RÉPUBLIQUE FRANÇAISE
Liberté Égalité Fraternité

Université de Toulouse

Doctorat de l'Université de Toulouse

préparé à l'ISAE-SUPAERO

Adaptation de domaine synthétique vers réel par génération d'image appliquée à la reconnaissance d'objets robustes

---

Thèse présentée et soutenue, le 10 juin 2026 par

Estelle CHIGOT

École doctorale
EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse

Spécialité
Informatique et Télécommunications

Unité de recherche
ISAE-ONERA MOIS MOdélisation et Ingénierie des Systèmes

Thèse dirigée par
Thomas OBERLIN et Dennis WILSON

Composition du jury
Mme Claire PAGETTI, Présidente, ONERA
M. Cédric DEMONCEAUX, Rapporteur, Université Bourgogne Europe
M. Stéphane LATHUILIÈRE, Rapporteur, INRIA
M. Julien RABIN, Examinateur, ENSICAEN
M. Thomas OBERLIN, Directeur de thèse, ISAE-SUPAERO
M. Dennis WILSON, Co-directeur de thèse, ISAE-SUPAERO

Membres invités
Mme Manon HUGUENIN, Co-encadrante de thèse, Airbus Operation SAS

Synthetic-to-Real Image-Based Domain Adaptation for Robust Object Recognition

Abstract: Deep learning models for object recognition require large annotated datasets, which are costly and difficult to obtain for industrial applications. Synthetic data generated by simulators offers a scalable alternative with perfect annotations, but models trained on synthetic images perform poorly on real data due to the domain gap. This thesis investigates image-based domain adaptation methods to bridge this synthetic-to-real gap, with a focus on semantic segmentation under adverse weather conditions.

We first propose two diffusion-based style transfer methods, CACTI and CACTI$_{\text{F}}$, that leverage semantic segmentation masks to transfer the visual style of real target images onto synthetic content while preserving structural coherence. Class-wise Adaptive Instance Normalization aligns feature statistics per semantic class, and selective attention filtering suppresses artifacts from weak cross-image correspondences. Experiments on the GTA5 to Cityscapes and ACDC benchmark show that these methods improve segmentation performance across difficult weather conditions.

We then formulate the selection and ordering of style transfer operators as a multi-objective optimization problem, MOOSS, solved with evolutionary algorithms. Using paired-image metrics for efficient evaluation, the method explores the trade-off between content preservation and style similarity. While the optimized pipelines produce visually diverse results, they reveal that current perceptual metrics do not reliably predict downstream task performance.

Finally, we apply these approaches to runway detection for automatic landing, an industrial use case where real data is scarce and domain shift is significant. Experiments confirm that style transfer improves robustness to adverse conditions, while also exposing the limitations of current benchmarks dominated by urban driving data.

Adaptation de domaine synthétique vers réel par génération d'image appliquée à la reconnaissance d'objets robustes

Résumé : Les modèles d'apprentissage profond pour la reconnaissance d'objets nécessitent de vastes ensembles de données annotées, coûteux et difficiles à obtenir pour les applications industrielles. Les données synthétiques générées par des simulateurs offrent une alternative avec des annotations parfaites, mais les modèles entraînés sur ces images sont peu performants sur des données réelles en raison du fossé entre domaines. Cette thèse étudie des méthodes d'adaptation de domaine basées sur l'image pour combler cet écart, avec un accent sur la segmentation sémantique en conditions météorologiques dégradées.

Nous proposons d'abord deux méthodes de transfert de style basées sur la diffusion, CACTI et CACTI$_{\mathrm{F}}$, qui exploitent les masques de segmentation sémantique pour transférer le style visuel d'images réelles cibles sur un contenu synthétique tout en préservant la cohérence structurelle. La normalisation adaptative par classe aligne les statistiques des caractéristiques par classe sémantique, et le filtrage sélectif de l'attention supprime les artefacts dus aux correspondances inter-images faibles. Les expériences sur le benchmark GTA5 vers Cityscapes et ACDC montrent que ces méthodes améliorent les performances de segmentation en conditions météorologiques difficiles.

Nous formulons ensuite la sélection et l'ordonnancement des opérateurs de transfert de style comme un problème d'optimisation multi-objectif, MOOSS, résolu par des algorithmes évolutionnaires. En utilisant des métriques par paires d'images pour une évaluation efficace, la méthode explore le compromis entre préservation du contenu et similarité de style. Si les pipelines optimisées produisent des résultats visuellement diversifiés, elles révèlent que les métriques perceptuelles actuelles ne prédisent pas de manière fiable les performances sur la tâche finale.

Enfin, nous appliquons ces approches à la détection de piste d'atterrissage pour l'atterrissage automatique, un cas d'usage industriel où les données réelles sont rares et le fossé entre domaines est important. Les expériences confirment que le transfert de style améliore la robustesse en conditions dégradées, tout en exposant les limites des benchmarks actuels dominés par les données de conduite urbaine.

Contents

1 Introduction **9**
- 1.1 Motivation for synthetic data . . . 10
  - 1.1.1 Uses of vision models . . . 10
  - 1.1.2 Synthetic data . . . 12
  - 1.1.3 General problem statement . . . 13
- 1.2 Strategies for synthetic data use . . . 13
  - 1.2.1 Domain adaptation . . . 13
  - 1.2.2 Input-level domain adaptation . . . 14
- 1.3 Industrial use cases . . . 14
  - 1.3.1 Autonomous driving . . . 14
  - 1.3.2 Runway detection for Autoland . . . 15
  - 1.3.3 Adverse conditions . . . 16
- 1.4 Thesis overview . . . 16
  - 1.4.1 Specific problem statement . . . 16
  - 1.4.2 Outline . . . 16
  - 1.4.3 List of publications . . . 17

2 Background **19**
- 2.1 Object recognition . . . 20
  - 2.1.1 Deep learning . . . 20
  - 2.1.2 Detection . . . 22
  - 2.1.3 Segmentation . . . 24
- 2.2 Synthetic data . . . 26
  - 2.2.1 3D computer graphics . . . 26
  - 2.2.2 Early-stage generative models . . . 28
  - 2.2.3 Diffusion models . . . 29
  - 2.2.4 Style transfer . . . 31
- 2.3 Domain adaptation . . . 33
  - 2.3.1 Pixel space adaptation . . . 34
  - 2.3.2 Feature space adaptation . . . 34

2.3.3 Other approaches . . . 35
2.4 Metrics . . . 35
2.4.1 Image quality . . . 36
2.4.2 Object recognition performance . . . 38
2.5 Datasets . . . 39

3 Domain adaptation using diffusion-based style transfer 43
3.1 Motivation . . . 44
3.2 Related work . . . 44
3.2.1 Style transfer with diffusion models . . . 44
3.2.2 Synthetic to real domain adaptation . . . 45
3.3 Methodology . . . 46
3.3.1 Class-wise adaptive instance normalization . . . 47
3.3.2 Selective attention filtering . . . 48
3.3.3 Combined approach . . . 50
3.4 Content-preserving style transfer . . . 51
3.4.1 Datasets . . . 51
3.4.2 Experimental setup . . . 52
3.4.3 Evaluation metrics . . . 53
3.4.4 Results and analysis . . . 53
3.5 Style transfer for adverse conditions domain adaptation . . . 55
3.5.1 Experimental setup . . . 55
3.5.2 Qualitative analysis . . . 56
3.5.3 Quantitative results . . . 58
3.5.4 Ablation study . . . 59
3.5.5 Computation time . . . 60
3.6 Conclusion . . . 60

4 Style transfer pipeline optimization 63
4.1 Motivation . . . 64
4.2 Context . . . 65
4.2.1 Style transfer for domain adaptation . . . 65
4.2.2 Data augmentation optimization . . . 65
4.2.3 Data similarity metrics . . . 66
4.3 Methodology . . . 66
4.3.1 Traveling Salesman Problem (TSP) formulation . . . 67
4.3.2 Multi-objective evolutionary algorithm . . . 68
4.4 Experiments . . . 68
4.4.1 Image operators . . . 68
4.4.2 Optimization configuration . . . 68
4.4.3 Optimization results . . . 70
4.5 Analysis of optimized pipelines . . . 70
4.5.1 Optimized pipelines . . . 70
4.5.2 Pipeline evaluation . . . 74
4.6 Conclusion and perspectives . . . 76

5 Application to runway detection **77**
5.1 Applicative context . . . 78
5.2 Synthetic-to-real domain adaptation for runway detection . . . 78
5.2.1 Motivation . . . 78
5.2.2 Related work . . . 79
5.2.3 Data overview . . . 80
5.2.4 Experimental setup . . . 82
5.2.5 Results . . . 83
5.2.6 Conclusion and perspectives . . . 85
5.3 CACTI_{F} application to runway detection . . . 86
5.3.1 Motivation . . . 86
5.3.2 Data overview . . . 86
5.3.3 Experiments . . . 88
5.3.4 Qualitative results . . . 89
5.3.5 Quantitative results . . . 93
5.3.6 Conclusion and perspectives . . . 97

6 Conclusion **99**
6.1 Summary of contributions . . . 100
6.2 Perspectives for the proposed methods . . . 101
6.2.1 Improvements to diffusion-based style transfer . . . 101
6.2.2 Improvements to style transfer pipeline optimization . . . 102
6.2.3 Observations on diffusion-based style transfer for domain adaptation . . . 103
6.3 Prompt-based style transfer: an exploration and its limits . . . 104
6.4 Reflections on metrics and evaluation . . . 105
6.4.1 The absence of a cheap proxy for downstream performance . . . 105
6.4.2 Benchmark bias and the GTA→Cityscapes monoculture . . . 106
6.4.3 Towards an open industrial domain adaptation benchmark . . . 107
6.5 General conclusion . . . 107

Chapter 1

Introduction

This chapter introduces the motivations and context of this thesis. We present the progressive adoption of computer vision models in industrial applications and the challenges posed by data acquisition and annotation. We discuss synthetic data as a cost-effective alternative to real-world data collection and introduce the synthetic-to-real domain gap as a central challenge. Input-level domain adaptation strategies, including style transfer methods, are presented as promising approaches to bridge this gap. We then detail two industrial use cases: autonomous driving and runway detection for automatic landing, with a focus on adverse weather conditions. Finally, we outline the thesis contributions and structure.

1.1 Motivation for synthetic data

This section presents the context motivating the use of synthetic data for computer vision.

1.1.1 Uses of vision models

Vision models are increasingly deployed in real-world applications, creating a demand for large annotated datasets that are costly and difficult to obtain.

Industrial use cases

Computer vision has experienced significant growth over the last decade, with applications spanning both industry and academia [Maslej et al. (2025)]. Leveraging the diverse range of available sensors, detectors and cameras, vision algorithms promise efficiency, effectiveness and affordability for solving a variety of problems. For instance, 3D reconstruction methods generate three-dimensional models from two-dimensional images; video tracking algorithms estimate the positions of moving objects in video sequences; and human pose estimators infer human skeletal structures from images or videos.

To solve these problems, traditional feature-based computer vision exploits image structures such as edges, points or simple shapes. On the other hand, deep learning methods use large amounts of data to train neural networks. Compared to traditional approaches that require manual feature engineering, they offer strong generalization capabilities and improved performance. As a result, many industries are increasingly adopting these methods in embedded products. For instance, neural networks are incorporated into mobile phones for automatic picture enhancement. In healthcare, deep learning models assist with medical image analysis and diagnostics [Lindroth et al. (2024)]. In transportation, fully autonomous vehicles rely on camera-based perception systems to navigate complex environments [Waymo (2025)].

Data sources

In deep learning based computer vision, needing large amounts of data translates into the necessity of having access to large image collections. Public datasets exist for generic tasks such as object classification [Russakovsky et al. (2015)], detection [Everingham et al. (2010)], or segmentation [Lin et al. (2014)]. However, those datasets are often unsuitable for industrial applications due to their lack of specificity. Neural networks trained on such datasets can recognize everyday items such as tables, cars or skis, but they rarely match the precise classes or environments required in practice. This makes additional domain-specific data necessary.

As a result, each use case typically requires the creation of a new annotated dataset. For vision tasks, this usually involves collecting images with standard cameras and manually labeling them afterward. Image acquisition can be costly depending on the context. For instance, vision-based collision detection for planes requires flying one or even two aircraft [Ghosh et al. (2023)]. Segmenting deep-sea marine life necessitates expensive underwater cameras and sea expeditions involving ships, divers or robots [Iyer et al. (2025)]. Data collection can also be challenging: rare events may occur too infrequently to build sufficient training datasets, while

certain situations may be too hazardous to capture safely, such as runway incursions during aircraft landings. Finally, each dataset piece needs relevant metadata, also called label or annotation, to be effectively exploited. In computer vision, this additional information may describe many aspects such as the objects appearing in an image, their shape, their localization or even their name or description. This labeling process itself is time-consuming. In the simplest case of image classification, annotation takes only seconds. However, for tasks requiring semantic segmentation, annotating a single image can exceed 1.5 hours [Cordts et al. (2016)].

Recent trends in vision models

As an alternative to labor intensive labeling, fine-tuning approaches have been developed to exploit already available databases. This process allows the training of a neural network on a broad unspecialized dataset, followed by a light retraining of specific parts of the model on the target dataset. The retrained section of the model is usually known as the *head*, as opposed to the frozen part of the neural network called the *backbone*. Until recently, model backbones were trained on ImageNet-1k, containing about 1.2 million images across 1000 classes [Russakovsky et al. (2015); Redmon et al. (2016); Ren et al. (2017)]. This enabled the use of relatively powerful models with short training time, due to the easy sharing of pre-trained backbones. The emergence of self-supervised learning has since shifted this paradigm toward much larger datasets. Those methods aim to build strong representations of the training datasets and capture meaningful relationships in the data, without requiring annotations. This paradigm allowed the creation of massive unlabeled datasets encompassing billions of images [Schuhmann et al. (2022)]. Large-scale models capitalizing on these considerable amounts of data emerged. Known as foundation models, they rely on self-supervised learning [Oquab et al. (2023); Zhou et al. (2022)]. As a result, foundation models have grown largely popular for their ability to perform novel tasks without any retraining. However, these large-scale models typically require substantial computational resources. Their size makes inference costly, limiting their adoption in all the industrial contexts where real-time constraints or hardware limitations apply. Additional limitations of large foundation models include restricted availability of training data. Since obtaining billions of images is out of reach for private individuals and most corporations, it leads to the practical impossibility of complete model retraining. The excessive computational power and data requirements leave only the option to use pre-trained models without guarantees or control over model biases and performance in edge cases. Extensive research focuses on addressing these issues. Explainability methods aim to uncover internal model biases and better understand the underlying decision-making processes [Kazmierczak et al. (2025)]. In parallel, resource-efficient fine-tuning techniques aim to specialize models using only a small number of images, improving model accessibility for specific use cases [Han et al. (2024)].

In summary, foundation models are powerful but expensive to deploy and often unsuitable for embedded resource-constrained pipelines. Classical models are smaller, faster and more specialized. However, they require carefully curated and annotated data to achieve good performance. Such data is difficult to obtain, whether due to acquisition costs or time-intensive labeling processes, and this obstacle prevents wider machine learning adoption [Luley et al. (2023)]. This data acquisition challenge motivates the use of synthetic data.



Figure 1.1: Simulators or videogames are able to generate images with accurate annotations. In this example, the GTA5 videogame [Richter et al. (2016)] provides pixel-level semantic segmentation maps for each generated image, which can be used for training computer vision models.

1.1.2 Synthetic data

Definition

Synthetic data refers to artificially generated data. In our context, it is produced by algorithms or, more broadly, software systems. This thesis focuses on image generation for computer vision applications. Synthetic data generation is used across all application domains because it allows for precise control over the output. The generator can be tailored to the target application, producing images that closely match the desired domain.

Simulators often rely on 3D modeling to represent the physical world. In this sense, any software capable of modeling real-world environments can serve as a simulator, including 3D graphics tools (Blender), video games (GTA5 [Richter et al. (2016)], X-Plane [Chen et al. (2023)]), game engines (Unreal Engine) or satellite imagery [Ducoffe et al. (2023)]. Depending on the use case, one can either use these 3D modeling tools to construct specific scenarios and create custom 3D scenes, or utilize the world representations embedded in existing simulators. Once a scene is defined, images and their corresponding annotations can be generated automatically with high confidence. For instance, a driving simulator can automatically export pixel-level semantic segmentation maps alongside each rendered frame, as shown in Figure 1.1.

Benefits

Synthetic data offer several advantages. First, they are inexpensive. No specialized hardware, field campaigns, or extensive manual annotation is required, as a single machine and appropriate software are sufficient. Even accounting for software licensing costs, it is substantially cheaper than the human resources needed to operate cameras in appropriate scenarios and label the resulting images.

Second, synthetic data can be generated at scale. Simulators can produce thousands of images in hours, enabling datasets to be precisely adapted to task requirements. Simulators enable extensive variability in generated images. Given their

complete controllability, it is straightforward to add or remove objects, modify lighting conditions, textures or weather. They also allow camera viewpoint adjustments, generating every necessary configuration.

Finally, annotations generated by simulators are highly accurate. Human annotations are error-prone and require substantial verification to ensure dataset accuracy. In contrast, simulators provide perfect ground truth by construction, which is particularly valuable for dense tasks such as semantic segmentation.

Domain gap Despite the high quality of synthetic data itself, using simulated data in place of real data for downstream applications does not necessarily translate to strong performance. When training computer vision models on simulated data and evaluating on real data, performance typically degrades. This phenomenon is known as the synthetic-to-real domain gap [Tobin et al. (2017); Tremblay et al. (2018)]. Domain gaps, or domain shifts, are not exclusive to the synthetic-to-real scenario and can occur whenever test data differs substantially from training data [Csurka (2017)]. In computer vision, this gap persists even with highly realistic simulators. Foundation models appear to mitigate this gap through extensive pretraining, but this requires billions of real images beforehand, in addition to their computational demands [Oquab et al. (2023)].

1.1.3 General problem statement

While synthetic data offers possible benefits, computer vision models struggle to generalize to real data when trained exclusively on synthetic images, due to the domain gap. For industrial applications with limited computational budgets or real-time constraints, the key question is whether this gap can be reduced to improve the performance of models trained primarily on synthetic data. This is a central question of this thesis.

1.2 Strategies for synthetic data use

Various strategies exist for incorporating synthetic data into machine learning applications [Schwonberg and Gottschalk (2025)]. The simplest approach involves directly mixing synthetic and real data in the training dataset, though this is not the most effective method [Tremblay et al. (2018)]. Other approaches are more sophisticated, including domain randomization [Tobin et al. (2017)], feature alignment, and input-level adaptation methods.

1.2.1 Domain adaptation

Domain adaptation encompasses methods that aim to mitigate the domain gap between a source and a target domain. This research field is well-suited to the synthetic-to-real scenario where a consistent and identifiable domain shift exists. Domain adaptation methods can be categorized according to how they modify a model [Schwonberg et al. (2023); Schwonberg and Gottschalk (2025)]. Some approaches alter the input data, for instance by augmenting or diversifying input images. Others modify the network architecture to learn internal representations that are robust to domain shift. In our work, we primarily focus on input-level approaches.

1.2.2 Input-level domain adaptation

Input-level domain adaptation aims to reduce the domain gap by modifying the neural network's training data. This strategy seems straightforward, as training on data that closely resemble the target domain reduces the need for the model to address the domain shift, enabling simpler architectures and training workflows. However, creating realistically appearing synthetic datasets is itself a complex problem. This challenge has been studied extensively and remains an active area of research [Chen et al. (2025)]. Considerable effort has been invested in modeling realistic 3D worlds, facilitated by improvements in computational resources. This remains highly labor-intensive work, with video game studios and game engine developers actively working to improve 3D model rendering. While some aspects appear highly realistic today, such as modeling fine hair or intricate textures, environmental physics still requires refinement especially for water or weather rendering [Wang et al. (2024d); Koschier et al. (2022); Li et al. (2024b)].

An alternative approach relies on generative models. Early methods used Variational Auto-Encoders (VAE) [Kingma and Welling (2014)] and Generative Adversarial Networks (GAN) [Goodfellow et al. (2014)], while recent works favor more powerful models such as flow-based generative models and diffusion models [Sohl-Dickstein et al. (2015)]. Diffusion models can generate highly realistic images, but they are computationally expensive, difficult to control, and do not naturally provide annotations. These limitations are shared by earlier generative approaches such as VAEs and GANs.

On the other hand, style transfer methods [Gatys et al. (2016); Alaluf et al. (2024)] aim to transfer style from one or multiple target images to a content image. This enables applying realistic styles to synthetic image datasets, effectively combining accurate geometry from 3D computer graphics and real-world information from target images. While this approach still requires substantial computational resources, it preserves the structure and objects present in synthetic images along with their associated labels. Moreover, the computational overhead is incurred only once during dataset generation and is not required during training or inference, in contrast to the foundation model-based methods described earlier.

1.3 Industrial use cases

The computer vision adoption in industrial applications (Section 1.1.1) has led to use cases with infeasible exhaustive real-world datasets and reliance on simulated data to cover corner cases. Thus, many applications suffer from domain shift and could benefit from domain adaptation methods. This section presents in more details the autonomous driving and runway detection use cases. While domain adaptation is generally task-agnostic [Peng et al. (2022); Ohkawa et al. (2023); Guan and Liu (2022)], this thesis focuses on the use of synthetic data for object recognition on real data.

1.3.1 Autonomous driving

Autonomous driving is a classical use case in object recognition literature. Thanks to various open-source datasets and methods easily transferable to other tasks, it

remains an active research application [Muhammad et al. (2022)]. Autonomous driving is particularly susceptible to domain shift. First, driving scenarios encompass a vast range of events that cannot always be captured before training. Second, there exists a large variety of classes, including objects, people, street furniture, trees and road configurations. Finally, significant intra-class variance also exists, with objects of the same class exhibiting very different sizes, shapes or colors. For example, a car can range from a few pixels to half the image depending on distance from the camera, can be various colors, and may refer to pickup trucks, sedans or small city cars. Environmental conditions vary considerably, and urban planning differs significantly between countries.

As a result, both content and appearance domain gaps are unavoidable when designing ML-based recognition systems. The content gap refers to discrepancies in the presence, frequency, or configuration of objects and scenes between training and testing domains, such as unseen object categories, rare events, or novel traffic layouts. In contrast, the appearance gap arises from differences in low-level visual characteristics, including illumination, weather conditions, sensor properties, textures, and rendering styles. Driving simulators have been used to address content shift between training and testing, introducing new challenges in the process in the form of appearance discrepancies [Li et al. (2024b)]. Hence, effective mitigation of the synthetic-to-real domain gap remains a crucial research problem.

1.3.2 Runway detection for Autoland

Autoland is a function embedded in aircraft that allows automatic landing, with supervision from the flight crew or completely autonomously in case of emergency. This functionality is useful in numerous cases, with its ability to land the aircraft in particularly bad weather conditions, or in case of incapacitated pilots. This system requires having access to an Instrument Landing System (ILS) available at the airport. An ILS is typically a radio emitter indicating the aircraft position relative to the landing runway, allowing the autopilot to control the aircraft through the landing procedure. However, ILS installations are limited to major airports, mainly in the northern hemisphere. ILS is also highly sensitive to interference from surrounding buildings or ground traffic, limiting its installation to specific runways. These limitations constrain Autoland usage to particular scenarios.

The Autoland function relies mainly on the ILS for aircraft position information. Vision-based pose estimation using onboard cameras is a promising alternative. Detecting runway location is the first step toward this goal. Given an aerial image from the aircraft's perspective, the problem consists of estimating a bounding box around the runway corners. While the problem involves a single object class, it presents several challenges. Embedded deployment imposes strict model size constraints, ruling out large foundation models. Additionally, there is a notable shortage of open-source datasets for runway detection or segmentation. While some synthetic datasets are available [Ducoffe et al. (2023); Wang et al. (2024c)], novel methods need real images to validate their conclusions. Object size varies greatly: during the landing phase, the detection system must function from 2200 meters before the runway threshold up to the touchdown point. Environmental conditions vary considerably as well, from clear skies to foggy nights, requiring runway detectors robust to domain shift. The same applies to camera-induced variability, depending

on the model, field of view or wear defects.

In the runway detection context, to train and validate systems like Autoland, simulators therefore offer significant value. Flying aircraft to capture data is particularly expensive and requires extensive operational, regulatory, and logistical overhead, especially for wide-body aircraft. Synthetic data can provide cost-effective data, airports from around the world, and diverse weather scenarios. Finally, simulators can provide accurate labels along with additional useful flight metadata.

1.3.3 Adverse conditions

Adverse conditions, as opposed to nominal conditions, refer to challenging weather scenarios that can be difficult for autonomous systems to navigate [Sakaridis et al. (2021)]. In outdoor recognition tasks, nominal conditions typically involve clear skies or cloud ceilings above the camera, with no precipitation. In contrast, adverse conditions include nighttime scenarios, fog, precipitation such as rain or snow, or combinations thereof. These conditions significantly impact object textures and colors. While shapes should remain recognizable, all other visual properties may change. A striking example is the visual difference between a lamppost during day versus night. In some cases such as fog, the objects can be no longer visible, requiring the use of other sensors such as radar in an information fusion strategy. In this thesis, we focus solely on camera.

Autonomous systems must account for these challenging environmental situations, as they will inevitably encounter them when deployed in real-world settings. Simulators are valuable for simulating those widely varying lighting and weather conditions, compensating for the typical scarcity of real examples in training datasets.

1.4 Thesis overview

1.4.1 Specific problem statement

This thesis investigates the problem of synthetic-to-real domain adaptation, with a particular emphasis on image generation approaches for object recognition in deep learning. This work focuses primarily on semantic segmentation, while also considering object detection. Evaluation of the proposed methods is conducted both in autonomous driving scenarios using public datasets and in an industrial application, namely runway detection for automatic landing. For assessment of their effectiveness in a technically constrained and under-researched application domain, we focus on adverse conditions in particular. Indeed, these conditions represent a challenging yet essential problem for any embedded system deployed outdoors, be it autonomous vehicles or runway detection for aircraft.

1.4.2 Outline

We first present a background section in Chapter 2 to introduce the necessary concepts and works related to our problem.

Next, we explore zero-shot style transfer approaches in Chapter 3 and discuss their benefits for domain adaptation applications. Using diffusion models for style transfer tasks offers numerous advantages at the cost of controllability. We mitigate this by

introducing a filtering module to the generation process and a class-specific transfer unit using semantic segmentation maps [Chigot et al. (2025b)]. We demonstrate that class-aware style transfer can improve neural network performance on semantic segmentation tasks in adverse conditions.

Selecting an appropriate style transfer method can be tedious given the large variety of methods in this field [Jing et al. (2019)]. For dataset generation, particularly in domain adaptation contexts, not all style transfer approaches are suitable or satisfactory. In Chapter 4, we propose an automated algorithm selection by optimizing sequences of style transfer functions and image transformations [Chigot et al. (2026b)]. We demonstrate the benefits of using image quality metrics as optimization objectives, and the computational efficiency of the method.

Finally, we apply domain adaptation methods to runway recognition in Chapter 5. We first explore the relevance of a domain adaptation method for object detection, CARE [Prabhu et al. (2023)], in nighttime domain adaptation scenarios. In this study, we find that using synthetic data greatly improves the performance of a runway detection model when lacking real data, especially in nighttime conditions [Chigot et al. (2025a)]. Subsequently, we analyze the impact of style transfer approaches for runway detection in adverse conditions [Chigot et al. (2026a)].

1.4.3 List of publications

[Chigot et al. (2025b)]
Estelle Chigot, Dennis G. Wilson, Meriem Ghrib and Thomas Oberlin
Style Transfer with Diffusion Models for Synthetic-to-Real Domain Adaptation.
In CVIU 2025.

[Chigot et al. (2025a)]
Estelle Chigot, Dennis G. Wilson, Meriem Ghrib, Fabrice Jimenez and Thomas Oberlin
Synthetic Data for Robust Runway Detection.
In CAIP 2025.

[Chigot et al. (2026b)]
Estelle Chigot, Thomas Oberlin, Manon Huguenin and Dennis Wilson
Multi-Objective Optimization for Synthetic-to-Real Style Transfer.
In EvoApplications 2026.

[Chigot et al. (2026a)]
Estelle Chigot, Solène Barrat, Manon Huguenin, Dennis G. Wilson and Thomas Oberlin
Enhancing Runway Detection Models with Synthetic-to-Real Style Transfer.
In DASC 2026.

Chapter 2

Background

This chapter introduces the main concepts and methods necessary to the thesis understanding. We review deep learning architectures for object recognition, covering detection and segmentation tasks. We then present synthetic data generation approaches, from 3D computer graphics to diffusion models and style transfer methods. Domain adaptation techniques are discussed, with a focus on pixel-space and feature-space approaches. Finally, we describe the evaluation metrics and benchmark datasets used throughout this work.

2.1 Object recognition

Object recognition aims at identifying and possibly localizing specific objects within images. As shown in Figure 2.1, object recognition can be divided into separate tasks: image recognition, or classification, identifies what objects are in a given image, while other tasks provide additional information about the localization of the different objects. This localization can be made through bounding boxes, the corresponding tasks being referred to as object detection; or via a pixel-level classification, denoted by semantic segmentation. Instance segmentation lies between these two tasks, by providing pixel-level segmentation within each bounding box. This section introduces the deep learning architectures commonly used for object recognition, followed by specific tasks including detection and segmentation.

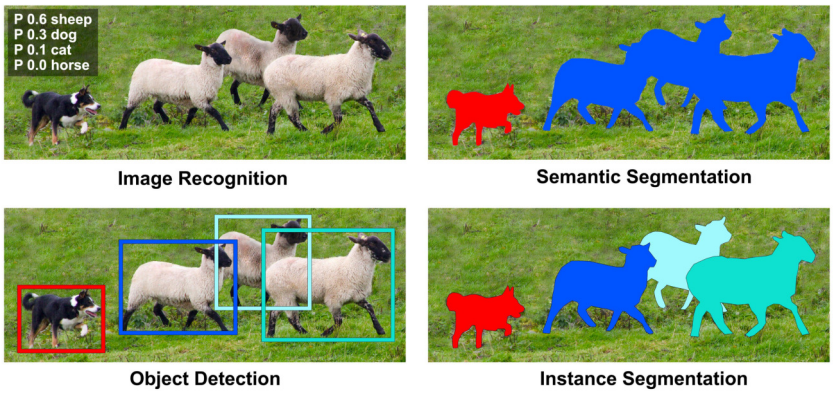


Figure 2.1: Object recognition tasks in computer vision [Tedrake (2024)]. In this chapter, we focus on object detection and semantic segmentation.

2.1.1 Deep learning

The object recognition tasks described above all rely on models that can learn visual patterns directly from data. Machine learning refers to a family of statistical algorithms that learn patterns from data and generalize to unseen examples. Deep learning is a subcategory of machine learning which uses Artificial Neural Networks (ANNs), organized into multi-layer architectures. The scale and depth of these neural networks, as well as their ability to learn features from raw data, give rise to the name *deep learning*. Deep ANN models, or deep neural networks, are the main type of model used in modern object recognition tasks; their scale and depth allow for the statistical learning of complex patterns needed to perform object recognition in natural images. For example, the delineation of the sheep in Figure 2.1 can be learned as a feature in a deep neural network trained on thousands of images, some including sheep. In this section, we describe neural networks and the methods used to train them.

ANN/MLP

An Artificial Neural Network (ANN) is a mathematical function inspired by biological neurons. A neuron receives multiple input values and produces a single output through a weighted sum followed by a non-linear activation function. A neural

network is a composition of multiple interconnected neurons typically organized in layers. Early layers learn low-level patterns (edges, colors, textures) while deeper layers combine these into higher-level representations (shapes, object parts, whole objects). This hierarchical composition is what allows deep networks to learn complex visual features from raw, high-dimensional pixel data.

The Multi-Layer Perceptron (MLP) [Rosenblatt (1958)] is the foundational neural network architecture, consisting of stacked layers of fully-connected neurons. Each neuron in a layer connects to all neurons in the previous layer, allowing information to flow forward through the network. While MLPs can effectively handle tabular or simple data structures, they struggle with high-dimensional data exhibiting spatial correlations, such as images. This limitation motivated the development of convolutional architectures. A 224×224 RGB image has over 150,000 input values, and a fully-connected first layer must learn a separate weight for every input-output pair, with no sharing across positions. This ignores the fact that nearby pixels are strongly correlated, wastes capacity on redundant weights, and scales poorly with image size. This spatial redundancy is what convolutions exploit.

CNN

Convolutional Neural Networks (CNN) [LeCun et al. (1998)] combine traditional fully-connected layers with convolutional layers specifically designed for spatial data. A convolution applies a small learnable matrix, or kernel, that slides across the input, computing a weighted sum at each position to produce a feature map, i.e., a spatial grid of activation values highlighting where a specific pattern occurs in the input. While convolutions can operate in any dimension (e.g., 1D for sequential data such as speech), we focus here on 2D convolutions for image processing. Multiple kernels are learned in each convolutional layer, each detecting different visual patterns. The convolutional operation preserves spatial relationships, making CNNs effective for capturing local and global patterns in visual data. Pooling layers, which aggregate local regions into single values (for example by keeping the maximum value in each region), are typically inserted between convolutional layers to progressively reduce spatial dimensions while retaining important features. This architecture became the standard choice for image-based tasks until the recent emergence of transformer-based approaches. CNNs capture patterns within a bounded spatial window (the receptive field), which limits their ability to relate distant image regions: for example, identifying whether a person on the left side of an image is looking at an object on the right requires global context that even deep CNN stacks struggle to capture without very large receptive fields. This gap motivated attention-based architectures.

