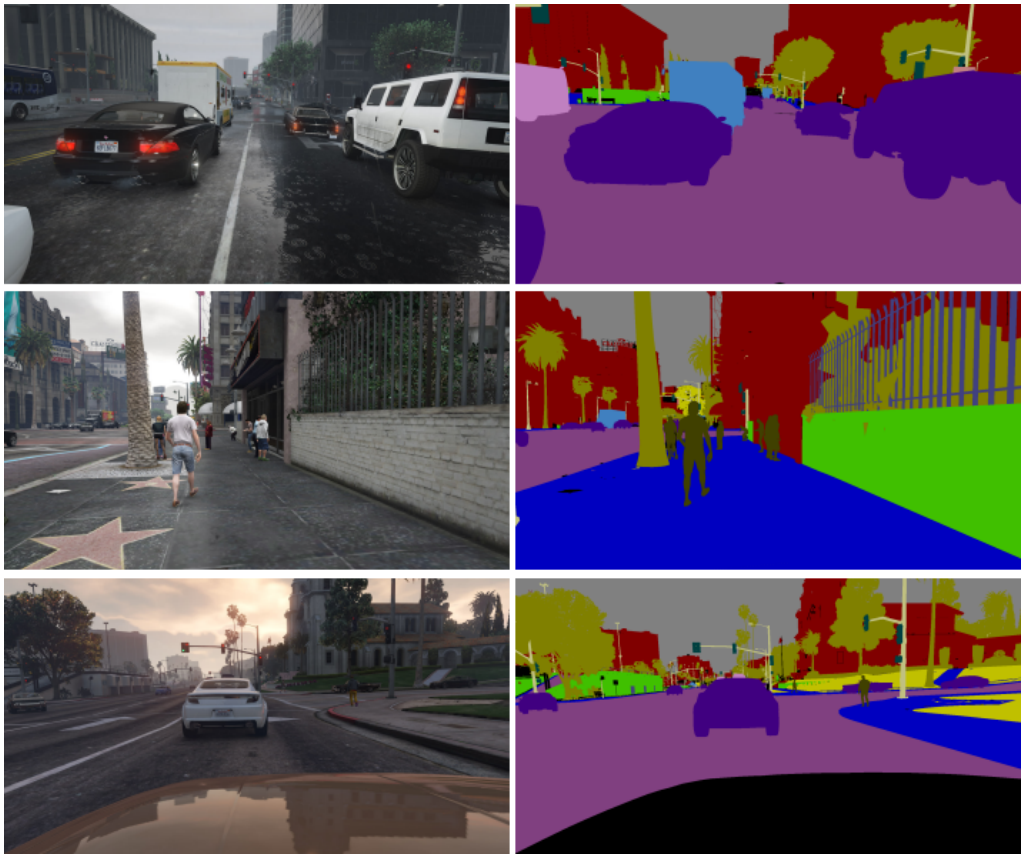


Figure 2.3: Examples of *Playing for Data* dataset and corresponding semantic annotations samples [4].

The procedure described previously returns a per-pixel ID map for each produced frame, which identifies the mesh, texture, and shader used for computing an individual pixel. These maps are later subsequently dissected into patches, which share the same Mesh-Texture-Shader (MTS) triplet. Objects often consist of multiple patches, but the boundaries between patches are consistently aligned with class boundaries. As a result, semantic labels can be obtained by simply grouping the patches by annotators.

This way, a dataset of $24,966$ frames along with semantic annotations was extracted from the video game. Each frame has a resolution of 1914×1052 pixels. Figure 2.3 demonstrates several examples of the extracted frames along with their corresponding semantic maps. The annotations encompass 19 classes, including road, building, sky, sidewalk, vegetation, car, terrain, wall, truck, pole, fence, bus, person, traffic light, traffic sign, train, motorcycle, rider, and bicycle. Therefore, the dataset is compatible with standard real-world datasets such as Cityscapes [29]. Furthermore, it reveals a relatively high degree of variability

Figure 2.4: Example of the VIPER dataset frame along with corresponding semantic, instance, flow, odometry and detection labels [5].

where 26.5% of all Mesh-Texture-Shader combinations only occur in one frame of the entire collected dataset.

In comparison to PfD, VIPER comprises significantly more data, containing 250,000 examples in contrast to the original 25,000, and covers a wider range of computer vision tasks, including semantic and instance segmentation, optical flow estimation, 3D object detection and visual odometry. Figure 2.4 shows several examples of described ground truth. Additionally, this ground truth is provided in a temporally consistent manner through a number of video sequences. The extended list of label types could be achieved by employing the dynamic software updating [74], which broadcasts resource identifier, depth, and transparency value for every single pixel by exploiting the bytecode of the shaders distributed on a GPU. Additionally, this method utilizes access to the transformation matrices of individual objects. Clustering such matrices enable the segmentation of the individual objects and the generation of the instance segmentation ground truth. These transformation matrices, along with meshes used for rendering, allow the reconstruction of camera positions and the generation of ground truth for visual odometry. Furthermore, the vertex buffers of these meshes provide information about bounding boxes.

The VIPER captures data in five distinct ambient conditions: daytime, sunset, rainy, snowy, and nighttime. The split between training, validation, and testing shares a balanced distribution of data acquired across these conditions but covers geographically diverse areas. Such split reveals relatively high realism regarding distributions of the number of categories and instances present in each image measured with Jensen–Shannon divergence (JSD). As a point of reference, the VIPER utilizes the Cityscapes [29] dataset for realism measures.

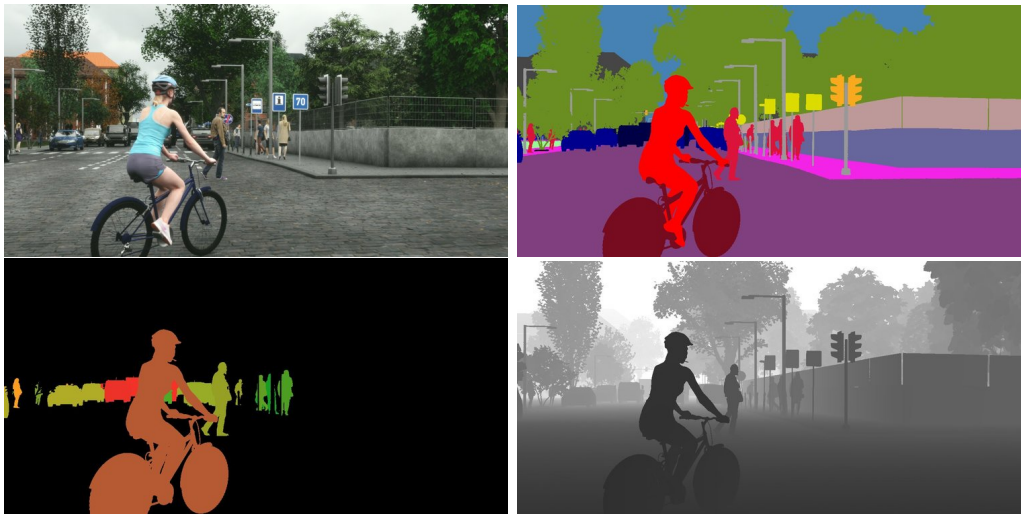


Figure 2.5: Example of the Synscape dataset frame along with corresponding semantic, instance and depth annotations [6].

2.2.3 Synscapes

Synscapes dataset [6] addresses several limitations of previous synthetic datasets by focusing on two main directions of improvement. Firstly, it captures the effects of illumination and the scene's geometric and material composition on sensors in a more accurate manner compared to datasets like SYNTHIA, which rely on off-the-shelf assets. Secondly, it tackles the scalability issues of the VIPER dataset by utilizing the technique known as the procedural content generation to create the scenarios. The VIPER bases on the grounds laid by the third-party commercial product and thus avoid the amount of technical and artistic work required to create a comprehensive virtual environment. Contrary to that, Synscapes uses a procedural engine introduced by Tsirikoglou et al. [75] to parameterise the scenarios for virtual world generation.

Synscapes defines a set of rules and parameters for constructing a unique virtual world for any single rendered frame. These rules include width, number of lanes for the roads, the height of curbs for sidewalks, and window sizes for buildings, in addition to standard parameters such as sizes and materials,

The data generation pipeline utilizes the unbiased path tracing and Monte Carlo integration [71] rendering technique, which calculates the transport of light based on the radiometric properties of the sun and sky and simulates light interaction using reflectance models. The pipeline renders $25,000$ RGB images of the resolution 2048×1024 pixels similar to the Cityscapes dataset. For every image, there is also an associated semantic map, an instance map, and a depth map.

2.3 Transfer Learning

In machine learning, the common assumption states that training and test data are independent and identically distributed (*iid*). However, such a setting is rarely achievable in the real world, where, more often, the domain in which data was acquired may differ from the domain of interest. Handling such discrepancy is immensely important in autonomous driving, as it can arise due to various factors such as outdated training data, multi-sensor setup, and changes in geographic, meteorologic, or temporal conditions. A subfield of machine learning that applies the knowledge gained in one domain to another is known as transfer learning [76]. The transfer learning methods aim to extract knowledge from a source domain and apply it in a target domain. Pan et al. [76] defines the transfer learning as follows:

"Given a source domain $\mathcal{D}_s$ and learning task $\mathcal{T}_s$, a target domain $\mathcal{D}_t$ and learning task $\mathcal{T}_t$, transfer learning aims to improve the learning of the target predictive function $f_t()$ in $\mathcal{D}_t$ using the knowledge in $\mathcal{D}_s$ and $\mathcal{T}_s$, where $\mathcal{D}_s \neq \mathcal{D}_t$, or $\mathcal{T}_s \neq \mathcal{T}_t$."

Here, $\mathcal{D} = \{\mathcal{X}, P(X)\}$ is described by a feature space $\mathcal{X}$ and X is a random variable so that $X = \{x_1, x_2, ..., x_n\} \in \mathcal{X}$ follows marginal distribution $P(X)$. Task $\mathcal{T} = \{\mathcal{Y}, P(Y|X)\}$ assumes estimation of the $P(Y|X)$ based on observation pairs $\{x_i, y_i\}$ where $x_i \in X$ and $y_i \in Y$. Source and target domains are denoted as $\mathcal{D}_s$ and $\mathcal{D}_t$, respectively. Two sets of $\{x, y\}$ sample and label pairs are described as follows:

$$\begin{aligned} D_s &= (x_i^s, y_i^s); x_i^s \in X_s \subset \mathcal{X}_s, y_i^s \in Y_s \subset \mathcal{Y} \\ D_t &= (x_j^t, y_j^t); x_j^t \in X_t \subset \mathcal{X}_t, y_i^t \in Y_t \subset \mathcal{Y} \end{aligned} \tag{2.5}$$

Based on the definition, Pan et al. [76] propose a taxonomy where transfer learning distinguishes between inductive transfer learning and transductive transfer learning. According to Pan et al., these two categories can be defined in relation to source and target domains as follows:

- Inductive transfer learning assumes that source and target domains are the same, but tasks are different (though related). It refers to transferring the learned knowledge from one task to another.
- Transductive transfer learning relates to a process where the tasks are the same, but the domains are different. Transferring knowledge from one domain into another is related to domain adaptation.

Therefore, the methods of transductive transfer learning involve adapting to a new setting where either the input space differs from target $\mathcal{X}_s \neq \mathcal{X}_t$ or the input space is the same $\mathcal{X}_s = \mathcal{X}_t$, but the distribution of the data differs $P_s(X) \neq P_t(X)$. In both cases, domain adaptation techniques reduce the difference between the

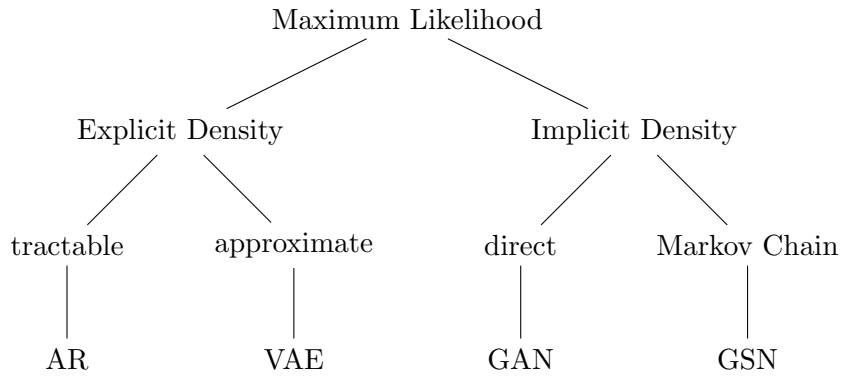


Figure 2.6: Taxonomy of deep generative models according to Goodfellow [7].

two domains. Some common strategies to achieve the reduction involve instance weighting, and feature transformation [77]. The first strategy performs domain adaptation by reweighing the importance of the individual samples based on their similarity to the target data. The second approach seeks shared latent features between the source and target. A model can discover such features explicitly by applying the loss function on learned feature vectors or by mapping source samples into the target domain. In computer vision, the task of finding this mapping function that approximates the target distribution $\hat{P}_t(X) \approx P_t(X)$ is often referred to as image-to-image translation.

2.4 Deep Generative Models.

Unlike discriminative models, which learn $p(y|x)$ conditional probability distribution over labels y given data samples x, generative models learn a distribution over x. Their goal is to find the parameters θ of a neural network so that it describes a distribution p_θ that matches the actual distribution of the data $p(x)$ or joint probability $p(x, y)$. According to the taxonomy proposed by Goodfellow [7], deep generative models distinguish the likelihood approximation or representation principle from the likelihood maximization. One category explicitly estimates the density function of the probability distribution of the underlying data p_θ, while the other can sample from p_θ. The challenge of explicit modelling lies in the complexity of the underlying information that a model must capture. There are two ways of dealing with that complexity. Methods like autoregressive models guarantee tractability by their structure, while other models use variational or Markov chain tractable approximations of density functions. Figure 2.6 shows the taxonomy proposed by Goodfellow. Currently, the research community's most popular deep generative models belong to one of these three categories:

- **autoregressive (AR) models** reproduce the conditional distribution of a particular pixel given previous pixels. This principle assumes that input space has an ordering, where every individual feature depends on the previous feature. These models define the joint distribution of features as a product of conditional distributions on each feature given the value of the preceding feature. Typically such models generate images by processing the pixels sequentially, starting from the top left and moving to the bottom right, so that the probability of each pixel is determined by conditional probabilities of previous pixels [78, 79].

- **variational autoencoder (VAE)** is a type of generative models, which acquire their generative characteristics by regularizing the learned latent space. Specifically, VAE encodes an input as a distribution over the latent space, allowing it to sample from this distribution. A decoder within the variational autoencoder then reconstructs the input from a sampled latent vector. Furthermore, regularised latent space allows encoding similar features close to each other and decoding any sampled point from latent space into a meaningful output. VAE achieves these two qualities by minimizing the reconstruction error on the output and the Kullback-Leibler divergence on the distribution over the latent space by maximizing the evidence lower bound (ELBO). In addition, variational autoencoder employs a technique known as the reparameterization trick to overcome the backpropagation problem on the sampling step [80].

- **generative adversarial network (GAN)** is a kind of model that is able, similarly to generative stochastic network (GSN) [81], to sample from the distribution directly. GAN involves a minimax game between two actors. One of the actors, the generator network, learns to produce samples from the training data, while the other actor, named a discriminator network, learns to classify samples as coming from training data or the generator. The task of the discriminator is to minimize the classification error, whereas the generator's goal is to maximize it [8].

Autoregressive models are very effective in generating plausible data, but they are computationally inefficient for sampling. On the other hand, VAEs are highly efficient in sampling, but they tend to produce images with less clear details. Finally, GANs can create images with distinguishable features, but they can be unstable during training. These three types of models are actively researched, and there is some overlap among the directions. Currently, the most active research focuses on improving image generation quality and increasing adversarial models' stability. Section 2.4.1 provides more information on GANs.

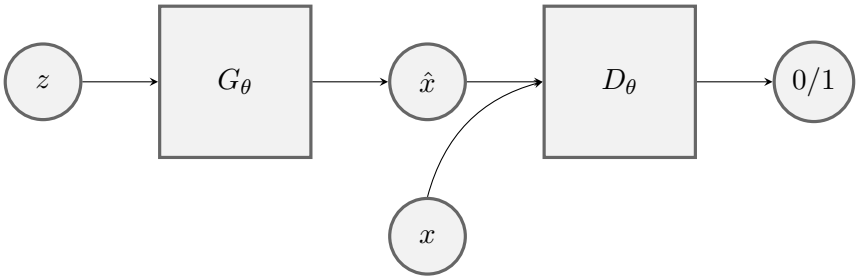


Figure 2.7: Architecture of the original generative adversarial network (GAN) with generator G, discriminator D, sampled variable z, generated sample $\hat{x}$ and real sample x.

2.4.1 Generative Adversarial Networks.

A generative adversarial network [8] is a class of generative models that intuitively distinguish themselves from discriminative models by being able to sample new data points following given data distribution in contrast to predicting labels y given existing samples x. Furthermore, generative models learn to model the joint probability $p(x, y)$ of the inputs x and the labels y or $p(x)$ if labels are not available, whereas discriminative models learn conditional probability $p(y|x)$ [82]. Compared to other generative models, GANs approximate a function that can draw samples from target distribution, making them an example of a direct implicit density model. However, unlike explicit density models, they do not estimate actual probability density functions.

GAN aims to learn the distribution of data P_g over samples x by approximating a differential function $G(z, \theta_G)$ called the generator, which takes a random noise vector $z \sim P_z$ as input and produces a sample $\hat{x} \sim P_g$ conditioned on the vector z so that this sample is indistinguishable from an arbitrary reference sample $x \sim P_{data}$. A multilayer perceptron (MLP) with parameters θ_G represents the generator function G. A generative adversarial network is exposed to the actual data through the target samples $x \sim P_{data}$. The loss function, which measures the similarity of the generated images, is derived from the second component of the model called discriminator D. The discriminator, in turn, learns to distinguish the images produced by the generator from those of reference data. The discriminator is trained to maximize $\log D(x)$ the probability of correctly classifying samples x and $G(z)$ and the generator to minimize binary cross-entropy $\log(1 - D(G(z)))$. The value function $\mathcal{L}$ guides the process of training the G and D simultaneously, which represents a zero-sum game that reaches its optimum in a Nash equilibrium when $P_g = P_{data}$:

$$\mathcal{L}(G, D) = \mathop{\mathbb{E}}_{x}[\log D(x)] + \mathop{\mathbb{E}}_{z}[\log(1 - D(G(z)))] \tag{2.6}$$

Thus, generative adversarial network aims to optimize following objective which is equivalent to minimizing Jensen–Shannon divergence between P_g and

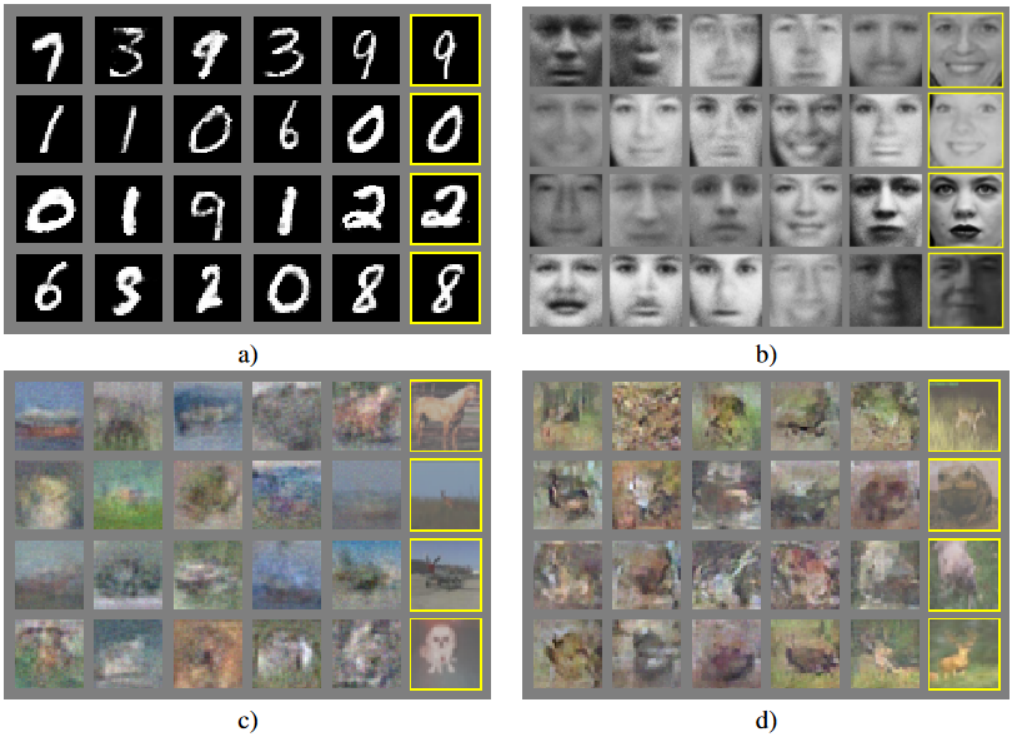


Figure 2.8: Images produced by the original generative adversarial network (GAN) [8].

P_{data}:

$$\arg\min_{G}\max_{D}\mathcal{L}(G,D) \tag{2.7}$$

2.4.2 Other Generative Models

Recently also, alternative generative mechanisms gained attention in the community. These research directions include flow-based, diffusion-based, and NeRF methods.

Diffusion. Denoising diffusion models [83] belongs to a class of energy-based model (EBM) that gradually deconstructs a data sample x_0 by adding, for instance, Gaussian noise. After T steps of forward diffusion trajectory q resulted x_T sample is normally distributed:

$$q(x_{0..T}) = q(x_0)\prod_{t=1}^{T} q(x_t|x_{t-1}) \tag{2.8}$$

The reverse trajectory is parameterized by θ and learns to gradually reconstruct the sample from noise $p_\theta(x_{t-1}, t)$:

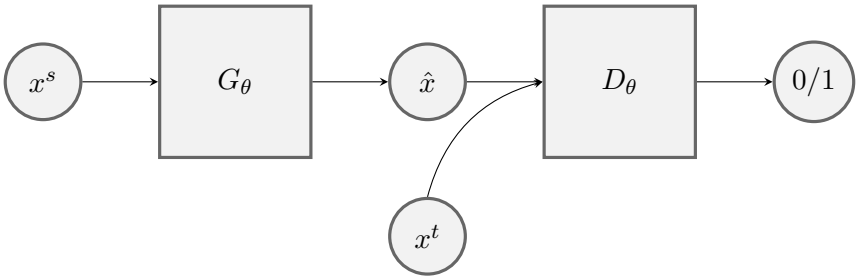


Figure 2.9: Architecture of the SimGAN approach along with generator G, discriminator D, synthetic input sample x^s, generated sample $\hat{x}$ and real sample x^t.

$$p_\theta(x_{0..T}) = p_\theta(x_T) \prod_{t=1}^{T} q(x_{t-1}|x_t) \tag{2.9}$$

Normalizing Flow. Similarly to auto-regressive models, normalizing flows belong to a class of generative models, which allow exact likelihood calculation and feature learning by applying series transformations to a random variable Z drawn from base distribution $P(Z = z)$. An invertible function $f : \mathbb{R}^n \to \mathbb{R}^n, x = f(z), z = f^{-1}(x)$ represents each transformation and maps z to observed data x. Normalizing flow model learns parameters θ of such invertible function f_θ over observed variables x so that marginal distribution $p_\theta(x)$ can be obtained from $p(z)$ and Jacobian matrix by change of variables rule.

2.5 Sim-to-Real Domain Adaptation

It became evident that learning from simulated is strongly affected by the domain gap between the simulator's output and real data distributions. The research community identified the necessity to reduce the discrepancy between them so that several methods based on generative adversarial network emerged to tackle the problem of sim-to-real adaptation. The adaptation's general idea is to improve the realism of renders using unlabelled real images.

2.5.1 SimGAN

One of the early works that identified the potential of generative adversarial networks to reduce the gap between synthetic and real distributions was the SimGAN approach [9]. In contrast to plain GAN, which generates images from random vectors, authors of SimGAN conditioned the generation process on synthetic images. Moreover, the authors identified a requirement to preserve the content of the conditioning image and avoid artifacts caused by adversarial adaptation in order to comply with synthetic annotations. To achieve this, the authors modified adversarial loss $\mathcal{L}$ from Equation 2.6 as follows:

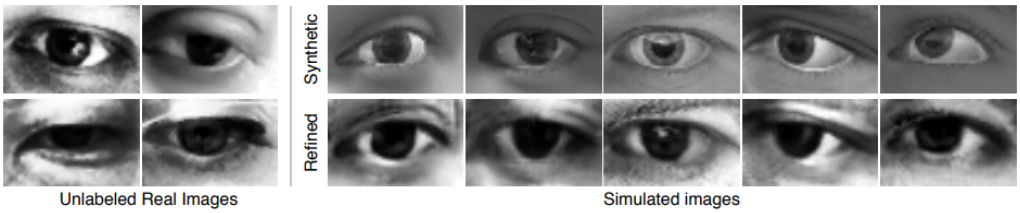


Figure 2.10: Realistic images generated by SimGAN method [9].

$$\begin{aligned} \mathcal{L}_G &= \mathbb{E}\log(1 - D(G(x^s))) \\ \mathcal{L}_D &= \mathbb{E}\log(D(G(x^s)) + \mathbb{E}\log(1 - D(x^t)) \end{aligned} \tag{2.10}$$

In Equation 2.10, x^s represents synthetic input image and x^t – real target image. The method further extended the generator's loss $\mathcal{L}_G$ with a regularisation term to enforce content preservation:

$$\mathcal{L}_R = \|G(x^s) - x^s\|_1 \tag{2.11}$$

In Equation 2.11, $\|\|_1$ is an l_1 norm. It was sufficient to employ the l_1 for the experiments on gaze and hand pose images conducted in [9], but for more complex images such as urban traffic scenes, it is not feasible anymore to employ this loss in an unsupervised setting.

2.5.2 CycleGAN

CycleGAN [10] is a method that utilizes the idea of conditional generative adversarial network [53] for creating the general-purpose image-to-image translation. It builds upon the *pix2pix* [53] framework, which learns a mapping function from source to target images. Its applicability for a general-purpose image translation in an unsupervised setting makes it suitable for domain adaptation from synthetic to real image domains. Furthermore, the CycleGAN enforces consistency between image generators in source and target domains through transitivity, a principle adopted in many areas [84, 85]. Introducing a *cycle consistency* allows CycleGAN to impose visual characteristics of target images upon source images without making assumptions about similarities between the source and the target in pixel, class, or feature space. In the context of sim-to-real transfer, such consistency improves the realism of generated images and makes them more visually aligned with target data.

