# POLITECNICO DI TORINO

Master's Degree in Automation and Intelligent Cyber-Physical Systems


Master's Degree Thesis

# Design and Validation of Autonomous Perception Pipelines: Knowledge Transfer from Micromobility to Macromobility

Supervisors

Dr. PANGCHENG DAVID CEN CHENG

Ing. VALERIA PROIETTI DANTE

Candidate

ANGELA DI FAZIO

December 2024

**Abstract**

In recent years, research on Advanced Driver Assistance Systems (ADAS) has gained significant momentum, due to the growing need for technologies that enhance safety, comfort and efficiency in transportation. ADAS are designed to support drivers in both Micromobility and Macromobility devices by monitoring the surroundings with sensors, cameras, RADAR and LiDAR. These systems provide real-time data to assist the driver navigate safely by detecting potential hazards, such as pedestrians or other vehicles. Through early warnings or automatic interventions, ADAS help to prevent collisions while also supporting drivers to maintain safe distances, adjust the driving speed to traffic and assist with lane-keeping.

To ensure the effective development of these advanced systems, a scalable and safe testing environment is essential. This allows for controlled and progressive testing of ADAS, minimizing risks and ensuring accurate validation. Using Micromobility devices instead of Macromobility vehicles reduces the potential for serious accidents during early testing phases. These smaller and lightweight vehicles operate at lower speeds, making it safer to identify and correct system errors, as well as accelerating the implementation of ADAS across various modes of transportation.

This thesis primarily focuses on Micromobility devices, leveraging their unique advantages to develop innovative ADAS solutions. These vehicles provide a cost-effective and flexible platform for testing and experimentation, which can later be transferred to Macromobility systems. By taking advantage of their lower cost, ease of implementation and simpler architecture, Micromobility vehicles serve as ideal testbeds for developing and refining ADAS that can be scaled up to more complex and larger transportation systems.

A key aspect of the research is the development of an Image Processing pipeline for Object Detection, focusing on the optimization and real-time computation of the input data. The pipeline validation was performed using a Micromobility platform developed in collaboration with the company Teoresi S.p.A, on which two perception sensors were installed: a stereocamera and a 4D RADAR.

This thesis emphasizes the pivotal role of Micromobility devices as a bridge for ADAS development, leveraging their cost-effectiveness and flexibility to generate insights and innovations that can be applied to more complex transportation systems.

# Table of Contents


**1 Introduction** **7**

1.1 Background . . . . . 7

1.2 Research Motivation . . . . . 8

1.3 Thesis Outline . . . . . 9

**2 Advanced Driver Assistance Systems (ADAS)** **10**

2.1 Introduction to ADAS . . . . . 10

2.2 Classification and Regulation of ADAS . . . . . 11

2.3 Main Technologies . . . . . 15

**3 Micromobility: a New Paradigm** **19**

3.1 Definition of Micromobility . . . . . 19

3.2 Types of Vehicles for Micromobility . . . . . 20

3.2.1 Bicycles . . . . . 20

3.2.2 E-bikes . . . . . 21

3.2.3 Electric Scooter . . . . . 21

3.2.4 Electric Skateboard . . . . . 22

3.2.5 Industrial Robots . . . . . 24

3.3 Advantages of Micromobility . . . . . 25

3.4 Integration of ADAS in Micromobility . . . . . 26

3.5 Possibilities and Technical Challenges . . . . . 28

3.6 Case Studies . . . . . 30

3.6.1 Honda Cooperative Intelligence . . . . . 30

3.6.2 ALBA Robot . . . . . 31

**4 Materials and Methods** **32**

4.1 Materials . . . . . 32

4.2 Hardware . . . . . 33

4.2.1 NVIDIA Jetson AGX Orin . . . . . 33

4.2.2 Perception Sensors . . . . . 35

4.3 Software . . . . . 46

4.3.1 ROS2 Framework . . . . . 46
4.3.2 NVIDIA Isaac ROS . . . . . 50
4.4 Methods . . . . . 52
4.4.1 General Architecture . . . . . 52
4.4.2 Vulnerable Road Users Detection . . . . . 53
4.4.3 System Setup . . . . . 60
4.4.4 Stereocamera Pipeline . . . . . 64
4.4.5 4D RADAR Integration . . . . . 71
4.4.6 Data Fusion . . . . . 75

**5 Results** **80**

**6 Conclusions and Future Works** **83**

**Bibliography** **87**

# Chapter 1

# Introduction

## 1.1 Background

Since their invention, cars have been the favorite means of transport for generations up to the present day. Nowadays, the car continues to be a cornerstone of humans' lives, so much so that it is also the protagonist of several studies aimed at making driving easier and safer with the introduction of some technologies. The purpose of some of these studies is to make the vehicles completely autonomous: able to use sensors to obtain information from the environment and to take decision based on them, without human intervention.

The first self-driving prototypes vehicles date back to the early 1900s, by an American company that designed a radio-controlled vehicle, tested first on the streets of New York and later in Milwaukee. In the 1950s, General Motors resumed these efforts by introducing vehicle acceleration and braking control. The first truly Autonomous Vehicles emerged in the 1980s in Germany and the United States, marking a significant milestone. Since then, technological advancements have enabled vehicles to travel longer distances with full autonomy [1].

A critical component of these self-driving vehicles is represented by the perception system, which is responsible for scanning the surrounding environment and determining whether any detected objects pose a risk. This system mimics human perception by using sensors, such as RADARs, LiDARs, sonars, and cameras. Based on this information, the vehicle's control system makes decisions to ensure safe operation.

The development of the perception system in vehicles facilitated the introduction of Advance Driver Assistant System (ADAS), technologies that assist the driver, and can be classified according to their primary functions [2]: navigation, path planning, perception and car control. Navigation involves planning a route from the origin to the destination, using maps in combination with with a perception system to detect

anomalies. Path planning consists in the definition of segments for the vehicle to move towards the destination.

A typical feature of Autonomous Vehicles is the use of Drive-by-Wire (DbW): the use of electronic or electro-mechanical systems in place of mechanical linkages that control driving functions.

Progress in Autonomous Vehicles has been significantly supported by developments in Computer Vision, a part of Artificial Intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos, and other visual inputs [3].

Researchers began exploring the possibilities of Computer Vision in the 1960s, with early milestones such as Optical Character Recognition (OCR) in 1974, which allowed systems to recognize printed text in any font.

Starting from 2000, the focus shifted to real-time Object Detection, an essential component for Autonomous Vehicles. Indeed Computer Vision enables vehicles to interpret and understand live inputs, captured through sensors, and process them to recognize significant obstacles for making autonomous driving decisions.

In addition to Object Detection, Computer Vision also supports other useful functions for Autonomous Vehicles, e.g., Lane Tracking, Road Mapping and Vehicle Detection. These applications are vital for ensuring both the safety and performance of self-driving cars, and advancements in this field are key to improving vehicle capabilities.

