UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

AMADEO TATO COTA NETO

ENDLESS: An End-to-End Framework for Urban Synthetic Dataset Generation

Recife

2025

AMADEO TATO COTA NETO

ENDLESS: An End-to-End Framework for Urban Synthetic Dataset Generation

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de bacharel em Ciência da Computação.

Área de Concentração: Mídia e interação

Orientador (a): Francisco Paulo Magalhães Simões

Co-Orientador (a): Veronica Teichrieb

Recife

2025

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Cota Neto, Amadeo Tato .
ENDLESS: An End-to-End Framework for Urban Synthetic Dataset Generation / Amadeo Tato Cota Neto. - Recife, 2025.
44 p. : il., tab.

Orientador(a): Francisco Paulo Magalhães Simões
Cooorientador(a): Veronica Teichrieb
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Ciências da Computação - Bacharelado, 2025.
Inclui referências, apêndices.

1. Synthetic Data. 2. Smart Cities. 3. Computer Vision. I. Simões, Francisco Paulo Magalhães. (Orientação). II. Teichrieb, Veronica. (Coorientação). IV. Título.

000 CDD (22.ed.)

AMADEO TATO COTA NETO

ENDLESS: An End-to-End Framework for Urban Synthetic Dataset Generation

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de bacharel em Ciência da Computação .

Aprovado em: 04/04/2025

BANCA EXAMINADORA

_____

Prof. Dr. Francisco Paulo Magalhães Simões (Orientador)

Universidade Federal de Pernambuco

_____

Prof. Dr. Cleber Zanchettin (Examinador Interno)

Universidade Federal de Pernambuco

I dedicate this work to my grandfather Manuel, the purest person I've ever known and who, today, is in a better place.

ACKNOWLEDGEMENTS

Firstly, I thank God for giving me strength during graduation; without it, I would not have been able to finish this process.

I also would like to express my gratitude to my parents, who always stimulated me to study and worked hard to ensure quality education for me and my siblings. I also want to express my gratitude to them, my siblings, for all the fun moments we had together and for all the support during this process.

I also thank Veronica for the orientation in conducting this research and for the kindness and care with which you treat me.

A special thanks to my colleagues at Voxar Labs, who helped with my professional development and gave me valuable lessons on how to follow in my career. In special, I thank Will, for the support during the conducting of this research and for the supervision and patience in the R&D projects in which we worked together.

I'm also thankful for the friends I made during the graduation, especially for Bruno, Cauê, Eduardo, and Matheus. This journey became lighter thanks to you and I am grateful for every time we laugh together.

Last but not least, I am deeply grateful to all the professors who shared knowledge with me during the graduation and beforehand.

RESUMO

Modelos de visão computacional são fundamentais para aplicações de cidades inteligentes. Esses modelos permitem que a cidade interprete dados visuais, advindos de sensores como câmeras de segurança, para otimizar suas tarefas e impactar positivamente a vida dos cidadãos. Contudo, esses modelos requerem quantidades cada vez maiores de dados anotados para serem treinados, os quais são custosos e trazem questões éticas quando coletados no mundo real. Por outro lado, motores gráficos 3D e simuladores permitem uma geração barata e em larga escala de dados sintéticos automaticamente anotados. Este trabalho propõe um gerador de bases de dados sintéticos no contexto de cidades inteligentes usando o simulador CARLA. O gerador proposto permite a geração fim-a-fim de bases de dados massivas com um único comando, o que inclui a simulação de elementos de cidades, como veículos e pedestres, a coleta e a anotação de dados visuais. Para demonstrar a capacidade do gerador, uma base de dados com mais de 300 mil imagens anotadas foi gerada e comparada com outras bases do estado da arte. Resultados da comparação evidenciam que o gerador proposto é capaz de gerar bases equiparáveis ao estado da arte em número de dados e de anotações. Espera-se que nosso gerador possa ser usado para criar bases de dados úteis para o treino e validação de modelos de visão computacional no campo de cidades inteligentes. Além disso, espera-se também que esse trabalho traga atenção para o uso de dados sintéticos em modelos para cidades inteligentes.

Palavras-chaves: Dados Sintéticos. Cidades Inteligentes. Visão Computacional.

ABSTRACT

Computer vision models are fundamental for smart city applications. These models enable the city to interpret visual data, obtained from sensors such as surveillance cameras, to optimize its tasks and positively impact the citizens' lives. However, these models require ever-growing amounts of labeled data for training, which is expensive and raises ethical concerns when collected in the real world. Conversely, 3D engines and simulators allow the cheap and large-scale generation of automatically annotated synthetic data. This work proposes a synthetic dataset generator for the smart cities field using the CARLA simulator. The proposed generator allows the end-to-end generation of massive datasets with a single command, which includes the simulation of city assets, such as vehicles and pedestrians, and the recording and annotation of visual data. To prove the generator's competence, a dataset with over 300K annotated frames was generated and compared with others from the state-of-art. The comparison results show that the proposed generator is capable of producing datasets comparable to the state of the art in terms of data volume and number of annotations. It's expected that the proposed generator could be used to create useful datasets for training and evaluating computer vision models in the smart cities area. It's also expected that this work bring attention to the synthetic data usage for smart city models.

Keywords: Synthetic Data. Smart Cities. Computer Vision.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
CIn	*Centro de Informática*
CV	Computer Vision
CVAT	Computer Vision Annotation Tool
DL	Deep Learning
ENDLESS	ExpaNdable Datasets Labeled and Empowered by Synthetic Simulation
FPS	Frames Per Second
GPU	Graphics Processing Unit
ICT	Information and Communication Technologies
IMU	Inertial Measurement Unit
IoT	Internet of Things
UFPE	*Universidade Federal de Pernambuco*
V2X	Vehicle-to-Everything
YOLO	You Only Look Once

CONTENTS

1 INTRODUCTION

Urban spaces must be capable of ensuring the well-being of their citizens, guaranteeing them a pleasant quality of life and easy access to resources and services. However, the rapid urbanization of the world makes cities increasingly populated, bringing challenges to the management of resources and the assurance of a good quality of life for their citizens.

In light of these problems, smart cities integrate advanced technological solutions to improve the quality of life of their residents and to optimize their tasks. Technologies from the fields of Artificial Intelligence (AI), Internet of Things (IoT), and big data are commonly used in smart city applications to achieve this, allowing the city to understand its context and make informed decisions about it.(SILVA et al., 2018; ZAMAN et al., 2024)

In this context, Computer Vision (CV) models play a fundamental role in enabling smart cities to analyze and understand events occurring in their streets by processing visual data collected from sensors such as RGB cameras (SYAHIDI; KIYOKAWA; OKURA, 2023). These models can be applied in diverse contexts, such as traffic monitoring (BARTHÉLEMY et al., 2019), accident detection (ADEWOPO et al., 2023), and safety (YAR et al., 2023).

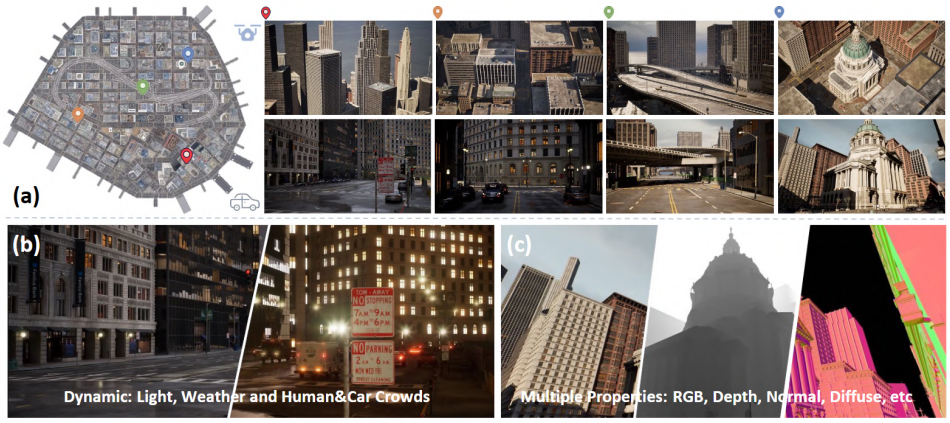
However, supervised deep learning models require an increasing amount of labeled data to be trained (DOSOVITSKIY et al., 2020). Traditionally, this data is obtained by collecting images and videos from the real world for a given period and annotating them afterward (GEIGER; LENZ; URTASUN, 2012; CORDTS et al., 2016). However, this approach is time-consuming (requiring more than 1.5h per image for fine annotations in (CORDTS et al., 2016)), expensive, and raises concerns regarding the privacy of the data and possible biases associated with the dataset.

On the other hand, synthetic datasets are collections of data generated in digital environments, such as game engines (GAIDON et al., 2016), specific software (HERZOG et al., 2023), or even computer games (RICHTER et al., 2016). These datasets simulate real-world physics, environments, and rules in the digital world, enabling the creation of data that closely resembles real-world datasets (GAIDON et al., 2016).

Compared to real-world datasets, synthetic datasets are a cheaper and faster alternative to gathering data to train CV models. These datasets can be generated with a massive amount of data, are automatically annotated, and offer better control of the dataset environment, allowing adjustments on the data diversity (PATHIRAJA; LIU; SENANAYAKE, 2024) and allowing the recording of rare real-world events (GAIDON et al., 2016).

Additionally, the constant improvements in computer graphics enhance the data quality of synthetic datasets. Modern game engines enable data to be recorded with photorealistic graphics and pixel-perfect annotations (LI et al., 2023; TURKCAN et al., 2024), resulting in high-quality dataset (see in Figure 1). Experiments described in (ROS et al., 2016) demonstrated that augmenting real-world datasets with synthetic data for training CV models can lead to improved results.

Figure 1 – Example of photorealistic urban synthetic dataset and it characteristics.