In detail, CycleGAN is constructed of two mapping functions G_X and G_Y such that:

$$G_Y : X \to Y; G_X : Y \to X \tag{2.12}$$

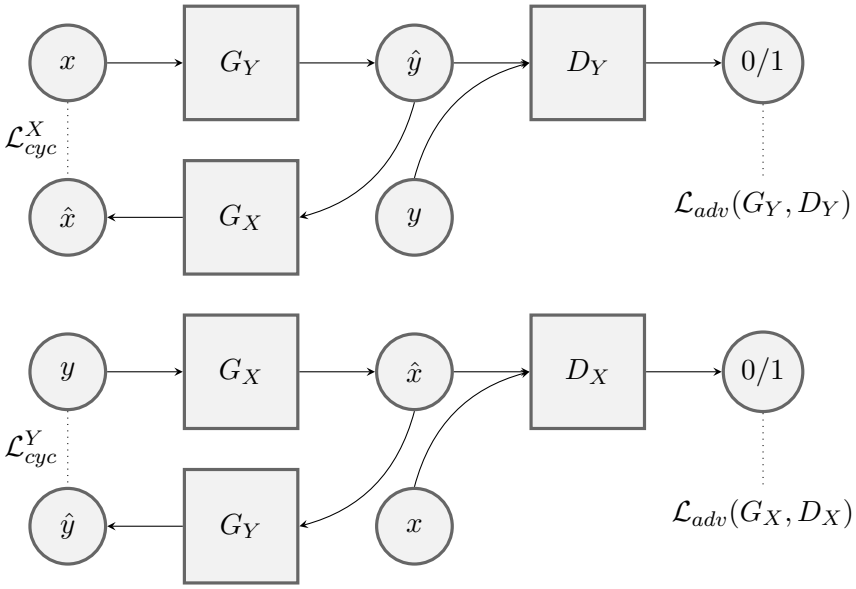


Figure 2.11: Architecture of the CycleGAN network with generators G_X and G_Y, discriminators D_X and D_Y, synthetic input sample x, reconstructed synthetic sample $\hat{x}$, real sample y and reconstructed real sample $\hat{y}$.

These mapping functions are learned with the training data $\{x_i\}, \{y_j\}$ from both domains which follow respective distributions P_X and P_Y:

$$\begin{aligned} &x_i \in X \subset \mathcal{X}, i = 0, 1, ..., N_X \\ &x \sim P_X(x) \end{aligned} \tag{2.13}$$

$$\begin{aligned} &y_j \in Y \subset \mathcal{Y}, j = 0, 1, ..., N_Y \\ &y \sim P_Y(y) \end{aligned} \tag{2.14}$$

For the learning both adversarial $\mathcal{L}_{adv}$ and cycle consistency $\mathcal{L}_{cyc}$ losses are applied so that total loss $\mathcal{L}$ is calculated as follows:

$$\mathcal{L} = \mathcal{L}_{adv}(G_X, D_X) + \mathcal{L}_{adv}(G_Y, D_Y) + \mathcal{L}_{cyc}(G_X, G_Y) \tag{2.15}$$

Here, D_Y is a discriminator network that learns to distinguish generated images $G_Y(x)$ from target images x via maximizing the following loss function, while G_Y minimizes it as described in 2.4.1:

$$\begin{aligned} &\mathcal{L}_{adv}(G_Y, D_Y) = \mathbb{E}_{y \sim P_Y}[\log D_Y(y)] \\ &+ \mathbb{E}_{x \sim P_X}[\log(1 - D_Y(G_Y(x)))] \end{aligned} \tag{2.16}$$

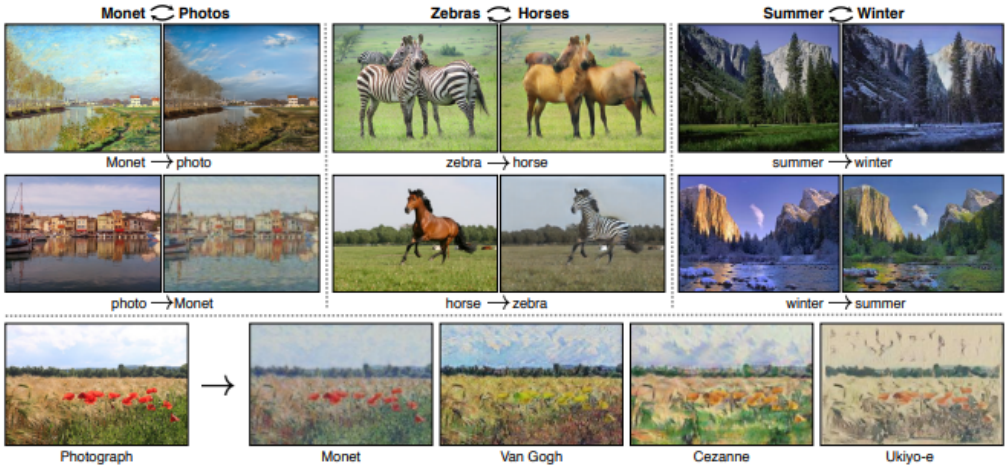


Figure 2.12: Images generated by the CycleGAN approach [10].

The loss term $\mathcal{L}_{adv}(G_X, D_X)$ with discriminator D_X is defined similarly to 2.16. Finally, the cyclic consistency loss is calculated as follows:

$$\begin{aligned}\mathcal{L}_{cyc}(G_Y, G_X) &= \mathbb{E}_{x\sim P_X}[\|G_X(G_Y(x)) - x\|_1] \\ &+ \mathbb{E}_{y\sim P_Y}[\|G_Y(G_X(y)) - y\|_1]]\end{aligned} \tag{2.17}$$

Equation 2.17 formalizes the intuition behind the cyclic consistency loss, which reduces the space of possible mapping functions so that the output distribution matches the target distribution. For example, this mapping function applied to source image x shall return an identical image: $G_X(G_Y(x)) \approx x$. This intuition is best visualized in Figure 2.11.

The approach aims to solve the following:

$$\arg \min_{G_X, G_Y} \max_{D_X, D_y} \mathcal{L}(G_X, G_Y, D_X, D_Y) \tag{2.18}$$

Both generators consist of three convolutional layers followed by 9 residual blocks. Discriminators are based on PatchGAN [53] operating on 70×70 patches.

Although CycleGAN achieves compelling and stable results in appearance learning, the results reveal multiple examples where failure cases prevail (see original publication [10]). Such cases include, for example, the confusion related to out-of-distribution objects in the horse-to-zebra experiments with a person on the horse. Another example includes class permutations in the image and semantic map task. The authors attribute such inconsistencies to the unsupervised nature of the method and the global statistics of the source and target data. Nevertheless, this work shows that the abovementioned problem is consistent in synthetic to real domain adaptation.

3 Related Work

3.1 Synthetic Data

3.1.1 Computer Vision

As previously mentioned in Chapter 1, a common approach to synthesizing traffic scene images is to use rendering engines to generate such images along with the ground-truth data required in computer vision. Many methods already employ rendered images in various recognition tasks, so researchers are increasingly interested in using virtual environments. Many modern approaches applied computer-aided design (CAD) models to a wide range of tasks, including human pose estimation, object detection, motion estimation, flow estimation, and others.

Algorithm Evaluation. Synthetic data for evaluating computer vision algorithms' performance has demonstrated effectiveness in various domains, including pose estimation, object recognition, segmentation, and flow estimation. Simulation as a controllable environment provides nearly ideal conditions for benchmarking and evaluation. For instance, one of the early works, known as OVVV [32], created a surveillance testbed using the well-known computer game to simulate various scenarios, including omnicams, controllable signal noise effects, as well as comprehensive ground truth. Another example, Kaneva et al. [86] used virtual worlds to test traditional image feature descriptors under varying illumination conditions and viewpoints. Additionally, Aubry et al. [87] analyzed convolutional neural network (CNN) feature changes originating from factors of variation occurring in real image data such as translation, scale, lighting, or colour. Other works [88], [89] used synthetic environments to evaluate algorithms for such tasks as tracking and visual odometry or SLAM. While the controllable factors of variation make synthetic data well-suited for algorithm evaluation, the recent dominance of deep learning-based computer vision methods led to the primary application of simulated data for training purposes.

Pose estimation. Shotton et al. [90] showed that the usage of large-scale synthetic imagery, which is highly varied, results in quick and accurate prediction of human body joints' position invariantly to a large number of factors of variation. Later Varol et al. [91] introduced the SURREAL dataset, which consists of 6.5 million rendered frames from motion sequences based on realistic 3D human augmentations. In the context of 6-DoF object pose estimation, [92] demonstrates that *reality gap* can be successfully mitigated by combining domain randomization and photo-realism. The authors of the DOPE network trained it entirely

on synthetic data from two datasets with objects of interest in different virtual environments: one dataset contained distractor objects on a random background and the other contained objects in a realistic background like a kitchen or forest. Another successful example of synthetic-only training for 3D pose estimation is described in [93]. In this work, a network maps feature extracted from a given real image into synthetic feature space to predict a 3D pose further.

Flow Estimation. Synthetically generated data is beneficial when labelling real data is laborious, such as estimating depth, optical flow, and scene flow. For example, Dosovitskiy et al. [94] created an unrealistic synthetic dataset called *Flying Chairs* and demonstrated that flow estimators could achieve substantial generalization abilities with synthetic images. Handa et al. [95] focused on depth-based semantic per-pixel labelling, and Papon et al. [96] rendered cluttered rooms with objects on the fly for room scene understanding, which was also a subject of study in [97]. Some early works, such as [30] and [98], proposed test suites for benchmarking flow algorithms. More recent works, including [99, 100], provide new benchmarks and evaluation methods. Mayer et al. [101] addressed the question of the synthetic data requirements for low-level tasks such as optical flow and disparity estimation and found that low-level realism involving textures or lighting is important for these tasks, while high-level realism may be necessary for tasks such as detection.

Detection. In addition, to pose estimation, several studies, such as [33], [102], [103], performed multi-category object class recognition based merely on CAD objects. Liebelt et al. [33] extracted view-agnostic class and pose features from 3D objects to find matches in real data and improve multi-view object class recognition. Aubry et al. [103] learned alignment between 3D CAD models and real examples based on specific parts. Sun et al. [104] demonstrated that a method trained on data augmented with non-photorealistic 3D CAD models can perform on par with real large-scale data when the dataset bias issue is addressed appropriately. Synthetic data in [105] demonstrated the effectiveness of training data augmentation with crowd-sourced 3D meshes without realistic texture, pose, or background. Precisely, Peng et al. [105] demonstrated that some detection models could generalize well on real test data regarding object shapes, despite the absence of textures, by varying those cues. In [106], the part model leverages CAD data of the cars to extend them with viewpoint and part-level geometry information and, by that, to achieve viewpoint estimation at any degree of granularity.

Segmentation. Segmentation tasks also face challenges similar to flow estimation, where pixel-dense labelling can be a resource-intensive problem. Thus, virtual environments were extensively used for indoor segmentation [107, 96] and surveillance [108]. The rendering pipeline from [96] enabled the production of 7000 random indoor scenes with a total of $59,784$ instances and demonstrated high adaptability to real data for the classification and pose estimation of objects. Other segmentation works using rendered data include [109], [110], [111]. Zhang et al. [109] studied the robustness of stereo feature matching by controlling the

texture or transparency of the objects in synthesized images. Su et al. [110] used a hybrid dataset with various renderings of the same synthetic scene for viewpoint estimation. McCormac et al. [112] extended upon the SceneNet dataset [107] scene layouts and objects from ShapeNet [113] by randomly sampling the scenes, camera positions, and camera trajectories. Additionally, randomized textures and lighting in [112] allowed acquiring 16,895 various configurations for the indoor scene, resulting in 5 million different samples. This work demonstrated that pre-training solely on synthetic data could be very beneficial and significantly improve indoor scene understanding with CNNs. The results of [112] confirm the findings from [107] but to a larger extent. Another significant advancement in indoor scene understanding is demonstrated in [114], which focuses on semantic scene completion, where complete voxel representations and semantic labels are produced based on single-view depth-map input. To this end, the authors trained an end-to-end network on a manually constructed synthetic dataset of indoor environments consisting of 45,000 indoor scenes with room layouts assembled manually from predefined 3D furniture models.

Miscellaneous. Other notable works employed rendered images for a wider variety of tasks. One of them presented the PHAV dataset [115] with 39,982 videos dedicated to action recognition tasks. The dataset includes 35 different action categories like *jump, run, stand, walk* but also more complex as *brush hair, moonwalk* or *walk holding hands*. The data synthesis pipeline introduced in PHAV allows the procedural generation of physically plausible variations of actions obtained through motion capture or even from scratch with programmatically defined behaviours. Another significant work, CLEVR [116], is focused on visual reasoning and represents an unbiased diagnostic dataset for intelligent systems answering questions about visual data. A visual question-answering (VQA) system must be able to recognize objects and their spatial relations and perform higher-level logical inference or comparisons. CLEVR, with its 853,000 questions, was designed to determine whether such an intelligent system can understand the underlying scene rather than simply finding statistical cues in biased datasets.

3.1.2 Autonomous Driving

Data synthesis is very beneficial in the autonomous driving field due to the complexity and diversity of data and scenarios. It has been utilized for various tasks related to traffic scene understanding, including pedestrian detection, to cover a wide range of potential traffic scenarios. For example, Pishchulin et al. [117] used 3D human models to create many shape variations and showed that only eleven models suffice for substantial dataset generation for the pedestrian detection task. Marin et al. [118] and Vazquez et al. [119] assessed commonplace histogram of oriented gradients (HOG), and linear support vector machines (SVM) in the context of pedestrian detection by applying them in a virtual environment. Veeravasarapu et al. [120] studied realism for urban scene understanding. Another work that exploited the idea of texture realism was [121], which used

two separate segmentation, and detection backbones focused on the texture realism of the background classes in one and the shape realism of the objects in another. Busto et al. [122] refined coarse annotations of car pose by discretization of 3D car model views, and Chen et al. [123] improved car segmentation using 3D bounding boxes. Finally, Shafei et al. [124] demonstrate a number of experiments to evaluate the fitness of computer graphics imagery for semantic segmentation and depth prediction.

Multiple recent works revealed synthetic datasets consisting entirely of rendered imagery. The most prominent of them were discussed in more detail in Section 2.2. For example, Haltakov et al. [125] provided accurate annotations of depth and segmentation for around $8,000$ frames. One of the significant large-scale datasets in the field is SYNTHIA [3], which provides more than $200,000$ frames of urban traffic scenes in different environmental conditions. Another important dataset, *Playing for Data* (PfD) [4], used a video game engine to annotate $25,000$ images. Finally, Gaidon et al. [126] introduced *proxy virtual worlds* by virtually mirroring the original KITTI traffic scenes [127]. *Virtual KITTI* with $17,000$ frames intended to cover the complete spectrum of computer vision tasks, including object detection, multi-object tracking, semantic and instance segmentation, optical flow, and depth estimation. Also, the authors quantitatively analyzed the domain gap between the proposed data and its real counterpart in the object detection downstream task. The authors found that the real-to-synthetic gap was relatively insignificant as the deep models trained on real KITTI performed comparably well on the proposed data. Another approach [128] that examined the domain gap between real and synthetic data for object detectors utilized the same game engine as [4]. While the evaluation protocol differed from that of [126], in that a detector was trained on synthetic data and evaluated on real data, their results were consistent with the findings of [126]. They show that the gap between generated and KITTI data for the detection task was relatively minor and even smaller than that from Cityscapes. There have been several synthetic datasets created for a wide range of tasks. One such dataset is VIPER [5], potentially containing the highest variability of scenes with approximately $500,000$ densely annotated frames. One of the recent datasets, Synscapes [6], is another dataset known for its realistic samples, achieved through unbiased rendering, camera effects simulation, motion blur, and procedural scene generation, which can significantly contribute to improvement in downstream object detection. Other recently developed datasets include *Parallel Domain* [129] and SHIFT [130]. These datasets became standard benchmarks frequently used for training and evaluating sim-to-real transfer methods, though several simulators can also offer the possibility to create customized synthetic data and extend it to a preferred learning task. There are several publicly available driving simulators, such as TORCS [131], [132], CARLA [34], and VIVID [133], which allow the capture of sensor data along with annotations. For example, TORCS is an open-source racing simulator that became widely acknowledged for its accurate simulation of physical characteristics of driving dynamics, including inertia, sus-

pension, and friction. However, its focus on the racing environment limits its application for typical urban driving-related tasks. Another open-source simulation, CARLA, aims to address such shortcomings and provide additional functions required for research in autonomous driving. It offers a configurable setup of sensors with ground truth data covering various computer vision tasks like segmentation, depth estimation, and detection. Additionally, CARLA provides customizable scenarios and environmental conditions described through scripts.

Several works have explored using semi-synthetic data by rendering single objects of interest rather than entire images. For example, [134] used virtual humans to study human pose estimation. MixedPeds, introduced by Cheung et al. [135], exploited the idea of pedestrian augmentation in autonomous driving. Huang et al. [136] introduced a data synthesis pipeline for so-called *long tail* traffic situations and a resulting dataset with pedestrians in unsafe traffic scenarios. Alhaija [137] also proposed a flexible technique for data generation by augmenting virtual car models into real images. The augmentation pipeline proposed by Alhaija et al. [137] aims to address some of the challenges of synthetic data generation by using realistic backgrounds to reduce the need to model complex 3D traffic scenes and environment maps to enable object placement consistent with the entire scene. Finally, Lee et al. [138] proposed a model that statistically estimates optimal locations and shapes of objects inserted onto layouts and its semantics for generating novel images. More advanced augmentation pipelines like AADS [139] used the several available techniques and information sources for traffic scene augmentation to achieve high-quality in-paintings with comprehensive annotations. For example, AADS relied on ApolloScape [140] camera images, point cloud labels, and trajectories as raw data. The augmentation process removes dynamic objects, estimates traffic, places synthetic cars into plausible locations, and performs illumination estimation and texture enhancement. While the earlier methods produced valuable training samples, they were still dissimilar to real samples. More sophisticated techniques, such as those based on generative models and generative adversarial network architecture, learned the appearance features, which help create examples looking like the domain of interest. So [141] proposed a flexible technique to augment real images with realistic car objects and confirmed that augmented data improves car detection compared to pure synthetic data. Such augmentation gets increasingly advanced with the recent successes demonstrated by GeoSim [142].

3.2 Generative Modeling

3.2.1 Domain Adaptation

Standard machine learning methods presume that the training and test examples are sampled from the same distribution and thus are independent and identically distributed (*iid*). However, when synthetic and real image data are used, this assumption may not hold as the underlying distributions of these two data sources

differ, leading to a phenomenon known as domain shift [41]. As a result, models trained solely on synthetic data can exhibit poor generalization performance on real-world data. Domain adaptation techniques aim to mitigate this issue by reducing the discrepancy between the distributions of the two data sources.

Unsupervised Domain Adaptation. Minimizing the discrepancy between two domains is especially difficult in an unsupervised setting, where pairs of images from corresponding domains are not accessible. This setting is idiosyncratic for sim-to-real domain adaptation, which seeks to attune models trained on synthetic data to real-world data. Multiple strategies fall into the category of unsupervised domain adaptation: naive joint training or combined pretraining with fine-tuning, entropy minimization [143], curriculum learning [144], generative adversarial network based approaches [10], and classifier discrepancy [145]. Many highly regarded works rely on the adversarial framework GAN [8] to facilitate the adaptation process. They use adversarial training for either data generation or direct task learning. Task learning methods commonly utilize synthetic and real images as input and aim to produce segmentation maps or other computer vision predictions but do not generate additional data. While the adversarial loss can alleviate the difference between rendered and real images, it is not sufficient for accurately classifying or detecting objects in the target domain. Multiple approaches introduced various regularization techniques to address this issue. Saito et al. [145] utilize discrepancy loss to align features from source and target samples. Luo et al. [146] follow a similar hypothesis, but they distinguish between well and poorly-aligned features in order to tackle the latter stronger. Instead of applying adversarial loss directly on features, Tsai et al. [147] apply it to inferred segmentation maps. Other examples include [121], [148], [149] and [150], [48], [151] [15] but they are out of scope of this dissertation. Another category of sim-to-real domain adaptation known as generative methods focuses on translating synthetic images to real ones, which are then used for task prediction learning. These mechanisms use an adversarial loss to generate high-resolution, visually pleasing images by minimizing the distance between the generated and reference distributions. This work focuses on exploring and advancing research in this direction.

3.2.2 Adversarial Domain Adaptation

Both task- and data-generation approaches often utilize generative adversarial network to perform image-to-image translation, and both are prone to generating semantics confusing samples. Various constraints have been introduced to address this issue and preserve the semantics of the generated samples. Many researchers focused on designing such constraints in adversarial models to overcome the *sim-to-real* mismatch problem. The CycleGAN, introduced by Zhu et al. [10], is one of the fundamental image transfer frameworks. CycleGAN restricts the generator network by adopting the transitivity principle upon bijective image transformations as a regularization in order to ensure common

semantics between the source and target samples. Liu et al. [50] use weight sharing between the layers of the generator and discriminator to learn the joint distribution of data. In subsequent work, the authors [51] introduced the UNIT framework, which extends the idea of weight sharing and common latent space. Huang et al. extend this idea in their work on a multi-modal version of the previous work, known as MUNIT [14], which allows the separation of internal image representation into the content and style features. GAN-based algorithms generate visually satisfying images, but they often cannot preserve the high-level semantic structure of the original image.

Semantic Consistency. Various attempts were made to address the problem of semantic inconsistency in the image-to-image translation. For example, SimGAN [9] is a model that uses self-regularization loss to discourage semantic changes and avoid significant alteration of the original image. The authors of CoupledGAN [50], and VAE-GAN [51] assume the existence of a common latent space for source and target domains. An alternative line of research involves using additional information, such as semantic maps, to identify changes between the source and translated samples. However, when translating to the target distribution $P_t(X)$, generative adversarial networks only operate in the input space $\mathcal{X}$, ignoring the target labels space $\mathcal{Y}$, which can lead to the introduction of semantically inconsistent artifacts in the generated samples, as shown in Figure 1.2. Recent research has used the target labels as privileged information about the distribution $P_t(Y)$ to address the inconsistency. For example, CyCADA [47] improves upon the work of Zhu et al. [10] by introducing perception loss, which preserves semantic consistency by constraining cycle consistent task-loss and encourages equivalent segmentation of the input sample before and after refinement. As an alternative to segmentation loss, [152] utilized the geometry consistency loss for a similar purpose. The task loss measures the difference between the inferred semantic maps from the original input and the translation. Chen et al. [48] also agree with the assumption that the original input sample in the target domain should result in equivalent segmentation prediction. Li et al. [49] demonstrate that integrating image transfer and segmentation leads to improved translation quality. Other studies, such as [51] and [15], combine cycle consistency and adversarial loss in an effort to separate appearance and content by learning the latent representation space.

There are various methods for image transfer problems preserving the macrostructure of original samples, apart from the ones leveraging perception models to measure the dissimilarity of source and translation. Li et al. [52] integrated the soft gradient-sensitive objective to maintain semantics. It tracks deviation by applying the Sobel filter to the image and its corresponding semantic map since alterations in the translated image would change the boundaries of objects. According to the proponents of the DLOW approach [153], intermediate domains are beneficial representations for mitigating the gap between domains. Their work proposes multiple target domains to generate corresponding intermediate domains, which are used as sources for discriminators in adversarial learning.

Chen et al. [154] proposed that, in addition to semantic labels, depth maps represent an effective tool for measuring the differences between sources and translations. Another method is to adapt the feature maps obtained from images. Hong et al. [44] employs an adversarial loss to differentiate source and target samples based on the features extracted by *Fully Convolutional Network* (FCN) [155].

Conditional Generation. Several works have adopted the strategy of conditional generation to circumvent the challenges and limitations of pure sim-to-real transfer. Isola et al. and Liu et al. in Pix2Pix [53, 51] proposed a GAN model conditioned on layouts corresponding to images. Other approaches, such as CRN [156], Pix2PixHD [157], SPADE [158], OASIS [159], and CUT [160] also utilize privileged information such as semantics. In the paper by Qi et al. [54], a different approach to conditioned image synthesis was proposed in which a model uses pre-stored patches from a collection to generate an image canvas. Wang et al. improved Conditional GANs in [55] and integrated instance information with image manipulation possibility. A plehora of these methods [161], [162], [163], [164] are based on generative adversarial network [8] for image synthesis. Alternative conditioning techniques include more conceptual inputs like natural language description [165] or scene graphs. The latter ones are data structures, which represent scenes as directed graphs with nodes embodying objects and edges embodying relations between them, they have been used as an alternative to textual query for image retrieval [66], and description [166], as well as to image generation [167, 67, 168]. Ashual et al. extended the graph-convolutional approach from [167] in a multitude of ways. The authors proposed dual embedding for layout and appearance introduced location attributes in the image space, and improved object shapes in the image layout by applying adversarial loss to them. Dhamo et al. in [168] enhanced the scene graphs with more complex semantic relations between objects and also enabled graph manipulation in an unsupervised fashion.

3.2.3 Metrics

The original GAN method and a plethora of subsequent generative adversarial networks lack direct objective function, which makes it challenging to objectively evaluate their performance and the quality of generated samples, in particular, [169]. Furthermore, physical realism pursued by rendering engines is only loosely related to the data realism of generative models. As discussed in Section 2.1, the rendering engines aim at the optically correct approximation of light transport, whilst generative models seek to reconstruct reference data distribution producing samples that are most similar to it. This similarity presents a complex problem of quantification.

Perceptual study. Generative methods for image synthesis have been widely studied, and humans are able to assess the similarity of generated images to reference data with relative ease. Perceptual studies formalize human evaluations

of the generated images, often using Amazon Mechanical Turk (AMT) to asses their realism. The AMT evaluation protocol involves presenting human participants with both reference data and generated images and asking them to choose which images they believe are authentic. These evaluations are conducted in multiple trials and sessions, and the final results are reported as the ratio of generated images labelled as authentic by the participants. While this metric can provide some general information about image quality, it is still subjective and may require domain expertise in certain cases, such as medical imaging.

Inception Score. More objective evaluation methods are necessary due to the lack of correlation between standard log-likelihood comparisons and the perceptual evaluation of generated images, besides it is difficult to use kernel density estimation in high-dimensional spaces. IS [170] and Fréchet inception distance (FID)[171] are commonly used metrics for this purpose, with various variations also existing. IS and FID rely on the Inception classification model pre-trained on the ImageNet dataset but assess different statistics. IS analyses generated images by computing the Kullback-Leibler divergence between a conditional and marginal class distribution. FID, in turn, calculates the 2-Wasserstein distance between the multi-variate Gaussian fitted to latent space of the Inception model applied to generated and real reference data. Although these metrics can capture certain aspects of the generated samples, they show a bias towards the ImageNet dataset and Inception model and therefore do not adequately evaluate generative models.

Alternatives. Several metrics have been proposed for evaluating the performance of generative models, with a focus on minimizing bias towards the ImageNet dataset. These are perceptual path length (PPL) [164] and local intrinsic dimensionality (LID) [172]. The PPL quantifies the disentanglement degree of the generator's latent space and how factors of variation are adequately separated, while LID measures the degree to which two manifolds of data distributions overlap. In contrast, learned perceptual image patch similarity (LPIPS) and structural similarity index measure (SSIM) assess the similarity in the image domain. The LPIPS utilizes a VGG model, and SSIM relies on luminance, contrast, and structure.

