DISSERTATION

submitted

to the

Combined Faculty of Natural Sciences and Mathematics

of

Heidelberg University, Germany

for the degree of

Doctor of Natural Sciences

Put forward by

M.Sc. Hassan Abu Alhaija

Born in: Damascus, Syria

Oral examination:

Deep-learning based image synthesis with geometric models

Advisor: Prof. Dr. Carsten Rother

Acknowledgements

I would like to first thank my supervisor, Prof. Carsten Rother, for his continuous support, scientific and career guidance, and most importantly for believing in me through out the years of this work. None of this would have been possible without him.
My thanks go also to Prof. Andreas Geiger with whom I was fortunate to collaborate through most of my thesis. He is a role model for me not only for his clear thinking about research questions, but also for his genuinely human and caring leadership approach. I am also grateful to Prof. Vladlen Koltun, Prof. Matthias Nießner, Dr. Justus Thies and Dr. Stephan Richter for the opportunity to work with them during my internships at Intel and TUM, to Dr. Daniel Kondermann and Dr. Anita Sellent for picking me up as a naive Master student and showing me the door into the research world, and to our amazing secretary Antje König who guided me through the bureaucracy jungle. And for making the past years full of laughs, unforgettable moments, and legendary kicker games, I say *thank you very much* to my lab friends Omid, Weihao, Siva and Sid.
Mostly I am thankful to my family. To my parents for what they sacrificed and went through to get me to this point is hard to imagine and impossible to repay. To my sisters for their love and brave souls have always inspired me to move forward.
Finally, I want to thank the people of Germany for opening their borders, homes and hearts for my fellow Syrian people escaping the horrors of war. Giving a second chance to hundreds of thousands of people who lost all hope is an immense act of humanity that shall not be forgotten.

Abstract

Data-driven machine learning approaches have made computer vision solutions more robust and easily adaptable to various circumstances. However, they are often limited by their dependency on large datasets with accurate ground-truth annotations for training. In most scene understanding tasks, like instance segmentation and object detection, training data is often scarce since annotations can not be measured directly from the real world using sensors but instead have to be manually created by humans at large cost. Virtual scenes could offer a feasible alternative in these cases since the full access to the underlying scene geometry enables generating fast and accurate annotations. However, scene understanding models trained on rendered images often do not perform well on real test images due to the difference in appearance between the synthetic and real images.

This thesis proposes several new methods for images synthesis with focusing on generating training images that could partially or totally replace real data for training deep-learning models. It first explore the use of augmented reality techniques for combining synthetic 3D objects and real scenes. This can greatly reduce the effort needed for generating diverse training scenes with accurate annotations. We study and compare the effect of various factors of image generation on the performance of the trained scene understanding models. To overcome the limitations of rendering engines, we next propose a novel geometric image synthesis approach that generates geometrically consistent and controllable images. The deep neural network learns to imitate the rendering process while at the same time optimizing for an explicit realism objective making the resulting images more suitable to train scene understanding models. Finally, to alleviate the need for rendered images, we introduce an unsupervised neural rendering model trained only using unpaired 3D models and real images of similar object class. This is achieved by learning the forward rendering and backward decomposition processes jointly. The results in this thesis indicate that deep-learning based image synthesis models could be an efficient tool for generating realistic images and high-quality synthetic training data.

Zusammenfassung

Datengesteuerte maschinelle Lernansätze haben Computer-Vision-Lösungen robuster und leichter an verschiedene Umstände anpassbar gemacht. Sie sind jedoch oft eingeschränkt durch ihre Abhängigkeit von großen Datensätzen mit genauen Ground-Truth-Annotationen, die für das Training benötigt werden. Bei den meisten Szenenverstehensaufgaben, wie z.B. der Instanzsegmentierung und Objekterkennung, sind die Trainingsdaten oft knapp, da Annotationen nicht direkt aus der realen Welt mit Hilfe von Sensoren gemessen werden können, sondern stattdessen manuell von Menschen mit großem Aufwand erstellt werden müssen. Virtuelle Szenen könnten in diesen Fällen eine praktikable Alternative bieten, da der volle Zugriff auf die zugrundeliegende Szenengeometrie die Erzeugung schneller und genauer Annotationen ermöglicht. Allerdings führt die Benutzung von Szeneverständnismodellen, die auf gerenderten und daher synthetischen Bildern trainiert wurden, bei realen Testbildern aufgrund des unterschiedlichen Aussehens zwischen den synthetischen und realen Bildern oft zu keinen guten Ergebnissen.

In dieser Arbeit werden mehrere neue Methoden für die Bildsynthese vorgeschlagen, wobei der Schwerpunkt auf der Erzeugung von Trainingsbildern liegt, die reale Daten für das Training von Deep-Learning-Modellen teilweise oder ganz ersetzen können. Zunächst wird die Verwendung von Augmented-Reality-Techniken zur Kombination von synthetischen 3D-Objekten und realen Szenen untersucht. Dies kann den Aufwand für die Generierung diverser Trainingsszenen mit genauen Annotationen stark reduzieren. Wir untersuchen und vergleichen die Auswirkung verschiedener Faktoren der Bilderzeugung auf die Leistung der trainierten Szenenverstehensmodelle. Um die Einschränkungen von Rendering-Engines zu überwinden, schlagen wir als nächstes einen neuartigen Ansatz zur geometrischen Bildsynthese vor, der geometrisch konsistente und kontrollierbare Bilder erzeugt. Das tiefe neuronale Netzwerk lernt, den Rendering-Prozess zu imitieren, während es gleichzeitig für ein explizites Realismus-Ziel optimiert wird, wodurch die resultierenden Bilder besser geeignet sind, um Modelle zum Verstehen von

Szenen zu trainieren. Um den Bedarf an gerenderten Bildern zu verringern, führen wir schließlich ein unbeaufsichtigtes neuronales Rendering-Modell ein, das nur mit ungepaarten 3D-Modellen und realen Bildern ähnlicher Objektklassen trainiert wird. Dies wird durch gemeinsames Lernen der Vorwärtsrendering- und Rückwärtszerlegungsprozesse erreicht. Die vorliegende Arbeit zeigt, dass Deep-Learning-basierte Bildsynthesemodelle ein effizientes Werkzeug zur Erzeugung realistischer Bilder und hochwertiger synthetischer Trainingsdaten sein könnten.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The modern approach to Artificial Intelligence is largely centered around using Machine Learning methods that utilize large collections of data to create highly adaptive solutions to difficult problems. In Computer Vision, the importance of learning feature extraction and visual perception models from data became especially visible after deep neural networks quickly overtook classical hand-crafted methods in performance on a wide range of visual tasks. Indeed, the flexibility of learning-based approaches opened new possibilities and accelerated the development in previous tasks since the same learning algorithm developed for one task could be quickly adapted to work for another task by changing its training data. But to better understand the relation between visual perception and learning, it is worth to ask first a more fundamental question: *Is visual perception in Humans learned or innate?*

The Human Visual System starts with the eyes which focus light coming from the environment on light sensitive cells in its retina. The first image processing already occurs here by the bipolar, horizontal and ganglion cells which aggregate the signal from those 100 million photo-receptors into around 1 million axons that transmit the information to the brain. However, most visual information processing that we consider as "seeing", like depth perception, motion detection, and object recognition, happens in the visual cortex of the brain. But the question of whether these abilities are innate or learned by experiencing the world has captured the interest of many philosophers,

psychologists and scientists over the centuries. The 12th century Andalusian philosopher Ibn Tufail alluded to the concept of learnable vision [200] which was developed by William Molyneux [174] in the 17th century into the following thought experiment: If a blind man who can feel the differences between shapes such as a sphere and a cube was suddenly able to see, could he distinguish them by sight alone? The British empiricist philosopher John Locke's (1689) answer to this question was "Not. For, though he has obtained the experience of how a globe, how a cube affects his touch, yet he has not yet obtained the experience, that what affects his touch so or so, must affect his sight so or so" [115]. This view that perception can only be directly learned from sensation and experience was the opinion of many empiricist philosophers and psychologists like Barkley (1709), Hobbes (1651), Hume (1758) and William James (1890) [94]. Starting in the 1950s, modern developmental psychology and cognitive science opened the doors to scientifically investigation of the question of learnable vision through the study of perception of human infants. The work of James and Eleanor Gibson [57] challenged the view that visual perception is actively learned and instead suggested that perception depends on hardwired features in the brain constructed through evolution. This theory was later advanced in computational approaches to perception by Marr [124]. The modern view of human visual perception is that it developes from innate foundations. But learning plays an important role in improving and calibrating some abilities while can be key in developing some others [94].

The invention of digital imaging combined with the rapid increase in power and availability of computation in 1970s led to a large interest in visual perception concepts but with a new computational perspective. Thus, the research field of Computer Vision was established with the goal of imitating this human ability to understand visual information using computational systems. During the earlier years of its history, many researchers focused on crafting physically-based computational models for reconstructing a scene and its properties from images. Computer Vision has been largely seen during this time as "inverse graphics" where methods like shape-from-shading [76] and shape-from-stereo [210] attempted to use the physical knowledge about the image formation process to reverse-engineer the scene from one or few images. This, however, often required making several simplifying assumptions about the imaging process and

scene structure like pin-hole camera models, calibrated noise-free image sensors, smooth geometry, single directional lighting, Lambertian materials and many others. Those conditions are rarely met in real images taken "in the wild", therefore limiting the usability of such hand-crafted models. Additionally, this meant that new solutions and algorithms had to be invented for each task and scenario which greatly slowed down progress.
To deal with the large variability and complexity of real images, researchers instead turned to supervised machine learning models. Instead of exploiting knowledge of the task and scene structures, this class of learning-based algorithms relies heavily on labeled training examples to build a good prediction model of target scene properties that can generalize to new images. This approach pushed forward the state-of-the-art performance on many computer vision tasks sustained by the continuous improvements in Deep Neural Network architectures, the large increase in dataset sizes and the wide availability of parallel computational power through the use of Graphical Processing Units (GPUs). Data-driven machine learning models helped advance computer vision from engineering single solutions to building more generic algorithms that can easily be adapted to new tasks and circumstances [127]. However, their Achilles' heel remained in their dependance on accurately labeled training data.

1.1 Motivation

1.1.1 Data is the new algorithm

The performance of deep learning models is strongly dependant on three factors: computational power, the model size and the amount of training data. Recent experimental studies in deep language models [88] show that each one of those three factors consistently follows a power-law relation to the performance, when not bottle-necked by the other two. A similar study [188] on deep vision models unveiled that the power-law relation between the dataset size and performance still holds even when increasing the dataset size by two-orders of magnitude while keeping the model size and compute power fixed. This suggests that even with current vision models, the available training data is still the limiting factor. What makes scaling datasets in computer vision

difficult is that creating ground-truth annotations is often done manually by humans, which means the cost of labeling larger training datasets grows linearly while their performance impact only grows logarithmically. For some tasks like image classification, crowd-sourcing methods were especially successful since labeling millions of real images [170] manually or hundreds of millions [33, 73] semi-automatically is still feasible. However, annotating images for tasks which require dense pixel-wise predictions, like motion estimation or intrinsic image decomposition, is very laborious and sometimes infeasible for human annotators. Synthetic data generation using computer graphics is increasingly used as an alternative to create accurate training data automatically at scale (see Fig. 1.1). However, two factors limit it from being a solution to the data scarcity problem. First, rendering has been historically tuned to produce visually appealing rather than physically accurate images, which can introduce strong bias into the learned vision model. Secondly, the cost of creating realistic and rich simulated worlds is very large and hard to scale. For example, the cost of creating video games has grown exponentially in the past 20 years [102] and can reach hundreds of millions of dollars [183] for a single virtual city. One motivation for this thesis is to alleviate these limitations by exploring new paradigms of image synthesis that can close the realism gap between synthetic and real images. It is believed that scalability of training data is one of the major challenges facing deep vision models in the future and using image synthesis techniques presented in this thesis could be an important step toward making larger and more powerful models possible.

1.1.2 Autonomous driving in the real world

This thesis largely focuses on scene understanding tasks in the context of automotive scenarios. Autonomous driving is one of the most promising and yet difficult applications of machine learning and computer vision in the next decades. The data challenge here is not only limited to the amount of annotated training images, but also in the diversity and controllability needed to build robust and safe driving systems. Critical and dangerous scenarios can be extremely rare in real data, yet they are the most crucial since the system's reaction to critical events like accidents is of highest importance. Fig 1.2 gives

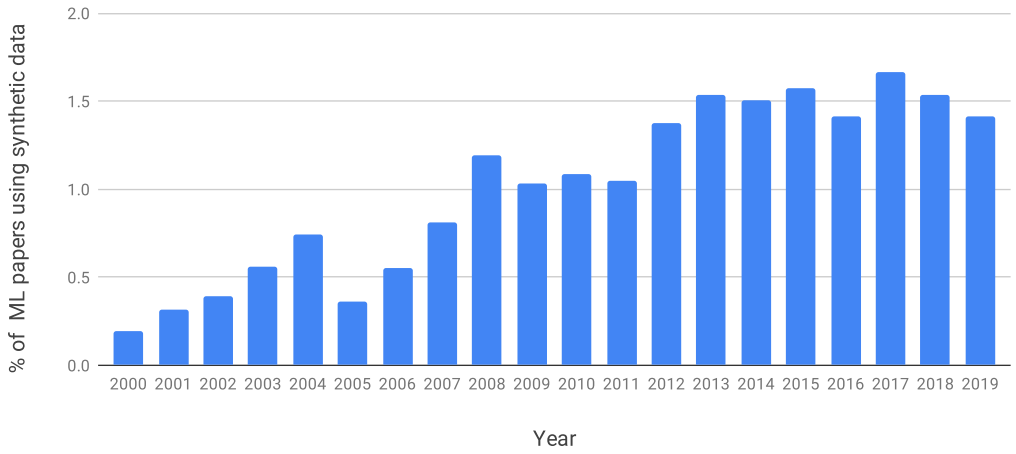


Figure 1.1. The percent of published machine learning research papers using the terms ”synthetic data” in title or abstract according to ScienceDirect database (data from [198]).

some examples of rare driving situations that humans can easily deal with but can pose a serious challenge for autonomous driving models trained on common driving datasets. Validation against such dangerous scenarios is essential when failures can be fatal but it is only feasible through realistic and controllable synthetic data. A recent study estimated that to achieve 95% confidence that an autonomous driving system has reached human-level failure rate, it needs to be tested for 440 million kilometers [87, 100]. Even with a test fleet of 100 cars driving 24 hours at 60 km/h, each system would require around 8 years to be validated. This strongly indicates that simulations are going to be an integral part of bringing autonomous driving systems safely into the real world.

1.1.3 The Realism Gap

While traffic modeling and physical simulation systems are relatively mature fields, the problem of generating realistic images from simulated environments has not been solved yet. Classical rendering algorithms, which have been engineered and tuned to produce visually pleasing results for humans, often fail to produce good training data for scene

(a) examples from Mapillary [142] (left) and Cityscapes [34] (right) driving datasets.

(b) Examples of rare but dangerous scenarios in real life.

Figure 1.2. Autonomous driving datasets often focus on average common driving scenes and traffic situations. However, some rare situation in the real world can pose a serious and maybe dangerous challenge for autonomous driving systems trained only on common driving scenes.

understanding deep learning models. One motivation of this thesis is to explore the causes behind this difference and test possibility of learning the rendering process using a neural network. Combining data-driven neural rendering methods with an explicit realism loss can help reduce the performance gap between real and synthetic training data.

1.2 Contributions

The contributions in this thesis are summarized here:

- We propose an alternative paradigm for generating training and test data for learning semantic instance segmentation and object detection models which combines real and synthetic data. Exploiting the fact that not all aspects of the scene are equally important for target tasks, we propose to augment real-world imagery with virtual objects of a target category. Specifically, we develope an efficient procedure for augmenting urban driving images with virtual car objects.

- We analyze the significance and contribution of different aspects of the image generation and augmentation process on the generated training data quality. This includes rendering quality, background realism, post-processing and realistic object placement in augmented scenes.

- We propose a trainable, geometry-aware image generation method that leverages various types of scene information, including geometry and segmentation, to create realistic looking natural images that match the desired scene structure. Our geometrically-consistent image synthesis method is a deep neural network which retains the advantages of a trainable method, e.g. differentiability and adaptiveness, but, at the same time, makes a step towards the generalisability, control and quality output of modern graphics rendering engines. We utilize this framework to insert vehicles in outdoor driving scenes, as well as to generate novel views of objects from the different datasets. We qualitatively show that our network is able to generalize beyond the training set to novel scene geometries, object shapes and segmentation. Furthermore, we quantitatively show that our framework can be used to synthesize large amounts of training data which proves beneficial for training instance segmentation models.

- We propose a novel approach to train a neural deferred renderer using unpaired 3D geometry and 2D appearance-images. We train a joint model for both generating realistic images from synthetic 3D models and the inverse problem of getting

intrinsic properties like shape and albedo from real images. We demonstrate that training the two inverse tasks jointly using unpaired real and synthetic datasets can improve in both. In contrast to a traditional graphics pipeline, this approach does not need to specify all remaining object properties, such as material parameters and lighting, which often require expert skills.

- We introduce two new datasets to help evaluate the proposed synthetic augmentation and image synthesis methods:

  - **KITTI-2015 semantic, instance and panoptic segmentation dataset** is an extension of the popular KITTI dataset and benchmark [131] into 3 new segmentation tasks. Semantic and instance level image annotations of 200 training images and 200 test images from the original KITTI-2015 dataset were labeled through crowd-sourcing services and were then carefully inspected for quality control. The training data and a public evaluation benchmark were made publicly available as part of the KITTI benchmark. The panoptic segmentation [99] labels were created by combining the semantic and instance segmentation. Additionally, I contributed in organizing two editions of the *Robust Vision Challenge* [221, 222] which included this new dataset as part of the multi-dataset benchmark to test robustness of computer vision algorithms.

  - **KITTI-360** is a dataset specialized for developing and testing virtual object augmentation in driving datasets. It consist of two parts:

    ◇ A set of 28 3D models of cars with 32 high quality physically-based materials including 16 different car paint materials and scripts for generating realistic rendered and post-processing images.

    ◇ A set of 200 image from the KITTI-360 [214] manually annotated with accurate car instance masks with matching 360° environment maps.

1.3 List of published research papers

The remaining chapters of this thesis are based on the following published research papers.

- **Alhaija, H. A.**, Mustikovela, S. K., Mescheder, L., Geiger, A., & Rother, C. *Augmented reality meets deep learning for car instance segmentation in urban scenes.* In British machine vision conference (BMVC) 2017 [3].
- **Alhaija, H. A.**, Mustikovela, S. K., Mescheder, L., Geiger, A., & Rother, C. *Augmented reality meets computer vision: Efficient data generation for urban driving scenes.* International Journal of Computer Vision (IJCV), 2018 [2].
- **Alhaija, H. A.**, Mustikovela, S. K., Geiger, A., & Rother, C. Geometric image synthesis. In Asian Conference on Computer Vision (ACCV) 2018 [1].
- **Alhaija, H. A.***, Mustikovela, S. K.*, Thies, J., Jampani, V., Nießner, M., Geiger, A., & Rother, C. Intrinsic *Autoencoders for Joint Deferred Neural Rendering and Intrinsic Image Decomposition.* In International Conference on 3D Vision (3DV) 2020 [4].
  (*: Equal contributions)
  Declaration: The idea for this paper and the initial implementation was done by me. Siva Karthik Mustikovela led the first stage of experiments and development while I was doing an internship. After returning, I led the second stage of experiments and development. The paper writing has been done jointly with him.

Additionally, I contributed to the following related research papers during working on this thesis. They will not be discussed here.

- Swoboda, P., Rother, C., **Alhaija, H. A.**, Kainmuller, D., & Savchynskyy, B. *A study of lagrangean decompositions and dual ascent solvers for graph matching.* In Computer Vision and Pattern Recognition (CVPR) 2017 [189].
- Behl, A., Hosseini Jafari, O., Karthik Mustikovela, S., **Alhaija, H. A.**, Rother, C., & Geiger, A. *Bounding boxes, segmentations and object coordinates: How*

important is recognition for 3d scene flow estimation in autonomous driving scenarios? In Proceedings of the IEEE International Conference on Computer Vision (ICCV) 2017 [12].

- Richter, S., **Alhaija, H. A.**, Koltun, V., *Enhancing photorealism enhancement* ArXiv 2021 [163].

1.4 Outline of The Thesis

The remaining part of this thesis is structured as follows: In Chapter 2 we present the current strategies for generating real and synthetic training data and discuss the challenges facing them. We next give a background on the growing field of neural image synthesis and its potential as a competitor for the classical rendering pipeline. In Chapter 3 we introduce a novel augmented-reality-based method for generating training images with accurate ground-truth annotations in urban driving scenarios. Here we study the effect of various factors of image generation on the training performance of the generated augmented data. Chapter 4 introduces Geometric Image Synthesis, a neural network model for generating geometry-consistent images from input 3D models with an explicit realism loss. Next, in Chapter 5 we present a novel unsupervised method for jointly learning neural rendering and intrinsic image decomposition from unpaired sets of 3D models and 2D images. Finally, we conclude in Chapter 6 with a discussion of the contributions in this thesis and an outlook of possible future directions.

Chapter 2

Background

2.1 Real data for deep learning

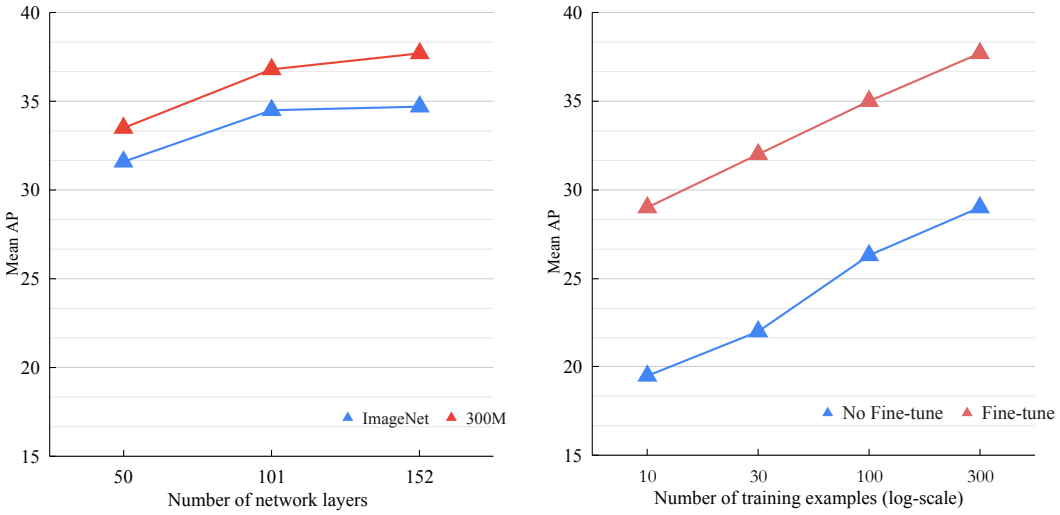
In statistical learning theory [202], the goal of a learning algorithm is to use a training set of input/output pairs to find a function that can best predict the output of new unseen input samples called a test set. This definition implicitly includes two important assumptions: First, that both training and test sets are sampled i.i.d.from the same data distribution. And second, that the training set is accurate, large and diverse enough to represent the full complex data distribution. In computer vision applications, the first condition implies that using real training images is preferred since the ultimate goal is to predict some properties of a real scene from one or few images of it. However, the second condition is much harder to satisfy using real data since real training images are usually labeled manually by humans which puts a limit on their accuracy and scalability due to high cost of annotation. Crowd sourcing platforms, like Amazon Mechanical-Turk, have greatly reduced the cost of labeling large image datasets by distributing the task among a wide population of workers. However, this human-driven approach to training data generation has its limits and drawbacks. This section describes three major challenges facing manual annotation as a way to create training data and a popular method to cheaply expand computer vision training sets through image transformation.

2.1.1 Challenges

Scale. Recently proposed Deep neural networks architectures have shown a clear trend toward increasing the number of network layers and therefore the number of learnable parameters of the model. Just in the past 7 years the number of parameters in the state-of-the-art image classification models has increased from tens of millions [182] to hundreds of millions [78] to billions [179]. This increase in model capacity, without strong over-fitting, was only possible due to the relatively large available datasets for image classification, namely ImageNet dataset [170]. Moreover, Sun *et al*. [188] have shown that using a dataset 300 times bigger can still yield improved performance for large image classification models. The catch, however, is that the performance improvement only grows logarithmically with the size of the dataset. Figure 2.1 shows how an order-of-magnitude more data is needed for a linear improvement in performance. Even with the reduced cost of annotation through crowd sourcing, the increase in labeling costs is still linear with the number of labeled examples which makes manual labeling a very cost-ineffective method for improving a model performance beyond a certain threshold.

Feasibility and accuracy. Computer vision tasks can be very different in the time and effort needed for labeling an image by a human annotator. This can vary largely between tasks like image classification where a single choice of label per example is needed, to dense annotation tasks like semantic and instance segmentation which can take up to 2 hours of annotation time per image [34]. Such difficulties in labeling can also reduce the accuracy of the results either due to the ambiguity of the task for humans (Fig 2.2b) or disagreement between different human annotators on the solution (Fig 2.2a). Moreover, some tasks require special skills or knowledge for annotation like 3D object reconstruction or bio-imaging applications. Other tasks are not even feasible for humans to manually annotate like dense depth or motion estimation.

Control and rare scenarios. Manually labeling images randomly collected from the real world is a good way to ensure the training dataset is unbiased and reflects the true data distribution. This unbiased approach operates under the assumption that the dataset size is large enough to cover all real world scenarios with enough samples. In some applications, however, critical scenarios can be more important to learn for a model



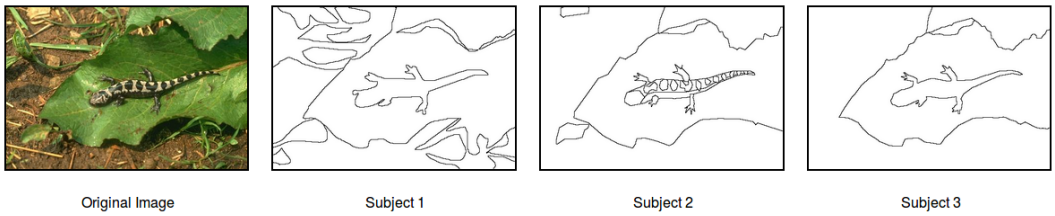
(a) Semantic segmentation **(b)** Object detection

Figure 2.1. Results from [188] show how the performance of object detection on the the COCO minival dataset changes depending on training set size and model capacity. (a) When the model is pre-trained on subsets of the the JFT-300M dataset, the performance improves linearly with the exponential increase of the training set size. (b) The higher capacity models are better at utilizing the larger JFT-300M dataset compared to ImageNet.

while being very rare in the real data. Collecting a large dataset of such critical data can be difficult, dangerous or costly. For examples, in Autonomous Driving applications the real training data is usually collected by driving cars in real traffic with high level of safety measures to avoid accidents. However, examples of accidents are crucial for training a safe and reliable Autonomous Driving system that is able to deal with rare but dangerous situations, which creates an ethical paradox.