Source: Li et al. (2023).

However, while there are plenty of urban synthetic datasets available, they often are created from very specific viewpoints. For example, synthetic datasets for autonomous driving applications are often generated using cameras placed on top of vehicles or in the perspective of vehicles' hoods (ROS et al., 2016; GAIDON et al., 2016). Other datasets contain top-down camera views that are useful for example to drone tasks (TURKCAN et al., 2024), for example. Although useful, these datasets are not focused on the city as a whole, which is limiting to tasks in which the city should be an active agent that perceives its context and actuates on it.

Additionally, even though these synthetic datasets can be modified, their frameworks often do not provide an end-to-end approach, which makes the generation of new data difficult. We consider a dataset generation framework to be end-to-end if it encapsulates all steps of dataset generation in a single command.

In this work, we propose the ExpaNdable Datasets Labeled and Empowered by Synthetic Simulation (ENDLESS) framework, an end-to-end dataset generation framework developed on top of the CARLA Simulator (DOSOVITSKIY et al., 2017) that enables the generation of large urban synthetic datasets by a simple script execution after the initial CARLA setup. Users can

change various parameters of the generated dataset such as weather, city maps, crowd density, and traffic intensity, allowing the generation of customized datasets. The generated datasets are automatically annotated regarding 2D bounding-boxes, instance segmentation, and depth maps.

To prove the competence of our generator, we used it to generate a novel dataset with over 300K frames and compared it with state-of-the-art urban synthetic datasets. Our analysis shows that our framework can generate datasets with a comparable number of frame and annotation counts as the literature's state-of-the-art.

This work is structured as follows: first, in chapter 2, we provide the necessary background to support this work. Next, in chapter 3, we present related works of urban synthetic datasets and dataset generation tools. After that, we present our generator at chapter 4. Then, we introduce our generated dataset and analyze it at chapter 5. Furthermore, we compare our dataset to others from the state-of-the-art urban synthetic datasets in chapter 6. Finally, we present our conclusions and discuss limitations and future works in chapter 7. Additionally, preliminary experiments with CARLA's digital twin tool are presented in Appendix A.

2 BACKGROUND

The constant advances in the field of Deep Learning (DL) have led to the widespread adoption of these models in various tasks, making them present in multiple aspects of everyday life. In particular, DL models for Computer Vision (CV) can interpret the real world through images and videos, making them essential for smart city tasks (SYAHIDI; KIYOKAWA; OKURA, 2023).

These models typically use a supervised learning approach, in which the model learns from examples provided in a labeled dataset. To make this training possible, the dataset must be annotated according to the task the model is designed to perform. However, as the field advances, DL models are increasingly large and require ever-growing amounts of data to be trained (DOSOVITSKIY et al., 2020). Nonetheless, obtaining fine annotations of real-world datasets is a time-consuming task (CORDTS et al., 2016) and the recorded dataset may contain intrinsic biases related to the location and conditions in which it was recorded.

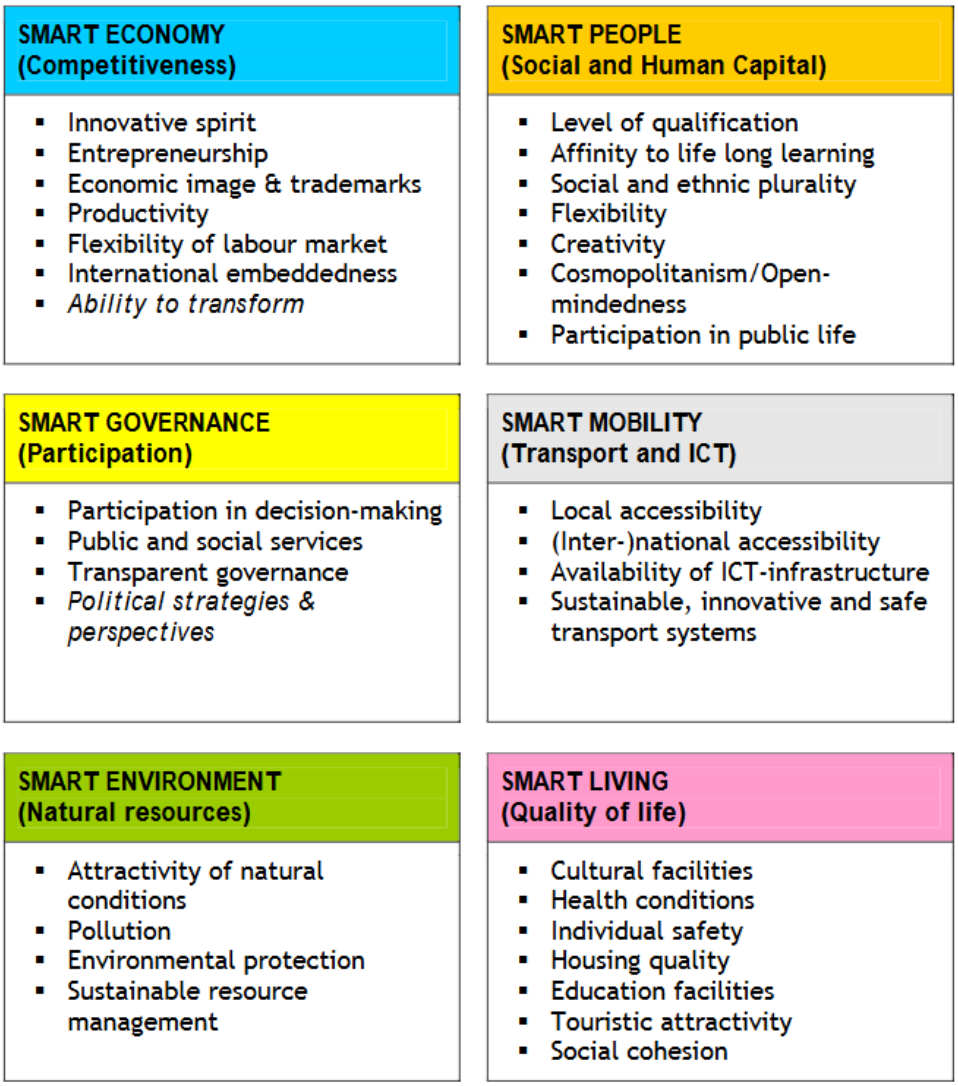
In this chapter, we provide the necessary background to support our work and to better understand its placement within the current literature. First, we provide an overview of the fields of smart cities and CV. Second, we present the types of annotations used in CV models. Finally, we present simulation engines used to generate synthetic datasets.

2.1 SMART CITIES

In an increasingly urbanized world, cities must be able to address challenges related to citizens' quality of life, service provision, air pollution, and other challenges arising from constant urbanization. The smart city concept emerges to describe a city that can tackle these challenges and thrive.

Although in increasing popularity, a unique definition of what constitutes a smart city is not universally accepted and many works provide different definitions of it (CHOURABI et al., 2012). Giffinger et al. (2007) provide a holistic definition of the term, involving various sectors of urban areas. According to them, a smart city should thrive in economy, people, governance, mobility, environment, and living; and their citizens must have active participation in the city. To better describe these characteristics, the authors provided 33 factors that provide a better understanding of them, as shown in Figure 2.

Figure 2 – Characteristics and factors of a smart city.



Source: Giffinger et al. (2007).

A concept also associated with smart cities is that these cities should be able to capture information about their context and interpret it to enable informed decision-making (ZAMAN et al., 2024; SILVA et al., 2018). In this regard, Information and Communication Technologies (ICT) solutions, such as Artificial Intelligence (AI), big data, and Internet of Things (IoT), are widely explored in research on the field (SILVA et al., 2018; ZAMAN et al., 2024).

IoT devices enable information exchange between city assets through the internet, creating a network where each asset contributes to improving its own performance and the city as a whole. These devices also provide real-time data from multiple regions and sectors of the city, providing a holistic overview of its context (ZAMAN et al., 2024). The high volume of data collected from all these devices can be processed by big data techniques, enabling the filtering

of redundant or erroneous data and normalizing it. The processed data can then be stored for future use and be analyzed to generate insights on how to improve different aspects of the city (SILVA et al., 2018). The raw or processed obtained device data can be used by AI models to enable the city to detect events and autonomously react to them through its devices (ADEWOPO et al., 2023; BARTHÉLEMY et al., 2019).

To illustrate how these technologies improve urban dynamics in smart cities, imagine the following situation: a traffic accident occurred on a busy avenue just before rush hour. A smart city should be able to automatically detect the accident, through surveillance cameras for example, and notify emergency services to assist those involved. At the same time, the city would work to prevent traffic jams by informing citizens of the incident and suggesting alternative routes to them. In a Vehicle-to-Everything (V2X) scenario, the city would also be able to communicate directly with autonomous vehicles so that they could recalculate their routes.

2.2 COMPUTER VISION

Computer vision is a field of AI that studies how computers can interpret visual inputs, such as images and videos, in an attempt to replicate the mechanisms of the human brain and vision. This field is rapidly expanding due to improvements on DL models, which enabled various CV tasks to be performed.

Classic algorithms in the field use image processing and statistics to extract features from images. These features could be used as input for traditional machine learning models to perform CV tasks. Today, DL models have become the new standard in the field, allowing automatic extraction of features from images and more accurate predictions.

Most DL models for CV are trained using supervised learning, meaning they generally require labeled data for training. Just as other AI models, labeled datasets for CV contain input data and ground-truth annotations for those inputs (GEIGER; LENZ; URTASUN, 2012).

Image classification may be the best-known task in the field. In this task, the model's objective is to predict the class of an image from a predefined set of classes. The LeNet model (LECUN et al., 1998) enabled handwritten digit recognition by classifying grayscale images of digits as one of ten possible classes. Today, with the advancements in the DL field, image classification models are capable of identifying the class of an image among thousands of categories for which the model has been trained (DOSOVITSKIY et al., 2020).

