POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Enhancing Test Generation for Autonomous Driving Using Multiple Conditionings Probabilistic Diffusion Models

Tesi di Laurea Magistrale in
Mathematical Engineering - Ingegneria Matematica

Jaime Enríquez, 10828410

Advisor:
Prof. Luciano Baresi

Co-advisor:
Davide Yi Xian Hu

Academic year:
2023-2024

Abstract: In recent years, Deep Learning (DL) models have played a critical role in the development of autonomous driving systems, performing essential tasks such as object detection, semantic segmentation, and decision-making. However, testing these models presents significant challenges, particularly in real-world scenarios where reproducing specific driving conditions is costly, unsafe, or impossible. Traditional testing methods lack the control needed to systematically evaluate DL models across a broad range of conditions, especially in rare or edge cases that are difficult to recreate, such as driving in extreme weather. This thesis addresses these challenges by proposing a novel approach that leverages conditional diffusion models for controlled image generation. The proposed method extends the T2I-Adapter model to support multi-conditioning, allowing it to generate test scenarios based on various inputs, such as image edges, colors, or semantic information. This enables a fine-grained control over the generation of realistic driving scenarios, significantly improving the ability to test DL models for autonomous driving under diverse conditions. The model is fine-tuned on the SHIFT dataset, a synthetic dataset collected in the CARLA simulation environment, which includes a wide variety of weather patterns, traffic conditions, and driving environments. The evaluation of the proposed solution demonstrates its effectiveness in generating valid, high-quality test cases. It also shows improved control over the generation process compared to existing state-of-the-art generative models, such as Stable Diffusion, ControlNet, and single-conditioning models. Additionally, this thesis contributes a newly augmented dataset built on top of SHIFT, along with various checkpoints of the trained model. These contributions provide a comprehensive framework for advancing the testing of ML-based autonomous driving systems, enabling more rigorous and diverse evaluation processes.

Key-words: Metamorphic Testing; Alternative Examples; Data Augmentation; Diffusion Models; Multi-conditioning; Autonomous Driving.

1. Introduction

Deep Learning (DL) [7] has revolutionized numerous industries, driving advancements in fields such as healthcare [20], finance[39], and transportation [1]. By enabling machines to automatically extract patterns from large datasets, DL models, which are based on neural networks, have become the main technique used in various AI

applications. In particular, DL-based systems are critical in autonomous driving [37], where they power key tasks such as object detection [33], semantic segmentation [6], and decision-making [18]. These tasks require high accuracy and robustness, as even minor errors can result in catastrophic outcomes that might threaten the safety of humans, such as the driver, the passengers, or the pedestrians.

As the deployment of DL models in safety-critical environments like autonomous driving grows, the need for rigorous testing becomes increasingly important. Testing DL models ensures that they perform reliably under a wide range of conditions, including edge cases that might not have been seen during training. However, traditional testing methods are inadequate in this context because DL models learn complex data-driven behaviors that are often difficult for humans to interpret. As a result, these models are typically treated as black boxes, making it challenging to anticipate their potential failures.

In the context of autonomous driving, testing DL models is even more challenging because reproducing test cases that result in failures can be extremely costly and unsafe. For instance, consider a scenario where a self-driving car crashes into a guardrail or, worse, hits a pedestrian. Recreating such situations in the real world to test the model behavior would not only involve significant financial costs but also pose serious risks to human lives and property. Due to these safety concerns, most testing is conducted offline, either by using pre-recorded data or in controlled simulation environments.

In addition to safety concerns, testing in autonomous driving is further complicated by the unpredictable nature of real-world scenarios. DL models must be capable of operating reliably across a variety of environments, handling diverse weather conditions, and responding to dynamic interactions with other vehicles and pedestrians. Some environments, however, are particularly difficult to reproduce. For instance, to test whether a self-driving car can drive safely during a hailstorm, one would need to wait for such weather conditions, which are beyond the control of the tester. To address these challenges, researchers have developed several testing frameworks, including simulation environments [5, 31], synthetic data generation [11], and adversarial attacks [10]. While these methods provide useful tools for evaluating model performance, they often lack precise control over the test scenarios being generated.

Recent work exploits metamorphic testing to explore how small and systematic changes to input data, referred to as metamorphic relationships, logically impact the output behavior of the model. Through these relationships, these testing approaches create new, varied test cases without needing to rely on rare or hard-to-reproduce conditions. However, one critical limitation of current methods is the inability to precisely control the test cases generated for DL models. Existing techniques often rely on random sampling or adversarial examples, which do not always align with realistic driving conditions. Furthermore, traditional synthetic data generation methods lack the flexibility to explore a range of metamorphic relationships [27], where minor changes to a scenario should preserve or logically alter the model behavior. This makes it difficult to fully stress-test DL models and identify corner cases that might lead to failures.

To address these challenges, this thesis proposes an approach that leverages conditional diffusion models for controlled image generation. Diffusion models [12] are a class of generative models that learn to generate data by gradually transforming noise into coherent outputs. By conditioning this transformation process on specific inputs, it becomes possible to exert fine-grained control over the generated scenarios. This allows the creation of diverse, realistic test cases that can simulate critical metamorphic relationships in autonomous driving.

The solution proposed in this thesis extends an existing conditional diffusion model, known as T2I-Adapter [21], to enable conditioning through multiple inputs simultaneously. These inputs, such as color, image edges, or semantic information, work together to guide and refine the generation process, offering greater control over the creation of new images.

The solution consists in a fine-tuning process that retrains a pre-trained T2I-Adapter architecture on an autonomous driving dataset called SHIFT [30], a comprehensive dataset collected within a simulation environment designed to capture the complexities of real-world driving scenarios. SHIFT includes a broad variety of driving conditions, with data covering various weather patterns such as rain, fog, snow, and bright sunlight, along with different traffic densities and road types. This diverse dataset ensures that the fine-tuned model learns to generate images representing a wide range of challenging driving situations.

This thesis includes an extensive evaluation of the proposed approach and compares its effectiveness with respect to existing conditional diffusion models in generating valid and realistic test cases for autonomous driving. Specifically, this thesis compares the proposed solution with state-of-the-art text-conditioned diffusion models, such as Stable Diffusion [25] and Stable Diffusion XL [22], and single-conditioning diffusion models, like ControlNet and T2I-Adapter.

Results show that the proposed solution improves both the validity of the generated test cases while also increasing their quality. We evaluated our approach against state-of-the-art solutions, namely ControlNet, T2I-Adapter, Stable Diffusion, and Stable Diffusion Image-to-image, and we the proposed solution was the only one capable of achieving both good validity and realism.

To summarize the main contributions of this thesis are the following.

- **Multi-conditioning Generative Solution**: this thesis presents a novel extension to the T2I-Adapter model that enables multi-conditioning input for image generation. By allowing the model to be condi-

tioned on various inputs simultaneously, autonomous driving testers can exert fine-grained control over the generation of images.

- **Testing Dataset**: This thesis also releases a newly built dataset on top of the SHIFT dataset by augmenting some of its existing images. These augmentations include modifications that reflect different driving environments, weather conditions, and traffic variations, making the dataset an even more useful tool for testing autonomous driving systems.
- **In-depth Evaluation**: The proposed solution is thoroughly evaluated to assess its effectiveness in generating diverse and realistic images for testing autonomous driving systems. This evaluation includes testing the model's ability to simulate different driving conditions and its capacity for generating high-quality images.
- **In-depth Comparison**: In addition to evaluating the performance of the proposed approach, the thesis also conducts an in-depth comparison with existing generative methods used for testing autonomous driving systems. The comparison includes benchmarks against models that use single-condition inputs or less flexible generative processes.

The thesis is structured in the following way:

1. **Introduction**: This section provides a general overview of the thesis. It introduces DL models as part of testing autonomous driving algorithms, discusses its challenges, and goes over the motivation behind using multiple conditionings for diffusion models for this task.
2. **Background**: This chapter goes over the main topics needed to understand the thesis. Specifically, it explains how metamorphic testing works, goes over the autonomous driving use case, discusses how generative models based on Diffusion Models work, both for training and inference, and formally defines the problem being addressed in the project.
3. **Related Work**: This section reviews state-of-the-art generative models, discussing their approach to the problem and highlighting both the advantages and disadvantages of controlling the desired generation. It includes an analysis of ControlNet, Instruct Pix2Pix, and, T2I-Adapter, and a brief review of using multiple adapters on a Stable Diffusion model
4. **Methodology**: This chapter details the proposed multi-conditioning generative solution, explaining how the T2I-Adapter model has been modified to allow multiple conditionings. It provides a brief overview of a small study on various conditioning combinations, followed by a discussion of the training process using the SHIFT dataset and the models that were trained.
5. **Evaluation**: This section explains the evaluations carried out of the proposed solution, including their capability of generating complex driving scenarios. It covers quantitative and human assessments, comparing the approach against baseline models detailed in Related Work.
6. **Conclusion and Future Work**: The final section summarizes the contents of the project and the results achieved. It also discusses possible directions for future research.

2. Background

In the current section, we discuss the main background technologies and methods of the thesis needed to better understand this thesis work. First, we will provide an overview of metamorphic testing approaches. Next, we will introduce one of the most popular generative artificial intelligence techniques for image generation. Finally, we will introduce a formal description of the problem at hand.

2.1. Metamorphic testing

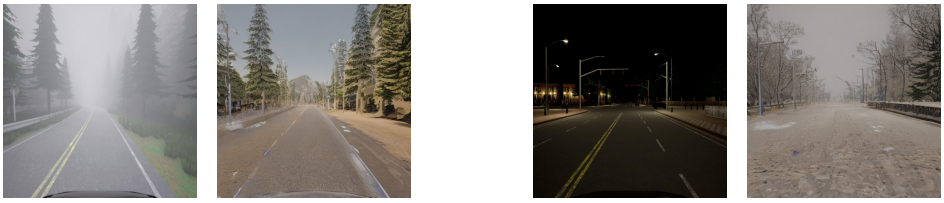
Testing [13] is a fundamental step of the software development lifecycle, playing a crucial role in ensuring that software behaves as intended across various conditions. It is indispensable for detecting defects, enhancing overall quality, and providing assurance of the software's performance and reliability. Effective testing not only uncovers issues early in the development process but also ensures that the final product aligns with user needs and expectations.

Metamorphic Testing [27] is a powerful testing approach designed to identify flaws in a system by examining the consistency of outputs generated from related inputs. Unlike traditional testing methods, which rely on predefined input-output pairs, metamorphic testing focuses on the relationship (also known as Metamorphic Relationship) between inputs and their corresponding outputs. This technique is particularly useful in situations where generating a comprehensive set of test cases is challenging or where the expected output is difficult to determine in advance.

Metamorphic testing is employed to assess the robustness of a model by exposing it to additional data that may be encountered during the testing process. This technique has been traditionally used in software engineering to test algorithms. For instance, in compiler testing, one might compile a program, make semantically equivalent

(a) Input Image. (b) Recolored. (c) Recolored. (d) Recolored. (e) Input image. (f) Rotated.

Figure 1: Example of data augmentation applied to an image from dataset MNIST [15].



(a) Original Image A. (b) Transformed Image A. (c) Original Image B. (d) Transformed Image B.

Figure 2: Example of transformations applied in metamorphic testing for autonomous driving.

changes (such as reordering independent statements), and expect the compiler to produce the same executable. This ensures that the compiler's output remains consistent despite variations in the input code structure.
Recently, the application of metamorphic testing has expanded to include machine learning-based systems. Traditional testing mechanisms for these systems often rely on testing datasets, which are typically of small size due to the manual process of data collection and labeling. Furthermore, these datasets usually represent only a limited subset of all possible inputs, making it challenging to evaluate how well the system generalizes to unseen data. This limitation is further complicated by the Oracle problem [2], which refers to the difficulty of determining the correct output for a given input, especially when the input falls outside the scope of the available dataset. The Oracle problem is particularly prevalent in complex systems with complex output spaces, where the correct behavior cannot be easily computed.
Metamorphic testing provides several advantages over traditional testing, the main one is being able to tackle the Oracle problem. It generates new test cases (input-output pairs) through an automated process that exploits metamorphic relationships. Figure 1 reports two examples of metamorphic testing employed in the context of digit recognition. Specifically, Figure 1a reports a digit (manually) labeled as *three*, while Figure 1b, 1c, and 1d report different synthetic augmentations of the original input image that are still classified as *three*. Thus, the metamorphic relationship for digit classification exploited in this example is "Changing the background and foreground colors of the image does not modify the label associated with it". On the other hand, Figure 1e reports an image labeled as *six*, and Figure 1f reports the same image rotated by 180 degrees, which should be classified as *nine*. Thus, the metamorphic relationship used in this examples can be formulated as "Rotating an image classified as *six* by 180 degrees, results in an image that is classified as *nine*".

2.1.1 Metamorphic Testing for Autonomous Driving

In the context of autonomous driving [1], metamorphic testing is of great importance because of the vast number of possibilities a car might encounter on a road. Self-driving cars use a combination of sensors, algorithms, and real-time data to drive through various conditions.
Metamorphic testing can address the consistency of the system's behavior under different conditions, taking into account the metamorphic relations between inputs and outputs for these different conditions. These relations have to usually be kept unaltered: for example, a car's response to an object detection or wheel direction scenario should remain consistent even when external conditions such as lighting, weather, or the position of the object change. Applying these transformations and verifying that the autonomous system's responses are kept in line with the desired output, tests can ensure that the vehicle will react safely and predictably under various circumstances.
Furthermore, the amount of test data can be reduced by using metamorphic testing for this use case. As stated above, the focus of the test data now relates to a smaller set of base scenarios that can be generated and transformed to explore a wide range of conditions. This makes the testing process more effective and manageable, being able to detect faults and errors early in the development Figure 2 reports transformations

applied in the context of autonomous driving for steering angle prediction. It is possible to observe that the transformed images depict a road that is very similar to the original one, and require to be driven with driving commands that are similar to the ones of the original image.

2.2. Generative models for images

In recent years, text generative artificial intelligence models have gained huge popularity due to the spread of the popular Transformer architecture [32] in commercial models like ChatGPT [1], Gemini [2] and Claude [3]. Image generative models have also gained traction in mainstream media thanks to open-source technologies like Diffusion Models [12], Generative Adversarial Networks (GANs) [8] and Variational Auto Encoders [14].
Even though image generation is a hugely researched area, it still falls short of the attention carried by the text generation area. This is due to several factors:

- **Data availability**: for text data, there is an abundant volume of data repositories, as well as being easier to collect and filter. Image data is more difficult to filter and clean, as well as organize, making it more difficult to automate its collection.
- **Computational resources**: image generation models usually are more computationally demanding than text generation models.
- **Evaluation**: evaluating image generations is more subjective and lacks standardized metrics since no metric captures all aspects of a generated image due to the differences that can arise in style and detail.

Even though these factors pose important obstacles to consider, they also show that the field of image generation models encompasses a broader range of research areas while being a very attractive field.
In this thesis work, we will focus on diffusion models because they have shown great ability in generating images. Even though other generative image models exist, such as GANs (Generative Adversarial Networks) [9] and VAEs (Variational Auto Encoders) [14], these will not be discussed throughout the current thesis, as they fall outside the scope of this project. However, we encourage readers to explore these models further to understand how other models work and how they compare to diffusion models.

2.2.1 Probabilistic Diffusion Models

Among the many image generation techniques, diffusion models stand out as the most popular by achieving state-of-the-art performance in generative tasks [25]. Diffusion models are a type of generative model focusing on learning the underlying data distribution of the data being trained on. They are built upon the concept of diffusion processes, which are used to describe the probabilistic distribution of data. In more detail, they work through an iterative refinement process. The idea is to start through a noisy distribution, usually Gaussian noise, and gradually refine it towards the data distribution through a series of these diffusion steps. These steps try to reduce the noise while increasing the detail and fidelity of the image.
The training of a diffusion model involves two main phases. The first phase, known as the "noising phase", involves progressively adding noise, typically Gaussian noise, to an image over a series of N steps. In the second phase, called the "denoising phase", the model begins with the final noisy image from the first phase and iteratively trains a neural network to gradually remove the noise that was added, effectively reconstructing the original image. Figure 3 reports an example of the diffusion process. First, the input image depicting a cat is noised gradually through a series of noising steps. Then, the image is denoised gradually until the original image is returned.
Depending on how these two phases are taken upon, two main approaches for diffusion models arise: pixel space diffusion models and latent diffusion models.
In traditional pixel space diffusion models, the noising and denoising processes occur directly in the high-dimensional pixel space of the image. While effective at generating high-quality images, this approach is computationally expensive due to the large size of pixel-based representations, especially for high-resolution images. Each step in the diffusion process requires manipulating a large number of pixels, resulting in high memory and compute demands, which can significantly slow down both training and inference.
This is where Latent Diffusion Models (LDMs), such as Stable Diffusion, provide a major breakthrough. Instead of performing the diffusion process in the pixel space, LDMs operate in a compressed, lower-dimensional latent space. This latent space is typically obtained using a Variational Auto Encoder (VAE), which encodes high-dimensional images into a more compact latent representation. By applying the diffusion process within this smaller latent space, LDMs dramatically reduce the computational cost and memory requirements while still preserving the essential features needed to reconstruct the image. Once the denoising process is complete, the latent representation is decoded back into the full image using the VAE decoder.