The widespread adoption of Autonomous Vehicles promises several benefits, including reduced traffic congestion, safer roads, and increased mobility for those who are unable to drive. However, the implementation of such technologies faces challenges, particularly in the realm of Computer Vision. These challenges include:

- environmental variability that can affect the visibility of the sensors, for example, particular weather conditions and road scenarios;
- lack of industry-wide standards and regulations;
- data privacy and security for sensitive information;
- scalability and cost to introduce such technologies into vehicles.

Despite these challenges, the continued development of Autonomous Vehicles and their underlying technologies holds great promise for the future of transportation.

## 1.2 Research Motivation

The thesis aims to develop a kit of sensors, for both Micro and Macromobility devices, with the final purpose of enabling Object Detection of the environment

around the vehicle. The research motivation deals with the interest in improving the perception system of Micromobility vehicles, making it more reliable and capable of recognizing its surroundings, and subsequently transferring this knowledge to Macromobility vehicles.

A fundamental aspect of advancing ADAS technology is a thorough understanding of both the systems themselves and the sensors that can be used to improve them. By analyzing various types of sensors and determining the most suitable combinations, significant improvements can be achieved. When selecting sensors for vehicles, several factors must be considered, including the specific application, the operational environment, sensor range, accuracy, response time, and integration capabilities.

The choice of sensors for vehicles also depends on the context and conditions in which they will be deployed. For instance, the challenges of Object Detection may vary between urban scenarios, outdoor or indoor environments, as well as the interference caused by other surfaces.

Among the sensors selected for this thesis, the 4D RADAR is particularly noteworthy for its ability to complement the data from cameras: it is more robust in adverse weather conditions and can detect obstacles even at considerable distances, offering a very wide Field of View (FOV). On the other hand, camera outputs tend to be more interpretable than RADAR data, with higher detection accuracy. The most effective solution arises from fusing the data received by these different sensors, in order to provide a more comprehensive understanding of the environment, integrating information from both RADAR and cameras.

## 1.3 Thesis Outline

The remainder of the thesis is organized as follows:

- **Chapter 2** presents with an overview of the Advanced Driver Assistance Systems, pointing out their classification and regulation;
- **Chapter 3** explains the concept of Micromobility, different types of vehicles and their advantages;
- **Chapter 4** refers to the material used, both hardware and software, and the methods adopted for the research;
- **Chapter 5** discusses the tests conducted and the obtained results.

# Chapter 2

# Advanced Driver Assistance Systems (ADAS)

## 2.1 Introduction to ADAS

The ADAS are technologies designed to enhance safety, both for the driver and passengers inside the vehicle, as well as for pedestrians and cyclists outside. In recent years, ADAS have become increasingly common, with various and innovative solutions being introduced in order to prevent accidents and assist the driver, making the driving experience being.

Vehicles equipped with ADAS protocols features a centralized system, capable of influencing the vehicle's movement, along with a kit of sensors that collaborate to provide real-time information about the surrounding environment.

The introduction of ADAS aims to reduce accidents, mainly caused by human distractions, fatigue or lapses in attention. In the early 20th-century, basic mechanical systems were introduced, like basic cruise control, which is useful to maintain the vehicle at a constant speed. In the 1970s, Anti-Lock Braking Systems (ABS) were developed to prevent wheel lockup during braking, helping the driver to maintain steering control. The 1990s saw the introduction of the Electronic Stability Control (ESC) and the first RADAR-Based Adaptive Cruise Control (ACC), that allows to keep a safe distance from the vehicles ahead perceived by the sensor. The ADAS have become progressively advanced, for example, incorporating devices for Lane Tracking, Traffic Sign Recognition and Driver Monitoring System [4]. Therefore, there are several benefits to introduce ADAS into vehicles:

- enhanced safety, providing warnings and interventions in case of danger;
- more comfort in driving, assisting the driver and reducing his/her workload;
- better efficiency in energy management, reducing emissions and improving fuel

consumption.

Despite their benefits, the development of ADAS faces challenges, such as the lack of universal regulations to ensure consistency and scalability across vehicles, as well as the complexity of the required technologies. Once these obstacles are overcome, the ultimate goal of ADAS is to enable fully Autonomous Vehicles that can operate without any human intervention, by improving the sensors and algorithms used.

## 2.2 Classification and Regulation of ADAS

Nowadays, there are many obstacles that hinders the progress of vehicles being completely autonomous. Among the most challenging scenarios, the heavy interference due to the environment plays an important role, mainly because of the layout of roads and the unpredictability of human behavior.

The Society of Automotive Engineers (SAE) defined a standard for the self driving car, including Dynamic Driving Task (DDT) that comprises all of the real-time operational and tactical functions required to operate a vehicle in on-road traffic, excluding the strategic functions such as trip scheduling and selection of destinations and waypoints [5]. The DDT includes some subtasks:

- Object and Event Detection and Response (OEDR);
- Operational Design Domain (ODD);
- Automatic Driving System (ADS);
- DDT Fallback;
- Minimal risk condition.

The OEDR is the subtask referred to the monitoring of the surrounding environment, including detection and classification of objects, as well as executing an appropriate response to these events [5]. The ODD defines the operating conditions under which an Automated Driving System (ADS) is designed to operate safely and effectively. It sets limitations in the road environment, in the behavior of the ADS-equipped subject vehicle and in the vehicle's operational state. The limitations regarding the road environments determine types of roads, temporary structures, weather, visibility conditions, while the information about ADS behavior and state include speed or maneuvers limitations [6]. The ADS comprehends both the hardware and the software capable of performing all DDT subtasks. The DDT fallback is the set of procedures and mechanisms to overcome system failures or situations in which the vehicle continue operating. As a consequence, the minimal risk condition is a state

sought by the user or the ADS, after the DDT fallback, in order to avoid dangerous situations.

Moreover the SAE defined levels of driver support, as shown in Figure 2.1. Such levels are six, from Level 0 to Level 5:

- Level 0 (No Driving Automation): all the basic functions in driving are demanded to the driver (steering, braking and accelerating), since there is support of assistance systems only for warnings or momentary assistance;
- Level 1 (Driver Assistance): the driver is still responsible of the main functions, however, the assistance system provides support either steering and braking or accelerating. It can act on the lateral and longitudinal dynamics, but not simultaneously;
- Level 2 (Partial Automation): the ADAS can control the longitudinal and lateral dynamics at the same time, in any case under the supervision of the driver;
- Level 3 (Conditional Automation): the vehicle can operate in certain tasks and scenarios without human intervention, but the driver is still necessary to intervene in unmanageable scenarios;
- Level 4 (High Automation): the vehicle can behave in self-driving mode, in a restricted area, even in case of system failures;
- Level 5 (Full Automation): human attention is not required, since all the main functions are carried out by the vehicle.