Object detection models also provide the class of objects from an image, but they also provide the location and dimension of these objects. The location of the objects is given by enclosing the object in bounding-boxes (REDMON et al., 2016), which will be better explained in Section 2.3.1.

Segmentation models take a step further by classifying each pixel in the image rather than just detecting objects, providing more detailed results (HE et al., 2017). These models require segmentation maps for training, which will be detailed in Section 2.3.2.

Depth estimation is another common task in CV, enabling the generation of depth maps, which estimate the distance of each pixel in an image from the camera (BHAT et al., 2023). More details on depth maps can be found in Section 2.3.3.

2.3 COMMON COMPUTER VISION ANNOTATIONS

As mentioned before, the training of CV models usually requires annotated data. The annotation process is usually slow and requires human annotators to manually obtain ground-truth data from images or the usage of specific hardware or software solutions. Nowadays, tools such as Roboflow[1] and Computer Vision Annotation Tool (CVAT)[2] speed up the annotation processes by using AI techniques, but still requires human annotators to generate finer annotations.

In this section, we present common annotations used to train CV computer vision models. We specifically focus on bounding-boxes, segmentation, and depth maps once these are the annotations we provide in our synthetic data generation framework.

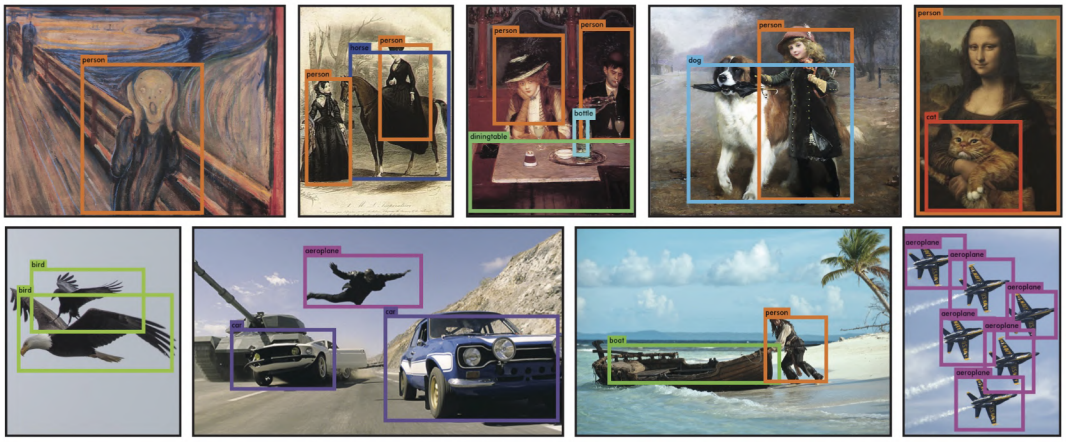
2.3.1 Bounding-Boxes

Bounding-Boxes are a type of annotation used to specify the position and dimensions of objects in images. This is done by enclosing the object within a box that contains it, as shown in Figure 3. This annotation is fundamental for training AI models in tasks that require the location or dimension of objects in the image, such as object detection (REDMON et al., 2016).

Just as there are various ways to represent squares, there are various ways to represent bounding-boxes of objects. The You Only Look Once (YOLO) models (REDMON et al., 2016)

[1] Available at: <https://roboflow.com/>. Accessed on: Mar. 18, 2025
[2] Available at: <https://www.cvat.ai/>. Accessed on: Mar. 18, 2025

Figure 3 – Example of 2D bounding boxes.



Source: Redmon et al. (2016).

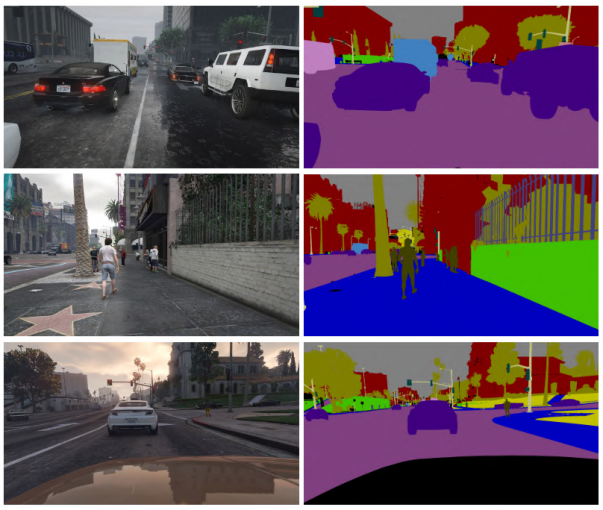
utilize bounding-boxes represented by the center of the box and its dimensions. Other works represent the bounding-boxes storing their top-left and bottom-right vertices coordinates such as the Faster-RCNN (REN et al., 2015).

Extending the conventional 2D bounding-boxes, 3D bounding-boxes contain additional depth information, using rectangles to enclose objects. The additional depth information makes 3D bounding-boxes useful for applications in the field of autonomous driving, 3D reconstruction, and extended realities. However, obtaining 3D bounding-boxes is more challenging as it requires depth data and a more complex annotation process.

2.3.2 Segmentation Maps

Segmentation maps are pixel-level annotations that encode information in the RGB values of each pixel of an image, making it rich in detail and precision. There are different types of segmentation maps, which differ in the information stored in the RGB values. For semantic segmentation maps, the RGB value stores the class of the object comprising the pixel. For instance, segmentation maps, both the object class and a unique ID for each object are encoded in the RGB values in the image. This is often achieved by storing the object class in one channel and the object ID in the remaining channels. Figure 4 provides examples of semantic segmentation maps.

Figure 4 – Example of semantic segmentation map.



Source: Richter et al. (2016).

2.3.3 Depth Maps

Depth maps store the distance from a given pixel to the camera, typically in meters, adding depth information to 2D images. Obtaining these maps required expensive sensors such as LiDAR, stereo cameras, or structured light systems, limiting the obtaining of this data.

However, improvements in DL and CV techniques democratized the obtaining of depth maps by enabling accurate depth estimation. Depth estition models such as the ZoeDepth (BHAT et al., 2023), can generate high-fidelity depth maps from monocular images captured by a conventional RGB camera (see in Figure 5), reducing the costs associated with specialized sensors.

Figure 5 – Example of depth maps estimated from deep learning model.

Source: Adapted from (BHAT et al., 2023).

2.4 SIMULATION ENGINES FOR SYNTHETIC DATASETS

Advancements in computer graphics and game development fields resulted in improved realism in digital environments, both in terms of graphical quality and physical accuracy. In this context, simulation engines emerge as computer graphics software that enables the simulation of real-world rules and environments in a digital setting, allowing experiments to be conducted in the virtual world. In particular, the fields of autonomous vehicles and smart cities have strong incentives to use these tools, as simulations make it possible to carry out experiments that would otherwise be costly or difficult to carry out in the real world (DOSOVITSKIY et al., 2017).

Various of these engines make it possible to obtain visual data through simulated sensors, which include RGB images, depth and segmentation maps, LiDAR scans, and others. The availability of these visual sensors combined with the realism of these engines enables their use to generate synthetic datasets that accurately replicate real-world dynamics, making them suitable for training CV models.

The BeamNG.tech[3] is a simulation engine aimed for applications on the autonomous vehicles and driver training fields. The engine has its own physical simulator, which uses a custom soft-body physics that helps to realistically simulate vehicle kinematics.

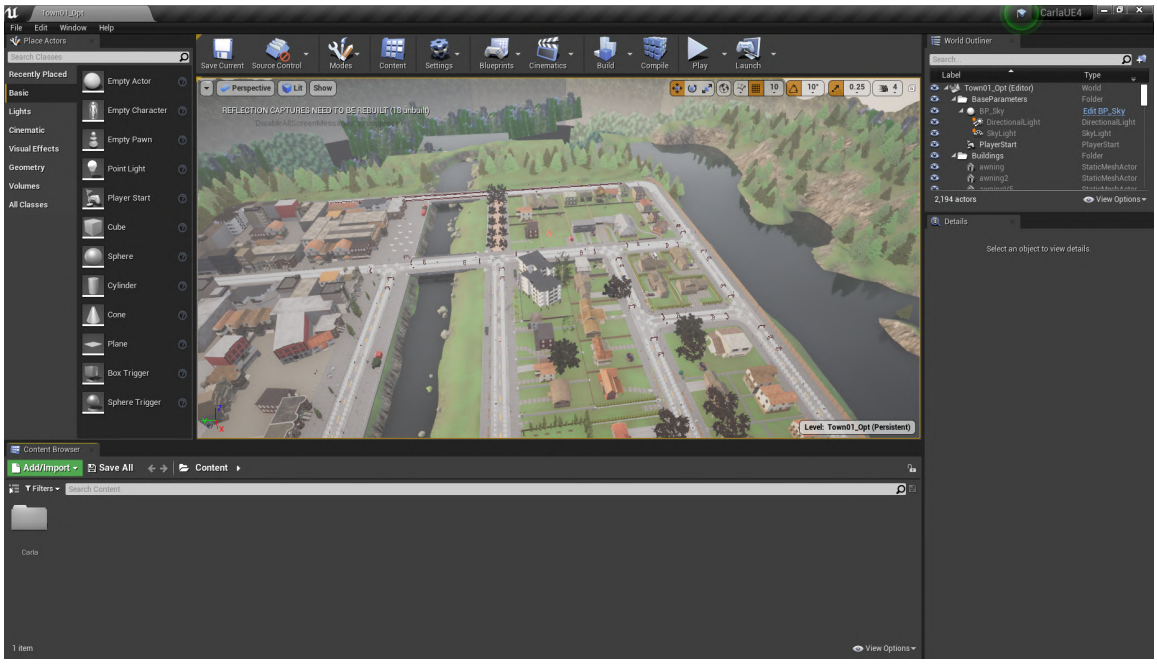
The engine provides a set of sensors for data acquisition through simulation, including cameras, LiDAR, Inertial Measurement Unit (IMU), and ultrasonic sensors. Additionally, it enables the acquisition of ground-truth data, such as bounding-boxes and segmentation. These features enable synthetic datasets to be recorded using the engine, exploiting the realistic vehicle kinematics.