2.1.2 Image Transformations for Data Augmentation

A common way to improve the effectiveness of a training dataset is to exploit the knowledge of the data invariance by artificially introducing transformations that alter the data without invalidating its ground-truth label. These data augmentation techniques are often used to reduce the over-fitting problem especially when the dataset size is small



(a) Berkeley Segmentation Data Set (BDSD500) [7]

(b) KITTI-2015 segmentation dataset examples.

Figure 2.2. Challenges with human annotated datasets. (a) Shows the different results given by different human workers when asked to segment the image from the BDSD dataset [7]. (b) Shows a difficult situation in two consecutive images in the KITTI-2015 dataset where the human annotator labels the wall on the right as "building" in the first and "wall" in the second. Similar inconsistencies can be noticed for the forklift in the center, the lighting poles, and car group in the back.

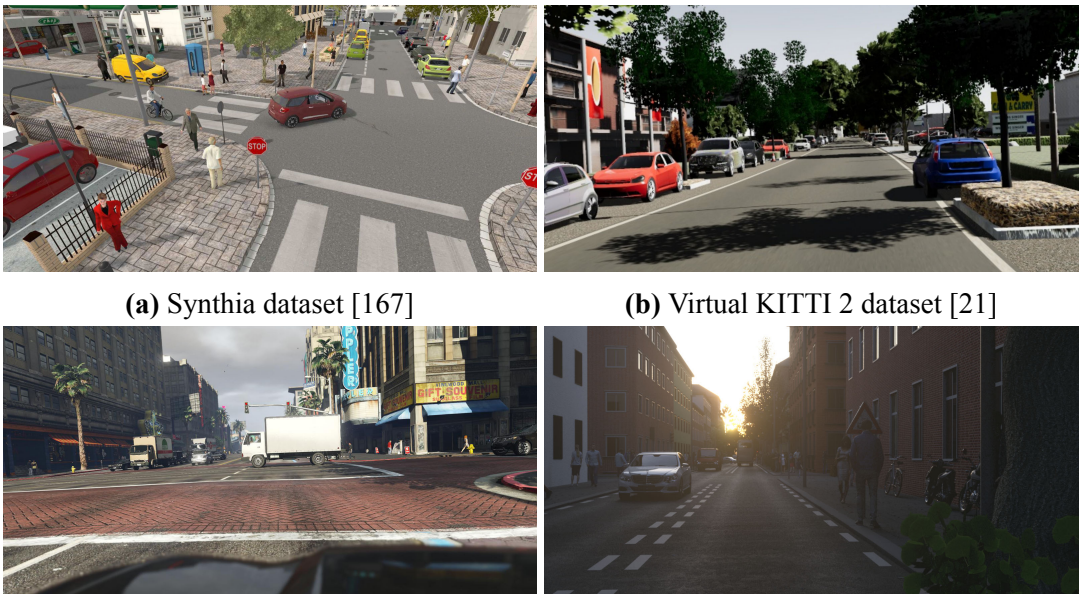
compared to the model capacity. For example, Krizhevsky *et al*. [104] reported in their landmark work on ImageNet classification reported over 1% performance improvement by using random cropping, horizontal flipping and simple color shifting during training. Since then, a large variety of data augmentation operations have been proposed in the literature including adding image noise [140], random style transfer [81] and random erasing [227]. The goal of the data augmentation step is not to increase the dataset size but rather improve the quality of the information extracted from it by the model. Therefore the expected benefit from the augmented samples is much smaller than from

new independent data samples and the performance increase is limited by the variance already existing in the dataset.

2.2 Synthetic data for deep learning

A possible alternative to annotating real training images is to use an exact or approximate process to generate input/output pairs for train a machine learning model. This can be especially applicable when the process of generating an input from an output is easier that predicting the output from the input. For example, generating gray scale or low resolution images from high resolution colored ones is an easy and exact process which can be used to generate a theoretically unlimited amount of training data for the tasks of image colorization and super-resolution. However these situations are the exception. And in most scene understanding tasks, the input is one or few images of a scene and the output is some unknown label or information about the scene that is non-trivial to compute from the input, e.g. semantic segmentation. In these cases, an exact automated process of generating real training data is not defined. Instead, an approximate process for generating images with ground-truth labels can be used to build synthetic training sets that could replace or enrich manually labeled real training data. This synthetic data generation approach has gained more importance recently (see Fig. 1.1) because it offers two main advantages. First, the cost of setting up the synthetic training data generation process is often constant and independent of the amount of generated data. This makes it more cost efficient for creating large training datasets for deep learning applications. Second, the generated ground-truth annotations are usually accurate, consistent and detailed.

In the following we discuss the main 3 types of Synthetic data generation strategies and their challenges.



(a) Synthia dataset [167] **(b)** Virtual KITTI 2 dataset [21]

(c) Playing-for-benchmarks dataset [164] **(d)** SynScapes dataset [211]

Figure 2.3. Examples of several synthetic datasets for autonomous driving applications.

2.2.1 Classical rendering

A rendering algorithm can be defined in general as the process of generating an image from a scene definition which can include geometry, materials, lights and a camera model [193]. Most classical rendering algorithms follow one of two approaches: *Rasterization* methods project the 3D objects in the scene onto the image grid and compute their visibility and appearance based on their position in space. These methods are largely used in real-time graphics due to their simplicity and possibility to be easily parallelized. *Ray tracing* methods compute the color value at each pixel by sending multiple ray samples from the camera center through the pixel into the scene and then recursively track those rays as they bounce and interact with different objects and materials in the scene until they hit a light source or reach the maximum number of bounces. Such techniques are much more computationally demanding but can produce very realistic-looking images especially when combined with physically-based material models. These models are often used in off-line rendering applications like animation movies and visualization. Most recently, Physically-based rendering [154] techniques where able to achieve

Training Data Type	Dataset	FRRN Mean IoU	DeepLab Mean IoU
Synth	Synthia[167]	21.78%	32.73%
	GTA[164]	20.88%	32.20%
	SynScapes[211]	45.20%	50.35%
Domain Gap			
Real	Cityscapes (CS)	68.27%	76.56%
Synth+Real	Synthia+CS	69.89%	77.45%
	GTA+CS	70.76%	77.57%
	SynScapes+CS	74.52%	78.85%

Table 2.1. The domain gap between real and synthetic training data. The results from [211] show the difference in performance of two semantic segmentation models (FRRN [157] and DeepLab [26]) when trained on synthetic, real or real+synthetic datasets and tested on the real Cityscapes [34] validation set.

unprecedented levels of photorealism by deeply integrating the physical properties of matter and building accurate models of light-matter interaction [154].

In computer vision, one of the first uses of virtual scene and rendering was to create synthetic images with accurate optical flow [10, 69] for the purpose of evaluating different optical flow estimation algorithms. Similarly, the MPI-Sintel dataset [20] used an open-source 3D movie in Blender [15] to create a synthetic dataset for training and benchmarking optical flow estimation models. The increasing popularity of deep-learning models in computer vision applications led to an increase in using virtual data and classical rendering to create large synthetic datasets that bridge the gap between the data-demanding machine learning models and the scarcity of real training data in many domains. In the automotive driving scenario, projects like VirtualKITTI [21, 50], SYNTHIA [167], CARLA [43] and Synscapes[211] all built custom virtual urban environments and used different simulation and rendering techniques to create realistic and accurate training data with full control over the scene contents. Others [5, 86, 164, 165] leveraged already existing video game environments to capture training data. For indoor scenes, synthetic datasets like SUNCG[184] and SceneNet[128] have helped accelerate research in indoor scene understanding and robotic navigation tasks since collecting real data in this domain is often hard due to either privacy concerns or technical difficulties in 3D

scanning indoor environments. Other projects like AI Habitat[123], Gibson[212] and AI2-THOR[101] also use fully simulated indoor virtual scenes to move from "Static AI", i.e. training machine learning system on fixed static datasets like ImageNet, to "Embodied AI" where the learning algorithms is trained through its continuous interaction with the environment. Synthetic data also plays a large role in many other domains of computer vision research like human body reconstruction [118, 159, 203] and UAV navigation [49, 178]. Despite its wide adaptation, synthetic image data generation using classical rendering faces several significant challenges.

2.2.2 Challenges

Virtual scene construction. An unlimited number of images can theoretically be rendered from a virtual scene just by moving the camera position or shuffling the poses and colors of objects in the scene. This simple method for creating variance and diversity in synthetic data can be enough for some tasks like optical flow [127]. However, for most scene understanding tasks, like Autonomous driving or indoor navigation, the context of the object can be as important as its appearance. This requires virtual scenes to be carefully designed and constructed to accurately imitate the complex and variance of the real world in order for the rendered training data to be useful, which is costly and time-consuming. For this reason, virtual urban driving environments like VirtualKITTI [21, 50], Synthia [167] and CARLA [43] often include only a handful virtual towns with only a few blocks and highly repetitive building templates. One proposed solution has been to reuse the large complex virtual world from established video games like Grand Theft Auto [5, 86, 164, 165] for generating synthetic training data. This can greatly enrich the variety and quality of the data. But it comes at the cost of less control over the environment since the data is mostly collected by reverse engineering the game rendering pipeline without full access to its assets and simulation engine.

Rendering photorealism. To create perfect training data for a learning algorithm, a synthetic data generator should mirror the real data generation process to avoid introducing any unwanted biases. This means that a rendering engine would have to accurately simulate the physical light-matter interaction process on a sub-atomic level in order to

reproduce all the optical phenomena that are captured in a real image. This is computationally infeasible and limited by our understanding of the physics of light-matter interaction. Instead, the long-standing goal of most rendering algorithms has been to achieve *photorealism*, defined as producing images that are indistinguishable to a human observer from real photographs. This focus on perceived realism rather than physical accuracy allows them to exploit the limitation of the human visual system and employ several approximations and heuristics to improve computational efficiency without sacrificing photorealism. Deep neural networks, on the other hand, learn to extract useful features from the training images directly. This means they can pick-up on various optical effects, patterns and artifacts unique to rendered images that are not noticeable to the human eye. Experimental results in Table 2.1 show how state-of-the-art semantic segmentation models trained using only real-time rendered images from the GTA [164] and Synthia [167] datasets fail to achieve the same performance on real test data compared to networks trained using real training images. Using more advanced ray tracing and physically-based rendering techniques, SynScapes dataset [211] is able to achieve very high level of photorealism (2.3) and significantly improve training performance compared to real-time rendering result. However, it still falls short when compared to training on real images even when using a much higher number of rendered images. These results indicate that the human-based photorealism of rendered images is not enough to predict their effectiveness as training data for deep-learning models. And a new more objective way to measure the quality of rendered images as training data is needed.

Imaging artifacts. Another source of disparity between real and rendered images is the presence of noise and aberrations in real images caused by the the characteristics and limitations of real camera systems. Meanwhile, rendering engines mostly use the pin-hole camera model which assumes that light rays coming from the scene pass through an infinitely small pin-hole and transfer the scene information perfectly onto the image without any distortions. While the pin-hole camera model is simple, using it to take real photos is not practical since it would require an extremely strong source of light in the scene to project a bright and focused image on the sensor. Instead, a real camera uses a complicated optical system consisting of multiple concave and convex lenses that help collect light from the scene and focus it on the imaging plane. Each one of those

lenses has its own optical properties and introduces a unique set of optical aberrations like distortions, vignetting, chromatic aberrations and others. Additionally, the imagine sensor itself introduces various artifacts due to technological limitations. Those include sensor noise, demosaicing patterns, over/under exposure, rolling shutter, motion blur, color corrections and compression. The nature and level of such artifacts in a specific training set can have a significant impact on the learned low-level features in a deep neural network [6, 41, 114]. Therefore, it is important that this imaging system specific effects are also addressed when generating synthetic training images to avoid introducing unwanted bias. One proposed solution was to learn the real image denoising and debluring process from raw data [39] in combination with learning the task model as an end-to-end process. This approach however requires careful calibration and access to the raw sensor data which is not always available. Another proposed mechanism to deal with this problem is to model the optical and sensor effects and apply them to rendered images [22, 171]. Results show how this can lead to significant gains in generalization ability of machine learning models when trained on synthetic images and testing on real images.

2.3 Synthetic-to-Real Domain Adaptation

An alternative approach to synthetic images is to not view them as "unrealistic" but rather as coming from a different data distribution to that of the real test images. This takes the emphasis away from the human-subjective concept of photorealism and focuses on the measurable difference in performance between training a model using synthetic and real training data, commonly known as the *Domain Gap*. For most semantic and scene understanding tasks, this gap is still significantly large even when using state-of-the-art rendered images as shown in Table 2.1. The task of adapting a machine learning model or its training procedure to be trained on data from a *source domain*, like rendered images, and tested on a *target domain*, like real images, is called *domain adaptation* [60]. The most common setup for this task is *unsupervised* domain adaptation where no target domain labels are available during adaptation or training process. The other setup is the

semi-supervised where a few labeled samples from the target domain are available for fine-tuning the model. Indeed, several benchmarks [151] have been proposed to evaluate methods for synthetic-to-real domain adaptation. For example, the Synth2Real benchmark [152] focuses on the tasks of object classification and detection. Similarly, the GTAV-to-Cityscapes benchmark measures semantic segmentation model performances when trained on the synthetic GTAV [164] dataset and evaluated on the Cityscapes [34] real dataset.
The domain gap problem has recently gained attention not only for training deep neural networks, but also in reinforcement learning for robotics agents [17] where training in simulated environments is more efficient and safe. In general, domain adaptation methods can be classified in thee catagories: image-space, feature-space and self-training domain adaptation.

2.3.1 Image-space domain adaptation

Image-space domain adaptation methods operate directly on the pixel values of the synthetic images. One proposed solution for removing the domain bias in training data has been *Domain Randomization* [172, 194, 220]. This method challenges the idea that photorealism is essential for creating good synthetic training data. Instead, it proposes to randomize the training image appearance [220], the rendering parameters (like camera parameters, noise, lighting, textures, etc.) [120, 194] or even the simulation physics dynamics [153]. The results are training images that vary largely in their appearance which prevents the trained model from over-fitting on any single particular appearance feature and makes it more robust across multiple domains. A major advantage of this approach is that it needs no access to even unlabeled test images since the randomization is supposed to reduce the model dependency on the training images domain in general [172]. Similarly, *Domain Stylization* [46] uses a standard style transfer method [109] to transform the overall appearance of real test images to look more like the synthetic training images. *Adversarial domain adaptation* is another group of methods based on Adversarial Generative Networks (GAN) [61] which uses an adversarial discriminator to ensure style similarity between real and synthetic images [180]. Supervised image-to-image

translation GANs [80, 206] can also be used for domain adaptation. They combine adversarial and supervised perceptual losses requiring matching pairs of real and synthetic images for training which is often not available. To train with only unpaired data, L1 loss [180] between the original and refined image can be added to ensure the image content does not change too much. But this is only valid when the source and target domains are relatively similar. Some methods [190] proposed to use an encoder-decoder architecture with reconstruction loss to overcome this need for paired training data. CycleGAN [228] takes this idea one step further by training two image-to-image translation networks (real-to-synthetic and synthetic-to-real) jointly with cycle-consistency losses. However, this does not guarantee that the adapted image still has the same semantic content as the input image and the networks can learn to "hallucinate" objects or large artifacts that can affect the usefulness of the data. CyCADA [74] attempts to solve this problem by combining unpaired adversarial loss with semantic consistency and feature losses, thus ensuring the semantic information gets correctly translated between domains.

2.3.2 Feature-space domain adaptation

Feature-space domain adaptation methods look instead at the representation learned by the deep-learning model and attempt to close the gap between the statistical distributions of features extracted from real and synthetic images. Some methods rely on minimizing an explicit distance measure between the two distributions like the Maximum Mean Discrepancy (MMD) [62, 116, 207]. Others use adversarial discriminators as well but between the learned features [51, 117] and integrate class or instance information [29, 205]. The advantage of those methods is that they can adapt the synthetic training data to a specific task and network architecture allowing them to achieve state-of-the-art performance on many domain adaptation tasks. However, this performance relies on having access to the exact model and large number of target domain images which might be restrictive in some practical applications.

2.3.3 Self-training

Self-training is a technique often used in semi-supervised learning where a small amount of labeled source domain data and a large amount of unlabeled target domain data are available [24, 197]. The main idea is to use the labeled synthetic training data to train an early model which is used to generate noisy pseudo-labels for the real data. Those in turn are used to fine-tune the supervised-learning models on real data [230]. The pseudo-labels are usually generated iteratively where at each iteration only predicted labeled with high confidence are used to re-train the model which will be used for another iteration of pseudo-label generation. However, applying this strategy naively can lead to the model drifting toward some easy classes as those will tend to have higher prediction confidence, thus making them more likely to enter the training data pool in the next iteration further reinforcing their dominance. Several ideas were proposed to deal with this issue, like class-balanced self-training (CBST) [230] framework which incorporates additional spacial priors to increase diversity. Another useful idea has been to use the scale-invariance property, especially in the semantic segmentation task, to constrain the generated pseudo-labels to be consistent across multiple scales of real images [187]. A combination of pseudo-labels and adversarial domain-adaptation has also been used [224] to further ensure the category-specific distributions of features of real and synthetic images are matching.

2.4 Neural image synthesis

2.4.1 Deep generative models

Generative models are a class of machine learning algorithms that try to learn the underlying data distribution of a training dataset rather than a mapping function between pairs of inputs and outputs. Some models allow for explicit evaluation of the probability distribution function while others learn only an implicit representation that allows for generating new samples from the hidden distribution [60]. The use of deep neural networks as generative models has followed on the large success of the gradient decent training

of classification or regression neural networks. After training, the network parameters should implicitly represent a specific instance from the family of distributions defined by the model architecture which best fits the training data. But unlike supervised learning where the training data consists of input-output pairs, the generative models need to be trained using only unlabeled data samples from an unknown distribution which makes the supervised training procedure unclear. Therefore, different models have been proposed to address this problem by either introducing additional constraints or using multiple networks.

Autoencoders [72] are a class of self-supervised neural network models that learn the identity function through an information bottle-neck which allows the learned representation to be efficient. They consist of an encoder E that takes an image x and produces a low-dimensional latent vector $z = E(x)$ and a decoder D that takes the latent vector z and produces a reconstructed image $x' = D(z)$ (Fig 2.4). The loss is then computed as the $L2$ norm between the input x and reconstructed image x'. In principle, Autoencoders can be used as generative models by randomly producing latent vectors z and using the decoder $D(z)$ as a generator. However, the structure of the learned latent space is usually non-regular due to lack of regularization and over-fitting on training examples. Using a random z vector often produces images not similar to those of the training set.

Variatianal Autoencoders (VAE) [98] instead encode the input x image as a conditional distribution $p_\theta(z|x)$ defined by the encoder network paramteres θ. A common choice is to model this distribution as multivariate Gaussian which can be parametrized by its mean μ and covariance matrix σ which are directly estimated by the encoder network based on input x. A latent vector can then be sampled $z \sim \mathcal{N}(\mu, \sigma^2)$ and used by the decoder to produce a reconstructed image sample x' to be compared to the input. A regularization term is added in the training loss as a KL-Divergence between the estimated distribution and the standard Gaussian distribution $\mathcal{N}(0, I)$. Variational autoencoders are relatively easy to train and produce regular latent space that allow meaningful interpolations between samples. However, they are often hard to scale to high-resolution images and often produce blurred results due to their stochastic nature. Several modification on image generation VAEs have been proposed to improve the result quality and

resolution like VDVAE [31], NVAE [201] and VAEBM [213].

Flow-based generative models take the idea of probabilistic generative models one step further by learning the explicit distribution of the input data $p(x)$ through normalizing flows. Normalizing flows are a sequence of invertible and differentiable transformation functions T that transform a complex multi-modal distribution $p(x)$ to simple one $p(z)$ or the other way around [147]. In practice, those transformations are implemented as a set of invertible neural network layers and the latent distribution $p(z)$ is defined as a standard multivariate Gaussian. RealNVP [40] introduced the *affine coupling layer* as an invertible transformation and was one of the first methods to show the effectivity of flow-based methods on image generation tasks. GLOW [97] further simplified the process by using *invertible 1x1 convolutions* to generate realistic high-quality face images.

2.4.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [61] have been the most popular and fastest growing category of image generative models in the past few years due to their ability to learn complex distributions and generate high-resolution semantically-consistent images. A GAN consists of two neural networks, a generator G and a discriminator D, which compete in an adversarial zero-sum game during training. The generator takes a random vector z and produces a fake image $x' = G(z)$. The discriminator takes either a batch of real images x or fake images x' and its task is to estimate the probability of the images coming from the real image distribution or being synthesized by the generator. The combined GAN loss $L(G, D)$ then can be expressed [61] as:

$$\min_G \max_D L(G, D) = \mathbb{E}_{x \sim p(x)}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$$

In practice this loss is optimized using an iterative approach alternating between training the discriminator to maximize $L(G, D)$ while the generator is fixed and training the generator to minimize $\mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$ while the discriminator is fixed.
A great number of modification has been proposed to the original GAN formulation to

add conditioning [136], improve their stability [168], generation quality [137] and resolution [18, 89]. Those have been applied across a large spectrum of image generation applications. Particularly, StyleGAN based methods [90, 91, 92] have achieved outstanding quality levels especially in the domain of face image generation. Nevertheless, GAN models often face several challenges:

- **Training stability.** The process of training GANs can be very unstable because of their adversarial game-theoretical nature which requires careful tuning to keep the balance between the generator and discriminator. Even though achieving a perfect Nash Equilibrium is not usually needed in practice to generate good quality images, a strong imbalance can lead the two GAN losses to oscillate and diverge very quickly [132].

- **Vanishing gradients.** The dependency of the generator loss $\mathbb{E}_{z\sim p(z)}[\log(D(G(z)))]$ on the discriminator network can lead to a smaller and smaller gradients in the generator as the discriminator is getting better. This further enforces the discriminator's dominance of the balance and decreases the generator's gradients in a positive feedback loop [8]. Goodfellow *et al*.proposed to reduce this by training the generator to maximize $\log(D(G(z)))$ [61] instead of minimizing $\mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))]$ while others suggested using the Wasserstein GAN fomulation instead [9, 63].

- **Mode collapse.** Another common problem of GANs is when the generator degrades to generating the same image or very similar images instead of covering the full training image distribution. This is called Mode collapse and often happens when the discriminator gets stuck in a local minima [8].

- **Evaluation.** A major challenge for GANs and image generative models in general is evaluating the quality of their results [16]. The golden standard for this is still human evaluation, often through crowd-sourcing platforms, since the goal is usually to make visually plausible results. For quantitative evaluation, using traditional image quality measures like SSIM or PSNR is not sufficient since those measure the low-level features and statistics while ignoring the high-level im-

age consistency and realism. Alternatively, measures based on distance between pre-trained features like Inception Score (IS) [173], Frechet Inception Distance (FID) [70] and Kernel Inception Distance (KID) [13] have been proposed to measure the distance between semantic feature distributions of the generated and real images.

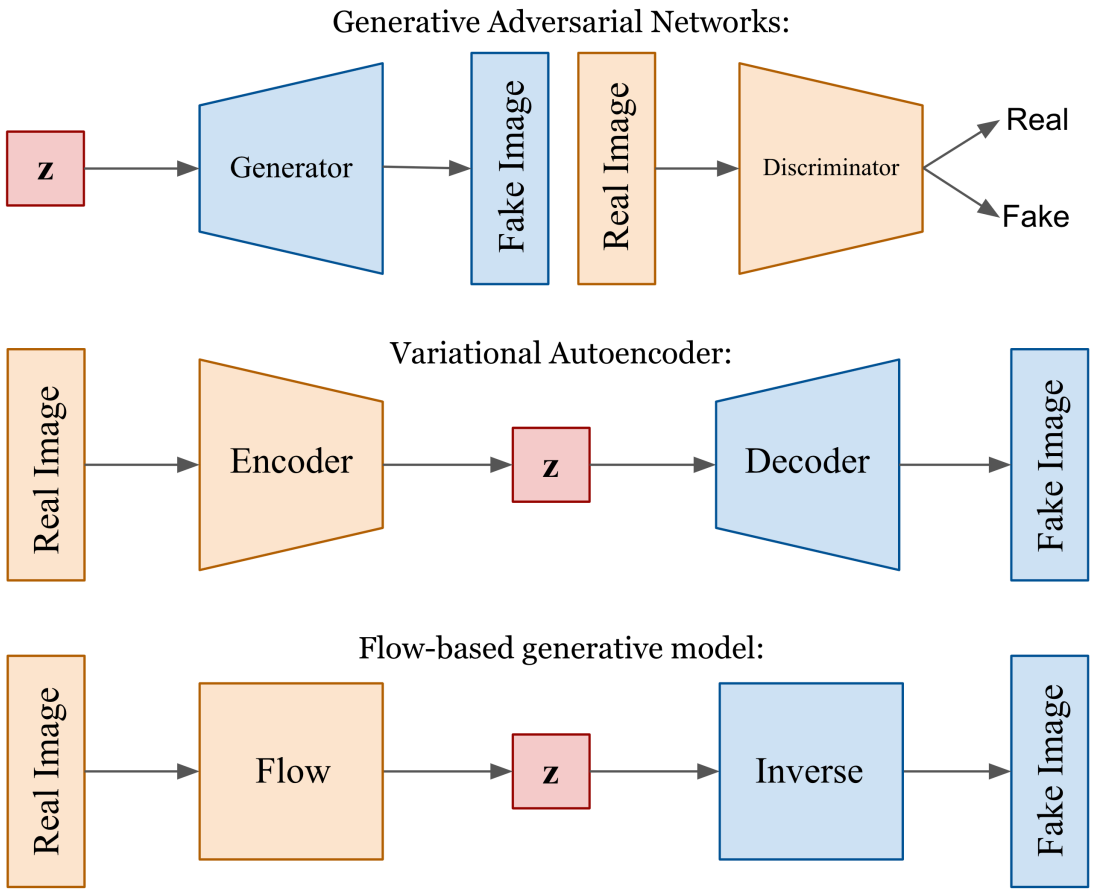


Figure 2.4. Comparison of different types of Generative neural network architectures. [209]

2.4.3 Image-to-image translation

Deep generative models are designed to capture the distribution $p(x)$ of training image $x \in X$ and allow for sampling random images from it. However, most practical image

synthesis applications require a degree of control over the output images to make them useful. Conditional generative models instead learn the distribution $p(x|y)$ of training images conditioned on a controllable input $y \in Y$. Early methods were built around conditioning on some global low-dimensional property like a class label [136], an image style [53], or a 3D object pose [45]. Image-to-image translation models [80, 206] instead extend the condition to be a 2D spatial map of similar size to the output image. This makes the conditioning more local and detailed as different regions of the generated image can correspond to different input conditions. Indeed the input to image-to-image translation networks can include images from different domains, 2D semantic maps, depth maps, edge maps or any other spatial maps that contain information related to an image. They can be split based on their training data into two categories: Paired and unpaired image-to-image translation.