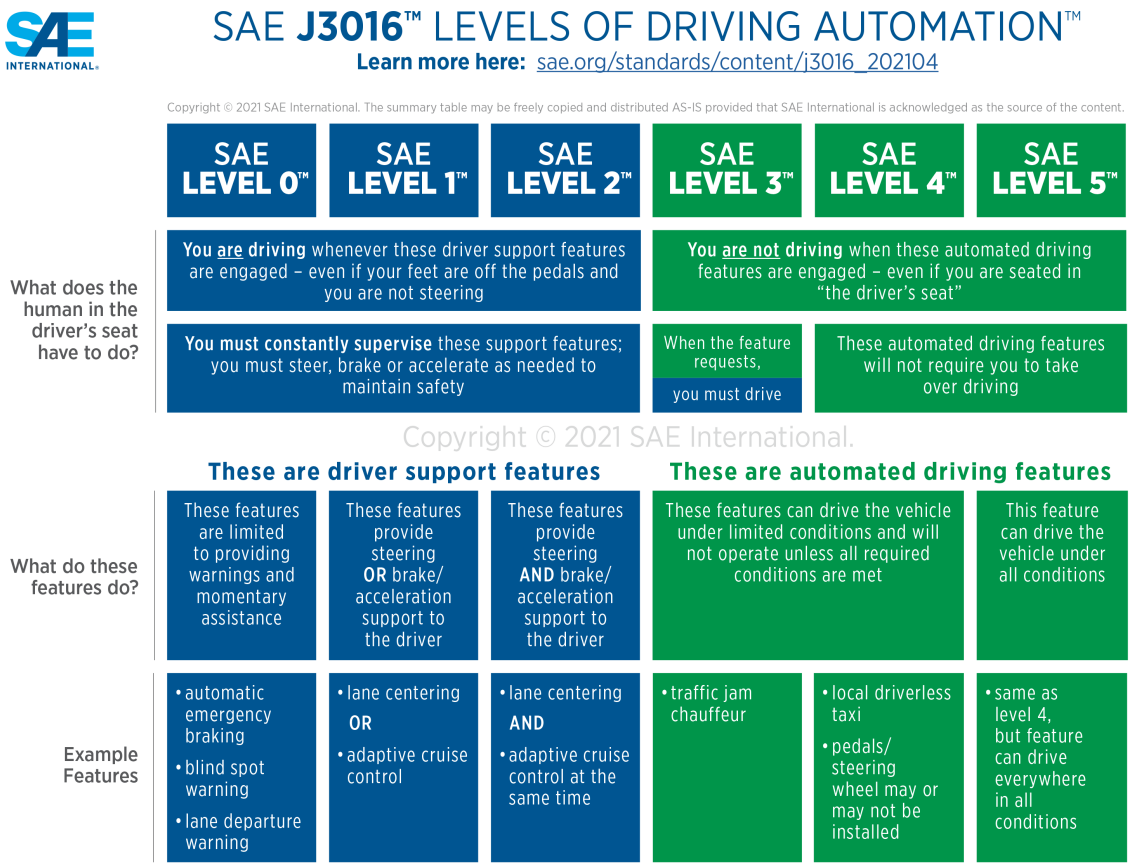



Figure 2.1: SAE levels of automation [7]

In addition, ADAS can be divided according to their functionalities [8]:

- Alerts and warnings, involving devices like parking sensors to help the driver with audio and camera warnings scanning the environments or Lane Departure Warning (LDW) system to alert the driver when the vehicle exits from its lane without using turn signals;
- Crash mitigation, designed to partially reduce the number of injuries, in particular between the vehicle and pedestrians;
- Driving task assistance, useful to make driving easier and more comfortable, combined with increased vehicle and road safety. For instance the following belong to this category: ACC, ABS, ESC;
- Visual and environmental monitoring, with either cameras, RADAR, LiDAR or both, to assist the driver in traffic sign recognition, pedestrian detection in low visibility, automotive night vision, as well as a full 360-degree view of the vehicle itself, known as omniview technology;

- Hands-off systems that allow driver to take hands off the steering wheel while still keeping his attention on the road.

The ADAS regulation is a topic still being defined and evolving, considering the recent introduction of these devices in vehicles. As mentioned before, some standards are adopted in order to standardize vehicle production. Furthermore, the regulations in Europe regarding safety features of vehicles have been drawn up in the General Safety Regulation (GSR2), an official document establishing requirements to be considered by new vehicles put on the market [9]:

- for the type-approval of vehicles, systems, components and separate technical units designed and constructed for vehicles, with regard to their general characteristics and safety, and to the protection and safety of vehicle occupants and vulnerable road users;

- for the type-approval of vehicles, in respect of tyre pressure monitoring systems, with regard to their safety, fuel efficiency and CO2 emissions;

- for the type-approval of newly-manufactured tyres with regard to their safety and environmental performance.

The goal of the GSR2 was to reduce the number of road incidents, improving the safety of drivers, passengers and road users. It established some mandatory ADAS on board vehicles: for example, ACC, Driver Fatigue Detector (DFD), Lane Keeping System (LKS). While the first versions focused mainly on passengers inside the vehicles, the latest version from 2024 targets Vulnerable Road Users (VRU), for which all the passive measures adopted inside the vehicle are not adequate, and it mandates the installation of several electronic safety systems on newly vehicles. Among these technologies, there are:

- Intelligent Speed Adaption (ISA) to alert the driver to speed limits, without using the engine;

- Autonomous Emergency Braking (AEB) which brakes when objects are identified;

- Driver Drowsiness and Attention Warning (DDR-AW) which alerts driver if he is detected distracted or tired;

- Emergency Stop Signal (ESS) to signal sudden braking with flashing lights;

- Lane Keeping Assist (LKA) useful to modify steering and brakes, consistently with the position of the vehicle in the lane;

- Pedestrian and Cyclist Collision Warning (PCW) to detect pedestrians and cyclists;
- Tire Pressure Monitoring System (TPMS) to monitor tire pressure;
- Reversing Detection (REV) to warn the driver of the presence of people or objects during manoeuvres;
- Alcohol Interlock Installation Facilitation (ALC) inhibits the driver to start engine if his blood alcohol exceeds a certain threshold;
- Blind Spot Information System (BLIS) to overcome blind spots;
- Event Data Recorder (black box) useful to record driving parameters to be analyzed in case of accidents.

The EU Regulation also establishes the introduction of further technologies starting from 2026, the Advanced Driver Distraction Warning (ADDW) to reduce eye movements, and starting from 2029 the Event Data Recorder (EDR), useful for dynamic analysis before and after an accident.

## 2.3 Main Technologies

ADAS use different types of technologies to observe and understand the surrounding environment and act accordingly to the received inputs (Figure 2.2) [10].