One of the most notable simulation engines is the CARLA simulator (DOSOVITSKIY et al., 2017), built using a fork of the popular Unreal Engine[4] and focused on experiments with autonomous driving models. The simulator provides digital cities in which vehicles can roam and simulate traffic agents such as vehicles, pedestrians, traffic lights, and others. Figure 6 shows the CARLA simulation interface on Unreal Engine 4.

The simulator also contains a great sensor suite that was originally designed to simulate how an autonomous car could interact with the environment. These sensors can be either attached to vehicles or placed around the available cities and include RGB, depth, and segmentation

[3] Available at: <https://beamng.tech/>. Accessed on: Mar. 17, 2025.
[4] Available at: <https://www.unrealengine.com/>. Accessed on: Mar. 17, 2025.

Figure 6 – Carla simulator interface on Unreal Engine 4.



Source: Created by the authors (2025).

cameras and others such as LiDAR, IMU, and optical flow.

Thanks to the traffic realism, great range of sensors, and open-source nature, the CARLA simulator is extensively used to record synthetic data aimed for urban tasks (HERZOG et al., 2023; KLOUKINIOTIS et al., 2022; DESCHAUD, 2021; DESCHAUD et al., 2021).

3 RELATED WORKS

In this chapter, we present related works in the field of urban synthetic datasets and synthetic data generation tools. We describe an overview of the field and prominent works on it. Subsequently, we compare these works with our own to discuss our contributions to the current literature.

3.1 URBAN SYNTHETIC DATASETS

Urban synthetic datasets simulate elements from city streets and avenues, such as vehicles, pedestrians, and traffic lights, which make them especially suitable for autonomous vehicles and smart city tasks. Due to the versatility of synthetic data, there are synthetic datasets available for diverse tasks, such as vehicle tracking (GAIDON et al., 2016; HERZOG et al., 2023), pedestrian detection (STAUNER et al., 2022; FABBRI et al., 2021), and 3D mapping (DESCHAUD et al., 2021).

The camera positioning on urban synthetic datasets can also be tailored for different application domains. Examples of camera positioning found on available urban synthetic datasets are egocentric vehicle cameras for autonomous vehicles tasks (GAIDON et al., 2016; ROS et al., 2016), aerial cameras (TURKCAN et al., 2024) and surveillance cameras (HERZOG et al., 2023).

The Synthehicle dataset (HERZOG et al., 2023) is a synthetic dataset built using the CARLA simulator (DOSOVITSKIY et al., 2017) and contains 17 hours of video recorded by cameras positioned similarly to surveillance cameras. It contains annotations for 2D and 3D bounding-boxes, depth, multi-camera tracking, and segmentation (instance, class, and panoptic).

A medium-altitude aerial dataset was developed to prove the effectivity of the "Boundless" simulator, developed by Turkcan et al. (2024). The dataset consists of 8K frames collected over a single intersection with 2D and 3D bounding-box annotations. The authors further created similar datasets using CARLA's simulator (22K frames) and their simulator to create a digital twin of a real-world intersection (8.7K frames).

We noticed that existing works predominantly focus on autonomous vehicles tasks, containing egocentric camera viewpoints that capture the vehicle perspective (DESCHAUD, 2021; GAIDON et al., 2016; ROS et al., 2016; KLOUKINIOTIS et al., 2022) while overlooking surveillance camera perspectives, which are crucial for smart city applications. Furthermore, many of these

datasets do not offer a simple end-to-end process for expanding them with new data under customizable settings, keeping their size fixed.

3.2 TOOLS FOR SYNTHETIC DATA GENERATION

Synthetic datasets can be generated by various tools, including game engines, simulation engines, and even computer games (PAULIN; IVASIC-KOS, 2023). Game engines, such as Unity[1] and Unreal Engine[2], are commonly employed to generate synthetic data from simulated environments, which can be done by obtaining data from simulated virtual cities (LI et al., 2023; ROS et al., 2016; KERIM et al., 2021) or by digitally cloning real-world dataset scenes (GAIDON et al., 2016). The increasing realism of computer games has also made them a valuable source for data generation. A notable example is Grand Theft Auto V[3], developed by Rockstar Games, which was used in the works of Fabbri et al. (2021) and Richter et al. (2016) to generate synthetic datasets.

Simulation engines, such as BeamNG.tech[4] and CARLA simulator (DOSOVITSKIY et al., 2017). are also valuable tools for generating synthetic datasets. More details about simulation engines, including CARLA and BeamNG.tech, can be found at section 2.4.

Boundless (TURKCAN et al., 2024) is a dataset generation tool built on top of the CitySample asset[5] from Unreal Engine. It can generate photorealistic RGB images with associated 2D and 3D bounding-box annotations. The simulator supports dynamic weather conditions and time-of-day progression throughout the same simulation.

The NOVA (KERIM et al., 2021) framework is also focused on generating synthetic data from urban scenes but with particular emphasis on virtual humans. The generator allows various camera positions, including surveillance positioning, and features four scenarios in its gallery. A procedural human generator system randomly selects features from a predefined set to create diverse, highly varied virtual humans. Ground-truth annotation is generated for 2D bounding-boxes (for humans), body part segmentation, body pose, optical flow, depth, surface normals, and instance and semantic segmentation.

Current methods for generating synthetic datasets still require some level of user inter-

[1] Available at: <https://unity.com/>. Accessed on: Mar. 17, 2025.
[2] Available at: <https://www.unrealengine.com/>. Accessed on: Mar. 17, 2025.
[3] Available at: <https://www.rockstargames.com/gta-v>. Accessed on: Mar. 17, 2025.
[4] Available at: <https://beamng.tech/>. Accessed on: Mar. 17, 2025.
[5] Available at: <https://www.fab.com/listings/4898e707-7855-404b-af0e-a505ee690e68>. Accessed on: Mar. 17, 2025.

vention during the generation process. This intervention can range from developing a data collection system to making smaller adjustments, such as modifying weather, map settings, or camera positioning.

Comparison with related works

In this work, we propose an end-to-end synthetic dataset generation framework capable of producing large and diverse datasets with a single command. The datasets are easily expandable, as the generator can continuously produce new data. Furthermore, the generated data is captured in the perspectives of surveillance cameras, making the generated datasets well suited for training smart city models.

Comparing our work with the aforementioned related works, we can map how our proposal differs in two ways: (1) Our framework for synthetic dataset generation can create customized and on-demand datasets in an end-to-end fashion, allowing the generation of large datasets and eliminating the need to manually define how the data will be collected. (2) Different from most works, our generated synthetic dataset was designed for smart city applications and is easily expandable due to our framework.

4 THE ENDLESS DATASET GENERATION FRAMEWORK

In this chapter, we describe how the ExpaNdable Datasets Labeled and Empowered by Synthetic Simulation (ENDLESS) framework was implemented and provide an overview of the data generation pipeline. We start the chapter presenting the reasons we developed ENDLESS on top of the CARLA Simulator (DOSOVITSKIY et al., 2017). Next, we provide an overview of the data generation pipeline from the user interaction to the post-processing step, in which the videos are generated from the recorded data.

4.1 DATASET GENERATOR

In order to allow a generator to simulate dynamics from cities and capture relevant ground-truth data, we argue that there are several benefits in using established simulators from the fields of autonomous driving and smart cities. The main benefit is the access to built-in tools that help to simulate city dynamics automatically, such as traffic and pedestrian flow, and to capture data from the environment. Without needing to develop these features from scratch, researchers can focus on generating meaningful and rich data, which may lead to better datasets.

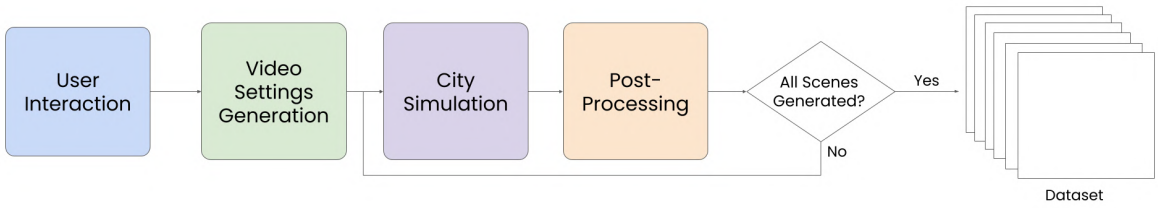
Our main reasons for building our simulation on top of CARLA were its large catalog of assets, widespread adoption in the literature, and the vast range of built-in features. These features include automatic control of city assets, weather manipulation, and sensor simulation for data retrieval. More details about the CARLA simulator can be found at section 2.4.

Our framework follows an automated end-to-end pipeline, simplifying the dataset generation process to the execution of a single Python script after the initial CARLA setup. We represent the full generation pipeline in Figure 7 and explain the steps from the user execution to dataset generation in the subsections below.

4.1.1 User Interaction

As we built our framework on top of the CARLA simulator, users must have it installed on their machines to use our framework. After starting the simulator, our framework can be executed by running a single Python script using the operational system terminal.

Figure 7 – Data generation pipeline



Source: Created by the authors (2025)

To allow users to create customized datasets easily, we provided arguments for the Python script. These arguments allow users to adjust various settings in their datasets. Specifically, users can define the target number of vehicles and pedestrians, enabling the generation of data from environments with varying levels of crowd density—from highly crowded cities to areas with fewer agents. Additionally, users can toggle settings for weather conditions, times of day, and city maps, as outlined in Table 1. Furthermore, these arguments offer control over technical settings related to the generated data, such as frame resolution, video frame rate, and video duration. Users can also specify the number of videos to be generated, allowing for further customization of the data.