Transformers

Transformers [Vaswani et al. (2017)] are neural networks based on attention mechanisms, originally designed for natural language processing. Attention computes correlation maps between all elements in the input data, allowing the model to capture global dependencies. A *token* is the smallest discrete unit of input: in natural language processing, a token is typically a word or a subword; in vision, it can be an image patch. Each token is mapped to an *embedding*: a dense vector in a high-dimensional space. The mapping from token to vector is learned during training such that semantically similar tokens end up nearby in this space. Each

token embedding is then assigned an importance weight relative to other token embeddings in the sequence, enabling the model to focus on context regardless of distance. Within an attention module, each input embedding matrix is projected into three submatrices: Query (Q), Key (K), and Value (V). Self-attention is computed as:

$$\text{Self-attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V,$$

where d is the embedding dimension. The query-key product represents attention maps.

Vision Transformers (ViT) [Dosovitskiy et al. (2021)] adapt this architecture for images by dividing each image into fixed-size patches. The original ViT paper uses 16×16 pixel patches on 224×224 images, giving 196 tokens per image, comparable to a short paragraph of text fed to a language model. Larger patches reduce computation but lose fine-grained spatial detail, so patch size is a key design choice. These patches are flattened and linearly projected into embeddings, then processed by transformer layers. Self-attention is applied across all patches, allowing the model to capture global relationships throughout the entire image. While transformers require more data and computation than CNNs for comparable performance, they demonstrate superior scalability. When trained on sufficiently large datasets, ViT models achieve state-of-the-art results and transfer better to downstream tasks, i.e., specific applications such as detection or segmentation for which the pretrained model is subsequently adapted (fine-tuned) on task-specific data.

Self-attention relates elements within a single input sequence, but many tasks require combining two separate inputs (e.g., a text query and an image, or a source and target sentence in translation). Cross-attention generalizes self-attention to this setting, making it the key mechanism for vision-language models. Cross-attention extends the self-attention mechanism to two distinct inputs. In natural language processing, cross-attention combines input and output sequences, enabling tasks like machine translation. In vision-language models, cross-attention combines image features to text descriptions, forming the basis of models like CLIP [Radford et al. (2021)] or multimodal generative models.

Cross-attention between two inputs a and b is computed as:

$$\text{Cross-attention}(Q_a, K_b, V_b) = \text{softmax}\left(\frac{Q_a K_b^T}{\sqrt{d}}\right)V_b.$$

2.1.2 Detection

Task definition

Object detection identifies which object classes are present in an image and their localization. Unlike image classification, which assigns a single label per image, detection must handle multiple objects of varying classes at different locations and scales. The localization is represented by a rectangular box around each object, called bounding box. A bounding box is often described by four coordinates: the top-left corner (x, y) and the width and height (w, h). Each detection also includes a class label and a confidence score indicating the model's certainty.

The primary challenges in object detection include handling objects at multiple scales, dealing with occlusions, and balancing localization accuracy with classification

performance. Additionally, the number of objects per image varies, requiring models to produce a variable number of predictions.

Popular networks

CNNs Two main paradigms are relevant in the object detection CNN-based literature. First, neural networks designed to predict object proposals from the image features, then select good detections. The second approach fuses the two steps to propose one-stage detection models. Figure 2.2 compares the two methods.

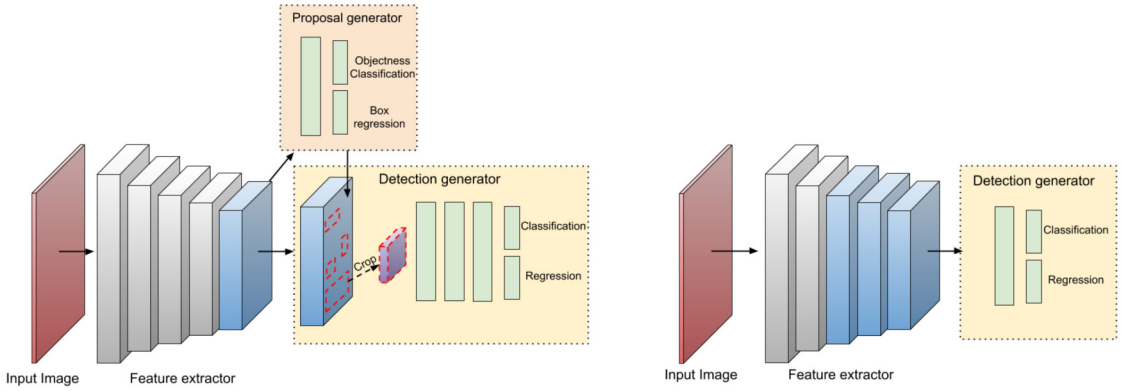
R-CNN [Girshick et al. (2016)] introduced a two-stage detection pipeline that separates localization from classification. First, approximately 2,000 region proposals are extracted using selective search, an algorithm that groups similar pixels into candidate object regions. Second, a CNN extracts features from each proposed region, and a classifier determines the object class. While achieving high accuracy, R-CNN is computationally expensive due to running the CNN on each proposal independently.

Faster R-CNN [Ren et al. (2017)] improved upon R-CNN by replacing selective search with a learned Region Proposal Network (RPN). The input image is passed through a CNN backbone, the main feature-extraction network, to extract a feature map. The RPN then generates proposals directly from the feature map by predicting objectness scores and bounding box refinements at each spatial location. An objectness score is a scalar prediction indicating how likely a given anchor region contains any foreground object at all, regardless of its class; it is essentially a binary foreground/background estimate used to filter candidate regions before classification. Next, proposals are classified and further refined by a detection head. By sharing the backbone computation between proposal generation and detection, Faster R-CNN achieves significantly faster inference than R-CNN while maintaining high accuracy.

YOLO (You Only Look Once) [Redmon et al. (2016)] framed object detection as a single regression problem. The network divides the input image into a grid. For each grid cell, YOLO directly predicts bounding boxes along with their confidence scores and class probabilities in a single forward pass. This one-stage architecture enables real-time detection at over 45 frames per second, making it suitable for embedded applications. However, first YOLO versions struggled with small objects and precise localization compared to two-stage methods.

SSD (Single Shot MultiBox Detector) [Liu et al. (2016)] is also a single-stage approach that addresses scale variation through multi-scale feature maps. An anchor box is a predefined bounding box of fixed size and aspect ratio, placed at each spatial location of the feature map. Rather than predicting absolute box coordinates, the detector predicts offsets that shift and rescale the anchor to fit the actual object. For example, an anchor might be 64×64 pixels with a 1:1 aspect ratio; if a car is slightly larger and to the right, the network learns to output small offsets accordingly. This design reduces the regression problem to learning small corrections rather than raw coordinates. The network uses predefined anchor boxes with various aspect ratios at multiple resolutions to match expected object sizes. By making predictions from different layers of the backbone network, SSD can detect objects at different scales more effectively than YOLO, while maintaining real-time performance.

Transformers DETR (Detection Transformer) [Carion et al. (2020)] combines CNNs with transformer architectures to create an end-to-end detection pipeline



(a) Two-stage Faster R-CNN

(b) One-stage RetinaNet

Figure 2.2: Comparison between two-stages (left) and one-stage (right) object detectors [Carranza-García et al. (2021)].

without hand-designed components. A CNN backbone extracts a feature map from the input image. This feature map is flattened and augmented with positional encodings (vectors that encode each element's spatial location, since the transformer has no built-in notion of position), then passed through a transformer encoder-decoder. The decoder uses a fixed set of learned object queries that attend, through an attention layer, to the encoded features. Object queries are N learnable embedding vectors, one per possible detected object, that interact with the encoded image features through cross-attention. Each query effectively "asks" about a specific object slot in the image, and the model learns during training what each slot should look for. The output is always exactly N predictions which allows end-to-end training without post-processing. Finally, a prediction head processes each decoder output embedding to predict either an object class with its bounding box or a *no object* class. DETR eliminates hand-designed components such as anchor boxes through its set prediction approach. While DETR simplifies the detection pipeline and demonstrates the interest of transformers for vision tasks, it requires longer training than CNN-based detectors and struggles with small objects.

2.1.3 Segmentation

Task definition

Semantic segmentation assigns a class label to every pixel in an image, producing a dense pixel classification map. Unlike object detection which localizes objects with bounding boxes, segmentation displays precise object boundaries and regions. Each pixel is classified into one category, such as road, building, vegetation, or vehicle in urban scene understanding.

The primary challenges in semantic segmentation include handling objects at multiple scales, preserving fine-grained boundary details, and managing class imbalance where some classes occupy significantly more pixels than others.

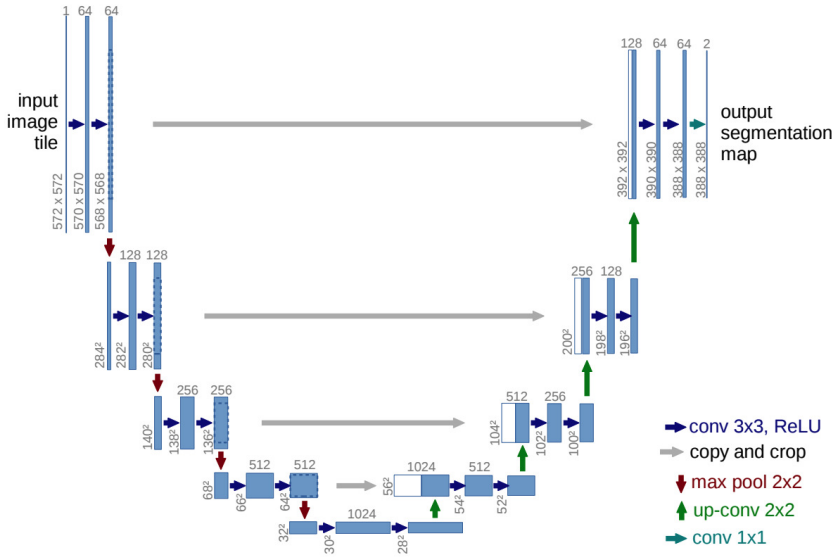


Figure 2.3: U-Net architecture [Ronneberger et al. (2015)].

Popular networks

CNNs FCN [Long et al. (2015)] has been the first fully convolutional architecture for pixel-wise dense prediction. FCN adapts standard image classification CNNs by replacing their fully-connected layers with convolutional layers, enabling pixel-wise predictions. By using only convolutions, the architecture accepts variable input sizes. Then, an upsampling layer, which increases spatial resolution by interpolating between values, restores the output to match the input image size.

Building on FCN, U-Net [Ronneberger et al. (2015)] adds a convolutional expansive path to the original contractive path, thus forming a U-shape. Figure 2.3 displays the U-Net architecture. The encoder progressively reduces spatial dimensions through convolutional and pooling layers, while the decoder restores resolution through upsampling. Skip connections link corresponding encoder and decoder layers at each resolution level. Without them, spatial detail (sharp object boundaries, fine textures) is lost during the aggressive downsampling of the encoder. Skip connections concatenate high-resolution encoder features directly to the corresponding decoder layer, allowing the network to recover fine spatial detail that the decoder's upsampled features alone cannot reconstruct. For instance, a skip connection at $1/4$ resolution provides boundary information that the decoder can use to refine coarse upsampled predictions. Originally designed for biomedical image segmentation with limited training data, U-Net became popular across various domains due to its effectiveness and simple architecture.

DeepLab [Chen et al. (2018)] introduces atrous (dilated) convolutions to enlarge the receptive field without reducing the spatial resolution of the feature maps or adding parameters. A standard convolution applies a kernel to contiguous input positions, while an atrous convolution inserts gaps (controlled by a dilation rate) between kernel elements, allowing each neuron to aggregate information from a wider spatial context at the same computational cost. DeepLab also introduces atrous spatial pyramid pooling (ASPP), which applies parallel atrous convolutions with multiple dilation rates. By capturing context at different scales simultaneously, ASPP allows the network to better handle objects of varying sizes.

Transformers SegFormer [Xie et al. (2021)] combines a transformer encoder with a lightweight MLP decoder for efficient semantic segmentation. Unlike vision transformers that process fixed-size patches at single resolution, SegFormer's encoder generates multi-scale features at different resolutions. The decoder aggregates these features through MLPs without requiring complex operations or positional encodings. This design achieves strong performance while maintaining computational efficiency.

Finally, Segment Anything Model (SAM) [Kirillov et al. (2023)] proposes the first promptable segmentation architecture, enabling zero-shot segmentation, the ability to segment objects from classes never seen during training. The main difference between former architectures and Segment Anything is the scale of its training dataset. Previous transformer models such as SegFormer have been fine-tuned on datasets of tens of thousands of images, such as ADE20K [Zhou et al. (2017)] (about 20,000 training images) or COCO-Stuff [Caesar et al. (2018)] (about 118,000 training images). On the other hand, SAM has been trained on 11 million images and 1.1 billion corresponding masks. This puts Segment Anything in the category of foundation models, large-scale models trained on broad data that generalize to a wide range of tasks without task-specific retraining. SAM's wide adoption has been made possible due to its promptable interface: the model accepts geometric prompts, like a click, a bounding box, or a mask, specifying which object the user wants segmented, and generalizes this to arbitrary objects without retraining. The large-scale training dataset is what makes zero-shot generalization possible.

2.2 Synthetic data

Synthetic data refers to any data created artificially. Any data can be replicated synthetically, but we focus on generating images for computer vision.

As described in the previous section, deep learning models require large amounts of labeled data, from thousands of annotated images for specialized detectors to over 11 million images and billions of masks for foundation models such as SAM. Annotating real images is expensive: pixel-level semantic labels for a single driving image can take over an hour of expert work [Cordts et al. (2016)]. Moreover, some scenarios, such as accidents, sensor failures, or extreme weather, are rare or unsafe to capture in the field. Synthetic data addresses both problems by providing free, exact ground-truth annotations and full control over object placement, camera viewpoint, lighting, and weather conditions.

There are two main methods to generate synthetic images: simulators that model 3D scenes or objects, and data-driven approaches using neural networks [Mumuni et al. (2024); Nikolenko (2021)].

2.2.1 3D computer graphics

3D computer graphics generate synthetic images by modeling scenes in three dimensions and projecting them onto two-dimensional image planes. This 3D-to-2D mapping preserves spatial relationships and geometric properties, enabling automatic generation of pixel-perfect annotations. Compared to data-driven 2D generation, 3D graphics provide explicit control over object shapes, positions, and scene configurations.

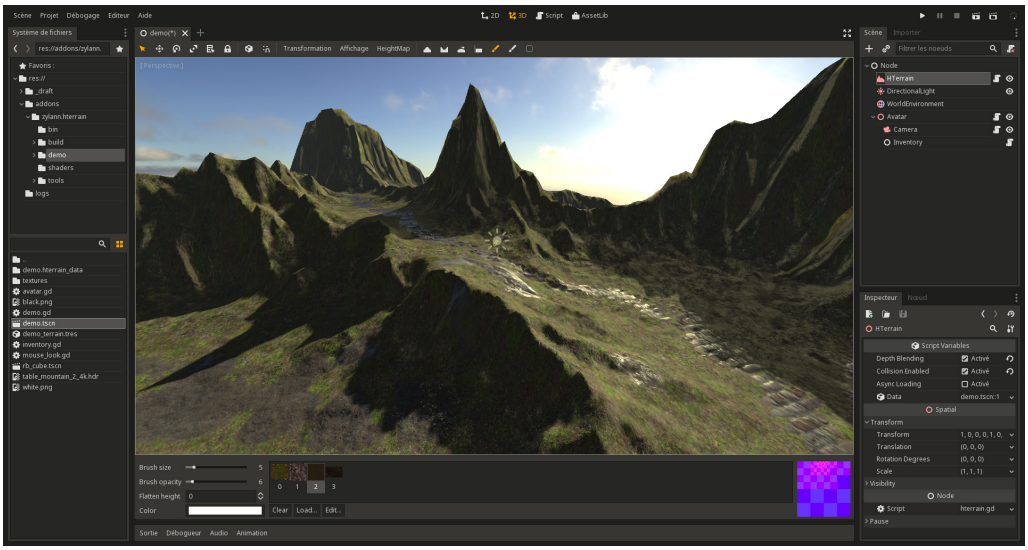


Figure 2.4: Landscape modeling example in the Godot game engine [Zylann (2018)].

3D objects

Creating 3D objects requires either manual modeling or scanning real objects. Tools such as Blender or Cinema4D enable manual modeling with precise geometric control. However, this process is time-consuming and requires specialized expertise. While it produces accurate geometry, achieving photorealistic textures and fine surface details remains challenging.

On the other hand, 3D scanning captures real objects directly, offering faster acquisition with high shape and color fidelity. Modern scanning techniques produce realistic representations suitable for simulation. However, reconstruction algorithms can introduce artifacts, particularly for complex geometries or reflective surfaces.

Several large-scale datasets provide ready-to-use 3D models, either manually modeled or scanned. ShapeNet [Chang et al. (2015)] contains over 50,000 clean models across common object categories. ModelNet [Wu et al. (2015)] offers simplified meshes for shape classification. OmniObject3D [Wu et al. (2023)] provides high-quality scanned objects with realistic textures, bridging synthetic and real appearances.

3D scenes

Scene generation requires assembling complete environments with physics, lighting, and dynamics. Game engines such as Unity, Unreal Engine, Godot or NVIDIA Omniverse provide comprehensive frameworks for this purpose. These platforms include physics simulation, weather modeling, dynamic lighting, and extensive asset libraries, as shown in Figure 2.4. They support animation and camera movement for generating diverse viewpoints and temporal sequences.

Research simulators target specific domains with tailored environments and sensor models. CARLA [Dosovitskiy et al. (2017)] provides an open-source urban driving simulator built on Unreal Engine, offering configurable vehicle sensors (cameras, LiDAR, radar), diverse weather conditions, and traffic scenarios for autonomous driving research. AirSim [Shah et al. (2017)] offers aerial simulation for drone and aircraft applications, featuring realistic physics and sensor emulation for unmanned vehicle research.

Video games offer pre-built, detailed environments for data generation. Grand Theft Auto V [Richter et al. (2016)], Microsoft Flight Simulator, and X-Plane provide visually realistic worlds at low cost. However, they offer limited scene modification flexibility and are rarely open-source, restricting their adaptability to specific research needs.

2.2.2 Early-stage generative models

Deep learning approaches for image generation have evolved significantly over the past decade. Early methods relied on variational autoencoders and generative adversarial networks, which dominated the field until the recent emergence of diffusion models as the state-of-the-art approach.

VAE

Variational Autoencoders (VAEs) [Kingma and Welling (2014)] introduced a probabilistic framework for learning latent representations of data. A VAE consists of an encoder network that maps input images x to a probability distribution $q(z|x)$ over latent variables z, and a decoder network $p(x|z)$ that reconstructs images from latent samples. By enforcing a prior distribution $p(z)$ (typically Gaussian) on the latent space, VAEs enable generation of new samples through sampling and decoding. The model is trained by maximizing a lower bound on the data likelihood, combining reconstruction accuracy with regularization of the latent distribution:

$$\mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{\mathrm{KL}}(q(z|x) \,\|\, p(z)). \tag{2.1}$$

The first term is a reconstruction loss (how well the decoder recovers the input from the sampled latent code z); the second is a divergence that regularizes the encoder distribution $q(z|x)$ toward the prior $p(z)$, preventing the latent space from collapsing to a lookup table.

While VAEs can generate diverse samples and provide smooth latent interpolations, they often produce blurry images due to the reconstruction loss formulation. Vector Quantized VAE (VQ-VAE) [van den Oord et al. (2017)] addressed this limitation by replacing continuous latent variables with discrete codes from a learned codebook, a fixed-size dictionary of embedding vectors. This discrete representation improved image quality and enabled better modeling of complex data distributions. VQ-VAE became particularly relevant for generative tasks when combined with autoregressive models, enabling high-quality image synthesis.

GANs

Generative Adversarial Networks (GANs) [Goodfellow et al. (2014)] proposed a novel image generation through an adversarial training framework. A GAN consists of two competing networks: a generator that produces images from random noise, and a discriminator that distinguishes real from generated images. The generator learns to produce increasingly realistic samples by attempting to fool the discriminator, while the discriminator improves at detecting fake images. This adversarial process pushes both networks toward better performance. GANs produce sharper images than VAEs but can suffer from training instability and mode collapse, where the generator fails to capture the full data distribution.

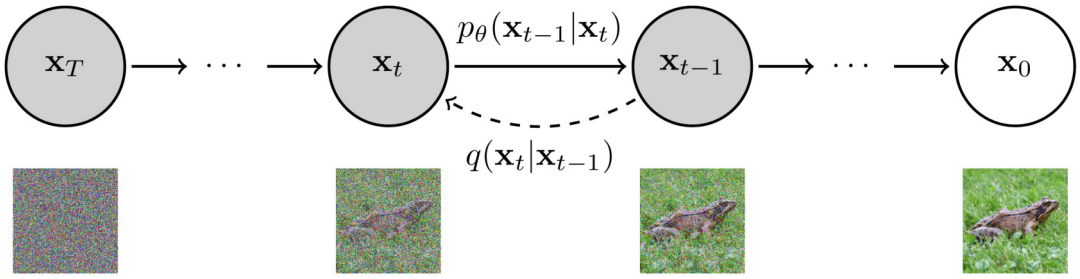


Figure 2.5: Denoising process of the DDPM diffusion framework [Ho et al. (2020)].

Conditional GAN [Mirza and Osindero (2014)] extended the framework by conditioning both generator and discriminator on additional information such as class labels or text descriptions. This enables controlled generation of images with specific attributes, making the approach more flexible for practical applications.

StyleGAN [Karras et al. (2019)] introduced a novel generator architecture that progressively synthesizes images while injecting learned style information at multiple scales. The generator maps input noise to an intermediate latent space, which controls visual attributes at different resolutions through adaptive instance normalization. This architecture enables precise control over image characteristics and produces state-of-the-art photorealistic results. StyleGAN demonstrated that GANs could generate highly realistic faces and other objects, establishing them as the dominant generative approach until the emergence of diffusion models.

2.2.3 Diffusion models

Diffusion models are a class of generative models that have recently surpassed GANs in image synthesis quality [Dhariwal and Nichol (2021)]. These models generate images through an iterative denoising process, gradually transforming random noise into coherent images. Figure 2.5 presents the denoising process developed by Ho et al. [Ho et al. (2020)], one of the most popular implementations of diffusion models.

The foundational work by Sohl-Dickstein et al. [Sohl-Dickstein et al. (2015)] established the theoretical framework by drawing an analogy to thermodynamic diffusion processes. The model learns to reverse a gradual noising process that progressively corrupts data until it becomes pure Gaussian noise. By learning this reverse process, the model can generate new samples by starting from noise and iteratively denoising. Sohl-Dickstein et al. explicitly modeled the process as a Markov chain, a sequence where each step depends only on the previous one, with Gaussian noise added at each transition.

Denoising Diffusion Probabilistic Models (DDPM) [Ho et al. (2020)] refined this approach for image generation. This enables a simpler and faster training. The diffusion process can be formulated as follows. The forward diffusion process gradually adds Gaussian noise to an image $\mathbf{x}_0$ over T timesteps according to:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{2.2}$$

where β_t is a variance schedule that controls the noise level at each step. After reparameterization, we can directly sample $\mathbf{x}_t$ from $\mathbf{x}_0$:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.3}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. As $t \to T$, the image becomes increasingly noisy until $\mathbf{x}_T \approx \mathcal{N}(\mathbf{0}, \mathbf{I})$.

The reverse process, or generation process, learns to denoise by modeling:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \sigma_t^2 \mathbf{I}) \tag{2.4}$$

Rather than directly predicting the mean $\boldsymbol{\mu}_\theta$, DDPM trains a U-Net ϵ_θ to predict the noise $\boldsymbol{\epsilon}$ that was added at step t. The authors simplify the training objective to:

$$\mathcal{L} = \mathbb{E}_{t \sim \mathcal{U}(1,T), \mathbf{x}_0, \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \left[\|\boldsymbol{\epsilon} - \epsilon_\theta(\mathbf{x}_t, t)\|^2 \right] \tag{2.5}$$

To generate new samples, the model starts from pure noise $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and iteratively denoises:

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}_t \tag{2.6}$$

where $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ for $t > 1$ and $\mathbf{z}_1 = \mathbf{0}$.

This reparameterization is mathematically equivalent to denoising score matching [Song et al. (2021b)], connecting DDPM to score-based generative models. DDPM produces more diverse samples than GANs on standard benchmarks, but requires many sequential reverse diffusion steps, making generation slow in practice.

Denoising Diffusion Implicit Models (DDIM) [Song et al. (2021a)] addressed this computational limitation by introducing a deterministic sampling process. DDIM rewrites the sampling process as:

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \left(\frac{\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \epsilon_\theta(\mathbf{x}_t, t) + \sigma_t \mathbf{z}_t \tag{2.7}$$

where $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. By setting $\sigma_t = 0$, the process becomes fully deterministic, removing the stochastic term present in DDPM. This deterministic formulation enables a non-Markovian sampling process, where each step can depend on $\mathbf{x}_0$ directly rather than only on the previous step, so that intermediate steps can be skipped. This reduces the number of required iterations from hundreds to tens while maintaining comparable image quality.

Latent diffusion models, also known as Stable Diffusion (SD) [Rombach et al. (2022)], further improved efficiency by performing the diffusion process in a model latent space rather than pixel space. A variational autoencoder first compresses images into lower-dimensional representations. The diffusion model then operates on these smaller features, significantly reducing computational requirements while preserving generation quality. The VAE in Stable Diffusion compresses a 512×512 RGB image to a 64×64×4 latent, an 8× downsampling factor. The denoising UNet contains self-attention layers at multiple resolution levels. These layers allow the network to capture long-range spatial dependencies within the image being generated. Operating on 64×64 latents rather than 512×512 pixels reduces the sequence length by a factor of $512^2/64^2 = 64$; since the computation cost in attention layers scales quadratically with sequence length, this reduces attention cost by a factor of 64^2. This is why latent diffusion made high-resolution generation practical on consumer hardware. This architecture enables text-to-image generation through cross-attention mechanisms that condition the denoising process on text embeddings.

DDPM inversion [Huberman-Spiegelglas et al. (2024)] maps a real image to its corresponding noise representation by applying the forward diffusion process. Given a real image, the procedure computes a sequence of increasingly noisy latents at each timestep, ultimately producing a sequence of noise maps from which the standard denoising process can reconstruct the original image. This inversion enables image editing: by modifying the text prompt or manipulating intermediate latents before running the denoising process, one can alter specific attributes of the image while preserving its overall structure.

ControlNet [Zhang et al. (2023a)] extends latent diffusion by adding spatial conditioning capabilities. The method introduces additional encoder branches that process conditioning inputs such as edge maps, depth maps, or segmentation masks. These branches guide the generation process while preserving the pretrained model's generative capabilities, allowing precise control over spatial layout and object placement.

Efficient adaptation techniques have also emerged. LoRA (Low-Rank Adaptation) [Hu et al. (2022)] enables fine-tuning large models with minimal parameter updates by learning low-rank matrices that modify the model's attention layers. This reduces memory requirements and training time significantly. DreamBooth [Ruiz et al. (2023)] personalizes text-to-image models on specific subjects by fine-tuning the full model on only a few example images, combined with a class-specific prior preservation loss. This enables generation of the learned subjects in novel contexts while maintaining the model's prior knowledge. In practice, DreamBooth is often combined with LoRA.

2.2.4 Style transfer



Figure 2.6: The diffusion-based method InstantStyle [Wang et al. (2024a)] transfers Van Gogh's painting style (center image) onto a photograph (left image). Resulting image is on the right.

Style transfer aims to apply the visual style of one image to the content of another image, producing a stylized output that preserves content structure while adopting style characteristics. Figure 2.6 displays a style transfer example. Style transfer is a specific case of image-to-image translation, which more broadly covers any learned mapping between two image domains (e.g., day-to-night, aerial-to-map, sketch-to-photograph). In style transfer, the domain difference is defined by visual style. This task separates images into two components: style and content.

Style encompasses appearance-related features independent of object identity, including colors, textures, brush strokes, and image grain. Content refers to the objects present in an image, their spatial arrangement, and geometrical properties.

In practice, content can often be approximated by object edges and boundaries, while style corresponds to their visual filling.

Style transfer was initially developed for artistic applications. Early methods employed stroke-based [Hertzmann (1998)] or region-based [Gooch et al. (2002)] techniques to reproduce image content with painterly rendering effects.

Gatys et al. [Gatys et al. (2016)] introduced the foundational neural style transfer approach using convolutional networks. They leverage a pretrained VGG network to extract features from both content and style images at multiple layers. Content is captured through deep layer activations representing high-level structure, while style is encoded through Gram matrices computed across multiple layers, capturing texture and color distributions. Given a feature map $F^l \in \mathbb{R}^{C \times HW}$ at layer l, its Gram matrix is $G^l = F^l(F^l)^T \in \mathbb{R}^{C \times C}$, capturing pairwise correlations between all feature channels summed over spatial positions. Because the spatial average removes location information, G^l encodes which textures and colors co-occur throughout the image without recording where, which is what defines "style" independently of content. An optimization process iteratively updates a generated image to minimize a loss measuring both content preservation and style similarity. This optimization-based approach produces high-quality results but requires separate optimization for each content-style pair, making it computationally expensive.

Pix2pix [Isola et al. (2017)] leverages the GAN framework to learn mappings between paired images from two domains. Both generator and discriminator are conditioned on input images, with the generator transforming inputs into the target domain while the discriminator distinguishes real from generated outputs. The training loss combines adversarial and reconstruction objectives. However, this method requires aligned paired datasets for training. Each source image must have a spatially corresponding target image at the pixel level, for example, the same scene photographed in daylight and at night from the same viewpoint, or a semantic label map paired with the corresponding photograph. Collecting such pairs is expensive in practice, which directly motivates the unpaired approach of CycleGAN in the next paragraph.

CycleGAN [Zhu et al. (2017)] extends the adversarial framework with cycle consistency constraints, ensuring that translating an image to the target domain and back returns to the original form. This unpaired approach eliminates the need for aligned datasets. However, both pix2pix and CycleGAN are trained for specific domain translations and require retraining to adapt to new styles.

AdaIN [Huang and Belongie (2017)] simplifies the style transfer framework by aligning feature statistics rather than computing Gram matrices. The method normalizes content features to match the mean and variance of style features extracted from a VGG encoder:

$$\text{AdaIN}(x, y) = \sigma(y) \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \mu(y), \tag{2.8}$$

where $\mu(x)$ and $\sigma(x)$ denote the mean and standard deviation computed across spatial dimensions for each channel. A decoder network reconstructs the stylized image from these aligned features. Unlike Gatys et al., AdaIN enables arbitrary style transfer in a single forward pass without per-style optimization, though with less fine-grained control than previous approaches.

Finally, recent methods turn to diffusion models to perform style transfer. Cross-Image Attention [Alaluf et al. (2024)] employs Stable Diffusion for zero-shot style

	GTA5 → GTA5	GTA5 → Cityscapes
mIoU	66.06	35.99

Table 2.1: DeepLabV3+ semantic segmentation model trained on GTA5 and evaluated on both GTA5 and Cityscapes datasets.

transfer. The method modifies self-attention layers in Stable Diffusion by injecting the keys and values from the style image into the self-attention computation of the content image. This shared attention mechanism produces outputs semantically consistent with the content image while adopting style characteristics, without requiring model retraining.

Other emerging approaches include flow matching and Schrödinger bridge methods. Flow matching learns a continuous transport map between source and target distributions, offering a simulation-free alternative to diffusion models [Lipman et al. (2023)]. Schrödinger bridge formulations provide a probabilistic framework for unpaired image-to-image translation by finding the optimal stochastic transport between two distributions [De Bortoli et al. (2021)].

2.3 Domain adaptation

Domain gap arises when the training dataset does not represent the same distribution as the test dataset, which happens when the context or the environment changes. In the domain adaptation literature, the training dataset is referred to as the source domain, while the test dataset is the target domain.

A semantic segmentation model trained on GTA5 synthetic images typically drops more than 30 mIoU points when evaluated on real Cityscapes images, as shown in Table 2.1, despite both datasets sharing the same object classes, such as cars and pedestrians. The model has learned to associate synthetic rendering artifacts and color palettes with class labels, for example the way cars appear in GTA5 determines the car class; those associations do not always transfer to real images. Closing the domain gap is critical in numerous applications, particularly in outdoor computer vision where environmental variability affects performance, including lighting, weather conditions, and unexpected scenarios. A robust machine learning system must account for this variability.

Domain adaptation methods can be broadly categorized into supervised and unsupervised approaches. Supervised domain adaptation assumes access to labeled data in both source and target domains, while unsupervised domain adaptation assumes labels are available only in the source domain. However, this distinction becomes less clear in synthetic-to-real scenarios, especially when using style transfer methods. Unaltered synthetic data serves as the source domain, while stylized synthetic data can function as either source or target. Additionally, having only a small number of target annotations (1-10 images) falls between unsupervised and fully supervised approaches.

Domain generalization is a closely related problem that aims for robustness to completely unseen scenarios. Some domain generalization methods focus on generating training datasets in adverse conditions, which make them close to synthetic-to-real domain adaptation approaches. This section covers methods from both supervised

and unsupervised domain adaptation, as well as relevant domain generalization techniques.