Downstream Task. Most approaches to evaluating generative models rely on the assumption that if a model can produce perceptually meaningful data, it should be able to support downstream tasks. Therefore, evaluation of the generated data commonly employs dense class prediction models such as *Fully Convolutional Network* (FCN) [155], *Dilated Residual Network* (DRN) [173], or Deeplab[174]. These semantic segmentation models are trained on generated images and evaluated on real ones. Segmentation models typically consist of several convolutional layers that learn feature vectors, which are then upsampled to produce a semantic map using a combination of interpolations or convolutions.

Part II

4 Augmentation

Chapter 1 discussed an immense demand for reliable training data for deep computer vision models. A training dataset must be exact, i.e., with error-free annotations; comprehensive, i.e., provide abundant training samples; diverse, i.e., comprise high variance of scenarios. Synthetic data aims to ease the fulfillment of some of those requirements but struggles with others. Domain gap between simulated and real data is an additional obstacle that arises while using the virtual 3D environment. Alhaija et al. [137] suggested a problem relaxation where a computer vision task such as car detection does not require the entire render but can rely on real images augmented with 3D models of cars. The method suggested in [137] places the cars onto the image and estimates an environment map to improve augmented cars' quality and retain generated images' low-level realism. The proposed relaxation decreases the discrepancy between both domains and, thus, the complexity of the sim-to-real setting for a computer vision model. If the downstream task permits, augmentation can reduce that discrepancy. Indeed, many tasks in autonomous driving, such as object detection, semantic segmentation, and instance segmentation, only involve identifying individual objects, such as cars, pedestrians, and cyclists.

This chapter proposes an augmentation pipeline for data synthesis focusing on a group of objects known as vulnerable road users (VRU). This approach involves separating the generated data's high-level realism and high-level realism by handling the content placement and appearance individually. More specifically, the strategy proposes to simulate the content automatically in a geometrically correct way and separately to learn the appearance of simulated objects from real reference data. The proposed appearance learning of the augmented pedestrians relies on an adversarial style transfer technique. As discussed previously the transfer concept involves learning a function that maps the samples from a source domain to a target domain.

4.1 Pedestrian Augmentation and Appearance Learning

The proposed synthesis pipeline includes data augmentation and appearance learning. The data augmentation inserts 3D models of pedestrians into the virtual scene and blends the resulting render with the actual camera frame of the scene. The goal of the augmentation is to ensure that the blending process is accurate in terms of geometry and optics. Geometrical correctness requires that the VRUs are placed only in designated areas of the scene, such as sidewalks and roads and that there are no collisions with other objects present in the scene,

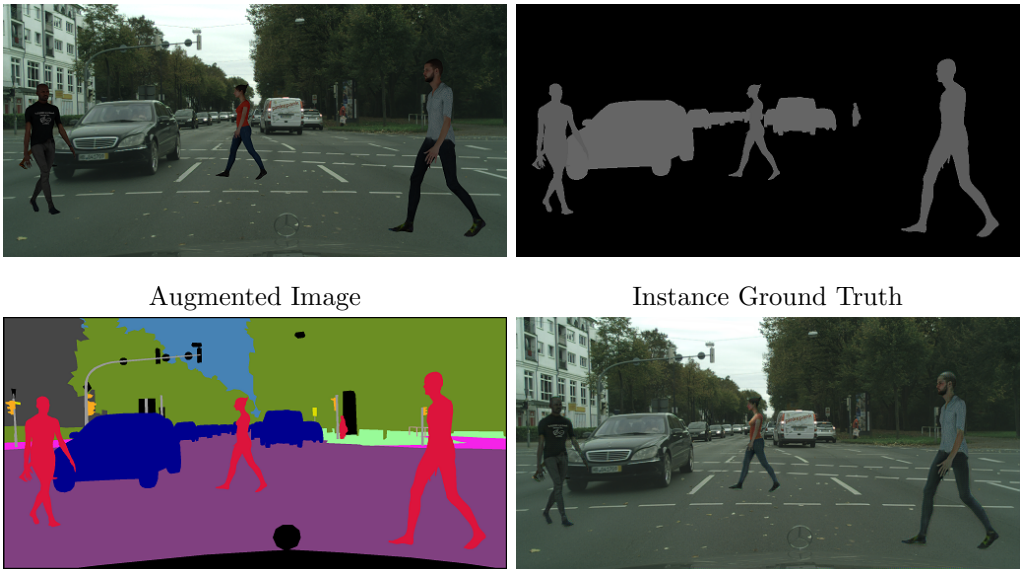


Figure 4.1: An example of a real image from Cityscapes dataset augmented with 3D pedestrian models along with generated semantic and instance maps and translated image © 2020 IEEE.

such as cars, trees, and poles. The calibration parameters of the virtual camera are adjusted to match those provided by the dataset to guarantee optical correctness. Since the blending only handles the correct placement of the virtual models in a 3D environment but not the visual realism, the in-painted 3D models in the resulting frames stand out in an obtrusive manner. Figure 4.1 shows an example of such appearance.

The synthetic appearance of the in-painted models is addressed by the appearance learning phase of the data generation pipeline, which extracts the target data's visual features and applies them to the blended models so that they look more like target data. In addition, the adversarial framework, which has demonstrated its effectiveness at reducing the visual discrepancy between data samples from different domains, helps to learn the realistic appearance. Thus, the final architecture of the pipeline consists of a generator and several masked discriminator networks. The generator uses the architecture proposed by Zhu et al. [10], and the discriminator adopts the proposed multi-discriminator architecture. Section 4.3 discusses the structure of the multi-discriminator in detail.

4.2 Data Augmentation

Spawn Map. The data augmentation phase begins with estimating a *spawn map*, which identifies the proper positioning of virtual pedestrians without the

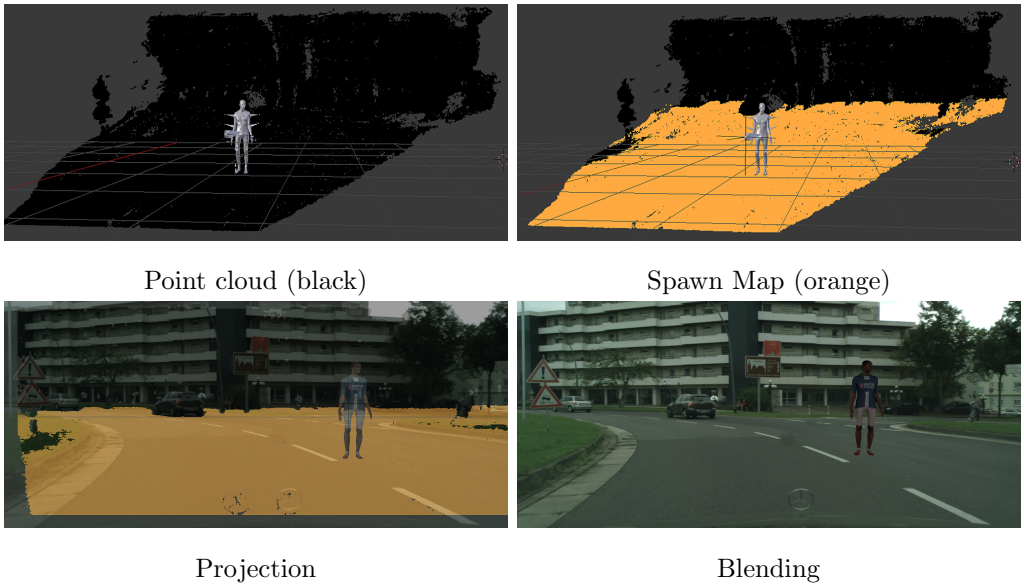


Figure 4.2: Visualization of 3D model placement pipeline with reconstructed stereo point cloud, estimated collision-free *spawn map*, and rendering of the 3D model blended with the scene frame. © 2020 IEEE

risk of collision. Collision avoidance prevents newly positioned objects from overlapping with objects already present in the scene, such as buildings, vegetation, or cars. The estimation of the spawn map relies merely on the spatial information about the virtual traffic scene, which can be obtained using sensors such as lidar or a stereo camera. Next, the proposed method estimates the underlying geometry of the traffic scene by calculating the depth from disparity maps. The Cityscapes dataset provides camera intrinsic and extrinsic parameters along with per-pixel disparity values for every camera frame to support depth estimation:

```
1  def get_depth_map(disparity_map, camera)
2      depth_map = camera.baseline*camera.f/disparity_map
3      return depth_map
```

Figure 4.2 demonstrates a point cloud obtained from the calculated depth map along with *spawn map* and the final blending of a 3D model and background images. Figure 4.3 shows the augmentation process based on the available Lidar point cloud in the KITTI dataset.

Next, estimating the ground plane where the algorithm can meaningfully place 3D models is necessary. A pragmatic approach to identifying the points of the obtained point cloud, which belong to the ground plain, relies on a naive threshold heuristic. Additionally, since the placement of the 3D model occurs automatically, it is necessary to remove the outliers resulting from the point cloud calculation so that the introduced models are not located outside the scene. A

Point cloud Spawn Map Blending

Figure 4.3: Visualization of 3D model placement pipeline with Lidar point cloud, estimated collision-free *spawn map*, and rendering of the 3D model blended with the KITTI scene frame.

technique known as *isolation forest* helps to remove the outlier points. Using the ground plane, which includes sidewalk and road surfaces - a commonplace location for pedestrians in the real world- helps to prevent placing them in inappropriate areas such as buildings or walls. Therefore, any arbitrary point of the *spawn map* represents the right area for a virtual pedestrian.

Collision Tracking. Additionally, the proposed method avoids collision between introduced 3D models and other objects already placed in the scene. To achieve this, a *collision map* tracks the positions occupied by these dynamic objects. This tracking represents an inverted *free space*. After *spawn map* calculation, all remaining points are added to the *collision map*. Subsequently, freshly introduced objects expand the collision map, helping to avert the overlapping. This strategy ensures that virtual pedestrians and other dynamic traffic participants do not overlap.

Blending. After the virtual scene is constructed, the rendering happens. It generates an image of virtual pedestrians with transparent backgrounds, blended with the scene's original image in the next step. Figure 4.2 illustrates the resulting blended frame with the described pedestrian layer, as well as the reconstructed point cloud, spawn map, and collision map estimates, which do not appear in the composite image. Optical correctness of the augmentation process is accomplished by adopting the intrinsic and extrinsic of the dataset's camera.

4.3 Appearance Learning

The appearance learning part addresses the problem of the synthetic look of the blended 3D models, which makes them highly discrepant with the original scene background. The main idea is to learn the visual style features of the real pedestrians in the reference dataset and apply them to augmented models. The style features can include a color scheme, camera sensor low-level features, or scene illumination.

4.3.1 Vanishing Pedestrians

Currently, most advanced techniques for image style transfer employ a framework known as generative adversarial network[8]. This framework consists of two networks involved in a mini-max game. One network learns to classify images into

Original Vanishing objects Translated

Figure 4.4: Examples of semantically inconsistent sim-to-real image transfer performed by the adversarial network during adaptation of SYNTHIA images and augmented images to the Cityscapes domain.

two categories, original and generated, by minimizing the objective function, and the other network, the generator, learns to create realistic images by maximizing the objective. The mini-max game ends ideally in the Nash equilibrium when distributions of the generated data and reference data are close, and the images are indistinguishable. The drawback of this approach is that adversarial training manipulates the structure of produced images in an attempt to restore target distribution.

As previously discussed, several domain adaptation methods exhibit this behavior. Figure 4.4 exemplifies perturbations in the augmented data resulting from the adversarial training. These perturbations can negatively impact the employing of the augmented images in the downstream task, as the images become inconsistent with the accompanied ground truth. In the case of domain adaptation between augmented and real data, the discriminator quickly learns to identify out-of-distribution objects and implicitly guides the generator to remove them. As shown in Figure 4.4, removing objects is undesirable in the designed data generation pipeline.

4.3.2 Multi-Discriminator

The proposed technique of multi-discriminator aims to address the issue of vanishing objects during the appearance learning phase by splitting the individual discriminator into multiple class-specific ones. This split decreases the decision-making freedom of the individual discriminator originally encompassing the entire

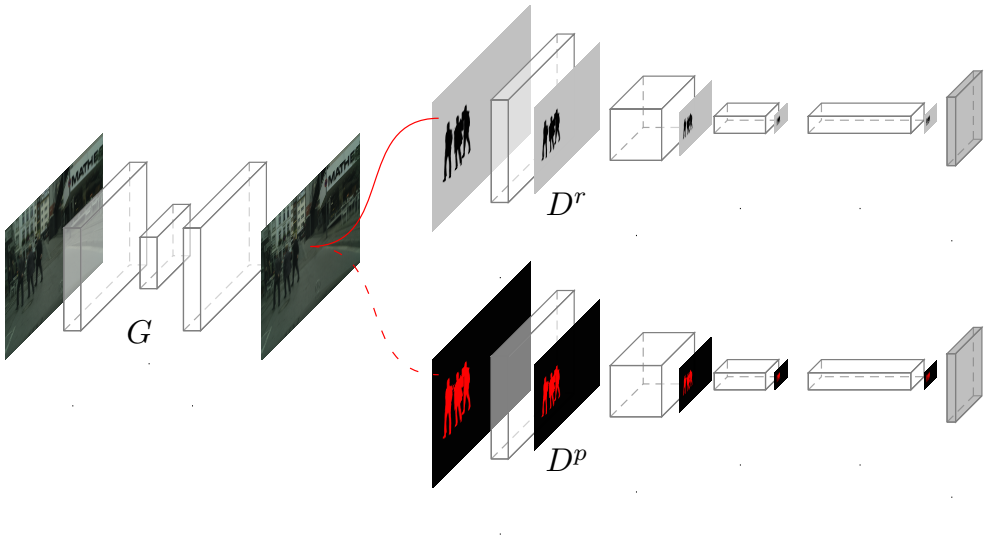


Figure 4.5: Multi-discriminator architecture consisting of generator G and two class specific discriminators D^p and D^r along with *MaskLayer* introduced after each convolution block. © 2020 IEEE

image context. Instead, the separation encourages the class-specific discriminator to assess a single class contrary to the whole image.

The multi-discriminator architecture employs several discriminators, where every individual only focuses on a patch of the images which belong to a particular class. To achieve that, dedicated masks dissect the original image into a set of disjoint patches along the semantic borders. Thus, the individual masked discriminator only gets a patch as an input. Figure 4.5 shows the overview of the masked multi-discriminator network.

During the adaptation process, a generator G takes an augmented image as an input and translates it into the original domain. Next, masking splits the translated image into patches, and each discriminator D^c receives a corresponding patch. Training a joint multi-discriminator D results in the specialization of a single D^c on the distinctive features of the corresponding class c. Optimization of the aggregated objective $\mathcal{L}_{adv}$ enables such specialization.

The objective $\mathcal{L}_{adv}$ combines several class-specific objectives:

$$\mathcal{L}_{adv}(G, D) = \sum_{c}^{N_c} \mathcal{L}(D^c, r) \tag{4.1}$$

In Equation 4.1, c denotes a class, and N_c represents the total number of classes. The pedestrian augmentation pipeline having only two classes requires merely a discriminator for the augmented objects p (pedestrians) and a discriminator for the background r (rest). As a result, the simplified version of the initial objective given in Equation 4.1 can be reduced to:

$$\mathcal{L}_{adv}(G, D) = \mathcal{L}(D^p, G) + \mathcal{L}(D^r, G) \tag{4.2}$$

4.3.3 Masking

The proposed method adopts the PatchGAN architecture [53] for each class-specific discriminator. Masking replaces the pixels of other classes with 0 values. However, convolutions allow the signal from undesired areas to propagate through the network layers and contribute to the final result. Therefore, it is necessary to suppress the activations that originate from outside the patches at each layer of the discriminator. A proposed MaskLayer applies an original mask M_c, of a class c to every feature map resulting from the discriminator's convolution layer. It requires the down-sampling of the mask to the size of the corresponding feature map. Figure 4.5 illustrates an overview of a specialized class discriminator.

Cost-sensitive loss. During the adaptation phase, the system receives pairs of input images x^i, each with a size of $3 \times h \times w$, along with the labels y^i from the augmented dataset: $\{(x_a^i, y_a^i)\}_{i=1}^{N_a}$ and pairs $\{(x_r^j, y_r^j)\}_{j=1}^{N_r}$ from the original dataset. The random variable X takes values x_a^i from the input space $\mathcal{X}$ and Y the values y_a^i from the labels space $\mathcal{Y}$, which are *iid* and follows the joint probability distribution P_a:

$$\begin{aligned} &x_a^i \in \mathcal{X}_a \subset \mathcal{X} \subset \mathbb{N}^{3\times h\times w}, i = 0, 1, ..., N_a \\ &y_a^i \in \mathcal{Y}_a \subset \mathcal{Y} \subset \mathbb{N}^{h\times w}, i = 0, 1, ..., N_a \\ &\{x_a^i, y_a^i\}_{i=1}^{N_a} \sim P_a \end{aligned} \tag{4.3}$$

In contrast, the real samples x_r^j follow the distribution P_r:

$$\begin{aligned} &x_r^j \in \mathcal{X}_r \subset \mathcal{X} \subset \mathbb{N}^{3\times h\times w}, j = 0, 1, ..., N_r \\ &y_r^j \in \mathcal{Y}_r \subset \mathcal{Y} \subset \mathbb{N}^{h\times w}, j = 0, 1, ..., N_r \\ &\{x_r^j, y_r^j\}_{j=1}^{N_r} \sim P_r \end{aligned} \tag{4.4}$$

As previously discussed, the loss function for each specialized discriminator calculates error values for the pixels belonging to dedicated patches, so the masks M_c are used for the loss calculation. Therefore, the objective function for the class discriminator omitting i and j can be expressed as follows :

$$\begin{aligned} &\mathcal{L}(D_r^c, G_r) = \\ &\mathbb{E}_{(x_r,y_r)\sim P_r}\left[\frac{1}{wh}\|D_r^c(x_r, M^c) \circ M^c(y_r)\|_{F^2}\right] + \\ &\mathbb{E}_{(x_a,y_a)\sim P_a}\left[\frac{1}{wh}\|(D_r^c(G_r(x_a), M^c) - J) \circ M^c(y_a)\|_{F^2}\right] \end{aligned} \tag{4.5}$$

In Equation 4.5, J represents an ones-matrix of size $h \times w$, and $\|\cdot\|_{F^2}$ is the Frobenius norm. The *Masked mean squared error (MSE)* is intentionally normalized by the sample size, allowing the masks of different sizes to contribute differently to the loss. The normalization enables learning the appearance features from the bigger instances where the mentioned features are more distinctive.

However, when this loss function is employed in an adversarial setting naively, it makes the background overweighing, as it commonly incorporates larger classes like road and building. The *Augmented Cityscapes* dataset is heavily imbalanced, with 95% of pixels representing non-pedestrian classes and contributing 19 times more strongly to the final objective compared to pedestrian pixels. This illustrates the importance of considering the impact of prevailing classes at the scale of the entire dataset when dealing with class-unbalanced data.

Experiments showed that using the weighting factor λ effectively enables this scaling. Furthermore, the empirical study indicates performance improvement when λ represents the class ratio:

$$\lambda = \frac{\sum \|M^p(y)\|_1}{\sum \|M^r(y)\|_1} \tag{4.6}$$

A similar calculation can be performed for multi-discriminator with higher c. Thus, the resulting *cost-sensitive objective* can be expressed as:

$$\begin{aligned} &\mathcal{L} = \lambda_{cyc}\mathcal{L}_{cyc} + \\ &\mathcal{L}_{adv}(D_r^p, G_r) + \lambda\mathcal{L}_{adv}(D_r^r, G_r) + \\ &\mathcal{L}_{adv}(D_a^p, G_a) + \lambda\mathcal{L}_{adv}(D_a^r, G_a) + \end{aligned} \tag{4.7}$$

In Equation 4.7, $\mathcal{L}_{cyc}$ denotes the *cyclic-consistency loss* along with the parameter λ_{cyc} analogous to [10]. The full optimization problem is therefore defined as follows:

$$\min_{G_r, G_a} \max_{D_r^p, D_a^p, D_r^r, D_a^r} \mathcal{L}(G_r, G_a, D_r^p, D_a^p, D_r^r, D_a^r) \tag{4.8}$$

4.4 Experiments

Augmentation learning experiments also reside on the Cityscapes public dataset [29], which contains camera frames, disparity maps, and calibration parameters – resources necessary for data augmentation. Additionally, the Cityscapes dataset includes ground truth for detection and segmentation, which can be used to evaluate the results of the augmentation pipeline on the downstream computer vision task.

Datasets. The selected dataset must fulfill certain criteria so as to be suitable for augmentation. For example, it must provide spatial information about

Original Image Virtual Pedestrians Adaptation

Figure 4.6: Examples of domain transfer performed by multi-discriminator network from augmented images to Cityscapes domain. © 2020 IEEE

the scene chosen for augmentation and ground truth for evaluating the performance of computer vision models on the generated data. The Cityscapes dataset is an example of a dataset that meets these requirements. It consists of 5000 stereo camera snapshots with a resolution of 2048×1024 pixels and dense pixel annotations. These labels comprise ground truth for semantic and instance segmentation as well as object detection. The proposed augmentation pipeline renders the *Augmented Cityscapes* dataset, which also incorporates 2975 images of 2048×1024 pixels with one to five augmented pedestrians. Each image is accompanied by generated semantic and instance maps, and the dataset follows the standard Cityscapes annotation format for classes and categories.

A multi-discriminator network is employed to adjust the appearance of the augmented objects to make them more realistic. The model is trained using the augmented images as the source domain and real Cityscapes images as the target domain, and the training process runs for 200 epochs. The training runs without initialization of the network with a cyclic loss weight of 10 and cost-sensitive loss parameter λ of 0.2. Similar to the work of Zhu et al. [10], the initial learning rate is 0.0002, remaining fixed for the first 100 epochs before decreasing to 0 over the subsequent 100 epochs. The experiments used images downsized by $\times 2$ without random crops.

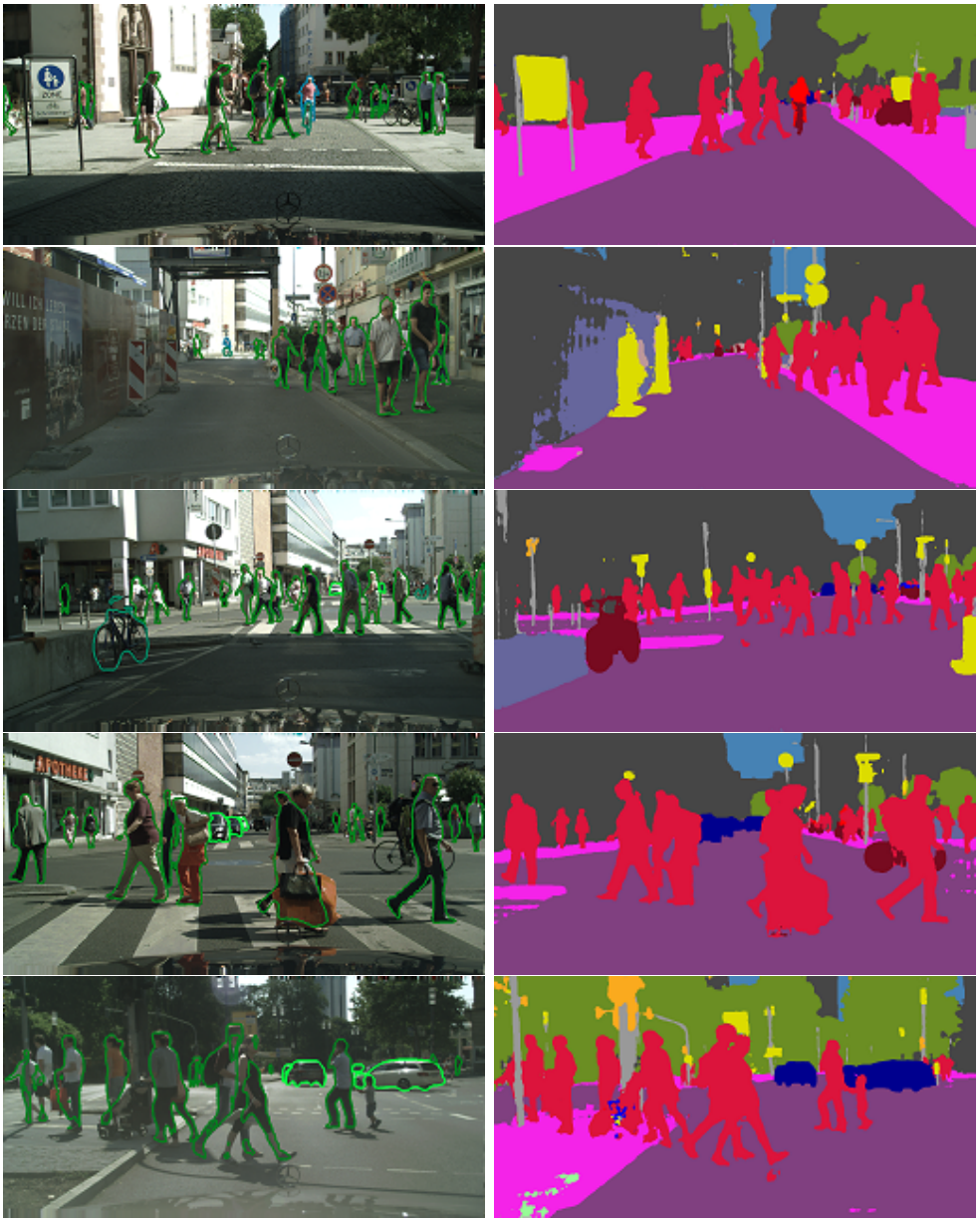
Figure 4.6 illustrates the images produced by the proposed framework. It visualizes the effects introduced via the adaptation approach by comparing both augmented and translated images. The impact of the appearance learning part is demonstrated by magnifying the parts of the image where they are most apparent. It is worth noting that the proposed masked discriminator architecture effectively alleviates inconsistent domain translation evident in vanishing pedestrians. Synthesized samples did not exhibit inconsistencies, and augmented virtual pedestrians remained in the image after translation. Another characteristic of the translated images is the appearance of the rendered objects, which follow the color scheme of the target dataset. Finally, the translated images incorporate realistic illumination extracted from the real data, as highlighted by the magnified segments of the images, which accentuate applied light spots, shades, and soft object borders.

The results of the experiments are additionally evaluated quantitatively by employing the obtained images in semantic and instance segmentation tasks. For this purpose, Deeplabv3 [174] and Mask-RCNN [175] models were trained with augmented images.