[1] `https://openai.com/index/chatgpt/`
[2] `https://blog.google/technology/ai/google-gemini-ai/`
[3] `https://www.anthropic.com/news/introducing-claude`

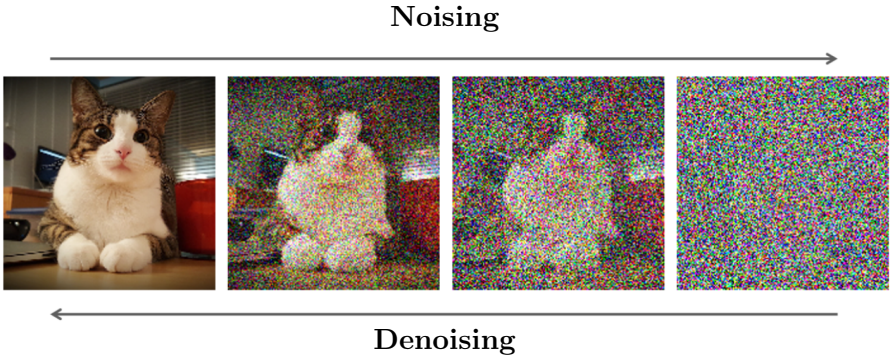


Figure 3: *Noising* process and *Denoising* process.

(a) G=0 (b) G=0.5 (c) G=1 (d) G=5 (e) G=7.5 (f) G=10

Figure 4: Example of images generated with Stable Diffusion using different guidance scales, denoted as G, for the same prompt: "A sunset over the mountains".

The result is that LDMs not only generate images faster but also with higher efficiency compared to pixel space diffusion models. The latent space allows the model to focus on more abstract, high-level features, enabling better generalization and more detailed image synthesis. This efficiency is particularly important when generating high-resolution images, as the dimensionality reduction allows for quicker processing without sacrificing output quality. Consequently, LDMs strike an excellent balance between computational efficiency and high-quality image generation, making them a superior choice for large-scale generative tasks in fields like art generation, image editing, and more.

2.2.2 Stable Diffusion: a Conditional Diffusion Model

Among the models using latent diffusion, Stable Diffusion [25] has emerged as the most popular and widely adopted approach, gaining significant traction in both research and industry. Its success can be attributed to its efficiency, scalability, and ability to generate high-quality images from text prompts as extra condition for the generation. For instance, on CivitAI [4], a platform that allows users to freely share customized checkpoints based on Stable Diffusion, it is possible to find more than thousands of customized models for all kinds of purposes (from generating landscapes to urban environments).
Stable Diffusion is a conditional diffusion model that can be guided through the usage of a textual prompt. A conditional diffusion model is an extension of the basic diffusion framework where the image generation process is guided by an additional input, or condition, such as text, class labels, or other forms of structured data. By incorporating this extra information, the model can generate images that adhere more closely to specific user-defined criteria. In the case of Stable Diffusion, this condition often comes in the form of text prompts, making it a text-guided conditional diffusion model.
For instance, in text guidance, a user provides a textual description such as "A sunset over the mountains" as input. The model then conditions the diffusion process to generate an image that aligns with the semantic meaning of the text. The process can also include varying levels of control through a parameter like the guidance scale, which determines how strongly the text condition influences the image generation. A higher guidance scale forces the model to stick more closely to the provided text prompt, whereas a lower scale gives the model more creative freedom. Figure 4 reports an example of images generated with Stable Diffusion using different guidance values and with the prompt "A sunset over the mountains". It is possible to observe that higher guidance values lead to images where the sunset and the mountains are more evident, while with lower values, it is possible to observe that images might not contain either the sunset or the mountains.
The simplicity and flexibility of guidance make it an incredibly powerful tool. It allows them to easily explore and generate a wide range of scenarios with minimal effort, just by altering the input. This enables quick

[4] `https://civitai.com/`

(a) Step 10 (b) Step 20 (c) Step 30 (d) Step 40 (e) Step 50

Figure 5: Example of denoising process of Stable Diffusion with prompt "Sunset over the mountains".

prototyping and experimentation, where new image variations can be produced in seconds by modifying or refining what we input into the model.
At its core, Stable Diffusion is built on the principles of diffusion models, which learn to generate images by progressively refining noisy data until it forms a coherent image. To understand how Stable Diffusion works, a more detailed explanation of the two main phases can be found next.

2.2.3 Training Stable Diffusion: An Overview

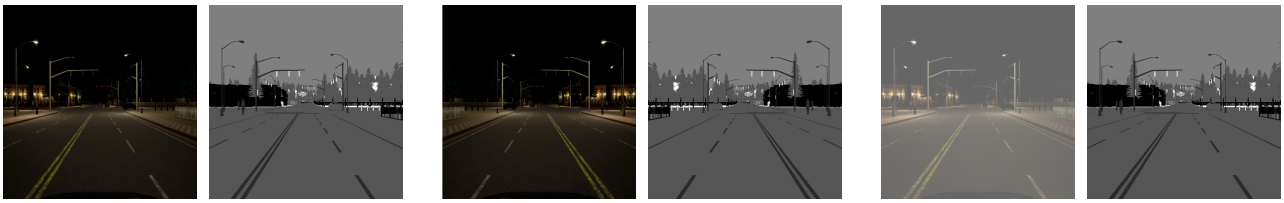
Training Stable Diffusion involves extensive practice on image-prompt pairs. During this training, the model learns to recognize and remove noise from images while aligning the results with the provided prompts. The training data is vast and diverse, encompassing countless images and their descriptions, which enables the model to generalize well to new and unseen prompts.
Here's a simplified look at how the training process unfolds:

- **Initial Pairing:** The model starts with clear images and their associated textual descriptions. Specifically, the original authors [25] used data from LAION [26], an openly available dataset of CLIP-Filtered pairs of images and corresponding text.
- **Adding Noise:** Gaussian noise is added to these images incrementally, creating multiple noisy versions.
- **Removing Noise:** The model learns to reverse the noising process, guided by the original clear images and descriptions. It practices removing the noise while trying to recreate the original image as closely as possible.

Over time, through many iterations of noising and denoising, the model becomes proficient at understanding the underlying structures and details that define clear images from their noisy counterparts.
Noising Phase. The noising phase is where we start. Having a clear, high-quality image, in the noising phase, Stable Diffusion takes this image and gradually adds Gaussian noise to it. Adding Gaussian noise is akin to adding a fine mist or static over a photograph, where each step of noising makes the original picture less visible and more chaotic (noisy).
The purpose of this phase is to simulate how images can degrade over time or through various distortions. By learning this degradation process, the model can understand what a noisy version of a given image looks like. This understanding is crucial because the ultimate goal of the model is to learn how to reverse this process effectively.
Denoising Phase The core of Stable Diffusion lies in the denoising phase. It is where the training of the model takes place. After an image has been sufficiently noised, the model undertakes the task of denoising it, aiming to retrieve the original image as accurately as possible. This phase is essentially about teaching the model how to clean up or denoise, to be more precise, the image, step by step, until it returns to its clear state.
Think of it like restoring a fogged-up window to its clear state, one wipe at a time. Each step in this process slightly reduces the noise, bringing the image closer to its original clarity. During training, the model learns to perform this denoising based on a vast number of image and noise pairs. It uses this learned capability to generate new, realistic images from random noise.

2.2.4 Generating New Images with Stable Diffusion

When it comes to generating new images, Stable Diffusion flips the process it learned during training. It starts with a completely noisy image, which is essentially a random pattern of pixels. Guided by a text prompt, the model uses the denoising techniques it mastered to transform this noisy image into a clear, meaningful picture that matches the prompt.
For instance, as seen in Figure 5, given the prompt "A sunset over the mountains", the model will start with a noise-filled canvas. Using the prompt as a guide, it incrementally refines the noisy canvas, removing noise in a way that gradually shapes the image into a landscape with the colors and elements of a mountain in the dawn



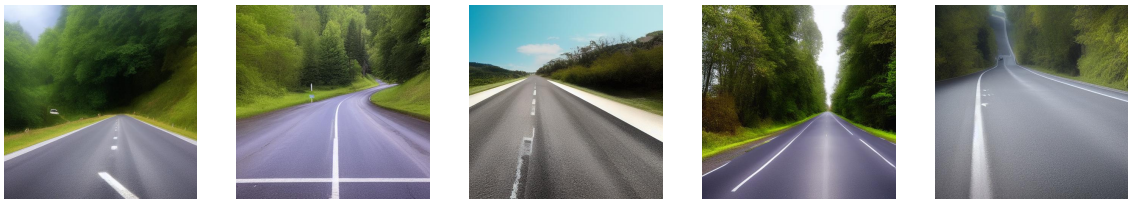
(a) Original Image. (b) Original Image Ground Truth. (c) Flipped Image. (d) Flipped Image Ground Truth. (e) Brighter Image. (f) Brighter Image Ground Truth.

Figure 6: Example of transformations applied in metamorphic testing for autonomous driving.

of the day. Note that the noise is added in the latent representation of the image. Thus, the result reported in the example is the decoding of the noisy latent space.
This process relies on the model's ability to interpret and integrate the details described in the prompt, applying them throughout the denoising steps to produce a coherent and visually appealing image.

2.3. Problem Statement

In the context of generative AI, new studies have concluded that we will have exhausted the stock of high-quality data for vision models in the next few decades, assuming the trends of computing power and models stay the same. This is why the creation of synthetic datasets will be of great importance in the years to come. Synthetic datasets allow researchers to control various factors such as lighting conditions, camera angles, and object properties, enabling the creation of highly tailored datasets for specific tasks. As the demand for data-intensive applications continues to grow, the development and utilization of synthetic datasets will play a crucial role in sustaining the advancement of vision models and ensuring their robustness and generalization capabilities in the face of evolving challenges.
Building on the need for diverse data to train vision models, data augmentation has become an essential technique in addressing the limitations of available datasets. Simple data augmentation approaches involve transformations such as flipping, rotating, and scaling images, which effectively create new variations of existing data. Other common techniques include adding noise, adjusting brightness or contrast, and applying blurring or sharpening filters. These methods, while relatively straightforward, significantly increase the diversity of data used to train models, improving their ability to generalize to new, unseen examples. For instance, as seen in Figure 6 flipping an image horizontally can help a model learn features independent of orientation, while adding noise or blurring can teach the model to be robust to distortions that might occur in real-world applications.
Beyond these basic methods, more advanced approaches to data generation involve generative models such as Generative Adversarial Networks (GANs) and Variational Auto Encoders (VAEs). These techniques are particularly powerful for generating synthetic datasets that go beyond the capabilities of basic augmentations, enabling the creation of entirely new images that reflect the statistical properties of the training set. By leveraging these models, autonomous driving testers can generate data that would be difficult or expensive to collect in the real world. Recently, diffusion models have gained traction as a promising alternative for data generation. These models work by progressively transforming noise into images, making them highly effective at creating diverse and detailed synthetic data. However, while diffusion models have shown significant potential, they often lack the precision needed for specific tasks and may fail to respect metamorphic relationships, which refer to consistent transformations or variations within a dataset. As shown in Figure 7, when asking for a road with vehicles in it ("An image of a road with two cars on the left, and one on the right"), Stable Diffusion does not fulfill the requirement as expected by the user.
This can be a critical limitation for applications where controlled changes, such as maintaining the geometry or physical properties of objects, are crucial for accurate model training. Consequently, while diffusion models are increasingly being used, their limitations underscore the need for further advancements in controllable data generation techniques.
Current solutions have proven to be very useful for metamorphic testing. Modifications to these algorithms have been proposed to improve their generative capabilities. Specifically, these variations have focused on making the generations more controlled through conditionings.
Guiding diffusion models through conditioning is a critical step toward gaining control over the image generation process. Conditioning provides additional information, such as textual descriptions, class labels, or partial images, to steer the model in the desired direction. However, while this helps generate more realistic and contextually appropriate images, it is not always sufficient for ensuring that the generated content adheres



(a) Generation A. (b) Generation B. (c) Generation C. (d) Generation D. (e) Generation E.

Figure 7: Example of five images generated by Stable Diffusion to the prompt "An image of a road with two cars on the left, and one on the right". The image generated by diffusion models changes by just using different starting noise.

(a) Original Image. (b) Transformed Image.

Figure 8: Example of transformations applied in metamorphic testing for autonomous driving with a clear error of mistaking a lorry for a building.

to specific, detailed constraints. Achieving precise control over what is generated is essential, especially for applications where consistency and accuracy are non-negotiable, such as in medical imaging, autonomous driving, or industrial design.

Without proper control, diffusion models may generate images that diverge from the intended outcome. For example, in autonomous driving, lack of control in synthetic data generation might lead to unrealistic road conditions or vehicle placements, undermining the effectiveness of the model's training, as seen in Figure 8. Therefore, refining the control mechanisms within diffusion models is paramount to ensure that generated outputs are not only visually appealing but also meet the strict requirements of the task at hand. Solving this problem is crucial to making these models more reliable for real-world applications where precision and alignment with specific constraints are critical.

Among the wide range of models we find, in this thesis, we focus specifically on T2I-Adapter [21]. A multimodal image generation algorithm created by Tencent in 2019 allows more than one conditionings to guide the model for a better generative task. When comparing T2I-Adapter and other multimodal models in terms of inference and training times, several key differences emerge. The T2I-Adapter, designed to enhance image-to-image translation with textual conditioning, generally offers a streamlined approach that can be more efficient in both the training and inference phases. Its architecture is optimized for integrating text-based instructions, which often simplifies the conditioning process and can reduce the computational overhead associated with training and generating images. This efficiency can lead to faster training cycles and quicker inference times, making the T2I-Adapter a practical choice for applications requiring rapid iterations and deployment.

Regarding the road-image generation use-case, control over generation is key for a critical task such as automatic driving data augmentation for several reasons.

Firstly, autonomous driving systems require vast amounts of data to train effectively, encompassing a wide variety of driving scenarios, weather conditions, and lighting environments. Generating this data synthetically through image generation models can significantly augment real-world datasets. However, to create these extensive datasets efficiently, the inference process must be controlled. The data augmentation task must respect the metamorphic relations between the newly generated images and the original and supervision over newly generated data is key.

Secondly, testers must have a fine-grained control over road-image attributes being able to manipulate textures, shapes, or styles. This means slight changes in input parameters should be able to affect certain parts of the generated image, while keeping others constant, ensuring validity and consistency are kept.

Lastly, the testing process must aid users in detecting faults or inconsistencies through controlled adjustments. Metamorphic testing can help reveal unintended changes in the expected output when transformations are made on an image, helping to identify potential weaknesses or biases. Robustness is a key quality to have for

(a) Sketch Conditioning. (b) ControlNet Generation. (c) Semantic Conditioning. (d) ControlNet Generation. (e) Depth Conditioning. (f) ControlNet Genearation.

Figure 9: Example of three images generated using ControlNet paired with input conditionings, with prompts "A cat", "An elephant in the wild" and "A modern living room", respectively.

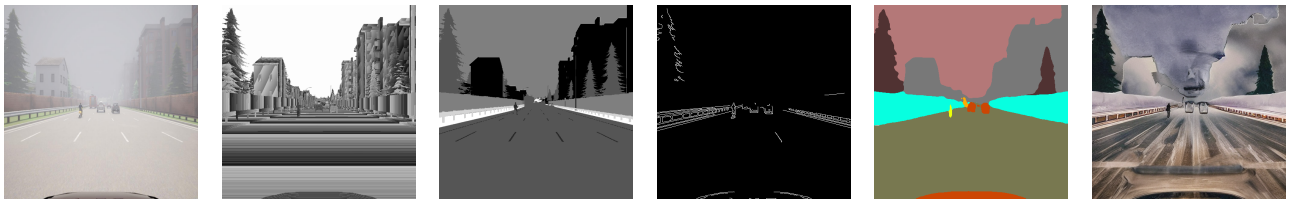
algorithms in deployment.

3. Related Work

In this section, we will go over the nature of the main state-of-the-art image-to-image algorithms and what the different use cases are for each of them. We will briefly explain how each tries to solve the problem at hand of controlling the generations and why they tackle only partially the problem presented in this thesis.
Overall, most of the openly available and used text-to-image and image-to-image models are based on or exploit the Stable Diffusion [25] architecture. By training a neural network composed of several convolutional neural networks, the model can denoise random noise and generate high-quality and realistic images.
These models enhance the behavior of Stable Diffusion and permit users to condition the denoising process using other conditioning inputs. This enables a better control over the generated images. This section explores three state-of-the-art models: ControlNet [35], Instruct Pix2Pix [3] and T2I-Adapter [21]. Each of these models builds on the foundational principles of Stable Diffusion but introduces different modifications.