2.3.1 Pixel space adaptation

Pixel space adaptation addresses the domain gap by transforming training data to resemble test data. This approach offers several advantages. First, models require no architectural modifications or specialized weighting schemes, allowing them to remain lightweight and efficient. Second, pixel space methods are architecture-agnostic and compatible with any neural network. Third, while generating sufficiently diverse stylized datasets can be expensive, the computational cost is incurred once during dataset generation rather than repeatedly during training or inference.

Several approaches generate adapted datasets to improve target domain performance. Among them, DACS [Tranheden et al. (2021)] mixes images from both domains to augment the training dataset size and improve robustness. Specifically, DACS uses ClassMix augmentation: a random subset of semantic classes is selected from a source image and the pixels belonging to those classes are copied onto a target image. The composite image retains the source labels for the pasted region and uses pseudo-labels, predicted by the current model, for the target region. This creates hybrid training images that bridge the two domains while providing dense supervision.

SC-UDA [Yu et al. (2022)] trains a CycleGAN model to transfer realistic style from the unlabeled target dataset onto synthetic images. A detector trained on these stylized images then generates pseudo-labels for unlabeled target images. The final detector is trained on both accurately labeled stylized images and pseudo-labeled target images.

DATUM [Benigmim et al. (2023)] fine-tunes the Stable Diffusion [Rombach et al. (2022)] model on a single target image using Dreambooth [Ruiz et al. (2023)] for a few iterations. The adapted model then generates target objects in the style of the reference image through text prompting.

Finally, DGInStyle [Jia et al. (2024)] takes a different approach by fine-tuning Dreambooth on the source domain before training ControlNet to generate images conditioned on segmentation maps. The method then applies a style swap operation, replacing the source-specific UNet with a pretrained generalist model. This process preserves segmentation conditioning while preventing style leakage from the source dataset. Unlike SC-UDA and DATUM, DGInStyle does not require target style images.

2.3.2 Feature space adaptation

Feature space adaptation addresses domain shift by aligning internal network representations rather than modifying input images. Unlike pixel space methods, these approaches integrate adaptation directly into the training process, eliminating the need for separate dataset generation. These methods operate through several mechanisms: aligning feature distributions between domains, learning domain-invariant representations, or adapting network architectures to handle domain shifts.

DAFormer [Hoyer et al. (2022a)] introduces a transformer-based architecture for domain-adaptive semantic segmentation. The method combines a transformer

encoder with a multi-level context-aware decoder. DAFormer employs self-training with pseudo-labels and rare class sampling to achieve state-of-the-art performance on synthetic-to-real benchmarks. The transformer encoder captures multi-scale context, which is useful for handling appearance variations across domains.

HRDA [Hoyer et al. (2022b)] extends DAFormer by incorporating context-aware high-resolution processing. The method processes both small high-resolution crops and large low-resolution crops jointly, using learned scale attention to combine them. This multi-resolution approach improves segmentation of small objects and precise boundaries while maintaining global context.

CARE [Prabhu et al. (2023)] targets object detection rather than segmentation, addressing the sim-to-real gap through conditional feature alignment and instance reweighting. The method aligns features between source and target domains while accounting for class disparities, assigning different weights depending on class statistics. This method assumes access to about 10% of the source dataset's size in labeled target images.

Finally, PØDA [Fahes et al. (2023)] leverages vision-language models for prompt-driven zero-shot domain adaptation. The method uses a frozen CLIP-based ResNet encoder to learn feature transformations guided by textual prompts describing target domains. These transformations align source feature embeddings with target domain prompt embeddings. The segmentation model is fine-tuned on feature-augmented source images, enabling inference on unseen target domains without requiring any target images.

2.3.3 Other approaches

Beyond pixel and feature space adaptation, several complementary strategies address the domain gap through alternative mechanisms. These approaches often combine multiple algorithmic elements from other types of methods, such as adaptations in the pixel or feature space.

Pseudo-labeling, mentioned in SC-UDA [Yu et al. (2022)] or DAFormer [Hoyer et al. (2022a)], is often used to generate intermediate annotations during the training process. Usually, those labels improve from low quality to high quality when the model improves over time. Methods that explicitly use this approach include Zheng et al. [Zheng and Yang (2021)] or MIC [Hoyer et al. (2023)].

Finally, another approach of interest is test-time domain adaptation, which adjusts model parameters at inference when encountering out-of-distribution data. Notable methods include Meta-DMoE [Zhong et al. (2022)], which uses meta-distillation from a mixture of experts, domain-aware batch normalization [Wu et al. (2024)], which adapts normalization statistics to the target distribution, and continual test-time adaptation [Wang et al. (2022a)], which updates the model online as new target data arrives.

2.4 Metrics

For domain adaptation, we aim to measure how well images match the target domain's style, how they preserve the content of the source image, and finally how well the images aid in training object recognition models. First, we consider the main image quality metrics used to evaluate generated or stylized images, with regards

to how well style is applied and how well content is preserved. Then, we consider performance metrics which evaluate downstream object recognition results on real data. We detail existing metrics here, noting that the necessity of more accurate metrics on downstream model performance is discussed in Chapter 6.

2.4.1 Image quality

Structure

PSNR (Peak Signal-to-Noise Ratio) measures pixel-level fidelity in decibels, where higher values indicate closer agreement to the reference at the individual pixel level; SSIM (Structural Similarity Index) [Wang et al. (2004)] compares local statistics on a 0–1 scale, with the aim of better accounting for the human visual perception. Both correlate poorly with human perceptual judgments of similarity, which is the main limitation that motivates learned metrics such as LPIPS.

LPIPS [Zhang et al. (2018)] is a perceptual metric learned on human similarity judgements from their proposed BAPPS dataset. It is defined as a weighted MSE in the feature space of a pretrained CNN classifier, the weights being adjusted to best match human judgement. To compute the distance between two images, each image is passed through the network and the layer activations are extracted. The distance is calculated by normalizing and averaging the activation differences across all layers:

$$d(x, y) = \sum_{l} \frac{1}{H_l W_l} \sum_{h,w} \left\| w_l \left(\hat{x}^l_{hw} - \hat{y}^l_{hw} \right) \right\|_2^2, \tag{2.9}$$

where:

- x and y are the two images to compare,
- $\hat{x}_l$ and $\hat{y}_l$ are the normalized activations of layer l,
- w_l are learned weightings for each layer l,
- H_l and W_l are the spatial dimensions of the feature maps at layer l.

While originally designed to evaluate perceptual quality, LPIPS is commonly used as a content preservation metric [Chung et al. (2024); Xu et al. (2025)]. The convolutional layers maintain strong spatial correlation with the original images, enabling structure comparison without strict pixel-wise matching. This makes LPIPS particularly suitable for evaluating images that preserve overall structure while differing in texture or color, unlike traditional metrics such as PSNR and SSIM.

DISTS [Ding et al. (2022)] is often used alongside LPIPS for content assessment. Built on a pretrained VGG network with aggregation weights optimized on image quality assessment datasets including KADID [Lin et al. (2019)], DISTS computes distance by comparing both structure and texture statistics of feature maps between

two images:

$$S_l(x,y) = \frac{2\mu_x\mu_y + \epsilon}{\mu_x^2 + \mu_y^2 + \epsilon}, \tag{2.10}$$

$$T_l(x,y) = \frac{2\sigma_{xy} + \epsilon}{\sigma_x^2 + \sigma_y^2 + \epsilon}, \tag{2.11}$$

$$\text{DISTS}(x,y) = \sum_l w_l(S_l(x,y) + T_l(x,y)), \tag{2.12}$$

where S_l and T_l represent structure and texture similarity at layer l by comparing the means μ_x, μ_y, variances σ_x, σ_y, and covariance σ_{xy} of corresponding feature maps (ϵ is added to avoid instability). The metric rewards texture and structure preservation while remaining insensitive to pixel-wise differences. Lower DISTS values indicate images share similar content.

Style

DreamSim [Fu et al. (2023)] is a perceptual metric designed to capture mid-level visual similarity between two images, as opposed to LPIPS and DISTS which focus on low-level image features. Low-level features correspond to early network layers and capture local patterns such as edges, textures, and colors; mid-level features from intermediate layers encode object parts, spatial relationships, and scene layout. The practical implication is that two images of the same scene rendered in different artistic styles may score poorly on LPIPS (large low-level difference) but well on DreamSim (similar scene structure and object layout), which is closer to how humans judge overall similarity. The method fine-tunes several foundation models (DINO, CLIP, OpenCLIP) on the NIGHTS dataset of human similarity judgments, computing distance from their feature representations. DreamSim demonstrates superior alignment with human perception on both custom and established benchmarks, including BAPPS.

The Fréchet Inception Distance (FID) [Heusel et al. (2017)] measures similarity between two image distributions rather than individual image pairs. To compute FID, both datasets are passed through InceptionV3, a CNN trained for ImageNet classification. Its penultimate-layer features (a 2048-dimensional vector) have been found to capture perceptually meaningful statistics of natural images. The Fréchet distance is calculated between the resulting feature distributions:

$$\text{FID} = ||\mu_x - \mu_y||^2 + \text{Tr}(\Sigma_x + \Sigma_y - 2(\Sigma_x\Sigma_y)^{1/2}), \tag{2.13}$$

where μ_x, μ_y are mean feature vectors of the two datasets and Σ_x, Σ_y the corresponding sample covariance matrices. Despite requiring large datasets (typically 50,000 images in the original work), FID remains the standard metric for evaluating generative models. The use of a fixed, pretrained classifier as a feature extractor is a choice worth noting: FID's behavior depends on how well InceptionV3 features align with human perception of image quality.

Finally, CMMD [Jayasumana et al. (2024)] addresses FID's limitations, particularly its large sample requirements and assumption of Gaussian distributions. CMMD replaces Inception embeddings with CLIP features and substitutes the

Fréchet distance with maximum mean discrepancy:

$$\widehat{dist}^2_{MMD}(X,Y) = \frac{1}{m(m-1)}\sum_{i=1}^{m}\sum_{j\neq i}^{m} k(\mathbf{x}_i,\mathbf{x}_j) \tag{2.14}$$

$$+\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j\neq i}^{n} k(\mathbf{y}_i,\mathbf{y}_j) \tag{2.15}$$

$$-\frac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} k(\mathbf{x}_i,\mathbf{y}_j) \tag{2.16}$$

where:

- $X = \{\mathbf{x}_1,\ldots,\mathbf{x}_m\}$ and $Y = \{\mathbf{y}_1,\ldots,\mathbf{y}_n\}$ are two vectors of dataset samples embeddings,
- m and n are the number of samples in sets X and Y respectively,
- $k(\mathbf{x},\mathbf{y})$ is the Gaussian radial basis function kernel.

These modifications enable reliable distance estimation from as few as 1,000 samples, improving robustness for smaller datasets.

2.4.2 Object recognition performance

Detection metrics

Object detection evaluation compares predicted bounding boxes against ground truth annotations. A detection is considered correct if it sufficiently overlaps with a ground truth box, measured by Intersection over Union (IoU). IoU computes the ratio between the intersection and union areas of two bounding boxes B_{pred} and B_{gt}, ranging from 0 (no overlap) to 1 (perfect match):

$$\mathrm{IoU}(B_{\mathrm{pred}}, B_{\mathrm{gt}}) = \frac{\mathrm{Area}(B_{\mathrm{pred}} \cap B_{\mathrm{gt}})}{\mathrm{Area}(B_{\mathrm{pred}} \cup B_{\mathrm{gt}})}. \tag{2.17}$$

A prediction is classified as a True Positive (TP) if its IoU exceeds a predefined threshold, usually 0.5. A False Positive (FP) is a detected bounding box that does not actually contain the predicted object. On the contrary, a False Negative (FN) is a missed detection.

Based on these predictions, precision and recall are computed. Precision measures the proportion of correct predictions among all model predictions:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \tag{2.18}$$

Recall measures the proportion of ground truth objects successfully detected:

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}. \tag{2.19}$$

High precision indicates few false alarms while high recall indicates few missed detections. These metrics exhibit a trade-off, as lowering the confidence threshold increases recall but decreases precision.

Average Precision (AP) summarizes this trade-off by computing the area under the precision-recall curve for a single class. Mean Average Precision (mAP) then averages AP across all classes to provide an overall detection performance measure. The choice of IoU threshold significantly impacts evaluation strictness. The standard AP50 uses a 0.5 IoU threshold, while the stricter AP75 requires 0.75 IoU. The COCO [Lin et al. (2014)] metric AP@[0.5:0.95], commonly referred to as MS-COCO AP or AP, averages AP across IoU thresholds from 0.5 to 0.95 in 0.05 increments, providing more balanced assessment by penalizing imprecise localizations while being less sensitive to arbitrary threshold choice.

Segmentation metrics

Semantic segmentation is primarily evaluated using mean Intersection over Union (mIoU). Similar to detection, IoU measures spatial overlap, but here it computes the ratio between the intersection and union of predicted and ground truth pixels for each class. The mIoU averages these per-class IoU scores, providing an overall measure of segmentation quality. Unlike pixel accuracy which can be misleading with class imbalance, mIoU treats all classes equally regardless of their frequency, making it sensitive to performance on rare classes.

Domain adaptation evaluation

Evaluating domain adaptation methods requires assessing the model's ability to generalize across the domain gap. The standard protocol trains models on the source domain (e.g. synthetic data) and evaluates exclusively on the target domain (e.g. real data), directly measuring generalization performance. Some works additionally report source domain performance to detect overfitting and verify that adaptation does not degrade source performance. This double evaluation is relevant when the adapted model may be deployed in both domains.

2.5 Datasets

This section reviews the datasets relevant to this thesis, organized into three categories: generic datasets used primarily for pretraining model backbones, driving datasets that serve as training and evaluation benchmarks for domain adaptation, and synthetic datasets used as source domains. Driving datasets receive the most attention, as urban scene understanding under varying conditions is extensively studied in this work.

Generic datasets

ImageNet [Russakovsky et al. (2015)] is the first large-scale image classification dataset. Introduced in 2009 by Deng et al., it originally comprised 1.2 million images across 1,000 classes for the ILSVRC challenge, which started in 2010. ImageNet is widely used for pretraining model backbones due to its broad coverage of object categories and natural image diversity. The full ImageNet-21K version contains about 14 million images spanning over 21,000 categories. This dataset has been a cornerstone of data-based computer vision.

The PASCAL VOC dataset [Everingham et al. (2010)] introduced the first standardized object detection benchmark. Initially released in 2005 with 1,578 images and 4 classes, it included bounding boxes for object detection. The final version, PASCAL VOC 2012, contains 11,500 images and adds semantic segmentation annotations with 20 object classes and one background class. The dataset focuses on everyday objects including people, animals, vehicles, and indoor items.

MS-COCO [Lin et al. (2014)], released in 2014, extends PASCAL VOC scope with 200,000 images covering 80 object categories. This dataset provides bounding boxes, semantic segmentation, and instance segmentation, describing each object individually within an image. MS-COCO emphasizes objects in context, with images containing multiple objects and complex scenes. The dataset also includes captions for each image, making it valuable for vision-language tasks.

Driving datasets

The KITTI vision benchmark suite [Geiger et al. (2012)] is the first dataset focusing on urban scenes for autonomous driving. This dataset was captured in Karlsruhe, Germany, using cameras and LiDAR sensors mounted on a vehicle. LiDAR (Light Detection And Ranging) is an active sensing technology that measures distance by emitting laser pulses and recording their return times, producing 3D point clouds of the surrounding environment. It complements cameras by providing precise depth measurements that passive imagery cannot. KITTI includes benchmarks for 3D object detection and tracking among others. The dataset comprises approximately 15,000 frames with 3D bounding box annotations for vehicles, pedestrians, and cyclists.

The Cityscapes dataset [Cordts et al. (2016)] contains approximately 5,000 high-resolution images of urban scenes with pixel-level semantic annotations covering 19 classes. Images were captured across 50 cities, predominantly in Germany, during spring, summer, and fall. The dataset also includes 20,000 images with coarse annotations. Thanks to its high-quality labels and relative object diversity, Cityscapes serves as the reference benchmark for autonomous driving and semantic segmentation tasks.

Mapillary Vistas [Neuhold et al. (2017)] aims to increase geographic and weather diversity for urban scene understanding, featuring 25,000 images captured across multiple continents using crowdsourced street-level imagery. The dataset originally includes 66 object classes with pixel-accurate annotations, covering a wider variety of geographic locations, seasons, and weather conditions than Cityscapes.

Moreover, BDD100K [Yu et al. (2020)] provides one of the largest and most diverse driving datasets with 100,000 video sequences recorded across different times of day, weather conditions, and locations in the United States. It includes annotations for various tasks, from object detection to lane marking recognition and semantic segmentation. The full-frame segmentation subset comprises 10,000 images and 19 classes.

Finally, ACDC [Sakaridis et al. (2021)] focuses specifically on adverse weather conditions. This dataset uniquely provides image pairs captured under both clear weather and challenging environmental conditions at the same locations. ACDC features four condition types (night, rain, fog, and snow), equally distributed across 4,000 images with pixel-level semantic annotations using the same 19 classes as Cityscapes. The paired clear/adverse images make ACDC particularly suited for

domain adaptation evaluation: a model trained on clear or synthetic images can be tested directly on the paired adverse-condition images, providing a controlled measure of robustness. ACDC is the primary real-data evaluation benchmark used throughout this thesis, as its four conditions cover the adverse scenarios of interest for robust perception systems.

Synthetic datasets

Synthetic datasets also exist for urban scene understanding. SYNTHIA [Ros et al. (2016)] proposes a virtual dataset of 200,000 images from a simulated European-style city, covering 11 semantic classes for segmentation tasks. The dataset includes multiple viewpoints, seasons, and weather conditions, with automatically generated ground truth annotations.

Playing for Data [Richter et al. (2016)] comprises 24,966 densely annotated images extracted from the Grand Theft Auto V (GTA5) video game. All annotations are compatible with Cityscapes classes (19 semantic categories), making it particularly suitable for domain adaptation benchmarks. The images capture several weather conditions, times of day, and urban environments within the game world. Despite its widespread use in synthetic-to-real domain adaptation research, this dataset lacks metadata for precise weather conditions, which limits its applicability in adverse condition scenarios requiring specific environmental control.

Chapter 3

Domain adaptation using diffusion-based style transfer

Semantic segmentation models trained on synthetic data often perform poorly on real-world images due to the domain gap, particularly in adverse conditions where labeled data is scarce. Diffusion-based foundation models can generate realistic images without task-specific training, presenting an opportunity to bridge this gap through style transfer. This chapter introduces two techniques for semantically consistent style transfer: Class-wise Adaptive Instance Normalization and Cross-Attention (CACTI) and its extension with selective attention Filtering (CACTI_F). CACTI applies statistical normalization selectively based on semantic classes, while CACTI_F further filters cross-attention maps based on feature similarity, preventing artifacts in regions with weak correspondences. Unlike approaches that apply global transformations or generate content without structural constraints, our methods preserve semantic boundaries and structural coherence. Experiments using GTA5 as source and Cityscapes/ACDC as target domains show that our approach produces higher quality images with lower FID scores and better content preservation. Combined with domain adaptation architectures, class-aware style transfer effectively bridges the synthetic-to-real domain gap even with minimal target domain data, improving segmentation performance across all adverse conditions. However, selecting the most appropriate style transfer operators and their configuration remains a manual and time-consuming process, motivating the automated approach presented in the next chapter.

This chapter covers the article *Style Transfer with Diffusion Models for Synthetic-to-Real Domain Adaptation* published in *Computer Vision and Image Understanding, September 2025* [Chigot et al. (2025b)].

3.1 Motivation

Among pixel-space domain adaptation methods, style transfer is a promising approach to bridge the gap between the synthetic and real domains. By transferring the visual characteristics of target domain images to source domain content, these techniques can generate training data that maintains the accurate labels of the source while exhibiting the appearance of the target. Traditional style transfer methods have focused primarily on artistic applications, with limited exploration of their potential for domain adaptation in semantic segmentation tasks [Alaluf et al. (2024); Go et al. (2026); Hertz et al. (2024)]. Recent advances in diffusion models have greatly improved image generation capabilities, demonstrating unprecedented quality and flexibility in content creation [Huang et al. (2025)]. These models, which generate images through an iterative denoising process, offer new mechanisms for style transfer that could potentially address the synthetic-to-real domain gap more effectively than previous approaches. However, applying diffusion-based style transfer specifically for domain adaptation presents unique challenges that remain largely unexplored.

In this chapter, we investigate diffusion-based style transfer for synthetic-to-real domain adaptation in semantic segmentation, with a particular focus on few-shot scenarios where only limited target domain data is available. We identify two key limitations in existing approaches: (1) standard style transfer methods apply global transformations that ignore semantic class boundaries, leading to unrealistic appearances within object categories, and (2) direct application of cross-attention mechanisms can create artifacts when structural correspondence between domains is weak.

To address these challenges, we propose two novel techniques: First, **C**lass-wise **A**daptive Instance Normalization and **C**ross-atten**TI**on (CACTI) applies statistical normalization selectively based on semantic classes, ensuring that style characteristics are transferred appropriately within semantic boundaries. Second, CACTI with Selective Attention **F**iltering (CACTI_F) extends this approach by selectively applying cross-attention based on feature similarity, preventing artifacts in regions where direct style transfer might produce inconsistencies.

Our experiments demonstrate that these techniques not only produce visually coherent images but also improve semantic segmentation performance when used for domain adaptation. Particularly noteworthy is their effectiveness in challenging adverse weather conditions such as fog, rain, snow, and nighttime, where the domain gap is especially pronounced. When combined with state-of-the-art segmentation architectures, our approach improves over existing methods, highlighting the potential of semantically-aware diffusion-based style transfer for addressing the synthetic-to-real domain gap.

3.2 Related work

3.2.1 Style transfer with diffusion models

Diffusion models generate images through an iterative denoising process that progressively refines Gaussian noise into coherent visual outputs. As detailed in Section 2.2.3, recent advancements such as DDPM [Ho et al. (2020)] and DALL·E 2 [Ramesh et al. (2022)] have demonstrated the ability to produce images of high quality and diversity.

Stable Diffusion (SD) [Rombach et al. (2022)] further improves efficiency by performing the denoising process in the latent space of a Variational Autoencoder (VAE), using a UNet-based architecture. This design significantly reduces computational cost while maintaining high fidelity in generated images. Building on this foundation, ControlNet [Zhang et al. (2023a)] introduces a conditioning mechanism that enables control over image generation using inputs such as human poses or segmentation maps. By incorporating zero-convolution layers during training, ControlNet aligns conditional inputs with the denoising process, allowing for fine-grained manipulation of image attributes.

A subclass of image generation methods focuses on transferring style characteristics from a reference image to a target content image—a process known as style transfer. Early approaches, such as the method proposed by Gatys et al. [Gatys et al. (2016)], utilized the features of a pretrained CNN classifier to design style and content losses that enable style transfer with optimization. To improve flexibility and efficiency, Adaptive Instance Normalization (AdaIN) [Huang and Belongie (2017)] was introduced. AdaIN aligns the channel-wise mean and variance of the style features with those of the content image in the latent space of a CNN, enabling arbitrary style transfer in real-time with a single forward pass. In parallel, Generative Adversarial Networks (GANs) [Goodfellow et al. (2014)] have also been widely adopted for style transfer tasks [Zhu et al. (2017); Chen et al. (2019); Lin et al. (2021)], at the expense of training stability and with a risk of mode collapse.

Developments in diffusion models have made them suitable to perform style transfer tasks. Several methods leverage text-based algorithms to stylize images by prompting a pretrained diffusion model without the need for finetuning. These approaches use textual descriptions to guide the generation process and achieve style transfer through prompt engineering and latent manipulation [Zhang et al. (2023b); Qi et al. (2024); Jeong et al. (2024); Hertz et al. (2024)]. Beyond text-based guidance, other techniques enable style transfer by directly blending features from two input images during inference. Cross-image attention [Alaluf et al. (2024)], Z* [Deng et al. (2024)] and StyleID [Chung et al. (2024)] use KV injection in the self-attention layers of Stable-Diffusion's UNet. Eye-for-an-eye [Go et al. (2026)] rearranges the features in those self-attention layers following a computed semantic correspondence. Additionally, some recent works incorporate Low-Rank Adaptation (LoRA) [Hu et al. (2022)], a parameter-efficient finetuning technique, to enable more flexible style-content fusion. These methods, such as B-LoRA [Frenkel et al. (2024)] and ZipLoRA [Shah et al. (2024)], fine-tune lightweight layers to blend styles from multiple reference images.

Those methods are generally used to generate artistic images, and only few of them are actually employed as data generation pipelines for other vision tasks. However, due to their large training datasets and generalization capabilities, diffusion based style transfer methods seem appropriate to bridge the synthetic to real domain gap.

3.2.2 Synthetic to real domain adaptation

As discussed in Chapter 1, synthetic to real domain adaptation, or Sim-to-real domain adaptation, refers to the attempt of reducing the gap between training on simulated data and testing on real-life data. This topic is particularly relevant when

data is scarce, expensive to annotate and scenes are complex such as in robotics or autonomous driving. Classical domain adaptation techniques typically rely on access to large, labeled datasets in both source and target domains. However, more recent approaches focus on one-shot or zero-shot adaptation, using limited to no target domain data. In the context of semantic segmentation, DAFormer [Hoyer et al. (2022a)] introduces a transformer-based architecture that leverages strong feature encoding and pseudo-label refinement, achieving state-of-the-art performance on synthetic-to-real benchmarks. Extending this work, HRDA [Hoyer et al. (2022b)] enhances performance by incorporating high-resolution feature fusion at multiple scales, allowing the model to better capture fine-grained details and spatial consistency across domains. PØDA [Fahes et al. (2023)] uses CLIP to integrate prompts embeddings to align the source domain and the target domain, without having access to target annotations.

Leveraging the generation capacity of diffusion models, some methods have integrated those into domain adaptation pipelines. SGG [Peng et al. (2023)] uses stable diffusion to generate intermediate domains during the segmentation training mechanism. Gong et al. [Gong et al. (2024)] plug a segmentation head onto a diffusion backbone, making use of the disentangled representations of style and content learned by diffusion models.

DGInStyle [Jia et al. (2024)] finetunes Dreambooth [Ruiz et al. (2023)] on the source domain, then trains ControlNet to generate images conditioned by segmentation maps. Finally, they apply a style swap operation, switching the source specific UNet to a generalist pretrained one. In this manner they only retain the segmentation conditioning without style leaking from the source dataset during the ControlNet training. DATUM [Benigmim et al. (2023)] also uses Dreambooth to finetune stable diffusion on only one target image for a few iterations. Then, they prompt the model to generate one target object in the style of the target image. Those two generation methods can be used to generate datasets tailored to the domain adaptation use-case, in a zero-shot setting for DGInStyle or one-shot setting for DATUM.

Diffusion based style transfer has not yet been used commonly to generate datasets for domain adaptation. DoGE [Wang et al. (2024e)] computes the mean difference of CLIP embeddings pairs between the source and target dataset (Domain Gap Embedding). Then, it adds this Domain Gap Embedding to the latent representation of a source image, before reconstructing the image with Stable UnCLIP. This method can use ControlNet, allowing the model to generate datasets conditioned on semantic segmentation maps. In this work, we propose to use Cross-image attention [Alaluf et al. (2024)] as our style-transfer method to generate datasets for its zero-shot generation capability.

3.3 Methodology

Existing approaches present several limitations. First, using synthetic data grants access to automatic labels, but standard style transfer methods do not exploit semantic segmentation maps. Also, artifacts often appear when content and style images are not semantically aligned i.e., both images do not contain the same classes, or with a strong imbalance. These artifacts can negatively impact model training if object boundaries change or object classes become unclear.

We introduce two novel techniques to address these limitations: Class-wise

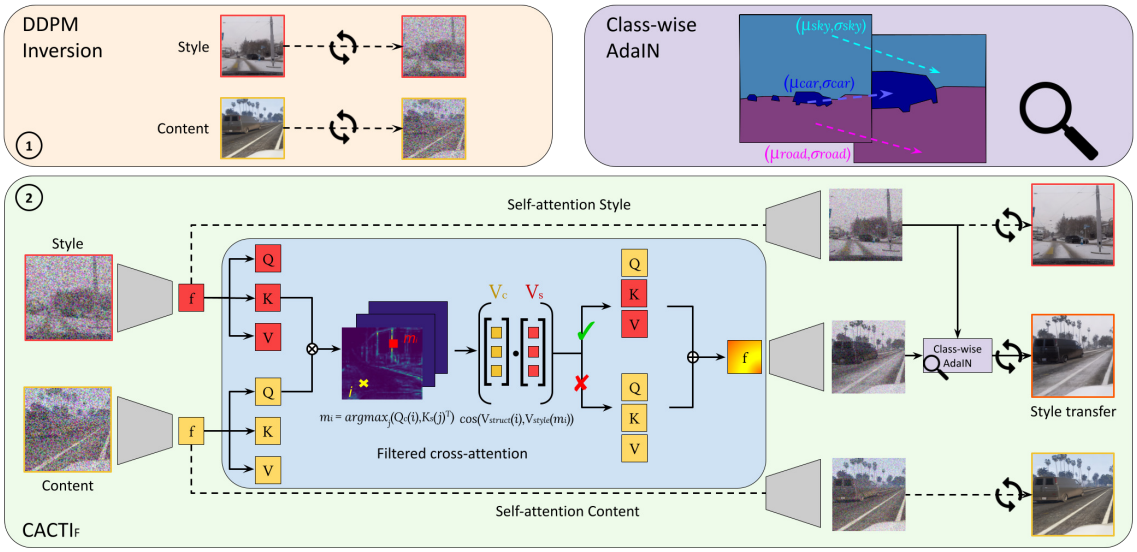


Figure 3.1: Overview of our proposed method for style transfer with diffusion models. Given a content image and a style image, we first perform DDPM inversion (left) to obtain their respective latent representations. These representations are then processed through our cross-attention mechanism, where our semantic filtering approach (middle) determines when to apply cross-image attention. When the KV injection yields coherent feature correspondences, cross-image attention transfers style features from the style image to the content image; otherwise, the original content features are preserved. Finally, our class-wise AdaIN module (right) refines the style transfer by computing class-specific statistics using segmentation masks from both source and content images.

Adaptive Instance Normalization with Attention (CACTI), possibly combined with Selective Attention Filtering (CACTI$_{\text{F}}$).

3.3.1 Class-wise adaptive instance normalization

In its standard form, AdaIN computes global statistics across entire feature maps in order to align them between images. This global calculation can be problematic when dealing with domain adaptation scenarios where class-specific appearances are critical for downstream tasks such as semantic segmentation.

As stated in Chapter 2, AdaIN is a common brick in style transfer methods. It is used as a way to match feature statistics between two images. It can be used in a supervised manner by training a feature extractor from a dataset of paired images, or in a zero-shot setting with a pre-trained auto-encoder. We use the latter, within the VAE of a latent diffusion model.

As in several applications of AdaIN with diffusion models [Go et al. (2026); Alaluf et al. (2024)], at each step t of the denoising process we apply the AdaIN operation between the features of the style image z_t^{style} and transferred image z_t^{out}:

$$z_t^{out} \leftarrow \text{AdaIN}(z_t^{out}, z_t^{style}). \tag{3.1}$$

This operation makes the features statistics from z_t^{out} match those of z_t^{style}, effectively transferring global color distributions and contrasts.

In our proposed CACTI method, we leverage the semantic segmentation masks available for both the synthetic content images and real style images to perform a class-specific statistical alignment. Rather than computing global statistics, we compute and apply statistics separately for each semantic class. This approach prevents dominant classes in the style image (such as sky or vegetation) from inappropriately influencing the appearance of other classes.

To implement CACTI, we first resize the segmentation masks to match the latent dimensions of Stable Diffusion's VAE. Then, for each semantic class c present in both images, we compute:

$$\mu_c(z) = \frac{\sum_{i,j} M_c(i,j) z(i,j)}{\sum_{i,j} M_c(i,j)}, \tag{3.2}$$

$$\sigma_c(z) = \sqrt{\frac{\sum_{i,j} M_c(i,j)(z(i,j) - \mu_c(z))^2}{\sum_{i,j} M_c(i,j)}}, \tag{3.3}$$

where M_c is a binary mask indicating pixels belonging to class c and (i,j) are spatial coordinates. We then assemble full maps of the statistics μ and σ, across all classes:

$$\mu(z) = \sum_c M_c \mu_c(z), \tag{3.4}$$

$$\sigma(z) = \sum_c M_c \sigma_c(z). \tag{3.5}$$

We finally apply the class-wise AdaIN operation using the aggregated class statistics μ and σ:

$$z_t^{out} \leftarrow \sigma(z_t^{style}) \left(\frac{z_t^{out} - \mu(z_t^{out})}{\sigma(z_t^{out})} \right) + \mu(z_t^{style}). \tag{3.6}$$

This class-specific approach ensures that statistical features from the style image influence only corresponding semantic regions in the content image. For example, the appearance of a "road" class in the style image affects only road pixels in the output image, preserving more realistic color distributions and reducing cross-class artifacts that can occur with global AdaIN. This is particularly important for domain adaptation scenarios where semantic consistency is crucial for downstream tasks.