Paired image-to-image translation. The simplest way to train image-to-image networks is to use a dataset of image pairs $(x \in X, y \in Y)$ with matching content but different domains. The network can then be trained in a supervised fashion using an $L1$ distance between the pixels of the generated and ground-truth images as loss. However, this can cause results to be blurry since the loss minimizes the average distance over the whole training set. The results often lack consistency and correct semantics since this loss considers image pixels to be independent and does not take into account the structure and semantics of the image [193]. *Perceptual loss* was proposed as a solution to include high-level features into the loss [85]. It computes the distance between images using the learned representations from a pre-trained image classification network. This allows it to better measure the difference in image content on different levels of abstractions based on the depth of the representation layer which can greatly improve the generated image details and consistency [27]. Similar to GANs, an adversarial discriminator can also be leveraged as a loss function to produce more detailed results. The difference between the perceptual loss and adversarial discriminator networks is that the former is pre-trained and fixed while the latter is trained together with the translation network allowing it to be more adaptive to the training data. A second difference is that it learns the conditional distribution $p(x|y)$ over the whole dataset rather than measuring

a distance with only one ground-truth image which allows for more flexibility in the output image as long as it matches the input conditioning. Unlike in the GAN architecture, the discriminator here takes two inputs: the conditioning image y and either the translation network output x' or the paired ground-truth image x to learn the conditional distribution $p(x|y)$. In practice, combining the adversarial with the perceptual losses can produce better results since it can ensure the content of the image does not change too much while making the adversarial training faster and more stable.

Unpaired image-to-image translation. Collecting a large dataset of matching image pairs from different domains is hard and even impossible in many applications. Therefore, it is desirable to rely on large collections of unpaired images to train the translation neural networks instead. The challenge however is that the translation is not unique for a specific input image $y \in Y$ since the training datasets only include samples from $p(x)$ and $p(y)$ but not any samples from $p(x|y)$. For this reason, the nature of the translation can vary greatly and is usually defined by the structure and constraints of the translation model. The most common assumption is that the translation function should replace the common features of the input set Y by the shared features of the output set X while keeping the unique attributes of the specific input sample $y \in Y$ intact in the output image x'. An early form of such image-to-image translation methods has been *Image-based Neural Style Transfer* [84] methods where the goal is to transfer the style or texture of one image to another while keeping its semantic content. To achieve this, they often use the image optimization technique pioneered by DeepDream [139]. It uses a pre-trained convolutional neural network to compute image features of the target and style images on different depths. Then, a new image is optimized using gradient decent to produce low-level features matching the style image and high-level features matching the content image [53, 54]. This process is slow due to the iterative nature of gradient decent optimization. It is also limited because it relies on pre-trained networks on tasks like object classification which might not produce the best features for all images domains. A more general solution is to train a model to translate between the two specific image domains. CycleGAN [228] has been one of the most successful in this field as it builds on the simple idea of cycle-consistency to ensure that the translation process preserves the image content while the adversarial discriminator loss ensures the appearance of the

image matches the target domain. Another common idea is to use two sets of encoder-decoder networks to map images from both domains into a shared content latent space [111]. The translation then happens by combining the encoder for one domain with the decoder of the other. Alternatively, an additional explicit style encoder can be used to condition the decoder allowing for learning multiple styles using the same network [77]. A large number of GAN-based unpaired image-to-image translation methods has been proposed since then to add attention mechanisms [95, 130], improve result diversity [107] and make it more data-efficient [112].

2.4.4 Neural rendering

Deep image generative models have made large leaps in performance in the past year making it possible to create high resolution photo-realistic images at near real-time rates using only a few user clicks [91, 148]. However, they remained rarely used in practical applications compared to classical rendering because they lack fine control over the scene structure, materials, lighting and rendering parameters. This limitation is a "deal-breaker" in many professional applications like computer animation where experts need full control over the generated images and therefore prefer to use classical rendering algorithms.

At the intersection of classical rendering and generative neural networks, the new field of *Neural Rendering* has recently emerged with the aim of combining the photo-realism of generative models with the precise control of classical rendering approaches. A recent review [193] has defined Neural Rendering as *"Deep image or video generation approaches that enable explicit or implicit control of scene properties such as illumination, camera parameters, pose, geometry, appearance, and semantic structure"*. Indeed, to make this control over the scene possible, Neural Rendering methods often try to learn an implicit or explicit representation of the scene structure from one or few of its images. However, they differ from traditional image-based rendering in that they do not use the input images at inference time and rely on their own constructed scene representation instead.

An early work showing the potential of neural networks to generate novel-views from a learned representation was Generative Query Network (GQN) [48]. The two-part model consists of a representation network that takes a few images of the target scene and generates a compact representation, and a generator network that takes the representation and a random query view-point from which it predicts the scene image. The method showed great results when trained on synthetic scenes with primitive shapes and colors, however it is hard to extend to real scenes due to the limited low-dimensional latent representation. To address the complexity in real scenes, Meshry *et al*. [133] proposed to first create a rich 3D point-cloud representation of the target scene from its images using traditional reconstruction methods. The point-cloud is then projected from the target camera view and used as an input to a neural network that generates the new image. This method allows for flexible novel view synthesis of large scenes but requires a large number of images of the same scene to create an accurate 3D point-cloud representation in addition to semantic segmentation maps. HoloGAN[144] focuses instead on learning controllable image generation from a large collection of different objects from the same semantic class. It using 3D convolutions to generate a 3D latent representation of the object at the early stage of the image generator network. This 3D feature volume is then transformed to a target camera viewpoint and projected into a 2D feature volume using a differentiable projection unit [143] which is further processed using 2D convolutions to produce an image. The neural 3D representation allows the generator to disentangle the object's pose, shape and appearance while still keeping the whole network trainable end-to-end. Nguyen-Phuoc *et al*. [145] extended this method further to scenes with multiple objects (including the background) allowing for independent control over the pose and appearance of each object. Mildenhall *et al*. [135] recently introduced the radically novel idea of representing a scene with a continuous volumetric function called a *Neural Radiance Fields* (NeRF) [135] which maps a 3D location (x, y, z) and viewing direction (ϕ, θ) in the scene volume to an image value (R, G, B) and radiance (σ). This function can be implicitly learned using a fully connected neural network from multiple images of the scene and used at inference time to generate novel views of it. This formulation combines 3D reconstruction and neural rendering intro a single task of estimating the radiance field which allows it to naturally handle situations like reflective or transparent

objects which are very difficult for traditional 3D reconstruction methods. The simplicity and high-quality results of Neural Radiance Fields [135] have inspired a large number of works exploring their use for non-rigid objects [196], dynamic scenes [158], human faces[52], unstructured [125] or few image collections [219], and many others.

2.5 Discussion

This chapter presents an overview of methods for creating large scale training data for deep neural networks and their limitations and challenges. Manual annotation of real images has been the golden-standard for training datasets in many practical applications for decades. But the exponentially growing size of datasets needed for training large neural networks clearly indicates that other more scalable sources of training data are needed. Rendering engines could be a cheap alternative for producing training data from virtual environments. However, their focus on human perceptual photorealism makes them less suitable for training deep neural networks. Additionally, building large and accurate virtual replicas of real environments can be very costly [102]. One possible solution to this problem is to carefully combine virtual objects with real scenes to easily expand the variability in the real images without largely impacting their realism. This idea is explored further in Chapter 3 of this thesis. On the other hand, recent developments in neural image synthesis models like Generative Adversarial Networks and neural rendering provide a viable alternative to traditional rendering engines. Their main advantage in relation to synthetic data generation is that they can be trained using different loss functions and objectives than just photorealism. This presents an opportunity to develop image synthesis models that explicitly specialize in generating training data for other machine learning models. This idea will be further explored in Chapters 4 and 5 of this thesis.

Chapter 3

Augmented reality for deep learning

3.1 Introduction

In recent years, deep learning has revolutionized the field of computer vision. Many tasks that seemed elusive in the past, can now be solved efficiently and with high accuracy using deep neural networks, sometimes even exceeding human performance [191]. However, it is well-known that training high capacity models such as deep neural networks requires huge amounts of labeled training data. This is particularly problematic for tasks where annotating even a single image requires significant human effort, e.g., for semantic or instance segmentation. A common strategy to circumvent the need for human labels is to train neural networks on synthetic data obtained from a 3D renderer for which ground truth labels can be automatically obtained, [50, 66, 141, 165, 167, 177, 203, 225]. While photo-realistic rendering engines exist [82], it is difficult and time-consuming to attain a level-of-detail comparable to real-world photographs (e.g., leaves of trees).

In this chapter, we demonstrate that state-of-the-art photo-realistic rendering can be utilized to augment real-world images and obtain virtually unlimited amounts of training data for specific tasks such as semantic instance segmentation and object detection. Towards this goal, we introduce a newly augmented dataset called KITTI-360 which contains real images augmented with virtual objects based on 360 degree environment

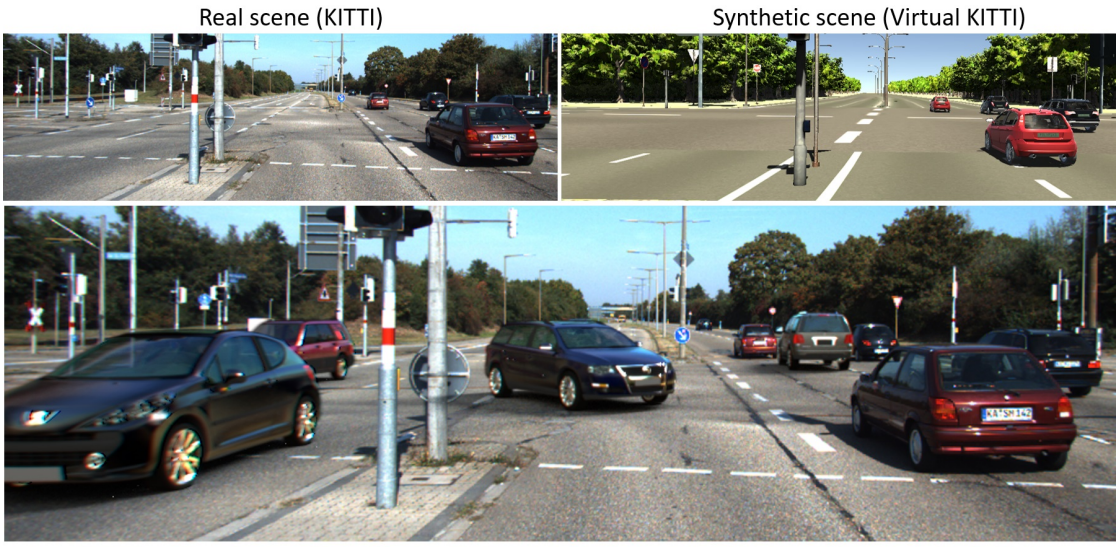


Figure 3.1. Obtaining synthetic training data usually requires building large virtual worlds (top right) [50]. We propose a new way to extend datasets by augmenting real training images (top left) with realistically rendered cars (bottom) keeping the resulting images close to real while expanding the diversity of training data.

maps and partially annotated with semantic and instance information. In particular, we augment the data with realistically rendered car instances. This allows us to keep the full realism of the background while being able to generate arbitrary amounts of foreground object configurations. Figure 3.1 shows a real image before and after augmentation. While our rendered objects rival the realism of the input data, they provide the variations (e.g., pose, shape, appearance) needed for training deep neural networks for instance aware semantic segmentation and bounding box detection of cars. Using those augmented images, we are able to considerably improve the accuracy of state-of-the-art deep neural networks trained on real data.

While the level of realism is an important factor when synthesizing new data, there are two other important aspects to consider - data diversity and human labor. Manually assigning a class or instance label to every pixel in an image is possible but tedious, requiring up to one hour per image [34]. Thus existing real-world datasets are limited to a few hundred [19] or thousand [34] annotated examples, thereby severely limiting the diversity of the data. In contrast, the creation of virtual 3D environments allows

for arbitrary variations of the data and virtually infinite number of training samples. However, the creation of 3D content requires professional artists and the most realistic 3D models (designed for modern computer games or movies) are not publicly available due to the enormous effort and cost involved in creating them [102, 183]. In the domain of urban driving scenario, a few recent projects [164, 165, 177] have demonstrated how content from commercial games can be accessed through manipulating low-level GPU instructions and used to create large amount of training data. But legal problems are likely to arise and often the full flexibility of the data generation process is no longer available.

In this chapter we demonstrate that the creation of an augmented dataset which combines real with synthetic data requires only moderate human effort while yielding the variety of data necessary for improving the accuracy of state-of-the-art instance segmentation network (Multitask Network Cascades) [35] and object detection network (Faster R-CNN) [162]. To assess the performance of networks trained on various datasets, we annotated the popular KITTI-2015 dataset [131] with semantic and instance labels. We show that a model trained using our augmented dataset generalizes better than models trained on real data or purely synthetic data. Finally, combining our augmented dataset with a purely synthetic dataset yields a noticeable increase in performance indicating that augmented and synthetic data can be advantageously combined for training high performance recognition models. Since our data augmentation approach requires only minimal manual effort, we believe that it constitutes an important milestone towards the ultimate task of creating virtually infinite, diverse and realistic datasets with ground truth. In summary, our contributions are as follows:

- We propose an efficient solution for augmenting real images with photo-realistic synthetic object instances which can be arranged in a flexible manner.
- We provide an in-depth analysis of the importance of various factors of the data augmentation process, including the number of augmentations per real image, the realism of the background and the foreground regions.
- We demonstrate through extensive experiments how augmentation of real images increases the variability in the data leading to more generalizable models that out-

perform training on real or purely synthetic datasets. Furthermore, we found that synthetic and augmented datasets are complementary and combining the two enhances performance further.

- For conducting the experiments in this chapter, we introduce two new datasets: KITTI-15 semantic, instance and panoptic segmentation dataset consisting of 400 manually labeled images from the KITTI dataset [56]. And the KITTI-360-Cars dataset consisting of 200 images from [214] with accurate manual instance segmentation for cars in addition to 28 3D car models with high-quality physically-based materials in Blender [15].

3.2 Related Work

Due to the scarcity of real-world data for training deep neural networks, several researchers have proposed to use synthetic data created with the help of a 3D rendering engine. Indeed, it was shown in [141, 165, 177] that deep neural networks can be trained on synthetic data and that the accuracy can be further improved by fine tuning on real data [165]. Moreover, it was shown that the realism of synthetic data is important to obtain good performance [129, 141]. Making use of this observation, several synthetic datasets have been released which we will briefly review in the following. [67] presents a scene-specific pedestrian detector using only synthetic data. [203] presents a synthetic dataset of human bodies and use it for human depth estimation and part segmentation from RGB-images. In a similar effort, [28] uses synthetic data for 3D human pose estimation. In [36], synthetic videos are used for human action recognition with deep networks. [226] presents a synthetic dataset for indoor scene understanding. Similarly, [66] uses synthetic data to train a depth-based pixelwise semantic segmentation method. In [225], a synthetic dataset for stereo vision is presented which has been obtained from the UNREAL rendering engine. [229] presents the AI2-THOR framework, a 3D environment and physics engine which they leverage to train an actor-critic model using deep reinforcement learning. [150] investigates how missing low-level cues in 3D CAD models affect the performance of deep CNNs trained on such models. [185] uses 3D

CAD models for learning a multi-view object class detector. [105]

In the context of autonomous driving, the SYNTHIA dataset [167] contains a collection of diverse urban scenes and dense class annotations. [50] introduces a synthetic video dataset (Virtual KITTI) which was obtained from the KITTI dataset [55] alongside with dense class annotations, optical flow and depth. [186] uses a dataset of rendered 3D models on random real images for training a CNN on viewpoint estimation. While all aforementioned methods require labor intensive 3D models of the environment, we focus on exploiting the synergies of real and synthetic data using augmented reality. In contrast to purely synthetic datasets, we obtain a large variety of realistic data in an efficient manner. Furthermore, as evidenced by our experiments, combining real and synthetic data within the same image results in models with better generalization performance.

While most works use either real or synthetic data, only few papers consider the problem of training deep models with mixed reality. [169] estimates the parameters of a rendering pipeline from a small set of real images for training an object detector. [64] uses synthetic data for text detection in images. [156] uses synthetic human bodies rendered on random backgrounds for training a pedestrian detector. [42] renders flying chairs on top of random Flickr backgrounds to train a deep neural network for optical flow. Unlike existing mixed-reality approaches for training data generation which are either simplistic, consider single objects or augment objects in front of random backgrounds, our goal is to create high fidelity augmentations of complex multi-object scenes at high resolution. A detailed survey of state-of-the-art photorealistic mixed-reality techniques is presented in [105]. In particular, our approach takes the geometric layout of the scene, environment maps as well as artifacts stemming from the image capturing device into account. We experimentally evaluate which of these factors are important for training good models.

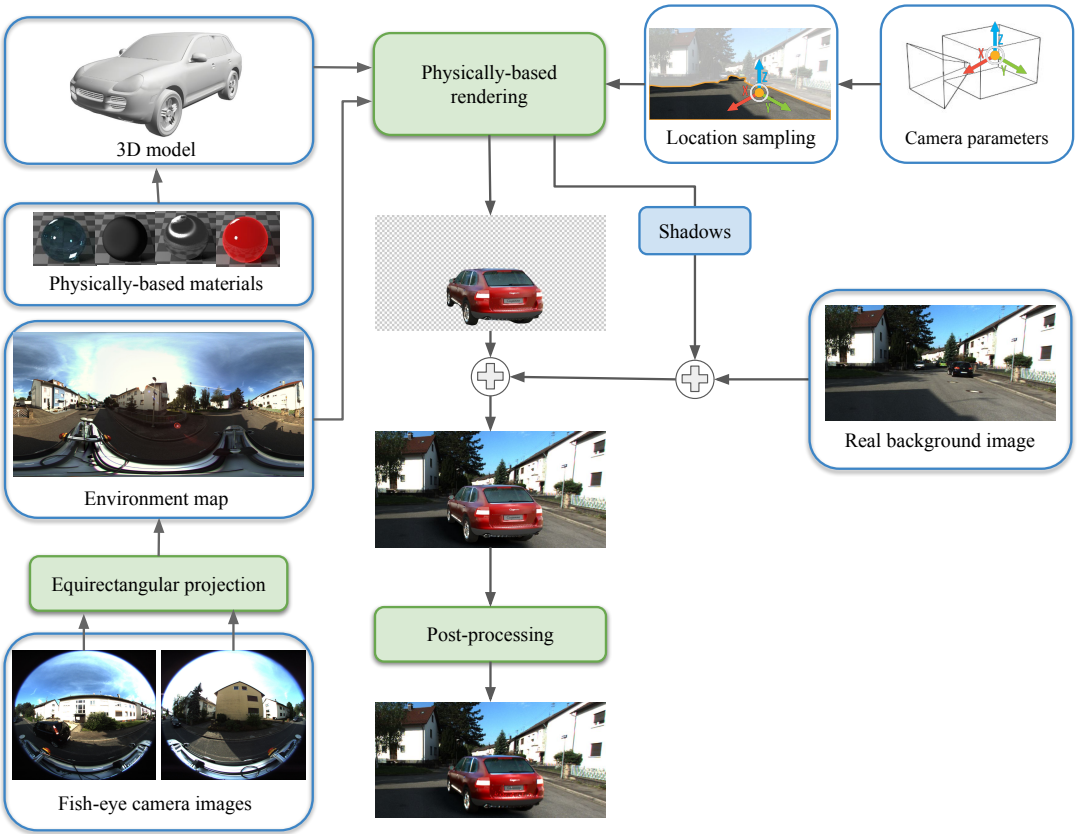


Figure 3.2. Overview of our augmentation pipeline. Given a set of 3D car models, physically-based materials, sampled locations and environment maps, we render high quality cars with shadows and overlay them on top of real images. The final post-processing step insures better visual matching between the rendered and real parts of the resulting image.

3.3 Data Augmentation Pipeline

In this section, we describe our approach to data augmentation through photo-realistic rendering of 3D models on top of real scenes. To achieve this, three essential components are required: (i) detailed high quality 3D models of cars, (ii) a set of 3D locations and poses used to place the car models in the scene and, (iii) the environment map of the scene that can be used to produce realistic reflections and lighting on the models that matches the scene. We use 28 high quality 3D car models covering 7 categories (SUV, sedan, hatchback, station wagon, mini-van, van) obtained from online model reposito-

ries[1]. The car color is chosen randomly during rendering to increase the variety in the data. To achieve high quality realistic augmentation, it is essential to correctly place virtual objects in the scene at practically plausible locations, matching the distribution of poses and occlusions in the real data. We explored four different location sampling strategies: (i) Manual car location annotations, (ii) Automatic road segmentation, (iii) Road plane estimation, (iv) Random unconstrained location sampling. For (i), we leverage the homography between the ground plane and the image plane, transforming the perspective image into a birdseye view of the scene. Based on this new view, our in-house annotators marked possible car trajectories on the road where cars can be placed (Figure 3.3). We sample the locations randomly from these annotations and set the rotation along the vertical axis of the car to be aligned with the trajectory set by the user. For (ii), we use the algorithm proposed by [192] which segments the image into road and non-road areas with high accuracy. We back-project those road pixels and compute their location on the ground plane to obtain possible car locations and use a random rotation around the vertical axis of the vehicle. While this strategy is simpler, it can lead to visually less realistic augmentations mainly due to random rotations and unrealistic overlap with neighboring real objects. For (iii), since we know the intrinsic parameters of the capturing camera and its exact pose, it is possible to estimate the ground plane in the scene. This reduces the problem of sampling the pose from 6D to 3D, namely the 2D position on the ground plane and one rotation angle around the model's vertical axis. Finally for (iv), we randomly sample locations and rotations from an arbitrary distribution. We empirically found manual car location annotations to perform slightly better than automatic road segmentation and on par with road plane estimation as described in Sec. 3.4. We use the manual location labeling in all our experiments, unless stated otherwise.

We leverage the 360 degree panoramas of the environment from the KITTI-360 dataset [214] as environment map proxies for realistic rendering of cars in street scenes. These 360 degree images are taken from the location of the capture vehicle. Thus, they are only an approximation of the true environment maps expected at the location of the augmented object. Using the 3D models, locations and environment maps, we render

[1] `http://www.dmi-3d.net`

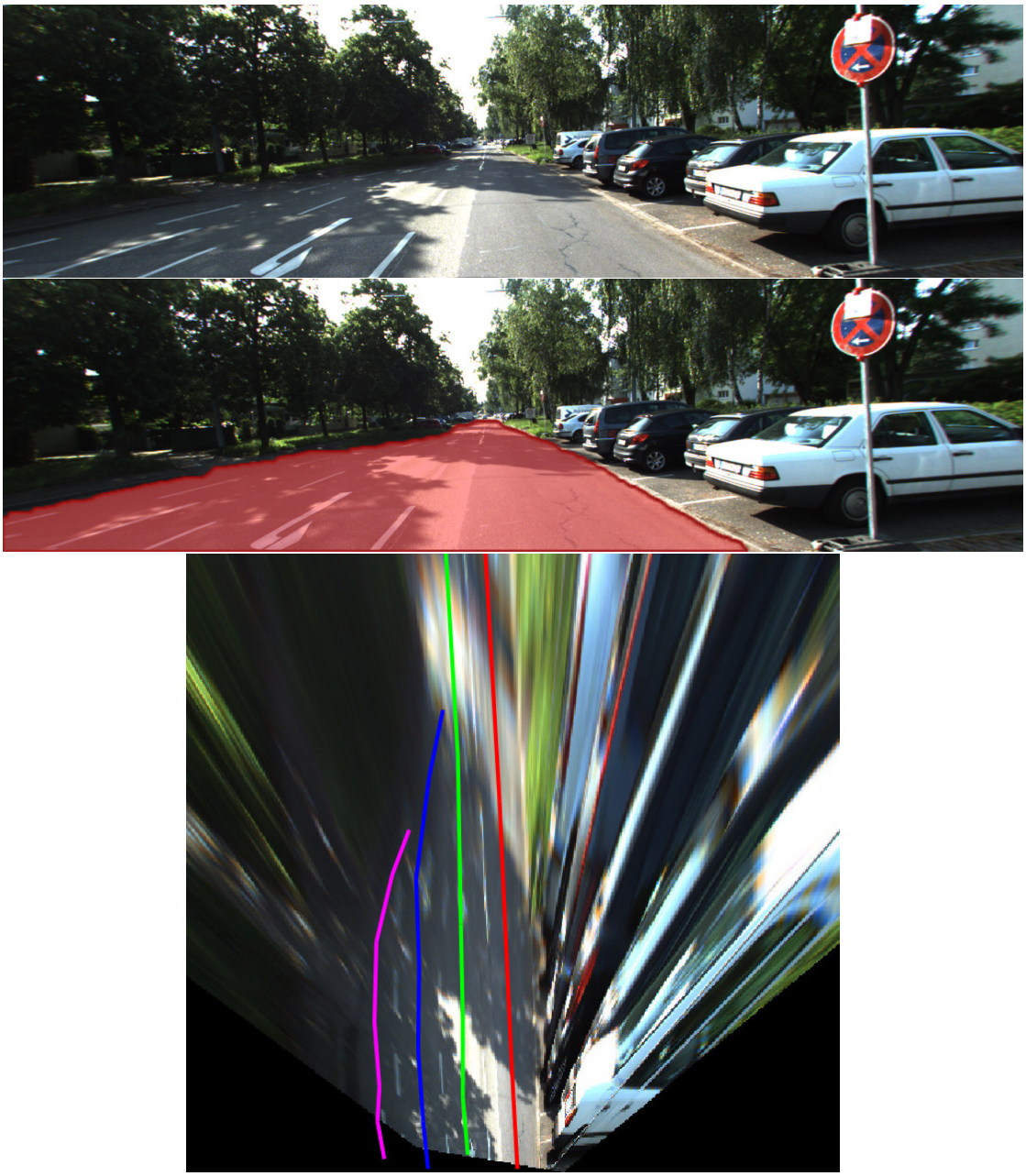


Figure 3.3. (Top) The original image. (Middle) Road segmentation using [192] in red for placing synthetic cars. (Down) Using the camera calibration, we project the ground plane to get a birdseye view of the scene. From this view, the annotator draws lines indicating vacant trajectories where synthetic cars can be placed.