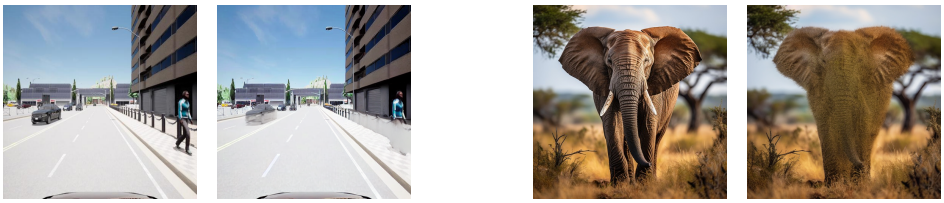
3.1. ControlNet

ControlNet [35] is an extension of the Stable Diffusion framework designed to offer enhanced control over the image generation process. Unlike the vanilla Stable Diffusion model, which relies on text prompts to guide the creation of images from noise, ControlNet integrates additional conditioning inputs, enabling more precise manipulations of the output image.
ControlNet operates by introducing control signals into the diffusion process. It does this by adding an extra convolutional neural network for each denoising step in parallel with the existing network. This new convolutional neural network gets fed the conditioning images to be later combined with the output of the existing network. This architecture seamlessly integrates additional inputs with the stable diffusion framework in a natural and efficient manner. Through it, ControlNet stands out for its robust handling of multi-modal inputs and excels in scenarios where a predefined structural input is essential. Because of this, it has become the predominant algorithm for image generation with extra input conditionings. In Figure 9, some examples of ControlNet-generated images are shown. These examples show the use of different conditionings and guidances with ControlNet.
Figure 9a provides the edges of an object as the conditioning input. The result, shown in Figure 9b, is an image that not only follows the given textual prompt but also incorporates the structural guidance provided by the edges from Figure 9a. This demonstrates how ControlNet can guide the generation process using both textual and visual inputs, ensuring that the final output respects the conditioning provided.
This behavior can be extended to different kinds of conditioning inputs. For example, Figures Figure 9c and Figure 9d report an example of semantic conditioning, while Figure 9e, and Figure 9f illustrate an example of depth conditioning. Each conditioning input adds a layer of control over the image generation process, making ControlNet a versatile and robust tool for tasks requiring precise adherence to both textual and multimodal guidance.
ControlNet has limitations in accommodating more than two conditionings, which can be insufficient when complex image generation requires enforcing multiple constraints. For example, if you need to apply two conditionings (e.g., semantic and sketch conditionings) to guide the image creation process, ControlNet capacity is restricted, preventing the combination of these inputs. In Figure 10 an example is shown where two conditionings are not enough to generate the expected image. The ground truth conditioning shown cannot be associated with the image, thus the metamorphic relationship between the original image and output image are



(a) Original Image. (b) Ground Truth Depth. (c) Ground Truth Semantic. (d) Sketch Conditioning. (e) Semantic Conditioning. (f) ControlNet Generation.

Figure 10: Example of a bad generation using ControlNet with both Segmentation and Sketch conditionings, paired with with the prompt "A car is driving down a foggy road with other cars and a motorcycle".



(a) Original Image. (b) Instruct Pix2Pix Generation. (c) Original Image. (d) Instruct Pix2Pix Generation.

Figure 11: Example of two bad generations using Instruct Pix2Pix paired with with the prompts "Remove the person and vehicle from the image", and "Remove the tusks from the elephant", respectively.

not robust enough. Furthermore, it does not support color conditioning, which is critical for controlling the correct color range in the generated image. Relying solely on a text prompt is often not enough, as text-based descriptions may fail to capture detailed requirements for color leading to less precise or unsatisfactory results. Again, in Figure 10, the same example of a ControlNet generation is shown where a text prompt conditioning is not enough to shift the generation to the desired generation. Therefore, a more flexible solution capable of handling multiple conditionings is necessary for generating high-quality, controlled images.

3.2. Instruct Pix2Pix

Instruct Pix2Pix [3] is an extension of the Stable Diffusion framework designed to modify images based on detailed textual instructions. Unlike ControlNet, it does not use extra image conditionings, in fact, it only alters the Stable Diffusion architecture by adding an image input.
Instruct Pix2Pix learns to edit images from a synthetically generated dataset. The dataset is generated in the following way. First, a pair composed of a textual description of an image and editing instruction is fed to a large language model (i.e., GPT-3 [4]) in order to obtain a textual description with the editing instruction applied. For example, the textual description can be "A dog sitting in the park" and the editing prompt can be

(a) Original Image. (b) Instruct Pix2Pix Generation. (c) Original Image. (d) Instruct Pix2Pix Generation.

Figure 12: Example of two generations using Instruct Pix2Pix paired with with the prompts "Turn the elephant blue and set it in the jungle", and "Turn him into a cyborg", respectively.

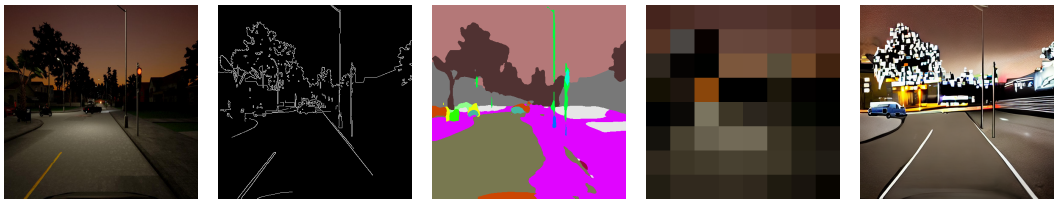
(a) Depth Conditioning. (b) T2I Adapter Generation. (c) Sketch Conditioning. (d) T2I Adapter Generation.

Figure 13: Example of images generated using T2I Adapter with Depth and Sketch conditioning, paired with the prompts "A photo of a room", and "A robot, Mount Fuji in the background" respectively.

"Make it running", while the results expected by the large language model is "A dog running in the park". Then, the pair of textual description and modified textual description are fed into a text-conditioned generative model (i.e., SDEdit [19]) which generates the images corresponding to the initial and modified textual description. Finally, Instruct Pix2Pix is trained with the pairs composed of the image corresponding to the original textual description and editing instruction with the objective of learning to generate the image corresponding to the modified textual description.
Like ControlNet, Instruct Pix2Pix struggles to handle multiple layers of edits on an original image. While it excels at adding new objects or making minor adjustments, it fails when tasked with generating an entirely new image that involves significant changes across several aspects, as it is shown in Figure 11: on the left example an image is passed to the model of a road along with the instruction of removing the person and vehicle from it. The model, while being able to keep most characteristics of the original image consistent, does not follow the instructions completely, blurring the car and person instead of removing them completely. Additionally, it lacks the fine-grained control needed for precise color changes or object placement. It is also prone to introducing unwanted changes or distortions on the image when trying to apply more complex edits. Its capabilities are better suited for subtle modifications rather than complex transformations, as shown in Figure 12: on the left example, the color of the original image of the elephant is asked to be modified, along with the setting, which is done successfully. It is also successful in maintaining the image quality and its overall context.

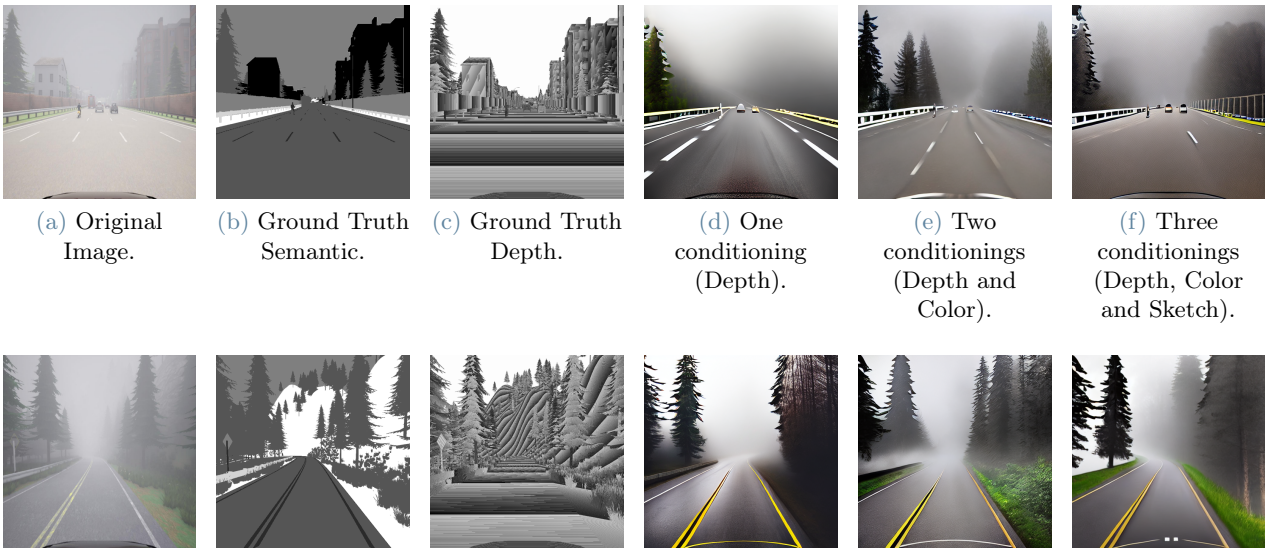
3.3. T2I-Adapter

T2I-Adapter [21] is a modified version of Stable Diffusion allowing additional control over the image generation process. As ControlNet, it maintains a frozen Stable Diffusion architecture, while having a parallel architecture plugged into the existing model.
It works by connecting the Stable Diffusion architecture with the inputs inserted into the model. As with the Stable Diffusion architecture, the encoded text prompt input is inserted at each denoising step, but not alone. It additionally combines the extra conditionings to the encoded text prompt, permitting extra control to condition the output for a better generation.
Additionally, T2I-Adapter offers a wider range of conditioning compared to ControlNet. Like ControlNet, it allows integrating several conditionings inputs simultaneously. This enables users to apply multiple constraints to the generated image with greater precision, offering more options to select from based on which details are most important for generating new images. Some examples of image generations using T2I-Adapter can be seen in Figure 13.
Furthermore, T2I-Adapter allows the combination of more than two conditionings simultaneously, while ControlNet allows up to two. In practice, however, T2I-Adapter is typically used with a single conditioning or at most two. As shown in Figure 14, introducing a third conditioning often leads to a decline in the quality of the generated images. One can see that the details from the sketch and semantic segmentation conditionings are not preserved, while color is. The model has completely removed the car in the middle lane and substituted the motorcycle on the left for a strange car. The added complexity tends to overwhelm the model, resulting in outputs that may not fulfill the intended details.
Moreover, since model weights for current implementations do not generate good images when combining more than two conditionings, fine-tuning the models' parameters might help them better handle the added complexity. This could potentially lead to more precise and controllable outputs, expanding the model's resourcefulness in image generation tasks.



(a) Original Image. (b) Sketch Conditioning. (c) Semantic Conditioning. (d) Color Conditioning. (e) T2I Adapter Generation.

Figure 14: Example of a bad generations using T2I Adapter with Sketch, Semantic Segmentation and Color conditionings, paired with with the prompt "A car is driving down a street at night. There are two other cars on the road, and a motorcycle is parked on the side. A traffic light is visible in the distance".



(a) Original Image. (b) Ground Truth Semantic. (c) Ground Truth Depth. (d) One conditioning (Depth). (e) Two conditionings (Depth and Color). (f) Three conditionings (Depth, Color and Sketch).

(g) Original Image. (h) Ground Truth Semantic. (i) Ground Truth Depth. (j) One conditionings (Depth). (k) Two conditionings (Depth and Color). (l) Three conditionings (Depth, Color and Sketch).

Figure 15: Example of using one, two and three conditionings from an original image. Road properties are preserved as the number of conditionings grow higher.

3.4. Multiple Adapters in Stable Diffusion

Building upon the capabilities of individual models like ControlNet, T2I-Adapter, and Instruct Pix2Pix, the concept of using multiple adapters along Stable Diffusion offers even more flexibility and control in image manipulation. Multiple adapters enable the combination of different guiding principles and input modalities, allowing for complex image transformations.

Multiple adapters refer to the integration of several distinct adapters into a single Stable Diffusion framework, as explained in previous sections. Each adapter is designed to handle specific aspects of the image generation or transformation process. By combining these adapters, the model can simultaneously manage multiple inputs or guiding constraints, producing images that meet a broader set of requirements, each adapter focusing on a specific condition of the image.

One of the primary benefits of multiple adapters is their ability to balance different types of guidance, leading to outputs that are coherent and well-aligned with the intended design specifications. This capability is invaluable in applications where maintaining a balance between structural integrity, stylistic coherence, and detailed modifications is crucial.

This approach allows for more complex and detailed image manipulations, providing a solution that can serve different image generation needs. It represents a significant advancement in propelling the foundational principles of Stable Diffusion to achieve customized and refined outputs.

(a) Original image. (b) Ground Truth Semantic. (c) Sketch Conditioning. (d) Semantic Conditioning. (e) Depth Conditioning.

(f) Output 1. (g) Output 2. (h) Output 3. (i) Output 4.

Figure 16: Example of images generated with the same conditionings using the prompt "An empty road during a cloudy night".

An example is shown in Figure 15 where the user can clearly visualize the increasing control that can be achieved by using one, two, and three conditionings. In the first example, the road conditions are preserved by using just one conditioning, but the vehicles on the road are not clearly shown until using three. In the second example, though the conditions have been well preserved since the first generation, the direction of the road is only clear as in the original image when using three conditionings. It is also clear in this example that even though the generated images retain the most important characteristics of the original image, they lack detail and tradeoff quality for control. This issue will be the main focus of the solution we propose below.

4. Methodology

This section introduces the main aspects of the proposed solution that provide more control in the image generation process. First, it introduces an overview of the solution and the main design choices. Next, it details the types of image conditionings chosen to enhance our model. Then, it explains how these conditionings are integrated to support the selected architecture. Finally, it outlines the training process of our algorithm.

4.1. Solution Overview

The solution proposed in this thesis consists of fine-tuning of a diffusion model to generate custom, realistic test scenarios for autonomous driving systems using multiple conditioning inputs.
Our solution is based on a latent diffusion process, as for other state-of-the-art solutions. This permits to generate a diverse set of images given the same input conditionings but by just changing the random noise from which the denoising process starts. Figure 16 provides an example of this feature. This property is particularly valuable, as it enables testers to generate a large variety of diverse test cases from a single input.
Our solution is built on top of the T2I-Adapter model (see Section 3.3), which was selected for its architectural flexibility and efficiency. This choice allows for the seamless integration of multiple conditioning inputs, such as edges, color maps, and semantic labels, to generate highly specific and realistic test cases for autonomous driving systems. Unlike ControlNet, which would require significant structural modifications to support multiple conditionings, the T2I-Adapter model handles these inputs more naturally without altering its core architecture. Furthermore, Instruct Pix2Pix, another potential candidate, was not considered due to its lack of support for conditioning inputs altogether, making it unsuitable for the task.
A key advantage of the T2I-Adapter is its ability to handle a broader range of conditionings than models like ControlNet. For instance, T2I-Adapter supports color-based conditioning, which ControlNet does not, giving testers more control over the details of the generated images. This wider range of conditionings is particularly useful in generating diverse and complex driving scenarios, such as varying weather conditions, traffic densities, and road structures, which are essential for testing autonomous vehicles.
Additionally, the T2I-Adapter is slightly smaller in size than ControlNet, making it a more efficient model in terms of computational load. This translates into faster image generation times, which is particularly beneficial

when large volumes of test data are needed for extensive testing pipelines. The ability to generate images quickly while maintaining high quality ensures that the model can be effectively used in real-world testing environments, where rapid iterations are often required.
Finally, the proposed solution was fine-tuned using the SHIFT dataset, which was collected within the CARLA simulation environment. This dataset includes a wide variety of weather conditions, traffic patterns, and road layouts, making it an ideal training ground for generating test cases that closely resemble real-world driving situations. By fine-tuning the T2I-Adapter on this dataset, the model can generate diverse, high-quality images that represent the complex conditions autonomous driving systems may encounter.

4.2. Input Conditionings

This section introduces the main input conditionings we used for generating new test cases for autonomous driving. Specifically, as anticipated in Section 2 and Section 3, there is a wide range of conditionings to choose from. Each conditioning input might highlight a specific characteristic a tester might want to keep from an original image onto the generated one. This is why, it is key to understand which conditionings are better for the specific use case of autonomous driving. While some might be useful for tasks like portraits, they might not provide enough information for autonomous driving. The input conditionings exploited by our solution are the following:

- **Color conditioning**: provides the ability to distinguish objects by color differences as well as adding context to the image for time of the day, weather conditions, and type of road. The process to obtain color conditioning is very straightforward: first shrink the image with smooth interpolation, calculating pixels through neighboring pixels; then resize back to its original size using sharp interpolation, using all pixels with the same value. Figure 17 reports examples of color maps corresponding to cloudy, sunset, and sunny scenes, and the corresponding images generated using these inputs.
- **Semantic conditioning**: aids through pixel-level information on object distinction, being able to isolate objects. Plus it is able to distinguish different levels of the same object. The process to obtain a semantic segmentation conditioning is done in the following way: a neural network is applied to the image to classify each pixel to a specific class, coloring with the same color pixels with the same class. Some examples can be found in Figure 18 showing a light blue color for the road, dark blue for vegetation, black for sidewalks, and red for buildings.
- **Sketch conditioning**: useful to understand the outlines of objects aiding in avoiding distractions of colors and textures, which is very useful for road images. Sketch conditioning images are obtained through an edge detection process: sharp changes in intensity or color within an image are identified, which usually signify boundaries between objects or regions, and a filter is passed to suppress possible noise. Examples are shown in Figure 19 where the edges of the road lines, vehicles, and buildings are highlighted.
- **Depth conditioning**: provides the distance of objects from the image perspective plus object placement in the image, which is of great relevance in road images. Depth conditioning is typically obtained using a convolutional neural network trained for depth estimation. It analyzes spatial relationships in the image to obtain distances of features in it. A few examples of depth conditioning are shown in Figure 20 where lighter colors are used for closer objects, as seen in the third example for a street lamp, and darker objects for ones being farther, as seen for the trees in the second example.