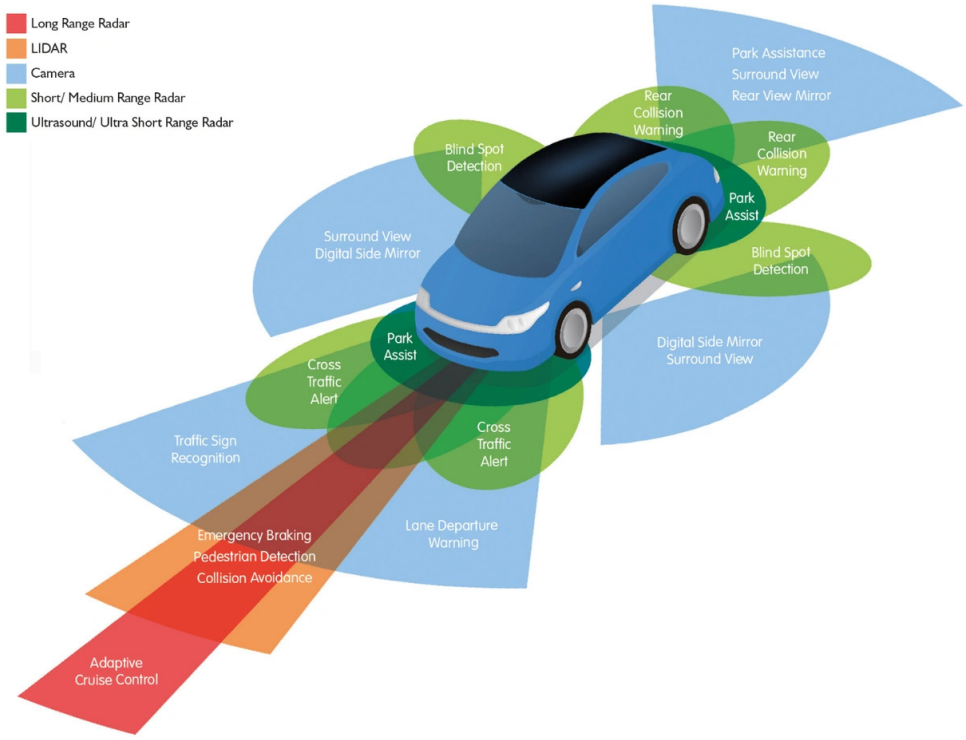



Figure 2.2: ADAS Technologies [11]

The main technologies include sensors, which are devices that detect or measure a physical property and then record, display, or otherwise respond to it. The key sensors used in ADAS are:

- Cameras: they capture images that are processed for different tasks, like Object Detection, Lane Detection and Traffic Sign Recognition. The collected information is then used to make decisions about what to do, for example, braking if a pedestrian is detected in front of the vehicle. Furthermore, cameras can be categorized based on the type of lenses they are made of:
  - Monocular Camera: it is composed of a single-lens, able to capture 2D images. It is widely used because of its simplicity and effectiveness for common tasks. It is also highly versatile and can be easily integrated with other sensors. However, the monocular camera is not able to accurately measure the distance from objects, as it does not have depth perception, and its performance degrades in extreme lighting conditions (both low-light and bright-light that reduce visibility) and adverse weather conditions;
  - Stereoscopic Camera: its peculiarity is to have two lenses positioned next to each other, in order to imitate the human binocular vision. By comparing the slight differences in object positions across the two lenses, it

can accurately measure the distance to objects;

  - 360-Degree Camera: with multiple cameras, it creates a 360° top-down (bird's eye) view surrounding the vehicle. Its main advantage is the absence of blind spots, making complex maneuvers simpler and increasing the safety of passengers and pedestrians.

- RADAR: it is an acronym that stands for Radio Detection and Ranging and it is among the most used sensors in ADAS. RADAR uses radio waves to detect the location of objects, reflecting waves back towards the RADAR. The "time of flight", that is the time necessary to the waves to travel to the object and return back to the sensor, is used to obtain an accurate measurement of the distance. Speed is determined by the Doppler effect, as the frequency of reflected waves shifts when an object is moving, allowing the sensor to calculate the object's radial velocity. The strong point of RADAR is its resistance to various weather conditions, since its working principle is independent of visibility. Among the most advanced types of RADAR, the 4D RADAR should be included: in addition to the three traditional dimensions (range, azimuth and elevation), it obtains further information on the identified objects on the scene, such as relative and absolute speed.

- LiDAR [12]: it stands for Light Detection and Ranging and it is used to measure 3D features from the environment, representing them in the form of a point cloud. The LiDAR generates laser pulses that travel from the sensor to the surrounding environment and reflect off of objects. By measuring how long laser pulses take to travel back, the LiDAR can get a precise distances. Through the creation of 3D maps of the surrounding environment, it is possible for the computer on board the vehicle to determine its position. Another relevant feature of the LiDAR is the possibility of carrying out Object Detection, with high frequency laser beams, and using this information for tasks like lane detection and obstacle avoidance, distinguishing objects into different classes (cars, pedestrians, bicycles). A distinction can be made on LiDAR types:

  - Solid-State LiDAR [13], characterized by a simple and small structure, totally built on a single chip, without rotating and moving parts that make it cheaper and more durable. However, this peculiarity does not allow the LiDAR to rotate 360 degrees, but only to have information in front by using optical emitters to send flashes of laser photons;

  - Electro-Mechanical LiDAR, composed of several moving elements in order to generate a set of laser beams. The main disadvantages of this type of LiDAR are its size and high cost.

- Ultrasonic Sensor [14]: it uses ultrasound to detect objects, by emitting an high-frequency sound waves that come back to the sensor after hitting an object. The distance between the identified object and the sensor is computed considering the time necessary to the waves to bounce back. These kind of sensors are widely used in parking scenarios, to measure and alert the driver of the eventually presence of obstacles during maneuvers. However, they are very affected by external noise that can worsen their performance and they have a limited range to detect objects.

In addition to these sensors, algorithms play a crucial role in analyzing the collected data and decisions making. A typical scenario involves identifying a hazard and taking appropriate action to avoid it [15]. For instance, algorithms are widely used for object identification and tracking, especially vehicles, by analyzing frames captured by cameras. If an object is detected in the vehicle's path, the algorithm can either issue a warning or decide on a maneuver to avoid it. Other tasks performed by these algorithms include pedestrian detection, traffic sign recognition, and driver monitoring.

# Chapter 3

# Micromobility: a New Paradigm

## 3.1 Definition of Micromobility

The term "Micromobility" refers to lightweight vehicles, characterized by low speed and small dimension. They are used both for personal purposes, being personally owned or deployed in shared fleets [16], and for industrial purpose, for example to avoid heavy and tedious work for employees.

The first criteria established for the inclusion in this category was the gross vehicle weight that must be less than 500 kilograms. Starting from 2018, the SAE International added other conditions to belong to the Micromobility regarding the top speed reachable, under 45 km/h, power source and form factor. Vehicles of this category are designed for short distances and can include a combination of any, human-powered, combustion and electric based propulsion.

The interest in the diffusion of Micromobility lies in the possibility of using a lightweight means of transport to better connect people, reducing traffic and consequently bringing benefits to the environment. Moreover, shared vehicles are widely spread thanks to their easy availability: they can be easily found and accessed with the use of a smartphone or similar devices [17].