Table 1 – Possible values for map, weather, and time of day

Setting	**Possible Values**
Maps	Town01_Opt, Town02_Opt, Town03, Town04_Opt, Town05_Opt, Town06_Opt, Town07_Opt, Town10HD_Opt
Weathers	clear_sky, rainy, cloudy, foggy, after_rain
Times of Day	morning, afternoon, night

Source: Created by the authors (2025)

Arguments also provide control over technical settings from the generated data such as image resolution, and target video frame rate and duration. The number of videos to be generated can also be tailored using arguments.

4.1.2 Video Settings Generation

Once executed, our framework receives the user-defined settings and begins generating the dataset by defining the scene settings. We define a scene as the simulated environment

from where data will be collected in one iteration of the data generation process. The scene settings include the desired number of pedestrians and vehicles, the name of the city that will be loaded on the CARLA simulator, the weather and time of day that will be simulated, and a pair of camera positions.

Using the user-defined settings, our framework obtains all possible scene settings by calculating the cartesian product from the values of city, weather, time of day, and distinct camera pairs. The resulting combination list is shuffled to prevent scenes with similar settings from being generated sequentially.

Since the number of possible combinations can differ from the desired number of videos, the system does one of the following changes:

- If the number of combinations is **higher** than the desired number of videos, the system selects a subset of combinations to match the number of videos;
- If the number of combinations is **lower** than the required number of videos, the system oversamples the combination list by repeating elements proportionally to match the required count.

Finally, the system groups combinations by city to minimize the number of times the CARLA simulator needs to change the active city. The settings from each scene are exported to JSON files, becoming available after the generation as metadata.

4.1.3 City Simulation and Camera Positioning

After the scene settings are generated, the framework retrieves the first of them to obtain the settings for the city simulation process. The city simulation process follows the steps of Algorithm 1.

Firstly, our framework loads the city map from the current scene. To allow the generation of diverse environments, we utilized the eight standard city maps from the CARLA catalog, which range from downtown streets to rural cornfields. Although we experimented with generating cities from a digital twin approach, as we explain in Appendix A, we decided that using the standard cities would allow an improved implementation of the annotation process, which is the critical step we are solving in this work.

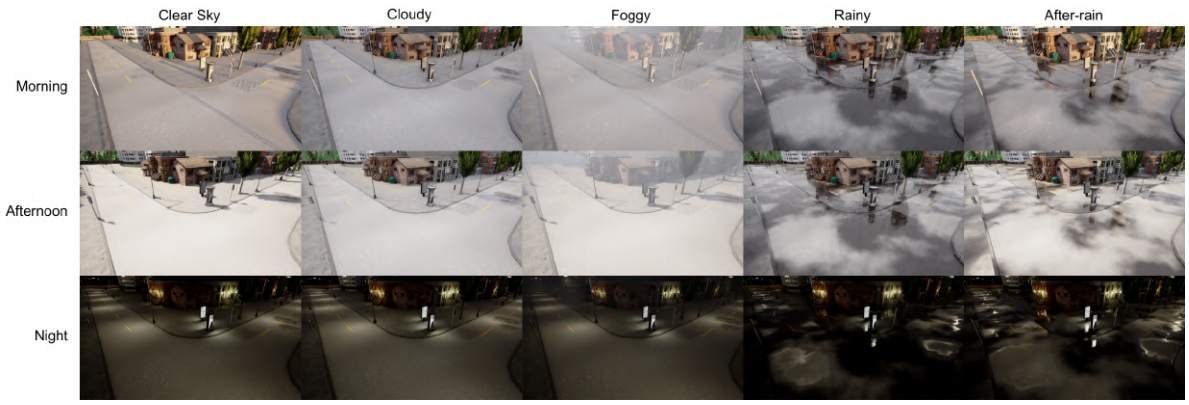
After that, we simulate the time of day and weather specified on the scene settings. By using the CARLA Weather API, we predefined a set of weather and time-of-day conditions that can

Algorithm 1 City Simulation

```
Load city map
Set weather and time of day
Set TrafficManager parameters
                                                          ▷ Spawn Vehicles
BatchVehiclesSpawnCommands ← []
MapSpawnPoints ← Get Map Spawn Points
for SpawnPoint ∈ MapSpawnPoints do
    if Reached desired vehicle count then
        break
    end if
    VehicleModel ← Get Random Vehicle Model
    Set VehicleModel Settings
    Append (Spawn VehicleModel at SpawnPoint, SetAutoPilot) to BatchVehiclesSpawn-
Commands
end for
VehiclesList ← Execute Commands at BatchVehiclesSpawnCommands
Enable Vehicle Lights Control on TrafficManager
                                                         ▷ Spawn Pedestrians
PedestrianSpawnLocations ← []
for i ∈ [0..Target Number of Pedestrians] do
    PedestrianSpawnPoint ← Get Random Pedestrian Spawn Point
    Append PedestrianSpawnPoint to PedestrianSpawnLocations
end for
BatchPedestriansSpawnCommands ← []
PedestrianSpeeds ← []
for PedestrianSpawnLocation in PedestrianSpawnLocations do
    PedestrianModel ← Get Random Pedestrian Model
    Append PedestrianModel.Speed to PedestrianSpeeds
    Append (Spawn PedestrianModel at PedestrianSpawnLocation) to BatchPedestriansS-
pawnCommands
end for
PedestriansList ← Execute Commands at BatchPedestriansSpawnCommands
Remove speeds of unspawned from PedestrianSpeeds
                                                  ▷ Spawn Pedestrian Controllers
PedestrianControllerModel ← Get Pedestrian Controller Model
BatchPControllerSpawnCommands ← []
for i ∈ [0.. Number of Spawned Pedestrians] do
    Append (Spawn PedestrianControllerModel as child of PedestriansList[i]) to BatchP-
ControllerSpawnCommands,
end for
PedestrianControllersList ← Execute Commands at BatchPControllerSpawnCommands
for PedestrianController ∈ PedestrianControllersList do
    Start PedestrianController
    Set Random Target to PedestrianController
    Set Pedestrian Max Speed to PedestrianController
end for
```

be applied in each scene. The available weather conditions include clear sky, rain, clouds, fog, and post-rain; while the possible time of day includes morning, afternoon, and night. To ensure that the generated data can reflect diverse and challenging real-world scenarios, we designed our framework to enable the weather and time-of-day conditions to be adjusted independently. Figure 8 presents the weather and time-of-day conditions available in the simulation.

Figure 8 – Weather and time-of-day conditions



Source: Created by the author (2025)

The city is then populated with agents to simulate the city dynamics. To do this, pedestrians and vehicles are randomly selected from the CARLA catalog and spawned in predefined positions on the city map. Each agent moves to random positions in a non-oriented and infinite path. In particular, vehicles are controlled by the CARLA's `TrafficManager`, which is responsible for their paths and behaviors, such as stopping at red traffic lights or stop signs, slowing down in speed signs, turning on the vehicle lights in reaction to traffic events, and so on.

To enable the data recording, the pair of cameras is positioned in the city according to the scene settings. To grant diversity of views on the recorded data, we positioned 5 cameras for each city simulating positions of real-world surveillance cameras, similar to (HERZOG et al., 2023). The cameras were spread around the city attached to building walls, streetlights, fences, and other structures. During the camera positioning, we tried to maximize the diversity of scenarios that the camera could capture. We included cameras capturing data from road intersections, highways, sloped roads, rural roads, urban parks, and so on.

4.1.4 Recording Data

While the city is being simulated, the framework uses the pairs of cameras, previously positioned, to record RGB images, depth maps, and instance segmentation maps from the environment for the video duration. We can achieve this by placing RGB, depth, and instance segmentation cameras in the same location and with the same settings. This allows us to obtain a perfect match between the sensors' data without any calibration process.

The data from the cameras are recorded as PNG images and named according to the current simulation frame. Instance segmentation images are stored in a way that the red channel defines the pixel class and the green and blue channels, the unique ID from the pixel's object. Depth is stored in RGB images in a way that the distance in meters from the object to the camera can be calculated, as described in the CARLA documentation[1], with the following equation:

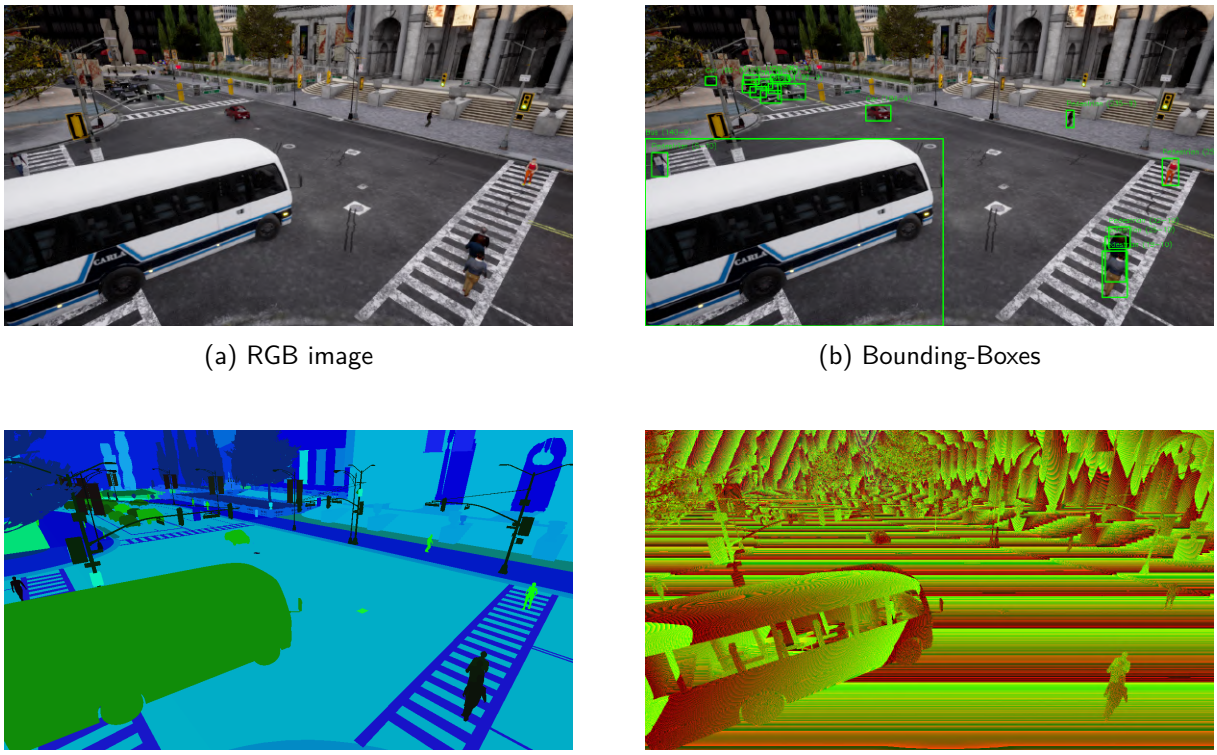
$$D_{\text{meters}} = 1000 \times \frac{R + G \times 256 + B \times 256^2}{256^3 - 1}$$

4.1.5 Post-Processing

Once the video recording finishes, a post-processing step generates videos from the collected data and uses the instance segmentation maps to calculate pixel-perfect bounding-boxes, similar to the method present in (PATHIRAJA; LIU; SENANAYAKE, 2024), from pedestrians and vehicles (which are labeled according to their class such as bus, car, truck, etc). Bounding-Boxes from objects with fewer than fifty visible pixels are discarded as they may represent highly occluded or distant objects. After the post-processing is complete, the generation process until the number of videos generated matches the number of user-defined number of videos on the dataset. Figure 9 shows an example of data that can be generated using our framework.

[1] Available at:<https://carla.readthedocs.io/en/0.9.15/ref_sensors/#depth-camera>. Accessed on: Mar. 17, 2025.

Figure 9 – Example of output data provided by the generator



(a) RGB image

(b) Bounding-Boxes

(c) Instance Segmentation

(d) Depth Map

Source: Created by the authors (2025)

5 A NOVEL DATASET FOR SMART CITIES APPLICATIONS

In this chapter, we present a novel synthetic dataset created to prove the ExpaNdable Datasets Labeled and Empowered by Synthetic Simulation (ENDLESS) framework capability of generating large and automatically annotated datasets. We describe the dataset and provide further analysis of the data distribution and dataset parameters to provide insights about the generated data.

5.1 DATASET DESCRIPTION

To demonstrate the ENDLESS capability, we used the framework to create a substantial synthetic dataset, consisting of 378,751 frames distributed across 244 HD videos recorded at a frame rate of 30 Frames Per Second (FPS). Figure 10 contains image samples collected from the generated dataset. Notice the diversity of environments, viewpoints, and environmental conditions obtained by using the framework.

Figure 10 – Samples from the dataset images

Source: Created by the authors (2025)

The final dataset was made composing six mini-datasets (hereafter referred to as "minisets") generated on different days using a consumer-grade computer equipped with an RTX 3090 Graphics Processing Unit (GPU). Notably, the final dataset is expandable, as users can use the

ENDLESS framework to generate additional minisets to extend the dataset size. Furthermore, as mentioned in subsection 4.1.1, the additional user-generated minisets can be customized to allow greater control over the generated data.

To better demonstrate our framework features, all minisets were generated with all the weather, times of day, and city maps enabled. The only distinction regarding their settings is the maximum number of pedestrians and vehicles and the number of videos to be generated.

5.2 DATASET ANALYSIS

In this section, we analyze the generated dataset to have better insights into the data distribution and verify the ENDLESS data generation. To achieve that, we analyzed the data distribution of the 6 generated minisets and also of parameters such as weather, time of day, and city maps from the final dataset.

In Table 2, we present the data distribution for the six minisets along with their target number of vehicles and pedestrians. The results evidence that the second miniset exceeds the number of frames and videos from the other minisets by a high margin. This result indicates that this miniset has the highest influence on the final dataset compared with the others.

Table 2 – Minisets specifications

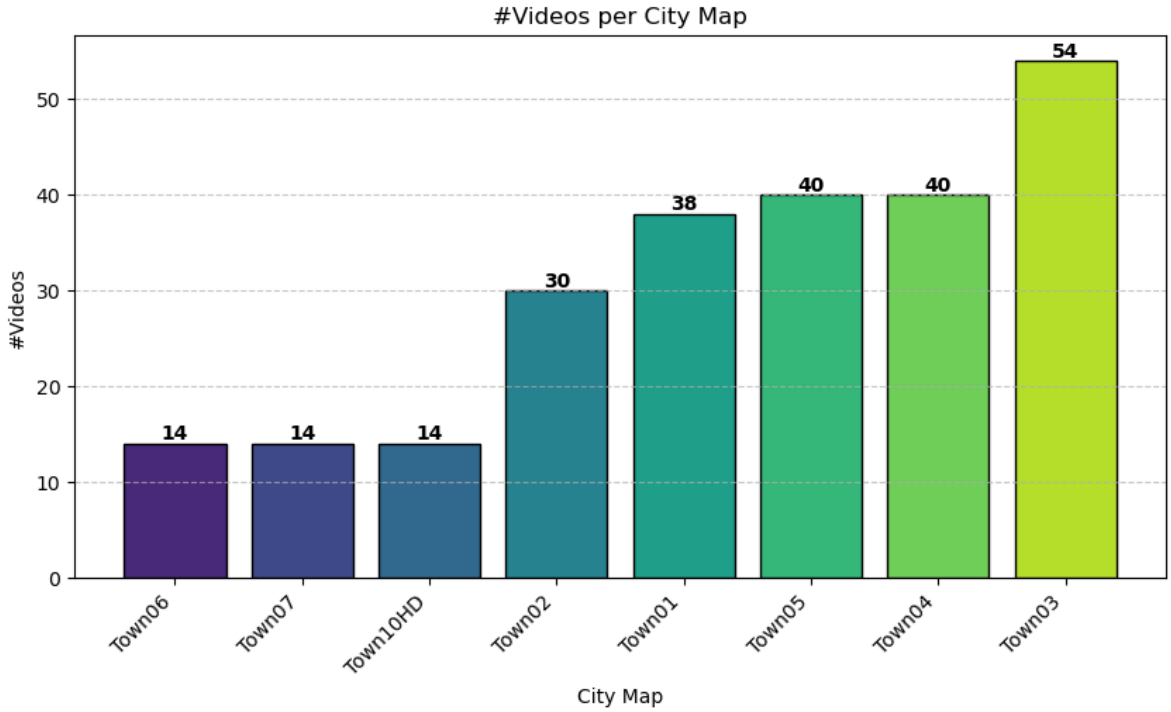
Miniset	#Videos	#Frames	Max Vehicles	Max Pedestriabs
1	18	27,632	30	20
2	104	157,269	30	20
3	30	44,957	30	20
4	40	64,050	100	60
5	30	46,840	100	60
6	22	38,003	150	100
Total	244	378,751		

Source: Created by the author (2025)

An analysis of the dataset city map distribution is presented in Figure 11. It's visible that the cities were not used evenly across the video generation. We suppose that this is due to the random city selection and order by the city map process effectuated during the scene settings generation, as mentioned in subsection 4.1.2.

To better analyze the environmental conditions of the dataset, we analyzed the time of day and weather distributions both independently and in combination. This distribution is presented on Figures 12, 13, and 14. From the graphs, we note that the least represented

Figure 11 – Dataset City Map Distribution

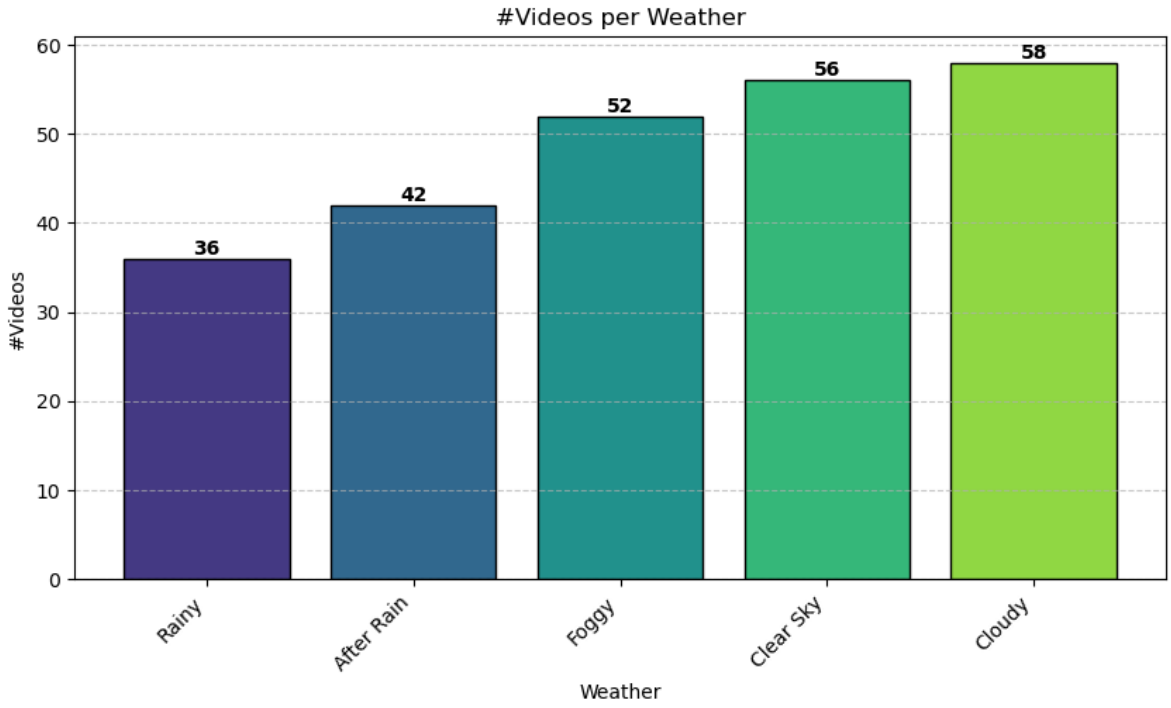


Source: Created by the authors (2025)

weather condition in our dataset is "*rainy*", while the most represented is "*cloudy*". It is also observed that the nighttime period has the most data, whereas the morning has the least. Additionally, it is worth noting that the three most representative environmental conditions in the dataset, which consider both the video's weather and time of day, have different weather and times of day values, namely "*clear-sky afternoon*", "*cloudy morning*", and "*foggy night*".