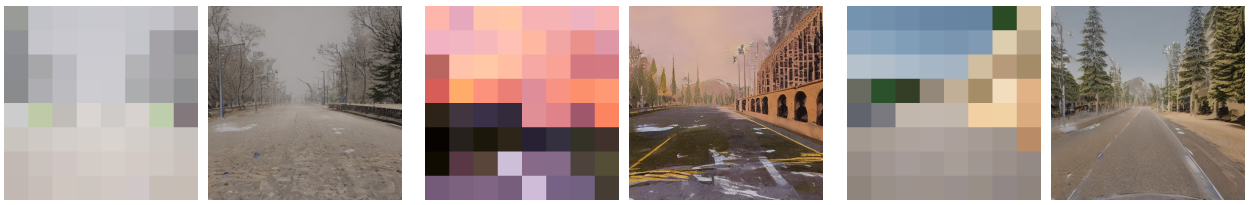
On the other hand, other conditionings were discarded due to them being not relevant to our use case. It is important to review some of these. For a more visual following, see Figure 21:

- **Canny conditioning**: while useful for tracing shapes of objects, is considerably similar to Sketch conditioning and can be perfectly replaced by it.
- **OpenPose conditioning**: it does not provide enough useful information to the model, perhaps only useful if wanting to focus on pedestrian presence and sacrifice all other aspects.
- **Style conditioning**: while useful for generating images resembling a specific style, that of a painter or a type of cartoon, would not be relevant for our use case.

It is important to highlight that Semantic conditioning and Depth conditioning, while having part of their implementation in place, were not fully supported as these features were not included in the Stable Diffusion 1.4 version of T2I-Adapter, but in subsequent ones. Consequently, their functionalities were implemented in the code base.

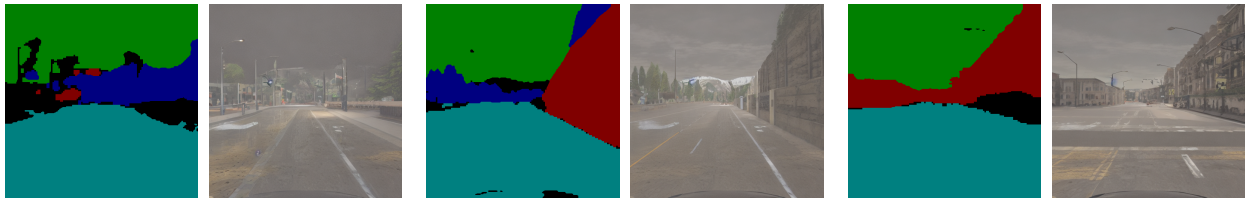
4.3. Mixing Conditionings

An important part of our solution is how it mixes the conditionings that are fed to the diffusion process. The key component of T2I-Adapter is the CoAdapter module. It is the part responsible for fusing multiple sources of conditioning information. It can be better understood through a step-by-step explanation and the visual



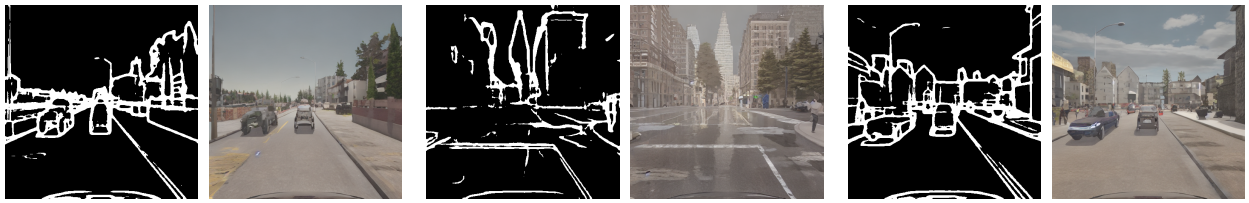
(a) Cloudy Color. (b) Cloudy Output. (c) Sunset Color. (d) Sunset Output. (e) Sunny Color. (f) Sunny Output.

Figure 17: Example of pairs of color conditioning and generations with prompt "A snowy road".



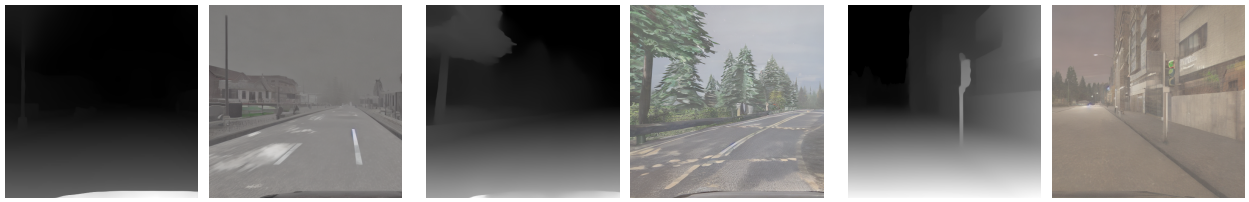
(a) Semantic A. (b) Output A. (c) Semantic B. (d) Output B. (e) Semantic C. (f) Output C.

Figure 18: Example of pairs of semantic segmentation conditioning and generations with prompt "A road during the night".



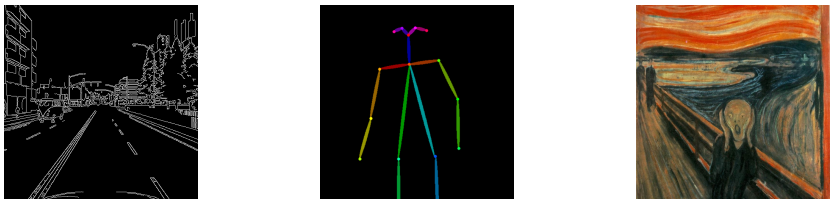
(a) Sketch A. (b) Output A. (c) Sketch B. (d) Output B. (e) Sketch C. (f) Output C.

Figure 19: Example of pairs of sketch conditioning and generations with prompt "A road in a city".



(a) Depth A. (b) Output A. (c) Depth B. (d) Output B. (e) Depth C. (f) Output C.

Figure 20: Example of pairs of depth conditioning and generations with prompt "A road in a city", "A road with trees to the side" and "A road with a traffic light to the side".



(a) Canny Conditioning. (b) OpenPose Conditioning. (c) Style Conditioning.

Figure 21: Respectively, Canny conditioning image, OpenPose conditioning image and Style conditioning inputs.

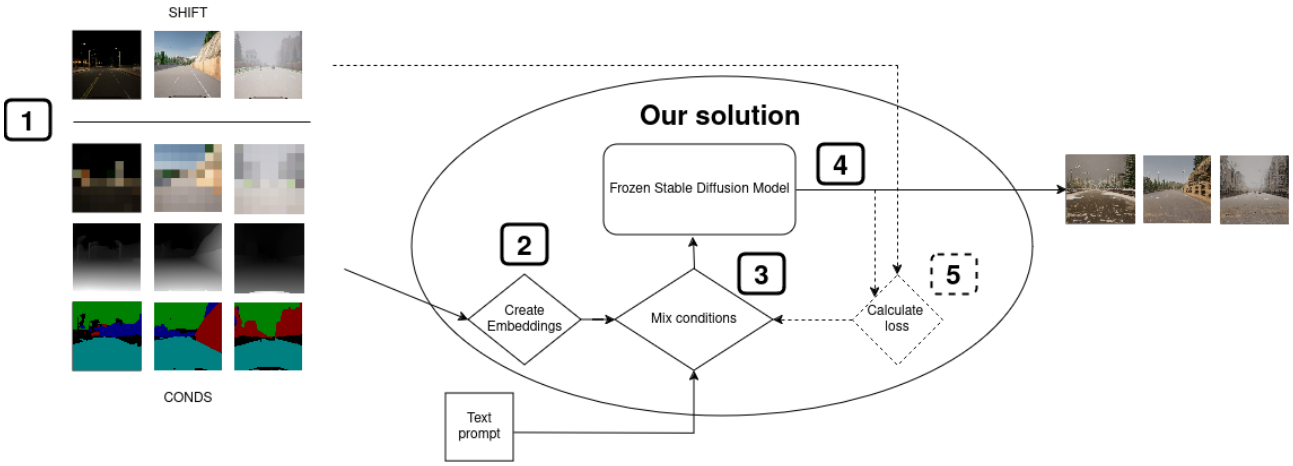


Figure 22: Our solution.

guide from Figure 22:

1. **Conditioning Extraction**: T2I-Adapter takes in two types of conditionings: text conditioning, represented by the textual prompt, and auxiliary conditionings, represented by the visual conditionings discussed in section 4.2.
2. **Conditioning Embedding**: each conditioning is embedded into a common latent dimensional space. The objective of this step is to be able to represent all conditionings in the same space so that they can be combined. For textual conditioning, a CLIP [23] text embedder is used, while for the auxiliary conditionings, pre-trained convolutional neural networks are used, different for each conditioning.
3. **Fusion**: The fusion process involves combining the embeddings to create a unified representation that can guide the diffusion process. This is done through the concatenation between conditioning embeddings and by applying the attention mechanism [32]. Furthermore, we use the cross-attention mechanism to allow the model to attend to different conditioning sources differently. In other words, the model can focus on certain parts of the conditioning information based on the current state of the image generation process.
4. **Guiding the Diffusion Process**: Once the conditioning information has been combined, this unified representation is passed to the diffusion process. This is done through what is used as text-prompt embedded input in the denoising process (the same as in Stable Diffusion).
5. **Loss calculation and model update**: When we are retraining the model, after generating an output image, the model compares this result with the real image by calculating a loss function, which measures how different the generated image is from the target image. Based on this loss, the model updates the weights of both the conditioning fusion component and the embedding models through backpropagation, allowing it to improve its accuracy in attending to the different conditioning sources.

Further modifications of the code had to be made to be able to use more than two conditionings while using Stable Diffusion 1.4. including adjustments to the conditioning pipeline and fusing functionalities. Even though in the current thesis we only experiment with up to three conditionings, the possibility of using four or more is possible but not explored due to limited computational resources. By considering three conditionings, four different combinations are possible. This will be discussed in Section 4.4.3.

4.4. Training

This section explains the details of the fine-tuning process employed to teach the model to learn how to generate new images while following the constraints specified in the input conditionings.

4.4.1 Trainable and Frozen Components

The proposed methodology is based on fine-tuning the T2I-Adapter architecture. This is obtained through a technique named Transfer Learning [38]. The pre-trained components remain static to retain their learned knowledge, while new layers or task-specific components are trained. Transfer learning is beneficial because it ensures that the model continues to leverage the extensive knowledge gained from the large datasets on which these components were originally trained. This prevents overfitting and reduces the need for excessive computational resources while allowing the model to specialize in new tasks by focusing only on the components

that require adaptation, such as handling multiple conditioning inputs. Additionally, training the model from scratch would require an immense amount of data and computational resources that are beyond our current capacity. The pre-trained components, having been trained on extensive datasets, provide a shortcut to achieving high-quality outputs without the need for vast amounts of additional data. Starting from scratch would not only require far more data than we have access to but would also involve intensive computation, making it infeasible within the scope of this thesis.
Specifically, the weights of the Stable Diffusion model and the CLIP text embedder are left fixed during training. This is a deliberate design choice to preserve the pre-trained generative capacity of Stable Diffusion and the rich semantic understanding of CLIP. These components, having been trained on extensive datasets, provide a robust starting point for generating high-quality images from text prompts. By freezing their weights, the training process becomes more efficient and focuses only on the parts of the system that need adaptation, ensuring stability and consistency in the model's performance. As a result of freezing the weights of the text encoder, we expect the model to increasingly rely on the image conditioning inputs to provide more detailed and relevant insights about output. This shift in emphasis can enhance the alignment between the generated images and the conditioning data, leading to more accurate and contextually rich visuals. However, a potential risk arises in that the frozen text encoder might be underutilized or potentially ignored, causing the system to focus more or even only on the image features. Despite this, freezing the text encoder was a necessary compromise, as retraining it would have significantly increased computational demands and complexity.
On the other hand, the fuser and the Adapters represent the parts of the algorithm that are actively trained. The fuser is responsible for integrating the outputs from different conditioning sources, such as semantic segmentations, depth maps, or other guiding information. The Adapters, in turn, are the modules that process these additional conditionings, allowing the model to incorporate multiple inputs effectively. Training these components enables the system to adapt to new tasks and conditionings without disturbing the core generative functionality of Stable Diffusion.
Compared to other State-Of-The-Art (SOTA) models, the training procedure for the T2I-Adapter stands out because it does not involve retraining Stable Diffusion or the CLIP embedder. In contrast, many SOTA models fine-tune both the generative backbone and the text encoding components during training. This difference highlights the modular nature of the T2I-Adapter, where the focus is on training only the additional components (the fuser and Adapters) while leveraging the power of pre-trained, frozen models. This approach allows for more focused and efficient training, potentially reducing the computational cost and time required compared to other systems that retrain their entire architectures.

4.4.2 Fine-tuning process

In this thesis, the fine-tuning process of the diffusion model follows a structured approach that shares similarities with the training procedures seen in SOTA models [5]. Since the model in question is a diffusion-based generative model, the training begins by loading the dataset, which is subsequently divided into smaller groups or "batches" of data. These batches are passed through the neural network during each training iteration, a process referred to as ´´mini-batch gradient descent". The purpose of this is to incrementally adjust the model's weights, allowing it to learn from the data gradually.
One key aspect of this training process, specifically for diffusion models, is the noise-adding mechanism. At each training step, random noise is introduced into the images, which the model is trained to progressively denoise. Over time, the model learns how to reverse the diffusion process, effectively transforming noise into realistic images. During fine-tuning, conditioning inputs such as image edges, color maps, or semantic labels are passed alongside the noisy data to guide the image generation toward specific goals. A notable feature in this work is that certain conditioning inputs, such as edge maps, are generated dynamically using the data loader during training. This allows the model to adapt to different conditioning scenarios in real-time, improving the diversity of training data.
Additionally, a validation step occurs after a fixed number of batches. Here, a separate validation dataset is used to assess the performance of the model without updating the weights. The algorithm can save both the training data and the images it generates side by side, offering a visual comparison of the learning progress.
Due to the large size of the model, checkpoints are saved at regular intervals—typically every hour. This precaution is crucial in case of hardware failures or interruptions, as it ensures the training process can be resumed from the last saved checkpoint without losing significant progress. In this way, the model can be efficiently fine-tuned even when dealing with resource constraints.

[5] urlhttps://huggingface.co/docs/diffusers/v0.13.0/en/training/text2image

4.4.3 Configurations

This section outlines the different configurations designed for our approach, which leverages three out of four available conditioning options: Color, Depth, Semantic, and Sketch conditioning. Since selecting three conditionings from the available four results in four possible model configurations, each configuration emphasizes different aspects of the generated images, offering a range of flexibility in controlling the output. The supported configurations for image conditionings are:

- **Color-Depth-Semantic**: Would likely prioritize a combination of rich color information, depth perception, and object differentiation, while sacrificing edge definitions.
- **Color-Depth-Sketch**: Cases where edge definitions are essential alongside color and spatial depth might benefit, while object differentiation should take a secondary role.
- **Depth-Semantic-Sketch**: Using this model, we would expect to achieve excellent edge and object definition, though at the expense of color accuracy in the generated image.
- **Color-Semantic-Sketch**: This configuration is likely to emphasize a balance between vivid color representation, strong object differentiation, and well-defined edges. It could be particularly useful in scenarios where clear visual distinctions between objects and sharp outlines are necessary, while depth information may be less critical.

The configurations used for prompts in combination with the image conditionings were:

- **Long prompt**: Image descriptions of over 20 words, focusing on all aspects of the original image, such as other vehicles, pedestrian presence, environment type, lighting, and time of day, plus other important characteristics of the image intrinsic to each image.
- **Short prompt**: Image description of up to 20 words, focusing only on the most important characteristics of each image, without considering minuscule details. Aspects like environment type or background details are overlooked.
- **No prompt**: A configuration where no prompt is provided to the model is added to understand how text prompts influence training, and whether enough control can be achieved with the image conditionings.

Each configuration offers different strengths and weaknesses, and the choice of conditionings significantly influences the generated content. While the supported set of conditionings enhances the ability to control the generation process, it is difficult to predict a priori which combination will perform best. As explored in the next section, we systematically evaluated all the different combinations along with varying prompt styles to determine the optimal configuration. These evaluations are designed to compare the configurations against each other as well as against existing state-of-the-art models, highlighting the trade-offs inherent in each setup.