The first idea of "on demand" Micromobility was introduced with bicycle sharing in Europe, followed by a first evolution in 1995 in Portsmouth, in which the bikes were unlocked with the use of a smart card, so it was possible to keep track of the user, and in Copenhagen with a coin-based system [18]. However, the revenues were not sufficient to cover the maintenance and repair works, also considering that some costs are decided by the city that grants the licenses for the stations to be placed. Starting from the 2000s, a revolution was brought about the "free-floating" or dockless, that allows to unlock and find the bicycles only using smartphones and GPS integrating in the vehicles. In China, many factories adopted this innovation, putting together a number of bicycles 20 times higher than the rest of the stations

in only one year. However, this strategy was not taken into consideration by either the USA or Europe, because it required adequate space in the streets to be able to park all the bikes, which could become an obstacle for pedestrians. What determined a greater diffusion of Micromobility was the addition of motorized vehicles, in particular shared “scooters” or electric stand-up or kick-scooters, which allowed startups to invest and improve the services offered.

Nowadays, the demand for Micromobility is increasingly evident in everyday life. However, its introduction has faced several challenges, primarily due to the lack of adequate bicycle infrastructure and limitations in factories, which have hindered their construction or expansion. Another challenge to overcome was the necessity to regulate these services to guarantee safety and equity. In particular, this involved establishing guidelines to manage public-use mobility, such as designating appropriate parking areas for vehicles and defining how to navigate streets without obstructing traffic.

Similarly, industrial robots have faced challenges in their implementation, including the need for precise safety standards to prevent accidents, compatibility issues with existing manufacturing systems, and the high costs associated with their integration and maintenance. Addressing these challenges is crucial to ensure their efficient and safe deployment alongside human workers.

## 3.2 Types of Vehicles for Micromobility

Among Micromobility devices there are: bicycles, e-bikes, electric scooters, electric skateboards and industrial robots.

### 3.2.1 Bicycles

The bicycle’s history reflects its evolution as a pivotal mode of transport and cultural symbol. Originating in the 1817 "Draisine," a foot-propelled vehicle by Karl von Drais, it gained traction with the introduction of high-wheeled bicycles like the Penny Farthing in the 1870s. The 1880s brought the revolutionary "safety bicycle," featuring equal-sized wheels, chain drives, and pneumatic tires, making cycling practical and accessible to the masses. By the late 19th century, bicycles symbolized freedom and social mobility, particularly for women and working-class individuals.

Despite competition from cars in the 20th century, bicycles experienced a resurgence due to environmental awareness and urban mobility needs. Modern innovations, including lightweight materials, electric models, and bike-sharing systems, continue to shape their role in sustainable transport [19] [20].

### 3.2.2 E-bikes

E-bikes can be considered as the evolution of bicycles, since their structure is the same, with the addition of an electric motor and control methods. Compared to other devices, the e-bike requires a physical effort that is opposed to scooters, but less than traditional bicycles, so that it is possible to cover longer distances, but with the benefits of an active transport. In fact, the small and efficient motor also represents a key to inclusiveness, since it allows it to be used by a wide variety of people, such as making tortuous routes easier.

First models of electric bicycles appeared already in the 1890s, followed by improvements until the introduction of the torque sensors and power controls in the late 1990s. From that moment on, the diffusion was rapid and different models were produced, that can be divided according to the control system and to the power that their electric motor can transport. The clearest classification concerns the rider's use of an pedal-assist system or power-on-demand one:

- Pedal-assist system is characterized by the regulation of electric motor through pedaling. Both the pedaling speed and force are detected with a sensor and braking can disable the motor.
- Power-on-demand system involves the activation of the motor by a throttle, that is a mechanism to control the flow of fuel, regulating vehicle speed.

These systems can be used separately or together, giving different characteristics and performances:

- Pedal-assist only, usually called 'pedelec' (pedal electric cyclic), refers to less powerful electric motor activated only if the rider is pedaling;
- Power-on-demand class includes e-bikes with a motor that is activated through an throttle;
- Power-on-demand and pedal-assist devices with Pedal Assist System (PAS) with or without throttle.

The typical motors used in electric bicycles are DC motors, that can be built in the wheel hub itself or mounted beside the bottom bracket shell. In addition to the engine, e-bikes use rechargeable batteries [21] [22].

### 3.2.3 Electric Scooter

Electric scooters are vehicles with two or three wheels and rechargeable battery. The first time that was mentioned an electric motorcycle was the October 1911 and they

Figure 3.1: E-scooter [25]

Figure 3.2: Electric Skateboards [26]

were vehicles with an average of 120 km to 160 km per charge; the first prototype was produced in 1919, but it never past the trial stage [23].

There are different types of vehicles that are included in 'scooters' group:

- Electric motorcycle;
- Electric scooter;
- E-scooter.

The e-scooters (Figure 3.1) are electric motorized scooter, stand-up and powered by an electric hub motor. They have two wheels, usually between 20 and 28 cm in diameter, positioned at the two end of a board, used by the rider to stand. The battery, that supplies power to the electric motor, is installed under the deck. The top speeds reachable are 19 to 120 km/h [24].

### 3.2.4 Electric Skateboard

The electric skateboard is a Micromobility vehicle based on skateboard, whose acceleration is controlled through a wireless hand-held throttle. The direction of the trajectory is regulated by tilting the board to the right or the left, according to which side to turn (Figure 3.2).

The first vehicle model similar to an electric skateboard was the MotoBoard in 1975, banned in some countries because of pollution and noise. The origin of such vehicle, as it is known today, dates back to the end of 1990s. However, its performance regarding torque and power board was not so efficient [27].

According to the characteristics, the electric skateboards can be divided in [28]:

- Shortboard Electric Skateboards are short and light (less than 11kg), used for shorter distances, at a quicker pace;
- Longboards Electric Skateboards have flexible and relatively long board. They are suitable for traveling around the city, at a speed that is not too high;
- Off-Road Electric Skateboards are suitable for more rough terrain due to their larger and heavier structure. The configuration of their motor is at least dual that guarantees power and speed.

The original purpose for which the electric skateboard was produced was local transportation. As the years went by, it was increasingly revolutionized, allowing to tackle longer distances, on terrains other than roads and with greater comfort. Its structure is usually composed of [29]:

- Battery, usually made of lithium-ion, provides the electricity necessary to power the device. Its capacity impacts the vehicle's performance in terms of:
  - range: the distance that can be covered before recharging;
  - top speed, the maximum velocity reachable: an higher-capacity battery allows to reach higher maximum speeds;
  - acceleration, that refers at the rate at which it can accelerate;
  - total weight of the vehicle, affected by a heavier battery; it is necessary to find the right compromise between weight and battery power;
  - recharging time, typically directly proportional to the size and power of the battery;
  - cost varies depending on the battery: more powerful ones tend to cost more than smaller capacity packs.
- Motor, that can be one or more, with different characteristics, converts electrical power into mechanical power. The motor can be hub, belt or brushless. Hub motors are located in the middle of the board, within the wheels; belt motors, typically on the bottom of the deck, produce the rotation of the wheels through a rotating belt; brushless motors involve magnetism, with a permanent magnet rotor;

- Wheels contribute to determining the characteristics of vehicles, mainly regarding the speed, the distance and the stability of the board; their rotation is made possible by ball bearings inside them;
- Board deck whose features determine the stability of the vehicles. To achieve high speed and give more stability, it is necessary to use longboard and longer wheelbase, that is the distance between the front and rear truck. Moreover, even the type of material can improve its performance;
- Electronic Speed Controller (ESC), necessary to vary the speed of the motor, manages the power drawn from the battery;
- Remote control: determines the acceleration and deceleration of the vehicles;
- Trucks are T-shaped metal parts mounted under the surface at the two ends of the board, allowing to turn;
- Brakes which cause the vehicle to slow down or stop;
- Drivetrains are different according to their use [30]:
  - Hub drive: the motor is inside the wheel, so it is subject to vibrations and shocks, and the motion is directly translated to the wheel;
  - Direct drive: the motor, inwards towards the truck and outside the wheel, directly drives a full wheel;
  - Belt drive: characterized by the connection of motors and wheels through pulleys and belts, that guarantee flexibility and easier maintenance.