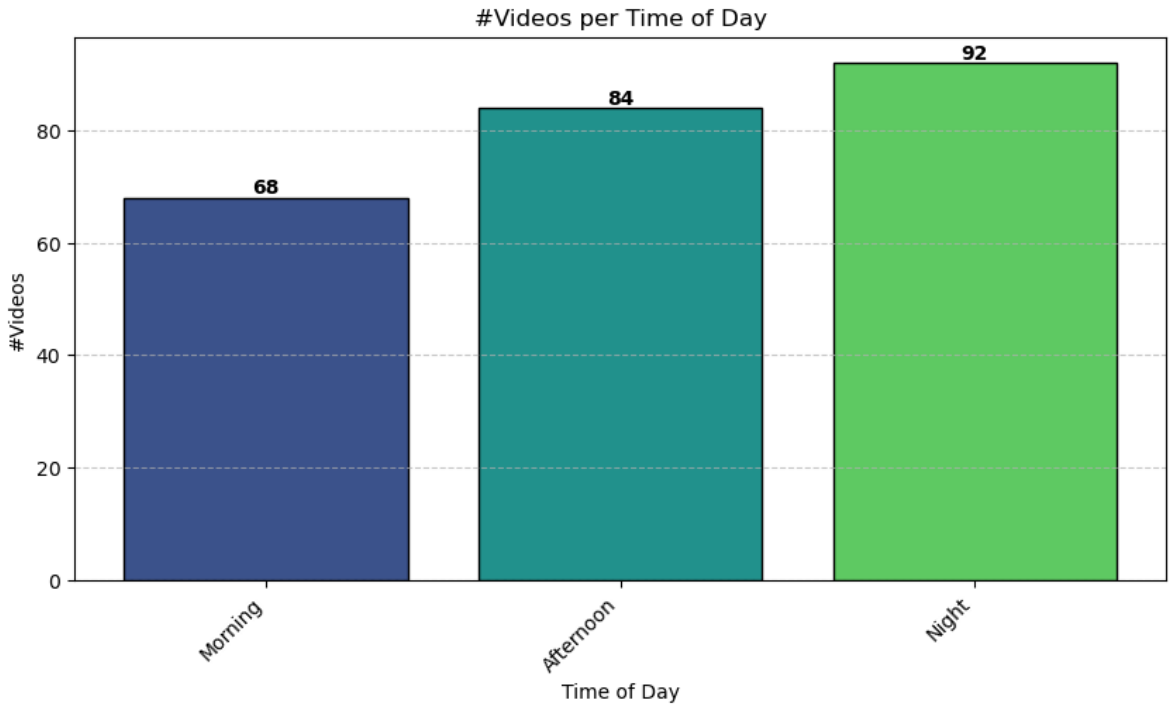
From the environmental graphs, we also observe that the disparity in the number of videos across classes increases as the number of classes grows. This is expected due to the random selection of weather and time of day values. This trend can be verified by comparing the time of day distribution, which has only three possible values, with the combined time of day and weather distribution, which has 15 possible values.

Figure 12 – Dataset weather distribution



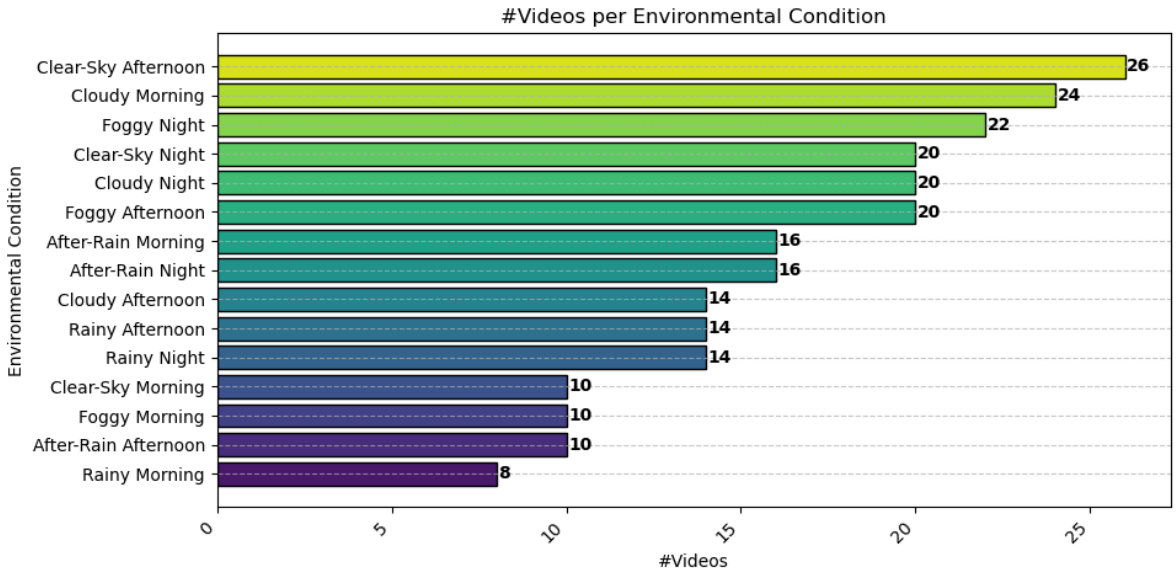
Source: Created by the author (2025)

Figure 13 – Dataset time of day distribution



Source: Created by the authors (2025)

Figure 14 – Dataset environmental condition distribution



Source: Created by the authors (2025)

6 RESULTS AND DISCUSSION

To prove that our generator can create competitive synthetic datasets, we compared the generated dataset with three related synthetic datasets that are publicly available. The datasets we chose for comparison were the Synthehicle dataset (HERZOG et al., 2023), and the Boundless, Boundless+Digital Twin, and CARLA datasets (TURKCAN et al., 2024). Further information about these datasets can be found in section 3.1.

We have chosen the Synthehicle dataset for comparison because its camera positioning and focus on smart city applications make it the most similar to ours. Boundless datasets were selected because they were released to prove the efficiency of a similar generator developed in the Unreal Engine 5[1]. We consider that comparing the resulting dataset with real-world datasets is out of the scope of this work.

The features selected to compare the datasets were: the number of frames; the number of distinct camera views; and the availability of 2D or 3D bounding-boxes, segmentation, or depth map annotations. In this work, we consider a camera view as a perspective captured from the camera based on its positioning in a given city.

Table 3 – Comparison with other datasets

Dataset	#Frames	#Views	2D Boxes	3D Boxes	Segmentation	Depth
Synthehicle	612,000	40	X	X	X	X
Boundless	8,000	1	X	X		
Boundless + Digital Twin	16,700	2	X	X		
Boundless (Carla)	22,000	1	X	X		
Ours	378,751	40	X		X	X

Source: Created by the author (2025)

The results of our comparison are presented in Table 3. Note that we consider the "Digital Twin+Boundless" dataset to have 2 distinct views as it is a merge of two datasets. Nevertheless, the camera positioning of both mimics the same real-world camera positioning. Additionally, the number of frames for the Synthehicle dataset was estimated based on the reported frame count per video and the total number of videos since the exact number was not provided in the published paper.

Our dataset contains more than seventeen more frames than the largest dataset from Boundless, made on CARLA, and more than twenty times the number of frames of the "Boundless+Digital Twin" dataset. Despite that, the Synthehicle contains approximately 61% more

[1] Available at:<https://www.unrealengine.com/en-US/unreal-engine-5>. Accessed on: Mar. 17, 2025.

frames than our dataset. In terms of view count, our dataset is equivalent to Synthehicle and surpasses Boundless' by a big margin, once they are recorded from the same viewpoint.

Regarding recorded ground truth, both Boundless and Synthehicle contain 3D bounding-box ground-truth data, which are not available on our dataset. Other than that, our dataset contains depth, instance segmentation, and 2D bounding-boxes annotations, which are also available on Synthehicle but absent on Boundless.

Regarding the features compared in Table 3, it's noticeable that Synthehicle is a larger dataset and contains a bigger range of annotations than ours. However, it is important to notice that our work is not a single dataset, but rather a framework for dataset generation. In this case, the quantity of frames should not be taken into account as to measure in any way quality.

7 CONCLUSION

In this work, we presented an end-to-end synthetic dataset generator for smart city tasks developed using the CARLA Simulator. The generator can create automatically annotated synthetic datasets by a single script execution, mitigating the need for costly manual annotation common in real-world datasets. The generated data includes diverse scenarios with distinct camera views, weather conditions, time of day, vehicles, and pedestrians, ensuring a diverse data generation.