(a) The two cars in the center are rendered

(b) The car to the left and in the center are rendered

(c) The three cars in the center are rendered

(d) The three cars on the road are rendered

Figure 3.4. Example images produced by our augmentation pipeline.

cars using the Cycles renderer implemented in Blender [15]. Figure 3.2 illustrates our augmentation approach. However, the renderings obtained from Blender lack typical artifacts of the image formation process such as motion blur, lens blur, chromatic aberrations, etc. To better match the image statistics of the background, we thus design a post-processing work-flow in Blender's compositing editor which applies a sequence of 2D effects and transformations to simulate those effects, resulting in renderings that are more visually similar to the background. More specifically, those operations include (i) independent color shifts on the RGB channels to simulate chromatic aberrations in the real lens, (ii) depth-blur operation to match the depth-of-field of the camera, (iii) radial motion-blur that matches the blur caused by the moving camera, (iv) color noise and (v) glow effects to imitate sensor overexposure. Finally, we use several color curve operations and Gamma transformations to visually match the color statistics and contrast of the real data. The parameters of these operations have been estimated empirically to optimize the visual similarity between the synthetic and real cars. Some results are shown in Figure 3.4.

3.4 Evaluation

In this section we show how augmenting driving scenes with synthetic cars is an effective way to expand a dataset and increase its quality and variance. In particular, we highlight two aspects in which data augmentation can improve the real data performance. First, introducing new synthetic cars in each image with detailed ground truth labeling makes the model less likely to over-fit to the small amount of real training data and exposes it to a large variety of car poses, colors and models that might not exist or be rare in real images. Second, our augmented cars introduce realistic occlusions of real cars which makes the learned model more robust to occlusions since it is trained to detect the same real car each time with a different occlusion configuration. This second aspect also protects the model from over-fitting to the relatively small amount of annotated real car instances.

We study the performance of our data augmentation method on two challenging

vision tasks, instance segmentation and object detection. Using different setups of our augmentation method, we investigate how the quality and quantity of augmented data affects the performance of a state-of-the-art instance segmentation model. In particular, we explore how the number of augmentations per real image and number of added synthetic cars affects the quality of the learned models. We compare our results on both tasks to training on real and fully synthetic data, as well as a combination of the two (i.e., training on synthetic data and fine-tuning on real or augmented data). We also experiment with different aspects of realism such as environment maps, post-processing and car placement methods.

3.4.1 Datasets

KITTI-360. For our experiments, we created a new dataset which contains 200 images chosen from the dataset presented in [214]. We labeled all car instances at pixel level using our in-house annotators to create high quality semantic instance segmentation ground truth. This new dataset (KITTI-360) is unique compared to KITTI [55] or Cityscapes [34] in that each frame comes with two 180° images taken by two fish-eye cameras on top of recording platform. Using an equirectangular projection, the two images are warped and combined to create a full 360° omni-directional image that we use as an environment map during the rendering process. These environment maps are key to creating photo-realistic augmented images and are used frequently in Virtual Reality and Cinematic special effects applications. The dataset consists of 200 real images which form the basis for augmentation in all our experiments, i.e., we reuse each image n times with differently rendered car configurations to obtain an n-fold augmented dataset.

VKITTI. To compare our augmented images to fully synthetic data, we use the Virtual KITTI (VKITTI) dataset [50] which has been designed as a virtual proxy for the KITTI-2015 dataset [131]. Thus, the statistics of VKITTI (e.g., semantic class distribution, car poses and environment types) closely resembles those of KITTI-15 which we use as a testbed for evaluation. The dataset comprises $\sim$12,000 images divided into 5 sequences with 6 different weather and lighting conditions for each sequence.

KITTI-15. To demonstrate the advantage of data augmentation for training robust models, we create a new benchmark test dataset different from the training set using the popular KITTI-2015 dataset [131]. More specifically, we annotated all the 200 publicly available images of the KITTI-2015 [131] with pixel-accurate semantic instance labels using our in-house annotators. While the statistics of the KITTI-15 dataset are similar to those of the KITTI-360 dataset, it has been recorded in a different year and at a different location / suburb. This allows us to assess performance of instance segmentation and detection methods trained on the KITTI-360 and VKITTI dataset.

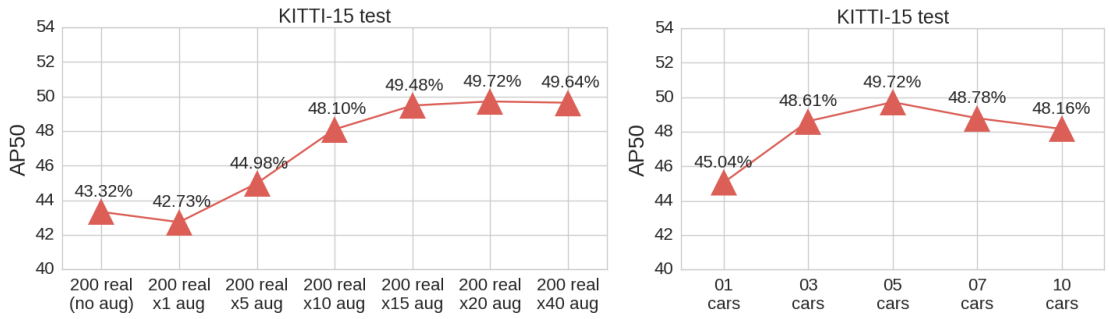
Cityscapes. To further evaluate the generalization performance of augmented data, we test our models using the larger Cityscapes validation dataset [34] which consists of 500 instance mask annotated images. The capturing setup and data statistics of this dataset is different to those of KITTI-360, KITTI-15 and VKITTI making it a more challenging test set.

3.4.2 Evaluation Protocol

We evaluate the effectiveness of augmented data for training deep neural networks using two challenging tasks, instance-level segmentation and bounding-box object detection. In particular, we focus on the task of car instance segmentation and detection as those dominate our driving scenes.

Instance segmentation. We choose the state-of-the-art Multi-task Network Cascade (MNC) by [35] for instance-aware semantic segmentation. We initialize each model using the features from the VGG model [182] trained on ImageNet and train the method using variants of real, augmented or synthetic training data. For each variant, we train the model until convergence and average the result of the best performing 5 snapshots on each test set. We report the standard average precision metric of an intersection-over-union threshold of 50% (AP50) and 70% (AP70), respectively.

Object detection. For bounding-box car detection we adopt the popular Faster-RCNN [162] method. We initialize the model using the VGG model trained on ImageNet as well and then train it using the same dataset variants for 10 epochs and average the



(a) **(b)**

Figure 3.5. Instance segmentation performance using augmented data. (a) We fix the number of synthetic cars to 5 per augmentation and vary the number of augmentations per real image. (b) We fix the number of augmentations to 20 and vary the maximum number of synthetic cars rendered in each augmented image.

best performing 3 snapshots on each test set. For this task, we report the mean average precision (mAP) metric commonly used in object detection evaluation.

3.4.3 Augmentation Analysis

We experiment with the two major factors for adding variation in the augmented data. Those are, (i) the number of augmentations, i.e the number of augmented images created from each real image, (ii) the number of synthetic cars rendered in each augmented images.

Figure 3.5a shows how increasing the number of augmentations per real image improves the performance of the trained model through the added diversity of the target class, but then saturates beyond 20 augmentations. While creating one augmentation of the real dataset adds a few more synthetic instances to each real image, it fails to improve the model performance compared to training on real data only since the introduced synthetic cars are likely to occlude other real cars behind them resulting in little gain in diversity. Nevertheless, creating more augmentations results in a larger and more diverse dataset that performs significantly better on the real test data. This suggests that the main advantage of our data augmentation comes from adding realistic diversity to existing datasets through having several augmented versions of each real image. In the rest of our exper-

iments, we use 20 augmentations per real unless stated otherwise.

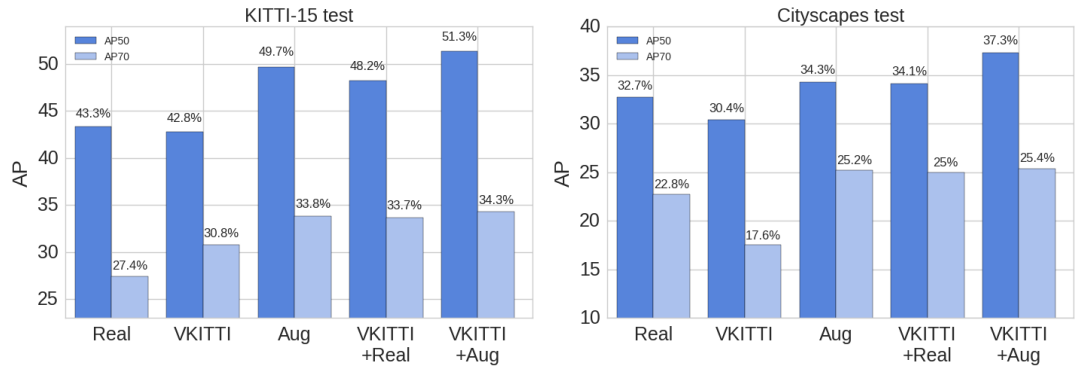
In figure 3.5b we examine the role of the synthetic content of each augmented image on performance by augmenting the dataset with various numbers of synthetic cars in each augmented image. At first, adding more synthetic cars improves the performance by introducing more instances to the training set. It provides more novel car poses and realistic occlusions on top of real cars leading to more generalizable models. Nevertheless, increasing the number of cars beyond 5 per image results in a noticeable decrease in performance. Considering that our augmentation pipeline works by overlaying rendered cars on top of real images, adding a larger number of synthetic cars will cover more of the smaller real cars in the image reducing the ratio of real to synthetic instances in the dataset. This negative effect soon undercuts the benefit of the diversity provided by the augmentation leading to decreasing performance. Our conjecture is that the best performance can be achieved using a balanced combination of real and synthetic data. Unless explicitly mentioned otherwise, all our experiments were conducting using 5 synthetic cars per augmented image.

3.4.4 Comparing Real, Synthetic and Augmented Data

Synthetic data generation for autonomous driving has shown promising results in the recent years. However, it comes with several drawbacks:

- The time and effort needed to create a realistic and detailed 3D world and populate it with agents that can move and interact.
- The difference in data distribution and pixel-value statistics between the real and virtual data prevents it from being a direct replacement to real training data. Instead, it is often used in combination with a two stage training procedure where the model is first pre-trained on large amounts of virtual data and then fine tuned on real data to better match the test data distribution.

Using our data augmentation method we hope to overcome these two limitations. First, by using real images as background, we limit the manual effort to modeling high quality

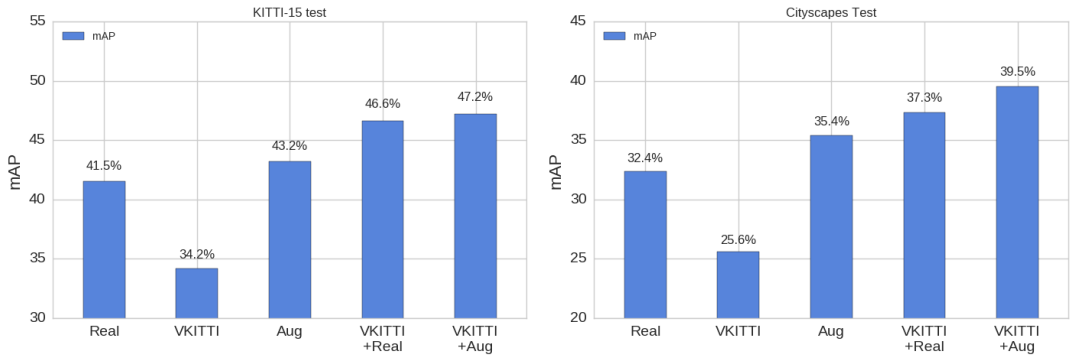


(a) **(b)**

Figure 3.6. Using our 20-fold augmented KITTI-360 dataset (Aug), we can achieve better performance on both (a) the KITTI-15 dataset and (b) Cityscapes [34] test set compared to using synthetic data (VKITTI) or real KITTI-360 data (Real) separately. We also outperform models trained on synthetic data and fine-tuned with real data (VKITTI+Real) while significantly reducing manual effort. Additionally, fine-tuning the model trained on VKITTI using our Augmented data (VKITTI+Aug) further improves the performance.

3D cars compared to designing full 3D scenes. A large variety of 3D cars is available through online 3D model warehouses and can be easily customized. Second, by limiting the modification of the images to the foreground objects and compositing them with the real backgrounds, we keep the difference in appearance and image artifacts at minimum. As a result, we are able to boost the performance of the model directly trained on the augmented data without the need for a two stage pre-training/refinement procedure.

In Figure 3.6, we compare models trained on the real KITTI-360 dataset with 200 images, the synthetic VKITTI dataset with $\sim$12000 images and the augmented dataset created from the same 200 real images of KITTI-360, but each now augmented 20 times with different car models and poses yielding a total of 4000 augmented images. To further compare our augmented data to fully synthetic data, we train a model using VKITTI and refine it with the real KITTI-360 training set. While fine-tuning the model with real data improves the results from 42.8% to 48.2%, our augmented dataset achieves a performance of 49.7% in a single step. Additionally, using our augmented data for fine-tuning the VKITTI trained model significantly improves the results (51.3%). This demonstrates that the augmented data is closer in nature to real than to synthetic data.



(a) **(b)**

Figure 3.7. Training the Faster RCNN model [162] for bounding box detection on various datasets. Using our augmented dataset we outperform the models trained using synthetic data or real data separately on both (a) KITTI-15 test set and (b) Cityscapes [34] test set. We also outperform the model trained on VKITTI and fine-tuned on real data (VKITTI+Real) by using our augmented data to fine tune the model trained on VKITTI (VKITTI+Aug).

While the flexibility of synthetic data can provide important variability, it fails to provide the expected boost over real data due to differences in appearance. On the other hand, augmented data complements this by providing high visual similarity to the real data, yet preventing over-fitting.

While virtual data captures the semantics of the real world, at the low level real and synthetic data statistics can differ significantly. Thus training with purely synthetic data leads to biased models that under-perform on real data. Similarly training or fine-tuning on a limited size dataset of real images restricts the generalization performance of the model. In contrast, the composition of real images and synthetic cars into a single frame can help the model to learn shared features between the two data distributions without over-fitting to the synthetic ones. Note that our augmented dataset alone performs slightly better than the models trained on VKITTI and fine-tuned on the real dataset only. This demonstrates that state-of-the-art performance can be obtained without designing complete 3D models of the environment. Figure 3.7a and 3.7b show similar results achieved for the detection task on both KITTI-15 and Cityscapes respectively.

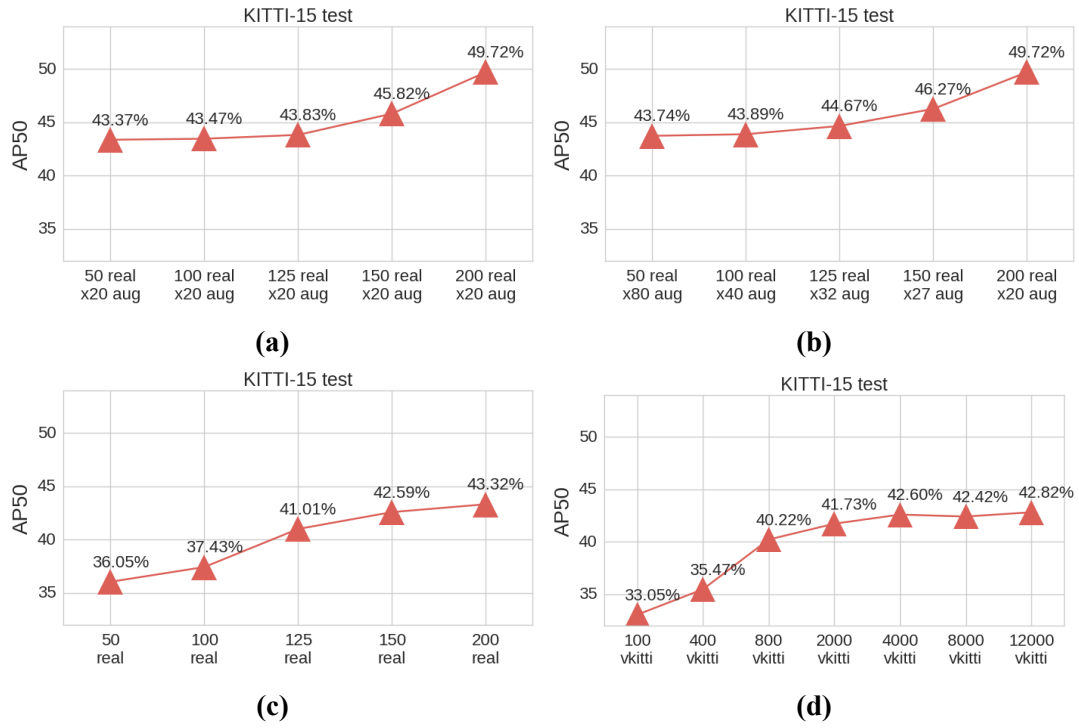


Figure 3.8. Instance segmentation performance using real, synthetic and augmented datasets of various sizes tested on KITTI-15. (a) We fix the number of augmentations per image to 20 but vary the number of real image used for augmentation. This leads to a various size dataset depending on the number of real images. (b) We vary the number real images while keeping the resulting augmented dataset size fixed to 4000 images by changing the number of augmentations accordingly. (c) We train on various number of real images only. (d) We train on various number of VKITTI images.

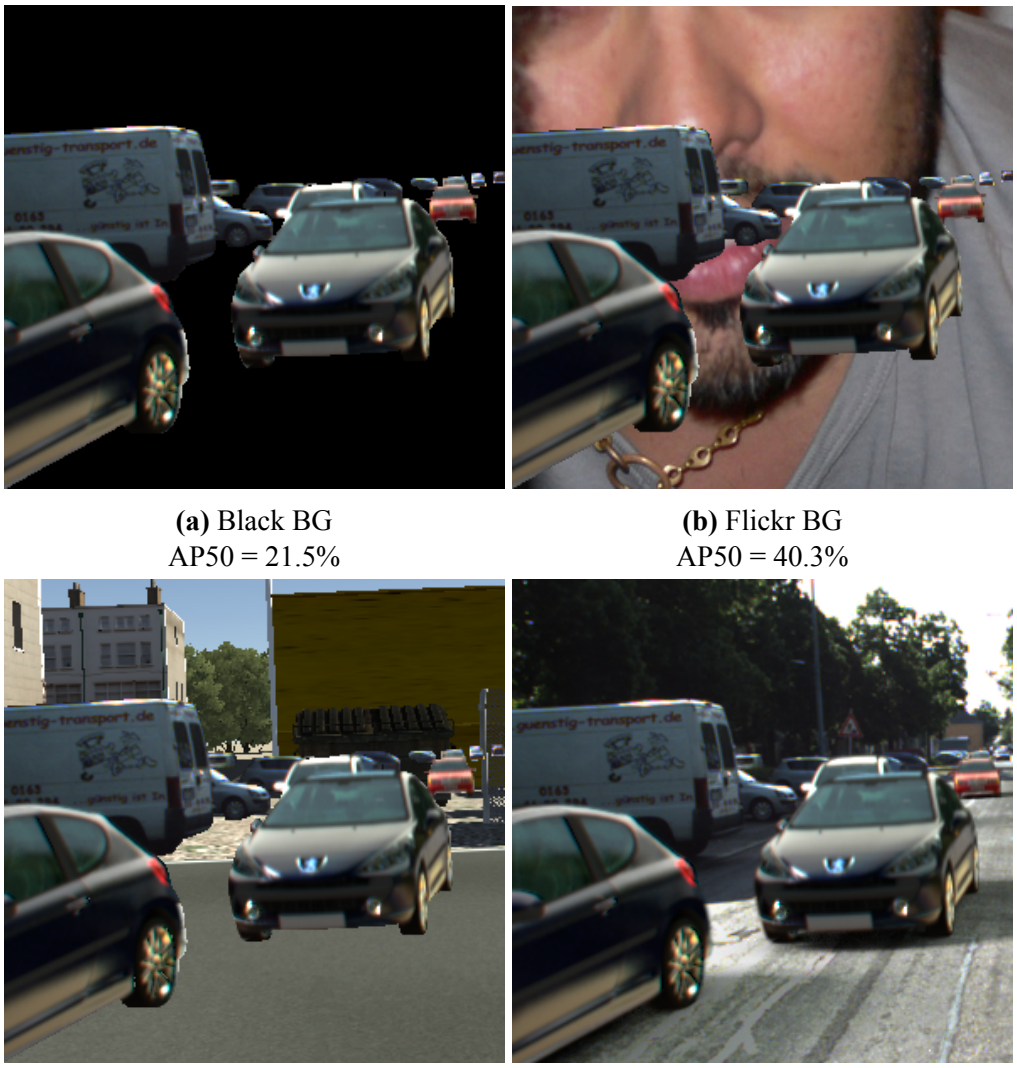
3.4.5 Dataset Size And Variability

The potential usefulness of data augmentations comes mainly from its ability to realistically expand a relatively small dataset and train more generalizable models. We analyze here the impact of dataset size on training using real, synthetic and augmented data. Figures 3.8a and 3.8c show the results obtained by training on various number of real images with and without augmentation, respectively. The models trained on a small real dataset suffer from over-fitting that leads to low performance, but then slowly improve when adding more training images. Meanwhile, the augmented datasets reach good performance even with a small number of real images and significantly improve

when increasing dataset size outperforming the full real data by a large margin. This suggests that our data augmentation can help improve the performance of not only smaller datasets, but also medium or even larger ones.
In figure 3.8b, the total size of the augmented dataset is fixed to 4000 images by adjusting the number of augmentations for each real dataset size. In this case the number of synthetic car instances is equal across all variants which only differ in the number of real backgrounds. The results highlight the crucial role of the real background diversity in the quality of the trained models regardless of the number of added synthetic cars.
Even though fully synthetic data generation methods can theoretically render an unlimited number of training images, the performance gain becomes smaller as the dataset grows larger. We see this effect in figure 3.8d where we train the model using various randomly selected subsets of the original VKITTI dataset. In this case, rendering adding data beyond 4000 images doesn't improve the model performance.

3.4.6 Realism and Rendering Quality

Even though our task is mainly concerned with segmenting foreground car instances, having a realistic background is very important for learning good models. Here, we analyze the effect of realism of the background for our task. In Figure 3.9 we compare models trained on the same foreground objects consisting of a mix of real and synthetic cars, while changing the background using the following four variations: (i) black background, (ii) random Flickr images [155], (iii) Virtual KITTI images, (iv) real background images. The results clearly show the important role of the background imagery and its impact even when using the same foreground instance. Having the same black background in all training images leads to over-fitting to the background and consequently poor performance on the real test data. Using random Flickr images improves the performance by preventing background over-fitting but fails to provide any meaningful semantic cues for the model. VKITTI images provide better context for foreground cars improving the segmentation. Nevertheless, it falls short on performance because of the appearance difference between the foreground and background compared to using real backgrounds.



(a) Black BG
AP50 = 21.5%

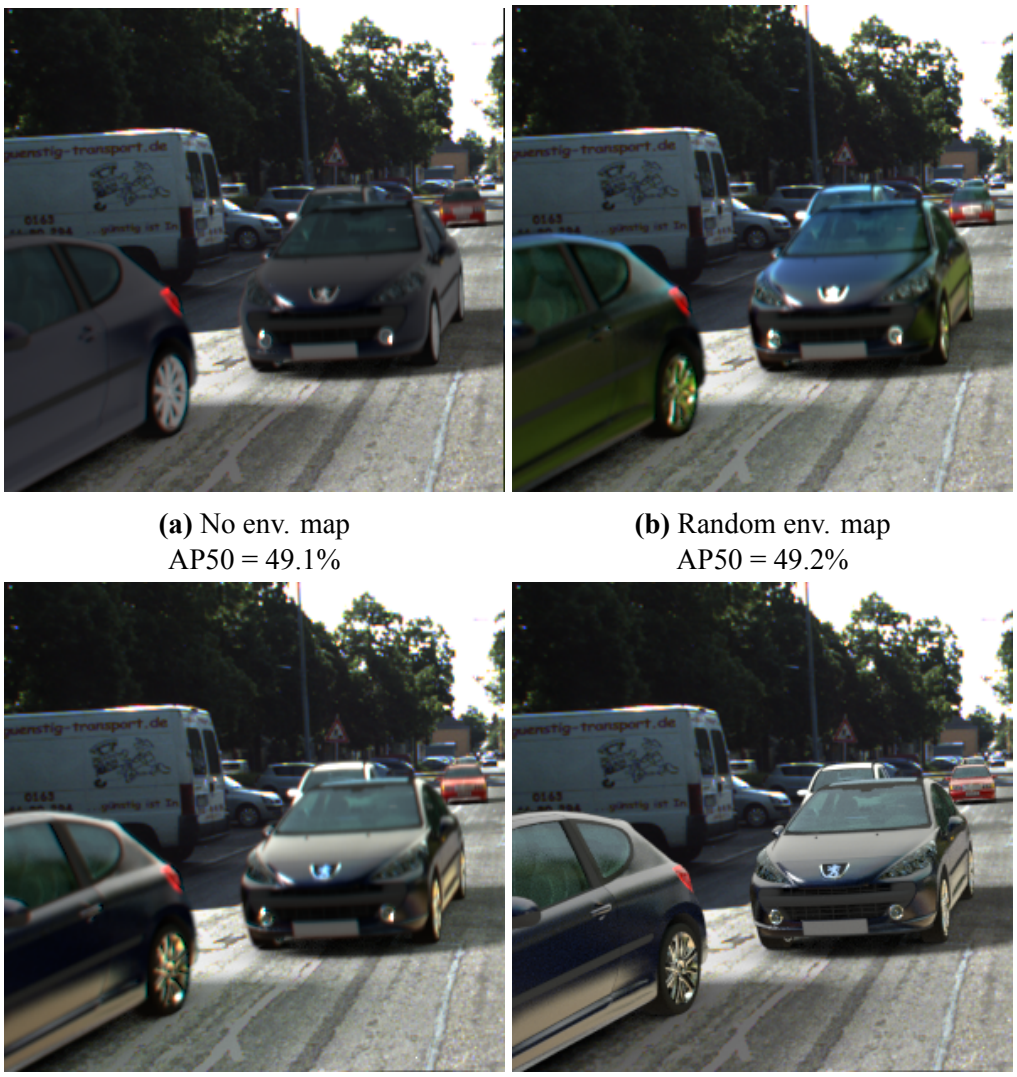
(b) Flickr BG
AP50 = 40.3%

(c) Virtual KITTI BG
AP50 = 47.7.3%

(d) Real BG
AP50 = 49.7%

Figure 3.9. Comparison of performance of models trained on augmented foreground cars (real and synthetic) over different kinds of background.