Instance Segmentation. The quality of the image transfer is assessed using the standard COCO average precision (AP) metric [176] in the instance segmentation task. During the experiment, it deploys a detection model Mask-RCNN pre-trained on the COCO dataset and fine-tuned on the introduced *Augmented Cityscapes* dataset. Table 4.1 reports the results of the instance segmentation model tested on 500 images from the Cityscapes *val*.

Semantic Segmentation. The effectiveness of the generated data for the semantic segmentation task is evaluated using the Deeplabv3 model. Similar to the previous evaluation, the segmentation model is trained on the Cityscapes training set and the augmented data. Figure 4.7 shows the model's results tested on 500 images from the Cityscapes validation set. In both experiments, the images are downsized to the resolution of 1024×512. Deeplabv3 model uses *xception65* backbone and is trained for $90,000$ iterations with 16 random crops of size 513×512 pixels in every iteration. The learning rate for this model is kept at 0.007. Table 4.1 reports mean IoU metric for the best-performing snapshots.

In these experiments, the generated data achieves comparable performance in overall metrics such as average precision (AP) but exhibits better scores for the class of interest - person. Specifically, the improvement in the AP_{person} metric in Mask-RCNN experiments is 2.6. In contrast, the Deeplab only showed a relatively small improvement in the pedestrian class, with an increase of less than 0.5%. The pixel accuracy and mean IoU for all 19 classes show a decrease of 0.3%, which is relatively insignificant. These results suggest that the proposed method for data generation does not introduce a domain gap between the generated and real data.



Instance segmentation Semantic segmentation

Figure 4.7: Prediction results on Cityscapes validation set, for instance, and semantic segmentation obtained by Mask-RCNN and Deeplabv3 models, respectively, trained on proposed augmented training images. © 2020 IEEE

4.5 Discussion

As previously discussed in Chapter 1, deep models for object detection in traffic scenes trained with only synthetic data face the challenges of the sim-to-real

Method	AP_{avg}	$AP50_{avg}$	AP_{person}	$AP50_{person}$	Accuracy	mean IoU	preson
Cityscapes	31.8	59.0	33.0	67.7	95.6	75.6	77.1
Augmented	32.6	60.4	35.6	74.2	95.3	75.3	77.3

Table 4.1: Prediction results obtained on Cityscapes validation set for instance segmentation (AP) obtained by Mask-RCNN and semantic segmentation (IoU) obtained by Deeplabv3, when trained on Cityscapes train set (top) and augmented images (bottom) and generated by proposed augmentation method.

domain gap. This gap leads to the suboptimal performance of named detectors in real-world scenarios. To address the decreasing performance, several works, such as [137], proposed to simplify the sim-to-real setting by augmenting real images with virtual objects instead of using pure rendered images. Inspired by the augmentation strategy of Alhaija et al. [137], the work described in this chapter proposes a pipeline for geometrically correct traffic scene augmentation with 3D pedestrians. The proposed method relies on spatial information obtained from the available point cloud to ensure geometrical correctness and avoid collisions among the objects. The experiments conducted for evaluation utilized the depth maps derived from available disparity maps.

Furthermore, the approach extends the augmentation with an adversarial appearance learning to make in-painted objects visually indistinguishable from the background scene. Adversarial learning of visual characteristics leads to the phenomenon of *vanishing pedestrians* motivated by the out-of-distribution kind of renderings. The proposed approach further extends the augmentation pipeline with novel masked multi-discriminators to learn the style of real pedestrians without removing rendered ones from the image.

Experiments demonstrated that the proposed pipeline can augment real traffic images with synthetic pedestrians that resemble the reference dataset's real pedestrians. Additional experiments on instance segmentation and pedestrian detection confirmed the decrease of the domain gap between augmented and original data. Further chapters look closer into the problem of the sim-to-real domain gap and the causes of semantic inconsistencies emerging during the adversarial domain adaptation.

5 Density Matching

5.1 Class Balance

The Cityscapes dataset is a large-scale dataset for computer vision tasks for urban traffic scenes. It comprises 2975 train images and 500 validation images and pixel-wise annotations for semantic and instance segmentation. These annotations include 19 classes: road, building, sky, sidewalk, vegetation, car, terrain, wall, truck, pole, fence, bus, person, traffic light, traffic sign, train, motorcycle, rider, bicycle. The *Playing for Data* is a synthetic dataset extracted from a video game comprising $24,966$ images. Every sample is provided with a dense semantic map annotated in a way that conforms with the Cityscapes dataset. The KITTI is also a real large-scale dataset, which supports ground truth for object detection in addition to the semantic segmentation task. Initially, it provides a limited number of semantic maps limited to 200 examples for the train and 200 for the test. Having the size of 1024×256, they are compatible with the Cityscapes dataset [137]. However, additional annotations from Kreso et al. [177] and Ros et al. [178] have been provided by some researchers, increasing the size of the dataset to 600 samples. The SYNTHIA dataset is a different synthetic dataset used for the experiments. It comprises $9,400$ frames created in a virtual environment and $9,400$ annotations consistent with the Cityscapes dataset.

Figure 5.1 illustrates the class distribution in the synthetic and real datasets regarding four main classes. The histogram shows that the class distribution in Cityscapes and *Playing for Data* datasets reveals a strong imbalance concerning certain classes, particularly sky and vegetation. The PfD dataset has almost three times more instances of the sky class than the Cityscapes dataset. Additionally, the vegetation class is two times stronger underrepresented in PfD than in Cityscapes. Another histogram shows class distribution differences between KITTI and PfD images. The KITTI dataset has a higher proportion of the vegetation class compared to the PfD dataset, which has more pixels corresponding to the building class. The third histogram compares the Cityscapes and SYNTHIA class distributions. The SYNTHIA dataset contrasts with other presented datasets in capturing the images from a surveillance viewpoint contrary to the ego car viewpoint. Viewpoint discrepancy is also reflected in the class distribution histogram provided in Figure 5.1. The comparison shows road and sidewalk are two imbalanced classes.

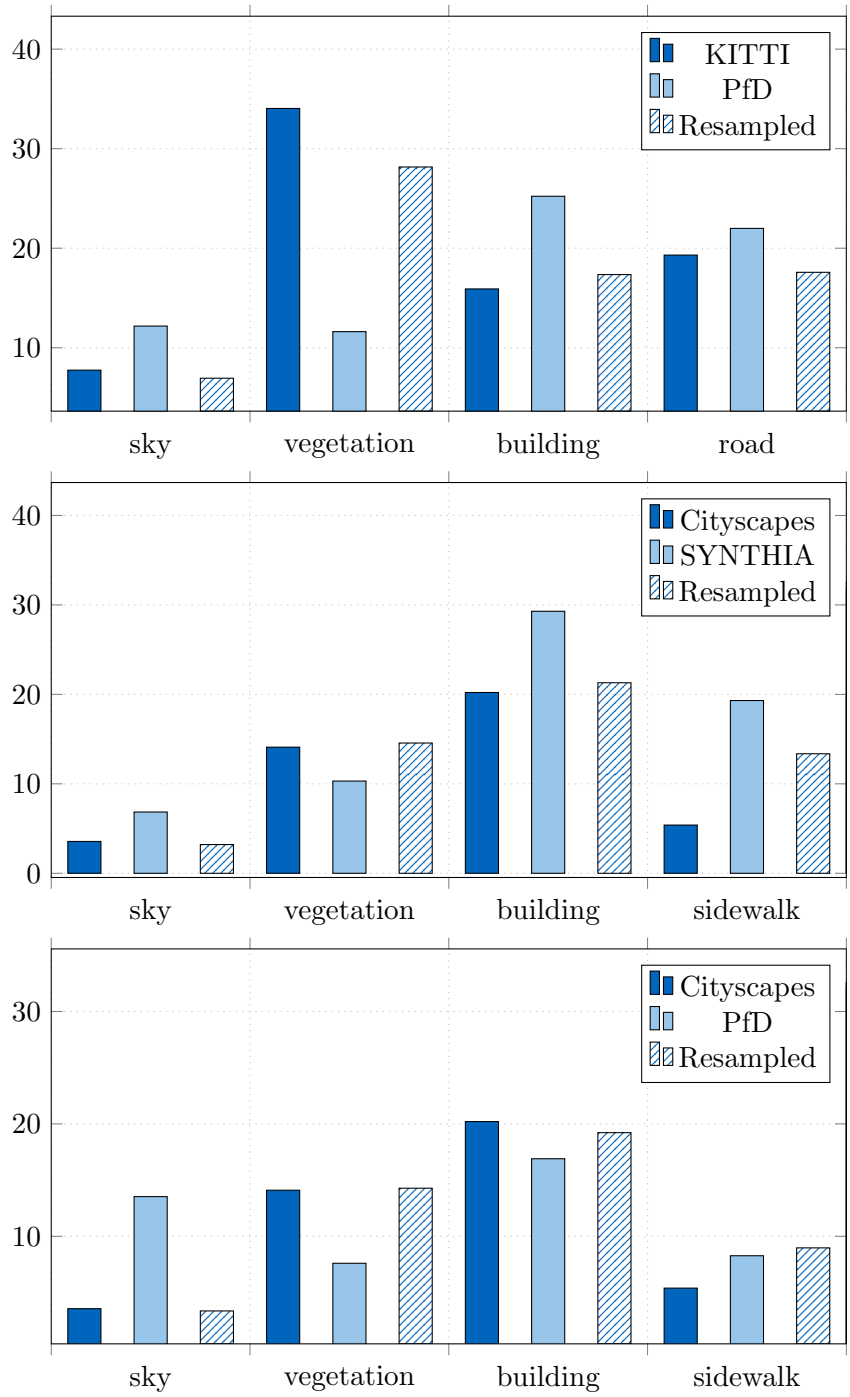


Figure 5.1: Histograms visualizing class balance statistics observed in real, original synthetic, and re-sampled synthetic datasets for the specific classes sky, vegetation, building, sidewalk, and road.

5.2 Importance Resampling

The hypothesis is that semantically mismatching artefacts in Figure 1.2 are related to the class imbalance observed in Figure 5.1. Under this hypothesis, it is plausible to approach the class inconsistency by resampling the synthetic samples so that the resulting distribution of the source samples P_s and the target distribution P_t are approximately equal. The Sampling Importance Resampling (SIR) rebalances the data in two phases [179]: first, it draws L samples $\{x_i^s\}_{i=1}^L$ from $P_s(x)$, then it calculates the weights of the drawn samples ω_i^s, known as *importance weights*. They can be obtained from the labels as follows:

$$\omega_i = \frac{p_t(y_i^s)}{p_s(y_i^s)} \tag{5.1}$$

Finally, a subset of samples $\{x_i^s\}_{i=1}^l$ is drawn from $\{x_i^s\}_{i=1}^L$, where $l < L$, with probabilities given by the importance weights ω_i^s. Importance weights are dedicated to remedying the bias of source data distribution. Given the source labels $\{y_i^s\}_{i=1}^{N_s}$, the probability distribution $p_s(y)$ in our task can be estimated in a relatively simple manner [41]:

$$p_s(y_i^s) = \sum_c \frac{N_{y_i^s}^c / N_{y_i^s}}{N_s^c / (N_{y_i^s} N_s)} \tag{5.2}$$

In Equation 5.2, $N_{y_i^s}^c$ represents the number of pixels belonging to class c and $N_{y_i^s}$ is the number of all pixels in a specific sample y_i^s. N_s represents the number of all source samples, $c \in \{road, building, etc\}$ are the label classes, and $N_s^c = \sum_i N_{y_i^s}^c$ denotes the number of class c pixels in the source dataset. For the target distribution P_t it can be obtained by similar calculation where $N_t^c = \sum_j N_{y_j^t}^c$:

$$p_t(y_i^s) = \sum_c \frac{N_{y_i^s}^c / N_{y_i^s}}{N_t^c / (N_{y_i^s} N_t)} \tag{5.3}$$

Therefore, given labels $\{y_i^s\}_{i=1}^{N_s}$ and $\{y_j^t\}_{j=1}^{N_t}$, the weights ω_i can be determined this way:

$$\omega_i = \frac{N_t}{N_s} \frac{\sum_c N_{y_i^s}^c / N_t^c}{\sum_c N_{y_i^s}^c / N_s^c} \tag{5.4}$$

Sampling Importance Resampling (SIR) algorithm states that as $L/l \to \infty$, the samples $\{x_i^s\}_{i=1}^l$ will be approximately distributed according to P_t. It is possible to adjust the synthetic dataset by resampling it based on the naively approximated class balance in the source and target domains. The estimate can be done using the probabilities given by ω_i.

The resulting class distributions for the resampled synthetic datasets show favourable agreement with the corresponding real data, at least for the four

Figure 5.2: Examples of synthetic images translated by the proposed method and baseline methods from PfD to Cityscapes domain. © 2019 IEEE

main classes. These equalized class distributions can be seen in Figure 5.1. The only exception is the sidewalk class in the Cityscapes-SYNTHIA setting, which presumably regards the significantly different viewpoint in the SYNTHIA dataset which prevents rebalancing.

5.3 Experiments

An established translation method CycleGAN [10] was trained with the resampled data to evaluate the impact of the importance sampling strategy on the domain adaptation process. In line with the original work, the model's training continued 200 epochs with resampled images as the source data and real images as the target data and then translated images were evaluated qualitatively and quantitatively. Preprocessing step downsampled the training images to a resolution of 1024×512 pixels in the experiment with Cityscapes. The KITTI experiment used the downsampling to 1024×256 pixels. In both settings, the downsizing was performed due to training hardware limitations. The experiments fall in three settings: the adaptation of PfD to Cityscapes, PfD to KITTI,

PfD CycleGAN Resampling

Figure 5.3: Examples of images translated by the proposed method and baseline methods from PfD to KITTI domain. © 2019 IEEE

and the transformation of SYNTHIA to PfD. The domain adaptation strategy was evaluated visually and through its performance on a downstream semantic segmentation task and compared to recent sim-to-real adaptation networks.

Qualitative Evaluation. The qualitative results are presented in Figures 5.2 and 5.3. In Figure 5.2, it can be seen that the original translation results confuse the patches of the classes vegetation and sky. However, by using resampling, the domain adaptation model can better maintain the image's semantics. Figure 5.3 compares samples generated in the PfD-KITTI experiment by CycleGAN trained on the original and resampled data. Resampling helps to prevent the translation model from introducing unwanted artefacts, such as vegetation patches on buildings and roads contrary to the results without resampling, where the model confuses imbalanced classes and produces mismatch errors.

Segmentation. Two segmentation models DRN-C-26 [173], and Deeplabv3 [174] were used in the downstream task evaluation experiment. The DRN network was first initialized with weights pre-trained on the ImageNet dataset [28] and then fine-tuned with random crops of 600×600 of the translated PfD to Cityscapes data for 250 epochs, using the momentum of 0.99 and a learning rate of 0.001 that decreased by a factor of 10 every 100 training steps. For Deeplab, training used a learning rate of 0.007 and crops of 513×513. Identical settings, including the hyper-parameters for both the DRN and Deeplab in all experimental settings, guaranteed a fair comparison with baselines. Benchmark translation methods included CycleGAN [10], CyCADA [47] and UNIT [51]. Figure 5.4 demonstrates several examples of semantic maps predicted using the benchmark data and proposed approach.

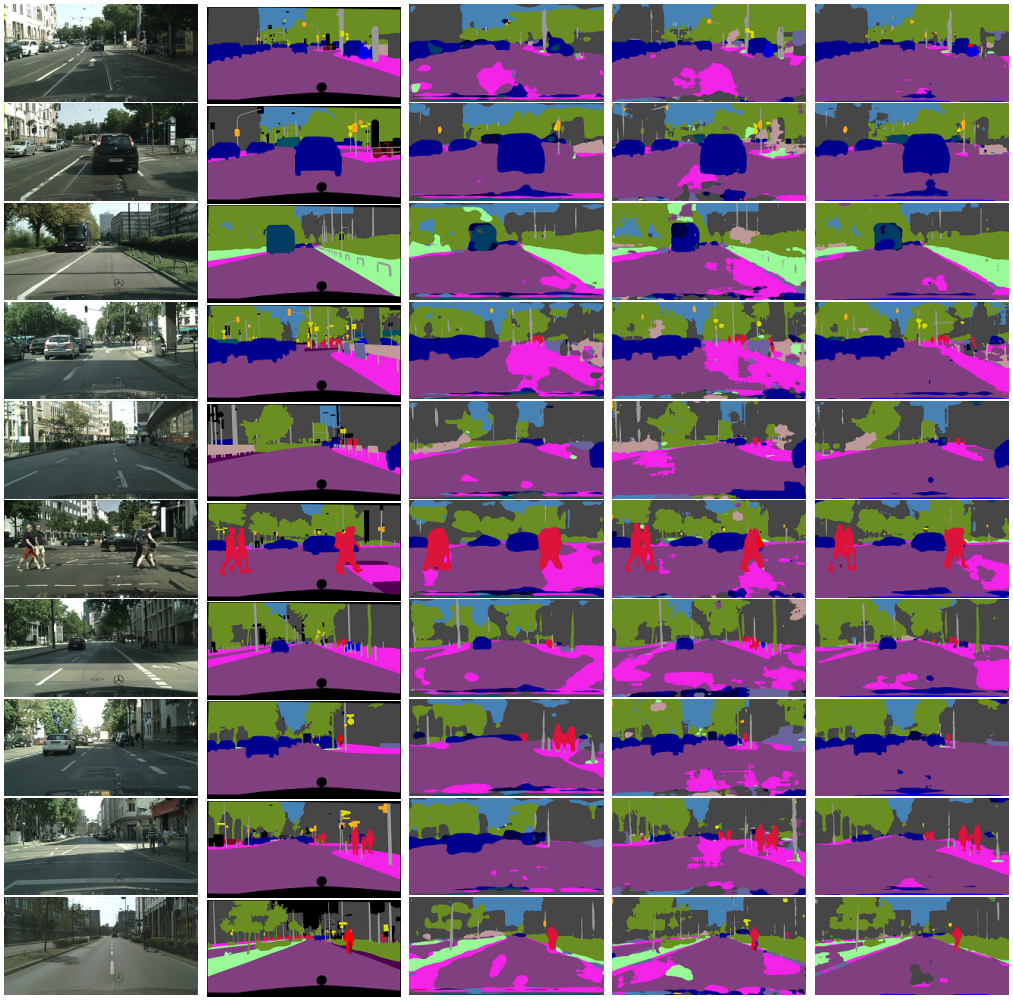


Figure 5.4: Semantic segmentation results obtained on Cityscapes validation set performed by DRN method, which was trained on the synthetic images translated by the proposed and baseline methods. © 2019 IEEE

The evaluation used the Jaccard index, also known as the intersection over union (IoU) metric, which for any specific class represents the ratio of correct prediction to the sum of true positive, false positive, and false negative predictions [180]. In addition, the mean value of the Jaccard index overall 19 classes was reported. Tables 5.1 and 5.2 present the results obtained in the experiments. The performance values for UNIT originate from the work of Dundar et al. [40].

In addition, Tables 5.1 and 5.2 show DRN and Deeplab results trained on the dataset obtained by translating PfD images to Cityscapes, as measured on the Cityscapes validation dataset. One can see that resampling demonstrates higher mean IoU performance compared to benchmark methods for both DRN

Trans. method	Train dataset	Evaluation	Accuracy	mean IoU	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorbike	bicycle
None	CS	CS	94.3	67.4	97.3	79.8	88.6	32.5	48.2	46.3	63.6	73.3	89.0	58.9	93.0	78.2	55.2	92.2	45.0	67.3	39.6	49.9	73.6
None	PfD	CS	62.5	21.7	42.7	26.3	51.7	5.5	6.8	13.8	23.6	6.9	75.5	11.5	36.8	49.3	0.9	46.7	3.4	5.0	0.0	5.0	1.4
CycleGAN	PfD → CS	CS	82.5	32.4	81.8	34.7	73.5	22.5	8.7	25.4	21.1	13.5	71.5	26.5	41.7	50.1	7.3	78.5	20.5	19.5	0.0	12.5	6.9
CyCADA	PfD → CS	CS	82.3	39.5	79.1	33.1	77.9	23.4	17.3	32.1	33.3	31.8	81.5	26.7	69.0	62.8	14.7	74.5	20.9	25.6	6.9	18.8	20.4
UNIT	PfD → CS	CS	87.1	39.1	90.5	38.5	81.1	23.5	16.3	30.2	25.2	18.5	79.5	26.8	77.8	59.2	17.4	84.4	22.2	16.1	1.6	16.7	16.9
Proposed	PfD → CS	CS	**88.0**	**39.7**	88.0	38.9	81.1	23.6	13.0	31.8	32.5	26.2	79.3	26.7	74.5	54.6	25.1	80.9	20.5	23.5	0.8	16.8	16.2
CycleGAN	PfD → KITTI	KITTI	-	19.6	68.0	16.9	43.8	5.0	8.0	24.0	15.8	2.7	47.5	16.2	13.8	30.4	0.0	65.6	0.0	0.0	15.5	0.0	0.0
Proposed	PfD → KITTI	KITTI	-	**21.8**	39.7	17.1	38.1	8.5	15.0	22.0	19.3	16.5	55.1	12.3	69.5	25.0	1.9	55.8	0.0	0.0	17.8	0.0	0.0
CycleGAN	SYNTHIA → CS	CS	-	24.4	54.3	21.2	67.6	3.8	0.1	26.7	3.5	4.0	61.0	0.0	46.8	50.0	16.0	64.2	0.0	18.0	0.0	6.9	19.9
Proposed	SYNTHIA → CS	CS	-	**25.4**	69.1	26.0	64.2	0.6	0.1	27.8	5.2	4.0	71.7	0.0	54.3	48.5	11.2	57.4	0.0	11.3	0.0	4.4	26.5

Table 5.1: Semantic segmentation results (IoU) on Cityscapes validation set obtained by DRN model, which was trained on synthetic images translated by the proposed sampling method and baseline methods. Translated dataset is denoted as Source → Target and CS denotes Cityscapes dataset. © 2019 IEEE

Trans. method	Train dataset	Evaluation	Accuracy	mean IoU	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorbike	bicycle
None	CS	CS	95.5	75.6	97.9	83.5	91.6	56.5	61.2	54.8	63.9	73.6	91.3	59.9	93.2	77.7	60.1	94.0	79.3	87.0	76.1	61.0	73.2
None	PfD	CS	82.9	40.0	79.2	26.9	79.5	19.1	27.4	29.4	34.9	17.9	81.8	25.3	71.9	60.8	22.1	81.5	34.1	17.0	2.5	21.3	26.9
CycleGAN	PfD → CS	CS	88.4	48.2	89.9	45.4	85.6	38.3	32.9	36.1	45.1	27.6	83.7	28.9	83.4	65.8	32.0	86.2	30.0	32.1	1.8	33.5	37.2
UNIT	PfD → CS	CS	86.0	47.7	87.6	39.4	85.2	38.2	35.1	35.3	42.3	33.4	76.0	21.0	69.4	63.7	29.3	85.4	34.1	31.4	28.5	31.5	39.8
Proposed	PfD → CS	CS	**89.7**	**54.1**	89.4	40.9	88.0	45.7	37.5	44.8	50.3	44.2	87.6	36.6	87.3	66.8	31.3	88.6	40.0	57.7	24.7	39.0	27.9

Table 5.2: Semantic segmentation results (IoU) on Cityscapes validation set obtained by Deeplab model, which was trained on synthetic images translated by the proposed sampling method and baseline methods. Translated dataset is denoted as Source → Target and CS denotes Cityscapes dataset.

and Deeplabv3 without introducing additional semantic-preserving constraints and losses.

Additionally, the proposed approach was compared to CycleGAN in the PfD-KITTI setting. In this experiment, the performance of the DRN network trained on 25,000 PfD images translated to KITTI was evaluated using the validation subset of the initial KITTI semantic segmentation data. It should be noted that, despite having relatively moderate mean IoU in this experiment, resampling helped to achieve performance improvement compared to baseline by over 2 per cent. However, the resampling of SYNTHIA-Cityscapes data demonstrated a significant decrease in segmentation performance. Such performance is likely due to the different sensor setups in the synthetic and real datasets, which makes class balancing across all five imbalanced classes non-trivial.

5.4 Adversarial Translation with Importance Estimation

As introduced in Section 5.2 the setting consists of synthetic tuples of images x and labels y: $\{(x_i^s, y_i^s)\}_{i=1}^{N_s}$ from the synthetic domain D_s and real tuples $\{(x_j^r, y_j^r)\}_{j=1}^{N_r}$ from the real domain D_r:

$$\begin{aligned} D_s &= \{x_i^s\}_{i=1}^{N_s} \\ D_r &= \{x_j^r\}_{j=1}^{N_r} \end{aligned} \tag{5.5}$$

Consider a variable x in the input space $\mathcal{X}$ that takes on values x_i^s, which are *iid* according to the probability distribution $P_s(x)$:

$$\begin{aligned} &x_i^s \in \mathcal{X}_s \subset \mathcal{X}, i = 0, 1, ..., N_s \\ &\{x_i^s\}_{i=1}^{N_s} \sim P_s(x) \end{aligned} \tag{5.6}$$

In contrast, the samples x_j^r are distributed according to P_r:

$$\begin{aligned} &x_i^r \in \mathcal{X}_r \subset \mathcal{X}, i = 0, 1, ..., N_r \\ &\{x_j^r\}_{j=1}^{N_r} \sim P_r(x) \end{aligned} \tag{5.7}$$

Samples of synthetic and real datasets are profoundly dissimilar, so the marginal distributions of these samples $\{x_i^s\}$ and $\{x_j^t\}$ are not equal $P_s(x) \neq P_r(x)$. On the other hand, the $P(y|x)$ conditional distribution of labels y given x is unchanged. This scenario is known as covariate shift [41], which is generally addressed by domain adaptation methods.

As discussed in Section 1.3, a number of methods related to sim-to-real domain adaptation aim to find a mapping function $g : \mathcal{X}_s \to \mathcal{X}_r$ that translates samples from the synthetic space into the real. Recent works generally use neural networks in order to approximate this mapping function for high-dimensional image

Original synthetic Translated

Figure 5.5: Semantically inconsistent sim-to-real adaptation performed by the adversarial network by translating *Playing for Data* and SYNTHIA images to the Cityscapes domain. © 2020 IEEE

spaces. A large number of these neural networks adopt a renowned generative adversarial network (GAN) framework [8] and are called generators. Training of such an adversarial model, as described in Section 2.4.1, involves another network, known as the discriminator, which is given input samples from the reference distribution P_r and from the generated distribution $P(g(x^s))$. During the training, two networks are involved in a zero-sum game, where the discriminator learns to classify synthesized and reference samples, and the generator learns to synthesize samples that reduce classification to random guessing. The mini-max game converges ideally to Nash equilibrium, where the samples synthesized by the

generator g are distributed similarly to reference data such that $P(g(x^s)) \sim P_r$. The use of adversarial loss effectively performs perturbations in the images at a low level so that the resulting images resemble the visual properties of reference samples. Such perturbations imposed on the images by the generator are driven by the fact that the discriminator learns certain regularities distinct for the reference data. These perturbations make the generated images appear similar to the target ones but introduce inconsistency between high-level image content and the corresponding semantic layout as demonstrated in Figure 5.5.