3.3.2 Selective attention filtering

Several methods use a mechanism of cross-attention, or *KV injection* for style transfer, as a means of mixing feature information from a content and style image together [Alaluf et al. (2024); Deng et al. (2024); Chung et al. (2024)], leveraging the self-attention layers of the denoising UNet. When fed to a self-attention layer, a feature map F is linearly projected into query Q, key K and value V. Those projections, of dimension d, enable learning the interactions between different spatial positions of the feature maps. The standard self-attention operation processes these Q, K and V projections as follows:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) V, \tag{3.7}$$



Figure 3.2: When transferring style from an image with enough vegetation, greenery often leaks on other objects. On the right images, you can observe unwanted foliage on buildings and roads.

where the softmax is applied independently on all rows of QK^T. In cross-image attention [Alaluf et al. (2024)], on which we base our method, the attention is computed using the content query Q_c and the style key K_s and value V_s, hence the term *KV injection*. Their hypothesis is that Q encapsulates the content within an image, while K and V represent the style information.

The style transfer of cross-image attention therefore modifies the standard attention calculation as such:

$$\text{Attention}(Q_c, K_s, V_s) = \text{softmax}\left(\frac{Q_c K_s^T}{\sqrt{d}}\right) V_s. \tag{3.8}$$

Cross-image attention establishes correspondences between semantically similar regions of source and target images, regardless of their spatial positions. However, in the context of synthetic-to-real domain adaptation, this approach faces a significant challenge. The cross-attention maps can be diffuse, with a single query in the source image attending to multiple semantically similar regions in the target image. This diffusion effect can lead to inconsistent style transfer and diminished adaptation performance. Figure 3.2 displays such style transfer inconsistencies using cross-image attention [Alaluf et al. (2024)].

To address this limitation, we propose cross-attention filtering, CACTI_F, as an extension of CACTI. This filtering refines the cross-image attention mechanism by keeping only the style features whose value vectors are similar to the one at the maximum attention position, reducing noise and ensuring more consistent and stable style transfer.

In the cross-attention mechanism, at each diffusion step and self-attention layer, the feature map is projected into queries (Q), keys (K), and values (V). The cross-attention maps $\mathcal{A}_{c,s} \in \mathbb{R}^{(h\times w)^2}$ are computed as:

$$\mathcal{A}_{c,s} = Q_c K_s^T, \tag{3.9}$$

where Q_c represents queries from the content image, K_s keys from the style image, h and w being the height and width of the self-attention layer. These attention maps indicate, for each position in the content features, which regions in the style features are most relevant.

In our proposed CACTI$_{\text{F}}$ method, we selectively filter these attention maps based on feature similarity. For each position i in the content image, we first identify the position m_i across all pixels j in the style image that receives maximum attention:

$$m_i = \arg\max_j (Q_c(i) \cdot K_s(j)^T). \tag{3.10}$$

After identifying these maximum-attention correspondences, we evaluate whether the style transfer at each position is likely to produce coherent results. To this end, we compute the scalar cosine similarity $s(i) \in \mathbb{R}$ between the value vectors at corresponding positions:

$$s(i) = \cos\left(V_{\text{c}}(i), V_{\text{s}}(m_i)\right) = \frac{V_{\text{c}}(i) \cdot V_{\text{s}}(m_i)}{\|V_{\text{c}}(i)\| \|V_{\text{s}}(m_i)\|}, \tag{3.11}$$

where $V_{\text{c}}(i)$ is the value vector at position i in the content image and $V_{\text{s}}(m_i)$ is the value vector at the corresponding position of maximum attention in the style image.

We then compute a threshold τ based on a percentile p of the similarity distribution:

$$\tau = \text{Percentile}(\{s(i)\}, p). \tag{3.12}$$

For positions where the similarity falls below this threshold, indicating potential inconsistencies in the transfer, we retain the original content features rather than applying style transfer:

$$\mathcal{A}_{\text{out}}(i) = \begin{cases} \mathcal{A}_{c,s}(i) & \text{if } s(i) \geq \tau \\ \mathcal{A}_{\text{c}}(i) & \text{if } s(i) < \tau \end{cases}, \tag{3.13}$$

$$V_{\text{out}}(i) = \begin{cases} V_{\text{s}}(i) & \text{if } s(i) \geq \tau \\ V_{\text{c}}(i) & \text{if } s(i) < \tau \end{cases}. \tag{3.14}$$

This selective approach preserves content structure in regions where direct style transfer might produce artifacts. The parameter p controls the strictness of the filtering, with higher values resulting in more conservative style transfer. Through empirical testing, we determined that values of p between 0.1 and 0.3 provide a good balance between style transfer effectiveness and content preservation.

3.3.3 Combined approach

Our complete methodology, CACTI$_{\text{F}}$, combines both class-wise AdaIN and attention filtering in a complementary fashion. The class-wise AdaIN ensures appropriate appearance transfer within semantic regions, while attention filtering prevents artifacts in areas where the cross-attention correspondence between content and style is weak.

In practice, we implement our approach within the Stable Diffusion framework, applying these techniques during the denoising process. As shown in Figure 3.1, at each diffusion step:

- We compute the cross-attention maps between content queries and style keys;
- We apply attention filtering based on value vector similarity;
- We perform the modified cross-attention operation;

- We apply class-wise AdaIN to refine feature statistics.

We empirically find that applying these operations at multiple resolution levels in the UNet decoder (32 × 64 and 64 × 128 feature maps) yields the best results.

To evaluate the effectiveness of our proposed techniques, we conduct two complementary studies. First, we perform a qualitative analysis of the style transfer results, examining how CACTI and CACTI_F reduce artifacts and improve visual coherence compared to existing methods. This analysis helps isolate the specific contributions of class-wise AdaIN and attention filtering to the overall quality of the generated images.

Second, we conduct a quantitative evaluation in the context of synthetic-to-real domain adaptation for semantic segmentation. By testing our generated images as training data for segmentation models under various adverse weather conditions, we demonstrate the practical utility of our approach for downstream computer vision tasks. This evaluation framework allows us to assess not only the visual quality of our style transfer but also its effectiveness in bridging the domain gap between synthetic and real data distributions.

Together, these two experiments highlight how class-based AdaIN improves color consistency and class-specific appearance transfer, while attention filtering preserves structural integrity and reduces artifacts. The following sections detail our experimental setup and results for both evaluations.

3.4 Content-preserving style transfer

The quality of synthetically generated images is crucial for effective domain adaptation in semantic segmentation tasks. In this section, we evaluate how our proposed methods, CACTI and CACTI_F, improve image quality compared to existing approaches. We specifically focus on reducing diffusion artifacts and enhancing alignment between segmentation masks and the corresponding generated images, which is essential for maintaining label consistency when transferring style from real to synthetic domains.

3.4.1 Datasets

We evaluate our methods using widely adopted datasets in the domain adaptation literature, focusing on driving scenes with semantic segmentation annotations.

GTA5 [Richter et al. (2016)] serves as our synthetic source domain. This dataset comprises 24,966 densely annotated images rendered from the Grand Theft Auto V video game environment, with a resolution of 1914 × 1052. The synthetic nature of this dataset provides perfect ground truth labels but exhibits the characteristic domain gap when used to train models for real-world deployment.

Cityscapes [Cordts et al. (2016)] represents our primary real-world target domain. It consists of 5,000 densely labeled images captured in 50 European cities during daytime and under good weather conditions. These images have a resolution of 2048 × 1024 and feature typical urban driving scenarios.

In our experimental framework, we address the synthetic-to-real domain adaptation scenario, using GTA5 as the source domain and either Cityscapes or ACDC as

the target domain. For style transfer, we adopt a few-shot setting where we assume access to a single representative image from each target domain condition, which aligns with realistic constraints in practical applications where target domain data may be scarce.

3.4.2 Experimental setup

We compare our proposed methods against several state-of-the-art approaches for style transfer and domain adaptation:

AdaIN diffusion represents the baseline approach where standard Adaptive Instance Normalization is applied at each step of the diffusion process without class-specific modifications or cross-attention mechanisms.

Cross-Image Attention implements the method from [Alaluf et al. (2024)], which uses cross-image attention with standard AdaIN to transfer appearance between images while preserving content structure.

DATUM [Benigmim et al. (2023)] employs personalized diffusion models for one-shot unsupervised domain adaptation, fine-tuning a text-to-image diffusion model on a single target sample to generate target-like images.

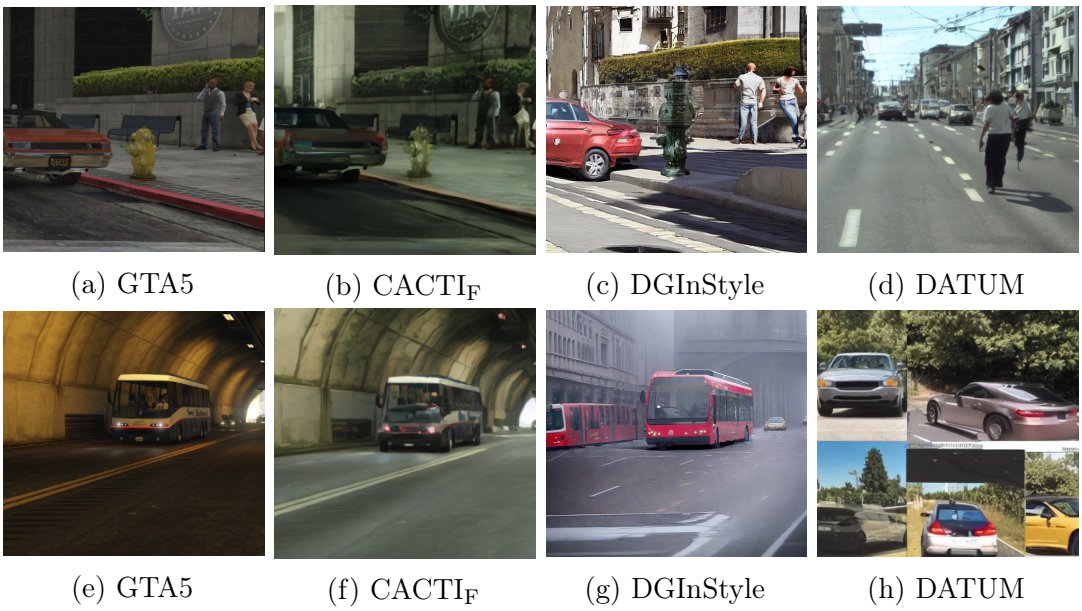
DGInStyle [Jia et al. (2024)] generates images in the target domain style without a content reference image, operating primarily as a generative model rather than a style transfer approach.

CACTI (our method) enhances Cross-Image Attention with class-specific AdaIN, leveraging segmentation labels to guide the feature statistic matching during the diffusion process.

CACTI$_\mathbf{F}$ (our method) extends CACTI by adding selective attention filtering based on feature similarity, further reducing artifacts in regions where direct style transfer might produce inconsistencies.

For all methods based on cross-image attention (Cross-Image Attention, CACTI, and CACTI$_\mathrm{F}$), we crop and resize all input images to a 1024×512 resolution. We found this size relevant to harmonize resolution between datasets, as well as keeping a good balance between generation time and image quality. We perform the transfer at the same resolution using Stable Diffusion v1.5 with an empty prompt. For the percentile threshold in attention filtering, we set $p = 0.25$ as our default setting based on preliminary experiments. To further balance generation quality with computational efficiency, we skip 30 out of 50 denoising steps. For AdaIN-based methods, we apply the normalization at every step throughout the diffusion process.

While we first measure the capacity of these methods to transfer styles, we note that DATUM and DGInStyle do not fully preserve content through segmentation mask guidance. We expect these methods to generate images showing artifacts and undefined objects. By using style transfer and not image generation without prior, we maintain strong structural coherence.



(a) GTA5 (b) CACTI$_F$ (c) DGInStyle (d) DATUM

(e) GTA5 (f) CACTI$_F$ (g) DGInStyle (h) DATUM

Figure 3.3: Qualitative comparison between dataset generation methods, with style transfer (ours) or generation without base image. Using the GTA5 dataset as base allows to keep object consistency and image structure. Note that DATUM does not rely on semantic segmentation masks and can therefore generate an image with multiple scenes, as shown here.

3.4.3 Evaluation metrics

We first employ two complementary metrics, introduced in Chapter 2, to evaluate the quality of generated images, studying the style transfer capacity of the proposed methods.

The Fréchet Inception Distance (FID) [Heusel et al. (2017)] measures the statistical similarity between generated images and real target domain images. Lower FID values indicate better alignment with the target domain distribution, reflecting successful style transfer. For FID computation, we generate 5,000 images per method and compare against the Cityscapes dataset.

The Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al. (2018)] quantifies the perceptual distance between pairs of images, providing a measure of content preservation. We compute LPIPS between each synthetic GTA5 image and its corresponding generated version. Lower LPIPS values indicate better preservation of the original content structure.

3.4.4 Results and analysis

Figure 3.3 provides a visual comparison between different methods. The limitations of approaches that do not use a content reference image become immediately apparent. Methods like DGInStyle and DATUM exhibit notable artifacts and poor spatial coherence, with objects appearing distorted (as seen with the red car in Figure 3.3c) or showing unrealistic spatial organization (Figure 3.3h).

In contrast, our proposed methods maintain object localization and structural integrity while successfully transferring the target domain style. This preservation of spatial relationships is crucial for downstream semantic segmentation tasks, as it

Method	$\text{FID}_{\text{cityscapes}} \downarrow$	$\text{LPIPS}_{\text{GTA5}} \downarrow$
AdaIN diffusion	73.26	**0.30**
Cross-image Attention	65.31	0.41
DATUM	113.12	-
DGInStyle	86.94	-
CACTI	**54.56**	0.39
CACTI_{F}	55.30	0.36

Table 3.1: Quantitative evaluation of style transfer quality. $\text{FID}_{\text{cityscapes}}$ measures similarity to the target domain (lower is better), while $\text{LPIPS}_{\text{GTA5}}$ measures content preservation (lower is better). LPIPS is not calculated for methods that do not use content reference images.

ensures alignment between generated images and their corresponding labels.

The quantitative results in Table 3.1 confirm the visual observations. Our CACTI method achieves the lowest FID score (54.56) compared to all other approaches, indicating superior style transfer quality and closest resemblance to the real Cityscapes distribution. The CACTI_{F} variant, which adds attention filtering, shows a slight increase in FID (55.30) but significantly improves content preservation as measured by LPIPS (0.36 compared to 0.39 for CACTI).

Standard Cross-Image Attention shows reasonable style transfer capabilities (FID = 65.31) but struggles with content preservation (LPIPS = 0.41). Basic AdaIN diffusion achieves the best content preservation (LPIPS = 0.30) but at the expense of style transfer quality (FID = 73.26). We note that there is no cross-image attention in this baseline, explaining the good content preservation as only AdaIN is used for transfer between the images.

We do not calculate LPIPS for methods that do not use a content reference image, DATUM and DGInStyle, as LPIPS is calculated over image pairs. However, we note that DGInStyle shows poor performance on FID (86.94), highlighting the challenge of generating domain-specific images without structural guidance.

These results demonstrate a fundamental trade-off between style transfer fidelity and content preservation. Our proposed methods, particularly CACTI_{F}, achieve a favorable balance between these objectives, generating images that both resemble the target domain style and maintain the structural information necessary for semantic segmentation.

The class-specific approach in CACTI prevents dominant classes in the style image (such as sky or road) from inappropriately influencing the appearance of other classes, resulting in more realistic color distributions. The selective attention mechanism in CACTI_{F} further enhances content preservation by identifying and filtering potentially problematic attention mappings.

These improvements in image quality and content preservation directly impact the effectiveness of the generated datasets for domain adaptation in semantic segmentation tasks. By reducing artifacts and maintaining structural coherence, our methods produce training data that better represents the target domain while preserving the semantic information from the source domain. This balance is essential for successful domain adaptation, as demonstrated in our subsequent semantic segmentation experiments.

3.5 Style transfer for adverse conditions domain adaptation

While the visual quality of generated images is important, the ultimate goal of our approach is to improve performance on downstream semantic segmentation tasks through effective domain adaptation.

In practical scenarios, access to target domain data is often limited, particularly for adverse weather conditions which may be difficult or dangerous to capture. Our method addresses this constraint by generating realistic target-domain-like images using only a small number of reference style images, enabling training of robust segmentation models without extensive target domain data collection.

In Section 3.4, we evaluated the visual quality of images generated by our proposed methods. Now, we focus on assessing how these generated images perform as training data for semantic segmentation models under various domain adaptation scenarios. This section examines whether the improvements in style transfer quality translate to better segmentation performance when adapting from synthetic to real domains, particularly under challenging adverse weather conditions.

3.5.1 Experimental setup

To evaluate the effect of style transfer for domain adaptation, we extend our experimental setting with an additional dataset and several training strategies.

Datasets We continue using the datasets described in Section 3.4: GTA5 as our synthetic source domain and both Cityscapes and ACDC as real target domains.

ACDC [Sakaridis et al. (2021)] contains 4,006 densely annotated images from driving scenarios in Switzerland, specifically focused on challenging weather conditions. The images are equally distributed across four adverse conditions: fog, nighttime, rain, and snow. Each image has a resolution of 1920×1080.

While Cityscapes serves as a standard domain adaptation benchmark, ACDC allows us to evaluate performance under more challenging conditions involving adverse weather. For the few-shot setting, we select 1 representative image from each target domain condition (normal, fog, night, rain, and snow) to serve as style references, reflecting a practical scenario where a limited number of target domain samples are available.

Network Architectures We evaluate our generated datasets using three network configurations:

- **DAFormer** [Hoyer et al. (2022a)] is a state-of-the-art domain adaptation framework for semantic segmentation that employs a transformer-based encoder with adaptive context-aware fusion. We use DAFormer with its default MiT-B5 backbone.
- **HRDA** [Hoyer et al. (2022b)] extends DAFormer with a hierarchical multi-resolution approach specifically designed to handle high-resolution input images, which can be particularly beneficial for detecting small objects and fine details in adverse conditions.

- **Segformer** [Xie et al. (2021)] is a transformer-based segmentation model without domain adaptation components. We include this model to evaluate whether our generated datasets can improve performance even without explicit domain adaptation techniques.

Training Protocol For all experiments, we follow a consistent training protocol. We first generate a dataset of 250 images for each target domain condition using the corresponding style reference image. For DAFormer and HRDA, we employ their standard training configurations, including self-training with pseudo-labels and curriculum learning strategies. For Segformer, we train directly on the generated images without domain adaptation components. All models are trained for 40,000 iterations with the AdamW optimizer, following learning rate and augmentation settings from their respective original implementations. We evaluate performance on the validation sets of Cityscapes and ACDC using the mean Intersection over Union (mIoU) metric presented in Chapter 2.

Baselines We compare our methods against several approaches, which we enumerate here. **Source Only** is trained only on the original synthetic GTA5 dataset without any adaptation. **Source + Style** uses the source dataset augmented with 1 style reference image from each target domain. **AdaIN Diffusion** uses images generated with standard AdaIN during the diffusion process without class-specific modifications. We also compare with images generated from the baseline methods **Cross-Image Attention** [Alaluf et al. (2024)], **DATUM** [Benigmim et al. (2023)], and **DGInStyle** [Jia et al. (2024)].

3.5.2 Qualitative analysis

Figures 3.4 and 3.5 provide visual comparisons of our methods when transferring styles from adverse weather conditions to synthetic images. In Figure 3.4, we observe how different methods transfer the snowy appearance from an ACDC image to a synthetic GTA5 scene. The Cross-Image Attention approach with standard AdaIN (Figure 3.4e) introduces several artifacts, including unrealistic blue coloration in trees and yellow color bleeding into mountain regions. Adding class-specific AdaIN as in our CACTI method (Figure 3.4c) partially mitigates these issues but still exhibits some color bleeding.

Our CACTI_{F} method (Figure 3.4f) produces the most coherent results, successfully transferring the snowy appearance while maintaining the structural integrity of the original scene. Notably, the snow accumulation on the road surface is realistic, and the color distribution across semantic classes is consistent with real-world expectations. The selective attention filtering prevents inappropriate transfers between dissimilar regions, which is particularly evident when comparing the rendering of distant objects like mountains and trees.

Figure 3.5 illustrates the challenge of transferring nighttime appearance, which involves significant lighting changes. The standard Cross-Image Attention approach (Figure 3.5e) produces an image with unnatural contrasts and loss of detail in darker regions. Class-based AdaIN (Figure 3.5c) improves the results in highlighting the vehicle and darkening the sky.



(a) Content (GTA5) (b) DGInStyle (c) CACTI

(d) Style (ACDC-snow) (e) Cross-image attention (f) CACTI_F

Figure 3.4: Style transfer comparison from GTA5 synthetic content (a) to ACDC-snow style (d). The results show: DGInStyle (b), Cross-Image Attention with standard AdaIN (e), Cross-Image Attention with class-based AdaIN (CACTI) (c), and complete CACTI_F (f). In dataset generation, images of 1024 by 512 pixels are used when possible. For DGInStyle, due to incompatibility with rectangular images, we show two images of 512 by 512 side-by-side. Note the artifacts in (e) (blue coloration in trees), color bleeding in mountains in (e) and (c), and the more realistic snow appearance on the road in our proposed methods.



(a) Content (GTA5) (b) DGInStyle (c) CACTI

(d) Style (ACDC-night) (e) Cross-image attention (f) CACTI_F

Figure 3.5: Style transfer results from GTA5 content (a) to ACDC-nighttime style (d), using the same methods as in Figure 3.4. Our methods better preserve visibility of important semantic elements while successfully transferring the nighttime appearance, maintaining better contrast and detail in critical regions.

Method	#S	Model	Fog	Night	Rain	Snow	ACDC	Cityscapes
Source + style	5	DAFormer	49.90	17.98	43.22	39.85	37.91	53.18
AdaIN diffusion	5	DAFormer	51.01	16.40	45.72	43.45	37.88	50.31
Cross-Image Attn	5	DAFormer	51.80	19.37	45.20	43.80	40.05	52.36
DATUM	5	DAFormer	52.70	22.07	**47.81**	**46.00**	42.16	**55.04**
DGInStyle	0	DAFormer	49.70	21.06	44.04	44.28	40.18	52.42
CACTI	5	DAFormer	**53.23**	**22.82**	47.78	44.49	**42.80**	53.47
CACTI_F	5	DAFormer	52.88	21.39	45.93	43.82	41.13	54.39
Source + style	5	HRDA	53.02	21.77	45.63	41.68	40.74	54.15
CACTI	5	HRDA	**58.24**	**26.07**	50.04	**45.44**	**45.66**	53.81
CACTI_F	5	HRDA	58.17	25.61	**51.36**	45.11	45.60	**54.82**
Source only	0	Segformer	44.32	16.02	**39.03**	37.88	34.78	43.99
CACTI	5	Segformer	**45.79**	18.23	**39.03**	**38.88**	36.62	43.30
CACTI_F	5	Segformer	44.53	**21.55**	36.79	38.78	**36.71**	**48.40**

Table 3.2: Semantic segmentation performance (mIoU %) on synthetic-to-real domain adaptation across different methods and models. We report results for each of the four adverse conditions in ACDC (Fog, Night, Rain, Snow) as well as their average (ACDC) and normal conditions (Cityscapes). Three network architectures are evaluated: DAFormer (domain adaptation transformer), HRDA (high-resolution domain adaptation), and Segformer (without explicit domain adaptation components). #S indicates the total number of target samples used for generation. All results are averaged over three independent trials with different random seeds. Bold values indicate the best performance in each model category and condition. Our proposed methods (CACTI and CACTI_F) consistently outperform baselines, with CACTI_F showing particular strength when paired with the HRDA architecture.

Our CACTI_F method (Figure 3.5f) achieves a more balanced nighttime appearance, where the overall scene is appropriately darkened but important semantic elements remain discernible like the edges of the vegetation. This is crucial for downstream segmentation tasks, as preserving the visibility and correct appearance of traffic participants and infrastructure elements enables better model training. The selective attention filtering is particularly effective in this challenging lighting condition, preventing the loss of semantic information while still transferring the nighttime style characteristics.

3.5.3 Quantitative results

Table 3.2 presents the quantitative results for semantic segmentation performance across different models and generation methods. When using the DAFormer architecture, both of our proposed methods (CACTI and CACTI_F) consistently outperform the baseline approaches across most conditions. Specifically, CACTI achieves the best performance on fog and night conditions, with mIoU scores of 53.23% and 22.82% respectively, surpassing the Source + Style baseline by +3.33% and +4.84%. DATUM performs slightly better on rain and snow conditions, but our CACTI method achieves the highest mean performance across all ACDC conditions (42.80%), demonstrating its robustness across diverse adverse conditions.

Method	FID ↓	LPIPS ↓	ACDC ↑	Cityscapes ↑
AdaIN diffusion	73.26	**0.30**	37.88	50.31
+ Cross-Image Attn	65.31	0.41	40.05	52.36
+ Class-wise AdaIN	**54.56**	0.39	**42.80**	53.47
+ Attn Filtering	55.30	0.36	41.13	**54.39**

Table 3.3: Ablation study on style transfer modules. We report FID relating to Cityscapes (lower is better), LPIPS relating to GTA5 (lower is better), ACDC mIoU (higher is better), and Cityscapes mIoU (higher is better). Bold values indicate the best score for each metric. AdaIN diffusion is the baseline, with the AdaIN operation applied at each step of the denoising process. Each row adds one component: cross-image attention, class-wise AdaIN (CACTI), and selective attention filtering (CACTI_F).

When comparing different model architectures, we observe that HRDA consistently outperforms DAFormer across all conditions when trained with both our CACTI and CACTI_F methods. The improvement is particularly significant for fog conditions, where CACTI with HRDA achieves 58.24% mIoU, representing a +5.01% improvement over the same method with DAFormer. This suggests that the hierarchical multi-resolution approach of HRDA is particularly effective at exploiting the high-quality and semantically consistent images generated by our method.

Examining performance across different adverse conditions reveals interesting patterns. The night condition remains the most challenging for all methods, with substantially lower mIoU scores than other conditions. However, our methods still provide considerable improvements over the baselines in this difficult scenario. For example, CACTI with HRDA achieves 26.07% mIoU on night conditions, representing a +4.30% improvement over the Source + Style baseline with the same architecture.

Perhaps most notably, our CACTI_F method enables even non-domain-adaptive architectures like Segformer to achieve competitive performance. Segformer trained with CACTI_F-generated images achieves 36.71% mean mIoU across ACDC conditions and 48.40% on Cityscapes, outperforming the Source Only baseline by +1.93% and +4.41% respectively. This demonstrates that our approach helps bridge the domain gap even without explicit domain adaptation techniques in the segmentation model.

Overall, these results confirm that our proposed methods not only generate visually appealing images but also produce semantically meaningful data that enables effective domain adaptation for semantic segmentation. The class-specific approach of CACTI ensures appropriate style transfer for different semantic regions, while the selective attention filtering in CACTI_F preserves critical structural information. Together, these techniques enable the generation of high-quality synthetic datasets that significantly improve segmentation performance across diverse target domains and model architectures.

3.5.4 Ablation study

We conduct an ablation study to assess the impact of the different modules introduced in this work. We show in Table 3.3 the addition of each component in the proposed style transfer pipeline. When using global AdaIN diffusion only, the method demon-

Method	Time (sec)
DDPM Inversion	6.47
Cross-Image Attention	19.91
CACTI	19.98
CACTI_F	46.72

Table 3.4: Comparison of generation time for Cross-Image Attention, CACTI and CACTI_F. DDPM Inversion is computed once before each style transfer method.

strates strong content preservation but limited style transfer and lower segmentation performance. Introducing cross-image attention improves the visual quality of stylized images and leads to better downstream performance. The addition of class-wise AdaIN (CACTI) further improves image realism and segmentation under challenging weather, demonstrating its value for domain adaptation. The selective attention filtering module (CACTI_F) then builds on this by slightly reducing performance in adverse conditions but improving generalization to simpler real-world environments. This progression confirms that each component of the method contributes to the general style transfer quality, with CACTI emphasizing robustness and style fidelity while CACTI_F offers more balance between generalization capability and content preservation.

3.5.5 Computation time

Table 3.4 presents a comparison of image generation times for Cross-Image Attention, CACTI and CACTI_F on a single NVIDIA RTX4090. The filtering approach in CACTI_F adds an extra computation of $\mathcal{O}((HW)^2)$ compared to standard cross-attention or CACTI. However, the overall impact on the computational cost is mitigated by the small size of the latents produced by the VAE. As a result, although CACTI_F shows longer generation times, the difference in computation time remains within the same order of magnitude. Without considering DDPM inversion, CACTI_F requires approximately 2.35 times the generation time of Cross-Image Attention. Regarding CACTI, its computation time is very close to the one of Cross-Image Attention since the custom AdaIN function is fast. This makes CACTI a good tradeoff between generation time and style transfer quality.

Overall, for the relatively small datasets used in our experiments the generation time remains reasonable. This is particularly true since DDPM Inversion is performed only once, during the first loading of each image into the style transfer pipeline.

3.6 Conclusion

In this work, we proposed two novel methods for synthetic-to-real domain adaptation: Class-wise Adaptive Instance Normalization (CACTI) and Selective Attention Filtering (CACTI_F). These methods leverage diffusion models for style transfer, generating high-quality images that maintain strong structural coherence while adapting to target domain characteristics. Our approach specifically targets the challenging scenario of few-shot domain adaptation, where only a limited number of target

domain images are available. The results demonstrate that class-aware style transfer can significantly improve semantic segmentation performance across various adverse weather conditions.

While our methods show clear improvements over standard style transfer approaches, the comparison with generative methods like DATUM reveals interesting trade-offs. DATUM achieves slightly better performance in some conditions using DAFormer, but it fundamentally transforms the content rather than preserving it. In contrast, our style transfer approach maintains structural coherence and semantic alignment with source images, which is critical when annotation preservation is required. The integration of our methods with HRDA demonstrates particularly strong results, achieving up to 58.24% mIoU on fog conditions and a mean of 45.66% across all adverse conditions, showing the potential of style transfer for bridging the synthetic-to-real gap.

Our experiments highlight the significant challenges that remain in domain adaptation for adverse weather conditions. The nighttime scenario presents the most difficulty, with our best method (CACTI with HRDA) achieving only 26.07% mIoU. This failure case can be attributed to the substantial appearance gap between synthetic and real nighttime scenes, as GTA5 contains fewer night examples and their visual characteristics differ significantly from real nighttime images in ACDC. These results underscore the importance of continued research in domain adaptation techniques specifically tailored for extreme lighting and weather conditions.

Several technical aspects of our approach warrant further investigation. The current filtering method relies on maximum attention values, which may not capture the full complexity of attention map distributions. Alternative pixel matching strategies, such as those employed in Eye for an Eye [Go et al. (2026)], could potentially improve correspondence quality, albeit at higher computational cost. Additionally, integrating ControlNet into the pipeline could provide more explicit guidance using segmentation masks, potentially improving structural coherence in challenging regions. Future work should also explore the integration of textual information through prompts to further guide the generation process.

This research demonstrates that class information can be highly valuable in guiding image generation and style transfer for domain adaptation. By leveraging semantic labels to constrain both statistical normalization and attention mechanisms, we can generate more realistic and semantically consistent target domain images. As autonomous systems continue to face deployment challenges in adverse conditions, approaches like ours that can effectively adapt models using limited target data will be increasingly important. Our work represents a step toward more robust perception systems that can operate reliably across diverse environmental conditions, though significant challenges remain, particularly for extreme conditions like nighttime driving.

A large number of style transfer methods have been considered in this study, which often come from the artistic style transfer community. However, each of these methods comes with its own limitations and is not always suitable for domain adaptation scenarios. Moreover, state-of-the-art approaches often combine multiple style transfer bricks within a single image generation pipeline, such as AdaIN [Go et al. (2026); Alaluf et al. (2024)] or ControlNet [Xu et al. (2025); Wang et al. (2024b)]. Evaluating an image generation method for domain adaptation implies generating a dataset and subsequently training a segmentation model, to assess the

generalization performance. Consequently, selecting the most appropriate methods and tuning their associated hyperparameters becomes a time-consuming process. In Chapter 4, we aim to automate this operation.

Chapter 4

Style transfer pipeline optimization

The style transfer methods presented in the previous chapter require choosing which operators to apply and in what order. This selection is typically performed manually, but the large combinatorial space of possible pipelines makes exhaustive evaluation impractical. This chapter formulates the construction of style transfer pipelines as a multi-objective optimization problem, modeled as a modified Traveling Salesman Problem and solved using NSGA-II. We study the use of paired-image metrics (DISTS for content preservation, DreamSim for style similarity) on individual image samples during evolution, enabling rapid pipeline evaluation without generating full datasets. After optimization, we evaluate the resulting Pareto front using distributional metrics and downstream segmentation performance. Experiments on the GTA5 to Cityscapes/ACDC benchmark show that evolutionary algorithms can efficiently explore the pipeline space and produce diverse solutions representing different trade-offs. However, the optimized pipelines do not match the downstream segmentation performance of ControlNet alone, revealing that current perceptual metrics do not reliably predict task performance, a limitation we discuss further in the conclusion.

This chapter covers the article *Multi-Objective Optimization for Synthetic-to-Real Style Transfer* published in *International Conference on the Applications of Evolutionary Computation (Part of EvoStar), April 2026* [Chigot et al. (2026b)].