The top speeds than can be reached by an electric skateboard is around 32-40 km/h, whereas the braking is performed as a dynamic or regenerative braking from the rear wheels only. The safety using electric skateboards is at the center of many debates, also considering the numerous accidents that have occurred, especially due to the lack of personal protection systems [31].

### 3.2.5 Industrial Robots

Among the Micromobility devices, industrial robots play an important role in changing human behavior at work. They can be manipulators, composed of a mechanical arm, or transportation robots, designed to move materials and products, in both autonomously or manually [32].

They are equipped with a variety of sensors, such as cameras and lasers to perceive the external environment and adapt their behavior. They can be divided into different kinds on the base of their characteristics and level of automation:

- Mobile manipulator [33] is equipped with a manipulation system and an autonomous locomotion system, and allow them to move reaching different locations and to be used for collaborative applications, automating repetitive and tedious tasks;

- Autonomous Mobile Robots [34] are designed for specific tasks that can be conducted autonomously, moving through different places and being collaborative with the human interaction. They are able to recognize the environment by integrating AI algorithms, making them able to identify objects or obstacles.

## 3.3 Advantages of Micromobility

The spread of Micromobility devices has some key advantages, first and foremost regarding health benefits, but also for the environment and, more generally, for the mobility of people.

Considering that in Europe and in Italy private cars are responsible for about 61% and 69% of all pollutant emissions due to the transportation, moving to the Micromobility can represent an important solution to reduce such pollution. Over the years, many studies have been carried out to understand whether it was possible to replace trips with private cars with Micromobility and understand any advantages, applying a systematic method in different cities. So the Micromobility can contribute in improving the sustainable mobility, for example, using shared bicycles, reducing the emissions and the utilization of non-renewable energy. A related aspect is the elimination of toxic emission from the automotive sector, since some of Micromobility modes are powered manually or using the electric ones. For example, e-bikes are zero-emissions vehicles: different studies regarding the traffic pollution found that e-bikes can reduce car emissions by up to 50%.

A direct consequence of the lesser use of private cars in favor of Micromobility methods provides for better efficiency in the management of urban areas, reducing traffic congestion and allowing faster short trips, with appropriate infrastructures. Furthermore, Micromobility spreads greater accessibility: for example it makes some travels possible for those who do not own a private car or it makes parking easier, having more dedicated areas available in cities.

These devices are also proposed as solutions for the "last-mile" concept, which refers to the last stretch to reach the final destination, often poorly connected and capable of making the entire journey more complicated.

One of the most significant advantages of the Micromobility is the health benefit: e-bikes, for example, can increase the amount of physical activity and, thanks to their accessibility, they also allow people with limited mobility to use the bike as a means

of transport, without experiencing too much fatigue, and to help them to lose weight or for cardiac rehabilitation programs.

The benefits brought by industrial robots extend beyond just improving productivity and efficiency; they also play a significant role in enhancing physical health and worker safety. By automating tasks that involve dangerous, repetitive, or physically demanding work, industrial robots help to reduce the risk of workplace injuries and long-term health problems, such as musculoskeletal disorders, which can result from lifting heavy loads or performing strenuous tasks over time.

For instance, in industries like automotive manufacturing or battery production, robots can take over tasks such as battery swapping in electric vehicles, which involves handling heavy and cumbersome batteries. This prevents human workers from having to lift or move these large, weighty components, thus reducing the physical strain and fatigue. By doing so, industrial robots not only improve the safety and well-being of workers but also contribute to maintain a more ergonomic and sustainable work environment.

In addition to reducing physical risks, robots also enhance precision and efficiency, allowing for more accurate and reliable operations that minimize human error. These improvements in both safety and productivity ultimately lead to a healthier workforce and a reduction in health-related costs for employers, while also ensuring that workers can focus on tasks that require more cognitive skills or oversight, rather than engaging in physically taxing labor.

In summary, industrial robots have revolutionized the workplace by alleviating the physical toll on workers, protecting them from dangerous tasks and improving both health outcomes and overall operational performance.

## 3.4 Integration of ADAS in Micromobility

Both Micromobility and ADAS are increasingly gaining ground in the mobility scenario, due to their countless benefits toward the environment and toward human lives. The possibility of combining ADAS with Micromobility means would consist of having mobility with more efficient lightweight vehicles, safer and more comfortable driving. By using new technologies with the growing AI and Deep Learning (DL) will help vehicle management and deployment optimization.

The need to integrate ADAS into Micromobility vehicles also arises from the numerous accidents to which drivers are subject: in European roads, there were 1901 cyclist fatalities in 2020 and 1880 in 2021; 30% of those were in car-to-cyclist rear-end crashes. The ADAS referred to the Micromobility devices are usually called Advanced Rider Assistance System (ARAS).

One of the technologies developed for these devices is a camera under the saddle


Figure 3.3: E-bike with ADAS [37]

of the bike, to record what happen around the cyclist [35] and if an obstacle is detected, a warning signal is sent to the driver on the display positioned in the handlebar area. E-bikes can be equipped with sensors and cameras also for collision avoidance and blind spot detection with warnings and speed regulation (Figure 3.3). In these scenarios, it is necessary to use Neural Networks (NN) to identify objects on the road, to be able to classify them as dangerous and to avoid them [36].

On electric motorcycles, the key sensor used is a RADAR, mounted both on front and rear side. Using these technologies, it was possible to introduce, on electric motorcycles, some of the subtasks already present on the vehicles that do not belong to the Micromobility:

- Adaptive Cruise Control (ACC) useful to maintain the correct distance to the vehicle in front, adjusting the vehicle speed;
- Blind Spot Detection (BSD) to recognize and alert the rider if there is a vehicle in areas where the view is obstructed;
- Forward Collision Warning (FCW) system that is activated, mainly with an acoustic signal, if it detects vehicles dangerously close.

Electric scooters are also the subject of studies for the evolution of ADAS, to improve safety features. Some technologies consist of locating an electric scooter, recognizing if the rider is driving on a sidewalk, through cameras, and, possibly, interrupting the rider, so as to try to solve a recurring problem in driving these vehicles. With the use of cameras and sensors, more advanced vehicles are equipped with [38]:

- Autonomous braking to brake when the system detects obstacle considered as danger;
- Intelligent speed limiter to monitor and modify the vehicle speed, according to the different situations;
- Rider alerting system to notify a danger to the rider with visual, acoustic and haptic signals;
- Object Detection, mainly to identify pedestrians, obstacles and crowd, useful to send signals to the system and avoid them.