To prove the competence of our generator, we used it to generate a proof-of-concept dataset with over 300K frames and compared it with state-of-the-art related synthetic datasets. Our comparison shows that the generated dataset is paired with state-of-the-art datasets regarding frame number and number of sensors, proving the generator's competence.

7.1 LIMITATIONS

The main limitation of our work is that synthetic data generated by our framework was not used yet to train and test Computer Vision (CV) models to evaluate how they can impact their metrics. However, we expect that our data can improve these models' performance as other works with urban synthetic datasets were able to do.

Another limitation is regarding the data generation. In the initial frames of the city simulation, the vehicles appear "falling from the sky" due to how the CARLA Simulator instantiates them. This unnatural event was recorded by the cameras during the construction of our dataset.

Finally, recorded videos may contain a different duration from the user-specified duration. We believe this happened due to how the CARLA simulator processes its ticks.

7.2 FUTURE WORKS

From its current state, there are plenty of possible ways to improve our generator. Adding new annotations such as 3D bounding-boxes, monk-skin tones of pedestrians, and traffic-light states would increase the possible usages of the generated data. Additionally, adding new maps inspired by Latin American scenarios would allow us to simulate representative data with the specificity of its cities and streets. Finally, testing generated datasets using meaningful deep-

learning models for smart city tasks would enable us to check how well these models perform on our data.

REFERENCES

ADEWOPO, V. A.; ELSAYED, N.; ELSAYED, Z.; OZER, M.; ABDELGAWAD, A.; BAYOUMI, M. A review on action recognition for accident detection in smart city transportation systems. *Journal of Electrical Systems and Information Technology*, Springer, v. 10, n. 1, p. 57, 2023.

BARTHÉLEMY, J.; VERSTAEVEL, N.; FOREHEAD, H.; PEREZ, P. Edge-computing video analytics for real-time traffic monitoring in a smart city. *Sensors*, MDPI, v. 19, n. 9, p. 2048, 2019.

BHAT, S. F.; BIRKL, R.; WOFK, D.; WONKA, P.; MÜLLER, M. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288*, 2023.

CHOURABI, H.; NAM, T.; WALKER, S.; GIL-GARCIA, J. R.; MELLOULI, S.; NAHON, K.; PARDO, T. A.; SCHOLL, H. J. Understanding smart cities: An integrative framework. In: *2012 45th Hawaii International Conference on System Sciences*. [S.l.: s.n.], 2012. p. 2289–2297.

CORDTS, M.; OMRAN, M.; RAMOS, S.; REHFELD, T.; ENZWEILER, M.; BENENSON, R.; FRANKE, U.; ROTH, S.; SCHIELE, B. The cityscapes dataset for semantic urban scene understanding. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 3213–3223.

DESCHAUD, J.-E. Kitti-carla: a kitti-like dataset generated by carla simulator. *arXiv preprint arXiv:2109.00892*, 2021.

DESCHAUD, J.-E.; DUQUE, D.; RICHA, J. P.; VELASCO-FORERO, S.; MARCOTEGUI, B.; GOULETTE, F. Paris-carla-3d: A real and synthetic outdoor point cloud dataset for challenging tasks in 3d mapping. *Remote Sensing*, MDPI, v. 13, n. 22, p. 4713, 2021.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S. et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

DOSOVITSKIY, A.; ROS, G.; CODEVILLA, F.; LOPEZ, A.; KOLTUN, V. CARLA: An open urban driving simulator. In: *Proceedings of the 1st Annual Conference on Robot Learning*. [S.l.: s.n.], 2017. p. 1–16.

FABBRI, M.; BRASÓ, G.; MAUGERI, G.; CETINTAS, O.; GASPARINI, R.; OŠEP, A.; CALDERARA, S.; LEAL-TAIXÉ, L.; CUCCHIARA, R. Motsynth: How can synthetic data help pedestrian detection and tracking? In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. [S.l.: s.n.], 2021. p. 10849–10859.

GAIDON, A.; WANG, Q.; CABON, Y.; VIG, E. Virtual worlds as proxy for multi-object tracking analysis. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 4340–4349.

GEIGER, A.; LENZ, P.; URTASUN, R. Are we ready for autonomous driving? the kitti vision benchmark suite. In: IEEE. *2012 IEEE conference on computer vision and pattern recognition*. [S.l.], 2012. p. 3354–3361.

GIFFINGER, R.; FERTNER, C.; KRAMAR, H.; KALASEK, R.; PICHLER-MILANOVIC, N.; MEIJERS, E. J. Smart cities. ranking of european medium-sized cities. final report. 2007.

HE, K.; GKIOXARI, G.; DOLLÁR, P.; GIRSHICK, R. Mask r-cnn. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2017. p. 2961–2969.

HERZOG, F.; CHEN, J.; TEEPE, T.; GILG, J.; HÖRMANN, S.; RIGOLL, G. Synthehicle: Multi-vehicle multi-camera tracking in virtual cities. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. [S.l.: s.n.], 2023. p. 1–11.

KERIM, A.; ASLAN, C.; CELIKCAN, U.; ERDEM, E.; ERDEM, A. Nova: Rendering virtual worlds with humans for computer vision tasks. In: WILEY ONLINE LIBRARY. *Computer Graphics Forum*. [S.l.], 2021. v. 40, n. 6, p. 258–272.

KLOUKINIOTIS, A.; PAPANDREOU, A.; ANAGNOSTOPOULOS, C.; LALOS, A.; KAPSALAS, P.; NGUYEN, D.-V.; MOUSTAKAS, K. Carlascenes: A synthetic dataset for odometry in autonomous driving. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2022. p. 4520–4528.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Ieee, v. 86, n. 11, p. 2278–2324, 1998.

LI, Y.; JIANG, L.; XU, L.; XIANGLI, Y.; WANG, Z.; LIN, D.; DAI, B. Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. [S.l.: s.n.], 2023. p. 3205–3215.

PATHIRAJA, B.; LIU, C.; SENANAYAKE, R. Fairness in autonomous driving: Towards understanding confounding factors in object detection under challenging weather. *arXiv preprint arXiv:2406.00219*, 2024.

PAULIN, G.; IVASIC-KOS, M. Review and analysis of synthetic dataset generation methods and techniques for application in computer vision. *Artificial intelligence review*, Springer, v. 56, n. 9, p. 9221–9265, 2023.

REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 779–788.

REN, S.; HE, K.; GIRSHICK, R.; SUN, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, v. 28, 2015.

RICHTER, S. R.; VINEET, V.; ROTH, S.; KOLTUN, V. Playing for data: Ground truth from computer games. In: SPRINGER. *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. [S.l.], 2016. p. 102–118.

ROS, G.; SELLART, L.; MATERZYNSKA, J.; VAZQUEZ, D.; LOPEZ, A. M. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 3234–3243.

SILVA, B. N.; KHAN, M.; JUNG, C.; SEO, J.; MUHAMMAD, D.; HAN, J.; YOON, Y.; HAN, K. Urban planning and smart city decision management empowered by real-time data processing using big data analytics. *Sensors*, MDPI, v. 18, n. 9, p. 2994, 2018.

STAUNER, T.; BLANK, F.; FÜRST, M.; GÜNTHER, J.; HAGN, K.; HEIDENREICH, P.; HUBER, M.; KNERR, B.; SCHULIK, T.; LEISS, K.-F. Synpeds: A synthetic dataset for pedestrian detection in urban traffic scenes. In: *Proceedings of the 6th ACM Computer Science in Cars Symposium*. [S.l.: s.n.], 2022. p. 1–10.

SYAHIDI, A. A.; KIYOKAWA, K.; OKURA, F. Computer vision in smart city application: A mapping review. In: IEEE. *2023 6th International Conference on Applied Computational Intelligence in Information Systems (ACIIS)*. [S.l.], 2023. p. 1–6.

TURKCAN, M. K.; LI, Y.; ZANG, C.; GHADERI, J.; ZUSSMAN, G.; KOSTIC, Z. Boundless: Generating photorealistic synthetic data for object detection in urban streetscapes. *arXiv preprint arXiv:2409.03022*, 2024.

YAR, H.; KHAN, Z. A.; ULLAH, F. U. M.; ULLAH, W.; BAIK, S. W. A modified yolov5 architecture for efficient fire detection in smart cities. *Expert Systems with Applications*, Elsevier, v. 231, p. 120465, 2023.

ZAMAN, M.; PURYEAR, N.; ABDELWAHED, S.; ZOHRABI, N. A review of iot-based smart city development and management. *Smart Cities*, MDPI, v. 7, n. 3, p. 1462–1501, 2024.

APPENDIX A – GENERATING SYTHETIC DATA FROM LATIN AMERICAN CITIES

Besides bridging the gap between real and synthetic data, one of the core concepts of our project is to also allow regional data generation, specifically with the objective of enabling smart cities interactions in Latin America. In this appendix, we discuss one of our experiments with generating a digital twin using CARLA from a section of the *Universidade Federal de Pernambuco* (UFPE) campus. Our goal was to enable the dataset generation for this portion of the campus as well.

In this experiment, we collected geographic data using OpenStreetMap[1] and recreated it in Unreal Engine using native tools for procedural generation on CARLA. The recreated region includes the *Centro de Informática* (CIn) and has a closed-loop traffic circuit where vehicles could go in and out (see in Figure 15). Buildings from the real world are replicated procedurally, respecting their sizes and dimensions contained in the input data.

Figure 15 – Digital twin experiment result.

Source: Created by the authors (2025)

The reconstruction is not perfect, however, it serves as a good starting point for creating more precise digital twins. In addition, the recreation includes a traffic circuit that resembles the original, allowing experiments about how autonomous vehicles could navigate in this campus loop.

[1] Available at:<https://www.openstreetmap.org/>. Accessed on: Mar. 17, 2025.