4.1 Motivation

In Chapter 3 we presented several diffusion-based approaches for style transfer [Chigot et al. (2025b); Alaluf et al. (2024)]. They generate synthetic images that match real data appearance while preserving semantic structure. This is necessary since the label mask of the synthetic image needs to remain valid after style transfer. However, these methods involve numerous steps such as AdaIN in Cross-Image Attention and Eye-for-an-Eye [Go et al. (2026)], or ControlNet in StyleSSP [Xu et al. (2025)] and InstantStyle-Plus [Wang et al. (2024b)]. This process is similar to data augmentation pipelines, and the selection and ordering of these operations remains manual. Operators such as image darkening and sharpening, or more complex operations like image normalization through AdaIN [Huang and Belongie (2017)], can help transfer style from one domain to another. As with data augmentation, these operators can be combined to improve data quality for training models on downstream tasks such as image classification. Previous work has shown evolutionary algorithms can automate the search for data augmentation strategies in image classification [Ho et al. (2019); Marc et al. (2023); Pereira et al. (2022)]. In this work, we study style transfer pipeline optimization for domain adaptation, where two distributions of data must be considered.

We formulate the construction of style transfer pipelines as a multi-objective optimization task, which we refer to as MOOSS (Multi-Objective Optimization for Synthetic-to-real Style transfer). A pipeline consists of a sequence of transformations applied to source domain images, with each transformation altering visual properties while preserving semantic content. We model this as a modified Traveling Salesman Problem (TSP) [Flood (1956)], where data operators correspond to cities and a special terminal node ends the sequence. This formulation differs from standard TSP by allowing variable-length solutions through early termination, similar to the Orienteering Problem [Golden et al. (1987)].

A main challenge in optimizing style transfer pipelines is evaluation. Metrics like FID [Heusel et al. (2017)] or CMMD [Jayasumana et al. (2024)] estimate distances between distributions of data but require costly generation of many images. Training a segmentation network on each candidate dataset would be even more expensive. Instead, we use metrics that operate on individual image pairs: DISTS [Ding et al. (2022)] for structural similarity and DreamSim [Fu et al. (2023)] for perceptual style matching. These metrics allow rapid evaluation during genetic algorithm optimization while maintaining correlation with distributional quality.

We follow the same evaluation protocol as in Chapter 3 by performing domain adaptation from GTA5 [Richter et al. (2016)] to Cityscapes [Cordts et al. (2016)] and ACDC [Sakaridis et al. (2021)], which includes adverse weather conditions.

Our contributions are: formulation of augmentation pipeline selection as a combinatorial problem suitable for evolutionary optimization; identification of paired-image metrics that enable feasible optimization while correlating with distributional quality; and demonstration that multi-objective evolution produces diverse solutions representing different trade-offs between content preservation and style adaptation.

4.2 Context

4.2.1 Style transfer for domain adaptation

Domain adaptation in semantic segmentation aims to train models on source domain data that generalize to target domains with different visual characteristics. Input-level adaptation methods transform source images to resemble the target domain while preserving semantic content. Several works have attempted to tackle the synthetic-to-real domain adaptation problem by using diffusion models to generate realistically looking datasets.

As detailed in Chapter 3, CACTI [Chigot et al. (2025b)] performs class-specific feature alignment for style transfer, ControlNet [Zhang et al. (2023a)] enables spatially conditioned image generation, and AdaIN [Huang and Belongie (2017)] transfers style by aligning global feature statistics. Beyond these methods, DGInStyle [Jia et al. (2024)], DATUM [Benigmim et al. (2023)], and DoGE [Wang et al. (2024e)] also leverage diffusion models for synthetic-to-real domain adaptation.

4.2.2 Data augmentation optimization

The optimization of data augmentation strategies has received attention as a method to improve model generalization without requiring additional labeled data. A set of operators modify data with the goal of improving model training; optimization focuses on choosing the appropriate data modification operations.

AutoAugment [Cubuk et al. (2019)] introduced a reinforcement learning approach where a controller network searches over augmentation operations to find policies that maximize validation accuracy. The method trains thousands of child models to evaluate policies, making the search computationally expensive. RandAugment [Cubuk et al. (2020)] simplifies the search space by randomly selecting augmentation operations with uniform magnitude. Fast AutoAugment [Lim et al. (2019)] employs density matching to approximate policy search, while Faster AutoAugment [Hataya et al. (2020)] uses backpropagation through differentiable augmentation implementations. However, these methods generally require repeated model training for evaluation.

Population-based methods offer an alternative. Population Based Augmentation (PBA) [Ho et al. (2019)] addresses computational cost by learning augmentation schedules rather than fixed policies. Pereira et al. [Pereira et al. (2022)] apply an evolutionary strategy to discover optimal sequences of transformation functions. Marc et al. [Marc et al. (2023)] consider class-specific augmentation strategies using a genetic algorithm. Recent differentiable methods like DADA [Li et al. (2020)] and FreeAugment [Bekor et al. (2024)] enable end-to-end optimization of augmentation policies.

These policy-based formulations share a common characteristic: they determine augmentations on a per-image or per-minibatch basis during training. This dynamic approach allows for adaptive augmentation but requires evaluating fitness through model training, which becomes prohibitively expensive when generating datasets for domain adaptation. For style transfer in semantic segmentation, the requirements differ. The goal is to generate a transformed dataset that can be used to train or test models that generalize to a target domain. Rather than determining augmentations per-image during training, we seek to optimize a single pipeline that can be applied

to the source domain to produce a dataset resembling the target domain. This formulation enables the use of powerful style transfer operators such as AdaIN [Huang and Belongie (2017)] and diffusion-based methods [Zhang et al. (2023b)], which have greater impact on domain gap reduction than traditional geometric augmentations. The fixed pipeline approach also enables evaluation using distributional metrics before training segmentation models. To our knowledge, this is the first work which aims at optimizing data augmentation strategies for style transfer.

4.2.3 Data similarity metrics

Paired-image metrics assess similarity between individual images. These metrics are detailed in Chapter 2. LPIPS [Zhang et al. (2018)] measures perceptual similarity using features from pretrained networks. Deep Image Structure and Texture Similarity (DISTS) [Ding et al. (2022)] measures perceptual similarity with robustness to texture variations, combining structure and texture similarity measurements. Lower DISTS values indicate images share similar content. The key difference between the two is that DISTS explicitly separates structure from texture similarity and is therefore more tolerant of texture variations that do not alter content. This makes DISTS better suited for evaluating style transfer, where texture is expected to change while structure should be preserved. DreamSim [Fu et al. (2023)] provides a metric for evaluating perceptual similarity with focus on mid-level variations such as pose, perspective, and style. These pairwise metrics evaluate individual transformations, allowing for the evaluation of style transfer pipelines on a small number of images.

Distributional metrics like FID and CMMD provide assessment of entire generated datasets but require generating many samples. Fréchet Inception Distance (FID) [Heusel et al. (2017)] computes distance between two feature distributions. CLIP Maximum Mean Discrepancy (CMMD) [Jayasumana et al. (2024)] addresses FID limitations by using CLIP embeddings and Maximum Mean Discrepancy with a Gaussian RBF kernel. These distributional metrics have been used to measure the quality of style transfer methods, using distributions of generated data compared to target domain data [Chung et al. (2024)], however such a calculation would be too costly for optimization.

4.3 Methodology

The task of synthetic-to-real domain adaptation through style transfer can be formulated as finding a sequence of transformations $T = [t_1, t_2, \ldots, t_n]$ that, when applied to source domain images $\mathcal{S}$, produces images $\mathcal{S}'$ that maintain semantic structure while matching visual characteristics of the target domain $\mathcal{T}$. Each transformation t_i belongs to a set of available operators $\mathcal{A}$.

The problem presents two competing objectives. First, transformed images must preserve semantic content and spatial structure to ensure ground truth annotations remain valid. Second, transformed images should match the visual style of the target domain to reduce the domain gap.

Evaluating candidate pipelines during optimization requires balancing accuracy and computational cost. To enable feasible optimization, we use paired-image metrics that operate on individual samples. For each candidate pipeline, we apply the sequence to a small set of source images and compute metrics on the resulting

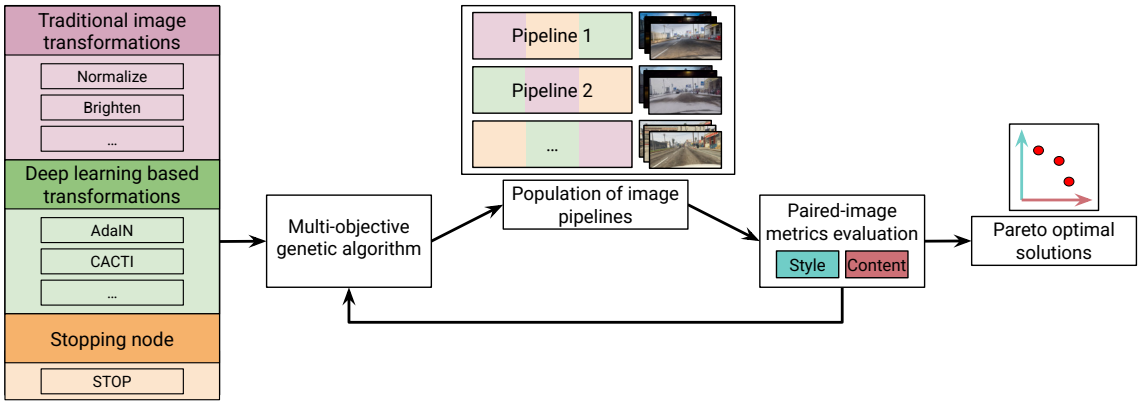


Figure 4.1: Overview of the proposed multi-objective pipeline optimization framework. Image transformation pipelines, composed of traditional and deep learning–based operators with a stopping node, are evolved using a multi-objective genetic algorithm. Each pipeline is evaluated using paired-image metrics that measure content preservation (DISTS) and style similarity to the target domain (DreamSim). Ultimately, the evolutionary process produces a Pareto front of non-dominated pipelines representing different trade-offs between both objectives.

pairs. This reduces evaluation cost by several orders of magnitude while maintaining correlation with distributional quality.

The problem becomes a multi-objective optimization task where we seek pipelines that minimize both DISTS (to preserve content) and DreamSim (to match target style). Multi-objective optimization produces a Pareto front of non-dominated solutions representing different tradeoffs between objectives. Figure 4.1 illustrates the method.

4.3.1 Traveling Salesman Problem (TSP) formulation

We model the augmentation pipeline selection problem as a variant of the TSP where each data operator represents a city. The TSP seeks the shortest route visiting a set of cities exactly once and returning to the start. We consider this suitable as the main style transfer operators used in this study, such as AdaIN [Huang and Belongie (2017)] and ControlNet [Zhang et al. (2023a)], only operate as a single application to an image. Other operators, such as sharpening the image or darkening it, could be applied multiple times, but do not represent the main contributors to style transfer and are thus included as one-off operator choices.

The key modification to the TSP is introduction of a terminal node t_{stop} which can be visited at any point to end the pipeline. The solution representation is a permutation of augmentation operators including the terminal node: $\pi = [\pi_1, \pi_2, \ldots, \pi_m]$ where $m = |\mathcal{A}| + 1$. When evaluating a solution, operators are applied sequentially until the terminal node is encountered. This is similar to the Orienteering Problem [Golden et al. (1987)], which extends TSP by selecting a subset of cities to visit subject to a budget constraint, however, unlike the Orienteering Problem, we do not impose a budget on total travel time. The discrete and combinatorial nature makes this problem appropriate for genetic algorithms. The solution space consists of permutations of operators, a representation studied extensively in TSP contexts. Genetic algorithms have a long and successful history of use for the TSP and variants, including the Orienteering Problem [Tasgetiren and Smith (2000)], for a multitude of

applications [Potvin (1996); Larranaga et al. (1999); Tsai et al. (2004); Groba et al. (2015); Davendra (2010)].

4.3.2 Multi-objective evolutionary algorithm

We employ NSGA-II[Deb et al. (2002)] to solve the multi-objective optimization problem. NSGA-II uses non-dominated sorting to classify individuals into fronts based on dominance relationships and maintains diversity through crowding distance.

The initial population is generated using permutation random sampling, which creates valid permutations of augmentation operators and terminal node. For crossover, we use the edge recombination crossover [Whitley et al. (1991)], which preserves adjacency information from parent solutions. For mutation, we apply inversion mutation, which selects two random positions and reverses the order of all elements between them. Duplicate elimination prevents the population from containing multiple copies of the same solution.

4.4 Experiments

4.4.1 Image operators

The augmentation pipeline consists of ten operations: nine transformations and one terminal node. The transformations include six traditional image processing operations applied using Pillow, a modern version of the Python Imaging Library [Clark and Contributors (2025)]. The normalize operation adjusts color distribution to match a reference using histogram matching. Blur applies Gaussian smoothing with a radius value of 2.0 pixels. Brighten and darken modify image luminosity by factors of 1.5 and 0.7 respectively. Contrast modifies contrast by a factor 1.5. Sharpen applies unsharp masking with a factor 2.0.

Three deep learning-based style transfer operations provide more substantial domain adaptation: AdaIN, CACTI, and ControlNet, described in detail in Chapters 2 and 3. In this pipeline, ControlNet takes as input an example style image from the target domain along with spatial conditioning derived from the source image, specifically a Canny edge map and a semantic segmentation mask. These conditioning inputs guide the generation process to preserve the spatial layout while adopting the target domain appearance. For each deep learning operation, we use as reference a single set of five images, one for each weather condition considered. This ensures the method is applicable in a wide range of situations, even in the case of limited target data.

4.4.2 Optimization configuration

We use source domain images from the GTA5 [Richter et al. (2016)] dataset. Target domain images are sampled from the Cityscapes [Cordts et al. (2016)] training set for clear day conditions and ACDC [Sakaridis et al. (2021)] training set for adverse weather conditions. The five selected images from these content and style datasets are respectively shown in Figure 4.2.

For each candidate pipeline during optimization, we perform evaluation using the five sampled source images. For each source image, we apply the pipeline

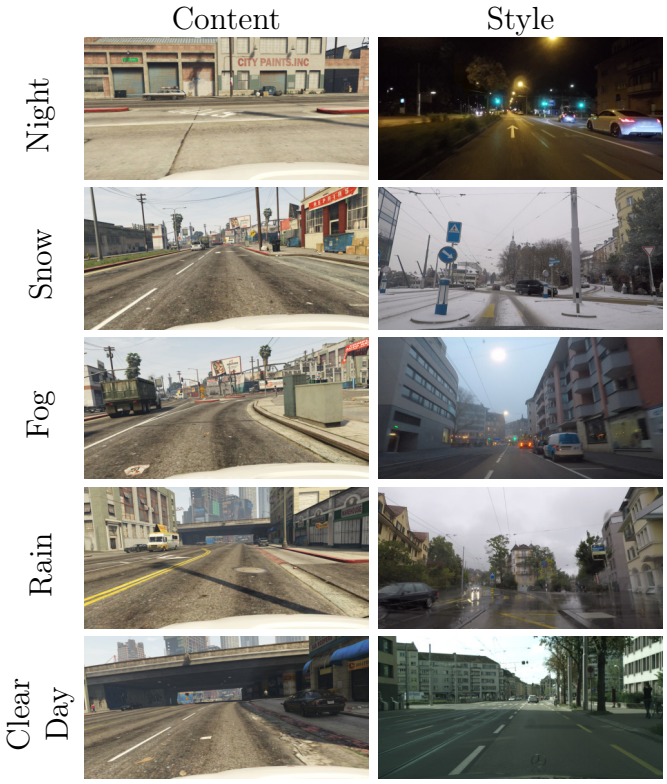


Figure 4.2: Pairs of images (content, style) used during the optimization process, respectively used for the night, snow, fog, rain, and clear day style transfer. Content images are from the GTA5 dataset. Style images are from the ACDC dataset (night, snow, fog, and rain style) and the Cityscapes dataset (clear day style).

transformations and compute the DISTS and DreamSim metrics with respect to the original and target domain images, respectively.

Our NSGA-II configuration uses a population size of 20 individuals and generates 20 offspring per generation. Inversion mutation and edge recombination crossover are used to generate all offspring. The relatively small population is sufficient given the limited size of the search space. The evolutionary process is run for 20 generations, as Pareto front convergence was observed.

Optimization was performed on a Linux server with a 32 core Intel i9 CPU and a RTX4090 GPU. The largest time cost was image generation. CACTI is the most expensive operator, with a maximum generation time of 40 seconds, with ControlNet in second, taking up to 2 seconds. All other operators function in less than 1 second for generation. The maximum evaluation time for a single pipeline was therefore roughly 200 seconds, or 3 minutes and 20 seconds on a single machine. Full optimization took 13 hours, meaning 39 minutes per NSGA-II generation on average and 23.4 seconds per generated image. Evaluation of chosen pipelines, requiring generation of 1250 images, also took up to 13 hours per pipeline, or 37.4 seconds per image, for solutions containing CACTI. This demonstrates the benefit of an optimization approach using pairwise metrics over a small number of images; the full optimization took as long as the distributional evaluation of a single pipeline. Model training on generated images using the same hardware took 5.5 hours and was performed 3 times for averages.

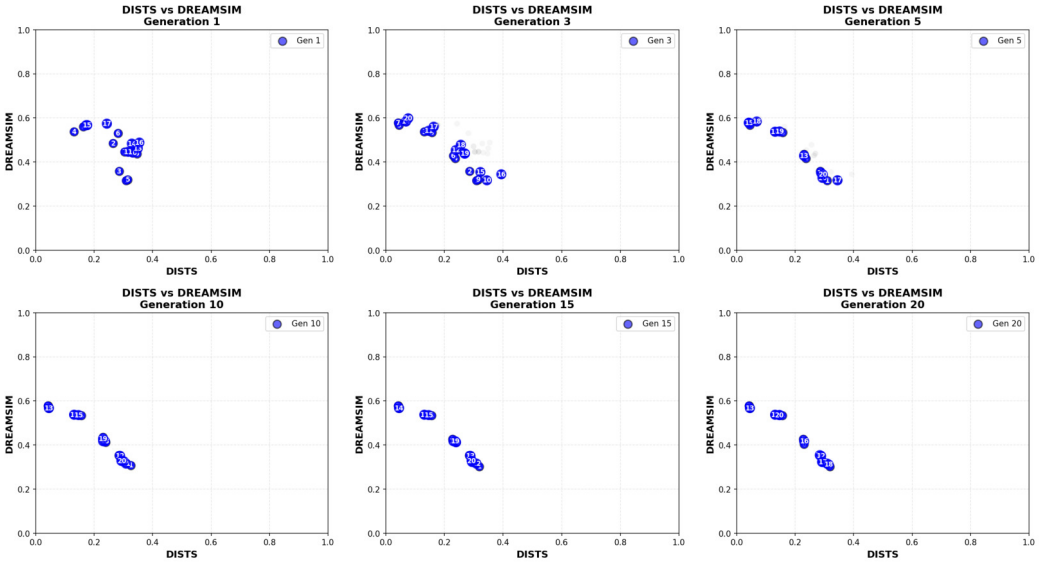


Figure 4.3: Populations produced at several stages of the optimization process, from generation 1 on the top left to generation 20 on the bottom right. On the x axis is the DISTS value representing content preservation, and on the y axis the DreamSim value representing style transfer. Both metrics range from 1.0 to 0.0, with lower values being better.

4.4.3 Optimization results

Figure 4.3 presents the progression of optimization across twenty generations. The evolutionary algorithm successfully drives the population toward the Pareto front, with individuals moving from high DISTS and DreamSim values toward the region where both objectives are minimized.

The initial population shows diversity, with solutions distributed across a wide range of combinations. Several solutions from the first generation are already close to what will become the final Pareto front, which is not surprising due to the relatively small search space. By generation 3, the characteristic shape of the Pareto front becomes visible.

From generation 5 onward, the population exhibits limited further evolution. Solutions remain clustered in approximately four to five distinct regions along the Pareto front. This stagnation after relatively few generations confirms that the search space is small enough to be thoroughly explored with a modest computational budget.

The observed stagnation demonstrates that the problem formulation and resolution method are capable of quickly identifying high-quality solutions, making the approach computationally feasible. The final Pareto front provides solutions spanning different points along the content-style trade-off curve, enabling practitioners to select pipelines according to specific requirements.

4.5 Analysis of optimized pipelines

4.5.1 Optimized pipelines

Optimization produces a Pareto front containing multiple non-dominated solutions. To evaluate these solutions, we select representative pipelines that span the front.

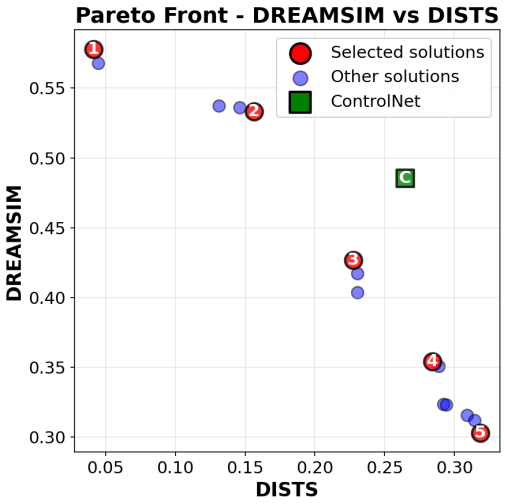


Figure 4.4: The final Pareto front of sequences optimized by NSGA-II. The five selected sequences (in red) are highlighted among all optimal solutions (in blue). A baseline sequence using only ControlNet is shown in green. The top-left solution corresponds to the highest content preservation (low DISTS), whereas the bottom-right solution achieves the most realistic style (low DreamSim).

We manually select one pipeline from each cluster, resulting in five representative solutions. Figure 4.4 shows these selected pipelines in red among the full set of optimal solutions. The selection ensures even distribution across the trade-off between content preservation and style transfer. For comparison, we also evaluate a baseline pipeline using only ControlNet:

C. ControlNet → Stop

The five selected pipelines, ordered from high content preservation to high style fidelity, are:

1. Sharpen → Stop
2. Darken → Sharpen → AdaIN → Stop
3. AdaIN → Darken → CACTI → Stop
4. ControlNet → CACTI → Darken → Sharpen → AdaIN → Stop
5. AdaIN → Normalize → Blur → Contrast → ControlNet → CACTI → Stop

These pipelines reveal patterns about the relationship between solution complexity and objective values. As DreamSim values decrease (indicating better style transfer), pipelines become progressively more complex. Pipeline 1 applies only a single operation, while Pipeline 5 uses six of the nine available augmentations. This suggests that achieving strong style transfer while maintaining reasonable content preservation requires composing multiple transformations.

The presence of deep learning operators AdaIN, ControlNet, and CACTI in the more style-focused pipelines reflects their effectiveness for this task. ControlNet appears in only two of the five selected pipelines, both of which have DreamSim values in the lower range. This distribution suggests that ControlNet contributes



Figure 4.5: Style reference images, then content images, then pipelines 1 → 5, and finally ControlNet on the Night (left) and Snow (right) environmental conditions.

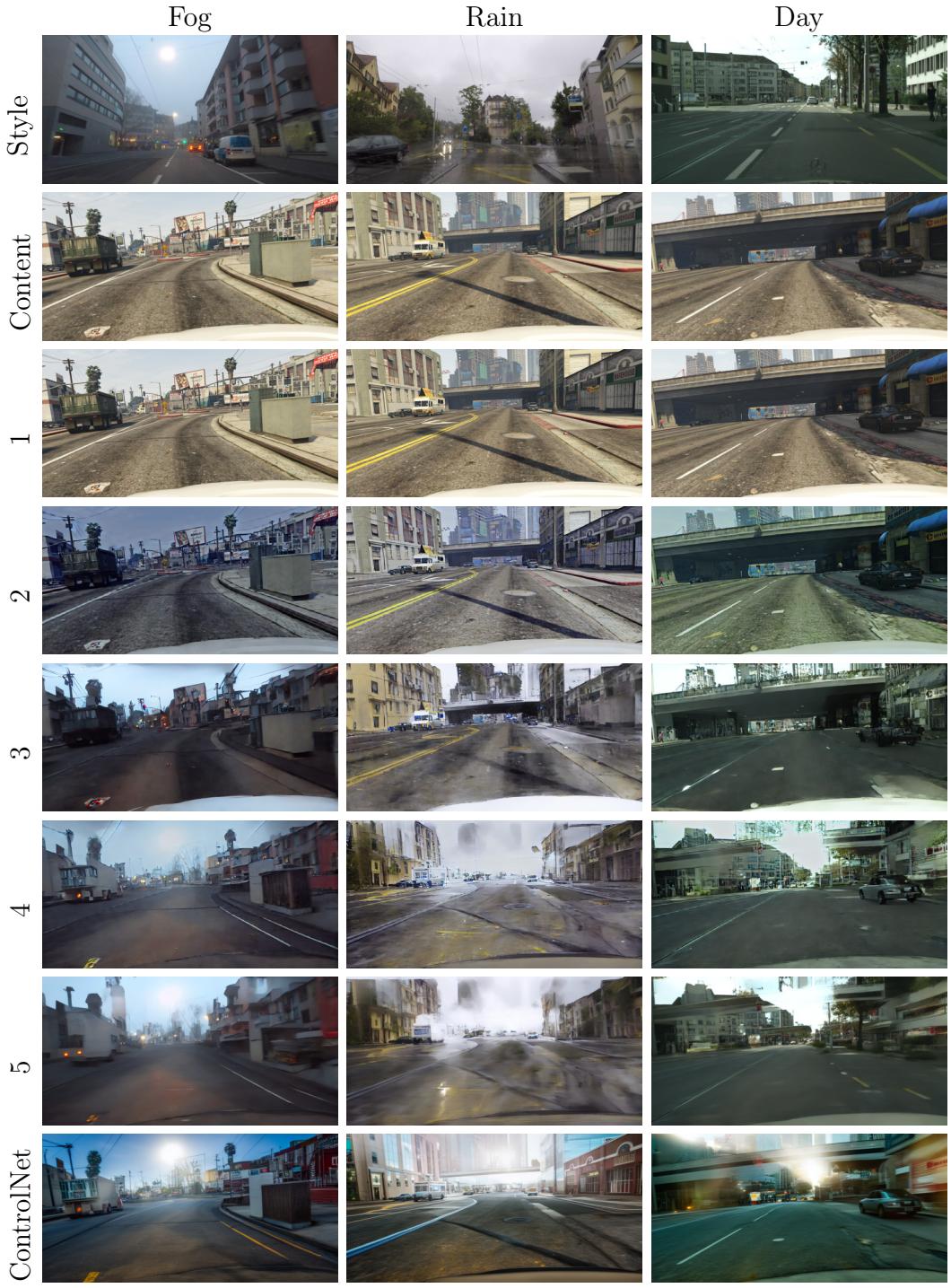


Figure 4.6: Style reference images, then content images, then pipelines 1 → 5, and finally ControlNet on the Fog (left), Rain (center), and Clear Day (right) environmental conditions.

more to the style transfer objective in this context, which is surprising given its intended effect of content preservation [Zhang et al. (2023a)].

The visual results from the five pipelines presented in Figures 4.5 and 4.6 confirm the progression suggested by DreamSim metric values. Across all three weather conditions, images become progressively more stylized from Pipeline 1 to Pipeline 5, with visual characteristics increasingly matching the style reference images.

Pipeline 1, which applies only the sharpen operator, produces images very similar to original GTA5 content. The primary visible change is enhanced edge definition, with little modification to color distribution, lighting, or texture. This minimal transformation reveals a limitation of relying solely on DISTS. DISTS is designed to measure perceptual similarity with emphasis on structural content, making it relatively insensitive to subtle stylistic differences between synthetic and real domains.

Images from Pipelines 2 through 5 show increasingly substantial modifications to visual appearance while maintaining recognizable scene structure. Colors shift to match real-world distributions, lighting conditions become less uniform, and texture characteristics change from smooth rendered appearance to more complex patterns of photographic imagery. These changes are what DreamSim is designed to capture: perceptual style similarity that encompasses broader visual characteristics distinguishing synthetic and real images.

The balance between DISTS and DreamSim becomes apparent through these visual results. DISTS prevents pipelines from applying transformations so aggressive that spatial structure is lost, which would invalidate ground truth annotations. DreamSim encourages sufficient style modification to reduce the domain gap. Pipelines in the middle of the Pareto front (particularly Pipelines 3 and 4) achieve a reasonable compromise.

On the other hand, the ControlNet outputs differ noticeably from those produced by the selected pipelines. Although the overall structure looks well preserved, the generated images tend to have an overly smooth texture that drifts further from the style reference. In the night scenario, ControlNet achieves a reasonably strong style transfer, producing a dark scene with pronounced lighting and a fading of the road markings. In contrast, the snow scenario shows only minimal stylistic modification, with slight adjustments in lighting and color. Compared to Pipeline 3, the ControlNet results appear to have higher perceptual quality, yet they exhibit weaker alignment with the style reference and less faithful preservation of content details such as building structure.

4.5.2 Pipeline evaluation

For each selected pipeline, we generate a complete dataset by applying the augmentation sequence to source domain images. We generate 250 transformed images for each of the five target weather conditions, producing a total dataset of 1250 images.

Table 4.1 reports the DISTS and DreamSim values used as optimization metrics in Figure 4.4, computed both on the small evaluation set (5 images) and on the full generated datasets (1250 images per pipeline). Across all pipelines, DISTS and DreamSim remain highly consistent when moving from 5 to 1250 images. This stability indicates that both metrics are robust to dataset scaling and can therefore be reliably estimated from only a few samples during optimization. As DreamSim improves and DISTS gets worse (moving from Pipeline 1 to Pipeline 5), CMMD

Samples	DISTS 5	DISTS 1250	DreamSim 5	DreamSim 1250	CMMD 1250
Pipeline 1	**0.0415**	**0.0396**	0.5779	0.5600	5.15
Pipeline 2	0.1563	0.1478	0.5334	0.5241	4.57
Pipeline 3	0.2277	0.2249	0.4266	0.3837	2.67
Pipeline 4	0.2840	0.2777	0.3540	0.3465	2.37
Pipeline 5	0.3185	0.3025	**0.3030**	**0.3177**	**2.35**
GTA5	—	—	0.5766	0.5577	5.18
ControlNet	0.2646	0.2666	0.4852	0.4653	3.48

Table 4.1: Image quality metrics for the five selected pipelines and the baseline, computed between content and style pairs of reference images, as well as generated datasets (1250 images) and the target domain.

	Fog	Night	Rain	Snow	ACDC	Cityscapes
Pipeline 1	48.57	16.65	42.50	40.34	37.07	51.41
Pipeline 2	50.65	19.89	45.65	43.54	39.40	51.51
Pipeline 3	49.99	19.32	45.17	42.55	39.79	50.69
Pipeline 4	50.94	19.46	45.94	44.02	40.27	51.36
Pipeline 5	48.91	17.57	44.29	42.69	38.73	50.09
GTA5	48.74	16.35	42.52	40.33	36.64	50.56
ControlNet	**55.73**	**24.84**	**46.66**	**44.39**	**42.42**	**52.12**

Table 4.2: Evaluation metrics for the five selected pipelines and the baseline. mIoU values are shown for each ACDC weather condition, the mean across ACDC conditions and Cityscapes clear day images.

generally decreases, indicating better distributional match. This correlation between DreamSim and CMMD further validates the use of pairwise metrics in the proposed optimization approach.

Next, we evaluate the generated data on semantic segmentation model training to evaluate the capacity of these style transfer pipelines to aid in domain adaptation. We train DAFormer [Hoyer et al. (2022a)], a transformer-based domain adaptation architecture, on each generated dataset. Following standard protocols [Hoyer et al. (2022a); Benigmim et al. (2023); Jia et al. (2024); Chigot et al. (2025b)], we train for 40,000 iterations with batch size 2. After training, we evaluate each model on ACDC and Cityscapes validation sets. We report mean Intersection over Union (mIoU), the pixel-wise segmentation accuracy averaged across all semantic classes. This metric directly reflects downstream task performance achieved by training on datasets generated with each optimized pipeline.

Table 4.2 presents downstream task performance of each pipeline. mIoU scores vary across pipelines and weather conditions. Furthermore, the baseline ControlNet achieves mIoU values that exceed most optimized pipelines across several conditions. Having been fine-tuned directly on the style images, ControlNet can effectively combine the learned target-domain features with strong class and structural con-

ditioning. Even though its outputs show weaker perceptual similarity, particularly in textures and lighting, ControlNet generates images that stay close to the target domain in terms of semantic content. This strong semantic coherence helps explain its comparatively high downstream performance despite less accurate style transfer.

The segmentation results indicate that distributional similarity between generated and target data does not directly correlate with training performance. This finding aligns with recent work on domain generalization, particularly DATUM [Benigmim et al. (2023)], which demonstrates that realistic appearance is not necessary for training segmentation networks.

4.6 Conclusion and perspectives

This work demonstrates that evolutionary algorithms can effectively explore the space of data augmentation pipelines for domain adaptation. The proposed TSP formulation allows multi-objective optimization using pairwise metrics to generate multiple candidate solutions. This formulation constitutes a novel application of evolutionary computation to automate the design of style transfer pipelines for semantic segmentation domain adaptation. As measured by DISTS and DreamSim, our approach produces pipelines with good style transfer properties and diverse trade-offs between content preservation and style matching.

A key advantage of this method is computational efficiency. Optimizing the complete set of pipelines requires the same time as evaluating a single pipeline for full data generation. This means the quick metrics can be applied to a new target domain before committing to expensive data generation, allowing practitioners to consider multiple pipeline options early in the adaptation process.