Industrial robots leverage advanced sensors, cameras, and AI to navigate autonomously within their environment. These robots are equipped with a variety of sensing technologies, including LiDAR, RADAR, ultrasonic sensors and high-definition cameras, which allow them to detect and understand their surroundings with high precision. These sensors work together to provide real-time data about the environment, enabling the robot to perform complex tasks such as Object Detection, obstacle avoidance and path optimization.

By continuously scanning and analyzing the area around them, industrial robots can identify and classify objects, ensuring they avoid potential collisions with obstacles, machinery or workers. This ability to detect and react to environmental changes in real-time is crucial for maintaining a safe and efficient workplace. For instance, if the robot detects an obstacle in its path, it can adjust its movement to avoid the object, find an alternative route, or pause its operation until the obstacle is cleared.

Moreover, the integration of Object Detection allows robots to focus on specific targets of interest, such as parts or components in a manufacturing process and optimize their workflow. Through intelligent algorithms, robots can determine the most efficient route to complete tasks, minimizing travel time, reducing energy consumption and increasing overall productivity. In applications like assembly, packaging, or material handling, the robot can identify and interact with specific objects or parts, facilitating the automation of tasks that were previously manual or time-consuming.

## 3.5 Possibilities and Technical Challenges

The objective of the research is to allow Micromobility to change the way of using means of transport in urban scenarios, especially with the improvements made by ADAS. Among the possibilities of the evolution in the technologies mounted on such devices, there is to make them fully autonomy, at the moment only in specific and limited areas. Moreover, the AI can be the base for Advanced Rider Assistance:

systems able to analyze rider behavior in real-time, to provide personalized support, adapted to the individual's riding style and to prevent accidents, for example when the rider is recognized as distracted or tired. To monitor rider's behavior, another solution is represented by biometric sensors, in handlebars or smart helmets, monitoring heart rate, fatigue level, stress and reporting or taking control of the device if the rider's state is unsuitable for driving. However, biometric sensors have limitations: eye trackers cannot always identify the pupil in direct sunlight or with lighter color eyes or heart rate monitoring is not always accurate, due to the pavement that makes the measurement inaccurate.

It may be possible to extend collision avoidance to make it predictive, analyzing and understanding the behavior of other riders, drivers or pedestrians, and to make decisions according to what is found. A critical situation is represented, for example, by predicting predict whether a vehicle will not stop at a red light and stop personal vehicle in order to avoid a collision at an intersection.

Furthermore, these devices can be equipped with Vehicle-to-Everything (V2X) communication [39] that connects vehicles and enables data to be transmitted over them: it can be used to warn whether there are VRUs, overcoming visibility-challenges due to other vehicle or infrastructure.

Other possibilities regard vehicle self-healing, predicting maintenance with self-diagnosis, acting immediately on minor problems, before they worsen and require more onerous and more expensive interventions. For example, these technologies can act to recognize problems and sending and alarm to the rider to make him aware.

In addition to Micromobility, another area of great potential lies in the use of industrial robots, which share many of the same technologies and principles. Industrial robots rely heavily on advanced sensors, AI, and real-time decision-making to perform tasks autonomously. These robots are used in industries, including manufacturing, logistics, and healthcare, where they can perform repetitive, dangerous, or labor-intensive tasks with high precision and efficiency.

Industrial robots can take over hazardous tasks, such as handling toxic substances or lifting heavy objects, reducing the risk of injury for human workers. By automating these processes, robots can work around the clock, improving productivity and consistency in production lines. Industrial robots can also perform tasks that require high precision, such as assembly, welding, or quality inspection, ensuring that products meet exact specifications.

All these possibilities inevitably lead to challenges, which add to the difficulties inherent in the use of Micromobility vehicles. Among the problems raised in the everyday use of the devices, the following ones are found, with some proposed solutions [40]:

- Adverse weather conditions may prevent the use of vehicles. An innovation in

their design can better protect riders;

- Safety, that can be improved with the integration of ADAS;
- Acts of vandalism that damage Micromobility shared vehicles, even irreparably. Different solutions are proposed to avoid this: for example, some companies uses cable locks or lock devices in case of non-payment [41];
- Infrastructures that are often unsuitable for the circulation of such vehicles.

However, the introduction of new technologies faces with different technical challenges [42]:

- Compliance with changing government norms, regarding limit speeds and restricted parking areas. These regulations require tracking vehicle metrics with sensors and GPS trackers;
- Data sharing with traffic authorities to monitor movement and any fines. The difficulty lies in sharing only information relating to driving the vehicle, leaving out personal information; furthermore, this information often comes from mobile applications, so it can be subject to incorrect or inaccurate data;
- Rider behavior is the main problem to address as it is often difficult to predict and therefore to solve. It is in fact difficult to find a common pattern to all riders, however new technologies aim to generalize them;
- High Initial Cost, referring to the upfront cost of implementing industrial robots that can be high;
- Integration with Existing Systems, which can be complex and time-consuming, particularly in industries with established processes.

## 3.6 Case Studies

### 3.6.1 Honda Cooperative Intelligence

Among the examples in using Micromobility devices, HONDA Cooperative Intelligence (CI) can be mentioned [43]. One of the objective of the factory is to consent the mutual interaction and understanding between machines and people, in a collision-free way. With a sight in the future, where the concept of "mobility" would be different from today, they develop new technologies: CiKoMa vehicle and WAPOCHI robot (Figure 3.4). The former is a ride-in Micromobility vehicle, used to transport people autonomously, while the latter is a robot linked to an user and

walking with it, for example carrying bags. The CiKoMa understands road structure by using cameras and recognizing objects, like pedestrians or vehicles, in order to plan and decide the trajectory to perform. It uses stereocameras to generate 3D point clouds, exploiting the parallax of the two lenses, to define a passable area in reaching a destination. WaPOCHI technology is based on multiple cameras that provide a 360° of its surroundings, used with AI to extract and track the user's peculiarities.

Figure 3.4: Honda CI Micromobility technologies [43]

### 3.6.2 ALBA Robot

Another case study is represented by ALBA Robot [44], Micromobility platform with the aim of making autonomous the transport of people with reduced mobility, with a focus in hospitals, museums and airports by using a wheelchair (Figure 3.5).


Figure 3.5: ALBA Robot [44]

The SEDIA project (SEat Designed for Intelligent Autonomy) focuses on developing an Autonomous Vehicle with smart, remote driving capabilities. It features indoor localization and precise obstacle avoidance, using a combination of sensors (two RealSense cameras, Time of Flight and ultrasound sensors) and advanced algorithms to ensure safety. The vehicle is designed to navigate narrow spaces effectively by leveraging the ToF and ultrasound technologies.

# Chapter 4

# Materials and Methods

The following chapter outlines the materials and methods used in this work. Using an industrial rover, cutting-edge technologies, perception sensors and real-time frameworks were explored, with the aim of studying technologies and algorithms necessary to create a Perception Pipeline for Object Detection to be integrated into urban vehicles, with full safety and flexibility.