5.4.1 Importance Function

The proposed method alleviates the sim-to-real bias and thereby decreases class inconsistencies in image transfer. The *importance weighting* procedure is a technique, which is effective in mitigating the covariate shift in machine learning. The central principle behind this technique involves a notion of *importance* of the training samples and consists of identifying and prioritizing the most informative samples in compliance to that importance. Indeed, the importance value of a particular sample can be defined by applying an importance function, provided the density functions of both distributions:

$$\omega(x) = \frac{p_r(x)}{p_s(x)} \tag{5.8}$$

5.4.2 Density Ratio Estimation

In an unsupervised sim-to-real setting estimating the densities for both distributions is reasonably difficult. However, as seen from Equation 5.8, it is possible to circumvent the estimation of individual densities by directly estimating their *density ratio*. To this end, the Kullback–Leibler Importance Estimation Procedure (KLIEP) introduced by Sugiyama et al. [65] is a compelling technique to address the task. By modeling the importance function $\omega(x)$, this method directly estimates the ratio of both distributions' densities instead of estimating the densities independently.

$$\hat{\omega}(x) = \sum_l \alpha_l \varphi_l(y), \tag{5.9}$$

In Equation 5.9 the parameters α_l are learned from the source samples y_i^s and target samples y_j^t and $\varphi_l(y)$ represent basis functions. The model $\hat{\omega}(x)$ approximates the density $\hat{p}_t(x) = \hat{\omega}(x)p_s(x)$ so that parameters α_l of the model minimize the Kullback-Leibler divergence between $p_t(x)$ and $\hat{p}_t(x)$.

$$\begin{aligned} KL(p_t||\hat{p}_t) &= \mathop{\mathbb{E}}_{x^t}\big[\log \frac{p_t(x)}{\hat{\omega}(x)p_s(x)}\big] \\ &= \mathop{\mathbb{E}}_{x^t}\big[\log \frac{p_t(x)}{p_s(x)}\big] - \mathop{\mathbb{E}}_{x^t}[\log \hat{\omega}(x)] \end{aligned} \tag{5.10}$$

Since $\mathbb{E}_{x^t}\left[\log \frac{p_t(x)}{p_s(x)}\right]$ does not depend on α let us only consider $\mathbb{E}_{x^t}[\log \hat{\omega}(x)]$:

$$\mathop{\mathbb{E}}_{x^t}[\log \hat{\omega}(x)] = \frac{1}{N_t} \sum_{j}^{N_t} \log \sum_{l} \alpha_l \phi_l(y_j^t) \tag{5.11}$$

Therefore, since $\hat{p_t}(x)$ is a probability density function, maximizing the 5.11 over α under the following constraint will minimize the Kullback-Leibler divergence:

$$\mathop{\mathbb{E}}_{x^s}[\hat{\omega}(x)] = \frac{1}{N_s} \sum_{i}^{N_s} \sum_{l} \alpha_l \phi_l(y_i^s) = 1, \tag{5.12}$$

Hence, the optimization problem is defined as follows:

$$\begin{aligned}
\underset{\alpha_l}{\text{maximize}} \quad & \sum_{j} \log \sum_{l} \alpha_l \varphi(y_j^t) && \text{(5.13a)} \\
\text{subject to} \quad & \sum_{l} \alpha_l \sum_{i} \varphi_l(y_i^s) = N_s, && \text{(5.13b)} \\
& \alpha \geq 0. && \text{(5.13c)}
\end{aligned}$$

Optimization employs the radial basis function (RBF) kernel K_{σ_t} centered at y_j^t and with width σ_t defined by grid search maximizing Equation 5.11. A similar optimization problem can be defined for the reversed direction of adaptation:

$$\begin{aligned}
\hat{\omega}(x^s) &= \sum_{l} \alpha_l K_{\sigma_r}(y^s, y_l^t), \\
\hat{\psi}(x^t) &= \sum_{k} \beta_k K_{\sigma_s}(y^t, y_k^s)
\end{aligned} \tag{5.14}$$

Finally, pre-matching the densities of marginal distributions employs the gradient ascent with constraint satisfaction to find $\hat{\omega}(x^s)$ and $\hat{\psi}(x^t)$.

5.4.3 Weighted Loss

The underlying image transfer method is based on an adversarial approach described in Section 5.2, which also utilizes a cycle-consistency loss [10]. Therefore, the final objective combines importance weights with both adversarial losses for both $g_r : \mathcal{X}_s \to \mathcal{X}_r$ and $g_s : \mathcal{X}_r \to \mathcal{X}_s$, and cyclic-consistency losses:

$$
\begin{aligned}
\mathcal{L} &= \mathcal{L}_{IWAdv} + \mathcal{L}_{IWCyc} \\
&= \mathop{\mathbb{E}}_{x^r}[\psi(y^r) \log d_r(x^r)] \\
&+ \mathop{\mathbb{E}}_{x^s}[\omega(y^s) \log(1 - d_r(g_r(x^s))] \\
&+ \mathop{\mathbb{E}}_{x^s}[\omega(y^s) \log d_s(x^s)] \\
&+ \mathop{\mathbb{E}}_{x^r}[\psi(y^r) \log(1 - d_s(g_s(x^r))] \\
&+ \mathop{\mathbb{E}}_{x^r}[\psi(y^r) \| g_r(g_s(x^r)) - x^r \|] \\
&+ \mathop{\mathbb{E}}_{x^s}[\omega(y^s) \| g_s(g_r(x^s)) - x^s \|]
\end{aligned}
\tag{5.15}
$$

5.5 Experiments

Evaluation of the importance weighting approach employs two experimental settings: a toy example and traffic data. In the first experiment, Gaussian and uniform samplers produce source and target distributions. In the second setup, experiments are conducted in the traffic environment using large-scale datasets in synthetic and real traffic scene scenarios.

5.5.1 Toy Example

Domain adaptation problem in the *toy example* setting is simulated by sampling one dataset with $10,000$ random vectors of size 300 per domain. The target domain is distributed normally with a mean value of 7.0 and a standard deviation of 0.5. The goal is to find a mapping function approximated by GAN model, which translates the source samples distributed uniformly in the segment $[0, 10)$ into samples so that their distribution and the target distributions are equal. Figure 5.6 shows the histograms of both distributions.

Toy experiment uses an original generative adversarial network as a baseline. The baseline goal is, similarly, to translate the source samples into a closer representation of the target distribution. Both models are trained on simulated datasets for 40 epochs with batches measuring 200 vectors. The original underlying GAN model comprises several activation functions and scaled exponential linear units [181] in the generator part and additional sigmoid activation in the discriminator part. A stochastic gradient descent optimizer for the generator uses a learning rate of $8e-3$ and the one for the discriminator $4e-3$. The proposed method extends the previously defined original GAN with the KLIEP-based importance loss according to Equation 5.16. Training of the extended model follows the same experimental setup.

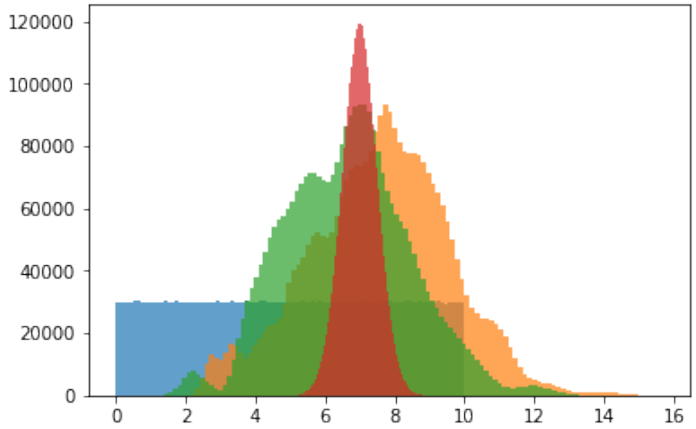


Figure 5.6: Histograms for source (blue), target (red) distributions along with distributions generated by original GAN (orange), and by proposed KLIEP GAN (green). © 2020 IEEE

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{IW\,Adv} &= \mathop{\mathbb{E}}_{x^s}[\log d(x^t)] \\ &+ \mathop{\mathbb{E}}_{x^s}[\sum_l \alpha_l K_{\sigma_t}(x^s, x_l^t) \log(1 - d(g(x^s))]\end{aligned} \tag{5.16}$$

After training, $10,000$ vectors are inferred from uniformly distributed vectors using the original GAN and importance weighted GAN. Figure 5.6 illustrates the resulting distributions. Additionally, Table 5.3 evaluates both generated distributions by presenting the moments and distances measured to the target distribution. The calculations of the distances rely on the *Wasserstein distance* and *Energy distances* metrics.

The results of the experiments with *toy data* show that distribution prematching using the density ratio estimation increases the similarity of the target and adversarially generated distribution. Similarity improvement is indicated by both moments as well as decreasing the Wasserstein distance by 20% and energy distances by 15%.

5.5.2 Traffic Data

The large-scale evaluation experiment follows a similar process to the toy example, comprising two stages. The first stage of this process involves learning the adaptation model to translate source images into the target domain. The network then produces a realistic synthetic dataset inferred from source images. Next, several segmentation networks use the translated dataset for training. The assessment of the sim-to-real translation quality of the images succeeds by evaluating the segmentation accuracy on the real data.

Distribution	μ	σ	Wasserstein distance	Energy distance
Target (Gauss)	7.0	0.5	-	-
Source (uniform)	5.0	2.9	2.56	1.39
Vanilla GAN	7.7	2.0	1.32	0.79
Instance weighted	6.7	1.8	1.08	0.67

Table 5.3: Distances between generated and target distributions (less is better).

The experimental setting with automotive data relies on two large-scale datasets: Cityscapes [29] and *Playing for Data* (PfD) [4]. The Cityscapes dataset represents the real or target domain and comprises 5000 frames of urban traffic scenes with resolution 2048×1024 pixels together with fine-annotated dense semantic maps. These samples include 50 cities captured at various times of the year under different weather and lighting conditions. The Cityscapes dataset provides ground truth for semantic, instance, and panoptic segmentation, covering 30 classes and annotating single instances of dynamic objects of class car, person, and rider. The evaluation of the proposed method focuses on 19 commonly used classes such as road, building, sky, sidewalk, vegetation, car, person, and others. On the other hand, the PfD dataset contains approximately 25000 frames with a resolution of 1914×1052, originated from a computer game engine and annotated with semantic labels. While the PfD dataset exhibits a large degree of high-level realism in terms of traffic scene composition, including a wide range of scenery, scenarios, and appearances, it also contains labelling inconsistencies. Nonetheless, the PfD dataset remains a popular choice for synthetic data among researchers in the field of autonomous driving.

Figure 5.7 shows the results of domain adaptation on the Cityscapes and PfD images performed by the proposed method and the benchmarks. According to the presented samples, the benchmarks introduce several mismatching patches in the samples during the translation. These inconsistent patches occur in several classes, mainly imbalanced vegetation and sky. On the other hand, the proposed density ratio pre-matching helps the translation model maintains semantic consistency, resulting in a more accurate translation of the source data.

Also, the quality of domain adaptation is quantitatively assessed on the downstream semantic segmentation task. For that, state-of-the-art segmentation models *Dilated Residual Network* (DRN) [173] and Deeplabv3 [149] are trained on the translated samples produced by the proposed method and benchmarks and evaluated on the Cityscapes validation dataset. It is important to note that the segmentation model had no access to the target training dataset. The experimental setup follows the protocol of the original works [173] and [149] for the quantitative evaluation. Specifically, preprocessing downsized the images to the

Figure 5.7: Examples of synthetic images translated by the density matching method and baselins from PfD to Cityscapes domain.

resolution of 1024×512. Additionally, DRN used pre-trained on the Imagenet [28] weights as initialization. After that, the model was trained for 200 epochs on randomly cropped patches of translated images of size 600×600. The entire training has a momentum of 0.99 and starting learning rate of 0.001, which decreases by a factor of 10 every 100 iteration. The Deeplabv3 model was trained for 90000 steps using a batch size of 16 patches of size 513×513. In this case, the optimizer had a learning rate of 0.007. The backbone used for the Deeplab model was the *xception65* network. Table 5.4 shows obtained segmentation performance. Additionally, the table shows the results for the previous version of Deeplab - v2 [174].

The evaluation uses intersection over union (IoU) for a particular class, also known as *Jaccard Index*, as a primary metric in all downstream experiments. IoU computes correctly predicted pixels as a fraction of the true positives, false positives, and false negatives summed[180]. The ratio enables evaluation that is agnostic to the class size. Table 5.4 shows the results obtained for both segmentation networks DRN and Deeplab trained on datasets generated by respective adaptation methods. Additionally, the table reports the mean IoU value for 19 classes and overall pixel accuracy. Furthermore, the table compares the results of the proposed approach to lower and upper bounds obtained by training the segmentation models on real and synthetic data only.

The findings in Table 5.4 show that pre-matching densities using the Kullback-Leibler Importance Estimation Procedure (KLIEP) improves the average as well as class-specific segmentation performance for classes such as road, building, vegetation, sky, and car. For example, the sky class increased segmentation by 7%, while the remaining classes saw about 2% improvement. In the Deeplab experiments, CyCADA achieved the highest mean IoU, but density pre-matching demonstrated better results for certain classes, including building, vegetation, sky, truck, and bus.

Figures 5.8 and 5.9 exemplify the segmentation results for DRN and Deeplab, respectively. Demonstrated semantic maps confirm that class-consistent image translation leads to improved class consistency in semantic prediction.

5.5.3 Ablation Study

Apart from the traffic scene experiments, a supplementary ablation study was conducted on traffic scene images. The goal of the study is to investigate the contribution of the importance estimation strategy to the performance of the proposed sim-to-real domain adaptation method. In this experiment, the source dataset PfD is divided into three proportionate groups according to the importance values of the samples. Every cohort includes 8322 samples and denotes a distinct importance range: low, medium, and high. The experiment closely recreates the evaluation protocol discussed in Section 5.5.2, with a few modifications to accelerate the process. Firstly, all samples are downsized to 512×256 pixels. After that, three instances of the proposed translation model are trained

Method	Accuracy	mean IoU	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorbike	bicycle
CS [29]	94.3	67.4	97.3	79.8	88.6	32.5	48.2	46.3	63.6	73.3	89.0	58.9	93.0	78.2	55.2	92.2	45.0	67.3	39.6	49.9	73.6
PfD [4]	62.5	21.7	42.7	26.3	51.7	5.5	6.8	13.8	23.6	6.9	75.5	11.5	36.8	49.3	0.9	46.7	3.4	5.0	0.0	5.0	1.4
CycleGAN [10]	82.5	32.4	81.8	34.7	73.5	22.5	8.7	25.4	21.1	13.5	71.5	26.5	41.7	50.1	7.3	78.5	20.5	19.5	0.0	12.5	6.9
CyCADA [47]	-	38.8	82.4	38.9	79.0	26.1	19.3	33.2	32.4	21.3	73.9	37.1	61.8	56.2	17.6	78.5	10.0	31.0	10.7	13.8	14.2
UNIT [51]	-	36.1	79.2	28.5	75.9	22.1	13.6	27.0	29.7	18.8	75.9	25.8	56.3	57.5	21.8	81.1	18.9	21.6	1.5	13.7	17.2
Proposed	-	**39.7**	**84.1**	34.6	**80.5**	24.4	17.7	32.5	31.1	**27.4**	**79.7**	26.9	**68.7**	**58.8**	21.1	**84.4**	**22.6**	21.2	1.0	20.1	**17.8**
CS [29]	95.5	75.6	97.9	83.5	91.6	56.5	61.2	54.8	63.9	73.6	91.3	59.9	93.2	77.7	60.1	94.0	79.3	87.0	76.1	61.0	73.2
PfD [4]	82.9	40.0	79.2	26.9	79.5	19.1	27.4	13.8	23.6	6.9	75.5	11.5	36.8	49.3	0.9	46.7	3.4	5.0	0.0	5.0	1.4
CycleGAN [10]	87.7	46.0	85.4	39.0	85.4	42.3	26.3	37.8	40.1	24.8	81.3	28.8	79.4	62.5	27.2	85.5	32.9	44.3	0.0	29.1	17.4
CyCADA [47]	88.5	**48.7**	89.4	45.1	85.3	42.1	23.0	39.3	39.1	25.9	84.4	42.7	79.9	63.6	29.7	86.3	35.3	44.6	11.7	30.8	26.6
UNIT [51]	86.8	47.6	85.2	33.3	85.4	46.8	28.7	35.8	36.2	26.4	83.1	36.5	81.8	63.2	27.0	88.5	43.0	50.9	0.0	30.1	19.6
Proposed	**88.7**	48.1	**89.7**	40.9	**85.9**	43.2	21.0	35.7	37.5	**29.8**	84.3	33.3	**87.4**	62.0	26.7	88.0	**43.4**	**53.6**	0.0	25.4	20.8
CS [29]	-	70.8	97.3	79.5	90.1	40.1	50.7	51.3	56.1	67.0	90.6	59.0	92.9	76.7	54.2	92.9	68.8	80.6	68.5	58.0	71.7
ROAD [149]	-	35.9	85.4	31.2	78.6	27.9	22.2	21.9	23.7	11.4	80.7	29.3	68.9	48.5	14.1	78.0	19.1	23.8	9.4	8.3	0.0
Adapt [147]	-	41.4	86.5	25.9	79.8	22.1	20.0	23.6	33.1	21.8	81.8	25.9	75.9	57.3	26.2	76.3	29.8	32.1	7.2	29.5	32.5
Proposed	-	**42.0**	81.4	28.6	**80.4**	27.4	12.0	**32.9**	**38.3**	**28.6**	**82.5**	29.4	**78.6**	**63.4**	16.7	**84.0**	25.5	**41.3**	0.2	33.6	12.4

Table 5.4: Semantic segmentation results (IoU) on Cityscapes validation set obtained by DRN (top), Deeplabv3 (mid), and Deeplabv2 (bottom) model trained on synthetic PfD images translated to Cityscapes domain by proposed density matching method and baselines.

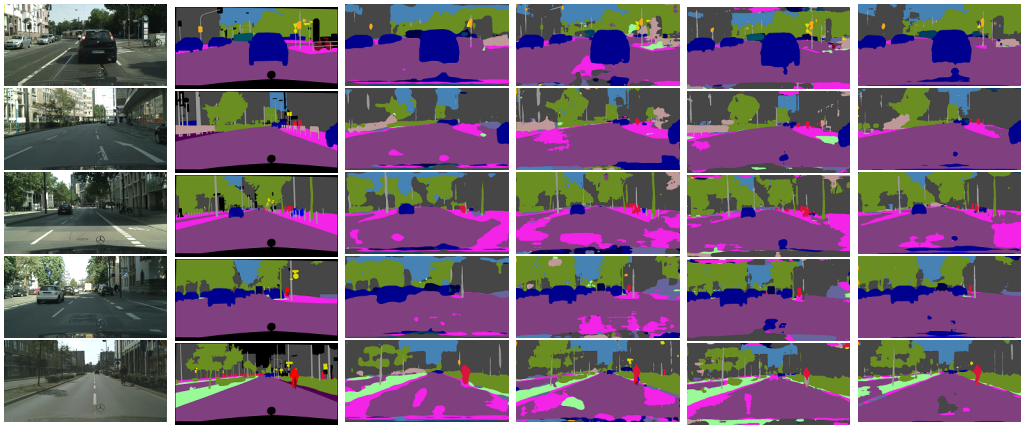


Figure 5.8: Semantic segmentation results obtained on Cityscapes validation set performed by DRN model, which was trained on the synthetic images translated by the proposed density matching method and baseline methods. © 2020 IEEE

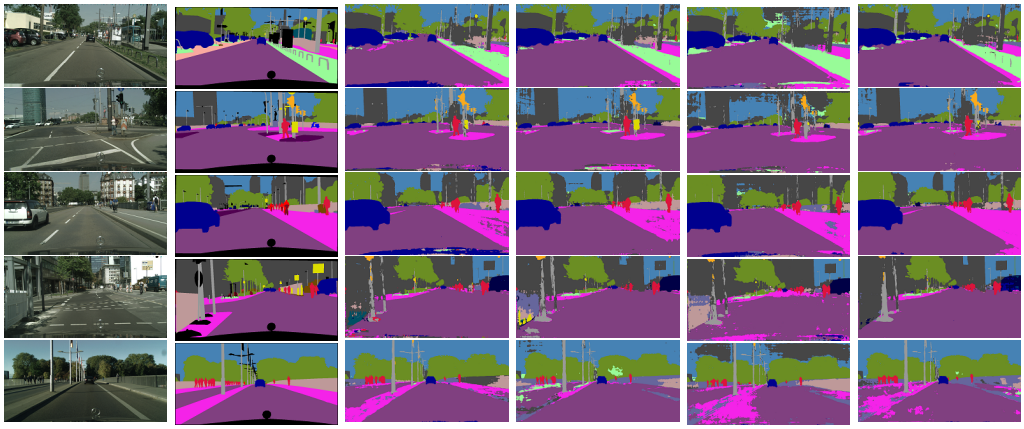


Figure 5.9: Semantic segmentation results obtained on Cityscapes validation set performed by Deeplab model, which was trained on the synthetic images translated by the proposed density matching method and baseline methods. © 2020 IEEE

using each importance group as the source and the entire Cityscapes training set as the target. Next, each of the three trained models performs inference on the entire PfD, providing three translated datasets. Each of the resulting datasets exhibits different adaptation quality. Finally, three DRN segmentation models were trained on the respective translated datasets and evaluated on the Cityscapes validation set.

Figure 5.10: Examples of image transfer by KLIEP GAN trained on translated synthetic images of different importance cohorts. © 2020 IEEE

The results of the ablation study, shown in Table 5.5, confirm that higher importance as estimated by KLIEP reflects greater similarity to the target distributions. This suggests that leveraging more informative samples, as indicated via their importance estimates, can increase the consistency of adversarial image-to-image domain adaptation. This trend can also be observed in the improvement of image translation going from the low-importance cohort to the high-importance cohort, as demonstrated by the increase in mean IoU. Figure 5.10 illustrates this qualitative improvement in translation consistency.

5.6 Discussion

As discussed in Chapter 1, traditional deep models for computer vision tasks in the context of autonomous driving cannot efficiently perform in the real world if they are trained purely on synthetic data. It is necessary for the synthesized training data to be closer to real test data. One way of achieving that is known as domain adaptation, a technique that aims to make both data distributions closer. An attempt to resemble target distribution typically results in structural perturbations of the source images and semantic inconsistency with ground truth annotations.

Chapter 4 showed distorted image semantics during appearance learning where an adversarial network removed synthetic pedestrians augmented into real scenes. Similarly, this chapter showed class confusion arising in multiple sim-to-real adaptation settings such as PfD-Cityscapes, SYNTHIA-Cityscapses, or PfD-KITTI. Subsequently, it analyzed the causes of the observed phenomenon by applying a SIR strategy to tackle the imbalance problem and restore the global statistics of both datasets in each setting. The experiments show that this strategy is efficient for a setting where target labels are available. Furthermore, applying SIR improves the visual quality of translated images of an underlying method as well

Method	mean IoU	road	sidewalk	building	wall	fence	vegetation	sky	car
Cityscapes [29]	67.4	97.3	79.8	88.6	32.5	48.2	89.0	93.0	92.2
PfD [4]	21.7	42.7	26.3	51.7	5.5	6.8	75.5	36.8	46.7
Low	27.9	75.6	28.7	69.1	14.5	18.5	63.4	45.8	75.7
Medium	28.3	77.8	24.0	71.6	10.7	17.1	69.3	69.6	73.5
High	30.2	82.2	40.2	72.1	15.3	23.2	72.9	69.5	77.6

Table 5.5: Semantic segmentation results (IoU) obtained on Cityscapes validation set by DRN method trained on translated synthetic images from different importance cohorts.

as the performance of the semantic segmentation network trained on the translated images. These results could be achieved without introducing additional constraining loss functions.

This chapter further improves the proposed density pre-matching strategy by integrating it directly into the adversarial sim-to-real adaptation model. Employing the KLIEP density ratio estimation procedure in conjunction with the cycle-consistency loss function allows tackling class covariate shift problems in synthetic and real datasets on-the-fly. The experiments show that this strategy is similarly effective in multiple sim-to-real settings in an autonomous driving context. First, the toy experiment showed the effects of the density ratio estimation by making a plane GAN model sampling the Gaussian target distribution more accurately. Consequently, additional large-scale experiments have shown that KLIEP loss is beneficial for adversarial learning not only due to improving the semantic consistency of translated images but also due to improved performance of segmentation networks trained on them. Finally, the ablation study demonstrated how obtained importance scores affect adversarial learning by comparing several cohorts of translated samples divided according to their importance weights. Equally to data subsampling using SIR, the extension based on density pre-matching is an effective tool for reducing the sim-to-real gap, but the availability of privileged knowledge about target data global statistics limits both techniques. This limitation can be circumvented by relying only on a representative annotated subset of target data; otherwise, more sophisticated methods are required.

6 Disentanglement

Recent domain adaptation techniques rely on adversarial loss and act in the image space. Adversarial training inflicts perturbations at the pixel level in the translated images attempting to reconstruct the distribution of reference data. As demonstrated in Chapter 5, the discrepancy in the global statistics results in significant corruption of the images arising during the adaptation procedure. Such hallucinations substantially reduce the applicability of translated images as they become inconsistent with the underlying ground truth. The previous chapter suggested alleviating these undesirable effects by pre-matching the densities of both datasets to circumvent the problem. The proposed technique showed its effectiveness but required privileged information about the global statistics of reference data unavailable in typically sim-to-real settings.

This chapter describes a mechanism that aims to perform semantically consistent sim-to-real domain adaptation without knowledge about underlying target distribution in any form, such as e.g. semantic labels as demonstrated in Figure 6.1. The proposed method follows the assumption that domain-agnostic content features and domain-specific style features can be learned independently. Consequently, this idea suggests learning the entire content manifold so that a disentangled style allows the generation of images from arbitrary content directly in the target domain.

6.1 Content and Style Disentanglement for Semantic Consistency

The proposed unsupervised domain adaptation method utilizes a single encoder to extract the domain-agnostic content features of images from both source and target datasets, along with fixed style codes to represent the domain-specific appearance features. A decoder can use a content vector learned from a source image and an appearance vector from the target domain to synthesize an image with original content looking similar to the images in the target data. This adaptation method operates unsupervised, as there is no correspondence between source and target images. By separating content and style learning, it aims to learn semantic consistency without using privileged information, such as pre-trained segmentation networks or semantic or depth maps.