However, our results reveal a limitation in the choice of objectives. While the evolved pipelines achieve good style transfer, they do not match the downstream segmentation performance of ControlNet alone when measured by mIoU. This indicates that the DISTS and DreamSim combination, while effective for assessing style transfer quality, does not serve as a reliable surrogate for downstream task performance on image segmentation. Future work should focus on identifying metrics that can cheaply estimate downstream segmentation quality. For domain adaptation specifically, an ideal formulation would include three objectives: DreamSim for style matching, a surrogate metric for segmentation performance, and compute time.

The methods presented in Chapter 3 and Chapter 4 were evaluated using widely adopted public datasets for semantic segmentation, namely GTA5, Cityscapes, and ACDC, which depict autonomous driving scenarios. However, these datasets are relatively small compared to industry standards, particularly for safety-critical systems that require robustness across a broad range of conditions. Moreover, autonomous driving remains the predominant application offering both synthetic and real-world data, as well as adverse environmental conditions. As a result, domain adaptation methods are often compared exclusively within this application domain, which limits the diversity of research in the field. In the next chapter, we extend this evaluation by assessing domain adaptation methods in the novel use case of runway detection.

Chapter 5

Application to runway detection

This chapter applies the domain adaptation methods to runway detection, an industrial vision task where real labeled data is scarce and adverse conditions such as night, fog, and snow are difficult to capture in large quantities. Two complementary studies are presented. The first adapts the CARE feature-level domain adaptation method to runway detection using a private dataset of real images and simulator-generated synthetic data. Mixing synthetic and real data consistently improves daytime detection, and adding synthetic nighttime images substantially raises nighttime performance. The second study applies the CACTI_{F} pixel-level style transfer pipeline from Chapter 3 to an expanded experimental setup covering four environmental conditions (day, night, fog, snow). Qualitatively, style transfer successfully conveys realistic lighting, textures, and weather appearance. Severe adverse conditions, however, introduce a tradeoff between realism and label preservation: fog and snow can erase the runway from the generated image, which CACTI_{F} partially mitigates by retaining content structure. On the held-out fog test set, style transfer substantially improves generalization to unseen airports. Together, these studies confirm that synthetic data is a practical complement to real data in safety-critical industrial settings, and that pixel-level and feature-level adaptation address different aspects of the domain gap.

5.1 Applicative context

Runway detection is a key step in vision-based landing assistance. As discussed in Chapter 1, a camera-based system must first localize the runway before it can support guidance or pose estimation. Recent work usually formulates this problem either as object detection [Wang et al. (2022b); Lindén et al. (2021); Li et al. (2024a)] or as instance segmentation [Wang et al. (2024c)] using deep learning. In this chapter, we mainly focus on detection, since it provides a simple and efficient localization output.

The main difficulty in formulating runway detection as a deep learning problem is data collection. Real runway images are expensive to acquire, difficult to label, and often redundant because many frames come from the same landing sequence. For this reason, runway detection studies rely heavily on synthetic data. LARD [Ducoffe et al. (2023)] provides both a Google-Earth-based generator and 1,800 real images collected from YouTube videos. FS2020 [Chen and Hu (2024)] offers a synthetic dataset based on Microsoft Flight Simulator for runway segmentation and line detection. Although not dedicated to runway localization, but rather aircraft detection from satellite perspective, RarePlanes [Shermeyer et al. (2021)] also shows the value of synthetic imagery for aerial vision tasks. In practice, most published runway-detection methods either generate their own simulated data [Wang et al. (2022b)] or rely on open synthetic resources such as LARD [Li et al. (2024a)].

This use of simulation is especially relevant in aviation because rare situations such as night, snow, or fog are the cases that are hardest to record in large quantity. At the same time, simulated images introduce the synthetic-to-real gap discussed in Chapters 1 and 2; models trained on synthetic data representing these situations may not generalize well to real data distributions. Thus, this chapter studies two complementary approaches for runway detection: feature-level domain adaptation for object detection in Section 5.2, and pixel-level adaptation through style transfer in Section 5.3. While previous chapters focus on common benchmark tasks such as GTA to Cityscapes adaptation, this chapter focuses on the central industrial application of this thesis, runway detection.

5.2 Synthetic-to-real domain adaptation for runway detection

This section covers the article *Synthetic Data for Robust Runway Detection* published in *Computer Analysis of Images and Patterns, September 2025* [Chigot et al. (2025a)].

5.2.1 Motivation

The data challenges and the synthetic-to-real gap described in Section 5.1 apply directly to this study. We have access to a small private dataset of real runway images from 27 U.S. airports, covering both daytime and nighttime conditions, alongside a larger synthetic dataset. Unlike public runway datasets [Ducoffe et al. (2023); Chen et al. (2023); Chen and Hu (2024)], which usually include few to no real images, this setting allows evaluation on real data. We focus on nighttime as an underrepresented condition: real nighttime images are available for evaluation but never used during training, making it a controlled testbed for out-of-domain robustness.

We make the following contributions, within the case of runway detection: (1) we demonstrate the benefits of using synthetic data to train a detection model in an industrial use case; (2) we adapt the CARE [Prabhu et al. (2023)] domain adaptation method to the runway detection setting by removing the class- and size-reweighting terms; (3) we show the advantages of using a domain adaptation approach to enhance model's robustness under adverse conditions, in our case nighttime.

5.2.2 Related work

A general background on object detection, synthetic data generation, and synthetic-to-real domain adaptation was introduced in Chapters 1 and 2. We only recall here the elements most directly connected to the present study.

Object detection Object detection remains the dominant formulation for runway localization in the literature. One-stage detectors such as YOLO [Redmon et al. (2016)] and SSD [Liu et al. (2016)] favor speed, while two-stage detectors such as Faster R-CNN [Ren et al. (2017)] usually offer stronger localization accuracy, as discussed in Chapter 2. Because Faster R-CNN is both well established in runway detection [Wang et al. (2022b); Lindén et al. (2021); Li et al. (2024a)] and widely used in object-detection domain adaptation, we adopt it in this first study. Its backbone, region proposal network, and detection head also make it straightforward to integrate feature-alignment losses such as CARE [Prabhu et al. (2023)].

Synthetic data The motivations for synthetic data in industrial vision were detailed in Chapter 1. In runway detection, they are particularly strong because real acquisition campaigns are costly and operationally constrained. Existing aviation datasets therefore combine limited to none real imagery with larger simulated databases.

First, LARD [Ducoffe et al. (2023)] provides a generator for runway detection, based on Google Earth, as well as 1800 real images taken from YouTube videos and manually labeled. LARD 2.0 [Bougacha et al. (2026)] extends this work with synthetic images captured from additional simulators. On other tasks than runway detection, FS2020 [Chen and Hu (2024)] provides a synthetic dataset generated with Microsoft Flight Simulator, meant for runway segmentation and lines detection. Finally, Rareplanes [Shermeyer et al. (2021)] integrates 3D models of aircraft into real images for remote sensing applications. Logically, the works published for runway detection use mainly synthetic data. They either create their own with flight simulators [Wang et al. (2022b)], or use available open datasets [Li et al. (2024a)] such as LARD. In an attempt to assess the benefits of synthetic images, Linden et al. [Lindén et al. (2021)] compare the impact on detection performance of several environmental conditions in a synthetic dataset and examine a style transfer approach on automotive data.

Synthetic to real domain adaptation Domain adaptation methods attempt to reduce the synthetic-to-real gap either by modifying the images or by aligning feature distributions across domains. SC-UDA [Yu et al. (2022)] follows the first route by stylizing synthetic images and using Faster R-CNN predictions to expand an unlabeled real set. CARE [Prabhu et al. (2023)] follows the second route and

introduces an additional loss that encourages synthetic and real object features to remain close during Faster R-CNN training. CARE also includes reweighting terms to compensate for differences in object size and class frequency across domains.

In runway detection, such methods remain relatively unexplored because real annotated data are scarce. This makes runway detection a relevant application for testing whether domain adaptation can improve robustness to real nighttime imagery.

5.2.3 Data overview

Data collection In our use case, real images are limited in quantity due to their high cost of acquisition. The images are captured using an aircraft equipped with cameras and sensors, navigating across various airports in the U.S. Data is acquired by photographing runways at selected airports during landing sequences. In this study, we have access to a database of real images from 27 airports with their associated metadata. However, the images of the same airports tend to be quite similar, showing the same runways from close points during the flight phases, resulting in limited diversity within the real data. Labeling is performed automatically based on the GPS positions of the aircraft and the runways, leading to potential errors in labels due to factors like miscalibration or cloud occlusions.

In contrast, we generate our own synthetic dataset using the commercial flight simulator X-Plane 12, which allows us to specify the aircraft's position, orientation, and various environmental conditions such as weather and time. This customization capacity enables us to collect diverse images from 207 airports worldwide. One of the key benefits of synthetic images is the ability to achieve automatic and accurate labeling of bounding boxes and other parameters that typically require human input in real-world scenarios. We use a custom pipeline rather than relying on existing datasets because it enables the generation of a large volume (>5,000) of high-quality images, across a wide variety of airports and under specific environmental conditions.

Taking advantage of this generation capacity, we want to trial synthetic data and assess if using simulated images enables a model to be robust under unseen conditions. To do so, we use the nighttime scenario. At night, runway appearance changes substantially: the surface texture visible in daylight is replaced by the runway edge and threshold lighting systems, which appear as rows of white and green lights against a dark background (Figure 5.1). The overall geometry remains unchanged, but the low-level texture and color statistics differ markedly from daytime images, making nighttime a controlled testbed for out-of-domain generalization. With this idea in mind, we generate nighttime synthetic data from the same airports as the daytime ones.

In Figure 5.1 we have examples of real and synthetic images for the runway detection task.

Mixing strategies In this work, our objective is to assess the use of synthetic data for runway detection. We also want to study the impact of synthetic data in unseen situations. To this end we design several training strategies, applicable for both the nominal and the nighttime studies.

We explore the utility of synthetic data by employing various configurations of real and synthetic datasets. Our study begins with a baseline experiment using only



(a) Real runway. (b) Real runway at night.

(c) Synthetic runway. (d) Synthetic runway at night.

Figure 5.1: Examples of real data and synthetic data from X-Plane 12 for runway detection. Notice the difference in overall textures for day scenes, especially for the sky, and the lighting intensity for nighttime scenario.

real images of runways (REAL), allowing us to establish a performance benchmark based solely on real data.

Next, we create a synthetic dataset using a flight simulator software (SYNTH). For all datasets, we ensure that each airport is represented by the same number of images, preventing imbalances in both training and validation sets.

To integrate both real and synthetic data during training, we assess different mixing strategies. The standard approach involves randomly sampling from the combined dataset (MIX), resulting in a variable ratio of synthetic to real data at each training step, dependent on the dataset composition.

To mitigate the domain shift between synthetic and real data, we also examine the CARE [Prabhu et al. (2023)] domain adaptation method (CARE) tailored to our use case. This method introduces an additional loss term during training, encouraging the detection model to represent real and synthetic objects similarly by minimizing the Euclidean distance between their feature maps.

Furthermore, the CARE framework employs a specific sampling strategy that ensures an equal number of synthetic and real images are used at each training step in the minibatch. While this sampling method is necessary to their loss terms, we also investigate its effectiveness independently, without the accompanying loss terms (SAMPLER).

5.2.4 Experimental setup

In this section we report our model overview, our implementation details and the evaluation modalities.

Model overview We consider a standard object detector, Faster R-CNN [Ren et al. (2017)], to follow the recent literature in domain adaptation for object detection [Prabhu et al. (2023); Yu et al. (2022)] and previous works in airport runway detection [Wang et al. (2022b); Lindén et al. (2021); Li et al. (2024a)].

The objective function used in this study, which the model tries to minimize, is the standard Faster R-CNN loss function, ℓ_{F-RCNN}. For both the RPN and the head we compute the sum of a classification loss and a regression loss. The RPN loss is defined as follows:

$$\ell_{RPN} = \ell_{RPN\ cls} + \ell_{RPN\ reg}. \tag{5.1}$$

$\ell_{RPN\ cls}$ is the proposals classification loss, which computes the cross-entropy between the proposals and ground truth to help the RPN learn to distinguish between foreground (object) and background (non-object) regions. $\ell_{RPN\ reg}$ is the proposals regression loss, which computes the smooth L1 loss between the proposals coordinates and the ground truth boxes coordinates to help the RPN producing better localized and shaped proposals.

Similarly, the head loss ℓ_{head} computes a classification loss and a regression loss, but on the head final boxes predictions instead of the RPN proposals. In the end, the loss function to minimize is the sum of both RPN and head losses:

$$\ell_{det} = \ell_{RPN} + \ell_{head} \tag{5.2}$$

In the experiments using CARE, we add to the standard loss the alignment term proposed. CARE also incorporates reweighting factors that are unnecessary in our monoclass setting where object size and aspect ratio are consistent. We do not include them in our loss function. The overall training loss then becomes:

$$\ell_{det} = \ell_{F-RCNN} + \lambda \ell_{align}. \tag{5.3}$$

λ is a balancing factor, which controls the contribution of the alignment loss in the overall loss. ℓ_{align} is the alignment loss proposed by CARE, a cycle loss computing the difference between features of synthetic and real objects.

Implementation details For all the experiments using the real dataset, our data being redundant, we use a small training dataset of 1,000 images from 19 airports. Then, for the simulated dataset we generate 10,000 images to keep a reasonable ratio of synthetic to real images. Those images are from 199 airports, including the ones in the real dataset.

In order to study the robustness to domain shift of each model, we introduce the nighttime condition. For this purpose, we generate another synthetic dataset composed of 5,000 daytime images and 5,000 nighttime images from the same 199 airports.

Our detection model is a Faster R-CNN with FPN and a ResNet50 backbone, pretrained on COCO2017 [Lin et al. (2014)]. The training configuration includes a

	REAL	SYNTH	MIX	SAMPLER	CARE
DAY AP	58.60	59.04	**65.25**	64.01	63.66

Table 5.1: Values of AP for each mixing strategy over daytime validation datasets. Models were trained *without* synthetic nighttime image.

	REAL	SYNTH	MIX	SAMPLER	CARE
DAY AP	58.60	57.55	66.77	64.73	**67.50**
NIGHT AP	15.21	43.22	43.38	42.00	**43.75**

Table 5.2: Values of AP for each mixing strategy over daytime and nighttime validation datasets. Models were trained *with* synthetic nighttime image.

batch size of 8, a learning rate of 0.002, and is trained during 10,000 iterations. We use $\lambda = 0.1$ for the CARE loss (Equation (5.3)). We perform the validation at the end of the training.

Evaluation Following previous work [Yu et al. (2022); Lindén et al. (2021)] we report COCO Average Precision (AP) for each strategy, AP being the average precision over several Intersection over Union values, from 0.5 to 0.95 by 0.05 steps.

To assess the training methods and the impact of training datasets, we build an evaluation dataset of 200 real daytime images from 8 airports. Those airports are not represented in any training set. In order to evaluate robustness to the nighttime condition, we build an additional validation dataset of 200 real nighttime images from the same 8 airports. Real nighttime images are never seen during training.

5.2.5 Results

Comparison of training strategies In this section, we evaluate models trained on daytime images only. Table 5.1 compares the results of the different training strategies and datasets on the daytime validation set. We observe that all methods outperform the model trained on real data, including the synthetic only experiment (+0.44%). We attribute this behavior to the model's pretraining; having already seen real images in COCO2017, it acquired features relevant to real images before the synthetic training. In this setting, adding synthetic data into the training dataset yields better performance on the validation dataset, and a simple method such as MIX brings the best results (+6.65% compared to REAL only). This conclusion was expected, as bringing more data with reliable labeling should lead to a better model.

Results on night domain shift We analyze the results of models integrating nighttime images in their synthetic training set, compared to the same models trained on daytime images only.

In Table 5.2, we first compare the models performance on the daytime validation dataset. The REAL model keeps the same training set throughout all experiments and therefore gets the same results. The synthetic only model SYNTH suffers from the addition of nighttime condition. On the contrary, while we expected the same results from the other training strategies, MIX (+8.17%) and especially CARE (+8.90%)



(a) Ground truth (b) REAL (c) SYNTH



(d) MIX (e) SAMPLER (f) CARE

Figure 5.2: A real image of an airport runway at night from the validation dataset, and the bounding box determined by inference through each model. The REAL model does not predict any bounding box, while all other models correctly predict the bounding box covering the runway.

benefit from the diversification of conditions. On average SAMPLER attains the same results, showing little use of the nighttime images. Overall all mixing methods still outperform the REAL experiment. The CARE method exhibits the best score showing the benefits of this domain adaptation strategy when diversifying the training dataset.

We also evaluate our models over the nighttime validation dataset. The REAL model experiences a significant drop in performance (-43.39%), and is unable to generalize properly on this unseen condition. All the other methods also experience a drop in performance. However, they are all able to maintain an AP score similar to the SYNTH model (-14.33% compared to daytime validation), substantially higher than the REAL model on both validation sets. In this scenario, adding synthetic data maintains an acceptable level of performance, compared to a completely unseen condition. Our customized CARE method still exhibits the best score in this difficult scenario by a small margin.

In Figure 5.2 we show inference results for all models on one validation image. This picture example is representative of the type of nighttime images we can find in the dataset. The REAL model is unable to see the runway whereas all the models trained on synthetic data could detect it.

Mixing synthetic and real data improves robustness to unseen environmental conditions, as shown by the AP scores in both standard daytime and nighttime conditions. Including nighttime synthetic images helps achieve acceptable performance in that difficult setting. In our case, domain adaptation improves performance with the customized loss of CARE allowing the model to generalize better. While

the sampling method introduced in CARE lowers the detection performance, its alignment loss compensates this drop and improves the final results when getting enough diversity in the training dataset.

5.2.6 Conclusion and perspectives

In this work, we studied the use of synthetic data for runway detection, specifically how data sampling schemes and domain adaptation can improve model performance. We find that, in general, synthetic data improves model performance and robustness in adapting to rare conditions.

In our first study on the nominal case of regular daytime conditions we find that synthetic images can improve model accuracy, but the presence of both real and synthetic images is necessary.

Synthetic data can be especially valuable to represent rare cases that are underrepresented in real data, in order to ensure model robustness. We found that synthetic images of nighttime conditions can greatly improve model performance on runway detection at night. Even training a model on only synthetic data vastly outperforms models trained only on real data. We consider that this is due to the lack of real nighttime images and the difficulty of adapting runway detection from daytime to nighttime in the absence of nighttime examples.

Domain adaptation seems to be a promising approach to enhance runway detection models. While the benefits of CARE compared to a simple mixing approach seem modest, the customized loss helps overcoming the decline introduced by the sampler, and CARE achieves the best performance overall.

In a future direction we aim to tackle other adverse conditions, such as snowy or foggy weathers that are more complicated to capture in reality or render in simulation. It would also be of interest to explore the impact of the CARE sampler, to further improve the benefits of this domain adaptation approach.

In summary, we show that synthetic data can aid in the task of airport runway recognition. Images from simulation software are especially useful in the representation of rare cases, such as nighttime data. By including synthetic data from diverse cases, we can ensure and improve computer vision model robustness, especially with the help of our customized CARE domain adaptation strategy.

On small scale datasets, it can be difficult to assess models performance and improvement over baseline. Indeed, validation on 200 images allows a great variance on the final results. In Section 5.3 we address this limitation by scaling up the experimental setup significantly. The synthetic training set grows from 10,000 to 20,000 images per condition. The real model is trained on 15,000 labeled real images rather than 1,000. Real validation sets of 8,000 images for day and night and 5,000 for fog replace the 200-image sets used here. Dedicated test sets for fog and snow strengthen the evaluation of generalization to unseen airports.

The two studies also differ in their downstream detector. A lightweight one-stage keypoint model replaces Faster R-CNN, making the pipeline more suitable for embedded real-time use. Moreover, unlike the COCO-pretrained Faster R-CNN used in this section, it is trained without pretraining on external large-scale datasets, placing the experiment closer to industrial deployment constraints where pretraining may not be possible. In Section 5.3, we apply the pixel-space domain adaptation methods of Chapter 3 to this expanded runway detection setting. Such methods are

rarely applied to runway detection in the literature, mainly due to the near absence of open real data. Thus, this study is one of the first to evaluate pixel-level style transfer for runway detection on an industrial dataset.

5.3 CACTI$_\mathbf{F}$ application to runway detection

This section covers the article *Enhancing Runway Detection Models with Synthetic-to-Real Style Transfer* in *Digital Avionics Systems Conference, September 2026* [Chigot et al. (2026a)].

5.3.1 Motivation

One key advantage of style transfer is its independence from the downstream model architecture. Once translated images are generated, any detection model can be trained on them, and the inference pipeline remains unchanged. This is attractive for embedded avionics systems, where runtime and memory must remain limited.

In Section 5.2, the CARE approach was tied to Faster R-CNN [Ren et al. (2017)], a two-stage detector that is not fit for real-time use because of its inference time. Here, we study whether the style-transfer methods introduced in Chapter 3 can also help runway detection in a more practical setting with a lighter detector family. Furthermore, unlike the COCO-pretrained Faster R-CNN of Section 5.2, the detector here is trained without pretraining on external large-scale datasets, placing the experiment closer to industrial deployment conditions. The problem is also different from the urban-scene experiments of Chapter 3. Runway images are of higher resolution, the runway can be very small at long distance, and real adverse-condition images can be scarce. Moreover, images contain only one or two objects of one class compared to the dense scenes and 19 classes of Cityscapes.

Open runway datasets often lack real data, but adverse conditions are also poorly covered either in their synthetic or real version. Few datasets provide detailed weather metadata, and even fewer contain enough real night, fog, or snow images for a full study. This makes runway detection in an industrial setting useful for testing whether style transfer can add missing realistic elements to synthetic training data, and whether it can enhance runway detection performance.

5.3.2 Data overview

Real images are captured from aircraft equipped with cameras during landing sequences across diverse airports, following the same collection protocol as in Section 5.2 but at a larger scale; the collection covers four environmental conditions: daytime, nighttime, fog, and snow. However, images from the same airports tend to be similar, as many frames come from the same landing sequence, resulting in limited diversity within the real data. Labeling is performed automatically from flight metadata, deriving runway bounding boxes from the aircraft's onboard data. Synthetic images are generated with the X-Plane 12 flight simulator following the protocol of Section 5.2; for this section, 20,000 synthetic images are generated for night and day conditions.

To obtain comparable real and synthetic subsets, all images comply with specific selection filters. First, images are restricted to an along-track distance between -2200 and -250 m, corresponding to the final approach phase where the runway is large

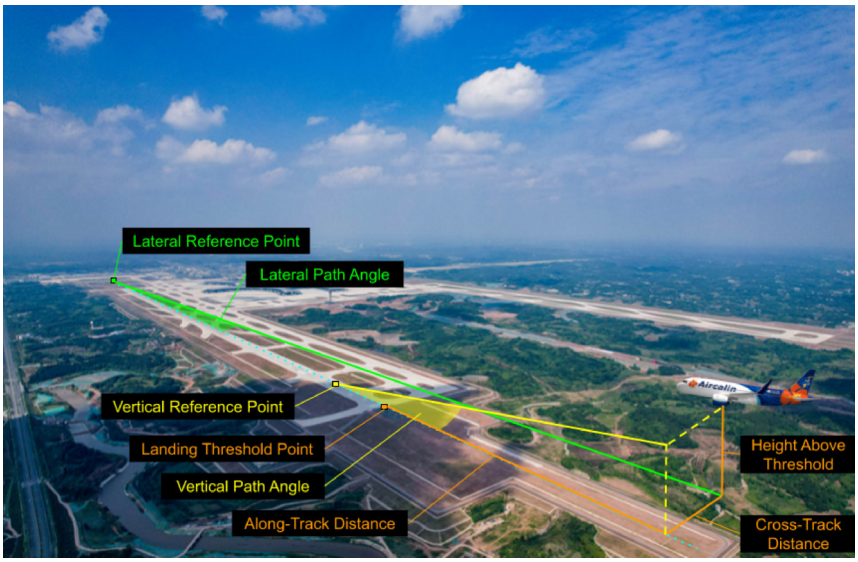


Figure 5.3: Representation of the runway coordinate system.

enough to be visible. In Figure 5.3 the along-track distance is represented in orange. This variable represents the longitudinal distance between the aircraft's position and the landing threshold point. Its value is denoted *negative* before landing and *positive* after. Second, images must match the requested environmental condition (daytime, nighttime, fog, snow). Third, a Kernel Density Estimation (KDE) weighting is applied to reduce distributional differences between synthetic and real images. A Gaussian kernel is fitted on the real dataset's aircraft kinematics parameters (position relative to the runway threshold, roll, pitch, yaw, attitude of the aircraft). Weights are then assigned to each synthetic image based on its similarity to the real-world distribution. Finally, a weight threshold can be applied to the synthetic data to achieve an optimal balance between quantity and similarity with real data.

Classical urban scene segmentation benchmarks such as Cityscapes are predominantly daytime. Adverse-condition datasets like ACDC provide real nighttime images but no synthetic nighttime ones. Unlike the GTA5 to Cityscapes setting of Chapter 3, our flight simulator can generate daytime and nighttime synthetic data, allowing study of appearance transfer across varying environmental conditions.

Table 5.3 summarizes the datasets covered in this section. Real images are split into four non-overlapping subsets by airport: style images (used only for style transfer); a validation set; a held-out industrial test set from a distinct database sharing no common airports; and real training images used as baseline. This airport-level separation ensures that performance reflects generalization to unseen locations rather than memorization of specific runways. This is coherent with urban driving benchmarks, where validation and test sets cover distinct cities [Sakaridis et al. (2021); Cordts et al. (2016)]. The validation and test sets have different structures: the validation set contains sparse samples across many airports and runway configurations, while the test set comprises complete landing trajectories over a smaller set of airports with more common geometries and configurations. Here, fog and snow have no synthetic datasets: for these conditions, the style transfer operations start from daytime synthetic images and translate toward real foggy or snowy references.

Snow is the most constrained condition, with only 15 real images available for

Condition	Synth (train)	Real (style)	Real (val)	Real (test)	Real (baseline)
Day	20,000	1,000	8,000	54,371	15,000
Night	20,000	1,000	8,000	26,852	15,000
Fog	—	1,000	5,000	2,564	—
Snow	—	15	—	964	—

Table 5.3: Dataset summary by condition. Style images are used as references for style transfer only and are excluded from all training, validation, and test sets; style-transferred models use only the style subset. The real baseline trains on real images.

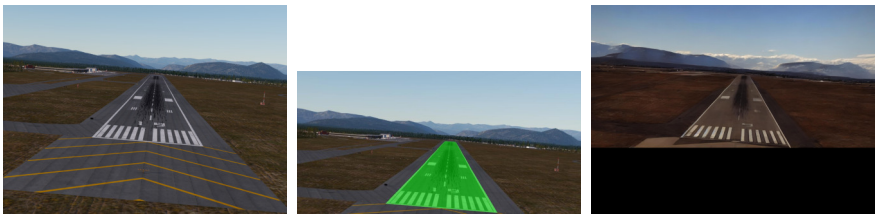


Figure 5.4: Processing steps for runway style transfer: original synthetic image (left), cropped low-resolution input for diffusion (center), translated image (right).

style references, no validation set, and a test set from only two trajectories. This severely limits the snow study and illustrates a recurring challenge: the rarest and most safety-critical conditions tend to be the ones with the least real data.

Overall, the training set sizes are constrained by the computational cost of style transferred dataset generation and model training. We use 20,000 synthetic images and 15,000 real images per condition for training. Test sets are kept large (over 54,000 daytime and 26,000 nighttime images) to provide reliable estimates of detection performance.

From the runway outline, we build binary runway/background masks for the style-transfer pipeline. The original 12 MP images cannot be processed directly: Stable Diffusion is trained at a 512×512 resolution, and GPU memory requirements scale quadratically with image size, making full-resolution inference impractical. The images are therefore resized and cropped to 1024×512 before transfer. After generation, the translated images are padded and resized back to 4096×3000 so that original runway location metadata and input images remain compatible with the downstream detection pipeline. Figure 5.4 displays this process.

The generated images are used to train the detection model, while real style examples are excluded from the learning process. Therefore, real images used as style references are not present in any training, validation or testing dataset.

Overall, the industrial setting makes dataset construction less controlled than standard benchmarks. Metadata filtering, environmental selection, and disparate availability across conditions make fair and systematic evaluation difficult.

5.3.3 Experiments

We design four transfer scenarios to study the effect of style in synthetic data, each corresponding to a different target domain. Table 5.4 summarizes the content and

style inputs for each scenario. Fog and snow have no synthetic counterpart, so both use daytime synthetic images as content; style transfer must bridge not only the synthetic-to-real gap but also the weather shift. Snow is the most data-limited case, with 15 style images from two landing trajectories only.

Scenario	Content (synthetic)	Style (real)	Style images
Day → Day	Synth day	Real day	1,000
Night → Night	Synth night	Real night	1,000
Day → Fog	Synth day	Real fog	1,000
Day → Snow	Synth day	Real snow	15

Table 5.4: Transfer scenarios. Each scenario defines the synthetic content images and the real style references used for style transfer.

To increase alignment between style and content, style images are paired with synthetic content images according to along-track distance. This keeps the apparent runway size roughly similar in both images and avoids extreme mismatches.

For the downstream detection task, we train a lightweight, one-stage deep learning architecture specifically optimized for high-throughput embedded deployment on avionics-grade hardware. The network predicts keypoints describing the runway edges, from which we reconstruct a bounding box used for evaluation. While these keypoints are designed to support a secondary camera pose estimation task, consistent with future vision-based navigation requirements, that specific application remains outside the scope of this chapter. Each detector is trained from scratch for 60 epochs on a single translated dataset corresponding to one environmental condition. No mixing between conditions is performed. The daytime model sees only daytime images, the nighttime model only nighttime images, and so on. No real images are added to the training set alongside the translated synthetic ones. We also train a baseline model on the original synthetic data without style transfer. Validation is carried out on the corresponding real dataset for each environmental condition (daytime, nighttime, fog). The snow condition does not include a validation dataset. Final testing uses an undisclosed industrial dataset composed of real images from airports absent from training or validation, containing 54,371 images for daytime, 26,852 images for nighttime, 2,564 images for fog and 964 images for snow. We report Average Precision (AP) on the bounding boxes reconstructed from the predicted keypoints.

5.3.4 Qualitative results

Daytime First, we visually compare each generative method (Cross-Image Attention, CACTI, CACTI$_F$) on example images.

Figure 5.5 shows representative daytime transfers obtained with Cross-Image Attention, CACTI, and CACTI$_F$. Overall, colors and global textures are transferred well. At close to medium distance, CACTI improves the runway appearance, which is consistent with the analysis from Chapter 3. This is due to Class-wise AdaIN, which transfers feature statistics inside the runway mask instead of using global statistics over the full image. In the first example, Cross-Image Attention borrows the road texture located before the runway, while CACTI restores a more plausible runway

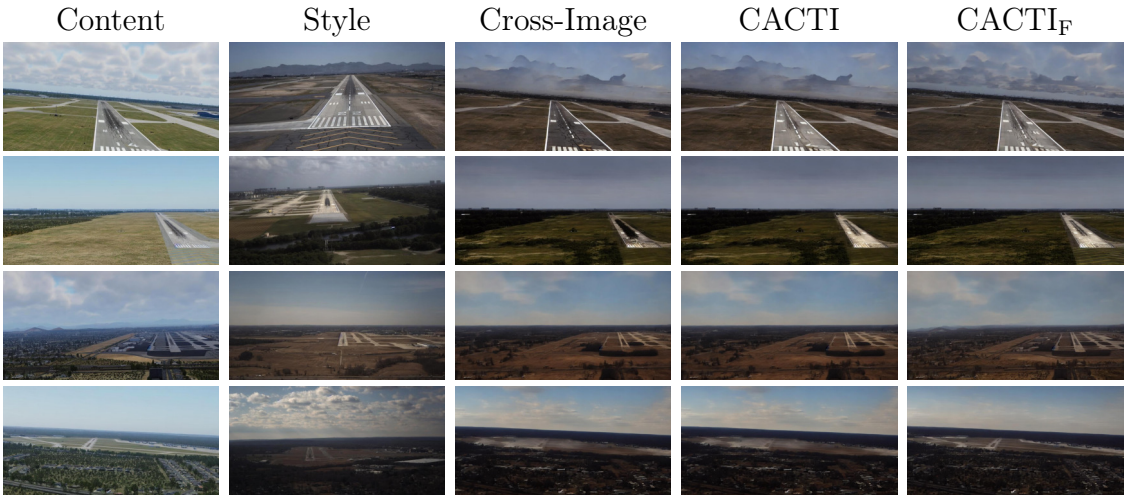


Figure 5.5: Daytime style transfer results. Each row shows the synthetic content image, the real style image, and the transfer results from Cross-Image Attention, CACTI, and CACTI$_F$.



Figure 5.6: Lack of meaningful correspondence failure case. All three methods produce an incoherent transfer when content and style images lack comparable scene elements. CACTI and Cross-Image Attention change the water body to land, CACTI$_F$ slightly mitigates the effect.

surface. In the second example, the runway color is again better transferred by CACTI. The limits of the approach are also visible. Runway markings remain difficult to preserve for both Cross-Image Attention and CACTI, and can lose precision or disappear. At long distance, the runway occupies only a small part of the image. The runway mask then becomes very small in the latent space, so the class-wise correction has much less effect and CACTI behaves almost like Cross-Image Attention. In these cases, the filtering stage of CACTI$_F$ retains the original content features when the style-content match is weak, suppressing part of the generated artifacts while keeping the transferred appearance. From close to long range, CACTI$_F$ is the most content-faithful method. The restoration of the pre-landing area is visible in the third image. Content preservation is most apparent in the last example, where the runway nearly disappears with Cross-Image Attention and CACTI but remains recognizable with CACTI$_F$.