Finally, we take a closer look at the importance of realism in the augmented data. In particular, we focus on three key aspects of realism that is, accurate reflections, post-processing and object positioning. Reflections are extremely important for visual quality when rendering photo-realistic car models (see Figure 3.10) but are they of the same importance for learning instance-level segmentation? In Figure 3.10 we compare aug-



(a) No env. map
AP50 = 49.1%

(b) Random env. map
AP50 = 49.2%

(c) True env. map
AP50 = 49.7%

(d) No postprocessing
AP50 = 43.8%

Figure 3.10. Comparison of the effect of post-processing and environment maps for rendering.

mented data using the true environment map to that using a random environment map chosen from the same car driving sequence or using no environment map at all. The results demonstrate that the choice of environment map during data augmentation affects the performance of the instance segmentation model only minimally. This finding means that it's possible to use our data augmentation method even on datasets that do

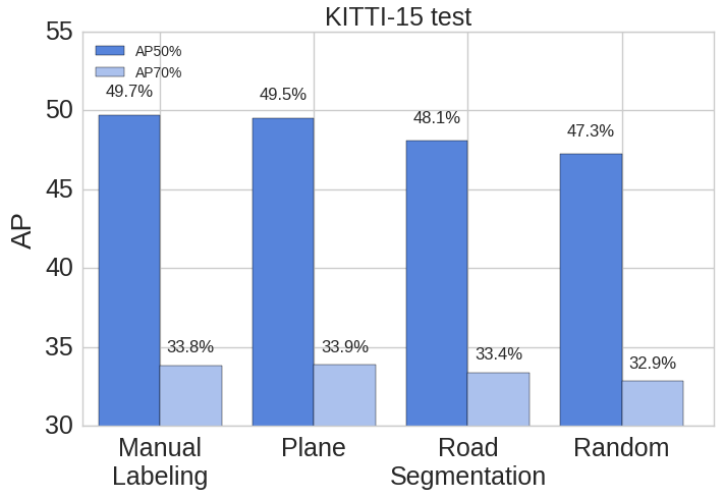


Figure 3.11. Results using different techniques for sampling car poses.

not provide spherical views for the creation of accurate environment map. On the other hand, comparing the results with and without post-processing (Figure 3.10c+3.10d) reveals the importance of realism in low-level appearance.

Another important aspect which can bias the distribution of the augmented dataset is the placement of the synthetic cars. We experiment with 4 variants: (i) randomly placing the cars in the 3D scene with random 3D rotation, (ii) randomly placing the cars on the ground plane with a random rotation around the up axis, (iii) using semantic segmentation to find road pixels and projecting them onto the 3D ground plane while setting the rotation around the up axis at random, (iv) using manually annotated tracks from birdseye views. Figure 3.11 shows our results. Randomly placing the cars in 3D performs noticeably worse than placing them on the ground plane. This is not surprising as cars can be placed at physically implausible locations, which do not appear in our validation data. The road segmentation method tends to place more synthetic cars in the clear road areas closer to the camera which covers the majority of the smaller (real) cars in the background leading to slightly worse results. The other 2 location sampling protocols don't show significant differences. This indicates that manual annotations are not necessary for placing the augmented cars as long as the ground plane and camera parameters are known.

3.5 Conclusion

In this chapter, we have proposed a new paradigm for efficiently enlarging existing datasets using augmented reality. The realism of our augmented images rivals the realism of the input data, thereby enabling us to create highly realistic datasets at a large scale which are suitable for training deep neural networks. The main limitation of our current model for data generation is that synthetic objects can only be placed on top of real images and thus cannot be partially occluded by real objects. This can be solved by using pixel accurate depth information if such information is available. In the future we plan to reduce the manual effort and improve the realism of our method by making use of additional labels such as depth and optical flow or by training a generative adversarial method which allows for further fine-tuning the low-level image statistics to the distribution of real-world imagery and makes it possible to expand it to other datasets and tasks.

Chapter 4

Geometric image synthesis

4.1 Introduction

Methods for generating natural images from noise or sparse input have gained significant interest in recent years with the developments in Generative Deep Neural Networks. Specifically, Generative Adversarial Networks (GANs)[61], allowed for trainable models that can produce natural-looking images with little or no prior knowledge input just by learning to imitate the distribution in a target image set. While the generated images often consist of realistic looking local patterns, the overall structure of the images can be inconsistent. Using more sparse cues, like edge maps or semantic segmentation[80], introduces some local control over the output but does not address the global structure. Further, recent works have addressed the problem of global consistency by generating the image at different scales [27] or using two separate global and local networks [206]. These solutions, nevertheless, address the global 2D structure of the image but not the 3D structure of the scene. This is evident when trying to generate an object in a different pose than those present most commonly in the training dataset (see figure 4.1b, 4.1c). While image generation from semantic segmentation can produce visually impressive images, it is not clear whether it can produce new training data for other vision tasks. This could be attributed to two factors: (i) The sparse input makes the image generation problem largely under-constrained leading to inconsistent image structure; (ii) The lack

(a) Cycles renderer **(b)** CRN [27] **(c)** Pix2PixHD [206] **(d)** GIS (Ours)

Figure 4.1. (a) The result of a state-of-the-art Physically-based Renderer (”Cycle” Renderer). (b,c) Results of two other deep neural network based image generation methods [27],[206]. While in both cases local image patches looks plausible the whole image does not look realistic. (d) Our GIS framework can realistically synthesize the car object with a specific pose using a deep neural network.

of control parameters over the image generation process (e.g.Pose and color of objects) makes it hard to define the desired attributes of the output image. On the other hand, generating natural images from known 3D geometry, texture and material properties through rendering engines has been widely used to generate training data for various computer vision tasks. While physically-based rendering engines aim at accurately reproducing the physical properties of light-material interactions, most available rendering engines use a set of carefully designed approximations, in order to reduce the computational complexity and produce results that are visually appealing to humans. Rendered images accurately matches the input scene structure but differ in local appearance from real images due the disparity between the real capturing process and the approximations in the software rendering pipeline. Previous works [195] pointed to the performance gap between synthetic and real data when used for training a task like semantic or instance segmentation. The other limitation of rendering engines is that they require accurate and complete information about the objects and the scene, namely, detailed 3D geometry, texture and material properties, lighting information, environment maps, and so on. This usually requires laborious manual work by experts to set up the 3D scene.

In this work, we propose a geometry-aware image generation method, called Ge-

ometric Image Synthesis (GIS), that leverages various types of scene information, like geometry and segmentation, to create realistic images which match the desired scene structure and properties. The network is trained with two objectives, the first is a supervised loss where the goal is to learn a mapping from the multi-channel input to an RGB image that matches the input structure. The second is an adversarial loss that learns to compare generated and real images enforcing the generator results to look similar to real data. We explore different input modalities like normals, depth, semantic and material segmentation and compare their usefulness. Using this rich input we are able to show visually a clear improvement over existing state-of-the-art image generation approaches, e.g. [27] [80] [206].

The goal of our approach is not only to generate visually realistic images, but also to explore whether the images generated can be useful for training other networks for various computer vision tasks. The advantage of using a trainable model instead of a software rendering engine is two-fold. Firstly, it can produce realistic looking images from geometry and segmentation while learning, from training data, to implicitly predict the remaining rendering parameters (e.g. material properties and lighting conditions). Secondly, the trainable model has the advantage of producing images that are fine-tuned to specific characteristics of the training dataset by leveraging the adversarial loss. For instance, it can capture the specific noise distribution and color shifts in the data.

In order to demonstrate the abilities of our GIS framework, we perform two types of experiments. In the first, we utilize an augmented reality dataset where synthetic vehicles were realistically rendered into a scene using "Cycles" renderer from Blender [2]. We use the normals, depth and material labels as input and the rendered images as the target in the supervised loss, while using real car images to train the discriminator in the adversarial loss. In this way, our network is able to generate realistic looking images (see Fig. 4.1d and supplementary video[1]) similar to the rendered data from [2] (see fig. 4.1a). In fact, we train the GIS network to give 9 diverse output-images and observed that each image captures a different lighting condition (e.g. direct sunshine, clear sky, or cloudy), all present in the training data. Using our trained network, we produce a new dataset of 4000 augmented images of car objects on top of real driving images. This dataset is

[1] `https://youtu.be/W2tFCz9xJoU`

used to train a state-of-the-art instance segmentation network, here Mask R-CNN [68]. This improves the performance of Mask R-CNN over the original augmented data [2]. In the second experiment, we demonstrate how our network can be trained directly using real images only. For that we utilize the Linemod dataset [71] that includes images of several objects and their 3D scanned models in addition to the corresponding 6D pose of the objects in each image. We show that using our GIS Network we are able to generate large amount of training data that helps improve the performance of instance segmentation. To summarize, our **contributions** are as follows :

- We introduce a trainable deep neural network, called Geometric Image Synthesis (GIS), that is able to generate geometry-consistent images from limited input information, like normals and material segmentation, While the remaining aspects of the image, e.g. lighting conditions, are learned from training data.
- We qualitatively show that our framework generalizes to novel scene geometries, objects and segmentation, for both synthetic and real data.
- We quantitatively show that our network can synthesize training data that improves the performance of a state-of-the-art instance segmentation approach, here Mask R-CNN ([68]). To the best of our knowledge, this was the first time that synthesized training data from a Neural Network is used to advance a state-of-the-art instance segmentation approach.

4.2 Related work

Synthetic Datasets.

The success of supervised deep learning models has fueled the demand for large annotated datasets. An alternative to tedious manual annotation is provided by the creation of synthetic content, either via manual 3D scene modeling [167, 226] or using some stochastic scene generation process [36, 128, 199, 204]. Mayer *et al*.[126, 127] demonstrate that simple synthetic datasets with “flying 3D things” can be used for training

stereo and optical flow models. Ros *et al*.[167] proposed the SYNTHIA dataset with pixel-level semantic segmentation of urban scenes. In contrast, Gaidon *et al*.[50] propose "Virtual KITTI", a synthetic dataset reproducing in detail the popular KITTI dataset [56]. Richter *et al*.[164, 165] and Johnson-Roberson *et al*.[86] have been the first to demonstrate that content from commercial video games can be accessed for collecting semantic segmentation, optical flow and object detection ground truth.

An alternative to synthesizing the entire image content is to render only specific objects into natural images. The simplest approach is to cut object instances from one image and paste them onto random background images [47] using appropriate blending or GAN-based refinement [216]. More variability can be obtained when rendering entire 3D CAD models into the image. Several works consider the augmentation of images with virtual [30, 67] or scanned humans [28, 203]. In contrast, [2] considers the problem of augmenting scenes from the KITTI dataset with virtual vehicles for improving object detectors and instance segmentation algorithms. In particular, it has shown that a well performing instance segmentation method, here MNC [35], can be considerably improved by intelligently generating additional training data.

While great progress has been made in rendering photo-realistic scenes, creating the required content and modeling all physical processes (e.g., interaction of light) correctly is a non-trivial and time-consuming task. In contrast to classical rendering, we propose a generative feed-forward model which maps an intermediate representation of the scene to the desired output. The geometry and appearance cue of this intermediate representation are easily obtained using fast standard OpenGL rendering.

Conditional Adversarial Learning.

Recently, generative adversarial networks (GANs) [61] have been proven to be powerful tools for image generation. Isola *et al*.[80] formulate the image-to-image translation problem by conditioning GANs on images from another domain and combining an adversarial with a reconstruction loss. Yang *et al*.[217] introduce an additional diversity loss to generate more diverse outputs. Wang *et al*.[206] propose a multi-scale conditional GAN architecture for generating images of up to 2 Megapixel resolution. Wang

et al.[208] use a GAN to synthesize surface normals and another GAN to generate an image from the resulting normal map. GANs' major advantage is that they do not require matching source and target images but rather enforce the generator to produce images that match the target data distribution. We exploit this by adding an Adverserial loss in our GIS framework such that the generated images are realistic. Besides, we explore a richer set of input modalities compared to just raw images [37, 160, 208] or semantic segmentations [80, 206, 217] for generating higher-quality outputs. We demonstrate that our model compares favorably to the High-Resolution Image Synthesis model of Wang *et al*.[206] (see fig 4.1d).

Feed-Forward Image Synthesis.

Dosovitskiy *et al*.[44] consider an alternative formulation to GANs using feedforward synthesis with a regression loss. Their work demonstrates that an adversarial loss is not necessary to generate accurate images of 3D models given a model ID and a viewpoint. In the same spirit, Chen *et al*.[27] consider the problem of synthesizing photographic images conditioned on semantic layouts using a purely feedforward approach. They demonstrate detailed reconstructions at resolutions up to 2 Megapixels, improving considerable upon the results of Isola *et al*.[80]. Our work also uses a feedforward formulation for the image synthesis problem. Unlike [27], however, our focus is on synthesizing controllable, high quality images. Thus, we consider 3D geometry and segmentation (semantic or material) as input, provided by a simple OpenGL rendering unit.

4.3 Method

A general image generation process can be defined as a mapping $\mathcal{F} : \{G, A, E\} \rightarrow I$ from scene description $\{G, A, E\}$ to an RGB image I. The scene description consists of three parts, (i) the geometry parameters G which include the poses and shapes of objects, (ii) the appearance parameters A which describe the objects' materials, textures and transparency and finally (iii) the environment parameters E which describe global

conditions of the scene that affect all objects such as lighting, camera parameters, and the environment. In this work in contrast, our goal is to train a mapping $\mathcal{R} : \{G, S\} \rightarrow I$ that can produce natural images from a given geometry G and material segmentation S only, without the knowledge of exact appearance A or environment parameters E. Similar to semantic segmentation, the material segmentation labels each pixel with a specific material label (e.g.metal, glass etc.) from a pre-defined set of materials without providing any properties or parameters of the material. The task of the network is to learn the unknown parameters from the training data directly and apply them to generate images from new input geometry.

The target image I, which is used for training, can either be a real image of a known scene geometry or a rendered synthetic image obtained through a high-quality software renderer. While learning image generation directly from real images is desirable, it is often difficult to obtain geometry and material labels which are pixel-accurately aligned with real-world images. For this reason, it is possible to exploit synthetically rendered data using a state-of-the-art physically based renderer as supervised target while using an adversarial loss with real images to acquire realistic looking results. Using realistically rendered images also gives us fine-grained control over the data, which we exploit to conduct various experiments for analyzing our model. Additionally, we demonstrate how our method can be trained directly using real images for the supervised loss when an exact 6D pose of the objects in the image is known.

4.3.1 Input Representation

Geometry plays a major role in defining the appearance of an object in an image since it defines its shape in addition to its shading through interaction with light. Providing the geometry as an input changes the learning objective from learning to create objects to learning a correlation between geometry and appearance. This makes the network more generalizable to new geometries as we show in later expriments. To use geometry in a deep neural network, it is important to find a compact representation of the object's 3D surface. While meshes are one of the most common representations for 3D

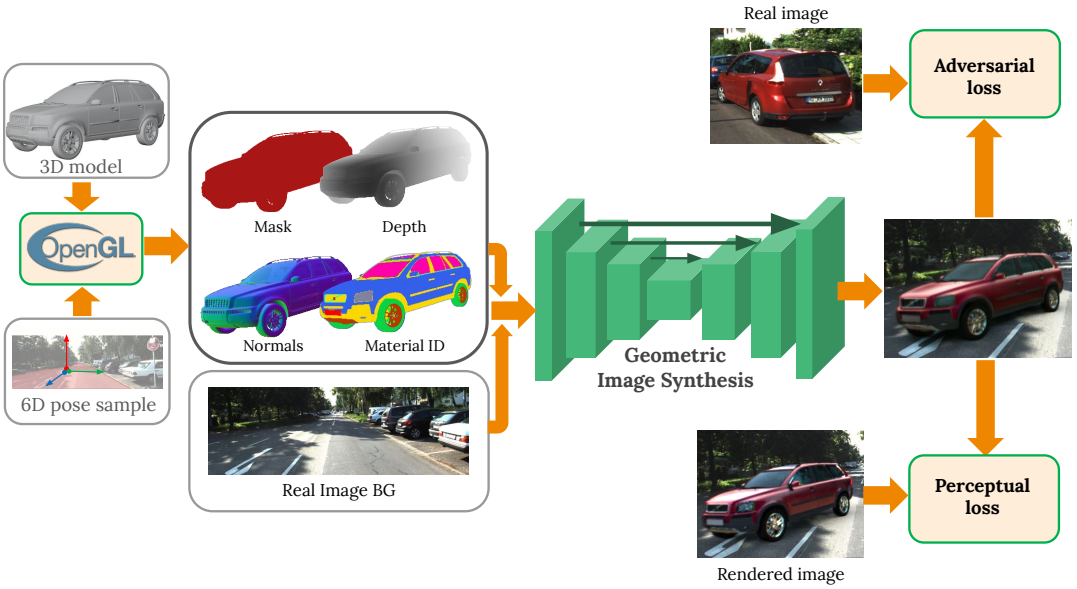


Figure 4.2. Overview of our approach. We propose Geometric Image Synthesis framework, feed-forward architecture for synthesizing RGB images based on geometric and semantic cues. Here we show the case where a car is augmented onto an empty road. Compared to existing image synthesis approaches, our model benefits from a rich set of input modalities, while learning realistic mappings which generalize to novel geometries and segmentations, and integrate the objects seamlessly into provided image content.

objects, they are problematic in the context of convolutional neural networks due to their irregular 3D structure. Another popular representation of 3D objects are voxels. Voxel-based representations can be handled using 3D convolutions [166] but suffer from two shortcomings: high memory requirements and comparably low resolution.

A common 2D representation of 3D shapes is their depth in the camera view. The advantage of such an image-based geometry representation is that it can be directly processed with a regular 2D convolutional neural network. Nevertheless, the object appearance does not usually depend on its absolute depth except for secondary effects like lens blur and environmental distortions. Rather, it depends on the small changes in depth between neighbouring points which defines the relative surface orientation with respect to the light source. This can be better characterized by computing the surface normals

in the camera coordinates system at each point of the visible surface.

The main advantage of learning the image generation from geometry compared to rendering is the ability to exploit high-level semantic and context cues to predict the appearance of an object. This allows it to *learn* non-geometric attributes of the object appearance such as material parameters, lighting, environment reflections and texture directly from data. Using semantic [27] or instance segmentation [206] can help the network to learn the appearance of semantically similar objects across multiple examples. Nevertheless, it can be challenging in cases where a semantic class has a large variety in appearance, e.g.cars with different models and colors. We propose in this work to use material segmentation instead. Each pixel in the segmentation input gets a label from a pre-defined set of materials (e.g.metal, plastic, glass etc.). This does not include any material properties or parameters. Rather, it groups parts made of similar materials together allowing the network to learn the material appearance model from multiple objects in different contexts, e.g.different lighting conditions. This results in more generalization power since the material appearance is often independent of the object class, pose or shape. We expect this labeling to be particularly effective when generating objects that consist of a small number of materials but vary significantly in shape, e.g., cars.

4.3.2 Network Architecture

We now define our network architecture in detail. As discussed before, our goal is to learn a mapping from an intermediate representation to a natural RGB image using a deep neural network. As input layers to our network, we use the normal map, the depth map, object mask and material segmentation of the object which can be easily obtained using OpenGL based rendering. Additionally, by providing the network with a background image I_{bg}, the network can learn to augment synthetic objects realistically into real images, e.g. add shadows underneath a synthetic car and blending edges.

Fig. 4.2 illustrates the input layers to our network. Let N, D, S be the 2D images representing the normal map, depth map and semantic segmentation of the input object respectively. Let M denote the material label where each pixel is represented by a one-

hot encoding vector which identifies its material ID, see Fig. 4.2 for an illustration. We are now ready to formally represent the mapping as $\mathcal{R} : \{N, D, S, M, I_{bg}\} \rightarrow I$.

For our generator R, we follow Chen *et al.*[27] and use a feed-forward coarse-to-fine network architecture for image synthesis. More specifically, we leverage a cascade of convolutional layer modules C starting from a very low resolution input and growing to modules of higher resolutions (Fig. 4.2). Each convolutional module C_i has an input resolution of $w_i \times h_i$ and produces a feature map F_i of the same size. C_i receives the feature map F_{i-1} from the previous module, upsampled to $w_i \times h_i$, and concatenated with the input downscaled to the same resolution. The following layer, C_{i+1}, operates at twice the resolution of the previous layer ($2w_i \times 2h_i$), and receives the feature maps F_i and the input rescaled to $2w_i \times 2h_i$. Each convolutional module C_i consists of an input layer, intermediate layer and output layer, each of which is followed by a set of convolutions, a layer normalization and a leaky ReLU nonlinearity. The output layer of the final module is followed by a 1×1 convolution applied to the feature map and normalized to obtain the synthesized image. For the adversarial discriminator D, we adopt a fully convolutional network architecture consisting of 5 convolutional layers each followed by a leaky ReLU with a stride of 2 for all except the last layer. The discriminator's output is a 2D binary map where each value describes the discriminator classification of a patch as real or synthesized by the generator. This is specially useful when synthesizing objects into real images where the same image would contain both real and synthetic patches. To further stabilize the adversarial training, we employ the simple discriminator gradient regularization method proposed in [168]

4.3.3 Training

We train the generator R in our GIS framework to produce synthesized images I_s that resemble the target images I_t obtained using the "Cycles" rendering engine while at the same time being close in appearance to real images in order to "confuse" the adversarial discriminator. Effectively, the task of the network is to learn the process of generating images, directly from the target images, given $\{N, D, S, M, I_{bg}\}$ without information such as lighting, environment map or material properties. Those properties are estimated

by the network during training and combined with the geometry and segmentation input to produce a high quality image. To achieve this, we choose to compute the perceptual loss (feature matching loss) as proposed in [54] between the generated and target image. The goal of the perceptual loss is to match the feature activations produced by I_t and I_s at various convolutional levels through a perception network, e.g., VGG. This helps the network to learn fine-grained image patterns while also preserving the global object structure. We use VGG pre-trained on ImageNet as our perceptual network. Let us denote this network by V, and let V_l denote a layer of this network. The loss between I_t and I_s is given by

$$\mathcal{L}^P = \sum_l S_l \, \lambda_l \, \|V_l(I_t) - V_l(I_s)\|_1$$

where $V_l(\cdot)$ denotes the feature activations of VGG at layer l and S_l is the binary mask of the object rescaled to the size of V_l. The GIS framework can also learn to synthesize objects on top of real images. In this case, our goal is to create augmented images by learning not just the target object appearance but also its interaction with the environment in the real image, including shadows, reflection and blending at the object's edges. Towards this goal, we add the background image to the input and train the network using an ℓ_1 loss for the background areas outside the object mask $\mathcal{L}^B = (1 - S)\|I_t - I_s\|_1$. The adversarial discriminator D_s is trained to segment the augmented images by the generator into real and synthetic parts using the Binary Cross-Entropy loss $\mathcal{L}^D = \mathbb{E}[\log(S - D_s(I_s))]$. By replacing the synthetic object mask S with its inverse $(1 - S)$, we define an adversarial loss for the generator $\mathcal{L}^A = \mathbb{E}[\log((1 - S) - D_s(I_s))]$ that evaluates the realism of the synthesized objects.

4.3.4 Diversity

Synthesizing images from geometry and segmentation alone is an ill-posed problem. That is, for a specific set of inputs, there are infinite plausible outputs due to different possible lighting conditions, object colors etc. Thus, we task our network to produce K diverse outputs from the last layer using multiple-choice learning [27, 65]. More specifically, we compute the loss for each of these outputs, but only back prop-

agate the gradient of the best configuration for the foreground prediction, while averaging the background predictions as none of them should deviate from the input: $\mathcal{L} = w \min_k [\mathcal{L}_k^P + \mathcal{L}_k^A] + (1-w)\frac{1}{K}\sum_k \mathcal{L}_k^B$. where w is a weight inversely proportional to the number of pixels of the synthesized object(s). Note that only the foreground object with the smallest loss is taken into account, thus the min operator effectively acts as a multiple choice switch. This encourages the network to output a diverse set of images to spread its bets over the domain of possible images that could be produced from the current input. In all our experiments we use $K = 9$ as the number of diverse outputs, see Fig. 4.5 for an illustration.

4.4 Experiments

To demonstrate the ability of our GIS network to synthesize realistic images, we perform a set of experiments which assess the quality and generalization capacity of the method. We mainly focus on two scenarios, outdoor driving and indoor objects. Realistically synthesizing augmented objects like cars or obstacles into real-world scenes is an important feature for expanding manually annotated training datasets. In the case of indoor objects, a learned network can be used to synthesize novel views of objects to provide extensive training data for various tasks. In the following experiments, we show that our GIS framework produces better images for training an instance segmentation network compared to a state-of-the-art software rendering engine.

4.4.1 Augmentation of KITTI-360 dataset

The augmented KITTI-360 [2] dataset features 4000 augmented images obtained from 200 real-world images through carefully rendering up to 5 high-quality 3D car models into each image using classical rendering techniques. The set of 28 car models have been manually created and placed to ensure high realism. Rendering was performed using the physically-based Cycles renderer in Blender and followed by a manually designed post-processing pipeline to increase the realism of the output. Additional scene

information like 360 degree environment maps and camera calibration has been used to ensure realistic reflections and good integration with the real image.