Figure 6.2 shows the overview of the approach. It consists of several networks: an encoder E, a decoder G, along with individual discriminator D_i per each domain $x_i \in X_i$ $(i = a, b)$, with a and b denoting source and target domains,

Figure 6.1: An example of semantically consistent *sim-to-real* adaptation achieved by disentanglement. © 2020 IEEE

respectively. As shown in Figure 6.2, the encoder E maps the input image x_i to a code $c_i = E(x_i)$, which aims to represent domain-agnostic content. Each domain also has a fixed style code s_i, that represents the appearance features of the particular domain. The style code s_i is initialized as s_i by sampling a vector of size 256 from a uniform distribution within (0, 1). During the training, a repeating cross-domain translation facilitates the disentanglement of the vectors c_i and s_i and stimulates them to embrace content and style features, respectively. Specifically, cross-domain transfer consists of extracting the content features from the source samples and normalizing them to the target style vector using AdaIN [136]:

$$\text{AdaIN}(c_i, s_i) = \sigma(s_i) \left(\frac{c_i - \mu(c_i)}{\sigma(c_i)} \right) + \mu(s_i) \tag{6.1}$$

In Equation 6.1, c_i represents the extracted content code. In contrast to instance normalization (IN), which normalizes the input to the style predefined by affine parameters, AdaIN normalizes the content input to an arbitrary given style.

The learning process is constrained to image reconstruction and translation to enable image generation in both domains. The first constraint uses reconstruction loss to facilitate feature learning by autoencoding the input. The encoder learns the content feature vector of an image, which for reconstruction, is then combined with the style of the domain where the image originates. To achieve that, the optimization task minimizes the reconstruction loss $\mathcal{L}_{rec}^{aa}$:

$$\mathcal{L}_{rec}^{aa}(E, G) = \mathbb{E}_{x_a \sim P_a} \| G(E(x_a), s_a) - x_a \|_1 \tag{6.2}$$

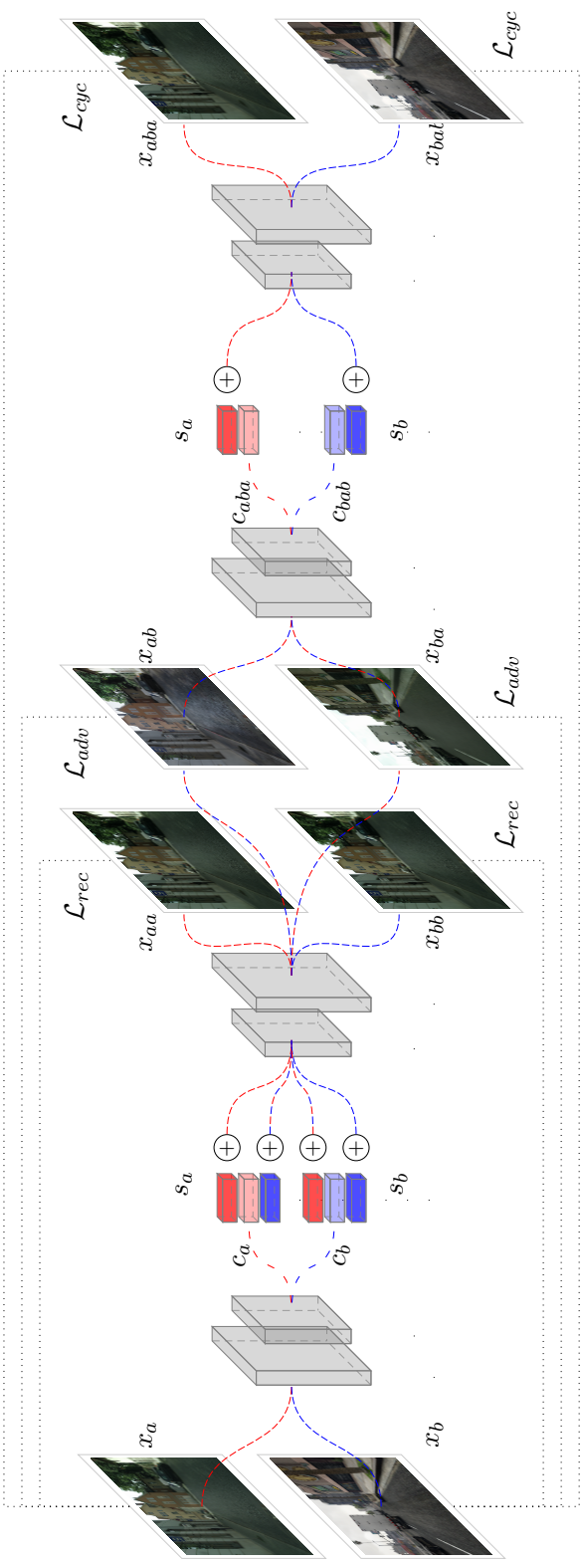


Figure 6.2: Overview of the content disentanglement network. Shared encoder extracts content codes c_a and c_b from given images. The decoder generates reconstructed images x_{aa}, x_{bb} and translated cross-domain images x_{ab}, x_{ba} by combining the c_a or c_b with domain-specific style code s_a or s_b.

Further constraint ensures consistency in cross-domain translation by employing the cycle reconstruction technique [10]. Specifically, the process translates an image x_a to the domain b, resulting in image x_{ab}, and maps it again to the original domain a, producing x_{aba}. The cross-domain constraint requires the network to reconstruct the input after translating it to the target domain and back to the source domain. The cycle reconstruction loss which implements cross-domain constraint is defined as follows:

$$\mathcal{L}_{cyc}^{aba}(E, G) = \mathbb{E}_{x_a \sim P_a} \|G(E(x_{ab}), s_a) - x_a\|_1 \tag{6.3}$$

In Equation 6.3, x_{ab} represents a sample translated from a to b.

Lastly, the proposed method employs standard adversarial loss to facilitate the similarity of the translated images to the target domain distribution and further ensure visual consistency. Similarly to [182], adversarial loss operates on random patches $\hat{x}$ of the translated and reference images. Figure 6.2 shows the reconstruction, cycle reconstruction, and adversarial losses. The adversarial loss is calculated as follows:

$$\begin{aligned} \mathcal{L}_{adv}^{a}(E, G, D_a) = \mathbb{E}_{x_a \sim P_a} \log D_a(\hat{x}_a) \\ + \mathbb{E}_{x_b \sim P_b} \log(1 - D_a(\hat{x}_{ba})) \end{aligned} \tag{6.4}$$

In Equation 6.4, $\hat{x}$ represents the random image patches, and x_{ba} represents the image translated from b to a. Other components such as $\mathcal{L}_{rec}^{bb}$, $\mathcal{L}_{cyc}^{aba}$, and $\mathcal{L}_{adv}^{b}$ are given analogously to Equations 6.2, 6.3 and 6.4. The following equation defines the aggregated loss:

$$\begin{aligned} \min_{E,G} \max_{D_a,D_b} \mathcal{L}(E, G, D_a, D_b) = \lambda_1(\mathcal{L}_{rec}^{aa} + \mathcal{L}_{rec}^{bb}) \\ + \lambda_2(\mathcal{L}_{cyc}^{aba} + \mathcal{L}_{cyc}^{bab}) \\ + \lambda_3(\mathcal{L}_{adv}^{a} + \mathcal{L}_{adv}^{b}) \end{aligned} \tag{6.5}$$

In Equation 6.5 λ_1, λ_2, and λ_3 define the contribution of each loss term.

Network Architecture. The proposed algorithm consists of a generator network comprising an encoder and decoder, and two discriminators, one for each domain. The encoder and decoder are similar to the MUNIT network [14], and PatchGAN is used as a discriminator [53]. The encoder comprises several blocks of convolutional layers, instance normalization layers, and ReLu as well as ResBlocks. Symmetrically, decoders have nearest-neighbor interpolation layers along with deconvolutional upsampling layers. Each discriminator obtains random patches from translated and source images with a size between 1/8 and 1/4 of the total image size on each side. Figure 6.2 gives a detailed overview of the encoder-decoder network. The objective function defined in Equation 6.5 guides the training of the generator network to translate images from the source domain

Figure 6.3: Architecture of encoder and decoder networks of the proposed disentanglement model. Parameters of the convolutional layers are: channels, kernel size, stride, padding. © 2020 IEEE

to the target domain in a visually indistinguishable manner while preserving the content.

The parameters λ_1, λ_2, and λ_3 of loss function in Equation 6.5 are set to 10, 10, and 1 respectively. The optimization is performed by Adam using mini-batch stochastic gradient descent [183] with coefficients of 0.5 and 0.999. The training process goes for 200 epochs using 2975 random images from the synthetic dataset *Playing for Data* (PfD) [4] and 2975 images from the Cityscapes [29] dataset.

6.2 Experiments

Datasets. Like the experiments described in the previous chapters, this experimental setting uses real Cityscapes [29] and the synthetic PfD large-scale datasets for urban traffic scenes. The Cityscapes dataset consists of $5,000$ images of resolution of 2048×1024 pixel, divided into a training set with $2,975$ samples, validation set with 500 samples, and remaining images as publicly not available test set. The semantic labels have 19 classes, with the road, building, fence, vegetation, sky, person, and car among the most crucial ones. The *Playing for Data* (PfD) dataset contains $24,966$ images of resolution 1914×512 pixels grabbed from the GTA [4] computer game. This dataset also includes annotations for 19 semantic classes as the Cityscapes dataset. Due to memory and time constraints, the disentanglement experiments use images that are downsampled to 1024×512 pixels for both datasets.

Evaluation. The proposed unsupervised image translation method is compared to the baselines that do not utilize privileged knowledge, such as pre-trained additional segmentation critics or plain semantic maps. These baselines include CycleGAN [10], MUNIT [14], DRIT [138], and CUT [160]. The configuration of the MUNIT disabled the perceptual loss to avoid using a pre-trained segmentation network. All models used the PfD and Cityscapes datasets for training, with images resized to 1024×512 pixels. During the evaluation experiments, each baseline and the proposed approach translate created a translated dataset by transferring the entire PfD dataset to the Cityscapes domain. These translated PfD samples were then used to train semantic segmentation networks, and the performance of these networks on the Cityscapes *val* built quantitative evaluation of the translation. A higher segmentation score indicates better domain translation.

Qualitative Results. Figure 6.4 demonstrates the translation results of the proposed network, which illustrates the high quality of the style transfer but also preserved semantics of the translated image. The style quality is especially prominent in the road texture or color of lane marking. Figure 6.7 shows a comparison with other baselines. Each row shows the synthetic source image along with the translated images. It is evident that CycleGAN translates a car in an image to appear as if it is part of a road and adds vegetation-like features to the sky region of the image. The network MUNIT can translate images in

Source Translated

Figure 6.4: Examples of synthetic images translated by the disentanglement method from PfD to Cityscapes domain. © 2021 IEEE.

multiple styles using the style sampling technique but struggles to preserve the semantic content unchanged. For example, MUNIT also covers the entire sky area with vegetation patterns. Like CycleGAN, it also represents the ego car to the part of the road. Images translated by DRIT also exhibit class perturbations. CUT demonstrates superior style learning by increasing the mutual information between the patches sampled from exact locations in generated images. However, it has the same translation mismatches as CycleGAN. In contrast, the proposed style distillation mechanism can perform domain adaptation while maintaining class consistency of the entire image.

Quantitative Results. In the quantitative experiment, the segmentation performance of the DRN-C-26 network [173] trained on the dataset generated by translating PfD images to the Cityscapes domain was assessed with the Cityscapes validation set. The first experiment in Table 6.1 serves as an oracle and represents the upper bound for semantic segmentation, which is the performance of the algorithm trained and evaluated within the same real domain. The second experiment represents the lower bound for the semantic segmentation, where a network is trained on the PfD dataset but evaluated on the Cityscapes validation set. It also shows the translation performance of the baselines discussed previously. According to the table, the proposed content and style disentanglement increases the segmentation score by an average of 2.9 points, resulting in 88.7. It also increases the intersection over union by 1.6 points, resulting in a score of 39.0. The proposed method also explicitly improves the segmentation of several classes, including buildings, vegetation, sky, and car.

6.2.1 Content Space

This experiment evaluates the effectiveness of the proposed approach for learning relevant content codes through an ablation study. To do this, the t-SNE algorithm [11] is applied to the content codes learned during the previously described training. Figure 6.5 depicts the processed vectors. The features obtained from source images overlap the codes obtained from cross-domain translated images, with c_a coinciding c_{ab} and c_b coinciding c_{ba}. The exact matching confirms effective learning of the domain-agnostic content vectors and detachment from style codes. On the other hand, the disentanglement is evident since the method does not directly apply constraints on content vectors but instead learns them implicitly through intra-domain and cross-domain reconstruction. Figure 6.6 demonstrates images reconstructed from interpolated content vectors in different styles.

6.3 Discussion

Chapters 4 and 5 demonstrated how adversarial domain adaptation tends to impose regularities learned in the target data onto the translated images when attempting to reconstruct the target distribution. Additionally, these chapters

Method	Train	Eval	Accuracy	mean IoU	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorbike	bicycle
None	CS	CS	94.3	67.4	97.3	79.8	88.6	32.5	48.2	46.3	63.6	73.3	89.0	58.9	93.0	78.2	55.2	92.2	45.0	67.3	39.6	49.9	73.6
None	PfD	CS	62.5	21.7	42.7	26.3	51.7	5.5	6.8	13.8	23.6	6.9	75.5	11.5	36.8	49.3	0.9	46.7	3.4	5.0	0.0	5.0	1.4
CycleGAN	PfD ⟶ CS	CS	82.5	32.4	81.8	34.7	73.5	22.5	8.7	25.4	21.1	13.5	71.5	26.5	41.7	50.1	7.3	78.5	20.5	19.5	0.0	12.5	6.9
DRIT	PfD ⟶ CS	CS	78.9	27.4	78.8	26.1	68.7	13.1	10.2	18.8	11.4	17.9	56.8	4.8	34.1	45.8	9.4	73.3	13.4	14.4	10.5	7.7	5.3
MUNIT	PfD ⟶ CS	CS	85.8	37.4	**85.9**	**35.9**	79	26.1	**18.4**	31.2	23.8	17.1	74.4	17.4	55.8	52.3	**17.3**	82.5	21.2	22.8	**12.0**	**20.7**	**16.7**
CUT	PfD ⟶ CS	CS	82.6	34.0	78.7	26.7	73.3	17.4	16.9	22.2	25.7	16.2	73.4	20.8	64.8	54.8	13	79.3	20.8	22.2	0	11.6	8.6
Proposed	PfD ⟶ CS	CS	**88.7**	**39.0**	82.0	28.1	**80.7**	**30.8**	15.0	**32.0**	**34.4**	**23.6**	**80.1**	**34.3**	**76.9**	**55.7**	10.2	**82.9**	**25.5**	**26.5**	1.6	11.9	8.8

Table 6.1: Semantic segmentation results (IoU) on Cityscapes validation obtained by DRN model trained on synthetic dataset trained on synthetic PfD images translated to Cityscapes domain by proposed disentanglement method and baselines. Translated dataset is denoted as Source → Target and CS denotes Cityscapes dataset.

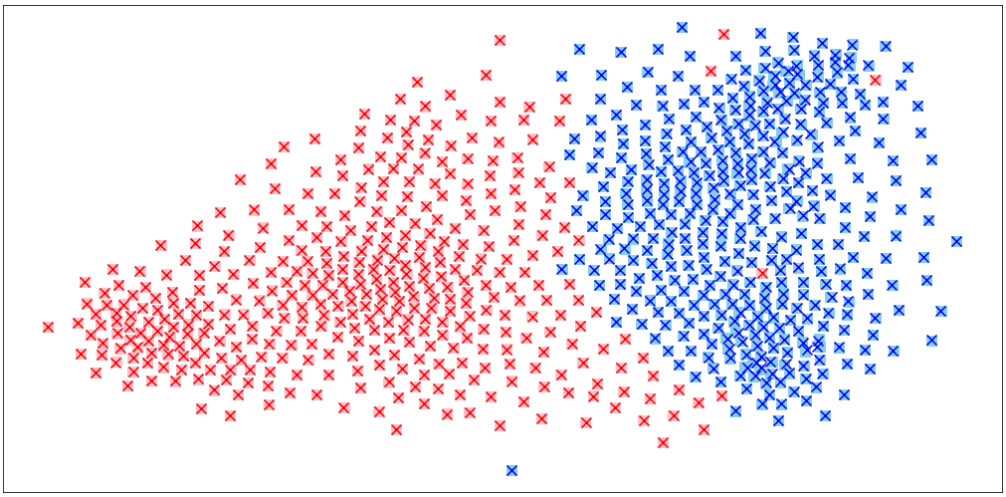


Figure 6.5: t-SNE[11] projections derived for the extracted random content codes of source synthetic images c_a (red □), synthetic images translated to target domain c_{ab} (red ×), target real images c_b (blue □), and real images translated to source domain c_{ba} (blue ×). © 2021 IEEE

Figure 6.6: Interpolation results for content vectors c_a^1 (leftmost) and c_a^2 (rightmost) extracted from two random PfD images with source (top) and target appearances (bottom). © 2021 IEEE

showed that known global statistics of source and target help to mitigate such perturbations if matching densities are applied. In a sim-to-real setting, though, the global statistics of target data, as well as image pairs, are not available. Therefore more complex unsupervised methods are required.

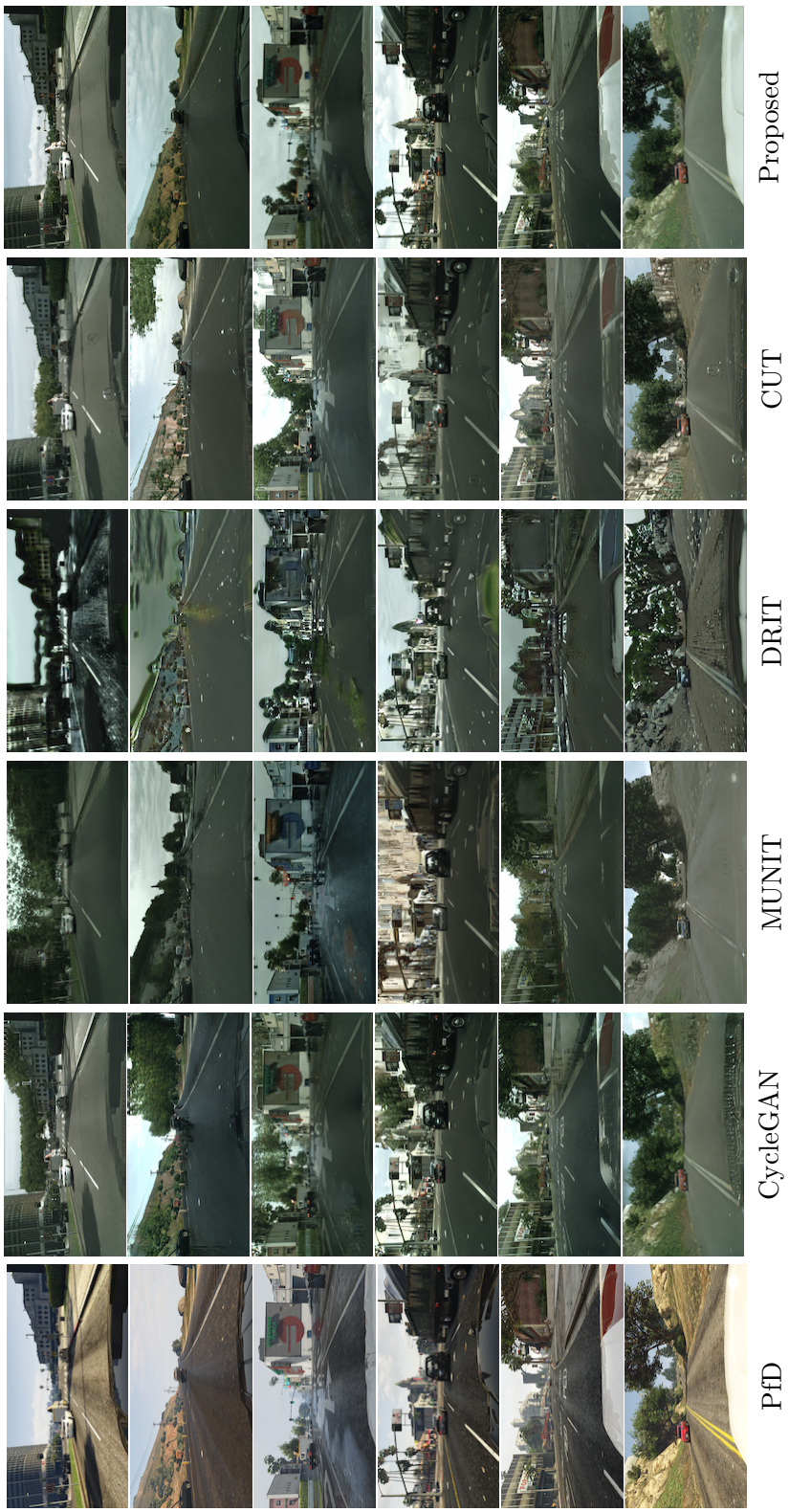


Figure 6.7: Semantic consistency of the images translated by proposed method compared to baselines.

Source Translated

Figure 6.8: Examples of texture-less images translated to Cityscapes domain with proposed disentanglement method.

This chapter proposed a technique of independent learning of content and style vectors in order to achieve consistent sim-to-real image transfer. Disentangling the domain-agnostic content feature and domain-specific style feature allows for domain transfer by simply applying the target style of interest to arbitrary content. Extracting a meaningful content feature can potentially enable learning of the entire content manifold so that generation from any particular content with a target style preserves the macro-structure consistency.

The proposed model embraces a lightweight architecture consisting of a single encode and decoder and multiple discriminators, one per domain. The architecture uses the AdaIN [184] technique for integrating domain-specific fixed style codes into the content learning process.

Efficient independent learning of content vectors is demonstrated through the ablation study and visualization of the codes using the t-SNE technique. Furthermore, a comparison of translated images shows that the proposed method improves the visual consistency of produced samples and alleviates the confusion of imbalanced patches. Additionally, the improved performance of a semantic segmentation network trained on the translated examples indicates improved transfer consistency and better suitability for autonomous driving tasks.

Nevertheless, class consistency has the other side of a medal such that the generative power of the transfer model is fairly limited. The textures of the translations are mostly derived from synthetic input images. Additional experiments with texture-less synthetic renders confirm that finding. Figure 6.8 shows that the transfer of synthetic images generated from a 3D scene without textures and materials to the real domain cannot produce samples that resemble target data.

Another indicator for such behavior is for example the bonnet area produced in the PfD-Cityscapes setting. The adaptation networks with high generative power produce the bonnet patch typical for Cityscapes, whereas networks with low generative power preserve the GTA game car bonnet. That means that a synthesis pipeline must still rely on the tedious and expensive creation of the virtual environment, or other methods not relying on high-fidelity synthetic images are required. The next chapter investigates it in more detail.

7 Scene Graphs

The independent learning of content and style features discussed in Chapter 6 offers numerous possibilities for data generation. Such content vector extraction enables learning the entire manifold of possible contents and consequent image generation using a predefined appearance. However, implicit content learning limits the ability to manually manipulate content, which is necessary for defining individual traffic scenarios as discussed in Chapter 1. Moreover, the style learned by the disentangling method is limited to high-level visual characteristics, so the data synthesis pipeline is still dependent on textures and materials provided by the original simulation.

An alternative to this is the explicit formulation of traffic scene content using available forms of scene description. One of such forms recently proposed in the area of visual question-answering (VQA) is scene graphs, which allow for reasoning about scene content and structure. While manual content definition does not encompass the wide range of potential real-world traffic scenarios, it can allow dedicated configuration of required scenarios. Therefore, the following data synthesis concept suggests using two principles: procedural content generation and appearance learning. The first makes an unbounded variety of generated scenarios possible while the latter ensures that produced images closely resemble the reference data. Moreover, explicit content formulation permits the manual design of traffic scenarios.

7.1 Image Generation with Procedural Synthetic Scene Graphs

Procedural generation is a technique that originates from video game development and also finds wide application in the movie industry. This technique is motivated by the fact that traditional manual content generation has several limitations in terms of scalability. First, it requires highly qualified and skilled computer graphic artists and designers to create complex virtual environments and objects. While the *Playing for Data* (PfD) [4] dataset may give the impression that training data is freely available, the data is only accessible to the research community due to the efforts of video game developers. Additionally, once created, such environments are difficult to modify. Furthermore, manual content generation is becoming increasingly complicated as computer graphics technology advances [185]. As an alternative, content can be produced in an automated fashion using an algorithm that is guided by predefined rules, pa-

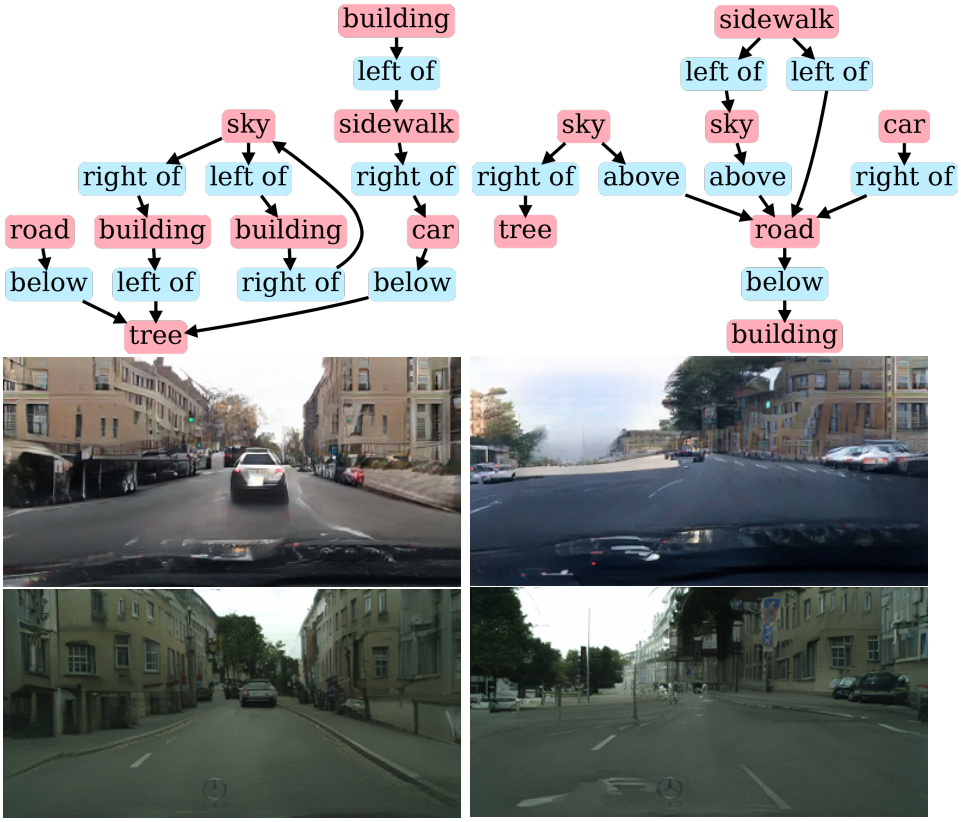


Figure 7.1: Examples traffic scenes images generated from synthetic scene graphs by the proposed method using BDD and Cityscapes appearances. © 2021 IEEE

rameters, and heuristics. In computer game development, procedural content generation techniques are often used to create textures and model the effects of smoke, fire, clouds, or trees. More sophisticated procedural content generation mechanisms aim to create pseudo-infinite cities and indoor environments [186]. For example, the proposed traffic environment generation pipeline, demonstrated in Figure 7.2, largely relies on city generation from [186].