5. Evaluation

This section introduces the experimental evaluation we conducted to assess the proposed solution. First, we outline the experimental setup, followed by a presentation and discussion of the results. Our experiments aim to answer the following research questions:

- **RQ1. Analysis of mixing conditionings.** Which is the best configuration of our solution?
- **RQ2. Validity and realism.** Which configuration generates the most valid and realistic images?
- **RQ3. Comparison with state-of-the-art.** How does our solution compare with SOTA baselines?

These questions are designed to systematically evaluate our models and their effectiveness. The first question seeks to identify the optimal configuration of conditionings and prompts, providing insights into which combination yields the best results for image generation. The second question focuses on the validity and realism of the generated images, which is crucial for ensuring that the outputs are practical and useful in real-world applications. Finally, the third question aims to benchmark our approach against existing state-of-the-art methods, allowing us to assess our models' performance relative to leading techniques in the field. Addressing these questions will provide a comprehensive understanding of our models' strengths, limitations, and potential areas for improvement.

5.1. Experimental Setup

This section introduces the setup of our experiments. First, it describes the dataset and then outlines the conditioning models used for each type of conditioning.

5.1.1 Dataset

Among the numerous available road image datasets, we selected the SHIFT dataset [30] primarily due to its large size and the diversity of conditions it covers. The SHIFT dataset contains approximately 2.5 million

(a) Scenario 1. (b) Scenario 2. (c) Scenario 3. (d) Scenario 4.

Figure 23: Example of images belonging to SHIFT dataset.

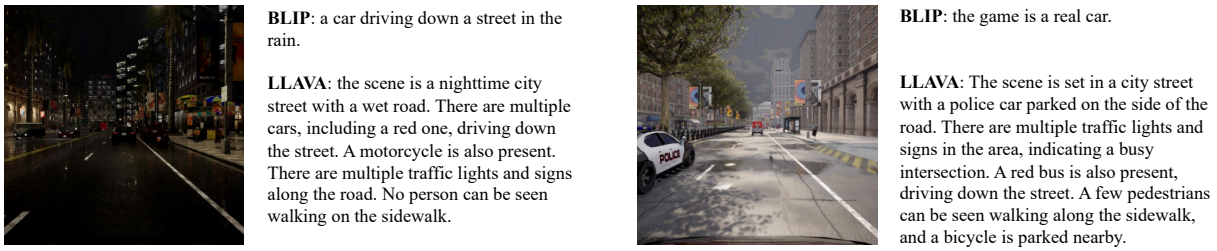


Figure 24: Example of text descriptions generated by BLIP versus LLAVA.

images, organized into training, testing, and validation subsets of 2D data. The dataset is further divided into continuous and discrete shifts. In this thesis, we exploited discrete shifts. Additionally, the images are categorized by perspective: front, right, and left. For our purposes, we concentrated on the images taken using the front-facing camera.
Even though the images are not real-life photographs, they are generated through videos mimicking the real world through shifted conditions. Not only is its size very beneficial for training, but it also contains images captured under diverse driving scenarios from a driver's perspective. It includes a diversified portfolio of images that covers a wide range of variables like weather conditions, time of day, image angle, types of road, vehicle and pedestrian presence, or environment type. Figure 23 reports four images available in the dataset, each representing a different scenario. Figure 23a shows a night setting, Figure 23b depicts a sunny setting, Figure 23c represents a vegetation setting and Figure 23d illustrates an urban setting. These data will serve as a robust foundation for training and evaluating models. Furthermore, both the CARLA simulator and the SHIFT dataset are very popular choices in the autonomous driving research field.

5.1.2 Conditioning Models

This section introduces the models we used to extract conditionings for text and auxiliary images.
Text Prompts. The one disadvantage that we encountered with the SHIFT dataset was that no type of textual description of each image was provided. Since this is a dataset whose primary focus is aiding in training for autonomous driving algorithms. For this reason, we employed an automated technique named Image Captioning [28] capable of generating textual descriptions from images. In this thesis, we considered two state-of-the-art techniques: BLIP [16] and LLAVA [17].
BLIP is a lightweight model designed for efficient image-text alignment, while LLAVA, though more powerful in terms of handling complex images, is significantly heavier in terms of computational resources. In our experiments, a GPU with 12GB of memory was enough to run the first, while the second required a more powerful GPU equipped with 24GB of memory. Figure 24 compares the two models provided for the same images with the prompt: "This image is taken from the first-person perspective of a car. Describe what you see in the scene. Take into account Environment Type, Weather Conditions, Pedestrian Presence, Traffic Lights, Traffic Signs, Time of the Day, Road Surface Condition, Road Markings, Vehicle Presence". After experimenting with various prompts, we found that this particular prompt produced the most accurate and detailed image descriptions overall.
The prompt length was also an issue since LLAVA captions were usually longer than those provided by BLIP, even in cases where the text instruction to the model specified for the text output to have a bigger length.

(a) Original Image. (b) Depth Ground truth. (c) Depth Algorithm. (d) Semantic Ground truth. (e) Semantic Algorithm.

Figure 25: Example of conditionings from SHIFT and generated by an algorithm.

Adding to the previous prompt the instruction "Do it in less than 20 words" still sufficed to have a much better prompt than the ones provided by BLIP. Overall, the LLAVA captions were of superior quality compared to the BLIP captions. However, the BLIP captions had the advantage of being significantly faster at being generated. It is important to note that, to try to bypass the speed bottleneck, a small study on the different parameters was made for LLAVA, but the generation of each prompt still required about 4 seconds per image.
Consequently, the approach of generating prompts through a multi-modal model did not seem too appealing. The second option we considered was training the model with an empty prompt. The idea behind this approach was to attempt to nullify the text conditioning effects inherent in the pre-trained model, thereby focusing the training process on the other forms of conditioning. By minimizing the influence of text-based cues, our goal was to enhance the model's responsiveness to alternative conditioning factors. We hypothesized that this would allow us to still achieve a balanced and comprehensive conditioning framework.
While the multi-modal model could serve as a more attractive approach in terms of detailed and high-quality descriptions added to control image generation further, its feasibility was unattainable. The option of an empty prompt presented a more viable path, aligning better with our objectives and optimizing model performance in the hopes of maintaining control.
Specifically, in this thesis, we used a pre-trained BLIP model with 86M parameters (checkpoint is available at [6]. Concerning LLAVA, we use the pre-trained version with 7B parameters (checkpoint is available at [7]).
Image Conditionings. The generation of image conditionings also had two options. In our model pipeline, depth, semantic segmentation, color, and sketch conditionings can be generated through dedicated algorithms integrated into the processing workflow. Both of these conditionings are generated as explained in 4.2, and incorporated in our pipeline before fusing them together. Specifically, the depth conditioning is extracted using the MiDaS model [24] (3.0 hybrid version), the semantic segmentation conditioning is obtained using a Segformer model [34] (B0 version), the sketch conditioning is automatically extracted using a PidiNet model (Pixel Difference Network) [29] (Base version). The color conditioning is obtained without the need of external algorithms, by a two-step resizing process, where the original image is first downscaled using cubic interpolation to a size 64 times smaller, and then upscaled back to the original resolution using nearest-neighbor interpolation.
However, included in the SHIFT dataset, are pre-computed depth and semantic segmentation conditionings for each image. These are ground truth conditionings, which ensure high accuracy and consistency. This means they have not been generated through any algorithms, and thus contain no errors. Some examples of ground truth conditionings versus their algorithm-generated equal are shown in Figure 25.
This reliance on pre-computed ground truth highlights a key limitation: if the model were to be deployed in production, it would not have access to such perfect annotations. Instead, it would need to rely on real-time algorithms, which may not achieve the same level of precision and could introduce discrepancies between training and deployment environments. As detailed in the following sections, both approaches were employed to train the models. This enabled us to evaluate the effectiveness of using ground truth conditionings in testing, in comparison to conditionings generated through an algorithm.

5.1.3 Hardware and Software Configuration

Experiments were run on two different servers with different settings. The first server is equipped with a dual Intel Xeon E5-2696 processor (48 cores, in total), 144GB of memory, and a Nvidia 3080ti GPU with 12GB of memory. The second server is equipped with an AMD 5950X processor (16 cores), 64 GB of memory, and a Nvidia 4090 GPU with 24GB of memory. Both servers were configured with Linux Ubuntu 22.04. The

[6] `https://huggingface.co/Salesforce/blip-image-captioning-base`
[7] `https://huggingface.co/llava-hf/llava-1.5-7b-hf`

software used in the experiments are: PyTorch 2.0.1[8], Pytorch lighting 1.6.0 [9], Huggingface diffursers 0.28.2 [10], Huggingface transformers 4.37.2 [11], CUDA 11.7 [12], Numpy 1.24.4 [13] and Pandas 2.0.3 [14].

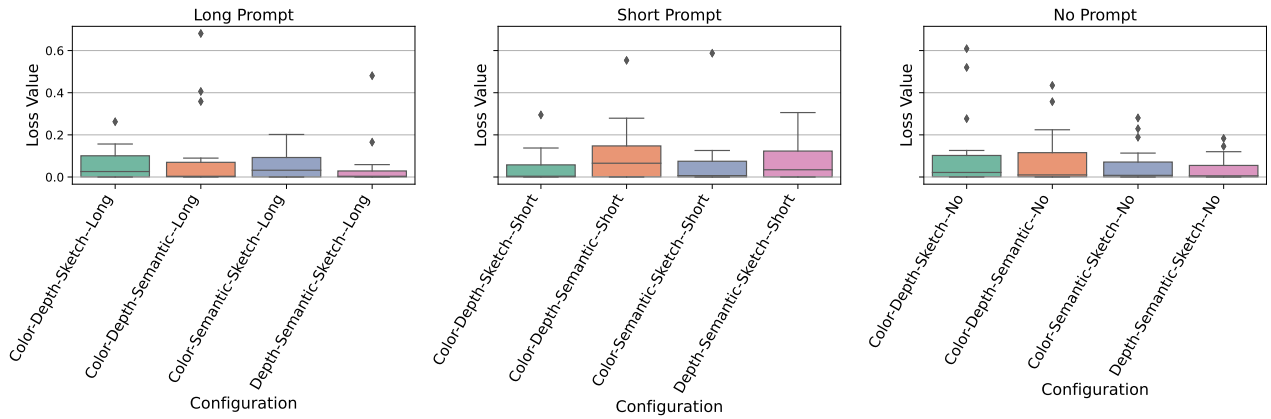
5.2. RQ1: Analysis of Mixing Conditionings

In this small-scale study, 12 different models were evaluated by combining four distinct conditionings: sketch, color, depth, and semantic segmentation. The models were tested across three types of prompts: long, short, and no prompt. Both short and long prompts were generated through LLAVA from a subset of SHIFT of 20 images. The conditionings were calculated by our model and not taken from the pre-computed conditionings of SHIFT.
The goal was to explore how different combinations of conditioning inputs and prompt lengths affected the quality of generated images and how well the model was trained for each combination. The list of the models are: Describe this combination. All possible combinations of conditionings multiplied by the different combinations of prompting. We will define the name codes for each model configuration used from here on out with a couple of examples. *Color-Depth-Semantic–Long*, means we used color conditioning, depth conditioning, and semantic conditioning with a long prompt. *Depth-Semantic-Sketch–No*, means we used depth conditioning, semantic conditioning, and sketch conditioning with no prompt.
The models were trained using a dataset of 20 images. These 20 images were cherry-picked out of the SHIFT dataset, to obtain a subset with different driving conditions able to summarize the wide range of settings from the whole dataset. The full 20 images are shown in Appendix A.
To ensure consistency in comparison, each model was allowed to train for up to 30 epochs, with batch sizes of one single image, where each epoch went over each of the images in the dataset.
In Figure 26a, we can see a comparison between the different models using long, short, and no prompt. The value in the plot is the average mean squared error (MSE) value of the original images versus the generated one. The MSE calculated between original image I and generated image I', with M and N being pixel height and width is:

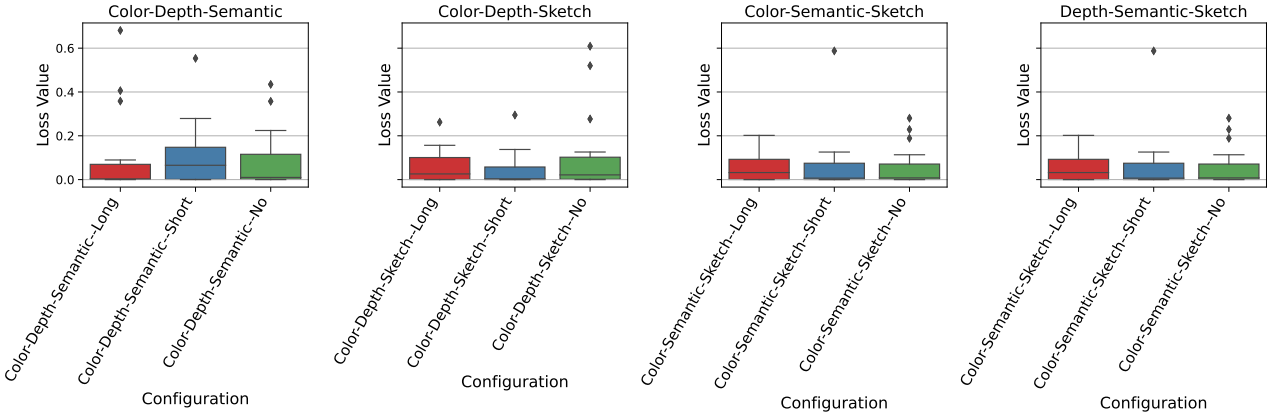
$$\text{MSE} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(I(i,j) - I'(i,j) \right)^2$$

Overall, all models obtain similar losses during training (no statistically significant difference). A t-test with $\alpha = 0.05$ was performed to determine if there were any statistically significant differences in the losses between each pair of models. The results of the test indicated that the differences were not statistically significant, suggesting that no model consistently outperformed the others in terms of training loss.
However, since training all 12 configurations is unfeasible due to computational resource constraints, we observed the quality of images generated during the final epochs. This allows for a more nuanced comparison of how quickly the models learn and how well they generalize. Additionally, we assessed the outputs during inference after training and validation. This helped us to understand each configuration's capabilities and the quality of their convergence, beyond what the loss metric can show.
Interestingly, when comparing each of the models being trained on the same conditionings with different prompts, the loss plots in Figure 26b show that using no prompt does not obtain worse results than using long or short prompts.
A validation dataset of 6 images was created to compare the generated images with the different conditionings and prompt types. It is important to understand that the desired output we want our models to have is that of the original image. Ideally, using conditionings generated directly from the original image should generate that same picture.
In Figure 27, some generations of the different configurations can be observed. When looking at these examples, we can see that the generations by Color-Depth-Semantic and Depth-Semantic-Sketch prove to be marginally superior to the other conditionings. Additional generations are present in Appendix B.
Based on these observations, it was decided that the Color-Depth-Semantic and Depth-Semantic-Sketch models, particularly when used without prompts, exhibited the most consistent performance across the initial validation stages. Their ability to produce good-quality images, combined with the clear improvements in generation between the first and last epochs, demonstrated their capacity for effective learning. Furthermore, the fact

[8] `https://pypi.org/project/torch/2.0.1/`
[9] `https://pypi.org/project/pytorch-lightning/1.6.0/`
[10] `https://pypi.org/project/diffusers/0.28.2/`
[11] `https://pypi.org/project/transformers/4.37.2/`
[12] `https://developer.nvidia.com/cuda-11-7-0-download-archive`
[13] `https://pypi.org/project/numpy/1.24.4/`
[14] `https://pypi.org/project/pandas/2.0.3/`



(a) Loss by type of prompt.