To train the GIS network, we use normals, depth, material segmentation and semantic masks of the augmented cars obtained using the augmented KITTI-360 pipeline. The material labels include 16 materials of different properties (e.g.plastic, chrome glass etc.) in addition to 15 car paint materials which differ only in color. We use the corresponding RGB images from augmented KITTI-360 as target images for training the parameters of GIS network. Mixing the real images with rendered cars presents an additional challenge since interactions between the inserted objects and the real background, e.g.shadows and transparencies, have to be taken into account. To deal with this, we input the background image to the network in addition to geometry and segmentation. The network's task is to learn the process of synthesizing cars realistically and blend them into the surrounding environment by appropriately adding reflections, shadows and transparencies, amongst others.

During inference, we obtain a new set of car model positions and orientations following the procedure in [2]. We render the mask, depth, normals and material labels from the camera viewpoint and use them as input to our GIS network. Note that during the inference phase, we do not require a sophisticated rendering pipeline, like Cycles renderer, since normals, depth maps and segmentation can be obtained directly using a simple OpenGL based renderer. We then leverage the trained GIS model to create a new dataset of 4000 augmented images with new car poses and combinations.

Qualitative evaluation:

Fig. 4.3 shows augmented images produced by our GIS framework when trained on the augmented KITTI-360 dataset. Note that the synthesized cars exhibit realistic appearance properties like shading, shadows, reflections and specularity, despite the fact that this information is not provided to our model. The material labeling of the cars allows the model to tune the synthesis process to each material. Importantly, note that the material label is just a semantic label of material and does not contain any information with respect to the physical properties of the material. Interestingly, our model is able to learn

Figure 4.3. Images from KITTI-360 dataset augmented with cars synthesized using our GIS framework (Real image without augmentation in upper left corner).

the transparency property of the material with label ”glass” from data, without providing any alpha channel or explicitly modeling transparency. Additionally, the model is able to replicate camera effects such as blur and chromatic aberrations, which are present in the augmented KITTI-360 dataset.

Quantitative evaluation:

To verify the effectiveness of data produced by our model, we train the state-of-the-art Mask R-CNN model [68] for car instance segmentation using the images produced by our network. Alongside, we also train the same model with images from the original Cycles rendering pipeline from [2] with the same 3D car models and poses, and a baseline model using the unaugmented real images from KITTI-360. We evaluate all models on the KITTI-2015 training set. The results are presented in Table 4.1. We observe that the model trained on images synthesized by the GIS network significantly outperforms the one trained only on real data, and marginally outperform the highly-tuned data from [2]. This clearly indicates that our model does not only learn to imitate the training data, but also the adversarial loss can contribute in make the resulting appearance more realistic and, therefore, more effective in training.

4.4.2 Generalization and Ablation Study

A key feature of our GIS framework is that it learns a mapping from any geometry to natural images and is not limited to a specific set of objects or shapes. In the following sections, we present an extensive experimental study to demonstrate that our model learns a generic image formation function and does not overfit or limit itself only to objects of certain geometry and material.

To show generalization ability, we present the network with two tasks: (i) synthesize seen objects with material combinations never seen before, and (ii) synthesize learned materials on new, unseen, geometries. In Fig. 4.4a we show the results of our model applied to the ”Monkey” model from blender with different material labels applied to it. Our results clearly demonstrate that the material properties have been learned by the network independently from the geometry. In Fig. 4.4b we replace the car paint with the chrome material previously seen in the training data only on the car wheel rims. The resulting image looks realistic, demonstrating that the material properties learned from one part of the model can be transferred to other, geometrically different, parts by simply changing the material label. Using the diversity loss, described in Section 4.3.4,

(a) (b)

Figure 4.4. (a) GIS output for a monkey model with material labels: car paint, chrome and glass. (b) GIS output for a car with material label chrome.

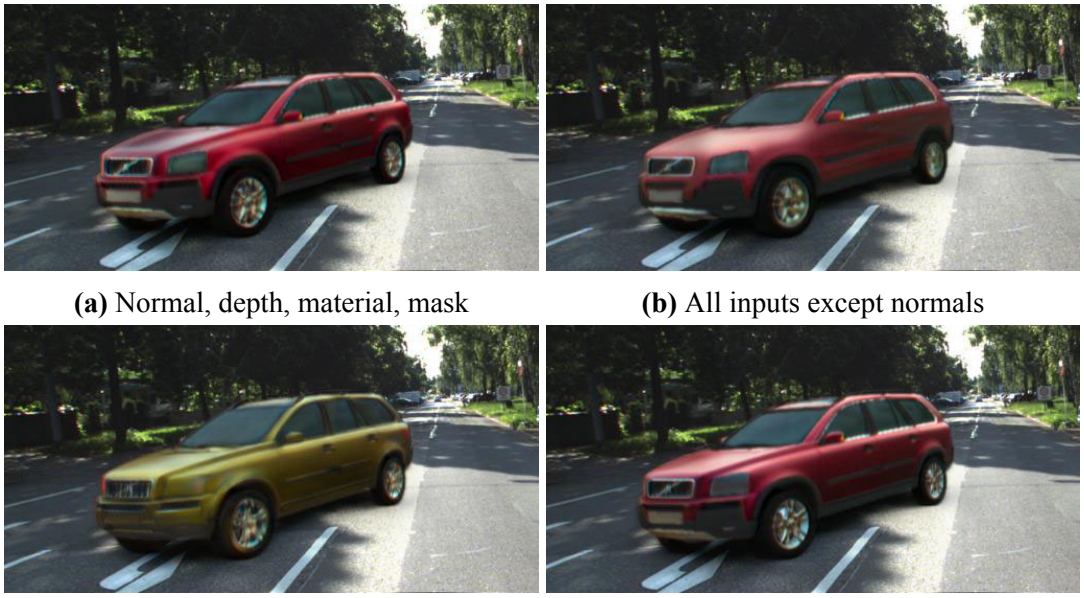
Figure 4.5. Three diverse outputs obtained from GIS on the KITTI-360 dataset. Note how the renderings vary in lighting conditions and reflections.

our GIS model produces 9 different possible images from the same input. The results in Fig. 4.5 show how the network can learn different lighting conditions (direct light, cloudy etc.) without providing any explicit lighting information.

To better understand the importance of different input modalities, we perform an ablation study where we train the GIS model from scratch using all inputs excluding one at a time. We qualitatively compare the results in Fig. 4.6. When normals are not used for training, the output images become smooth and lack fine geometric details. Excluding depth maps from the input, on the other hand, leads to no noticeable difference. We hypothesize that this is due to the fact that most of the shading of the object can be modeled based on the local geometry cues that are expressed well in the normals, but little difference in appearance relates to the absolute depth of an object. In contrast, removing the material segmentation results in blurry images.

Dataset	IoU 50%	AP
Real KITTI-360	58.80%	31.92%
Augmented KITTI-360	66.68%	37.88%
GIS (ours)	67.74%	38.69%

Table 4.1. Accuracy of Mask R-CNN when trained with real, augmented or GIS generated images.



(a) Normal, depth, material, mask **(b)** All inputs except normals

(c) All inputs except material ids **(d)** All inputs except depth

Figure 4.6. Output of GIS using various types of inputs. Note that GIS with all four inputs, or all inputs except depth, synthesizes realistic images.

4.4.3 Novel view synthesis on Linemod dataset

The Linemod dataset was introduced by Hinterstoisser *et al*.[71] for evaluating 6D object pose estimation algorithms. This dataset contains real images of multiple known objects, each of them annotated with a 6 degree-of-freedom pose. It provides 3D CAD models of all the objects in the dataset for which we annotated the materials present on each CAD model with a material label. Hence, using the 6D pose and the CAD model, we can obtain the normal map (N), material segmentation (S) and depth map (D) of objects.

The objective of this experiment is to use real images as target training data for

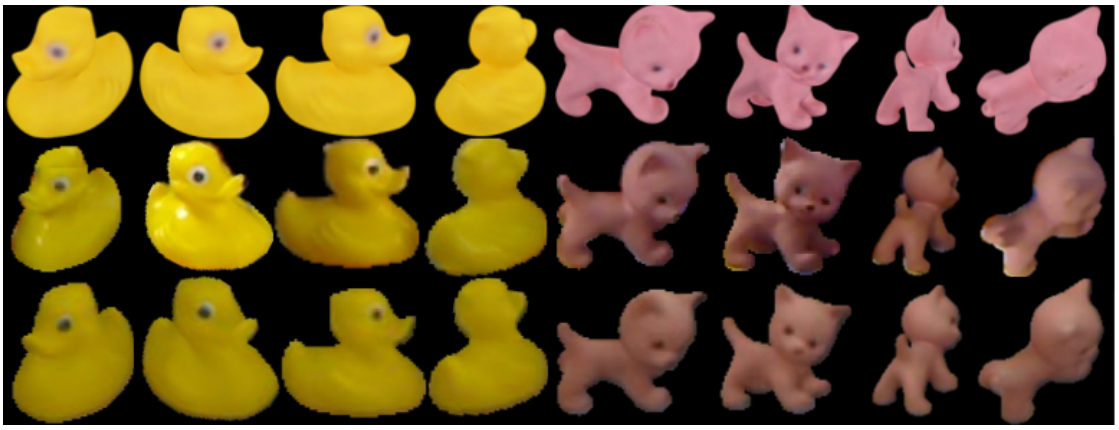


Figure 4.7. Top row contains rendered images. Middle row contains real images in similar poses. Bottom row contains images synthesized by our network. The middle row images are not seen during training phase. GIS is still able to synthesize novel views of objects realistically.

our network with the corresponding geometric information (N, S, D) as inputs. Unlike the KITTI-360 dataset, where the target data is acquired using a manually designed rendering framework, the availability of real images as target data in this case allows the network to model real world images and their statistics directly. We use the objects $Ape, Can, Cat, Duck, Eggbox, Holepuncher$ each with 1200 images and their pose annotations. We use 600 images of each object for training and the rest for testing.

Due to the generalisability of our method, we can use the trained GIS network to generate new images of the objects in previously unseen poses. To demonstrate the efficacy of the images produced by our GIS network for training, we compare them to a training set generated using the traditional OpenGL rendering engine. To this end, we use the two kinds of datasets to train the Mask R-CNN. First, we crop the rendered images and place them at random locations on NYU dataset [181] images. We repeat the same process for object images generated by our network. To evaluate the performance of Mask R-CNN, we test it on Linemod-Occluded dataset, proposed by Michel *et al*.[134] (note that we do not use this data while training our GIS network). We observe that the Mask R-CNN trained with rendered images performs at an average accuracy of 21.1% for all objects. On the other hand, the Mask R-CNN trained with our synthesized images performs at an average accuracy of 68.4%. This clearly indicates that the images synthe-

sized by our network are highly realistic and are useful for training other deep networks which cannot be achieved with rendered data. Qualitatively, our GIS generated images appear more similar to real images as shown in Fig. 4.7.

4.5 Conclusion

In this chapter, we have proposed GIS, a deep neural network which is able to learn to synthesize realistic objects by leveraging semantic and geometric scene information. Through various experiments we have demonstrated the generalization performance of our GIS framework with respect to varying geometry, semantics and materials. Further, we have provided empirical evidence that the images synthesized by GIS are realistic enough to train the state-of-the-art instance segmentation method Mask R-CNN, and improve its accuracy on car instance segmentation with respect to a baseline model trained on non-augmented images from the same dataset. We believe that our approach opens new avenues towards ultimately reaching the goal of photo-realistic image synthesis using deep neural networks.

Chapter 5

Joint learning of image synthesis and decomposition

5.1 Introduction

State-of-the-art sampling-based rendering engines (e.g., Mitsuba [82]) are able to generate photo-realistic images of virtual objects which are nearly indistinguishable from real-world photographs. However, this is not an easy task to accomplish since all intrinsic physical aspects of the virtual object must be accurately modeled, such as accurate 3D geometry, detailed textures and physically-based materials. While some of these intrinsics are abundant on the internet, such as the geometry of 3D objects (e.g. Turbosquid and 3D Warehouse), others are hard to obtain, such as high-quality materials – ideally in the form of a highly-accurate spatially-varying BRDF. In addition, sophisticated and slow rendering algorithms with many tunable parameters (lighting, environment map, camera model, post-processing) are required for turning 3D content into photo-realistic images. These parameters are often tuned individually with each rendered image, making it hard to create a large and diverse set of rendered images. On the other hand, obtaining a large number of real images which capture the complex interaction of light with scene geometry and surface properties is easy. This makes the idea of learning neural image synthesis from real images very attractive.

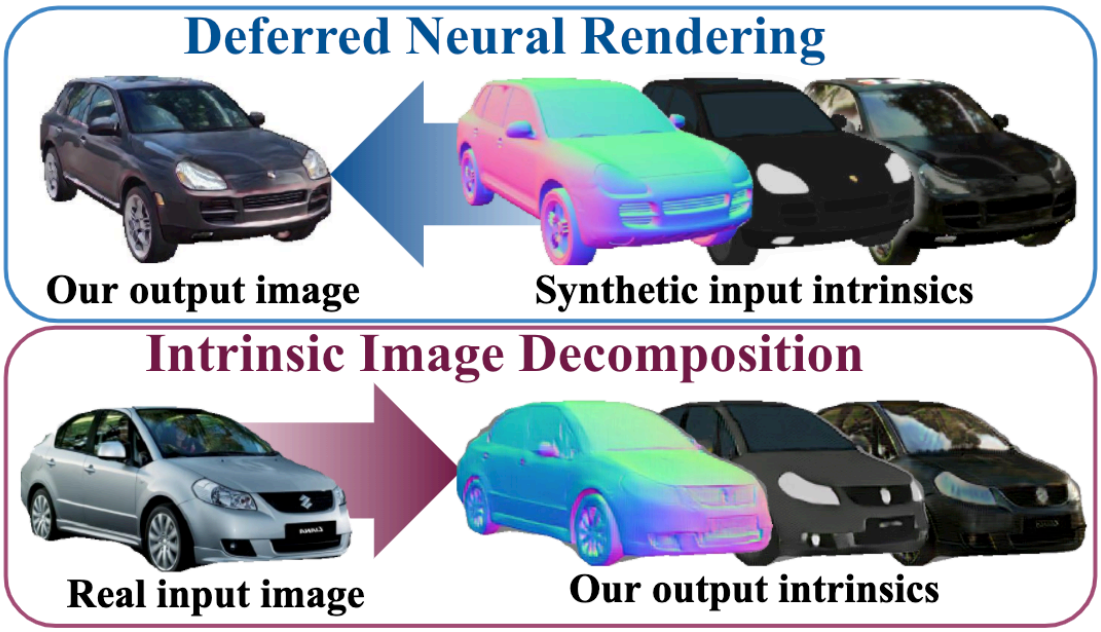


Figure 5.1. Deferred Neural Rendering and Intrinsic Image Decomposition. At training time, our model exploits normals, albedo and reflections from a small set of 3D models as well as a large set of *unpaired* RGB images of the same object category. Our model solves two tasks simultaneously: (i) generating photo-realistic images given the input geometry and basic intrinsic properties, and (ii) decomposing real images back into their intrinsic components.

Several works on conditional image generation [27, 80, 148, 206] have exploited paired datasets of real images with semantic information, including semantic segmentation [27, 148] and body part labels [106] for training realistic image synthesis models. However, such sparse inputs only allow limited control over the generated image. This limits the applicability of these methods, e.g., in virtual reality or video game simulations where precise control over the output is essential. Training a conditional image generation model from richer control inputs would require a large dataset of paired real images with pixel aligned intrinsic properties such as 3D structure, textures, materials and reflections. Obtaining such a dataset is hard in practice.

Our goal is to take a step towards learning a highly controllable realistic image synthesis model without requiring real world images with aligned 3D models. Our key insight is that learning the inverse task of intrinsic decomposition is helpful for learning

image synthesis from real images and vice-versa. We therefore train both, the forward rendering process and the reverse intrinsic decomposition process, jointly using a single objective as illustrated in Fig. 5.1. Inspired by recent results in unpaired image-to-image translation [77, 111, 228], we train our model using an small set of synthetic 3D models of an object category as well as a large unpaired dataset of real images of the same category.

Towards this goal, we exploit a technique from real-time rendering called *Deferred Rendering* which splits the rendering process into two stages and thus improves efficiency. In the first stage, the geometry of the scene along with its textures and material properties are projected onto a 2D pixel grid, resulting in a set of 2D intrinsic images which capture the geometry and appearance of the object. This step is efficient since it does not require physically accurate path tracing but relies on simple rendering operations. In the second "deferred" stage, lighting, shading and textural details are added to form the final rendered image. Our goal is to replace this second deferred stage of the rendering process with a neural network which we call *Deferred Neural Rendering* (DNR) network. To ensure that the input information is represented in the output image, we decompose it back into its intrinsics using a second Intrisic Image Decomposition (IID) network. However, we found that using this cycle alone leads to overfitting, especially in the IID network. To improve the IID network, we introduce a second *Decomposition cycle* in which we train the IID network to decompose real images.

Overall, our model follows a similar dual cycle training setup as proposed in [228] and [218]. However, an important conceptual difference to these works is that our task is not a one-to-one but a one-to-many mapping. Different realistic images can be generated from the same set of intrinsic maps as the intrinsics do not uniquely define the image. Likewise, a single image can be explained using different intrinsic decompositions due to projection from the higher dimensional intrinsics into the RGB image space.

We therefore introduce a shared adversarial discriminator between the input and the reconstruction at the end of each cycle. Our model enables both highly photo-realistic image synthesis and accurate intrinsic image decomposition. We summarize our main contributions as follows:

- We propose the Intrinsic Autoencoder, a method to jointly train photo-realistic image synthesis and intrinsic image decomposition using cycle consistency losses without using any paired data.
- We propose a shared discriminator network that enables better generalization and proves key for learning both tasks without paired training data.
- We analyze the importance of various model components using quantitative metrics and human experiments. We also show that our method recovers accurate intrinsic maps from challenging real images.

5.2 Related Work

Differentiable Rendering. A standard way of synthesizing images from a given geometry and material is to use rendering engines. Several works try to implement the rendering process in a differentiable manner, amenable to neural networks. The work of [14] used differentiable rendering with deformable face models for face reconstruction. The works of [119] and [93] proposed rasterization-based differentiable renderers but only support local illumination. In order to support more realistic image formation, some other works [25, 58, 59, 108] propose to back-propagate though path tracing. Differentiable rasterizers are relatively fast, but at the same time highly restrictive as they do not support complex global illumination. While differentiable path tracers produce more realistic images, they are usually quite slow, thus restricting their usage to specific applications. Another drawback of differentiable renderers is that they require a detailed representation of the rendering input in terms of geometry, illumination, materials and viewpoint. In this work, we bypass the specification of complex image formation by training a CNN to directly generate realistic images from given geometry and material inputs.

Neural Image Synthesis. Generative models such as Generative Adversarial Networks [61] and Variational Auto-Encoders (VAE) [98] are widely use to synthesize realistic images from a latent code. In contrast, our goal is to perform conditional image synthesis

which allows more fine-grained control over the image generation process. Some popular conditional image generation approaches are label-to-image translation [138, 146], image-to-image translation [27, 44, 80, 111, 206, 228] and text-to-image generation [75, 161, 215, 223]. Earlier works [27, 44, 80] on conditional image-to-image generation are mostly supervised with paired data from both domains. Several works [111, 228] propose a way to use unpaired data from both domains for conditional image generation. Other advances in conditional image generation include innovations in network architectures and loss functions for generating high resolution images [206] and generating multiple diverse images [32, 77, 79, 217]. In this work, we develop a model for photo-realistic geometry-to-image translation using only unpaired training data as supervision. Our work is closely related to [1] which also considers geometry-to-image translation, but requires paired training data. Our work belongs to the family of unpaired conditional image generation models with architecture and losses (e.g., shared discriminator) specialized for the geometry-to-image translation. Our model outperforms state-of-the-art unpaired image-to-image translation models [77, 228] by a large margin.

Intrinsic Image Decomposition is a long standing problem in computer vision. [11] poses the task as an optimization problem with a set of hand-crafted priors for shape, shading albedo etc. On the other hand supervised methods like [83, 175, 176] use synthetic data to train the model followed by refinement on real images. However, synthetic data might not capture all the real world statistics and models trained with synthetic data might now generalize well to real images. Recently, several self-supervised intrinsic decomposition methods have been proposed [110, 113, 121]. [113] uses single images during both training and inference stages. [110, 121] make use of multiview images or video sequences of the scene during training and infer on a single image . Our work falls in the realm of self-supervised intrinsic image decomposition. We do not use any paired synthetic data or multiview sequences to train out model. Instead, we rely on jointly training models for neural rendering and image decomposition.

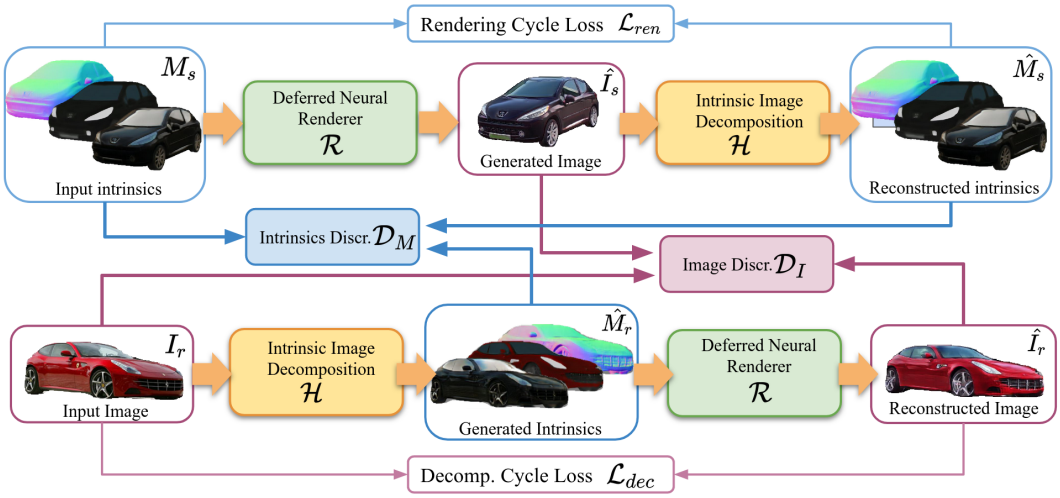


Figure 5.2. Intrinsic Autoencoder. Our model comprises two cycles: The first cycle (blue) auto-encodes a set of intrinsics rendered from 3D CAD models using appearance as latent representation. The second cycle (red) auto-encodes real images using image intrinsics as representation. Consistency is achieved through a combination of cycle losses and shared adversarial losses. Networks sharing the same weights are illustrated with the same color (green/yellow).

5.3 Method

Our Intrinsic Autoencoder model (Fig. 5.2) consists of two generator networks $\mathcal{R}$ and $\mathcal{H}$ for Deferred Neural Rendering and Intrinsic Image Decomposition, respectively. The Deferred Neural Rendering Network $\mathcal{R} : M \rightarrow \hat{I}$ takes as input a set of intrinsic maps $M = \{A, N, F\}$. The object's surface normal vectors in the view coordinate system $N \in \mathbb{R}^{H \times W \times 3}$ provides the Deferred Neural Rendering Network important information about the local shape of the object which is necessary for creating shading and reflection in the output image. The albedo $A \in \mathbb{R}^{H \times W \times 3}$ is a pixel-wise RGB value that describes the material or texture color at every pixel, ignoring any lighting effects. Finally, the environment reflections $F \in \mathbb{R}^{H \times W \times 3}$ are computed by projecting a high dynamic range environment map onto the 3D model. Note that this simple projection operation does not involve any complicated sampling or ray-tracing operations. As shown in our experiments, the Deferred Neural Renderer can also be trained with a subset of those inputs since it is able to compensate for the missing information. The DNR network

$\mathcal{R} : M \rightarrow \hat{I}$ transforms all the input intrinsics M into a realistic image $\hat{I} \in \mathbb{R}^{H \times W \times 3}$ that corresponds to the input intrinsics. Similarly, the Intrinsic Image Decomposition (IID) network $\mathcal{H} : I \rightarrow \hat{M}$ performs the opposite task by taking an input image I and predicting its intrinsics $\hat{M} \in \mathbb{R}^{H \times W \times 9}$.
Supervised training of $\mathcal{R}$ and $\mathcal{H}$ on real data is typically difficult due to the lack of real training image and intrinsics pairs (I_r, M_r). Instead, we use a combination of cycle-consistency losses and adversarial losses that require no paired training examples. This allows us to leverage a large dataset of real images $\{I_r^i\}_{i=0}^n$ and an unpaired set of synthetically generated intrinsic maps $\{M_s^i\}_{i=0}^m$. In the following, we detail our cycle consistency losses and the novel shared adversarial losses.

5.3.1 Cycle Consistency

Rendering Cycle. The goal of the rendering cycle is to train $\mathcal{R}$ in order to produce realistic images $\hat{I}_s = \mathcal{R}(M_s)$ from synthetic intrinsic maps M_s. To train $\mathcal{R}$ without paired data, we use the inverse transformation $\mathcal{H}$ which decomposes the predicted image $\hat{I}_s$ back into its intrinsic maps $\hat{M}_s = \mathcal{H}(\mathcal{R}(M_s))$ as illustrated in Fig. 5.2. We encourage consistency of the intrinsics using the rendering cycle consistency loss which is defined as the *Smooth*-L_1 distance between the input and reconstructed intrinsics

$$\mathcal{L}_{ren}(\mathcal{R}, \mathcal{H}, M_s) = \|\mathcal{H}(\mathcal{R}(M_s)) - M_s\|_1. \tag{5.1}$$

Decomposition Cycle. Similarly, we train $\mathcal{H}$ to generate intrinsic maps $\hat{M}_r = \mathcal{H}(I_r)$ from real images I_r. To ensure consistency with the input I_r, the output intrinsics $\hat{M}_r$ are passed to the deferred neural renderer $\mathcal{R}$ to reconstruct the image $\hat{I}_r = \mathcal{R}(\mathcal{H}(I_r))$. The decomposition cycle consistency loss is defined by:

$$\mathcal{L}_{dec}(\mathcal{R}, \mathcal{H}, I_r) = \|\mathcal{R}(\mathcal{H}(I_r)) - I_r\|_1. \tag{5.2}$$

The combined cycle consistency loss is then defined as:

$$\mathcal{L}_{cyc}(\mathcal{R}, \mathcal{H}, I_r, M_s) = \mathcal{L}_{ren}(\mathcal{R}, \mathcal{H}, M_s) + \mathcal{L}_{dec}(\mathcal{R}, \mathcal{H}, I_r) \tag{5.3}$$

To ensure that the predicted normals $\hat{N}_s = \mathcal{H}_N(I_r)$ and the reconstructed real normals $\hat{N}_r = \mathcal{H}_N(\mathcal{R}(M_s))$ are properly normalized, we exploit an additional normalization loss $\mathcal{L}_{\text{norm}}$:

$$\mathcal{L}_{\text{norm}}(\mathcal{R}, \mathcal{H}, I) = \mid 1 - \|\mathcal{H}_N(I_r)\|_2 \mid + \mid 1 - \|\mathcal{H}_N(\mathcal{R}(M_s))\|_2 \mid$$

5.3.2 Shared Adversarial Loss

While the cycle consistency loss ensures that the network input can be reconstructed from its output, it does not place any importance on the realism of that output. Additionally, the cycle consistency loss assumes a one-to-one deterministic mapping between the input and output. While this is a reasonable condition for some image-to-image translation tasks [228], it is violated when translating between images and their intrinsic properties. Decomposing an RGB image into its high-dimensional intrinsic properties is a one-to-many transformation since multiple decompositions can be consistent at the same time with the same image, e.g., a gray patch may correspond to a gray diffuse surface or a black glossy surface with specular highlight. Likewise, the process of creating an image from an incomplete set of intrinsic properties involves making additional predictions about missing attributes like lighting conditions, optical aberrations, noise or higher-order light interactions. To better capture this multi-modal relationship, we use an adversarial loss between the input and its reconstruction.