7.1.1 Synthetic 3D Scene Graphs

Similar to previously discussed sim-to-real settings, the graph-based approach focuses on unsupervised traffic scene generation. In a graph-based setting, like in unsupervised image-to-image translation, pairs of synthetic inputs and real scenes are unavailable. The general idea is to condition the generation process on the content of a procedurally generated traffic scene that does not have typical visual characteristics like textures and materials. This condition is formulated as a scene graph describing the scene's content and geometry. The scene graph

conventionally serves as an alternative to textual scene description proposed initially by Johnson et al. [167]. Graph-based description allows the networks to "reason" explicitly about the objects in the scene and their relations, in this case, spatial relations. The data synthesis method aims to produce an image of a traffic scene with the content defined by the dedicated synthetic graph and a realistic style resembling the target data. Figure 7.4 gives a detailed overview of the proposed pipeline, which consists of synthetic scene graph construction and realistic image generation. The graph processor adopts the architecture from the original work of Ashual et al. [67]. And the image generator represents a ResNet architecture which consists of two blocks with convolutional layers, normalization layer and ReLu as well as nine ResBlocks with the following set of two transposed convolutional layers.

Scene graph generation involves procedural 3D scene modeling and a scene graph generator. Procedural modeling requires a road network to build upon and a set of rules to produce typical traffic content. Using the existing street network from an OpenStreetMap [187] ensures high-level realism of generated scenes in terms of the content, but any arbitrary network can also be used. The rules are divided into rules that define the appearance of buildings and rules that define the placement of dynamic objects. Figure 7.2 shows several examples of such procedural traffic scene generation without textures. These examples contain traffic scenes generated in a randomized fashion. Additionally, as the meta-information about the scene is available, the data synthesis pipeline produces the ground truth for each generated image sample. For example, Figure 7.3 demonstrates semantic segmentation ground truth corresponding to the procedurally generated traffic scenes.

After the procedural generation of a synthetic 3D scene, a scene graph can be constructed based on the spatial information available. The relative position between a pair of objects in the scene determines their spatial relation. A simple comparison of z coordinates of the objects establishes the relation *in front of* or *before*. Additionally, rasterizing the z-coordinate mentioned before into several bins provides a depth parameter for scene manipulation.

For the construction of the scene graphs, the proposed approach adopts the setup from [167] for traffic scene data. In this setting, each scene graph consists of the nodes n_i encoding the objects i of the scene and edges e_{ij} representing relations between any two nodes n_i and n_j. Each scene graph serves as input for the graph processor P, which is based on the work of Ashual et al. [67] but enhances it in several ways, fostering spatial information fusion. First, the presented method expands the set of basic object relations like *left of* or *above* described in [67] by incorporating the spatial information related to objects. This information results in spatial relations between objects such as *in front of* or *behind*, as well as spatial attributes of single objects like depth component z. Importantly, this information is available in the simulation at no additional cost. The approach extends the set of objects to generate a comprehensive traffic scene

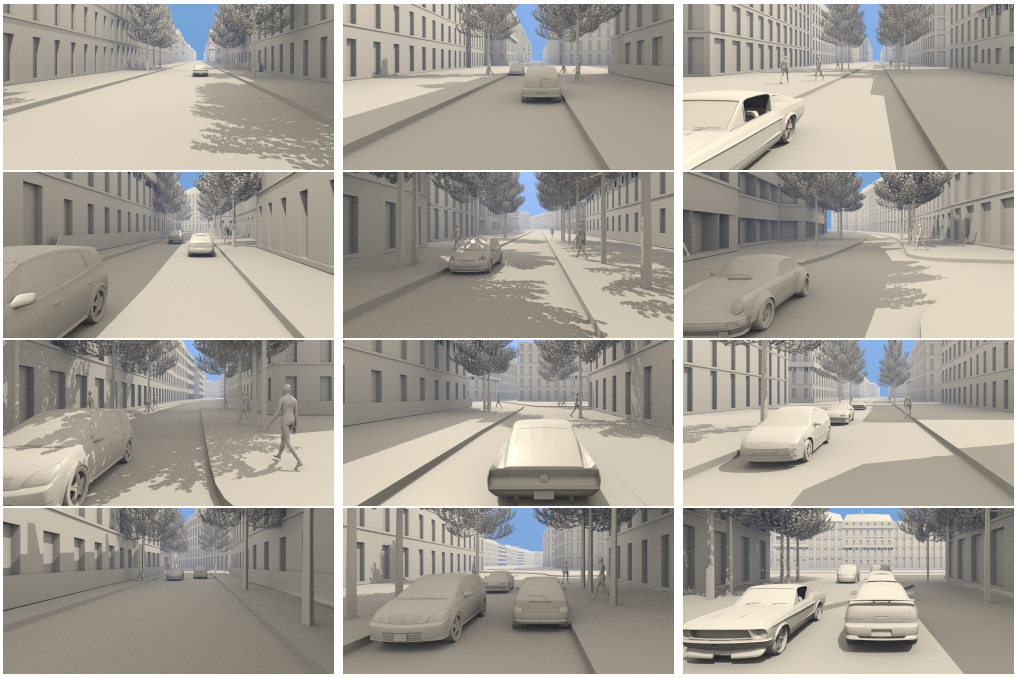


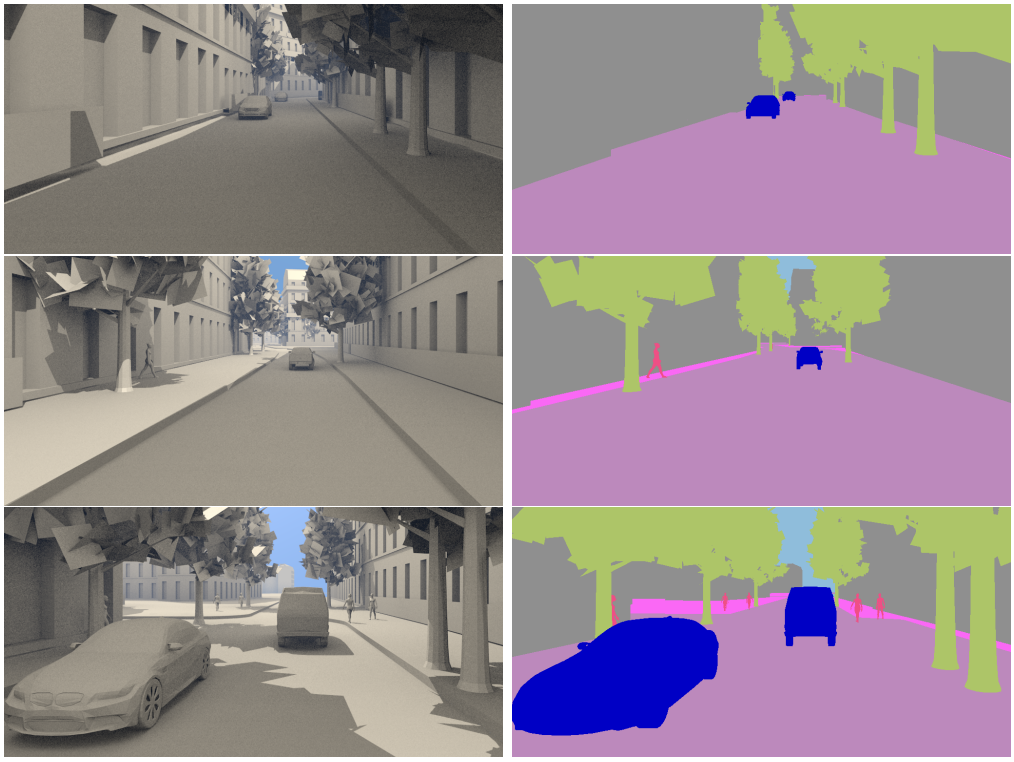
Figure 7.2: Examples of traffic scene images created with procedural generation using Blender without textures [12].

and integrates background classes sky, building, and vegetation. Accordingly, n_i denotes every single node in a graph:

$$n_i = [o_i, l_i, z_i], o_i \in \mathbb{R}, l_i \in \{0,1\}^{L \times L}, z_i \in \{0,1\}^Z \tag{7.1}$$

In Equation 7.1, o_i is an index of class C, l_i represents the location of the object on the image grid of the size L, and z_i represents the position of the object along the z-axis of the scene with depth Z. Similarly, each edge $e_{i,j} \in \mathbb{R}$ is a code of a particular relation from the dedicated list.

Given a scene graph, the graph processor P produces an intermediate representation called a scene layout $t \in \mathbb{R}^{H \times\ W \times C}$, which serves as the basis for sequential image generation. Therefore, the graph-layout tuples are derived from the simulation to train the graph processor P in a supervised fashion. This training requires mask m_i and bounding box $b_i \in \mathbb{R}^4$ for each object i from the simulated 3D scene. Similar to [167], the graph processor P is composed of 3 networks: *graph network*, the *mask regression network*, and the *box regression network*. The graph network encompasses graph-convolutional layers that extract features from scene graphs and encode per-object embeddings. A network dedicated to the generation of masks for each object comprises a number of deconvolution layers, and the network for bounding box generation is a plain MLP. Both synthetic



Textureless traffic scenes | Semantic segmentation ground truth

Figure 7.3: Semantic segmentation maps corresponding to images of traffic scenes generated procedurally.

scene graphs and corresponding layouts produced by the graph processor P are based solely on the scene's content and do not require visual characteristics.

7.1.2 Traffic Scenes

In the second phase, the generated layout t represents a basis for the synthesis of the realistic image x by the image generator G. Generated image x visually resembles the target data $X = \{\hat{x} \in \mathcal{X}\}$ while maintaining the content of the original scene graph. The image generation phase takes synthetic graphs and real images as input, a setting where training pairs of samples are not feasible so that the training occurs in an unsupervised fashion. Generated layouts lack distinctive characteristics, so using only adversarial loss [8] is insufficient. Experiments show that pure adversarial training exhibits insufficient generative power for learning and creating textures. The results of those experiments are demonstrated in Figures 7.5 and 7.6. A combination of adversarial and contrastive [188] losses have been shown to produce better-quality generated textures, as

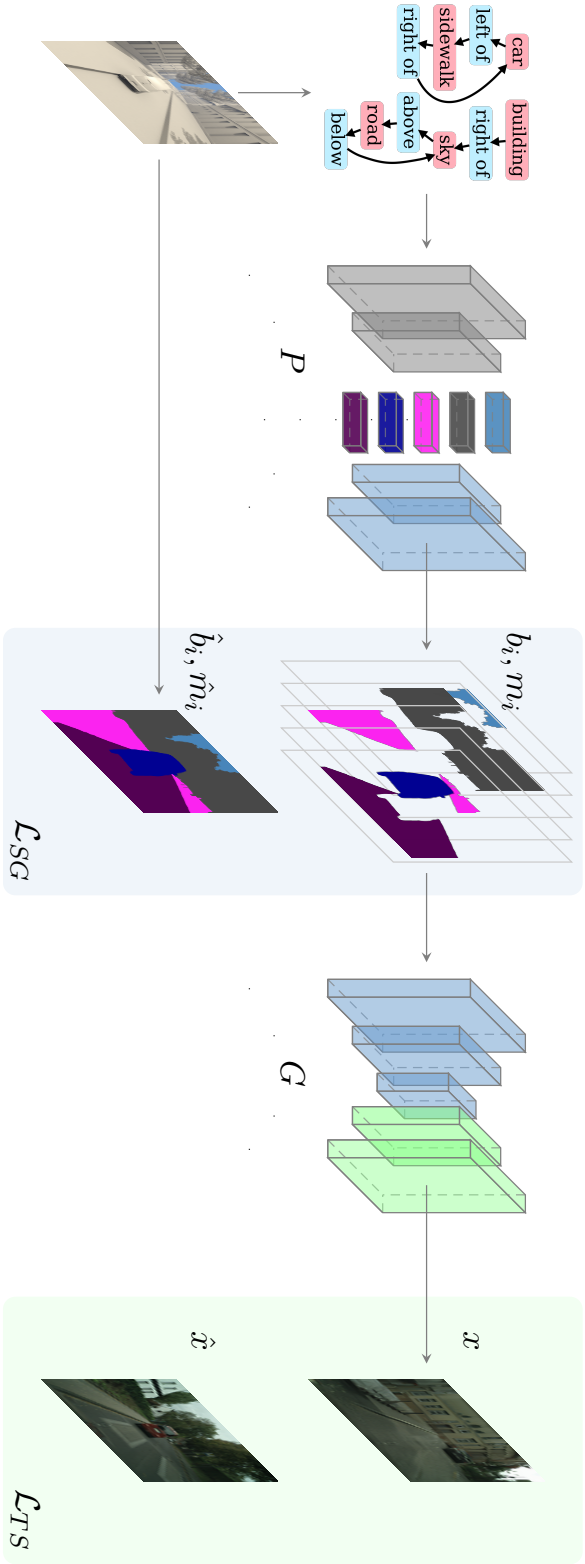


Figure 7.4: Overview of the network with graph processor P, traffic scene generator G, and loss functions $\mathcal{L}_{SG}$ and $\mathcal{L}_{TS}$. © 2020 IEEE

observed in Figure 7.7. Therefore, the final objective for training the processor P and generator G can be defined:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{SG} + \mathcal{L}_{TS} \\ &= \mathcal{L}_{MSE}(b_i, \hat{b}_i) + \mathcal{L}_{GAN}(m_i, \hat{m}_i) + \mathcal{L}_{FM}(m_i, \hat{m}_i) \\ &+ \mathcal{L}_{GAN}(x, \hat{x}) + \mathcal{L}_{NCE}(x, \hat{x})\end{aligned} \tag{7.2}$$

In Equation 7.2, $\mathcal{L}_{FM}$ represents a *feature matching loss* [170], $\mathcal{L}_{NCE}$ represents a multi-layer, patch-wise contrastive loss [189, 160] and $\mathcal{L}_{GAN}$ represents an adversarial loss on masks m_i and images x, $\hat{m}$ denotes a synthetic ground-truth mask, and $\hat{x}$ is a target domain images:

$$\begin{aligned}&\mathcal{L}_{GAN}(m_i, \hat{m}_i) = \\ &\mathbb{E} \log D_m(m_i) + \mathbb{E} \log(1 - D_m(\hat{m}_i)) \\ &\mathcal{L}_{FM}(m_i, \hat{m}_i) = \|\mathbb{E} f(m_i) - \mathbb{E}_{x \sim} f(\hat{m}_i)\|_2^2 \\ &\mathcal{L}_{GAN}(x, \hat{x}) = \mathbb{E}_{x \sim X} \log D(x) + \mathbb{E}_{\hat{x} \sim X} \log(1 - D(\hat{x})) \\ &\mathcal{L}_{NCE}(x, \hat{x}) = \mathbb{E}_{x \sim X} \sum_l \sum_s \ell(\hat{x}_l^s, x_l^s, \overline{x}_l^s)\end{aligned} \tag{7.3}$$

In Equation 7.3, f represents activations in the discriminator's intermediate layers, ℓ denotes a cross-entropy loss for a positive pair of patches as defined by contrastive loss, and x_l^s is the generator's l-th layer feature at specific location s [189].

7.2 Experiments

To accurately evaluate the proposed method, it is necessary to assess its performance on established public datasets from the autonomous driving domain. These datasets include Cityscapes [29], and *Berkeley DeepDrive* (BDD) [190], which are widely adopted real datasets, and PfB [5] and Synscapes [6], which are synthetic datasets. The evaluation process requires the generation of scene graphs for the existing synthetic datasets. The graph generator takes bounding boxes along with semantic and instance maps as input and creates synthetic scene graphs for both the PfD and Synscapes datasets. During the evaluation experiments, the graph processor uses the resulting triples of a graph, bounding boxes, and semantic maps for the training. In particular, the synthetic datasets PfB [5] and Synscapes [6], which are used for scene graph generation, are traffic datasets with about 25000 urban traffic scene images at a resolution of 1914×1052 and 1440×720 respectively. In addition, bounding boxes, semantic maps, and instance maps accompany the images. For image generation, the experiments utilize the real Cityscapes [29] and BDD [190] datasets as a source of visual style.

Figure 7.5: Examples of synthetic scene graphs and corresponding generated traffic scenes with Cityscapes appearance. © 2021 IEEE

Figure 7.6: Examples of synthetic scene graphs and corresponding generated traffic scenes with BDD appearance. © 2021 IEEE

Figure 7.7: Comparison of the generated images (top→ bottom) using other unsupervised generative networks: CycleGAN[10], DualGAN[13], MUNIT[14], DRIT[15], proposed. © 2020 IEEE

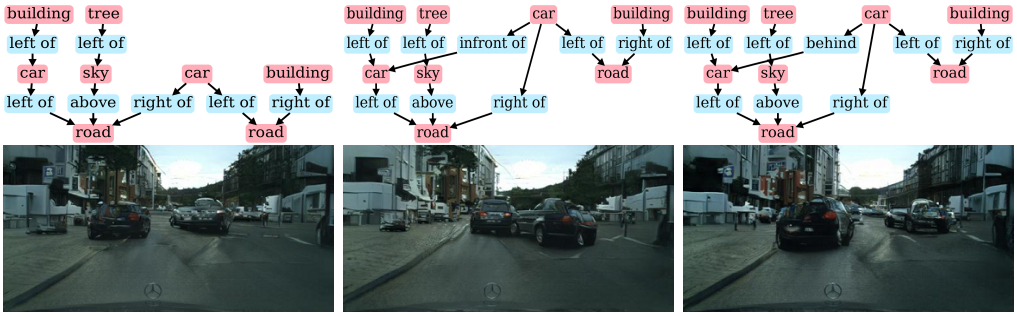
The Cityscapes dataset includes approximately 3000 images of traffic scenes with labeled semantic maps, while the BDD dataset includes approximately $100,000$ images, with only $10,000$ used in the experiments.

Scene graphs generation. The scene graph generator initially creates a dataset with $5,000$ synthetic scene graphs. The experiments were conducted with these samples used as input for the training of the scene graph processor, while the image generator uses images from Cityscapes and BDD with a similar setting for both datasets. The preprocessing step downsized the samples to a 256×512 pixels resolution. After that, the training of all networks goes for 150 epochs. The optimizer uses a learning rate of $1e-3$ and a momentum of 0.99. Figures 7.5 and 7.6 show the results of the experiments with Cityscapes and BDD datasets. The results demonstrate that the proposed pipeline can accurately capture the appearance of real datasets, resulting in the production of realistic imagery. While the generated objects exhibit a certain degree of inaccuracy in details within specific objects, such as cars, the images are fairly consistent at the class level. The generator effectively creates a coherent appearance for objects belonging to the classes road, sidewalk, building, and vegetation.

The influence of the conditioning of the generation process on scene graphs is demonstrated in Figures 7.5 and 7.6. Furthermore, Section 7.2.1 provides further details on the conditioning possibilities and demonstrates how to manipulate traffic scenes by introducing new classes, spatial attributes, and spatial relations.

No car z-attribute 5 z-attribute 2

Figure 7.8: An example of traffic scene manipulation by changing the spatial attribute of the car object. © 2021 IEEE



No relation *in front of* *behind*

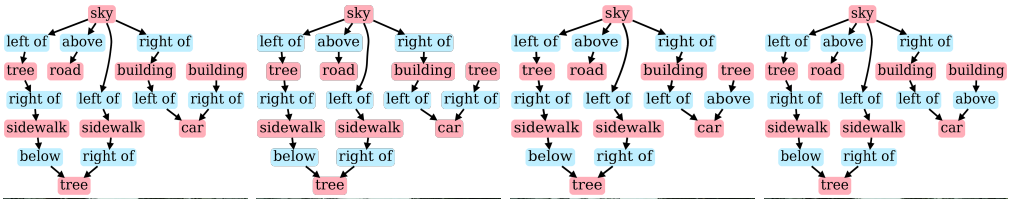
Figure 7.9: An example of traffic scene manipulation by changing the spatial relation of two car objects. © 2021 IEEE

Also, Figure 7.7 shows the comparison with baseline unsupervised image transfer techniques. The comparison includes CycleGAN [10] and MUNIT [14]. Figure 7.7 illustrates that the proposed method produces more convincing textures, which is also reflected in the lower FID score, with CycleGAN achieving 103.05, MUNIT - 75.98, and introduced approach 47.26.

7.2.1 Traffic Scene Manipulation

Spatial Attributes and Relations. The synthetic scene graph generation process enables incorporation of the spatial information about the traffic scenes. For example, Figure 7.8 illustrates how changing the value of an object's z-coordinate affects the image synthesis from the same scene graph. The first image does not contain a car in the scene. In the following images, a car appears, and modifications of its z-attribute moves it along the traffic scene.

The integration of spatial information provided by the scene, in addition to the z-attribute of objects, allows the representation of spatial relationships between them. For example, Figure 7.9 shows several images of a traffic scene created from a scene graph by exchanging the spatial relation between two car objects. Initially, both car objects appear in the scene spatial without constraint circa equally sized. However, introducing the *in front of* relation leads to moving the



Building right of the car | *Vegetation right of the car* | *Vegetation above the car* | *Building above the car*

Figure 7.10: An example of traffic scene manipulation by changing classes. © 2021 IEEE

right car closer to the ego camera and the left car farther away. Changing the relation to *behind*, in turn, pulls the left car before the right car.

Traffic Scene Classes. The proposed method extends the number of object classes that characterize traffic scene scenarios. They include road, sidewalk, vegetation, building, sky, car. The class manipulation experiments confirm the effectiveness of the introduced novel classes. Figure 7.10 illustrates the impact of manipulating the object classes in corresponding scene graphs. Substituting the classes of the particular nodes pointedly adjusts the semantic layout of the synthesized image. In the first image the building is located on the right side of the car, changing the node's class puts vegetation instead of building. Next, adjusting the *right of* relation to *above* results in placing the car under the newly introduced vegetation area. In the final example, the class vegetation is reverted, which leads to returning of the building object.

Global Manipulation. The scene graph makes possible parametrization of any traffic scene characteristic of interest. Including an additional parameter only requires an additional one-hot vector in the scene graph definition. Figures 7.12, 7.11, 7.13 show examples of images generated from a single scene graph but with parameter variation related to weather conditions, daylight conditions and overall appearance of reference dataset or sensor suite.

7.2.2 Image Segmentation

Quantitative assessment of the data generation relies on the downstream vision task. In quantitative experiments, synthesized data is used for training a state-of-the-art semantic segmentation network. This experiment uses the scene graphs derived from the *Playing for Data* dataset, where it randomly selected two sets with $5,000$ and $10,000$ scene graphs. The pipeline generated images and corresponding semantic maps for these scene graphs, which were then used in a training of *Dilated Residual Network* (DRN) segmentation model [173] for 200

Day Night

Figure 7.11: Examples of image manipulation with global scene graph parameter related to daylight conditions.

Normal Snowy Rainy

Figure 7.12: Examples of image manipulation with global scene graph parameter related to weather conditions.

epochs. The baseline, in this case, was trained on plain synthetic data. Both models have been evaluated on the real Cityscapes validation set concerning 8 main supported classes. Table 7.1 shows the IoU scores also known as *Jaccard Index* [180].

The table shows that the model trained on a dataset of $5,000$ generated examples is 5% less accurate compared to the model trained on the original PfD dataset. However, increasing the size of the generated dataset to include twice as many scene graphs and corresponding images narrows this gap to 2 percentage points. Adding $5,000$ synthesized samples to original PfD further improves the performance of the baseline. The table also includes the experiment's results with scene graphs resembling class balance on a real dataset used for evaluation. Knowing the global target data class statistics makes it possible to reduce the number of scene graphs to 2000 examples and segmentation score by 5% compared to the baseline synthetic dataset of $25,000$ examples.

7.3 Discussion

The preceding chapter investigated the methods of semantically consistent transfer of synthetic images into the real domain. Such transfer aimed to leverage available data rendering pipelines but make produced images resemble visual

Figure 7.13: Traffic scene images generated from the same synthetic scene graph using appearances BDD and Cityscapes. © 2021 IEEE

	Sky	Road	Tree	Building	Person	Car	Bus	Truck	Mean
PfB 25k [5]	62.8	41.1	67.8	64.3	14.0	38.8	1.1	8.1	37.3
3D Scene graphs 5k	23.4	76.1	44.7	46.3	8.1	41.2	3.5	1.6	30.6
3D Scene graphs 10k	34.2	68.0	62.5	60.8	9.7	40.7	1.3	3.0	35.0
3D Scene graphs 5k + PfB	25.4	80.2	55.9	64.6	29.3	64.2	6.3	7.7	41.7
3D Scene graphs 2k*	64.7	91.3	67.4	64.9	13.2	39.5	1.5	2.8	43.2

Table 7.1: Semantic segmentation results (IoU) on Cityscapes validation obtained by DRN model trained on images generated from synthetic 3D scene graphs or/and synthetic PfD images. (*) denotes class balanced scene graphs. © 2021 IEEE

characteristics of target data. Image transfer is necessary for downstream tasks which are intended to be trained on synthesized data but used on real data.

Chapter 6 proposed a mechanism for independent content and style learning which resulted in the reduced generative power of the employed transfer mechanism. This effect retains a strong dependency of data synthesis on the quality of the manually created 3D environment as discussed in Chapter 1. The creation of a such high-fidelity environment of the video game GTA, needed for dataset *Playing for Data*, required many years of development and substantial investment budgets. This chapter attempts to circumvent this disadvantage and proposes a pipeline for controllable and salable image synthesis for realistic traffic scenes.

The proposed pipeline comprises a procedural content generation part and an appearance or texture learning part. It utilizes domain-agnostic scene representation in the middle, known as scene graphs. This representation is an alternative to commonly used photo-realistic image rendering. The proposed method utilizes the scene graphs processing introduced by Ashual et al. [67] but extends it in several ways. First, it complements the scene graphs by spatial attributes z and spatial relations such as *behind*. Furthermore, it extends image generation in an unsupervised fashion.

The proposed pipeline shows convincing generation results, which a qualitative evaluation demonstrates. Traffic scene manipulation also illustrates the effectiveness of the pipeline in image generation. The data synthesized in that way can be effectively used in a downstream task and demonstrate an improvement in their performance compared to pure synthetic data. Nevertheless, the data produced by the proposed pipeline exhibit significant incongruities and disagreement within individual classes and objects. Furthermore, several classes, such as pedestrians, are not synthesized in a feasible way. These limitations can be addressed in further research.

8 Summary

8.1 Conclusion

This dissertation explored data synthesis methodology in the field of autonomous driving. Recent recognition algorithms based on deep learning and deployed in autonomous vehicles demonstrated the necessity of abundant training data. This data have to be large-scale, error-free and have considerable variance. Simulated data offered a solution for satisfying these criteria. Nevertheless, the use of simple computer graphics commonplace for data generation exhibited limitations due to the revealed domain gap. This work is dedicated to the problem of the sim-to-real domain gap and the ways of mitigating it.