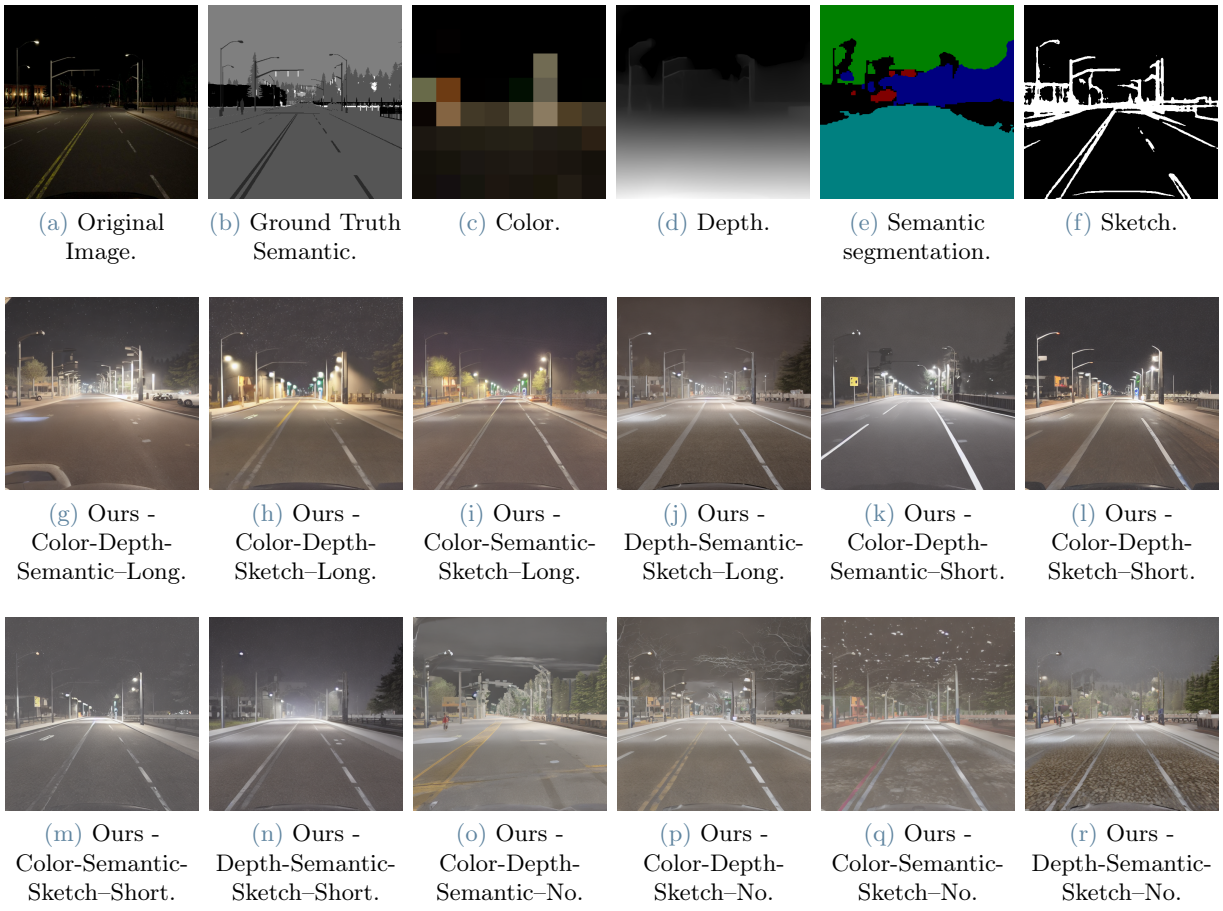
(b) Loss by type of conditioning group.

Figure 26: Losses for different model configurations.

that no significant degradation in performance was observed when no prompt was used, coupled with the impracticality of exhaustively training all models on the entire SHIFT dataset, led to the conclusion that these two configurations offered the best balance between quality and performance. Given their promising results with limited training, both models were chosen for full-scale training on the entire SHIFT dataset, with an extended number of epochs, to explore their potential in greater depth.

5.3. RQ2: Validity and realism.

In this section we present the two selected models from the previous small-scale study: *Color-Depth-Semantic–No* and *Depth-Semantic-Sketch–No*, both of which showed consistent performance during initial testing. In addition to these, we included a ground truth option in the training process. Differently from the previous experiment, we trained the two configurations with more data (about 70000 images). Since SHIFT includes manually generated depth and semantic segmentation conditionings, the rationale we followed was to investigate whether using Color conditioning alongside ground-truth depth and semantic segmentation conditioning would outperform our algorithmic conditionings and directly compare algorithm-based conditionings with ground-truth generation. What we expected before going into these models' evaluation was that the model using ground truth conditionings would outperform our automatic-generating conditioning models. The name we use for the model using Color, depth ground truth and semantic segmentation ground truth will be named *Color-DepthGT-SemanticGT–No*, as for the rest of the models.
The specifics of the training for each model are as follows: all *Color-Depth-Semantic–No*, *Depth-Semantic-Sketch–No* and *Color-DepthGT-SemanticGT–No* models were trained for a total of 6 hours each. We used a batch size of 4 images per batch across all models, ensuring consistency during the training process. The learning rate was set to 10^{-4}, as it proved to be the optimal configuration among a range of tested rates, spanning from 10^{-2} to 10^{-8}. The same training settings were applied to the model using ground-truth conditionings, which provided a controlled environment for evaluating the impact of systematically generated conditionings versus ground-truth conditioning on model performance. Additionally, checkpoints were saved every 20 minutes during



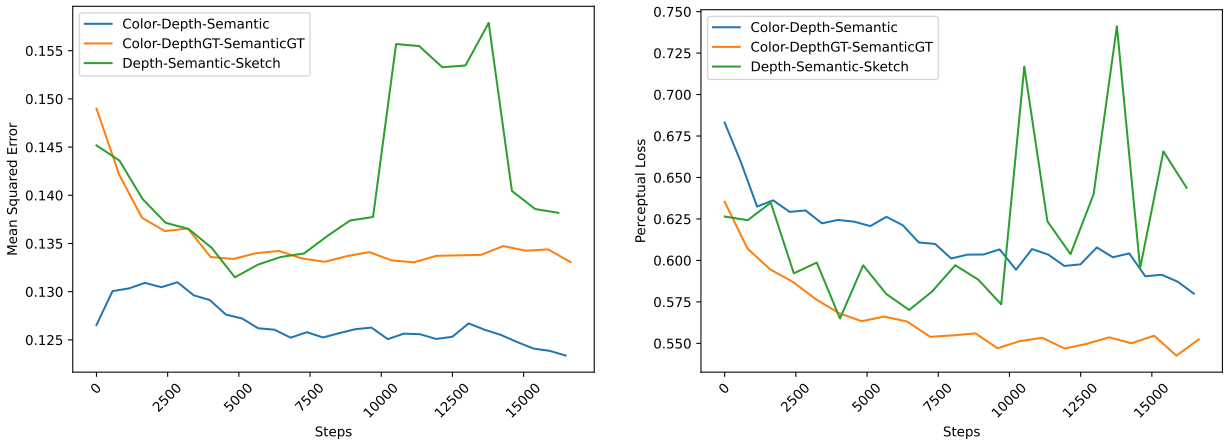
(a) Original Image. (b) Ground Truth Semantic. (c) Color. (d) Depth. (e) Semantic segmentation. (f) Sketch.

(g) Ours - Color-Depth-Semantic–Long. (h) Ours - Color-Depth-Sketch–Long. (i) Ours - Color-Semantic-Sketch–Long. (j) Ours - Depth-Semantic-Sketch–Long. (k) Ours - Color-Depth-Semantic–Short. (l) Ours - Color-Depth-Sketch–Short.

(m) Ours - Color-Semantic-Sketch–Short. (n) Ours - Depth-Semantic-Sketch–Short. (o) Ours - Color-Depth-Semantic–No. (p) Ours - Color-Depth-Sketch–No. (q) Ours - Color-Semantic-Sketch–No. (r) Ours - Depth-Semantic-Sketch–No.

Figure 27: Example of generations by the 12 different configurations.

training, allowing us to compare the evolution of each model's performance over time if needed.
A subset of the dataset of 20 images was used during the generation of images to test the validity and performance of the models. These validation images were selected to provide a representative sample of various driving conditions from the SHIFT dataset, ensuring that the models were evaluated across diverse scenarios, as done in the small study. All the images are shown in Appendix C. Furthermore, to ensure the validity of the validation process, auxiliary conditionings for all models were generated automatically, without the use of ground truth data. This approach reflects real-world scenarios where ground truth data, like that in the SHIFT dataset, would not be available during deployment.
In Figure 28, we present the validation loss per epoch plot for each of the three models, illustrating their training progress and convergence rates on two metrics, MSE and Learned Perceptual Image Patch Similarity [36] (LPIPS). LPIPS provides a perceptual similarity metric by comparing high-level feature representations of images, making it better to compare textures and structures that are important for human vision. By using both MSE and LPIPS, we gain insight not only into the pixel level but also how closely they resemble the target images from a perceptual point of view.
Each model's loss was tracked over time to assess the rate at which the models learned from the dataset and how effectively they minimized error during training, by using each of the checkpoints saved in the process. The loss was determined by computing the validation loss for each of these checkpoints, based on the comparison between the 20 original validation images and their corresponding generated counterparts.
The losses shown demonstrate the capability of the selected configurations to learn over time. The Color-Depth-Semantic model, in blue, shows a steady decrease in MSE and LPIPS as the number of epochs increases. This indicates that the model is improving in terms of minimizing the error for this conditioning combination, leading to more accurate imitations of the original image. The Color-DepthGT-SemanticGT model, in orange, has a stable MSE with a slight decline over epochs. It has a consistent performance, worse in terms of MSE compared to its non-ground-truth counterpart, but better in terms of perceptual loss. The Depth-Semantic-Sketch model, in green, exhibits some fluctuations with a notable peak around the halfway mark of its training. While there's a general decline, the sharp peak could be understood as a temporary instability in the model's training, perhaps due to overfitting.



(a) Mean Squared Error.

(b) Perceptual Loss.

Figure 28: Validation losses for Color-Depth-Semantic, Color-DepthGT-SemanticGT and Depth-Semantic-Sketch models.

The generation process involved adjusting the guidance scale, a key parameter that influences the trade-off between adherence to conditioning inputs and the overall quality or diversity of the generated images. The guidance scale effectively determines how much weight is given to the conditioning data relative to the model learned prior. Higher guidance scales push the model to produce outputs more closely aligned with the conditioning inputs, but can sometimes lead to less diverse or overly constrained outputs, while lower guidance scales allow for more variation but can result in images that deviate from the intended conditioning.
In this study, the guidance scale was tested at three different levels: 5, 7.5, and 10. These values were selected to explore how varying the strictness of the conditioning affects the image quality and relevance to the provided auxiliary inputs. A guidance scale of 5 allowed for more creative freedom in the generation process. On the other hand, a guidance scale of 10 enforced a strict following of the conditionings. The intermediate value of 7.5 aimed to balance between strictness and creativity, as it corresponds to the default value used by most Stable-Diffusion-based models.
Most image generations produced by the models at varying guidance scales are included in Appendix D, with a few selected examples shown in Figure 29 for the last checkpoints of each model. These examples provide a visual comparison of the outputs generated by each model, highlighting not only the differences in image quality but also the impact of the different guidance scales (5, 7.5, and 10) and conditioning inputs. The results demonstrate how varying the conditionings can influence the generated image. Specifically, one can see that by not using color conditioning, the output pictures result in a non-blue sky. Similarly, without the sketch conditioning, the windows in the building on the right are inaccurately sized.

5.4. RQ3: Comparison with State-of-the-Art

In this section, we evaluate our models—*Color-Depth-Semantic–No*, *Depth-Semantic-Sketch–No*, and *Color-DepthGT-SemanticGT–No*—against four state-of-the-art algorithms: Stable Diffusion, ControlNet, Stable Diffusion Image2Image, and T2I-Adapter. The first part presents a human experiment we carried out with users assessing the quality of our models' images compared with other SOTA models, while the second describes the results we obtained from the experiment in fine-grained detail.
To assess the performance of our models, we conducted a human experiment where participants were asked to evaluate the generated images based on four criteria: (1) whether the road in the generated image maintained the same shape as the original image, (2) whether the pedestrians were preserved during the transformation, (3) whether the cars were preserved in the transformation, and (4) the overall image quality, which was rated as "Good", "Ok", or "Bad". The first three criteria required yes or no responses, focusing on the preservation of key visual elements, while the fourth criterion assessed the subjective quality of the generated images. Five users were tasked with rating the validation dataset composed of 20 images for 66 models in total. We interviewed 5 participants. Each one evaluated 1320 images. In total, we received 6600 ratings.
Each combination of checkpoint, prompt type, and guiding scale was evaluated to understand how different configurations affected the quality and preservation of key elements in the generated images. For our models, Color-Depth-Semantic, Color-DepthSG-SemanticGT, and Depth-Semantic-Sketch, we used three different

(a) Original Image. (b) Ground Truth Semantic Segmentation. (c) Ours - Color-Depth-Semantic–No, 5 guidance level. (d) Ours - Color-Depth-Semantic–No, 7.5 guidance level. (e) Ours - Color-Depth-Semantic–No, 10 guidance level. (f) Ours - Color-DepthGT-SemanticGT–No, 5 guidance level.

(g) Ours - Color-DepthGT-SemanticGT–No, 7.5 guidance level. (h) Ours - Color-DepthGT-SemanticGT–No, 10 guidance level. (i) Ours - Depth-Semantic-Sketch–No, 5 guidance level. (j) Ours - Depth-Semantic-Sketch–No, 7.5 guidance level. (k) Ours - Depth-Semantic-Sketch–No, 10 guidance level.

Figure 29: Example of generations by our models.

	Image 1	**Image 2**	**Image 3**	**Image 4**	**Image 5**	**Image 6**	**Image 7**	**Image 8**	**Image 9**	**Image 10**
Pedestrian presence	No	Yes	No	Yes	Yes	No	No	Yes	No	No
Car presence	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes	No
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Pedestrian presence	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
Car presence	Yes	Yes	Yes	No	Yes	Yes	No	No	No	Yes

Table 1: Car and pedestrian presence for each image.

checkpoints obtained from our training combined with the three guidance scales defined before. For the models utilizing ControlNet, we experimented with three distinct prompt types, generated by LLAVA, by BLIP, and by a manual description made by us, to explore how varying prompts influenced the image generation process, and applied three different guiding scales, with a guidance level of 5, 7.5 and 10. Similarly, for the T2I-Adapter, we tested three prompt types, three guiding scales, and as auxiliary conditionings, used the two combinations employed by our models: Color-Depth-Semantic and Depth-Semantic-Sketch. The Stable Diffusion model was tested under the three prompt types, while the Stable Diffusion image-to-image model was evaluated with three prompt types and three guiding scales, ensuring consistent comparison with the other models. Examples of the other SOTA models are shown in Figure 30, while some other examples are shown in E.

To evaluate the generations of our models and the results of the human experiment, we first report Table 1 which indicates whether any pedestrian or car is present in the images. The table demonstrates a balanced distribution of the various combinations of car and pedestrian presence.

To assess the validity of the images, including road shape, pedestrian, and car preservation, we translate the answers from users of "Yes" and "No", to values 0 and 1. We then calculate the average value for each image for all configurations of the model. We show the results in Table 2, 3, 4, 5, 6, 7 and 8. Overall, the ControlNet

	Image 1	**Image 2**	**Image 3**	**Image 4**	**Image 5**	**Image 6**	**Image 7**	**Image 8**	**Image 9**	**Image 10**
Road validity	0.37	0.62	0.84	0.60	0.57	0.75	0.53	0.60	0.82	0.50
Pedestrian validity	-	0.52	-	0.26	0.42	-	-	0.51	-	-
Car validity	0.78	0.82	0.68	-	0.70	0.69	-	0.59	0.64	-
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Road validity	0.81	0.76	0.74	0.64	0.73	0.60	0.72	0.65	0.63	0.63
Pedestrian validity	-	0.36	0.45	0.53	0.74	0.57	0.30	-	-	-
Car validity	0.70	0.44	0.68	-	0.67	0.51	-	-	-	0.59

Table 2: Color-Depth-Semantic generations validity.

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
Road validity	0.42	0.67	0.86	0.64	0.62	0.79	0.57	0.66	0.85	0.53
Pedestrian validity	-	0.56	-	0.36	0.52	-	-	0.56	-	-
Car validity	0.81	0.85	0.71	-	0.75	0.70	-	0.61	0.70	-
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Road validity	0.84	0.80	0.78	0.67	0.78	0.66	0.76	0.70	0.68	0.69
Pedestrian validity	-	0.46	0.50	0.57	0.78	0.64	0.37	-	-	-
Car validity	0.75	0.51	0.70	-	0.71	0.49	-	-	-	0.58

Table 3: Color-DepthGT-SemanticGT generations validity.

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
Road validity	0.49	0.72	0.88	0.70	0.67	0.82	0.62	0.71	0.87	0.59
Pedestrian validity	-	0.62	-	0.45	0.59	-	-	0.61	-	-
Car validity	0.84	0.87	0.75	-	0.78	0.74	-	0.64	0.74	-
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Road validity	0.87	0.83	0.81	0.72	0.81	0.70	0.80	0.73	0.73	0.74
Pedestrian validity	-	0.53	0.57	0.60	0.81	0.69	0.39	-	-	-
Car validity	0.78	0.56	0.74	-	0.74	0.50	-	-	-	0.58

Table 4: Depth-Semantic-Sketch generations validity.

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
Road validity	0.49	1.00	1.00	0.93	0.93	1.00	0.71	0.87	1.00	0.89
Pedestrian validity	-	0.76	-	0.29	0.62	-	-	0.87	-	-
Car validity	1.00	1.00	1.00	-	1.00	1.00	-	0.98	1.00	-
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Road validity	1.00	0.98	1.00	1.00	1.00	1.00	0.98	0.64	0.93	0.96
Pedestrian validity	-	0.40	0.58	0.78	1.00	1.00	0.6	-	-	-
Car validity	1.00	0.51	0.87	-	0.93	0.91	-	-	-	0.87

Table 5: ControlNet generations validity.

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
Road validity	0.38	0.6	0.77	0.61	0.55	0.73	0.49	0.57	0.77	0.42
Pedestrian validity	-	0.49	-	0.15	0.37	-	-	0.57	-	-
Car validity	0.80	0.84	0.73	-	0.69	0.70	-	0.63	0.67	-
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Road validity	0.73	0.70	0.69	0.67	0.71	0.59	0.67	0.59	0.58	0.65
Pedestrian validity	-	0.28	0.43	0.49	0.71	0.59	0.32	-	-	-
Car validity	0.71	0.37	0.67	-	0.60	0.58	-	-	-	0.62

Table 6: T2I-Adapter generations validity.