An adversarial discriminator $\mathcal{D}$ is a classification model trained to predict if a data sample is produced by a generative model or if it stems from the true data distribution. To train our Intrinsic Autoencoder, we use two adversarial discriminators, $\mathcal{D}_I$ for discriminating generated images $\hat{I}_{\{s,r\}}$ from real images I_r, and $\mathcal{D}_M$ for discriminating generated intrinsic maps $\hat{M}_{\{r,s\}}$ from synthetic intrinsic maps M_s. The discriminators help our model to learn the distribution of real images and synthetic intrinsics by optimizing the following adversarial [61] loss function

$$\mathcal{L}_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_I, \mathcal{D}_M) = \mathcal{L}^I_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_I) + \mathcal{L}^M_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_M) \tag{5.4}$$

where

$$\begin{aligned}\mathcal{L}^{I}_{\text{adv}}(\mathcal{R},\mathcal{H},\mathcal{D}_I) = \log(\mathcal{D}_I(I_r)) + \log(1-\mathcal{D}_I(\mathcal{R}(M_s)) \\ + \log(1-\mathcal{D}_I(\mathcal{R}(\mathcal{H}(I_r)))\end{aligned} \tag{5.5}$$

is our novel *shared adversarial image loss* which discriminates both between the real image I_r and the generated synthetic image $\hat{I}_s = \mathcal{R}(M_s)$, as well as between the real image I_r and the reconstructed real image $\hat{I}_r = \mathcal{R}(\mathcal{H}(I_r))$. Similarly, we define the *shared adversarial intrinsic loss* as

$$\begin{aligned}\mathcal{L}^{M}_{\text{adv}}(\mathcal{R},\mathcal{H},\mathcal{D}_M) = \log(\mathcal{D}_{\mathcal{M}}(M_s)) + \log(1-\mathcal{D}_M(\mathcal{H}(I_r)) \\ + \log(1-\mathcal{D}_M(\mathcal{H}(\mathcal{R}(M_s))))\end{aligned} \tag{5.6}$$

Using the reconstructed inputs $\hat{I}_r$ and $\hat{M}_s$ in addition to the generated samples $\hat{I}_s$ and $\hat{M}_r$ for training $\mathcal{D}_I$ and $\mathcal{D}_M$ makes the discriminators more robust and prevents overfitting. This is especially important when a relatively small number of 3D objects are used to create the synthetic intrinsic maps which can lead to a discriminator that recognizes the model features rather that the image realism.

5.3.3 Implementation and Training

We train our Intrinsic Autoencoder networks $\mathcal{R}, \mathcal{H}$ in addition to the adversarial discriminators $\mathcal{D}_M, \mathcal{D}_I$ from scratch by optimizing the joint objective

$$\min_{\mathcal{R},\mathcal{H}} \max_{\mathcal{D}_I,\mathcal{D}_M} \mathcal{L}_{\text{cyc}} + \mathcal{L}_{\text{norm}} + \mathcal{L}_{\text{adv}} \tag{5.7}$$

Our framework is implemented in PyTorch [149] and trained using Adam [96] with a learning rate of 0.0002. The Deferred Neural Rendering Network is a coarse-to-fine generator introduced in [206] for the deferred neural rendering network. The input to the network is of size 256×512 constructed by concatenating normals, albedo and reflections. The output of the network is an RGB image of size $256 \times 512 \times 3$. We use three networks $\mathcal{H} = \{\mathcal{H}_N, \mathcal{H}_A, \mathcal{H}_F\}$ for estimating the surface normals N, Albedo A and environment reflections F, respectively, from an image I. Each network has a ResNet architecture with 5 ResNet blocks.

Adversarial Discriminator Networks. Since the local structure of the generated images is mostly controlled by the input intrinsics, we want the image discriminator $\mathcal{D}_I$ to mainly focus on the global realism of the output. To address this, we use a multi-scale PatchGAN [206] discriminator which comprises two fully-convolutional networks that classify the local image patches at two scales, full and half resolution. The discriminator outputs a realism score for each patch instead of a single prediction per image. This has been shown to produce more detailed images for similar conditional image generation tasks [80, 206, 228]. The intrinsics discriminator $\mathcal{D}_M$ has the same architecture except that the input is a 9-channel stack combining all three intrinsic maps. We found that using a single discriminator for the combination of the intrinsic maps performs better than separate networks for each. This is likely due to the inter-dependence between the different intrinsic properties that allows the discriminator to detect inconsistencies between the generated intrinsic maps. We provide more architecture and training details in the supplementary material.

5.4 Experiments

Synthetic Data Generation. To generate the synthetic training data, we use dataset from [2] containing 28 3D car models covering 6 car categories (SUV, sedan, hatchback, station wagon, mini-van and van). Apart from the geometry, we do not need any physically-based materials or textures for the models. Instead, we assign to each car part a simple material with only two properties, the color and a scalar glossiness factor for computing reflection maps. We assign each 3D car part a fixed material from a set of 18 fixed materials. Additionally, we randomly pick one of 15 materials with different colors for the car body during the rendering process. Next, a camera position is randomly chosen within a radius of 8 meters and a maximum height of 3 meters. We use a fast OpenGL based rendering engine which operates at around 3 frames per second including the model loading time. It outputs the surface normals of the car model in the camera coordinate space and the albedo channels indicating the material color at each pixel without any lighting or shading. Finally, we produce the environmental reflections

by using a 360 degree environment map from [2]. These kind of reflections are very efficient to compute since they only require the view vector and the surface normal and do not rely on expensive path-tracing. We render 20,000 synthetic samples of normals, albedo and reflections.

Real training data. We obtain the real images from a fine grained car classification dataset presented in [103]. For convenience, we refer to this as the real car dataset. It contains 16,000 images of cars captured in various lighting conditions, resolutions and poses and with different camera sensors and lenses.

5.4.1 Baselines

Since our goal is to train with only unpaired data, we choose to benchmark our method against two state-of-the-art unpaired image generation approaches, CycleGAN[228] and MUNIT[77]. However, since both methods were originally designed for image-to-image translation rather than deferred rendering, we setup two additional strong baselines that highlight the importance of our contributions in improving the quality of our results.

CycleGAN and MUNIT. CycleGAN[228] is a generic method for translating between two domains without available paired data. MUNIT[77] aims at producing a diverse set of translations between different domains. We modify the two methods slightly to use our stacked 9 channel synthetic intrinsic maps as inputs.

Without shared discriminator. In this setup, we do not use the shared adversarial discriminator discussed in 5.3.2. Instead, we only use the discriminator $\mathcal{D}_I$ between generated image $\hat{I}_s$, real image I_r. Similarly, the discriminator $\mathcal{D}_M$ is used only between synthetic intrinsics M_s and generated intrinsics $\hat{M}_r$.

Only rendering cycle. Here, we train the model using only the deferred rendering cycle discussed in (Sec. 5.3.1) and do not use the decomposition cycle.

Figure 5.3. Images generated using our Deferred Neural Renderer. Inputs to the network are intrinsic maps consisting of albedo, normals and reflections, shown above the generated images.

5.4.2 Deferred Neural Rendering

To evaluate our approach for deferred neural rendering, we use the network $\mathcal{R}$ to produce images given synthetic intrinsic maps (albedo, normals, reflections) and compare it to other baselines, both qualitatively and quantitatively.

Qualitative results

Fig. 5.3 shows car images generated using our deferred neural renderer from the input synthetic intrinsic maps shown above them. The car models in the evaluation set have been previously seen by the generator, but the unique combination of pose and paint color has not been seen during training. Our approach is able to generate detailed photo-realistic images of cars with consistent geometry and distinct parts. We emphasize that the deferred neural rendering network is trained without any rendered or real geometry-image pairs. Instead, it is able to learn the appearance of different car parts from a large set of real car images. For more results see supplementary [1]

In Fig. 5.4 we compare the results of our full model to various baselines. The

[1] `https://youtu.be/FOWoCeOAiug`

Figure 5.4. Qualitative Comparison with baselines on Neural Rendering. Inputs to the network are intrinsic maps consisting of albedo, normals and reflections, shown above the generated images. Additional higher resolution results are provided in the supplementary materials.

results clearly show the improvements in visual quality achieved when using our full model. Specifically, MUNIT appears to be unable to preserve the geometry and albedo of the input in the generated image, CycleGAN images has significant artefacts on the windows, body, etc. When training our model without the shared discriminator, the resulting images suffer from irregular reflection patterns and a noisy image. This is

likely due to the strong overfitting required by the network to reproduce the input image exactly when using only an L_1 loss. The model trained without the decomposition cycle is not able to preserve the input intrinsics in terms of albedo and reflection.

In figure 5.5, we show the effect of input intrinsic maps on the quality of rendered images. When the model is trained only with normals as intrinsic input, the geometry of the result is well rendered but the color of different parts poorly defined. The model trained on both normals and albedo demonstrates sharper image quality but the hallucinated reflections by the network lacks lack realistic details. Finally, using the environmental reflections helps the network produce consistent and realistic images with sharp details.

Figure 5.5. Images generated using models trained with ablated inputs.

Quantitative results

We evaluate the quality of generated images using Fréchet Inception Distance(FID) [70] and Kernel Inception Distance(KID) [13]. Both metrics compute the distance between the features of two sets of images, obtained from a pre-trained CNN. Table 5.1 presents both the FID and KID between the images generated using various methods and the real images. Our full model achieves the lowest FID and KID values (47.6, 4.2) indicating that the rendered images from our model are closest to the distribution of real images compared to MUNIT [77] and CycleGAN [228]. Further, when we ablate each of the intrinsic map inputs, both FID and KID increase substantially. Notably, in the case of

	Cycle GAN	MUNIT	**Ours**	w/ Shared Discr	w/o Decom. Cyc.	w/o A	w/o N	w/o F
FID	103.3	99.0	**47.6**	59.2	99.6	88.7	60.2	56.7
KID	10.2	13.5	**4.2**	4.8	11.8	5.4	4.9	5.9

Table 5.1. FID and KID between real images and generated samples. All inputs are provided to the generator (Albedo, Normals and Reflections).

ablating albedo input, the highest increase in distances can be observed (88.7, 5.4), implying its importance for photo-realistic image generation. We conclude that albedo is the most important for our task followed by normals and reflections maps. In both cases where we ablate the decomposition cycle or rendering cycle, we observe a huge increase in the distances signifying the importance of using both cycle consistency losses during training. Finally, training with the setup of separate discriminators as mentioned in 5.4.1 leads to an increase in the distances.

Human Experiments

We design two experiments to measure the visual realism of generated car images using the Amazon Mechanical Turk platform to crowd source human evaluations. For each comparison, we presented 40 human subjects each with 50 image pairs to choose the more realistic looking image. The results are presented in Table 5.2. The first row presents experiments where one image is picked from the real images and the other is from one of the synthesis methods and presented in a random order. Images from our full model seem to be most confused with real car image since only 67.5% of choices were correct while in 32.5% of the trials the subjects choose our images to be the real one.

In the second experiment subjects are presented with an image generated by our full model and a matching image generated by one other synthesis methods. The results in the second row of Table 5.2 show that subjects choose our results to be more realistic over 80% of times when compare to CycleGAN and MUNIT. This clearly indicates a

	Cycle GAN	MUNIT	**Ours**	w/o Shared Discr.	w/o Decom Cyc.
Real Im	77.7%	75.6%	**67.5%**	68.9%	71.0%
Ours	80.0%	85.8%	–	57.6%	63.8%

Table 5.2. Human Subject Study. Comparisons to identify realistic images in an A/B test using Amazon Mechanical Turk. The numbers indicate the ratio of trials where the image from real or our model was chosen as more realistic compared to the image from the method on the header.

high level of visual quality of our generated images compared to those generated from existing methods. On the other hand, images from our ablated models appear to be much closer to our full model visual quality.

5.4.3 Intrinsic Image Decomposition

Qualitative results

In fig. 5.6, we show that the intrinsic decomposition network is able to decompose real car images into their intrinsic maps. We would like to emphasize that the model does not have access to ground-truth intrinsic maps for real images during the training phase. Also, these car models are not present in the synthetic training data.

Figure 5.7 compares the decompositions produced by our model to those from other baselines. Both CycleGAN [228] and MUNIT [77] show significant artificats and inconsistencies when trained to decompose real images. The USI3D [113] fails to generalize to real models since it was trained using synthetic data from ShapeNet [23]. Our model without decomposition cycle also recovers noisy albedo and normals due to overfit only to synthetic data. On the other hand, training without the shared discriminator leads to severe artefacts. This is because the rendering network tries to encode intrinsics information in the generated images in the form of high frequency artefacts such that the decomposition network can easily recover them.

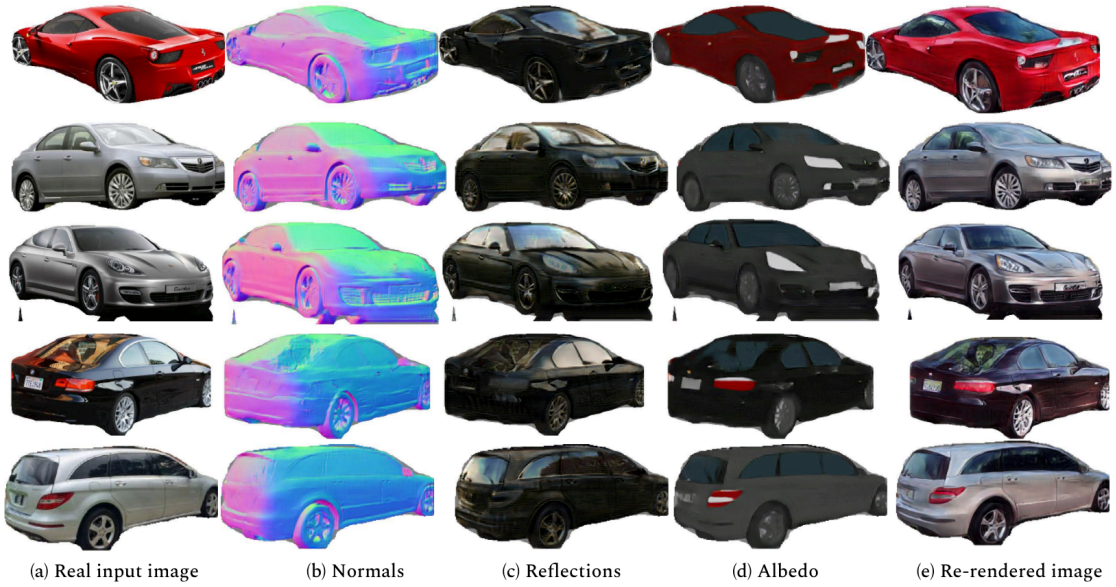


Figure 5.6. Results of our intrinsic decomposition network on real images. The first column shows the inputs to the network. Our model is able to decompose the sport car in first row accurately even though our synthetic training dataset does not include any sport cars at all. The car models of other inputs images are also not present in our synthetic dataset.

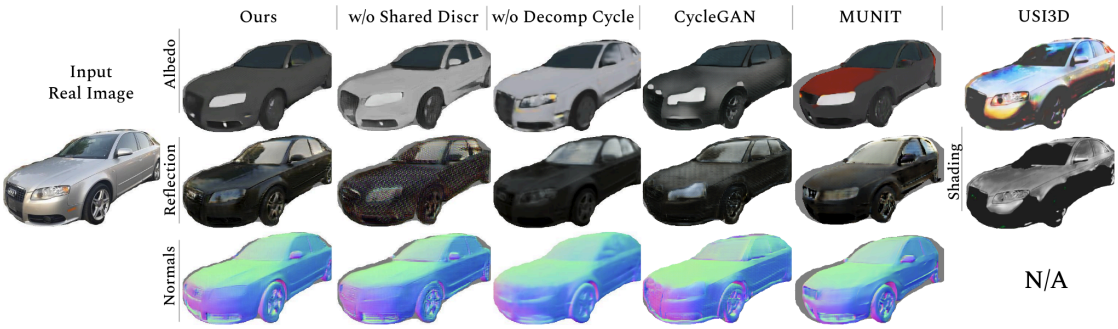


Figure 5.7. Comparison with Baselines for intrinsic decomposition. Note that USI3D [113] only produces albedo and shading and not reflections or normals.

Quantitative results

To evaluate the intrinsic maps predicted by the intrinsic image decomposition network ($\mathcal{H}$) we construct a synthetic dataset containing rendered RGB images and their corresponding intrinsic maps rendered using a standard Physically Based Renderer (Blender

	w/o Shared Discr.	w/o Decom. cycle	Cycle GAN	MUNIT	Ours
Normal Err.	17.75°	18.80°	27.82°	29.15°	**14.73°**
Albedo Err.	54.00	67.21	68.18	81.44	**52.74**
Reflection Err.	55.60	71.00	73.18	74.75	**51.74**

Table 5.3. Errors for the Intrinsic Decomposition Task. Our method achieves the lowest error on all tasks.

[15]). To obtain the error between predicted and ground truth normals, we compute the average cosine distance between them. The errors for albedo and reflection are the average ℓ_1 distances between the predicted and ground truth maps. Table 5.3 presents the errors of various methods for predicting intrinsic maps. Our full model has the least error for all the modalities followed by our model without the shared discriminator, without decomposition cycle and finally MUNIT and CycleGAN. This indicates that our model is able to learn accurate image decomposition while keeping generalization. Note that these PBR-rendered images have not been presented to our network during training.

5.4.4 Results on ShapeNet Airplanes

We train our model for the object class "Airplanes". We obtain the real images from FGVC-Aircraft dataset [122] which contains 10,000 images of airplanes. We use the 3D models of airplanes from the Shapenet dataset [23] to obtain our intrinsic maps. We follow the process mentioned in sec.5.4 to generate input training data. We use the normals and albedo as inputs to the network. Figure 5.8 illustrates realistic images generated using our deferred rendering network, demonstrating the ability of our method to handle low-quality mesh and texture models.

Figure 5.8. Images generated by our network trained on airplanes from ShapeNet [23].

5.5 Conclusion

In this chapter, we presented a joint approach for training a deferred rendering network for generating realistic images from synthetic image intrinsics and an intrinsic image decomposition network for decomposing real images of an object into its intrinsic properties. We trained the model using unpaired 3D models and real images. Our qualitative and quantitative experiments revealed that using a combination of shared adversarial losses and cycle consistency losses is able to produce images that are both realistic and consistent with the control input.

Chapter 6

Discussion

The chapters in this thesis present several novel ideas and paradigms for synthetic image generation with focus on making the generated images more effective in training scene understanding machine learning models. In this final chapter, the main ideas from each chapter are first summarized and then the advantages and limitations of the proposed methods are discussed. Finally, possible directions for future work based on ideas in this thesis are presented.

6.1 Augmented reality for deep learning

Chapter 3 explored the use of mixing real and synthetic objects in the same training image as an alternative to pure synthetic images. This method is especially efficient for augmenting urban driving datasets where capturing and labeling the static background (e.g. streets, buildings and street signs) are relatively easy. Meanwhile, foreground objects (e.g. cars, bicycles and people) are dynamic and hard to capture using 3D sensors like lidar and label. Indeed, those foreground objects are usually of high importance when making driving decisions, which makes them crucial for training autonomous driving systems. The experimental results showed how using mixed images can create more effective training data for deep learning models. Several ablation experiments in this chapter studied the various factors that affect the usefulness of augmented im-

ages. Those point to two main conclusions: First, the training performance of rendered objects is mostly limited by the low-level local image features, like noise and blur, more than by the global data distribution parameters, like car position, direction or shadow. Second, the balance between the number of real and rendered objects in the same augmented image is important for training performance since that helps to avoid network over-fitting on the synthetic objects.

Despite its effectiveness and simplicity, the augmentation framework introduced in Chapter 3 has several limitations. The realism and quality of the generated mixed images rely heavily on the accurate manual tuning of the rendering parameters and post-processing pipeline done by experts. This tuning is done once per dataset but still requires high-level skills. This limitation is mainly addressed by the Geometric Image Synthesis model presented in Chapter 4 which aims to learn the process of tuning the object appearance to fit the background. The proposed method also requires manual annotation of street lanes for realistic virtual car placement. This could be replaced and improved by using a combination of scene understanding techniques, like lane detection and road segmentation, with traffic simulation models and physics engines which can generate realistic and physically accurate traffic scenarios automatically. Constructing a scene structure graph of real images [38] can also be employed to ensure the simulated traffic scenario matches the real background scene. Another limitation of using 2D image information only in this work is that it can not generate synthetic objects partially occluded by real objects due to lack of depth data. A possible future extension could be to use the 3D structure of the scene captured through lidar sensors or light-field systems which can capture the material properties additionally. This can be used to better match the reflections, integrate shadows, and handle occlusions between real and synthetic objects.

6.2 Geometric image synthesis

Chapter 4 proposed replacing a classical rendering engine with a learned model called Geometric Image Synthesis. Using a differentiable neural network for rendering has two advantages compared to traditional rendering engines. First, it allows the object appearance to be partially learned from real images of similar objects. This largely reduces the amount of manual tuning of rendering and post-processing parameters required for producing realistic results. Second, the flexibility and differentiable nature of a neural network allows for using different inputs or multiple loss functions depending on the target task. For example, the goal of experiments in Chapter 4 was to render images that are both realistic and blend well with the real background. This required giving the background image as input to the synthesis network and using an adversarial discriminator that examines rendered objects in context of the real background image to determine if it is real or fake. An interesting direction for developing this approach would be to jointly train the rendering network with the scene understanding task network. This could make the resulting images even more challenging for the task network leading to better training data and a more robust model. One limitation of the proposed network architecture is that it follows the fully convolutional image-to-image network design which requires the input to be defined pixel-wise. This means global rendering parameters like light direction, camera parameters or object pose can not be easily controlled or learned. Instead, GIS relies on a simple traditional non-differentiable rendering engine to convert the object mesh and sampled global parameters into several 2D maps that constitute the input to the network. A possible way to alleviate this limitation would be to replace the traditional non-differentiable rendering engine with a differentiable one. This would allow gradients of the loss function to back-propagate not only into the GIS network but also to the global rendering parameters opening up the use of this method for tasks like post estimation. The other limitation of the proposed GIS network is that it uses a traditional rendering engine during training to produce matching RGB images for the input. This is used as a form of regularization to ensure that the image produced by GIS is not only realistic but also matching the input structure. A more desirable setup would be where the image synthesis network is trained solely based on real images which is

explored in Chapter 5.

6.3 Joint learning of image synthesis and decomposition

Chapter 5 introduced Intrinsic Autoencoders, a novel model for training a neural rendering network without any paired real or rendered images. This was achieved by combining the two complimentary tasks of neural rendering and intrinsic image decomposition into a single model that can be trained with unpaired sets of 3D models and real images using a combination of adversarial discriminators and cycle consistency. One of the main challenge in this setup is that the adversarial discriminator loss should depend only on the appearance of the generated image and not on its content. This is difficult in practice since it requires the 3D models and images to be coming from the exact same data distribution (e.g. the same set of car models). The Intrinsic Autoencoder model introduces the shared adversarial discriminator as a mean to solve this problem by comparing real images not only to rendered 3D models but also to re-rendered real images after decomposing them into their intrinsic maps. The real and re-rendered images match in their content and only differ in appearance creating thus forcing discriminator to focus on the latter. Another advantage of this approach is that it can be learned even using a small number of 3D models (e.g. 28 car models used in experiments in Chapter 5) making it more applicable in practice.

Similar to GIS, the Intrinsic Autoencoders model relies on basic non-differentiable rendering methods to generate input maps like normals and reflections. Replacing this step with a differentiable rendering model could allow the use of the neural rendering and intrinsic image decomposition networks as intermediate modules for other tasks. For example, the intrinsic decomposition model combined with differentiable rendering could be seen as single-image 3D shape reconstruction method making it useful in many new applications. Another future direction would be to replace the two networks solving the inverse tasks of rendering and decomposition with a single flow-based invertible neural network [209] making the model much simpler and the training more straightforward.

Bibliography

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

[2] H. A. Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother, "Augmented reality meets computer vision: Efficient data generation for urban driving scenes," *International Journal of Computer Vision (IJCV)*, 2018.

[3] H. A. Alhaija, S. K. Mustikovela, L. M. Mescheder, A. Geiger, and C. Rother, "Augmented reality meets deep learning," in *British Machine Vision Conference (BMVC)*, 2017.

[4] H. A. Alhaija, S. K. Mustikovela, J. Thies, V. Jampani, M. Nießner, A. Geiger, and C. Rother, "Intrinsic autoencoders for joint deferred neural rendering and intrinsic image decomposition," in *International Conference on 3D Vision (3DV)*, 2020.

[5] J. L. Alireza Shafaei and M. Schmidt, "Play and learn: Using video games to train computer vision models," in *British Machine Vision Conference (BMVC)*, E. R. H. Richard C. Wilson and W. A. P. Smith, Eds. BMVA Press, Sep. 2016.

[6] A. Andreopoulos and J. K. Tsotsos, "On sensor bias in experimental methods for comparing interest-point, saliency, and recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, 2011.

[7] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour Detection and Hierarchical Image Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, May 2011.