The proposed augmentation and appearance learning approach relied on the known technique of combining real-world and virtual content but employed recent advances in image transfer to alleviate the discrepancy between augmented virtual pedestrians and real traffic scenes. The method realized a pipeline with a focus on the geometrical correctness of objects' placement and their realistic appearance. The first part relied on the underlying geometry of the traffic scene in order to provide an allocation of virtual models in meaningful locations collision-free. The other part proposed class-specific adversarial training allowing synthetic pedestrians to acquire a realistic look and consistent illumination while mitigating the problem of vanishing out-of-distribution objects. The mechanism showed qualitative improvement of synthesized images as compared to mere rendering. Furthermore, quantitative evaluation on downstream tasks showed comparable performance of the augmented and real data and its interchangeability.

The work further investigated the problem exemplified by vanishing virtual objects from transferred images during the sim-to-real domain adaptation process. It analyzed the class balance statistics of several synthetic and real datasets and observed a correlation between classes exhibiting strong imbalance with semantic mismatches introduced during transfer. This assumption was further verified by calculating the importance weights of particular samples based on global class statistics and resampling the data according to these importance values. As expected, resampling solved the content corruption problem, so an adversarial network combined with importance-based densities pre-matching was proposed. This network can estimate the informativeness of specific samples on the fly and effectively perform sim-to-real domain transfer in a semantically consistent manner. The consistency of the produced images was evaluated both qualitatively and quantitatively in a downstream task of semantic segmentation. Despite its

effectiveness, the method exhibited dependency on prior knowledge about dataset global statistics.

The disentanglement further explored the problem of semantically consistent transfer in a preferred sim-to-real setting with inaccessible privileged information. The proposed technique assumed that separating learned content and style vectors enables semantic consistency. The disentanglement implies that learned image representations can be divided into domain-agnostic components interpreted as content and domain-specific component interpreted as appearance. The method attempts to sieve these feature vectors in the process of repeated inter-domain and intro-domain transformation so that the style vector only embraces the low-level style changes, therefore, preserving the images' content. The evaluation illustrated the macro consistency of the generated images but also established decreasing generative capacity of the model. Results obtained in a setting with texture-less virtual scenes demonstrated mere changes in the colour spectrum.

The final part attempted to address the trade-off between consistency and generative abundance of produced imagery and to view the synthesis pipeline from a different perspective. It proposed to replace extensive virtual environments created by computer graphic artists with procedurally generated 3D scenes only replicating its content and geometry but not textures and materials. It was suggested to synthesize images directly from these scenes using intermediate representation in the form of scene graphs. These synthetic scene graphs obtained from procedurally generated scenes allow users to manipulate generated images in a simple manner and create scenarios of interest. The proposed method relied on the established scene graph processing mechanism but extended it to an unsupervised synthetic-real setting. Additionally, it integrated spatial components available in synthetic scenes at no extra cost. The proposed solution demonstrates its effectiveness in image generation and manipulation but leaves the integrity of synthesized objects for future works.

8.2 Outlook

With the development of deep perception models and few-shot learning future recognition systems will be able to handle the broader gap between domains. This includes the gap stemming from simulation, temporal aspects, location or sensor suite. Nevertheless, accessible data generation as an alternative to physical data acquisition can still be advantageous given that future recognition models *make sense* of it. Thus, the perception models and data synthesis will develop in concert in the near future.

This work introduced several mechanisms for effectively mitigating the domain gap between synthetic and real data in the autonomous driving field. Throughout the work, it employed generative adversarial network (GAN) framework due to its good generative capabilities which allow the creation of realistic images

based on their synthetic counterparts. Proposed methods helped maintain image consistency during the adversarial translation.

Realism. Future research can further decrease the gap between generated and real data so that they potentially can replace real data acquisition and annotation. Recent advances in diffusion models can indeed make it possible but the process of generation needs to be more finely controllable, which can not even require rendered input as a condition.

Democratization. Future research needs to address the accessibility of data synthesis. Currently, the creation of virtual environments and rendering views requires substantial investments. Recently differential rendering and neural radiance field (NeRF) made significant progress in this direction providing impressive results in novel view synthesis. Thus, researchers need to focus efforts in this direction, especially on aspects of manipulation of rendered scenes preferably in automatic or randomized ways.

Safety. Improvement in the quality of data synthesis requires, though, to guarantee that generated images do not introduce harmful artefacts. The tendency of generative modelling to hallucinate things, especially in cases where particular information is absent needs to be addressed so that it does not hurt downstream tasks and road users as a result. In general, synthesis must regard safety requirements more concretely.

Human. Whether direction demonstrates more fruitful results in data synthesis humans remain part of this process in near future, therefore generation tooling is required to establish an easy interface for manual configuration. In this case, a human can cease a different role than an annotator or graphics artist, having domain knowledge this role evolves more into the high-level design of training scenarios.

Bibliography

[1] I. Pomerleau, "Alvinn: An autonomous land vehicle an a neural network," in *Advances in Neural Information Processing Systems*, 1988.

[2] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile, "Modeling the interaction of light between diffuse surfaces," in *ACM SIGGRAPH Computer Graphics*, 1984.

[3] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *European Conference on Computer Vision (ECCV)*, 2016.

[5] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[6] M. Wrenninge and J. Unger, "Synscapes: A photorealistic synthetic dataset for street scene parsing," in *Advances in Neural Information Processing Systems*, 2018.

[7] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," in *NIPS Tutorial*, 2016.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014.

[9] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[11] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," in *Journal of Machine Learning Research (JMLR)*, 2008.

[12] "Blender software." `http://blender.org`. Accessed: 2022-12-28.

[13] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[14] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *European Conference on Computer Vision (ECCV)*, 2018.

[15] R. Shu, H. H. Bui, H. Narui, and S. Ermon, "A DIRT-T approach to unsupervised domain adaptation," in *International Conference on Learning Representations (ICLR)*, 2018.

[16] J. Fleetwood, "Public health, ethics, and autonomous vehicles," in *American Journal of Public Health*, 2017.

[17] M. Taiebat, A. Brown, H. Safford, S. Qu, and M. Xu, "A review on energy, environmental, and sustainability implications of connected and automated vehicles," in *Environmental Science and Technology*, 2019.

[18] H. S. M. Lim and A. Taeihagh, "Autonomous vehicles for smart and sustainable cities: An in-depth exploration of privacy and cybersecurity implications," in *Energies*, 2018.

[19] E. Dickmanns, R. Behringer, D. Dickmanns, M. Hildebrandt, T.and Maurer, F. Thomanek, and J. Schiehlen, "The seeing passenger car 'vamors-p'," in *Proceedings of the Intelligent Vehicles '94 Symposium*, 1994.

[20] M. Bertozzi and A. Broggi, "Gold: A parallel real-time stereo vision system for generic obstacle and lane detection," in *IEEE Transactions on Image Processing*, 1998.

[21] Y. Ruichek and J. G. Postaire, "Real-time neural vision for obstacle detection using linear cameras," in *Proceedings of the Intelligent Vehicles '95. Symposium*, 1995.

[22] N. Griswold, N. Kehtarnavaz, and K. Miller, "A transportable neural network controller for autonomous vehicle following," in *Proceedings of the Intelligent Vehicles '95. Symposium*, 1995.

[23] I. Rivals, D. Canas, L. Personnaz, and G. Dreyfus, "Modeling and control of mobile robots and intelligent vehicles by neural networks," in *IEEE Conference on Intelligent Vehicles*, 1994.

[24] A. A. Kornhauser, "Neural network approaches for lateral control of autonomous highway vehicles," in *Vehicle Navigation and Information Systems Conference*, 1991.

[25] M. Rosenblum and L. S. Davis, "An improved radial basis function network for visual autonomous road following," in *IEEE Transactions on Neural Networks*, 1996.

[26] G. Yu and I. K. Sethi, "Road-following with continuous learning," in *Proceedings of the Intelligent Vehicles '95. Symposium*, 1995.

[27] M. Montemerlo, S. Thrun, H. Dahlkamp, and D. Stavens, "Winning the darpa grand challenge with an ai robot," in *AAAI Conference on Artificial Intelligence*, 2006.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012.

[29] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[30] J. L. Barron, D. J. Fleet, S. S. Beauchemin, and T. A. Burkitt, "Performance of optical flow techniques," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1992.

[31] A. Broggi, A. Fascioli, P. Grisleri, T. Graf, and M.-M. Meinecke, "Model-based validation approaches and matching techniques for automotive vision based pedestrian detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2005.

[32] G. R. Taylor, A. J. Chosak, and P. C. Brewer, "Ovvv: Using virtual worlds to design and evaluate surveillance systems," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[33] J. Liebelt, C. Schmid, and K. Schertler, "Viewpoint-independent object class detection using 3d feature maps," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[34] A. Dosovitskiy, G. Ros, F. Codevilla, A. López, and V. Koltun, "CARLA: an open urban driving simulator," in *Conference on Robot Learning (CoRL)*, 2017.

[35] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017.

[36] A. M. López, J. Xu, J. L. Gomez, D. Vázquez, and G. Ros, "From virtual to real world visual perception using domain adaptation - the DPM as example," in *Domain Adaptation in Computer Vision Applications*, 2016.

[37] S. Zhao, X. Yue, S. Zhang, B. Li, H. Zhao, B. Wu, R. Krishna, J. E. Gonzalez, A. L. Sangiovanni-Vincentelli, S. A. Seshia, *et al.*, "A review of single-source deep unsupervised visual domain adaptation," in *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[38] A. Kundu, V. Vineet, and V. Koltun, "Feature space optimization for semantic video segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[39] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," in *Pattern Recognition Letters*, 2009.

[40] A. Dundar, M.-Y. Liu, T.-C. Wang, J. Zedlewski, and J. Kautz, "Domain stylization: A strong, simple baseline for synthetic to real image domain adaptation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2020.

[41] M. Sugiyama and M. Kawanabe, *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation.* The MIT Press, 2012.

[42] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," in *International Conference on Learning Representations (ICLR)*, 2017.

[43] J. Hoffman, D. Wang, F. Yu, and T. Darrell, "Fcns in the wild: Pixel-level adversarial and constraint-based adaptation," in *Arxiv*, 2016.

[44] W. Hong, Z. Wang, M. Yang, and J. Yuan, "Conditional generative adversarial network for structured domain adaptation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[45] Y. Hong, U. Hwang, J. Yoo, and S. Yoon, "How generative adversarial networks and their variants work: An overview," in *ACM Computing Surveys (CSUR)*, 2019.

[46] M. Toldo, A. Maracani, U. Michieli, and P. Zanuttigh, "Unsupervised domain adaptation in semantic segmentation: a review," in *Technologies*, 2020.

[47] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," in *International Conference on Machine Learning (ICML)*, 2018.

[48] Y.-C. Chen, Y.-Y. Lin, M.-H. Yang, and J.-B. Huang, "Crdoco: Pixel-level domain transfer with cross-domain consistency," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[49] Y. Li, L. Yuan, and N. Vasconcelos, "Bidirectional learning for domain adaptation of semantic segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[50] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in Neural Information Processing Systems*, 2016.

[51] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, 2017.

[52] P. Li, X. Liang, D. Jia, and E. P. Xing, "Semantic-aware grad-gan for virtual-to-real urban scene adaption," in *British Machine Vision Conference (BMVC)*, 2018.

[53] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[54] X. Qi, Q. Chen, J. Jia, and V. Koltun, "Semi-parametric image synthesis," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[55] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[56] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *International Conference on Learning Representations (ICLR)*, 2018.

[57] M. C. Du Plessis and M. Sugiyama, "Semi-supervised learning of class balance under class-prior change by distribution matching," in *Neural Networks*, 2014.

[58] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data," in *Expert Systems with Applications*, 2017.

[59] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," in *Artificial Intelligence Research*, 2002.

[60] J. Zhu, "Active learning for word sense disambiguation with methods for addressing the class imbalance problem," in *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.

[61] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," in *Information Sciences*, 2013.

[62] D. B. Rubin, "Using the sir algorithm to simulate posterior distributions," in *Bayesian statistics*, 1988.

[63] R. D. Hjelm, A. J. Paul, T. Che, K. Cho, and Y. Bengio, "Boundary-seeking generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[64] M. Sinn and A. Rawat, "Non-parametric estimation of jensen-shannon divergence in generative adversarial network training," in *AISTATS*, 2017.

[65] M. Sugiyama, S. Nakajima, H. Kashima, P. v. Bünau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Advances in Neural Information Processing Systems*, 2007.

[66] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. Shamma, M. Bernstein, and L. Fei-Fei, "Image retrieval using scene graphs," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[67] O. Ashual and L. Wolf, "Specifying object attributes and relations in interactive scene generation," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[68] T. Whitted, "An improved illumination model for shaded display," in *Communications of the ACM*, 1980.

[69] G. P. Purcell, "Ray tracing on a stream processor," in *Dissertation*, 2004.

[70] F. Nicodemus, "Directional reflectance and emissivity of an opaque surface," in *Applied Optics*, 1965.

[71] J. T. Kajiya, "The rendering equation," in *ACM SIGGRAPH Computer Graphics*, 1986.

[72] D. S. Immel, M. F. Cohen, and D. P. Greenberg, "A radiosity method for non-diffuse environments," in *ACM SIGGRAPH Computer Graphics*, 1986.

[73] G. Hunt, "Detours: Binary interception of win32 functions," in *USENIX Windows NT Symposium*, 1999.

[74] M. Hicks and S. Nettles, "Dynamic software updating," in *ACM Transactions on Programming Languages and Systems*, 2005.

[75] A. Tsirikoglou, J. Kronander, M. Wrenninge, and J. Unger, "Procedural modeling and physically based rendering for synthetic data generation in automotive applications," in *Arxiv*, 2017.

[76] S. J. Pan and Q. Yang, "A survey on transfer learning," in *IEEE Transactions on Knowledge and Data Engineering*, 2010.

[77] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," in *Proceedings of the IEEE*, 2020.

[78] A. van den Oord, N. Kalchbrenner, L. Espeholt, k. Kavukcuoglu, O. Vinyals, and A. Graves, "Conditional image generation with pixelcnn decoders," in *Advances in Neural Information Processing Systems*, 2016.

[79] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *International Conference on Machine Learning (ICML)*, 2016.

[80] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations (ICLR)*, 2013.

[81] G. Alain, Y. Bengio, L. Yao, J. Yosinski, E. Thibodeau-Laufer, S. Zhang, and P. Vincent, "Gsns: generative stochastic networks," in *Information and Inference: A Journal of the IMA*, 2016.

[82] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *Advances in Neural Information Processing Systems*, 2001.

[83] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International Conference on Machine Learning (ICML)*, 2015.

[84] C. Zach, M. Klopschitz, and M. Pollefeys, "Disambiguating visual relations using loop constraints," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[85] F. Wang, Q. Huang, and L. J. Guibas, "Image co-segmentation via consistent functional maps," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013.

[86] B. Kaneva, A. Torralba, and W. T. Freeman, "Evaluation of image features using a photorealistic virtual world," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2011.

[87] M. Aubry and B. C. Russell, "Understanding deep features with computer-generated imagery," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.

[88] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison, "Real-time camera tracking: When is high frame-rate best?," in *European Conference on Computer Vision (ECCV)*, 2012.

[89] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, "A benchmark for rgb-d visual odometry, 3d reconstruction and slam," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[90] J. Shotton, R. B. Girshick, A. W. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, and A. Blake, "Efficient human pose estimation from single depth images," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013.

[91] G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, and C. Schmid, "Learning from synthetic humans," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[92] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," in *Conference on Robot Learning (CoRL)*, 2018.

[93] M. Rad, M. Oberweger, and V. Lepetit, "Feature mapping for learning fast and accurate 3d pose inference from synthetic images," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[94] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.

[95] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla, "Understanding realworld indoor scenes with synthetic data," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[96] J. Papon and M. Schoeler, "Semantic pose using deep networks trained on synthetic rgb-d," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.

[97] S. Satkin, J. H. Lin, and M. Hebert, "Data-driven scene understanding from 3d models," in *The British Machine Vision Conference (BMVC)*, 2012.

[98] B. McCane, K. L. Novins, D. Crannitch, and B. Galvin, "On benchmarking optical flow," in *Computer Vision and Image Understanding*, 2001.

[99] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," in *International Journal of Computer Vision (IJCV)*, 2011.

[100] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conference on Computer Vision (ECCV)*, 2012.

[101] N. Mayer, E. Ilg, P. Fischer, C. Hazirbas, D. Cremers, A. Dosovitskiy, and T. Brox, "What makes good synthetic training data for learning disparity and optical flow estimation?," in *International Journal of Computer Vision (IJCV)*, 2018.

[102] M. Stark, M. Goesele, and B. Schiele, "Back to the future: Learning shape models from 3d cad data," in *The British Machine Vision Conference (BMVC)*, 2010.

[103] M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, and J. Sivic, "Seeing 3d chairs: Exemplar part-based 2d-3d alignment using a large dataset of cad models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[104] B. Sun and K. Saenko, "From virtual to reality: Fast adaptation of virtual object detectors to real domains," in *The British Machine Vision Conference (BMVC)*, 2014.

[105] X. Peng, B. Sun, K. Ali, and K. Saenko, "Learning deep object detectors from 3d models," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.

[106] B. Pepik, M. Stark, P. Gehler, and B. Schiele, "Multi-view and 3d deformable part models," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015.

[107] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla, "Scenenet: Understanding real world indoor scenes with synthetic data," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[108] H. Hattori, V. N. Boddeti, K. M. Kitani, and T. Kanade, "Learning scene-specific pedestrian detectors without real data," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[109] Y. Zhang, W. Qiu, Q. Chen, X. Hu, and A. Yuille, "Unrealstereo: Controlling hazardous factors to analyze stereo vision," in *Conference on 3D Vision (3DV)*, 2018.

[110] H. Su, C. R. Qi, Y. Li, and L. J. Guibas, "Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.

[111] W. Qiu, F. Zhong, Y. Zhang, S. Qiao, Z. Xiao, T. S. Kim, Y. Wang, and A. Yuille, "Unrealcv: Virtual worlds for computer vision," in *ACM Multimedia Open Source Software Competition*, 2017.

[112] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation?," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[113] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[114] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[115] C. R. de Souza, A. Gaidon, Y. Cabon, and A. M. Lopez, "Procedural generation of videos to train deep action recognition networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[116] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick, "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[117] L. Pishchulin, A. Jain, C. Wojek, M. Andriluka, T. Thormählen, and B. Schiele, "Learning people detection models from few training samples," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[118] J. Marin, D. Vázquez, D. Gerónimo, and A. M. López, "Learning appearance in virtual scenarios for pedestrian detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[119] D. Vázquez, A. M. López, J. Marín, D. Ponsa, and D. Gerónimo, "Virtual and real world adaptation for pedestrian detection," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2014.

[120] V. S. R. Veeravasarapu, C. A. Rothkopf, and V. Ramesh, "Model-driven simulations for deep convolutional neural networks," in *Arxiv*, 2016.

[121] F. S. Saleh, M. S. Aliakbarian, M. Salzmann, L. Petersson, and J. M. Alvarez, "Effective use of synthetic data for urban scene semantic segmentation," in *European Conference on Computer Vision (ECCV)*, 2018.

[122] P. P. Busto, J. Liebelt, and J. Gall, "Adaptation of synthetic data for coarse-to-fine viewpoint refinement," in *The British Machine Vision Conference (BMVC)*, 2015.

[123] L.-C. Chen, S. Fidler, A. L. Yuille, and R. Urtasun, "Beat the mturkers: Automatic image labeling from weak 3d supervision," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[124] A. Shafaei, J. J. Little, and M. Schmidt, "Play and learn: Using video games to train computer vision models," in *CoRR*, 2016.

[125] V. Haltakov, C. Unger, and S. Ilic, "Framework for generation of synthetic ground truth data for driver assistance applications," in *German Conference on Pattern Recognition (GCPR)*, 2013.

[126] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[127] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[128] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan, "Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks?," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[129] V. Guizilini, J. Li, R. Ambrus, and A. Gaidon, "Geometric unsupervised domain adaptation for semantic segmentation," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[130] T. Sun, M. Segu, J. Postels, Y. Wang, L. Van Gool, B. Schiele, F. Tombari, and F. Yu, "Shift: A synthetic driving dataset for continuous multi-task domain adaptation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[131] B. Wymann, E. Espie, C. Guionneau, C. Dimitrakakis, R. Coulom, and A. Sumner, "Torcs, the open racing car simulator," in *http://www.torcs.org*, 2014.

[132] C. Chen, A. Seff, A. L. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[133] K.-T. Lai, C.-C. Lin, C.-Y. Kang, M.-E. Liao, and M.-S. Chen, "Vivid: Virtual environment for visual deep learning," in *ACM International Conference on Multimedia*, 2018.

[134] W. Chen, H. Wang, Y. Li, H. Su, C. Tu, D. Lischinski, D. Cohen-Or, and B. Chen, "Synthesizing training images for boosting human 3d pose estimation," in *Conference on 3D Vision (3DV)*, 2016.

[135] E. Cheung, A. Wong, A. Bera, and D. Manocha, "Mixedpeds: Pedestrian detection in unannotated videos using synthetically generated human-agents for training," in *AAAI Conference on Artificial Intelligence*, 2017.

[136] S. Huang and D. Ramanan, "Expecting the unexpected: Training detectors for unusual pedestrians with adversarial imposters," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[137] H. A. Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother, "Augmented reality meets deep learning for car instance segmentation in urban scenes," in *British Machine Vision Conference (BMVC)*, 2017.

[138] D. Lee, S. Liu, J. Gu, M.-Y. Liu, M.-H. Yang, and J. Kautz, "Context-aware synthesis and placement of object instances," in *Advances in Neural Information Processing Systems*, 2018.

[139] Li, "Aads: Augmented autonomous driving simulation using data-driven algorithms," in *Science Robotics*, 2019.

[140] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The apolloscape dataset for autonomous driving," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2018.

[141] H. A. Alhaija, S. K. Mustikovela, A. Geiger, and C. Rother, "Geometric image synthesis," in *Asian Conference on Computer Vision (ACCV)*, 2018.

[142] Y. Chen, F. Rong, S. Duggal, S. Wang, X. Yan, S. Manivasagam, S. Xue, E. Yumer, and R. Urtasun, "Geosim: Realistic video simulation via geometry-aware composition for self-driving," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[143] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez, "Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[144] Y. Zhang, P. David, and B. Gong, "Curriculum domain adaptation for semantic segmentation of urban scenes," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[145] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[146] Y. Luo, L. Zheng, T. Guan, J. Yu, and Y. Yang, "Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[147] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. K. Chandraker, "Learning to adapt structured output space for semantic segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[148] J. Wu, Z. Huang, D. Acharya, W. Li, J. Thoma, D. P. Paudel, and L. V. Gool, "Sliced wasserstein generative models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[149] Y. Chen, W. Li, and L. Van Gool, "Road: Reality oriented adaptation for semantic segmentation of urban scenes," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[150] Y. Zou, Z. Yu, B. Vijaya Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *European Conference on Computer Vision (ECCV)*, 2018.

[151] A. Almahairi, S. Rajeshwar, A. Sordoni, P. Bachman, and A. Courville, "Augmented CycleGAN: Learning many-to-many mappings from unpaired data," in *International Conference on Machine Learning (ICML)*, 2018.

[152] H. Fu, M. Gong, C. Wang, K. Batmanghelich, K. Zhang, and D. Tao, "Geometry-consistent generative adversarial networks for one-sided unsupervised domain mapping," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[153] R. Gong, W. Li, Y. Chen, and L. V. Gool, "Dlow: Domain flow for adaptation and generalization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[154] Y. Chen, W. Li, X. Chen, and L. V. Gool, "Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[155] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[156] Q. Chen and V. Koltun, "Photographic image synthesis with cascaded refinement networks," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[157] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[158] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[159] E. Schönfeld, D. Sushko, Vadim amd Zhang, J. Gall, B. Schiele, and A. Khoreva, "You only need adversarial supervision for semantic image synthesis," in *International Conference on Learning Representations (ICLR)*, 2021.

[160] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, "Contrastive learning for unpaired image-to-image translation," in *European Conference on Computer Vision (ECCV)*, 2020.

[161] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *International Conference on Machine Learning (ICML)*, 2017.

[162] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017.

[163] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *International Conference on Learning Representations (ICLR)*, 2019.

[164] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[165] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[166] A. Newell and J. Deng, "Pixels to graphs by associative embedding," in *Advances in Neural Information Processing Systems*, 2017.

[167] J. Johnson, A. Gupta, and L. Fei-Fei, "Image generation from scene graphs," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[168] H. Dhamo, A. Farshad, I. Laina, N. Navab, G. D. Hager, F. Tombari, and C. Rupprecht, "Semantic image manipulation using scene graphs,"

in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[169] A. Borji, "Pros and cons of gan evaluation measures," in *Computer Vision and Image Understanding*, 2019.

[170] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, 2016.

[171] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017.

[172] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey, "Characterizing adversarial subspaces using local intrinsic dimensionality," in *International Conference on Learning Representations (ICLR)*, 2018.

[173] F. Yu, V. Koltun, and T. Funkhouser, "Dilated residual networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[174] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.

[175] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask r-cnn," *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[176] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014.

[177] I. Krešo, D. Čaušević, J. Krapac, and S. Šegvić, "Convolutional scale invariance for semantic segmentation," in *German Conference on Pattern Recognition (GCPR)*, 2016.

[178] G. Ros, S. Ramos, M. Granados, A. Bakhtiary, D. Vazquez, and A. M. Lopez, "Vision-based offline-online perception paradigm for autonomous driving," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015.

[179] D. B. Rubin, "The calculation of posterior distributions by data augmentation: Comment: A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The sir algorithm," in *Journal of the American Statistical Association*, 1987.

[180] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," in *International Journal of Computer Vision (IJCV)*, 2015.

[181] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Advances in Neural Information Processing Systems*, 2017.

[182] T. Park, J.-Y. Zhu, O. Wang, J. Lu, E. Shechtman, A. A. Efros, and R. Zhang, "Swapping autoencoder for deep image manipulation," in *Advances in Neural Information Processing Systems*, 2020.

[183] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2014.

[184] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[185] T. Roden and I. Parberry, "From artistry to automation: A structured methodology for procedural content creation," in *International Conference on Entertainment Computing (ICEC)*, 2004.

[186] Y. I. H. Parish and P. Mueller, "Procedural modeling of cities," in *SIGGRAPH Annual Conference on Computer Graphics and Interactive Techniques*, 2001.

[187] B. Poore and S. Coast, *The Book of OSM.* CreateSpace Independent Publishing Platform, 2015.

[188] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *International Conference on Artificial Intelligence and Statistics*, 2010.

[189] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning (ICML)*, 2020.

[190] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving video database with scalable annotation tooling," 2018.