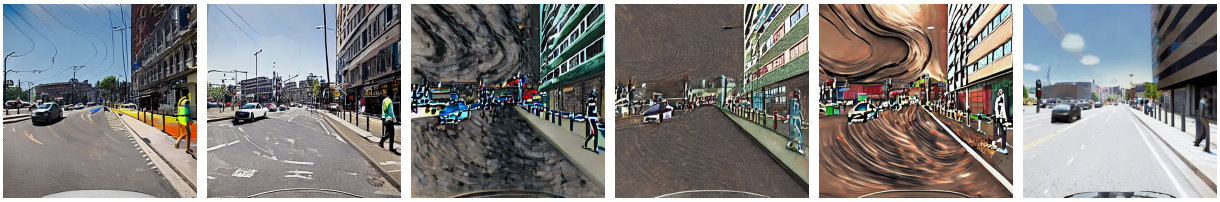
	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
Road validity	0.34	0.57	0.67	0.57	0.47	0.60	0.42	0.47	0.67	0.44
Pedestrian validity	-	0.48	-	0.14	0.31	-	-	0.5	-	-
Car validity	0.70	0.77	0.60	-	0.53	0.57	-	0.62	0.50	-
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Road validity	0.60	0.59	0.53	0.50	0.60	0.53	0.66	0.42	0.47	0.48
Pedestrian validity	-	0.20	0.29	0.46	0.60	0.50	0.40	-	-	-
Car validity	0.57	0.29	0.60	-	0.50	0.52	-	-	-	0.57

Table 7: Stable Diffusion generations validity.

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
Road validity	0.39	0.55	0.80	0.57	0.48	0.69	0.47	0.53	0.78	0.38
Pedestrian validity	-	0.42	-	0.11	0.30	-	-	0.47	-	-
Car validity	0.72	0.78	0.61	-	0.62	0.61	-	0.52	0.56	-
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Road validity	0.77	0.70	0.67	0.58	0.68	0.52	0.67	0.58	0.54	0.58
Pedestrian validity	-	0.22	0.36	0.49	0.68	0.46	0.28	-	-	-
Car validity	0.62	0.35	0.60	-	0.60	0.54	-	-	-	0.55

Table 8: Stable Diffusion Image-to-image generations validity.

(a) Original Image. (b) Ground Truth Semantic Segmentation. (c) ControlNet, 5 guidance level. (d) ControlNet, 7.5 guidance level. (e) ControlNet, 10 guidance level. (f) T2I-Adapter (Color-Depth-Semantic), 5 guidance level.



(g) T2I-Adapter (Color-Depth-Semantic), 7.5 guidance level. (h) T2I-Adapter (Color-Depth-Semantic), 10 guidance level. (i) T2I-Adapter (Depth-Semantic-Sketch), 5 guidance level. (j) T2I-Adapter (Depth-Semantic-Sketch), 7.5 guidance level. (k) T2I-Adapter (Depth-Semantic-Sketch), 10 guidance level. (l) Stable Diffusion Image-to-Image, 5 guidance scale.

(m) Stable Diffusion Image-to-Image, 7.5 guidance scale. (n) Stable Diffusion Image-to-Image, 10 guidance scale. (o) Stable Diffusion.

Figure 30: Example of generations by ControlNet, T2I-Adapter, Stable Diffusion and Stable Diffusion Image-to-Image with LLAVA prompt "The scene is set in a city with a busy street filled with cars. There are multiple traffic lights and signs, indicating a well-regulated traffic system. The street is surrounded by tall buildings, giving the impression of a bustling urban environment. The weather appears to be sunny, making it a pleasant day for driving".

and Depth-Semantic-Sketch models obtain the best image validity among the 7 compared models with averages of $0.92, 0.69$ and 0.93, and $0.74, 0.60$, and 0.71. As worse validation, the model Stable Diffusion obtained the worst results with averages of $0.53, 0.39$, and 0.56, followed by Stable Diffusion Image-to-image with scores of $0.59, 0.38$, and 0.59. Moreover, all models had a good road shape preservation score, well over 0.5, while pedestrian preservation was the most difficult characteristic to maintain.

The quality of images was also rated, ranging from -1 to 1. In Table 9 we present the the averages of quality for each model concerning each of the validation images. Stable Diffusion is the model that obtains the best overall quality of images with a score of 0.43, followed by our three models Depth-Semantic-Sketch, Color-DepthGT-SemanticGT, and Color-Depth-Semantic with scores $0.33, 0.29$, and 0.27, respectively. On the other hand, T2I-Adapter obtained the worst quality score with 0.13, followed closely by ControlNet with 0.21. Additionally, our models consistently rank within the top three for image quality, ranking first in 3, and are never among the lowest-rated for any images. Achieving higher quality than Stable Diffusion while incorporating additional conditionings is challenging due to the complexity introduced by multiple input modalities. These conditionings can sometimes limit the model's creative freedom, making it harder to match the high visual fidelity of Stable Diffusion. Furthermore, balancing various conditionings while maintaining image quality adds additional difficulty, as it requires the model to process more information without degrading the overall quality. While Stable Diffusion achieved the highest scores for overall image quality, our models Color-Depth-Semantic, Color-DepthGT-SemanticGT, and Depth-Semantic-Sketch demonstrated notable advantages, particularly in balancing validity and quality. The Depth-Semantic-Sketch model, in particular, achieved the second-best

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
Color-Depth-Semantic	0.33	0.4	0.34	0.43	0.1	0.34	0.04	0.23	-0.02	0.04
Color-DepthGT-SemanticGT	0.32	**0.43**	0.32	0.41	0.15	0.37	0.09	0.26	0.04	0.08
Depth-Semantic-Sketch	0.32	**0.43**	0.32	0.42	0.2	0.38	**0.14**	**0.29**	0.1	0.16
ControlNet	-0.07	0.4	0.11	0.44	-0.07	0.2	-0.29	-0.51	-0.22	**0.33**
T2I-Adapter	0.34	0.24	0.29	0.39	-0.1	0.23	-0.01	0.03	-0.28	-0.17
Stable Diffusion	**0.37**	**0.43**	**0.46**	**0.56**	**0.23**	**0.47**	-0.08	0.21	-0.01	0.07
Stable Diffusion I2I	0.33	0.38	0.38	0.42	0.06	0.3	0.01	0.18	-0.12	-0.08
	Image 11	**Image 12**	**Image 13**	**Image 14**	**Image 15**	**Image 16**	**Image 17**	**Image 18**	**Image 19**	**Image 20**
Color-Depth-Semantic	0.33	0.34	0.29	0.36	0.37	0.43	0.45	0.06	0.36	0.25
Color-DepthGT-SemanticGT	0.36	0.37	0.28	0.4	0.4	0.44	0.45	0.11	0.31	0.29
Depth-Semantic-Sketch	0.39	0.39	0.33	0.44	0.43	0.44	0.46	0.17	0.38	0.34
ControlNet	0.4	**0.73**	0.04	0.58	0.33	0.64	0.87	-0.33	0.56	-0.04
T2I-Adapter	0.19	0.19	0.09	0.28	0.11	0.31	0.39	-0.1	0.21	0.02
Stable Diffusion	**0.63**	0.63	**0.39**	**0.69**	**0.6**	**0.79**	**0.93**	**0.23**	**0.78**	**0.38**
Stable Diffusion I2I	0.27	0.29	0.2	0.29	0.29	0.41	0.44	-0.03	0.27	0.17

Table 9: Realism (user preference) of each model.

validity and second-best quality, making it highly reliable for preserving critical elements in road images as road shapes, pedestrians, and cars, while maintaining good image quality. Similarly, the Color-DepthGT-SemanticGT and Color-Depth-Semantic models offer robust performance with good validity and quality scores.
Furthermore, the results show that training the model with ground-truth conditionings did not offer a significant advantage over models trained with non-ground-truth conditionings in terms of validation. This suggests that the models can effectively be trained using automatically generated conditionings, eliminating the need for manually crafted conditionings.
Despite some state-of-the-art models performing better in terms of either image quality or validity, it is evident that excelling in one often came at the cost of the other. For example, models like Stable Diffusion achieved high-quality ratings but struggled significantly with image validity. Again, ControlNet performed well in terms of validity but lagged in visual quality. This trade-off highlights the difficulty of optimizing both criteria simultaneously. However, achieving a balance between validity and quality is crucial, particularly in applications where both the structural accuracy of generated images and their visual appeal are important, such as metamorphic testing. Our models, although not outperforming all SOTA models in both quality and validity, demonstrated a more balanced performance, making them more versatile and reliable across various tasks where both high validity and quality are needed.

6. Conclusion and Future Work

In this thesis we propose a novel image generation solution that integrates multiple conditionings, from sketch, color, depth, and semantic segmentation conditionings, to enhance the quality and realism of generated images for testing autonomous driving systems. Our approach aims to address key challenges in image generation by exploring various configurations of conditionings and prompt lengths, and by comparing our models against state-of-the-art methods, including ControlNet, T2I-Adapter, and Stable Diffusion.
Current solutions do not provide autonomous driving testers with sufficient control over the generation process. Having control of the generated images is crucial because it allows testers to leverage data that is either unexplored or with a small presence on current datasets while ensuring some properties. Additional conditionings to current models prove to be a valuable contribution, as it enable a targeted and effective exploration of unseen domains.
Our solution leverages a combination of different conditioning inputs to guide the image generation process, leveraging the T2I-Adapter architecture. We implemented and trained several models with different configurations, including a mix of color, depth, semantic segmentation, and sketch conditionings. This approach allowed us to assess the impact of each conditioning type and prompt length on the quality of generated images.
Through a comprehensive evaluation, including both small-scale studies and full-scale training, we addressed three main research questions. Firstly, our analysis revealed that models using a combination of color, depth, sketch, and semantic conditionings, with and without prompts, demonstrated robust performance in terms of image quality and training stability. Specifically, the Color-Depth-Semantic and Depth-Semantic-Sketch models showed promising generations and were selected for further evaluation.
Secondly, in assessing validity and realism, we compared models trained with algorithmically generated conditionings to those using ground-truth data. Our findings indicate that while ground-truth conditionings offered some improvements in perceptual similarity, the automatically generated conditionings were competitive and effective, reducing the need for manual data preparation.
Lastly, our models were compared against state-of-the-art techniques using human evaluations and quantitative metrics such as MSE and LPIPS. Our models, particularly Depth-Semantic-Sketch and Color-DepthGT-

SemanticGT, demonstrated a balanced performance, achieving high validity and competitive image quality. While some state-of-the-art models excelled in either quality or validity, ours achieved a notable equilibrium between the two, demonstrating its suitability for applications that require both accurate and quality pictures. As future work, we plan to experiment with additional conditionings that were not explored in this thesis, such as OpenPose or lighting conditionings, to further enhance the image generation process. Specifically, by integrating OpenPose conditioning, we could aim to enhance pedestrian preservation, which has proven to be the most challenging aspect to maintain from original images.
Furthermore, testing the generated images by including them in the SHIFT and training an autonomous driving algorithm with this extended dataset could help assess the robustness and generalization of our models, and determine if they have successfully achieved the objective of a metamorphic testing experiment.

References

[1] Hanan Almukhalfi, Ayman Noor, and Talal H. Noor. Traffic management approaches using machine learning and deep learning techniques: A survey. *Eng. Appl. Artif. Intell.*, 133:108147, 2024.

[2] Earl T. Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. The oracle problem in software testing: A survey. *IEEE Trans. Software Eng.*, 41(5):507–525, 2015.

[3] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 18392–18402. IEEE, 2023.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[5] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López, and Vladlen Koltun. CARLA: an open urban driving simulator. In *Proceedings of the Annual Conference on Robot Learning*, volume 78. PMLR, 2017.

[6] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Gläser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Trans. Intell. Transp. Syst.*, 22(3):1341–1360, 2021.

[7] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[8] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. *CoRR*, abs/1406.2661, 2014.

[10] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[11] Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[13] Timo Hynninen, Jussi Kasurinen, Antti Knutas, and Ossi Taipale. Software testing: Survey of the industry practices. In Karolj Skala, Marko Koricic, Tihana Galinac Grbac, Marina Cicin-Sain, Vlado Sruk, Slobodan Ribaric, Stjepan Gros, Boris Vrdoljak, Mladen Mauher, Edvard Tijan, Predrag Pale, and Matej Janjic, editors, *41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018, Opatija, Croatia, May 21-25, 2018*, pages 1449–1454. IEEE, 2018.

[14] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[16] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 12888–12900. PMLR, 2022.

[17] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[18] Qi Liu, Xueyuan Li, Shihua Yuan, and Zirui Li. Decision-making technology for autonomous vehicles: Learning-based methods, applications and future outlook. In *24th IEEE International Intelligent Transportation Systems Conference, ITSC 2021, Indianapolis, IN, USA, September 19-22, 2021*, pages 30–37. IEEE, 2021.

[19] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[20] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T. Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings Bioinform.*, 19(6):1236–1246, 2018.

[21] Chong Mou, Xintao Wang, Liangbin Xie, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *CoRR*, abs/2302.08453, 2023.

[22] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021.

[24] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(3):1623–1637, 2022.

[25] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10674–10685. IEEE, 2022.

[26] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. LAION-400M: open dataset of clip-filtered 400 million image-text pairs. *CoRR*, abs/2111.02114, 2021.

[27] Sergio Segura, Gordon Fraser, Ana Belén Sánchez, and Antonio Ruiz Cortés. A survey on metamorphic testing. *IEEE Trans. Software Eng.*, 42(9):805–824, 2016.

[28] Matteo Stefanini, Marcella Cornia, Lorenzo Baraldi, Silvia Cascianelli, Giuseppe Fiameni, and Rita Cucchiara. From show to tell: A survey on deep learning-based image captioning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(1):539–559, 2023.

[29] Zhuo Su, Wenzhe Liu, Zitong Yu, Dewen Hu, Qing Liao, Qi Tian, Matti Pietikäinen, and Li Liu. Pixel difference networks for efficient edge detection. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 5097–5107. IEEE, 2021.

[30] Tao Sun, Mattia Segù, Janis Postels, Yuxuan Wang, Luc Van Gool, Bernt Schiele, Federico Tombari, and Fisher Yu. SHIFT: A synthetic driving dataset for continuous multi-task domain adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 21339–21350. IEEE, 2022.

[31] Udacity. A self-driving car simulator built with Unity. `https://github.com/udacity/self-driving-car-sim`, 2017. Online; accessed 25 October 2023.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[33] Li-Hua Wen and Kang-Hyun Jo. Deep learning-based perception systems for autonomous driving: A comprehensive survey. *Neurocomputing*, 489:255–270, 2022.

[34] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, José M. Álvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 12077–12090, 2021.

[35] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 3813–3824. IEEE, 2023.

[36] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 586–595. Computer Vision Foundation / IEEE Computer Society, 2018.

[37] Jingyuan Zhao, Wenyi Zhao, Bo Deng, Zhenghong Wang, Feng Zhang, Wenxiang Zheng, Wanke Cao, Jinrui Nan, Yubo Lian, and Andrew F. Burke. Autonomous driving system: A comprehensive survey. *Expert Syst. Appl.*, 242:122836, 2024.

[38] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proc. IEEE*, 109(1):43–76, 2021.

[39] Liang Zong. Deep neural network based application capacity analysis in finance system. In *ICMLT 2021: 6th International Conference on Machine Learning Technologies, Jeju Island, Republic of Korea, April 23 - 25, 2021*, pages 122–126. ACM, 2021.

A. Appendix A

Images selected for training all configurations in the small scale study include a variety of scenes in weather (rainy, sunny, foggy), time of day (night, dawn, day), type of road (rural, city), vehicle presence, pedestrian presence, and traffic light presence.

(a) Image 1.

(b) Image 2.

(c) Image 3.

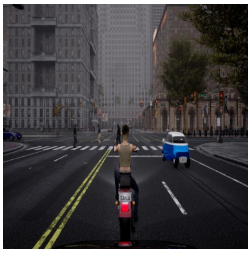
(d) Image 4.

(e) Image 5.

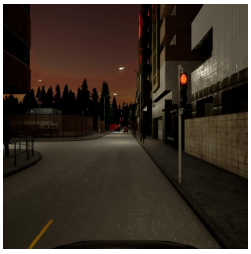
(f) Image 6.

(g) Image 7.

(h) Image 8.



(i) Image 9.



(j) Image 10.

(k) Image 11.

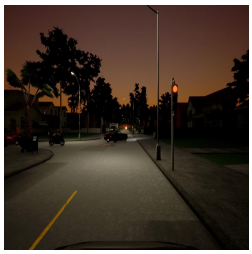
(l) Image 12.

(m) Image 13.

(n) Image 14.

(o) Image 15.

(p) Image 16.



(q) Image 17.

(r) Image 18.

(s) Image 19.

(t) Image 20.

B. Appendix B

Additional generations by different configurations in small-scale experiment.

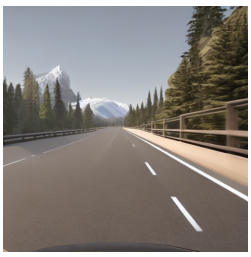
(a) Original.

(b) Color.