[8] M. Arjovsky and L. Bottou, "Towards Principled Methods for Training Generative Adversarial Networks," *arXiv:1701.04862 [cs, stat]*, Jan. 2017.

[9] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv: 1701.07875 [stat.ML]*, 2017.

[10] J. L. Barron, D. J. Fleet, S. S. Beauchemin, and T. A. Burkitt, "Performance Of Optical Flow Techniques," *International Journal of Computer Vision (IJCV)*, vol. 12, no. 1, 1994.

[11] J. T. Barron and J. Malik, "Shape, illumination, and reflectance from shading," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 8, 2015.

[12] A. Behl, O. Jafari, S. K. Mustikovela, H. A. Alhaija, C. Rother, and A. Geiger, "Bounding boxes, segmentations and object coordinates: How important is recognition for 3D scene flow estimation in autonomous driving scenarios?" in *International Conference on Computer Vision (ICCV)*, 2017.

[13] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton, "Demystifying MMD GANs," in *International Conference on Learning Representations (ICLR)*, 2018.

[14] V. Blanz, T. Vetter *et al.*, "A morphable model for the synthesis of 3D faces." in *Siggraph*, vol. 99, 1999.

[15] Blender Online Community, *Blender - a 3D Modelling and Rendering Package*. Blender Institute, Amsterdam: Blender Foundation, 2006.

[16] A. Borji, "Pros and cons of GAN evaluation measures," *Computer Vision and Image Understanding*, vol. 179, 2019.

[17] J. Borrego, A. Dehban, R. Figueiredo, P. Moreno, A. Bernardino, and J. Santos-Victor, "Applying domain randomization to synthetic data for object category detection," *arXiv:1807.09834*, 2018.

[18] 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.

[19] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic Object Classes in Video: A High-definition Ground Truth Database," *PRL*, vol. 30, no. 2, Jan. 2009.

[20] 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.

[21] Y. Cabon, N. Murray, and M. Humenberger, "Virtual KITTI 2," *arXiv:2001.10773 [cs, eess]*, Jan. 2020.

[22] A. Carlson, K. A. Skinner, R. Vasudevan, and M. Johnson-Roberson, "Modeling Camera Effects to Improve Visual Learning from Synthetic Data," in *European Conference on Computer Vision (ECCV) Workshop*, S. Roth and L. Leal-Taixé, Eds. Cham: Springer International Publishing, 2019, vol. 11129.

[23] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q.-X. 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," *arXiv*, vol. 1512.03012, 2015.

[24] O. Chapelle, B. Scholkopf, and A. Zien, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, 2009.

[25] C. Che, F. Luan, S. Zhao, K. Bala, and I. Gkioulekas, "Inverse transport networks," *arXiv:1809.10820*, 2018.

[26] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," in *European Conference on Computer Vision (ECCV)*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., vol. 11211. Cham: Springer International Publishing, 2018.

[27] Q. Chen and V. Koltun, "Photographic Image Synthesis with Cascaded Refinement Networks," in *International Conference on Computer Vision (ICCV)*. Venice: IEEE, Oct. 2017.

[28] W. Chen, H. Wang, Y. Li, H. Su, Z. Wang, C. Tu, D. Lischinski, D. Cohen-Or, and B. Chen, "Synthesizing Training Images for Boosting Human 3D Pose Estimation," in *International Conference on 3D Vision (3DV)*, 2016.

[29] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool, "Domain Adaptive Faster R-CNN for Object Detection in the Wild," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT, USA: IEEE, Jun. 2018.

[30] E. Cheung, T. K. Wong, A. Bera, and D. Manocha, "STD-PD: Generating synthetic training data for pedestrian detection in unannotated videos," *arXiv:1707.09100*, 2017.

[31] R. Child, "Very deep VAEs generalize autoregressive models and can outperform them on images," *arXiv:2011.10650*, 2020.

[32] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[33] F. Chollet, "Xception Deep Learning with Depthwise Separable Convolutions," *arXiv:1610.02357 [cs]*, Apr. 2017.

[34] 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 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[35] J. Dai, K. He, and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[36] C. R. de Souza, A. Gaidon, Y. Cabon, and A. M. L. Peña, "Procedural generation of videos to train deep action recognition networks," *arxiv*, vol. 1612.00881, 2016.

[37] E. L. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks," in *NIPS*, 2015.

[38] J. Devaranjan, A. Kar, and S. Fidler, "Meta-sim2: Unsupervised learning of scene structure for synthetic data generation," in *ECCV*, 2020.

[39] S. Diamond, V. Sitzmann, S. Boyd, G. Wetzstein, and F. Heide, "Dirty Pixels: Optimizing Image Classification Architectures for Raw Sensor Data," *arXiv:1701.06487 [cs]*, Jan. 2017.

[40] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," in *International Conference on Learning Representations (ICLR)*, 2017.

[41] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, 2016.

[42] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *International Conference on Computer Vision (ICCV)*, 2015.

[43] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017.

[44] A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox, "Learning to generate chairs, tables and cars with convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, 2017.

[45] A. Dosovitskiy, J. Tobias Springenberg, and T. Brox, "Learning to generate chairs with convolutional neural networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[46] A. Dundar, M. Y. Liu, Z. Yu, T. C. Wang, J. Zedlewski, and J. Kautz, "Domain stylization: A fast covariance matching framework towards domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[47] D. Dwibedi, I. Misra, and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in *International Conference on Computer Vision (ICCV)*, 2017.

[48] S. M. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, D. P. Reichert, L. Buesing, T. Weber, O. Vinyals, D. Rosenbaum, N. Rabinowitz, H. King, C. Hillier, M. Botvinick, D. Wierstra, K. Kavukcuoglu, and D. Hassabis, "Neural scene representation and rendering," *Science*, vol. 360, no. 6394, Jun. 2018.

[49] M. Fonder and M. V. Droogenbroeck, "Mid-Air: A multi-modal dataset for extremely low altitude drone flights," in *Conference on Computer Vision and Pattern Recognition (CVPR) Workshop*, Jun. 2019.

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

[51] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International Conference on Machine Learning (ICML)*, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 2015.

[52] C. Gao, Y. Shih, W.-S. Lai, C.-K. Liang, and J.-B. Huang, "Portrait neural radiance fields from a single image," *arXiv: 2012.05903 [cs.CV]*, 2020.

[53] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image Style Transfer Using Convolutional Neural Networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016.

[54] L. Gatys, A. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style," *Journal of Vision*, vol. 16, no. 12, Aug. 2016.

[55] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, 2013.

[56] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[57] J. J. Gibson, *The Perception of the Visual World.* Houghton Mifflin, 1950.

[58] I. Gkioulekas, A. Levin, and T. Zickler, "An evaluation of computational imaging techniques for heterogeneous inverse scattering," in *European Conference on Computer Vision (ECCV)*, 2016.

[59] I. Gkioulekas, S. Zhao, K. Bala, T. Zickler, and A. Levin, "Inverse volume rendering with material dictionaries," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 6, 2013.

[60] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*. MIT press Cambridge, 2016, vol. 1.

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

[62] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate Shift by Kernel Mean Matching," in *Dataset Shift in Machine Learning*, J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, Eds. The MIT Press, Dec. 2008.

[63] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein GANs," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[64] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic Data for Text Localisation in Natural Images," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[65] A. Guzman-Rivera, D. Batra, and P. Kohli, "Multiple choice learning: Learning to produce multiple structured outputs." in *NIPS*, vol. 1, 2012.

[66] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla, "Understanding Real World Indoor Scenes With Synthetic Data," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

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

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

[69] D. J. Heeger, "Model for the extraction of image flow," *JOSA A*, vol. 4, no. 8, 1987.

[70] 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 *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[71] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. R. Bradski, K. Konolige, and N. Navab, "Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes," in *ACCV*, 2012.

[72] G. E. Hinton, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, Jul. 2006.

[73] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv*, vol. abs/1503.02531, 2015.

[74] 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.

[75] S. Hong, D. Yang, J. Choi, and H. Lee, "Inferring semantic layout for hierarchical text-to-image synthesis," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[76] B. Horn and M. J. Brooks, Eds., *Shape from Shading*, ser. Artificial Intelligence. Cambridge, Mass: MIT Press, 1989.

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

[78] Y. Huang, Y. Cheng, A. Bapna, O. Firat, M. X. Chen, D. Chen, H. Lee, J. Ngiam, Q. V. Le, Y. Wu, and Z. Chen, "GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism," *arXiv:1811.06965 [cs]*, Jul. 2019.

[79] L. Hui, X. Li, J. Chen, H. He, and J. Yang, "Unsupervised multi-domain image translation with domain-specific encoders/decoders," in *2018 24th International Conference on Pattern Recognition (ICPR)*, 2018.

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

[81] P. T. Jackson, A. A. Abarghouei, S. Bonner, T. P. Breckon, and B. Obara, "Style augmentation: Data augmentation via style randomization." in *Conference on Computer Vision and Pattern Recognition (CVPR) Workshop*, 2019.

[82] W. Jakob, *Mitsuba Renderer*, 2010.

[83] M. Janner, J. Wu, T. Kulkarni, I. Yildirim, and J. B. Tenenbaum, "Self-supervised intrinsic image decomposition," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[84] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "Neural Style Transfer: A Review," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 11, Nov. 2020.

[85] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision (ECCV)*, 2016.

[86] 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 *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[87] N. Kalra and S. M. Paddock, *Driving to Safety: How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability?* Santa Monica, CA: RAND Corporation, 2016.

[88] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling Laws for Neural Language Models," *arXiv:2001.08361 [cs, stat]*, Jan. 2020.

[89] 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.

[90] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training generative adversarial networks with limited data," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

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

[92] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[93] H. Kato, Y. Ushiku, and T. Harada, "Neural 3d mesh renderer," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[94] P. J. Kellman and M. E. Arterberry, "Infant Visual Perception," in *Handbook of Child Psychology*, W. Damon and R. M. Lerner, Eds. Hoboken, NJ, USA: John Wiley & Sons, Inc., Jun. 2007.

[95] J. Kim, M. Kim, H. Kang, and K. Lee, "U-GAT-IT: Unsupervised Generative Attentional Networks with Adaptive Layer-Instance Normalization for Image-to-Image Translation," *arXiv:1907.10830 [cs, eess]*, Jul. 2019.

[96] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[97] D. P. Kingma and P. Dhariwal, "Glow: Generative Flow with Invertible 1x1 Convolutions," *arXiv:1807.03039 [cs, stat]*, Jul. 2018.

[98] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv:1312.6114*, 2013.

[99] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollar, "Panoptic Segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[100] M. Klischat and M. Althoff, "Generating Critical Test Scenarios for Automated Vehicles with Evolutionary Algorithms," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. Paris, France: IEEE, Jun. 2019.

[101] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-THOR: An interactive 3D environment for visual AI," *arXiv*, 2017.

[102] R. Koster, "The cost of games," https://www.raphkoster.com/2018/01/17/the-cost-of-games/, Jan. 2018.

[103] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[104] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *NIPS*, 2012.

[105] J. Kronander, F. Banterle, A. Gardner, E. Miandji, and J. Unger, "Photorealistic Rendering of Mixed Reality Scenes," *Computer Graphics Forum*, vol. 34, no. 2, May 2015.

[106] C. Lassner, G. Pons-Moll, and P. V. Gehler, "A generative model for people in clothing," in *International Conference on Computer Vision (ICCV)*, 2017.

[107] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. K. Singh, and M.-H. Yang, "Diverse Image-to-Image Translation via Disentangled Representations," *arXiv:1808.00948 [cs]*, Aug. 2018.

[108] T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen, "Differentiable monte carlo ray tracing through edge sampling," in *SIGGRAPH Asia 2018 Technical Papers*, 2018.

[109] Y. Li, M.-Y. Liu, X. Li, M.-H. Yang, and J. Kautz, "A closed-form solution to photorealistic image stylization," in *European Conference on Computer Vision (ECCV)*, 2018.

[110] Z. Li and N. Snavely, "Learning intrinsic image decomposition from watching the world," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[111] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised Image-to-Image Translation Networks," *arXiv:1703.00848 [cs]*, Mar. 2017.

[112] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz, "Few-Shot Unsupervised Image-to-Image Translation," *arXiv:1905.01723 [cs, stat]*, May 2019.

[113] Y. Liu, S. You, Y. Li, and F. Lu, "Unsupervised learning for intrinsic image decomposition from a single image," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[114] Z. Liu, T. Lian, J. Farrell, and B. Wandell, "Soft prototyping camera designs for car detection based on a convolutional neural network," in *International Conference on Computer Vision (ICCV) Workshop*, 2019.

[115] J. Locke, *An Essay Concerning Human Understanding*. Kay & Troutman, 1847.

[116] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International Conference on Machine Learning (ICML)*, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 2015.

[117] M. Long, Z. CAO, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Conference on Neural Information Processing Systems (NeurIPS)*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[118] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A Skinned Multi-Person Linear Model," *SIGGRAPH*, 2015.

[119] M. M. Loper and M. J. Black, "OpenDR: An approximate differentiable renderer," in *European Conference on Computer Vision (ECCV)*, 2014.

[120] A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: From simulation to reality with domain randomization," *IEEE Transactions on Robotics*, vol. 36, no. 1, 2019.

[121] W.-C. Ma, H. Chu, B. Zhou, R. Urtasun, and A. Torralba, "Single image intrinsic decomposition without a single intrinsic image," in *European Conference on Computer Vision (ECCV)*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., 2018.

[122] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," Tech. Rep., 2013.

[123] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra,

"Habitat: A Platform for Embodied AI Research," in *International Conference on Computer Vision (ICCV)*, 2019.

[124] D. Marr, "Vision: A computational investigation into the human representation and processing of visual information, henry holt and co," *Inc., New York, NY*, vol. 2, no. 4.2, 1982.

[125] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "NeRF in the wild: Neural radiance fields for unconstrained photo collections," *arXiv: 2008.02268 [cs.CV]*, 2021.

[126] N. Mayer, E. Ilg, P. Haeusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[127] 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?" *International Journal of Computer Vision (IJCV)*, vol. 126, no. 9, Sep. 2018.

[128] 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 *International Conference on Computer Vision (ICCV)*, 2017.

[129] S. Meister and D. Kondermann, "Real versus realistically rendered scenes for optical flow evaluation," in *2011 14th ITG Conference on Electronic Media Technology*, 2011.

[130] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, "Unsupervised Attention-guided Image to Image Translation," *arXiv:1806.02311 [cs]*, Jun. 2018.

[131] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[132] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for GANs do actually converge?" in *International Conference on Machine Learning (ICML)*, 2018.

[133] M. Meshry, D. B. Goldman, S. Khamis, H. Hoppe, R. Pandey, N. Snavely, and R. Martin-Brualla, "Neural rerendering in the wild," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[134] F. Michel, A. Kirillov, E. Brachmann, A. Krull, S. Gumhold, B. Savchynskyy, and C. Rother, "Global hypothesis generation for 6D object pose estimation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[135] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *European Conference on Computer Vision (ECCV)*, 2020.

[136] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," *arXiv:1411.1784 [cs, stat]*, Nov. 2014.

[137] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral Normalization for Generative Adversarial Networks," in *International Conference on Learning Representations (ICLR)*, Feb. 2018.

[138] T. Miyato and M. Koyama, "cGANs with projection discriminator," *arXiv:1802.05637*, 2018.

[139] A. Mordvintsev, C. Olah, and M. Tyka, "Inceptionism: Going deeper into neural networks," *arXiv*, 2015.

[140] F. J. Moreno-Barea, F. Strazzera, J. M. Jerez, D. Urda, and L. Franco, "Forward Noise Adjustment Scheme for Data Augmentation," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, Nov. 2018.

[141] Y. Movshovitz-Attias, T. Kanade, and Y. Sheikh, "How Useful Is Photo-Realistic Rendering for Visual Learning?" in *European Conference on Computer Vision (ECCV) Workshop*, 2016.

[142] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *International Conference on Computer Vision (ICCV)*, 2017.

[143] T. Nguyen-Phuoc, C. Li, S. Balaban, and Y.-L. Yang, "RenderNet: A deep convolutional network for differentiable rendering from 3D shapes," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[144] T. Nguyen-Phuoc, C. Li, L. Theis, C. Richardt, and Y.-L. Yang, "HoloGAN: Unsupervised learning of 3D representations from natural images," in *International Conference on Computer Vision (ICCV)*, Nov. 2019.

[145] T. Nguyen-Phuoc, C. Richardt, L. Mai, Y.-L. Yang, and N. Mitra, "BlockGAN: Learning 3D object-aware scene representations from unlabelled images," in *Conference on Neural Information Processing Systems (NeurIPS)*, Nov. 2020.

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

[147] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing Flows for Probabilistic Modeling and Inference," *arXiv:1912.02762 [cs, stat]*, Dec. 2019.

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

[149] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

[150] X. Peng, B. Sun, K. Ali, and K. Saenko, "Learning Deep Object Detectors from 3D Models," in *International Conference on Computer Vision (ICCV)*, 2015.

[151] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko, "Visda: The visual domain adaptation challenge," *arXiv:1710.06924*, 2017.

[152] X. Peng, B. Usman, K. Saito, N. Kaushik, J. Hoffman, and K. Saenko, "Syn2Real: A new benchmark forSynthetic-to-Real visual domain adaptation," *CoRR*, vol. abs/1806.09755, 2018.

[153] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-Real Transfer of Robotic Control with Dynamics Randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD: IEEE, May 2018.

[154] M. Pharr, W. Jakob, and G. Humphreys, *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann, 2016.

[155] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

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

[157] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, "Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI: IEEE, Jul. 2017.

[158] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-nerf: Neural radiance fields for dynamic scenes," *arXiv:2011.13961*, 2020.

[159] A. Pumarola, J. Sanchez, G. Choi, A. Sanfeliu, and F. Moreno-Noguer, "3DPeople: Modeling the geometry of dressed humans," in *International Conference on Computer Vision (ICCV)*, 2019.

[160] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv:1511.06434*, 2015.

[161] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv:1605.05396*, 2016.

[162] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *NIPS*, 2015.

[163] S. R. Richter, H. A. Alhaija, and V. Koltun, "Enhancing photorealism enhancement," *arXiv:2105.04619*, 2021.

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

[165] 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.

[166] G. Riegler, A. O. Ulusoy, and A. Geiger, "OctNet: Learning Deep 3D Representations at High Resolutions," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[167] 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 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[168] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann, "Stabilizing Training of Generative Adversarial Networks through Regularization," *arXiv:1705.09367 [cs, stat]*, Nov. 2017.

[169] A. Rozantsev, V. Lepetit, and P. Fua, "On rendering synthetic images for training an object detector," *CVIU*, vol. 137, 2015.

[170] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, 2015.

[171] K. Saad and S.-A. Schneider, "Camera vignetting model and its effects on deep neural networks for object detection," in *International Conference on Connected Vehicles and Expo (ICCVE)*, 2019.

[172] F. Sadeghi and S. Levine, "CAD2RL: Real single-image flight without a single real image," in *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, Jul. 2017.

[173] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs," in *NIPS*, 2016.

[174] B. Sassen, "Kant on Molyneux's Problem," *British Journal for the History of Philosophy*, vol. 12, no. 3, Aug. 2004.

[175] S. Sengupta, J. Gu, K. Kim, G. Liu, D. Jacobs, and J. Kautz, "Neural Inverse Rendering of an Indoor Scene From a Single Image," in *International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019.

[176] S. Sengupta, A. Kanazawa, C. D. Castillo, and D. W. Jacobs, "SfSNet: Learning shape, refectance and illuminance of faces in the wild," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[177] A. Shafaei, J. J. Little, and M. Schmidt, "Play and Learn: Using Video Games to Train Computer Vision Models," *arXiv*, vol. 1608.01745, 2016.

[178] 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.

[179] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer," *arXiv:1701.06538 [cs, stat]*, Jan. 2017.

[180] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from Simulated and Unsupervised Images through Adversarial Training," *arXiv:1612.07828 [cs]*, Jul. 2017.

[181] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor Segmentation and Support Inference from RGB-D Images," in *European Conference on Computer Vision (ECCV)*, 2012.

[182] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *ICLR*, 2015.

[183] B. Sinclair, "GTA V dev costs," https://www.gamesindustry.biz/articles/2013-02-01-gta-v-dev-costs-over-USD137-million-says-analyst, 2013.

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

[185] M. Stark, M. Goesele, and B. Schiele, "Back to the Future: Learning Shape Models from 3D CAD Data," in *BMVC*, 2010.

[186] 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 *International Conference on Computer Vision (ICCV)*, 2015.

[187] M. N. Subhani and M. Ali, "Learning from Scale-Invariant Examples for Domain Adaptation in Semantic Segmentation," in *European Conference on Computer Vision (ECCV)*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, vol. 12367.

[188] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," *arXiv:1707.02968 [cs]*, Aug. 2017.

[189] P. Swoboda, C. Rother, H. A. Alhaija, D. Kainmüller, and B. Savchynskyy, "A study of lagrangean decompositions and dual ascent solvers for graph matching," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[190] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised Cross-Domain Image Generation," *arXiv:1611.02200 [cs]*, Nov. 2016.

[191] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[192] M. Teichmann, M. Weber, J. M. Zöllner, R. Cipolla, and R. Urtasun, "MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving," *arXiv*, vol. 1612.07695, 2016.

[193] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner *et al.*, "State of the Art on Neural Rendering," *arXiv preprint arXiv:2004.03805*, 2020.

[194] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World," *arXiv:1703.06907 [cs]*, Mar. 2017.

[195] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Conference on Computer Vision and Pattern Recognition (CVPR) Workshop*, 2018.

[196] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Lassner, and C. Theobalt, "Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video," *arXiv: 2012.12247 [cs.CV]*, 2020.

[197] I. Triguero, S. García, and F. Herrera, "Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study," *Knowledge and Information systems*, vol. 42, no. 2, 2015.

[198] A. Tsirikoglou, G. Eilertsen, and J. Unger, "A Survey of Image Synthesis Methods for Visual Machine Learning," *Computer Graphics Forum*, vol. 39, no. 6, 2020.

[199] A. Tsirikoglou, J. Kronander, M. Wrenninge, and J. Unger, "Procedural modeling and physically based rendering for synthetic data generation in automotive applications," *arXiv:1710.06270*, 2017.

[200] I. Tufayl, *Ibn Tufayl's Hayy Ibn Yaqzan: A Philosophical Tale*, 1201.

[201] A. Vahdat and J. Kautz, "NVAE: A deep hierarchical variational autoencoder," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[202] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Berlin, Heidelberg: Springer-Verlag, 1995.

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

[204] V. S. R. Veeravasarapu, C. A. Rothkopf, and V. Ramesh, "Model-driven Simulations for Deep Convolutional Neural Networks," *arXiv*, vol. 1605.09582, 2016.

[205] H. Wang, T. Shen, W. Zhang, L.-Y. Duan, and T. Mei, "Classes Matter: A Fine-Grained Adversarial Approach to Cross-Domain Semantic Segmentation," in *European Conference on Computer Vision (ECCV)*. Cham: Springer International Publishing, 2020, vol. 12359.

[206] 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 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[207] W. Wang, H. Li, Z. Ding, and Z. Wang, "Rethink Maximum Mean Discrepancy for Domain Adaptation," *arXiv:2007.00689 [cs, stat]*, Jul. 2020.

[208] X. Wang and A. Gupta, "Generative image modeling using style and structure adversarial networks," in *European Conference on Computer Vision (ECCV)*, 2016.

[209] L. Weng, "Flow-based deep generative models," *lilianweng.github.io/lil-log*, 2018.

[210] R. Woodham, "Photometric method for determining surface orientation from multiple images," *Optical Engineering*, vol. 19, Jan. 1992.

[211] M. Wrenninge and J. Unger, "Synscapes: A photorealistic synthetic dataset for street scene parsing," *arXiv:1810.08705*, 2018.

[212] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: Real-world perception for embodied agents," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[213] Z. Xiao, K. Kreis, J. Kautz, and A. Vahdat, "VAEBM: A symbiosis between variational autoencoders and energy-based models," in *International Conference on Learning Representations (ICLR)*, 2021.

[214] J. Xie, M. Kiefel, M.-T. Sun, and A. Geiger, "Semantic Instance Annotation of Street Scenes by 3D to 2D Label Transfer," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[215] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[216] W. Xu, Y. Li, and C. Lu, "Generating instance segmentation annotation by geometry-guided gan," *arXiv:1801.08839*, 2018.

[217] Z. Yang, H. Liu, and D. Cai, "On the diversity of realistic image synthesis," *arXiv:1712.07329*, 2017.

[218] Z. Yi, H. Zhang, P. Tan, and M. Gong, "DualGAN: Unsupervised dual learning for image-to-image translation," *CoRR*, vol. abs/1704.02510, 2017.

[219] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelNeRF: Neural radiance fields from one or few images," *arXiv: 2012.02190 [cs.CV]*, 2020.

[220] X. Yue, Y. Zhang, S. Zhao, A. Sangiovanni-Vincentelli, K. Keutzer, and B. Gong, "Domain Randomization and Pyramid Consistency: Simulation-to-Real Generalization Without Accessing Target Domain Data," in *International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019.

[221] O. Zendel, "Robust Vision Challenge 2018," in *European Conference on Computer Vision (ECCV) Workshop*, 2018.

[222] ——, "Robust Vision Challenge 2020," in *Conference on Computer Vision and Pattern Recognition (CVPR) Workshop*, 2020.

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

[224] Q. Zhang, J. Zhang, W. Liu, and D. Tao, "Category anchor-guided unsupervised domain adaptation for semantic segmentation," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[225] Y. Zhang, W. Qiu, Q. Chen, X. Hu, and A. L. Yuille, "UnrealStereo: A Synthetic Dataset for Analyzing Stereo Vision," *arXiv*, vol. 1612.04647, 2016.

[226] Y. Zhang, S. Song, E. Yumer, M. Savva, J.-Y. Lee, H. Jin, and T. Funkhouser, "Physically-based rendering for indoor scene understanding using convolutional neural networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[227] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random Erasing Data Augmentation," *arXiv:1708.04896 [cs]*, Nov. 2017.

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

[229] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning," *arXiv*, vol. 1609.05143, 2016.

[230] Y. Zou, Z. Yu, B. V. K. Vijaya Kumar, and J. Wang, "Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-training," in *Eu-*

ropean Conference on Computer Vision (ECCV). Cham: Springer International Publishing, 2018, vol. 11207.