Figure 5.6 illustrates a foreseeable failure case. The synthetic content image contains a large and bright water area. On the other hand, the style image contains a very limited amount of darker water. As a result, the transfer replaces this region with a land-like appearance because the attention mechanism cannot establish a meaningful match. This behavior is consistent with the analysis of cross-image attention in Chapter 3. When content and style images do not share comparable objects or scene composition, the attention maps cannot establish adequate correspondences and the transfer becomes incoherent.



Figure 5.7: Nighttime style transfer results. Each row shows the synthetic content image, the real style image, and the transfer results from Cross-Image Attention, CACTI, and CACTI$_F$.

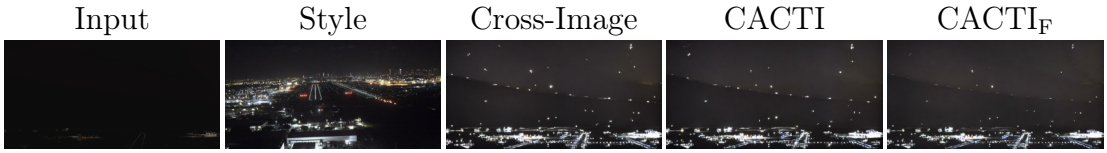


Figure 5.8: Nighttime failure case. All three methods produce an incoherent transfer when the style image contains very strong lighting elements that are absent from the content image. The CACTI$_F$ filtering module removes part of the star artifacts but cannot fully restore a realistic sky.

Nighttime The nighttime examples in Figure 5.7 show a different behavior. Because the runway texture is barely visible at night, Cross-Image Attention, CACTI, and CACTI$_F$ produce very similar outputs. In this case, the main contribution of style transfer is not texture refinement but lighting transfer. Runway and environmental lights become much more visible, and the runway shape is easier to recognize. Global color is also transferred, for example through the reddish tone visible in the first example.

The filtering stage of CACTI$_F$ is still useful for suppressing local artifacts. In the second nighttime example, the cross-attention mechanism tries to transfer an orange sky without enough semantic guidance, which produces a large colored artifact. The CACTI$_F$ filtering module removes this inconsistent region and restores a more stable rendering. In the last example, CACTI suppresses the synthetic centerline lights, which are absent from the style image, while Cross-Image Attention preserves them. This case highlights a difficulty specific to nighttime: runway lights are simultaneously a structural cue and an appearance feature, so the boundary between content and style is less clear than during the day.

In Figure 5.8, the transfer amplifies faint stars in the synthetic sky into bright spots resembling runway lights. The style image also has a heavily lit building at the bottom, which modifies the background lighting statistics and results in over-illumination throughout the scene. With a binary mask, CACTI treats sky, ground, and buildings as the same region, so none of these artifacts can be corrected. More semantic classes in the mask (sky, building, ground) would be a straightforward way to address this.

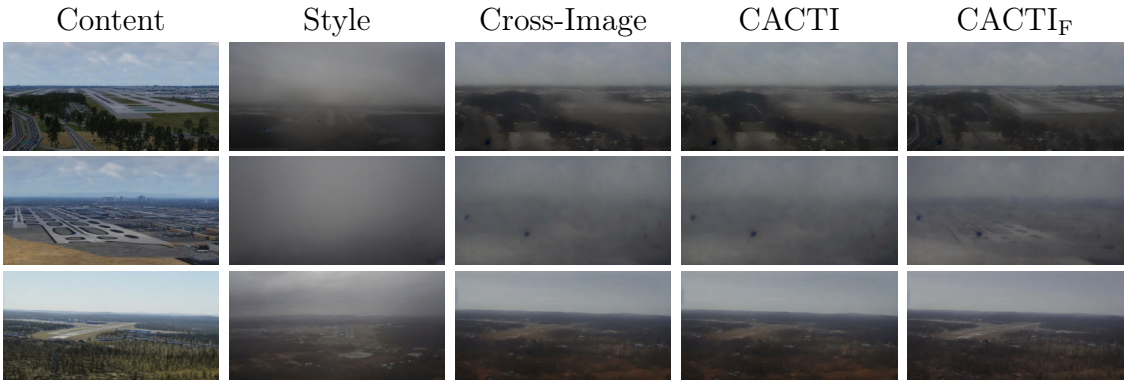


Figure 5.9: Fog style transfer results. Each row shows the synthetic content image, the real style image, and the transfer results from Cross-Image Attention, CACTI, and CACTI$_F$.

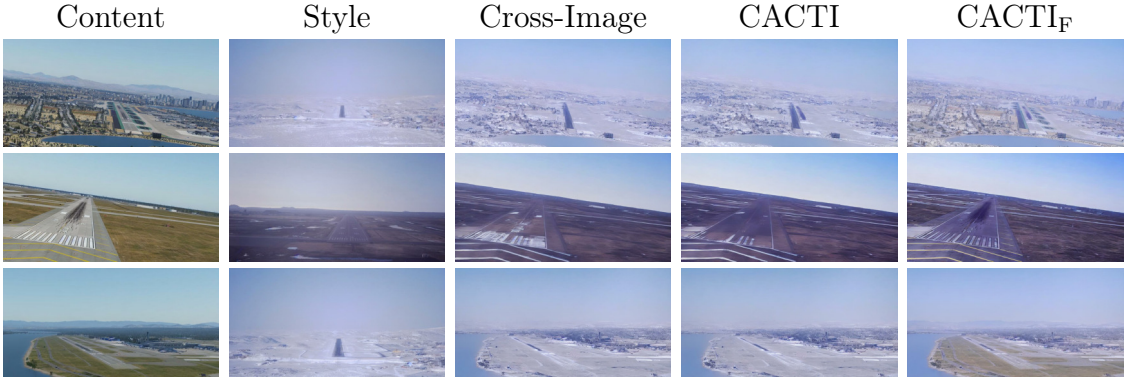


Figure 5.10: Snow style transfer results. Each row shows the synthetic content image, the real style image, and the transfer results from Cross-Image Attention, CACTI, and CACTI$_F$.

Fog and snow Figure 5.9 illustrates style transfer toward foggy images. Cross-Image Attention transfers the fog appearance very strongly, to the point that the runway becomes almost impossible to see in the translated image. Although this gives a realistic fog effect, it is not suitable for supervised training because the labeled object is nearly erased. Here, CACTI produces results very similar to Cross-Image Attention. In contrast, CACTI$_F$ keeps part of the runway structures while still adding haze. The results are less stylized, but better preserve the supervision signal. In the second example, black dot artifacts from the diffusion process remain visible despite CACTI$_F$.

The snow examples in Figure 5.10 follow the same pattern. Cross-Image Attention transfers the snow appearance strongly, and even erases the runway completely on the first image. CACTI$_F$ applies a lighter transformation that reconstructs the runway more accurately while still conveying a snowy appearance. CACTI produces an intermediate result between Cross-Image Attention and CACTI$_F$.

Fog and snow reveal a broader limitation, especially important for CACTI$_F$, which highlights differences between style and content. Severe weather does not only change the color or texture of an image, but also changes what is visible in the scene. This creates a tradeoff for style transfer. If the translation stays too close to the original content, the weather effect is too weak. If it follows the style too strongly,

the runway may disappear from the generated image. In that case the bounding box label could become incorrect. A direct way to reduce this problem would be to render fog or snow directly in the flight simulator. Style transfer would then only need to refine the appearance rather than create weather structures from scratch.

Overall, the cross-attention mechanism transfers the appearance of real images effectively. The class-wise AdaIN module improves texture and color when the runway remains visible. However, with only a binary runway-or-background segmentation, the gain remains limited. At long range, the runway becomes too small. CACTI then behaves almost like Cross-Image Attention. In contrast, CACTI$_F$ filters artifacts from the initial transfer, which helps the style transfer process when cross-attention layers propose incorrect correspondences. Richer semantic masks from the simulator would likely improve realism when using class-wise AdaIN. Adding classes such as *sky*, *ground*, *water*, or *building* would require a small amount of additional annotation on real images, but could substantially improve realism.

5.3.5 Quantitative results

We now evaluate the impact of style transfer on downstream runway detection performance. We first quantify the synthetic-to-real domain gap, then compare training strategies on nominal and adverse conditions using both validation and held-out test sets.

Domain gap quantification To quantify the domain gap, we build specific training and validation datasets where synthetic and real images share the same airports and weather conditions. Thus, synthetic and real images are chosen to be highly similar. Airports are identical as well between the training and validation datasets. This isolates the synthetic-to-real domain gap from other domain shift. We evaluate three protocols. The first uses synthetic images for both training and validation (Synth $\rightarrow$ Synth). The second uses synthetic training images and real validation images (Synth $\rightarrow$ Real). The third uses real images for both training and validation (Real $\rightarrow$ Real). Results are reported in Table 5.5, using 32K training images and 8K validation images.

	Synth $\rightarrow$ Synth	Synth $\rightarrow$ Real	Real $\rightarrow$ Real
DAY AP	90.30	46.36	97.66

Table 5.5: Domain gap quantification. Average Precision (AP) is evaluated on 32K training and 8K validation images. The gap between synthetic-to-synthetic and synthetic-to-real highlights the domain shift.

Performance drops by more than 40 points when moving from synthetic to real validation, while the model trained on real data reaches 97.66 AP. This large gap confirms that, despite the visual quality of simulator images, the domain shift is substantial and motivates the use of style transfer to bridge it.

Nominal daytime condition Table 5.6 reports AP on the daytime validation set for each training configuration stated in Table 5.3. The configurations are unaltered synthetic data (Synth), synthetic data translated with Cross-Image Attention,

CACTI, or CACTI_F, and real data (Real). The real model is trained on 15K labeled real images. The style-transfer models only use real images as style references and do not train on them. The real scenario therefore represents an upper bound rather than a competing method.

	Synth	Cross-Image	CACTI	CACTI_F	Real
DAY AP	27.12	26.36	27.12	**28.37**	53.73

Table 5.6: AP for daytime validation. Among synthetic or style-transferred datasets, CACTI_F achieves the best score by a small margin.

The real model reaches 53.73 AP, confirming the advantage of labeled real data when training and validation share the same domain. All synthetic-based models produce significantly lower AP, highlighting the domain gap. Among the transfer methods, CACTI_F achieves the highest AP (28.37), slightly above the untransferred synthetic baseline (27.12) and Cross-Image Attention (26.36). While this improvement is small, it shows that CACTI_F preserves the content structure better than the other methods.

Figure 5.11 shows inference results on a representative daytime image for three training configurations. The real model produces no bounding box on this example. The synthetic model misses one runway and produces a spurious detection. The CACTI_F-trained model detects all runways, but with low confidence scores (0.10 – 0.13) and imprecise bounding boxes.

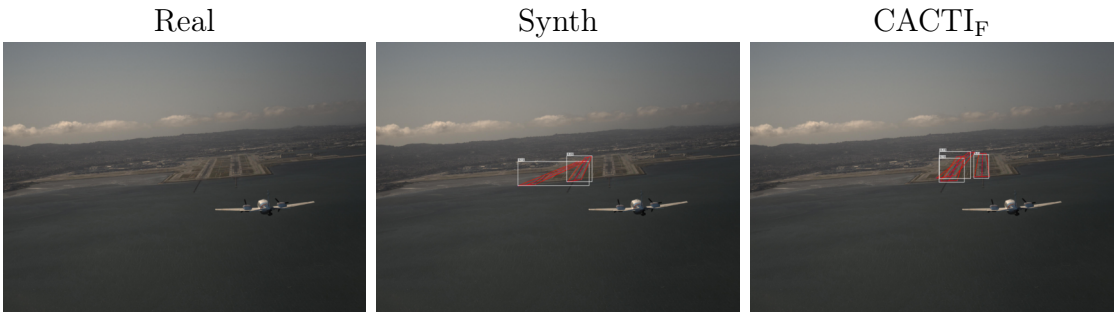


Figure 5.11: Daytime inference examples. From left to right: predictions from the models trained on real data, synthetic data, and CACTI_F-translated data. The real model produces no detection, the synthetic model misses one runway and adds a false positive, while the CACTI_F model detects all runways, though with low confidence and imprecise boxes.

Adverse conditions Table 5.7 reports AP on the nighttime and fog validation sets. The two conditions use different training protocols. For nighttime, all models are trained on nighttime images only. The real model trains on real night images, the synthetic baseline on synthetic night images, and the style-transferred models on synthetic night images translated toward real night images. For fog, the setup is different because no synthetic fog images exist in our dataset. The real and synthetic baselines both train on daytime images and are evaluated directly on foggy images.

	Synth	Cross-Image	CACTI	CACTI$_F$	Real
NIGHT AP	58.83	56.04	56.53	**58.94**	63.18
FOG AP	34.27	31.77	28.92	**36.98**	77.09

Table 5.7: AP on adverse-condition validation sets (nighttime and fog). CACTI$_F$ consistently achieves the best score among transferred methods. However, the gain against plain synthetic data is minimal.

The style-transferred models also start from daytime synthetic images, translated into fog using 1,000 real foggy images as style references.

At nighttime, the gap between methods narrows. Darkness hides most textures and leaves mainly lights and silhouettes, making the task simpler for all models. Even the synthetic baseline (58.83) approaches the real scenario (63.18). Among style-transferred models, CACTI$_F$ reaches 58.94 AP, above Cross-Image Attention (56.04) and CACTI (56.53). Nighttime transfer mainly modifies lighting rather than texture, which limits the benefit of class-wise corrections.

On fog, the real model achieves 77.09 AP, far above all synthetic-based methods. Among them, CACTI$_F$ stands out with 36.98 AP, above the synthetic baseline (34.27). As observed qualitatively, Cross-Image Attention erases the runway under heavy fog, while CACTI$_F$ preserves more of its structure, leading to better supervision during training.

Test set evaluation Table 5.8 presents AP measured on the held-out industrial test set, which contains airports absent from all training sets. This provides a more reliable evaluation of generalization.

	Synth	Cross-Image	CACTI	CACTI$_F$	Real
DAY AP	48.95	81.59	85.63	**87.97**	91.03
NIGHT AP	62.57	**80.17**	75.87	79.53	86.27
FOG AP	49.22	70.68	72.08	**75.58**	82.85
SNOW AP	9.49	**56.88**	46.30	27.45	15.42

Table 5.8: AP on the held-out industrial test set across all conditions. Style transfer substantially improves over synthetic-only training in all cases. For the fog and snow experiments, the synthetic and real datasets contain only daytime images, as the original training sets include no foggy or snowy images. These conditions are introduced exclusively through the style-transferred datasets.

First, performance on the test set is higher than on the validation set. The test set contains fewer airports and focuses on more standard configurations, making it less diverse and easier than the validation set, which sparsely samples a larger number of airports with greater variability.

On the daytime test set, style transfer shows the largest gains: CACTI$_F$ achieves 87.97 AP, compared to 48.95 for the synthetic baseline (+39.0 points), and is only 3.1 points below the real model (91.03). Cross-Image Attention (81.59) and CACTI (85.63) also bring substantial improvements.

At nighttime, Cross-Image Attention performs best with 80.17 AP, closely followed by CACTI_F (79.53), while CACTI reaches 75.87. All three methods substantially surpass the synthetic baseline (62.57), with the real baseline at 86.27 AP.

On the fog test set, CACTI_F achieves 75.58 AP, a substantial improvement over the synthetic baseline (49.22, +26.36 points) and the best score among all transferred methods. Even Cross-Image Attention (70.68) and CACTI (72.08) improve considerably over the baseline, confirming that style transfer benefits the detector in fog condition. The gap with the real scenario (82.85) narrows to only 7.27 points, a notable result given that the real model has access to 15K labeled real daytime images, even if the input dataset does not contain the target weather condition.

Snow is the most challenging condition and the only one where style-transferred models surpass the real model. No training set contains snow images: both the real and synthetic baselines are trained on daytime data only. The real model reaches 15.42 AP and the synthetic model 9.49. Using only 15 real snow images as style references, Cross-Image Attention achieves 56.88 AP, CACTI 46.30, and CACTI_F 27.45. All three surpass both baselines by a large margin. The ordering reflects the aggressiveness of each method. CACTI_F suppresses features when style and content differ too much, which limits snow transfer, as seen in Figure 5.10. CACTI does not apply this filtering and transfers more snow appearance. Cross-Image Attention applies the strongest transformation and scores highest. This scenario highlights the potential of style transfer to inject new environmental conditions into training, even with a very limited number of real images. However, the ordering contradicts earlier experiments, where CACTI_F outperformed CACTI under heavy weather by preserving runway structure. Since the snow test set contains only two landing trajectories, this discrepancy may be explained by the limited diversity and size of the evaluation data. This result illustrates that reliable evaluation requires a sufficient amount of diverse real data, at least for testing.

Real-to-real transfer As a complementary experiment, we apply Cross-Image Attention and CACTI_F in a real-to-real setting, translating daytime real images toward foggy real images. Both the real model and the style-transferred models start from the same real daytime images. If style transfer is useful here, the transferred models should outperform the real scenario. Results on the validation set are reported in Table 5.9.

	Cross-Image	CACTI_F	Real
FOG AP	72.42	76.83	**77.09**

Table 5.9: Real-to-real fog transfer. Models trained on fog-transferred real images are compared to a model trained on the same daytime images without transfer. Both are evaluated on real foggy images from the validation set.

Neither method surpasses the real daytime scenario. CACTI_F reaches 76.83 AP, marginally below the real model (77.09), while Cross-Image Attention drops further to 72.42 AP. This result suggests that real daytime images are already sufficient to recognize runways in foggy conditions, without any fog augmentation. Applying style transfer on top of real images does not add useful information, and Cross-Image

Attention even introduces artifacts that hurt performance. When evaluated on the validation set, the effectiveness of style transfer as a data augmentation strategy is inconclusive.

5.3.6 Conclusion and perspectives

Synthetic data alone already provide a reasonable starting point. At night, the simulator renders runway lights and silhouettes with enough fidelity that the synthetic baseline performs close to the real model. Fog and snow offer no synthetic equivalent, so models rely on daytime images and are evaluated under a weather shift they were never exposed to during training.

This second study also confirms both the value and the limitations of pixel-space adaptation. Its main practical advantage is independence from the detector. Once the translated images have been generated, the downstream model and its inference cost remain unchanged. This makes the pipeline suitable for embedded systems.

The domain gap quantification confirms that the synthetic-to-real shift is large despite the visual quality of the simulator, motivating the use of style transfer. Among the methods evaluated, CACTI$_F$ is the most consistent: it achieves the best or near-best AP across daytime, nighttime, and fog conditions. It also preserves more runway information than Cross-Image Attention under heavy weather. The real-to-real fog experiment suggests that style transfer does not add useful information when real training images are already available. A detector trained on clear daytime real images already generalizes well to fog, and neither method surpasses the real daytime model.

Several limitations should be noted. On the validation sets, style transfer provides no clear and consistent advantage over the untransferred synthetic baseline. Gains are modest, though the ranking of methods is consistent across all three validation conditions: CACTI$_F$ ranks first among transferred models in every case, reinforcing confidence in this result despite the small absolute differences. Reporting mean and standard deviation over multiple training runs would strengthen the statistical significance of comparisons between methods. The clearest improvements appear on the held-out test sets, suggesting that translated images generalize better to unseen airports with standard configurations.

A second limitation is the object-scale constraint introduced by the along-track distance filter. Images beyond approximately 2,200 m are excluded to keep the runway large enough in the frame. As discussed in Chapter 3, diffusion-based style transfer degrades for small objects because the runway mask covers only a few feature map cells in the compressed latent space, which restricts the study to the close approach phase and excludes longer-range detection scenarios.

Third, each model is trained on a single condition in isolation. There is no mixing across environmental conditions, such as combining daytime and foggy images in training. No real images are added to the synthetic training set either. Section 5.2 showed that both of these forms of mixing can bring substantial gains. In that section, adding synthetic daytime and nighttime data improved daytime AP by up to 8.9 points. Neither strategy is explored here, which likely limits the performance ceiling of all models. The experiments also scale to only 20,000 training images per condition rather than the 40,000 available, because the computational cost of diffusion-based translation constrains the training scale for this study.

Several directions follow from these observations. Richer semantic masks from the simulator would make the class-wise AdaIN statistics more informative. Adding classes such as sky, ground, water, and buildings to the runway segmentation would bring this setting closer to the urban-scene experiments where the method was most effective. It would require a small amount of additional annotation on real images for potential large gains. Rendering fog or snow directly in the simulator, so that style transfer only needs to refine the appearance, could also reduce the tradeoff between realism and label preservation. Geometric alignment or feature warping during diffusion would help preserve runway geometry when style and content images are poorly matched. This could also help with the small markings and details that get lost during style transfer.

Finally, no feature-level domain adaptation strategy is applied to the detector. Section 5.2 with CARE, and Chapter 3 with DAFormer, demonstrated that such strategies can substantially improve performance alongside data-level adaptation, and their combination with pixel-level style transfer remains unexplored.

Including methods from the automated pipeline selection framework of Chapter 4, such as ControlNet-based transfer, would allow more controllable guidance during generation. It could also help preserve runway markings and geometry better. More broadly, style transfer appears most useful when real data are scarce. Similar benefits could be expected for other safety-critical tasks such as foreign-object detection, runway incursion scenarios, or obstacle segmentation, where adverse conditions are difficult to acquire at scale. In foreign-object detection, for example, debris on the runway must be identified under varying lighting and weather, yet real incidents are rare and difficult to reproduce. Runway incursion detection faces a similar problem: real examples of unauthorized vehicles or aircraft on the runway are scarce for obvious safety reasons. In both cases, synthetic-to-real style transfer could augment limited real datasets with realistically styled synthetic images.

Chapter 6

Conclusion

This chapter summarizes the contributions of this thesis and discusses their limitations, open questions, and perspectives. We first recapitulate the three main contributions: the CACTI$_{\text{F}}$ style transfer methods for semantic segmentation domain adaptation, the MOOSS evolutionary optimization framework for style transfer pipelines, and the application of these methods to runway detection for automatic landing. We then discuss possible improvements to both CACTI$_{\text{F}}$ and MOOSS, and reflect on broader observations about when style transfer is beneficial for domain adaptation. An exploration of prompt-based style transfer reveals current limitations of text-guided diffusion for realistic data generation. Finally, we address two recurring themes: the absence of cheap metrics that reliably predict downstream task performance, and the bias introduced by existing benchmarks and foundation models pretrained on urban driving data. We argue for more diverse, industrially relevant benchmarks that remain isolated from foundation model training sets.

6.1 Summary of contributions

This thesis studied synthetic-to-real domain adaptation for object recognition in industrial settings where real labeled data is scarce. We focused on scenarios where target domain data is rare and expensive to acquire, such as adverse weather conditions (night, fog, snow, rain). We made three contributions, each addressing a limitation of the previous one, and the industrial application in Chapter 5 tested the methods on a domain outside standard benchmarks.

In Chapter 3, we proposed two diffusion-based style transfer methods for semantic segmentation pixel-based domain adaptation. Both approaches leverage generative foundation models to transfer the style of real images, making synthetic images more realistic and reducing the sim-to-real gap.

Within the framework of Cross-Image Attention [Alaluf et al. (2024)], CACTI introduces class-wise Adaptive Instance Normalization, which transfers style statistics between objects of the same semantic class rather than applying a global transformation. CACTI$_{\text{F}}$ extends this approach with a selective attention filtering mechanism that suppresses artifacts caused by irrelevant pixel correspondences in the attention layers. Experiments on the GTA5→Cityscapes/ACDC benchmark showed that CACTI achieved the best style fidelity among all compared methods, while CACTI$_{\text{F}}$ improved content preservation. Combined with the DAFormer architecture, CACTI outperformed both the synthetic baseline and existing generative methods such as DGInStyle and DATUM across ACDC adverse conditions. Nighttime was the most challenging condition, due to the large appearance gap between synthetic and real night scenes.

During the development of CACTI$_{\text{F}}$, we observed that style transfer operators (AdaIN, CACTI$_{\text{F}}$, subsequently ControlNet) and simple image transformations (normalization, contrasting) can be combined in different orders, and that these choices significantly affect the performance of style transfer and corresponding domain adaptation. To automate this process, we proposed MOOSS (Chapter 4), a multi-objective evolutionary algorithm that optimizes the sequence of style transfer operations. The problem is formulated as a modified Traveling Salesman Problem with variable-length solutions. To enable rapid evaluation during evolution, we used paired-image metrics (DISTS for content preservation and DreamSim for style similarity) instead of expensive distributional metrics or full model training. The optimization produced a Pareto front of diverse pipelines representing different trade-offs between content preservation and style matching. In practice, the entire optimization took the same computation time as evaluating a single pipeline for full dataset generation. However, the evolved pipelines did not match the downstream segmentation performance of ControlNet alone. This revealed that DISTS and DreamSim, while effective for assessing visual quality, do not reliably predict downstream segmentation performance, a problem we return to in Section 6.4.1.

Finally, we applied these methods to an industrial use case in Chapter 5: synthetic-to-real domain adaptation for runway detection in the context of vision-based automatic landing. This application is one of the few in the runway detection literature that uses real images for model evaluation, representing an industrial setting not covered by existing domain adaptation benchmarks. In a first study, we adapted the CARE feature-level domain adaptation method to the runway detection setting. CARE is designed for industrial settings where labeled data exists in both domains

but in limited quantities, making it a coherent choice for runway data. We showed that mixing synthetic and real data improves daytime detection, and that adding synthetic nighttime data increases nighttime performance with CARE. The day-to-night shift is underrepresented in the runway detection literature, and this study confirmed the value of synthetic data for representing rare conditions. In a second study, we applied CACTI_F to runway detection. We extended the runway datasets to cover more conditions (fog, snow, night) with more reliable validation sets and dedicated test sets. On the held-out fog test set, CACTI_F clearly outperformed the untransferred synthetic baseline, showing that pixel-level adaptation generalizes better to unseen airports. However, style transfer provided inconsistent gains on validation sets, and the binary runway/background segmentation masks limited the effectiveness of class-wise AdaIN.

6.2 Perspectives for the proposed methods

Each of the preceding chapters identified specific limitations and open questions. We now revisit them, discussing architectural improvements to CACTI_F, extensions to the MOOSS optimization framework, and general observations on style transfer for domain adaptation. These directions were chosen because they represent the most direct limitations observed during the experimental work: the artifact problem in CACTI_F, the metric problem in MOOSS, and the practical boundaries of when style transfer offers an advantage over simpler baselines.

6.2.1 Improvements to diffusion-based style transfer

When cross-attention focuses on semantically irrelevant regions (for instance, when a road feature pixel is linked to vegetation), artifacts appear in the generated image. A natural solution is to restrict attention so that each pixel is only associated to pixels of the same semantic class. In the self-attention layer, the attention is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V.$$

During the development of CACTI_F, we considered this class-based filtering after the softmax operation. However, at that stage the attention map is already concentrated on a small number of pixels: filtering either removes too much information, producing visual anomalies, or removes too little, yielding results nearly identical to unfiltered cross-image attention.

A subsequent investigation revealed that the attention map *before* the softmax operation is naturally aligned with semantic class boundaries. For example, for a car pixel, the pre-softmax attention scores produce a natural segmentation of car regions. Applying class-based filtering before the softmax, masking out key-value pairs from different classes, would enforce semantic consistency while the attention distribution is still spread across the image, rather than after it has collapsed onto a few dominant locations. This direction could suppress cross-class artifacts more effectively than the current post-softmax filtering in CACTI_F.

Beyond attention filtering, the feature transfer mechanism itself could be improved. Attention filtering controls which source pixels contribute to the style transfer; the

question of how those features are applied once selected is a separate concern. Class-wise AdaIN transfers style by matching the mean and standard deviation of features within each class. This statistical approach is geometry-agnostic: it works regardless of object shape or position. A more direct alternative would be to copy style features from one object and paste them onto the corresponding object in the content image. However, direct copy-paste fails when the source and target objects have different shapes. Thin-Plate Spline (TPS) warping algorithms [Wang et al. (2026)] address this by deforming the feature map of a source object to match the geometry of a target object, interpolating missing values. This would allow copying the style features of a runway in the reference image and warping them to match the runway geometry in the synthetic image. The approach is more straightforward for objects with simple, well-defined shapes, clear boundaries, and no shared borders between two objects of the same class. That would make runway detection a suitable starting point. For more complex scenes such as GTA5→Cityscapes, one could explore selective warping strategies: applying feature copy-paste only at specific diffusion steps, only for certain objects, or combining warping with statistical alignment for different regions of the image.

The effectiveness of class-wise AdaIN also depends on the granularity of the available segmentation masks. In the runway detection setting, images are segmented into only two classes: runway and background. During the approach phase, the runway occupies a small fraction of the image, so the background class dominates (sometimes exceeding 90% of the image) and class-wise AdaIN becomes nearly equivalent to global AdaIN. Adding semantic classes such as sky, terrain, water, and buildings would make the class-wise statistics more informative and bring the setting closer to the urban driving experiments where CACTI was most effective. Since runway scenes are relatively simple, a moderate number of additional classes (far fewer than the 19 classes in Cityscapes) would be sufficient. This would require a small annotation effort on real images but could yield significant improvements.

Finally, the resolution of diffusion models is a practical constraint. Stable Diffusion 1.5 operates at a default resolution of 512×512 pixels, extended to 512×1024 in our experiments. This is insufficient for industrial cameras with specific aspect ratios and for detecting small or distant objects. Higher-resolution diffusion models would help, but the self-attention computation scales quadratically with image size, making it prohibitively expensive on standard GPUs. Tile-based generation or more efficient attention mechanisms could help with this constraint. In tile-based generation, the image is divided into overlapping patches that are processed independently by the diffusion model and then composited; this reduces memory requirements but risks visual inconsistencies at tile boundaries, which is a challenge for structured scenes such as runways.

6.2.2 Improvements to style transfer pipeline optimization

Chapter 4 showed that the DISTS/DreamSim combination can guide pipeline optimization effectively in terms of visual quality, but the optimized pipelines do not necessarily achieve the best downstream segmentation performance. Finding a metric that is both cheap to evaluate and correlated with downstream performance, typically mIoU, remains the main open problem for MOOSS and for data generation methods more broadly (see also Section 6.4.1). One direction is to design metrics

that explicitly verify whether semantic boundaries are preserved after style transfer. For instance, comparing edge maps (e.g. via Canny edge detection) between the generated and content images could provide a lightweight proxy for label preservation. If object boundaries remain intact after style transfer, the associated segmentation masks are more likely to remain valid. Instead of training a segmentation model from scratch for each candidate pipeline, one could also use zero-shot segmentation models (e.g. SAM [Kirillov et al. (2023)]) to evaluate the quality of generated images. This would avoid the cost of retraining and still give a task-relevant signal. However, zero-shot models are themselves trained on web-scale data and carry the same biases as foundation models: they may perform well on urban driving scenes but poorly on aviation or agricultural imagery. Their reliability as fitness evaluators would need to be checked domain by domain, since their scores may not correlate with downstream performance.

A complementary direction is to change what operators are available, rather than how pipelines are evaluated. Current operators in MOOSS mostly ignore semantic masks. For example, an operator that applies a different color shift to the sky class versus the road class conditions directly on semantic labels and preserves class boundaries by construction, without relying on a structure metric to verify preservation after the fact. The current formulation also maximizes image quality without considering generation cost. In industrial settings, computation time matters. Adding generation time as a third objective would produce Pareto fronts that explore the trade-off between quality and speed, helping users select pipelines that fit their computational budget. Finally, MOOSS optimizes the sequence of operators but uses fixed hyperparameters for each (e.g. style transfer strength, number of diffusion steps). Including these hyperparameters in the search space, such as the diffusion steps at which to apply style operators or the strength of each transformation, would enable finer-grained optimization and potentially produce better-adapted pipelines. This increases the dimensionality of the search, but evolutionary algorithms are well-suited to such tasks.

6.2.3 Observations on diffusion-based style transfer for domain adaptation

Drawing on the experiments in Chapter 3 and Chapter 5, several observations emerge about when style transfer is and is not beneficial. In our experiments, style transfer helped the most when target domain data was limited. When sufficient real images are available, straightforward fine-tuning on those images generally outperforms style transfer. However, for rare conditions such as adverse weather, uncommon viewpoints, or unusual scenes, collecting real data is expensive or dangerous. In those cases, style transfer offers a practical way to generate diverse and realistic training data. In Chapter 3 and Chapter 5, the largest improvements from style transfer appeared for these underrepresented conditions. Moreover, when real data is available but unlabeled, pixel-level and feature-level adaptation are complementary. Style transfer generates labeled images that resemble the target domain, while feature-level methods such as DAFormer or CARE align internal representations across domains. The two strategies can be combined. Style transfer can also reduce how many real images are needed for training, freeing them for validation and testing. This is particularly valuable when total data is limited.

Our style transfer approach relies on a pretrained diffusion model, which could introduce biases from its training data. Methods that generate images without a style reference, relying solely on text prompts or learned priors, are more sensitive to these biases. In zero-style methods, the model must guess the style to apply purely from the text prompt and its prior knowledge of similar images seen during pretraining. When the target domain is well-represented in the pretraining data this can produce reasonable results, for example in artistic domains. When it is not, the model lacks the necessary visual references, and outputs tend to be unrealistic or inconsistent. Diffusion models were originally designed for artistic image generation, and their training data reflects this: they easily produce stylized content but struggle to generate ordinary realistic scenes without explicit visual guidance. Providing a style reference image, as in CACTI$_{\mathrm{F}}$, anchors the generation to actual visual information from the target domain and produces more realistic results. This reduces the model's dependence on its prior knowledge and limits pretraining artifacts. For example, in Chapter 3, DGInStyle's zero-style nighttime generation produces images with a synthetic 3D-rendered appearance, while CACTI with a real nighttime reference produces more realistic results. For industrial domains that are underrepresented in large-scale pretraining data, image-based guidance is still needed.