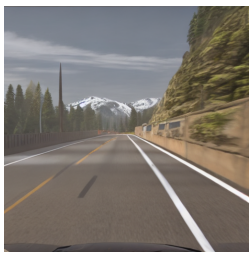
(c) Depth.

(d) Semantic.

(e) Sketch.



(f) Color-Depth-Sem–Long.



(g) Color-Depth-Sketch–Long.

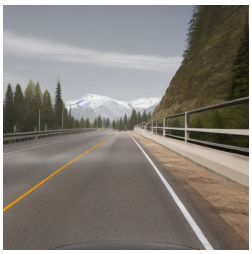
(h) Color-Sem-Sketch–Long.

(i) Depth-Sem-Sketch–Long.

(j) Color-Depth-Sem–Short.

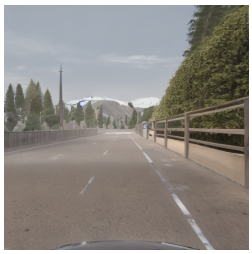
(k) Color-Depth-Sketch–Short.

(l) Color-Sem-Sketch–Short.

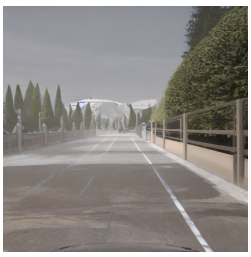


(m) Depth-Sem-Sketch–Short

(n) Color-Depth-Sem–No.



(o) Color-Depth-Sketch–No



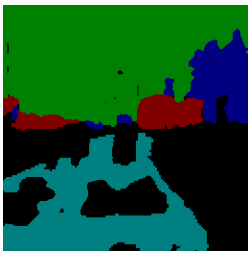
(p) Color-Sem-Sketch–No.

(q) Depth-Sem-Sketch–No.

(a) Original.

(b) Color.

(c) Depth.



(d) Semantic.

(e) Sketch.

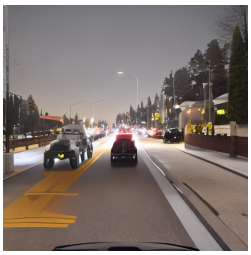
(f) Color-Depth-Sem–Long.

(g) Color-Depth-Sketch–Long.

(h) Color-Sem-Sketch–Long.

(i) Depth-Sem-Sketch–Long.

(j) Color-Depth-Sem–Short.



(k) Color-Depth-Sketch–Short.

(l) Color-Sem-Sketch–Short.

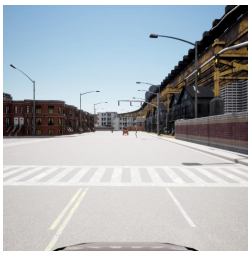
(m) Depth-Sem-Sketch–Short

(n) Color-Depth-Sem–No.

(o) Color-Depth-Sketch–No

(p) Color-Sem-Sketch–No.

(q) Depth-Sem-Sketch–No.



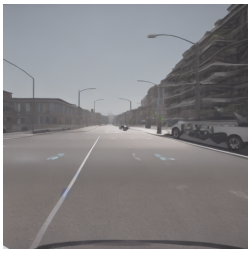
(a) Original.

(b) Color.

(c) Depth.

(d) Semantic.

(e) Sketch.



(f) Color-Depth-Sem–Long.

(g) Color-Depth-Sketch–Long.

(h) Color-Sem-Sketch–Long.

(i) Depth-Sem-Sketch–Long.

(j) Color-Depth-Sem–Short.

(k) Color-Depth-Sketch–Short.

(l) Color-Sem-Sketch–Short.

(m) Depth-Sem-Sketch–Short

(n) Color-Depth-Sem–No.

(o) Color-Depth-Sketch–No

(p) Color-Sem-Sketch–No.

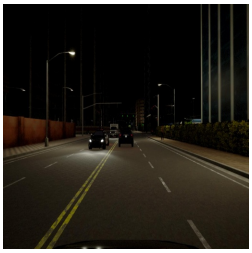
(q) Depth-Sem-Sketch–No.

C. Appendix C

Images selected for validation for evaluation. It includes a variety of situations, maximizing the representation of the possible combinations of the following characteristics: Type of environment, Weather conditions, pedestrian presence, Traffic lights, traffic signs, time of day, Road surface condition, Road markings, Vehicle presence.

(a) Image 1.

(b) Image 2.



(c) Image 3.

(d) Image 4.

(e) Image 5.

(f) Image 6.

(g) Image 7.

(h) Image 8.

(i) Image 9.

(j) Image 10.

(k) Image 11.

(l) Image 12.

(m) Image 13.

(n) Image 14.

(o) Image 15.

(p) Image 16.

(q) Image 17.

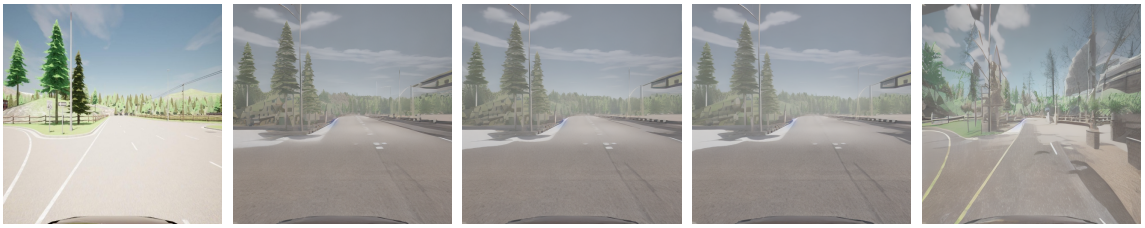
(r) Image 18.

(s) Image 19.

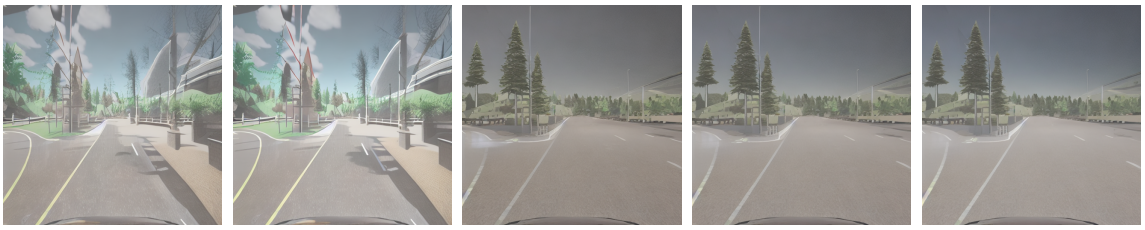
(t) Image 20.

D. Appendix D

Additional generations of Color-Depth-Semantic, Color-DepthGT-SemanticGT, and Depth-Semantic-Sketch. All blocks of 10 images correspond to: Original image; Color-Depth-Semantic–No, 5 guidance level; Color-Depth-Semantic–No, 7.5 guidance level; Color-Depth-Semantic–No, 10 guidance level; Color-DepthGT-SemanticGT–No, 5 guidance level; Color-DepthGT-SemanticGT–No, 7.5 guidance level; Color-DepthGT-SemanticGT–No, 10 guidance level; Depth-Semantic-Sketch–No, 5 guidance level; Depth-Semantic-Sketch–No, 7.5 guidance level; Depth-Semantic-Sketch–No, 10 guidance level.



(a) Original. (b) Color-Depth-Sem–No (5). (c) Color-Depth-Sem–No (7.5). (d) Color-Depth-Sem–No (10). (e) Color-Depth-Sem–No (5).



(f) Color-Depth-Sem–No (7.5). (g) Color-Depth-Sem–No (10). (h) Depth-Sem-Sketch–No (5). (i) Depth-Sem-Sketch–No (7.5). (j) Depth-Sem-Sketch–No (10).

(a) Original. (b) Color-Depth-Sem–No (5). (c) Color-Depth-Sem–No (7.5). (d) Color-Depth-Sem–No (10). (e) Color-Depth-Sem–No (5).

(f) Color-Depth-Sem–No (7.5). (g) Color-Depth-Sem–No (10). (h) Depth-Sem-Sketch–No (5). (i) Depth-Sem-Sketch–No (7.5). (j) Depth-Sem-Sketch–No (10).

(a) Original. (b) Color-Depth-Sem–No (5). (c) Color-Depth-Sem–No (7.5). (d) Color-Depth-Sem–No (10). (e) Color-Depth-Sem–No (5).

(f) Color-Depth-Sem–No (7.5). (g) Color-Depth-Sem–No (10). (h) Depth-Sem-Sketch–No (5). (i) Depth-Sem-Sketch–No (7.5). (j) Depth-Sem-Sketch–No (10).

(a) Original. (b) Color-Depth-Sem–No (5). (c) Color-Depth-Sem–No (7.5). (d) Color-Depth-Sem–No (10). (e) Color-Depth-Sem–No (5).

(f) Color-Depth-Sem–No (7.5). (g) Color-Depth-Sem–No (10). (h) Depth-Sem-Sketch–No (5). (i) Depth-Sem-Sketch–No (7.5). (j) Depth-Sem-Sketch–No (10).

E. Appendix E

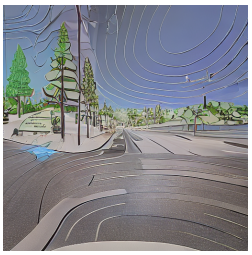
Additional generations of ControlNet, T2I-Adapter, SD and SDI2I. All blocks of 14 images correspond to: Original image; ControlNet Depth-Semantic-Sketch, 5 guidance level; ControlNet Depth-Semantic-Sketch, 7.5 guidance level; ControlNet Depth-Semantic-Sketch, 10 guidance level; T2I-Adapter Color-Depth-Semantic, guidance level 5; T2I-Adapter Color-Depth-Semantic, guidance level 7.5; T2I-Adapter Color-Depth-Semantic, guidance level 10; T2I-Adapter Depth-Semantic-Sketch, guidance level 5; T2I-Adapter Depth-Semantic-Sketch, guidance level 7.5; T2I-Adapter Depth-Semantic-Sketch, guidance level 10; SDI2I, guidance scale 5; SDI2I, guidance scale 7.5; SDI2I, guidance scale 10; SD.

(k) Original.

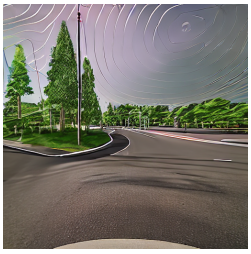
(l) ControlNet (5).

(m) ControlNet (7.5).

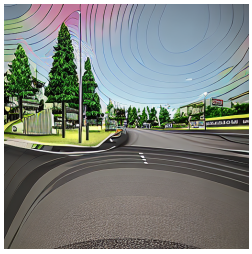
(n) ControlNet (10).



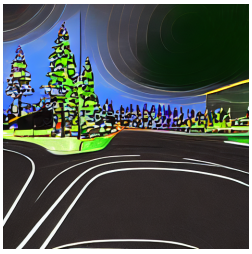
(o) T2I-Color-Depth-Sem (5).



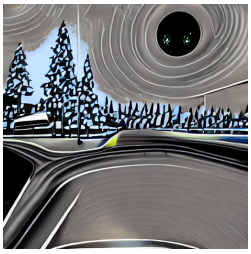
(p) T2I-Color-Depth-Sem (7.5).



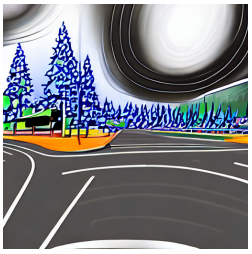
(q) T2I-Color-Depth-Sem (10).



(r) T2I-Depth-Sem-Sketch (5).



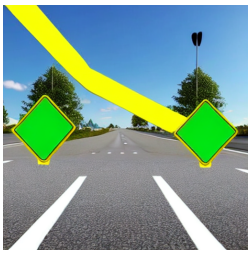
(s) T2I-Depth-Sem-Sketch (7.5).



(t) T2I-Depth-Sem-Sketch (10).

(u) SDI2I (5).

(v) SDI2I (7.5).



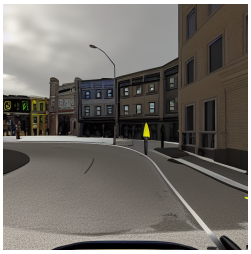
(w) SDI2I (10).

(x) SD.

(a) Original.

(b) ControlNet (5).

(c) ControlNet (7.5).

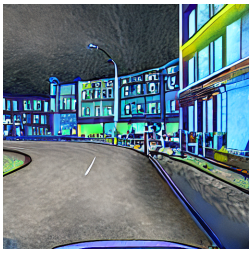


(d) ControlNet (10).

(e) T2I-Color-Depth-Sem (5).

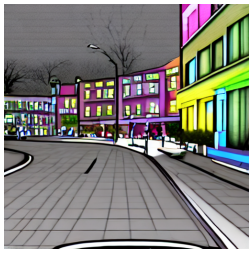
(f) T2I-Color-Depth-Sem (7.5).

(g) T2I-Color-Depth-Sem (10).



(h) T2I-Depth-Sem-Sketch (5).

(i) T2I-Depth-Sem-Sketch (7.5).



(j) T2I-Depth-Sem-Sketch (10).

(k) SDI2I (5).

(l) SDI2I (7.5).

(m) SDI2I (10).

(n) SD.

(a) Original.

(b) ControlNet (5).

(c) ControlNet (7.5).

(d) ControlNet (10).

(e) T2I-Color-Depth-Sem (5).

(f) T2I-Color-Depth-Sem (7.5).

(g) T2I-Color-Depth-Sem (10).

(h) T2I-Depth-Sem-Sketch (5).

(i) T2I-Depth-Sem-Sketch (7.5).

(j) T2I-Depth-Sem-Sketch (10).

(k) SDI2I (5).

(l) SDI2I (7.5).

(m) SDI2I (10).

(n) SD.

(a) Original.

(b) ControlNet (5).

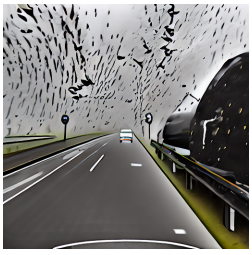
(c) ControlNet (7.5).

(d) ControlNet (10).

(e) T2I-Color-Depth-Sem (5).

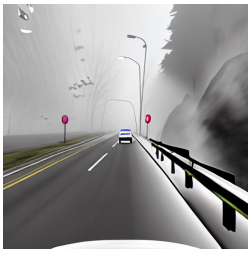
(f) T2I-Color-Depth-Sem (7.5).

(g) T2I-Color-Depth-Sem (10).



(h) T2I-Depth-Sem-Sketch (5).

(i) T2I-Depth-Sem-Sketch (7.5).



(j) T2I-Depth-Sem-Sketch (10).

(k) SDI2I (5).

(l) SDI2I (7.5).

(m) SDI2I (10).

(n) SD.

Abstract in lingua italiana

Negli ultimi anni, i modelli di Deep Learning (DL) hanno svolto un ruolo cruciale nello sviluppo dei sistemi di guida autonoma, eseguendo compiti essenziali come il rilevamento degli oggetti, la segmentazione semantica e la presa di decisioni. Tuttavia, testare questi modelli presenta sfide significative, in particolare in scenari reali, dove riprodurre specifiche condizioni di guida risulta costoso, pericoloso o impossibile. I metodi di test tradizionali mancano del controllo necessario per valutare sistematicamente i modelli di DL in una vasta gamma di condizioni, specialmente nei casi rari o estremi difficili da ricreare, come la guida in condizioni meteorologiche estreme.
Questa tesi affronta queste sfide proponendo un nuovo approccio che sfrutta i modelli di diffusione condizionale per la generazione controllata di immagini. Il metodo proposto estende il modello T2I-Adapter per supportare la multi-condizionamento, consentendogli di generare scenari di test basati su vari input, come i contorni delle immagini, i colori o le informazioni semantiche. Ciò permette un controllo preciso sulla generazione di scenari di guida realistici, migliorando significativamente la capacità di testare i modelli di DL per la guida autonoma in condizioni diverse.
Il modello è stato affinato sul dataset SHIFT, un dataset sintetico raccolto nell'ambiente di simulazione CARLA, che include una vasta gamma di condizioni meteorologiche, di traffico e ambienti di guida. La valutazione della soluzione proposta dimostra la sua efficacia nella generazione di casi di test validi e di alta qualità. Mostra inoltre un controllo migliorato sul processo di generazione rispetto ai modelli generativi all'avanguardia esistenti, come Stable Diffusion, ControlNet e i modelli a singolo condizionamento.
Inoltre, questa tesi contribuisce con un nuovo dataset ampliato, costruito sopra SHIFT, insieme a vari checkpoint del modello addestrato. Questi contributi forniscono un quadro completo per il progresso nel test dei sistemi di guida autonoma basati su ML, abilitando processi di valutazione più rigorosi e diversificati.

Parole chiave: Metamorphic Testing; Esempi Alternativi; Aumento dei Dati; Modelli di Diffusione; Multi-condizionamento; Guida Autonoma

Acknowledgements

I would like to express my gratitude to my advisor, Professor Luciano Baresi, and my co-advisor, Davide Yi Xian Hu, for their guidance throughout the extensive development of this thesis. I also wish to thank Politecnico di Milano for providing me with the knowledge necessary to undertake this project, as well as all my professors from the Mathematical Engineering program.