6.3 Prompt-based style transfer: an exploration and its limits

As part of an internship study, we investigated whether text prompts could replace or complement style reference images for guiding diffusion-based style transfer. Two approaches were examined: prompt-only transfer using Step1X-Edit [Liu et al. (2025)], and prompt-augmented transfer using InstantStyle-Plus [Wang et al. (2024b)] alongside a style image. The goal was to determine whether prompts could improve overall realism, suppress artifacts, or add fine-grained details to generated images. The results showed that the role of the prompt is usually important but limited in a photorealistic sim-to-real context.

When using methods that combine a style image and a text prompt, domain-relevant prompts (e.g. describing the target weather or lighting) had virtually no effect on the output, even with high prompt guidance values. Only prompts describing radically different domains (e.g. "a Lego city") produced visible changes, but at the cost of diverging from the style reference. No fine-grained control over details could be achieved, even with highly detailed prompts exceeding one hundred words. This suggests that the style image acts as a dominant conditioning signal, anchoring the output distribution to the reference's visual content regardless of the text: the model adjusts its generation to match the style image far more than the prompt.

For methods relying solely on the text prompt, we observed three behaviors. When the source image is already near-realistic, the model can improve realism by adjusting illumination or textures, and the prompt provides useful guidance in this easy case. When a drastic stylistic change that simplifies the scene is requested (e.g. converting a scene to pixel art), the model also succeeds because simplification does not require generating new realistic detail. However, in the intermediate case relevant to sim-to-real transfer, where the source image has coherent geometry but lacks realistic textures and lighting, the model fails to generate the missing realistic

details. We attribute this to a mismatch with the model's training distribution, which emphasizes artistic or stylized content rather than ordinary realistic scenes.

This points to a current limitation of diffusion models for training data generation: text prompts alone do not provide sufficiently fine-grained and reproducible control over the applied style. For industrial applications where dataset consistency and condition diversity are critical requirements, a style reference image from the target domain, as used in CACTI_F, is still needed to guide the generation and reduce pretraining biases.

6.4 Reflections on metrics and evaluation

Two broader issues recur throughout this thesis: the lack of cheap evaluation metrics for generated datasets, and the biases of the dominant benchmark.

6.4.1 The absence of a cheap proxy for downstream performance

The only reliable way to assess whether a generated dataset improves object recognition is to train a model on it and evaluate on real data. This requires generating thousands of images and training a detection or segmentation model for each configuration, a process costing several GPU-hours per run. This bottleneck affects every chapter of this thesis. For CACTI_F, both during method development and when benchmarking other methods, every hyperparameter configuration, style transfer setting, or baseline comparison required a full generation-training-evaluation cycle, making the process very long and costly. For MOOSS, where hundreds of pipeline candidates must be evaluated, the cost is even more prohibitive.

Metrics such as FID, CMMD, DISTS, and DreamSim can evaluate image quality or dataset similarity much faster than training a model. However, they do not consistently correlate with downstream performance. In Chapter 4, DISTS and DreamSim successfully guided pipeline optimization toward visually appealing results, but the best-scoring pipelines did not achieve the highest mIoU. This happened for both paired-image metrics (LPIPS, DISTS, DreamSim) and distributional metrics (FID, CMMD). Image quality is subjective, and human judgments do not directly translate into better-trained models. Such metrics may be appropriate for artistic image generation, but they are not necessarily linked to downstream recognition performance. A visually pleasing style transfer may alter textures and colors in ways that confuse a segmentation network, while a less aesthetically appealing transformation may preserve the features that matter for recognition. Conversely, a transformation may score poorly on perceptual metrics while leaving model performance unchanged, since networks respond to feature statistics that do not always align with human visual judgments. On the other hand, usual structural metrics are overly sensitive to pixel-level changes, so they penalize style transfer that changes appearance even without altering object boundaries. Finding metrics that are robust to appearance changes and better correlated with downstream performance remains an open problem.

Several approaches could reduce this evaluation gap. Rather than training from scratch, one could fine-tune a large pretrained model on a small number of generated images and evaluate on real data. This is cheaper, but the pretrained model may carry

biases that produce performance discrepancies between finetuning and training from scratch: a backbone already familiar with the target domain may inflate apparent quality, masking weaknesses in the generated data that would be visible when training from scratch. Alternatively, a frozen zero-shot model could evaluate generated images directly, without any training. One could generate a few images as evaluation samples (rather than training data) and score them with the same model across all methods, ensuring a fair comparison. However, one must check that the zero-shot model is not biased toward a particular domain, and that its scores actually correlate with downstream performance. Finally, dataset-level metrics may be more predictive of training outcomes than pixel-level statistics. Measuring the semantic coverage of a generated dataset, meaning the diversity of object configurations, scene layouts, or condition representations, could better capture what matters for downstream training.

6.4.2 Benchmark bias and the GTA→Cityscapes monoculture

Most current foundation models are pretrained on web-scale datasets that heavily overlap with the visual distribution of urban driving scenes. For instance, DINOv2 [Oquab et al. (2023)] reports using Mapillary Street-Level Sequences containing 1.5 million urban driving images in its training data. These images are unlabeled but cover a wide range of situations and environmental conditions. DINOv2 is then used as a backbone in numerous domain adaptation methods. This raises questions about the absolute performance numbers reported on GTA5→Cityscapes benchmarks. If the backbone has already seen millions of images similar to Cityscapes during pretraining, the measured domain adaptation gain may partly reflect familiarity with the target domain rather than the effectiveness of the adaptation method itself. This does not invalidate relative comparisons between methods using the same backbone, but it inflates absolute metrics and diminishes the benefits of synthetic data, since the models have already seen large amounts of similar real images. The same concern applies to other widely used foundation models: CLIP [Radford et al. (2021)] and Stable Diffusion [Rombach et al. (2022)] are similarly pretrained on large web crawls with substantial overlap with urban driving imagery, and any method that relies on them as a backbone or generative prior inherits this bias. The runway detection experiments in Chapter 5 illustrate this problem. Runway imagery is largely absent from web-scale training sets, and the domain gap observed in practice is substantially larger than what urban driving benchmarks suggest [Yun et al. (2025)]. Evaluating on domains absent from pretraining data would give a more accurate picture of actual generalization ability.

Beyond the choice of backbone and benchmark, the standard problem formulation in the literature also creates a mismatch with industrial settings. Most methods in the literature are formulated as unsupervised domain adaptation (UDA), assuming that target images are abundant but unlabeled. In industrial sim-to-real settings, the situation is often reversed: labels are available for synthetic data, and some labeled real data may exist, but in smaller quantities. For style transfer methods in particular, the transferred images inherit the labels from the synthetic ones, creating a fully labeled pseudo-target domain. Despite this, the UDA formulation prevents exploiting these labels during adaptation. Classical supervised training methods, on the other hand, do not necessarily account for the domain gap. More work on fully

supervised domain adaptation, using all available annotations while accounting for the distribution shift, would be valuable. CARE [Prabhu et al. (2023)] is one such method for detection; similar approaches for segmentation remain underexplored. Combining supervised feature-level adaptation with pixel-level style transfer, using all available labels, is a direction worth exploring.

6.4.3 Towards an open industrial domain adaptation benchmark

To address the benchmark bias problem, an open domain adaptation benchmark is needed that satisfies the following criteria: (i) the source domain should be clearly distinct from the web-scale urban driving distribution on which current foundation models are pretrained, to avoid implicit contamination; (ii) it should target a real industrial application such as aviation, maritime operations, agriculture, or medical imaging; (iii) it should provide both synthetic and real data with ground-truth labels for rigorous evaluation; and (iv) it should be sufficiently challenging to expose the limitations of methods that rely implicitly on Cityscapes pretraining.

The LARD dataset [Ducoffe et al. (2023)] and the runway data used in Chapter 5 are a step in this direction, although the real image set remains private and small. A public benchmark of this kind would be valuable. It would allow evaluating domain adaptation methods under realistic conditions and measuring the actual contribution of synthetic data in applications where real data is rare and expensive. It would also encourage the development of methods that account for the specificities of industrial applications, rather than optimizing exclusively for urban driving benchmarks.

Such benchmark datasets should also be kept out of future foundation model training sets, to preserve their value for evaluation.

6.5 General conclusion

This thesis studied synthetic-to-real domain adaptation for object recognition in industrial settings where real data is scarce. We proposed and evaluated diffusion-based style transfer methods (CACTI_F), an evolutionary optimization framework for style transfer pipelines (MOOSS), and applied these approaches to the industrial task of runway detection alongside feature-level adaptation.

Synthetic images are often presented as ideal solutions, but the domain gap remains large. In all our experiments, synthetic data was necessary but not sufficient. Real data is needed at every stage: as style references for pixel-level adaptation, as target features for feature-level alignment, and as evaluation sets. The most effective approach consistently combined synthetic and real data rather than relying on either alone. Style transfer and data augmentation are the most valuable when generating diverse data in conditions that are difficult to capture in reality, such as adverse weather, rare events, or unusual viewpoints. As a complement to a model trained on real data, synthetic data can extend coverage to underrepresented conditions and improve robustness. Although CACTI_F was developed for scenarios with very few real images, this setting is in practice unrealistic because models must still be validated on real operational conditions. Rather, the contribution of this work is to make the most of scarce real data: using it as a style reference for pixel-level

adaptation, as a calibration signal for feature-level alignment, and as a held-out evaluation set, while synthetic data provides complementary coverage for conditions that are rare or difficult to capture in the field.

A recurring challenge throughout this work is the absence of reliable, cheap metrics for evaluating generated datasets. Image quality metrics do not consistently predict downstream task performance, and the only trustworthy evaluation requires full model training, an expensive process that drastically slows down the research cycle. This bottleneck affects method development, hyperparameter tuning, and benchmark comparisons. Better surrogate metrics for downstream performance would make data generation methods more practical. In this thesis, MOOSS was motivated precisely by the need to reduce this cost by using cheap image-level proxies rather than full model training, but as shown in Chapter 4, no proxy metric proved reliable enough to replace the full evaluation cycle.

The GTA5→Cityscapes benchmark, while widely used, increasingly suffers from contamination by foundation models pretrained on similar urban driving data. This inflates absolute performance numbers and may not reflect the true difficulty of domain adaptation for industrial applications outside the urban driving domain. The runway detection experiments in this thesis illustrate the gap: methods that perform well on urban benchmarks do not necessarily transfer to aviation settings where pretraining data is absent. Beyond contamination, the existing benchmarks themselves have significant limitations. GTA5 provides approximately 25,000 images but offers no metadata on weather conditions or time of day, and all images depict a US-style environment, creating content discrepancies with the European scenes of Cityscapes and ACDC (e.g. road widths, car models, urban layouts). Cityscapes contains only 5,000 fully annotated images and ACDC approximately 4,000. Although other datasets exist (Synthia for synthetic data, BDD100K for real data), they target the same urban driving application. These scales are small by industrial standards, and there is little diversity across domain adaptation benchmarks overall. Other application domains could be explored, such as satellite imagery with pre- and post-disaster data or multi-sensor acquisitions, but finding datasets that share similar object categories and annotation formats while exhibiting a significant style gap remains difficult. More diverse benchmarks are needed: ones that cover domains underrepresented in web-scale training data, include both synthetic and real labeled data, and target real industrial applications. They should also remain isolated from foundation model training to avoid future contamination.

Diffusion-based style transfer can partially bridge the synthetic-to-real gap, but it requires careful semantic conditioning to produce training-compatible data. Evolutionary optimization can automate pipeline design, though the absence of a cheap downstream performance proxy limits its full potential. The standard benchmarks are increasingly insufficient for evaluating progress in non-urban industrial settings. Overall, further progress will depend on better metrics and more diverse evaluation protocols, alongside methods that explicitly account for the characteristics of both real and synthetic data in industrial applications, especially under rare and adverse conditions.

Bibliography

Alaluf, Y., Garibi, D., Patashnik, O., Averbuch-Elor, H., and Cohen-Or, D. (2024). Cross-Image Attention for Zero-Shot Appearance Transfer. In *ACM SIGGRAPH 2024 Conference Papers*.

Bekor, T., Nayman, N., and Zelnik-Manor, L. (2024). FreeAugment: Data augmentation search across all degrees of freedom. In *European Conference on Computer Vision (ECCV)*.

Benigmim, Y., Roy, S., Essid, S., Kalogeiton, V., and Lathuilière, S. (2023). One-shot Unsupervised Domain Adaptation with Personalized Diffusion Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Bougacha, Y., Delhomme, G., Ducoffe, M., Fuchs, A., Ginestet, J.-B., Girard, J., Kraiem, S., Mamalet, F., Mussot, V., Pagetti, C., and Sammour, T. (2026). Lard 2.0: Enhanced datasets and benchmarking for autonomous landing systems. In *13th European Congress of Embedded Real Time Systems (ERTS)*.

Caesar, H., Uijlings, J., and Ferrari, V. (2018). COCO-Stuff: Thing and stuff classes in context. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. In *European Conference on Computer Vision (ECCV)*.

Carranza-García, M., Torres-Mateo, J., Lara-Benítez, P., and García-Gutiérrez, J. (2021). On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data. *Remote Sensing*.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. (2015). ShapeNet: An Information-Rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*.

Chen, H., Xiang, Q., Hu, J., Ye, M., Yu, C., Cheng, H., and Zhang, L. (2025). Comprehensive exploration of diffusion models in image generation: a survey. *Artificial Intelligence Review*.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2018). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Chen, M. and Hu, Y. (2024). An image-based runway detection method for fixed-wing aircraft based on deep neural network. *IET Image Processing*.

Chen, W., Zhang, Z., Yu, L., and Tai, Y. (2023). BARS: a benchmark for airport runway segmentation. *Applied Intelligence*.

Chen, X., Xu, C., Yang, X., Song, L., and Tao, D. (2019). Gated-GAN: Adversarial Gated Networks for Multi-Collection Style Transfer. *IEEE Transactions on Image Processing*.

Chigot, E., Barrat, S., Wilson, D. G., Huguenin, M., and Oberlin, T. (2026a). Enhancing runway detection models with synthetic-to-real style transfer. *Digital Avionics Systems Conference*.

Chigot, E., Oberlin, T., Huguenin, M., and Wilson, D. G. (2026b). Multi-objective optimization for synthetic-to-real style transfer. In *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*.

Chigot, E., Wilson, D. G., Ghrib, M., Jimenez, F., and Oberlin, T. (2025a). Synthetic data for robust runway detection. In *Computer Analysis of Images and Patterns*.

Chigot, E., Wilson, D. G., Ghrib, M., and Oberlin, T. (2025b). Style transfer with diffusion models for synthetic-to-real domain adaptation. *Computer Vision and Image Understanding*.

Chung, J., Hyun, S., and Heo, J.-P. (2024). Style Injection in Diffusion: A Training-Free Approach for Adapting Large-Scale Diffusion Models for Style Transfer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Clark, A. and Contributors (2025). Pillow: Python imaging library.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Csurka, G. (2017). Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Davendra, D. (2010). *Traveling salesman problem: Theory and applications*. BoD–Books on Demand.

De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. (2021). Diffusion Schrödinger bridge with applications to score-based generative modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*.

Deng, Y., He, X., Tang, F., and Dong, W. (2024). Z*: Zero-shot style transfer via attention reweighting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Dhariwal, P. and Nichol, A. (2021). Diffusion Models Beat GANs on Image Synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Ding, K., Ma, K., Wang, S., and Simoncelli, E. P. (2022). Image Quality Assessment: Unifying Structure and Texture Similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*. PMLR.

Ducoffe, M., Carrere, M., Féliers, L., Gauffriau, A., Mussot, V., Pagetti, C., and Sammour, T. (2023). LARD–Landing Approach Runway Detection–Dataset for Vision Based Landing. *arXiv preprint arXiv:2304.09938*.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*.

Fahes, M., Vu, T.-H., Bursuc, A., Pérez, P., and De Charette, R. (2023). Poda: Prompt-driven zero-shot domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Flood, M. M. (1956). The traveling-salesman problem. *Operations research*.

Frenkel, Y., Vinker, Y., Shamir, A., and Cohen-Or, D. (2024). Implicit style-content separation using b-lora. In *European Conference on Computer Vision (ECCV)*.

Fu, S., Tamir, N., Sundaram, S., Chai, L., Zhang, R., Dekel, T., and Isola, P. (2023). DreamSim: Learning New Dimensions of Human Visual Similarity using Synthetic Data. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Gatys, L. A., Ecker, A. S., and Bethge, M. (2016). Image style transfer using convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ghosh, S., Patrikar, J., Moon, B., Hamidi, M. M., and Scherer, S. (2023). AirTrack: Onboard Deep Learning Framework for Long-Range Aircraft Detection and Tracking. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2016). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Go, S., Choi, K., Shin, M., and Uh, Y. (2026). Eye-for-an-eye: Appearance transfer with dense semantic correspondence in diffusion models. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Golden, B. L., Levy, L., and Vohra, R. (1987). The orienteering problem. *Naval Research Logistics (NRL)*.

Gong, R., Danelljan, M., Sun, H., Mangas, J. D., Marin, N., and Gool, L. V. (2024). Prompting Diffusion Representations for Cross-Domain Semantic Segmentation. In *35th British Machine Vision Conference, BMVC*.

Gooch, B., Coombe, G., and Shirley, P. (2002). Artistic vision: painterly rendering using computer vision techniques. In *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*.

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

Groba, C., Sartal, A., and Vázquez, X. H. (2015). Solving the dynamic traveling salesman problem using a genetic algorithm with trajectory prediction: An application to fish aggregating devices. *Computers & Operations Research*.

Guan, H. and Liu, M. (2022). Domain Adaptation for Medical Image Analysis: A Survey. *IEEE Transactions on Biomedical Engineering*.

Han, Z., Gao, C., Liu, J., Zhang, J., and Zhang, S. Q. (2024). Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey. *arXiv preprint arXiv:2403.14608*.

Hataya, R., Zdenek, J., Yoshizoe, K., and Nakayama, H. (2020). Faster AutoAugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision (ECCV)*.

Hertz, A., Voynov, A., Fruchter, S., and Cohen-Or, D. (2024). Style aligned image generation via shared attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hertzmann, A. (1998). Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *International Conference on Neural Information Processing Systems (NeurIPS)*.

Ho, D., Liang, E., Chen, X., Stoica, I., and Abbeel, P. (2019). Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning (ICML)*.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*.

Hoyer, L., Dai, D., and Van Gool, L. (2022a). DAFormer: Improving Network Architectures and Training Strategies for Domain-Adaptive Semantic Segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hoyer, L., Dai, D., and Van Gool, L. (2022b). HRDA: Context-Aware High-Resolution Domain-Adaptive Semantic Segmentation. In *European Conference on Computer Vision (ECCV)*.

Hoyer, L., Dai, D., Wang, H., and Van Gool, L. (2023). MIC: Masked Image Consistency for Context-Enhanced Domain Adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2022). LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations (ICLR)*.

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

Huang, Y., Huang, J., Liu, Y., Yan, M., Lv, J., Liu, J., Xiong, W., Zhang, H., Cao, L., and Chen, S. (2025). Diffusion Model-Based Image Editing: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Huberman-Spiegelglas, I., Kulikov, V., and Michaeli, T. (2024). An edit friendly DDPM noise space: Inversion and manipulations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Iyer, K. H., Marnor, C. M., Schmid, D. W., and Hartz, E. H. (2025). Detecting and quantifying deep sea benthic life using advanced object detection. *Frontiers in Marine Science*.

Jayasumana, S., Ramalingam, S., Veit, A., Glasner, D., Chakrabarti, A., and Kumar, S. (2024). Rethinking FID: Towards a Better Evaluation Metric for Image Generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jeong, J., Kim, J., Choi, Y., Lee, G., and Uh, Y. (2024). Visual style prompting with swapping self-attention. *arXiv preprint arXiv:2402.12974*.

Jia, Y., Hoyer, L., Huang, S., Wang, T., Van Gool, L., Schindler, K., and Obukhov, A. (2024). DGInStyle: Domain-Generalizable Semantic Segmentation with Image Diffusion Models and Stylized Semantic Control. In *European Conference on Computer Vision (ECCV)*.

Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., and Song, M. (2019). Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics*.

Karras, T., Laine, S., and Aila, T. (2019). A Style-Based Generator Architecture for Generative Adversarial Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kazmierczak, R., Berthier, E., Frehse, G., and Franchi, G. (2025). Explainability and vision foundation models: A survey. *Information Fusion*.

Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., and Girshick, R. (2023). Segment Anything. In *IEEE/CVF International Conference on Computer Vision (ICCV)*.

Koschier, D., Bender, J., Solenthaler, B., and Teschner, M. (2022). A Survey on SPH Methods in Computer Graphics. *Computer Graphics Forum*.

Larranaga, P., Kuijpers, C. M. H., Murga, R. H., Inza, I., and Dizdarevic, S. (1999). Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artificial intelligence review*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*.

Li, Y., Angelov, P., Yu, Z., Lopez Pellicer, A., and Suri, N. (2024a). Federated Adversarial Learning for Robust Autonomous Landing Runway Detection. In *International Conference on Artificial Neural Networks*.

Li, Y., Hu, G., Wang, Y., Hospedales, T., Robertson, N. M., and Yang, Y. (2020). Differentiable automatic data augmentation. In *European Conference on Computer Vision (ECCV)*.

Li, Y., Yuan, W., Zhang, S., Yan, W., Shen, Q., Wang, C., and Yang, M. (2024b). Choose Your Simulator Wisely: A Review on Open-Source Simulators for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles*.

Lim, S., Kim, I., Kim, T., Kim, C., and Kim, S. (2019). Fast AutoAugment. In *International Conference on Neural Information Processing Systems (NeurIPS)*.

Lin, C.-T., Huang, S.-W., Wu, Y.-Y., and Lai, S.-H. (2021). GAN-Based Day-to-Night Image Style Transfer for Nighttime Vehicle Detection. *IEEE Transactions on Intelligent Transportation Systems*.

Lin, H., Hosu, V., and Saupe, D. (2019). Kadid-10k: A large-scale artificially distorted iqa database. In *2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX)*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision (ECCV)*.

Lindén, J., Forsberg, H., Haddad, J., Tagebrand, E., Cedernaes, E., Ek, E. G., and Daneshtalab, M. (2021). Curating Datasets for Visual Runway Detection. In *IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*.

Lindroth, H., Nalaie, K., Raghu, R., Ayala, I. N., Busch, C., Bhattacharyya, A., Moreno Franco, P., Diedrich, D. A., Pickering, B. W., and Herasevich, V. (2024). Applied artificial intelligence in healthcare: A review of computer vision technology application in hospital settings. *Journal of Imaging*.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. (2023). Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*.

Liu, S., Han, Y., Xing, P., Yin, F., Wang, R., Cheng, W., Liao, J., Wang, Y., Fu, H., Han, C., Li, G., Peng, Y., Sun, Q., Wu, J., Cai, Y., Ge, Z., Ming, R., Xia, L., Zeng, X., Zhu, Y., Jiao, B., Zhang, X., Yu, G., and Jiang, D. (2025). Step1X-Edit: A Practical Framework for General Image Editing. *arXiv preprint arXiv:2504.17761*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In *European Conference on Computer Vision (ECCV)*.

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

Luley, P.-P., Deriu, J. M., Yan, P., Schatte, G. A., and Stadelmann, T. (2023). From Concept to Implementation: The Data-Centric Development Process for AI in Industry. In *2023 10th IEEE Swiss Conference on Data Science (SDS)*.

Marc, S. T., Belavkin, R., Windridge, D., and Gao, X. (2023). An evolutionary approach to automated class-specific data augmentation for image classification. In *International conference on the dynamics of information systems*.

Maslej, N., Fattorini, L., Perrault, R., Gil, Y., Parli, V., Kariuki, N., Capstick, E., Reuel, A., Brynjolfsson, E., Etchemendy, J., et al. (2025). Artificial intelligence index report 2025. *arXiv preprint arXiv:2504.07139*.

Mirza, M. and Osindero, S. (2014). Conditional Generative Adversarial Nets. *arXiv preprint arXiv:1411.1784*.

Muhammad, K., Hussain, T., Ullah, H., Ser, J. D., Rezaei, M., Kumar, N., Hijji, M., Bellavista, P., and de Albuquerque, V. H. C. (2022). Vision-Based Semantic Segmentation in Scene Understanding for Autonomous Driving: Recent Achievements, Challenges, and Outlooks. *IEEE Transactions on Intelligent Transportation Systems*.

Mumuni, A., Mumuni, F., and Gerrar, N. K. (2024). A Survey of Synthetic Data Augmentation Methods in Machine Vision. *Machine Intelligence Research*.

Neuhold, G., Ollmann, T., Bulò, S. R., and Kontschieder, P. (2017). The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In *IEEE International Conference on Computer Vision (ICCV)*.

Nikolenko, S. I. (2021). *Synthetic data for deep learning*. Springer.

Ohkawa, T., Furuta, R., and Sato, Y. (2023). Efficient Annotation and Learning for 3D Hand Pose Estimation: A Survey. *International Journal of Computer Vision*.

Oquab, M., Darcet, T., Moutakanni, T., Vo, H. V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Howes, R., Huang, P.-Y., Xu, H., Sharma, V., Li, S.-W., Galuba, W., Rabbat, M., Assran, M., Ballas, N., Synnaeve, G., Misra, I., Jegou, H., Mairal, J., Labatut, P., Joulin, A., and Bojanowski, P. (2023). Dinov2: Learning robust visual features without supervision.

Peng, D., Hu, P., Ke, Q., and Liu, J. (2023). Diffusion-based image translation with label guidance for domain adaptive semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*.

Peng, J., Huang, Y., Sun, W., Chen, N., Ning, Y., and Du, Q. (2022). Domain Adaptation in Remote Sensing Image Classification: A Survey. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*.

Pereira, S., Correia, J., and Machado, P. (2022). Evolving data augmentation strategies. In *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*.

Potvin, J.-Y. (1996). Genetic algorithms for the traveling salesman problem. *Annals of operations Research*.

Prabhu, V. U., Acuna, D., Mahmood, R., Law, M. T., Liao, Y.-H., Hoffman, J., Fidler, S., and Lucas, J. (2023). Bridging the Sim2Real gap with CARE: Supervised Detection Adaptation with Conditional Alignment and Reweighting. *Transactions on Machine Learning Research*.

Qi, T., Fang, S., Wu, Y., Xie, H., Liu, J., Chen, L., He, Q., and Zhang, Y. (2024). Deadiff: An efficient stylization diffusion model with disentangled representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., and others (2021). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*. PMLR.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ren, S., He, K., Girshick, R., and Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Richter, S. R., Vineet, V., Roth, S., and Koltun, V. (2016). Playing for Data: Ground Truth from Computer Games. In *European Conference on Computer Vision (ECCV)*.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. (2016). The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. (2023). Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*.

Sakaridis, C., Dai, D., and Van Gool, L. (2021). ACDC: The Adverse Conditions Dataset with Correspondences for Semantic Driving Scene Understanding. In *IEEE/CVF International Conference on Computer Vision (ICCV)*.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C. W., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., Schramowski, P., Kundurthy, S. R., Crowson, K., Schmidt, L., Kaczmarczyk, R., and Jitsev, J. (2022). LAION-5B: An open large-scale dataset for training next generation image-text models. In *Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Schwonberg, M. and Gottschalk, H. (2025). Domain Generalization for Semantic Segmentation: A Survey. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schwonberg, M., Niemeijer, J., Termöhlen, J.-A., schäfer, J. P., Schmidt, N. M., Gottschalk, H., and Fingscheidt, T. (2023). Survey on Unsupervised Domain Adaptation for Semantic Segmentation for Visual Perception in Automated Driving. *IEEE Access*.

Shah, S., Dey, D., Lovett, C., and Kapoor, A. (2017). AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and service robotics: Results of the 11th international conference*.

Shah, V., Ruiz, N., Cole, F., Lu, E., Lazebnik, S., Li, Y., and Jampani, V. (2024). Ziplora: Any subject in any style by effectively merging loras. In *European Conference on Computer Vision (ECCV)*.

Shermeyer, J., Hossler, T., Etten, A. V., Hogan, D., Lewis, R., and Kim, D. (2021). RarePlanes: Synthetic Data Takes Flight. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

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

Song, J., Meng, C., and Ermon, S. (2021a). Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2021b). Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*.

Tasgetiren, M. F. and Smith, A. E. (2000). A genetic algorithm for the orienteering problem. In *Proceedings of the 2000 congress on evolutionary computation*.

Tedrake, R. (2024). *Robotic Manipulation*.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Tranheden, W., Olsson, V., Pinto, J., and Svensson, L. (2021). DACS: Domain Adaptation via Cross-domain Mixed Sampling. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., and Birchfield, S. (2018). Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Tsai, H.-K., Yang, J.-M., Tsai, Y.-F., and Kao, C.-Y. (2004). An evolutionary algorithm for large traveling salesman problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*.

van den Oord, A., Vinyals, O., and kavukcuoglu, k. (2017). Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Wang, H., Wang, Q., Bai, X., Qin, Z., and Chen, A. (2024a). InstantStyle: Free Lunch towards Style-Preserving in Text-to-Image Generation. *arXiv preprint arXiv:2404.02733*.

Wang, H., Xing, P., Huang, R., Ai, H., Wang, Q., and Bai, X. (2024b). InstantStyle-Plus: Style Transfer with Content-Preserving in Text-to-Image Generation. *arXiv preprint arXiv:2407.00788*.

Wang, Q., Feng, W., Zhao, H., Liu, B., and Lyu, S. (2024c). VALNet: Vision-Based Autonomous Landing with Airport Runway Instance Segmentation. *Remote Sensing*.

Wang, Q., Fink, O., Van Gool, L., and Dai, D. (2022a). Continual test-time domain adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, X., Guo, Z., Li, Z., Yu, J., Zhang, H., Iwasawa, Y., Matsuo, Y., and Guo, J. (2026). One-shot portrait stylization via geometric alignment. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Wang, X., Xu, Y., Liu, S., Ren, B., Kosinka, J., Telea, A. C., Wang, J., Song, C., Chang, J., Li, C., Zhang, J. J., and Ban, X. (2024d). Physics-based fluid simulation in computer graphics: Survey, research trends, and challenges. *Computational Visual Media*.

Wang, Y. O., Chung, Y., Wu, C. H., and De La Torre, F. (2024e). Domain Gap Embeddings for Generative Dataset Augmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*.

Wang, Z., Zhao, D., and Cao, Y. (2022b). Visual Navigation Algorithm for Night Landing of Fixed-Wing Unmanned Aerial Vehicle. *Aerospace*.

Waymo (2025). Demonstrably Safe AI For Autonomous Driving.

Whitley, D., Starkweather, T., and Shaner, D. (1991). *The traveling salesman and sequence scheduling: Quality solutions using genetic edge recombination*. Colorado State University, Department of Computer Science Fort Collins.

Wu, T., Zhang, J., Fu, X., Wang, Y., Jiawei Ren, L. P., Wu, W., Yang, L., Wang, J., Qian, C., Lin, D., and Liu, Z. (2023). OmniObject3D: Large-vocabulary 3D object dataset for realistic perception, reconstruction and generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, Y., Chi, Z., Wang, Y., Plataniotis, K. N., and Feng, S. (2024). Test-time domain adaptation by learning domain-aware batch normalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3D ShapeNets: A deep representation for volumetric shapes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. (2021). SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. *Advances in Neural Information Processing Systems (NeurIPS)*.

Xu, R., Xi, W., Wang, X., Mao, Y., and Cheng, Z. (2025). StyleSSP: Sampling StartPoint enhancement for training-free diffusion-based method for style transfer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., and Darrell, T. (2020). BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yu, F., Wang, D., Chen, Y., Karianakis, N., Shen, T., Yu, P., Lymberopoulos, D., Lu, S., Shi, W., and Chen, X. (2022). SC-UDA: Style and Content Gaps aware Unsupervised Domain Adaptation for Object Detection. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Yun, S., Chae, S., Lee, D., and Ro, Y. (2025). SoMA: Singular Value Decomposed Minor Components Adaptation for Domain Generalizable Representation Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, L., Rao, A., and Agrawala, M. (2023a). Adding conditional control to text-to-image diffusion models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, Y., Huang, N., Tang, F., Huang, H., Ma, C., Dong, W., and Xu, C. (2023b). Inversion-based style transfer with diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zheng, Z. and Yang, Y. (2021). Rectifying Pseudo Label Learning via Uncertainty Estimation for Domain Adaptive Semantic Segmentation. *International Journal of Computer Vision*.

Zhong, T., Chi, Z., Gu, L., Wang, Y., Yu, Y., and Tang, J. (2022). Meta-DMoE: Adapting to domain shift by meta-distillation from mixture-of-experts. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. (2017). Scene parsing through ADE20K dataset. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhou, J., Wei, C., Wang, H., Shen, W., Xie, C., Yuille, A., and Kong, T. (2022). Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations (ICLR)*.

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

Zylann (2018). Heightmap terrain demo – Godot Asset Library. MIT License. [Accessed 02-09-2026].