The Materials section is divided into Hardware, describing the selected technologies and their key characteristics, and Software, outlining the tools and frameworks used in the project.

Finally, in the Methods section the development of the pipeline is outlined in two parts: the first is a comparative analysis of two Neural Network models, in order to choose the best trade-off model between accuracy and computational resources; the second is the description of the code development process, from the capture of raw data received from the sensors to Data Fusion, passing through the detection of obstacles.

## 4.1 Materials

The Materials section has been structured into two distinct parts to provide a clear and detailed overview of the resources used in this work. The first part, Hardware, focuses on the physical technologies employed, offering an in-depth description of their key features and the reasons behind their selection based on the specific needs of the project. The second part, Software, delves into the digital tools and frameworks adopted, explaining their functions, how they were applied within the workflow, and their contribution to achieving the project's goals.

## 4.2 Hardware

The following paragraphs provide a description of the hardware used in this project. Specifically, the system is based on the NVIDIA Jetson AGX Orin, a high-performance computing platform designed for AI applications. It also incorporates a stereo-camera, along with a 4D RADAR for accurate environmental sensing and Object Detection.

### 4.2.1 NVIDIA Jetson AGX Orin

The NVIDIA Jetson modules are flexible end-to-end hardware solutions to combine power and performance, able to satisfy different and complex tasks [45]. One of the strongest points of all NVIDIA Jetson modules and developer kits is the NVIDIA Jetson software stack: NVIDIA Jetpack SDK [46]. It provides Jetson Linux, developer tools and CUDA-X accelerated libraries, that are useful to accelerate AI applications, AI inferencing with NVIDIA Tensor Runtime (TensorRT) and cuDNN, general computing using CUDA and Computer Vision and Image Processing with Vision Programming Interface (VPI). CUDA Toolkit supplies a development environment for GPU-accelerated applications, based on GPU-accelerated libraries, a C/C++ compiler, runtime library and tools for debugging [47]. TensorRT, built on CUDA, is used for DL inference runtime for Object Detection, Image Classification and Segmentation. VPI is a software library, including algorithms for Computer Vision and Image Processing. It leads to an increase in throughput since provides zero-copy memory mapping and to more portability with different applications, both in Python and C/C++ [48].

The NVIDIA Jetson platforms represent the solution for the introduction of AI complex systems at the edge. Among the members of the Jetson family, the NVIDIA Jetson AGX Orin series (64GB module, Figure 4.1) has been chosen for this work (4.1), since its improved performance in AI applications [49]: compared to the Jetson AGX Xavier, it is able to obtain 8 times its performance (4.2).

Jetson AGX Orin modules are composed of an integrated Ampere GPU, divided into two Graphic Processing Clusters (GPCs), up to eight Texture Processing Clusters (TPCs), up to 16 Streaming Multiprocessors (SMs). The Tensor cores have been improved compared to the previous generation, providing acceleration for AI applications. Together with Ampere GPU, Tensor cores allow to support sparsity in Deep Learning Networks, doubling throughput and reducing memory usage. The aforementioned NVIDIA TensorRT library can be used by the customers to accelerate the inference process, optimizing the use of GPU memory and bandwidth, working with fusion of nodes in a kernel. The performance of Deep Learning opera-

Figure 4.1: Jetson AGX Orin [49]


Figure 4.2: Jetson AGX Orin delivers 8× the AI performance of Jetson AGX Xavier [49]

| | |
|---|---|
| AI Performance | 275 TOPS (INT8) |
| GPU | NVIDIA Ampere architecture with 2048 NVIDIA CUDA cores and 64 Tensor Cores |
| Max GPU Freq | 1.3 GHz |
| CPU | 12-core Arm Cortex-A78AE v8.2 64-bit CPU; 3MB L2 + 6MB L3 |
| CPU Max Freq | 2.2 GHz |
| DL Accelerator | 2x NVDLA v2.0 |
| DLA Max Frequency | 1.6 GHz |
| Vision Accelerator | PVA v2.0 |
| Memory | 64GB 256-bit LPDDR5 204.8 GB/s |
| Storage | 64GB eMMC 5.1 |
| CSI Camera | Up to 6 cameras (16 via virtual channels) 16 lanes MIPI CSI-2 |
| Video Encode | 2x 4K60 |
| Video Decode | 1x 8K30 |
| Other I/O | 4x USB 2.0; 4x UART, 3x SPI, 4x I2S, 8x I2C, 2x CAN, DMIC & DSPK, GPIOs |
| Power | 15W - 60W |

Table 4.1: Jetson AGX Orin Series technical specifications

tions is further improved by the NVIDIA Deep Learning Accelerator (DLA), which provides full hardware acceleration of Convolutional Neural Network inference.

The enhancements in both the GPU and DLA elevate the performance of the Jetson AGX Orin, enabling the use of even more complex models in AI field, with multi-sensor perception, mapping and localization. The improved capabilities of the Jetson AGX Orin are also attributed to its CPU, the Cortex-A78AE, which features up to twelve CPU cores. A major advantage of the Jetson AGX Orin is its ability to support a wide range of devices, including the key sensors commonly used in ADAS, making it ideal for the perception task required in this work.

### 4.2.2 Perception Sensors

In Autonomous Vehicles, 'perception' refers to the process of analyzing and interpreting data from sensors, with the purpose of identify, detect, classify and track objects. The perception sensors allow the vehicle to understand its surroundings, by providing crucial information about the physical world, which is useful for decision-making and control.

#### Stereocamera

Stereoscopic vision is the process of obtaining depth information from a pair of images of the same scene, captured by two different cameras. A stereoscopic camera

(stereocamera) applies the same concept, using two different lenses located at a distance similar to that of human eyes.

The process relies on the geometric analysis of the two images and the two lenses can either have parallel optical axes or can be positioned in a more general arrangement (Figure 4.3) [50].

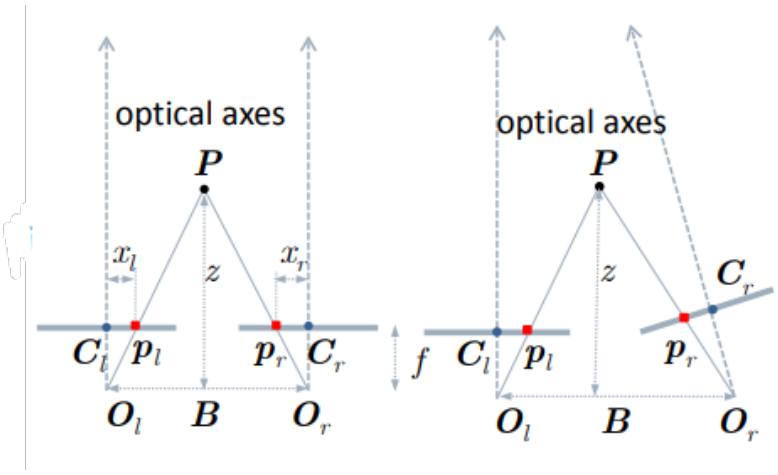



Figure 4.3: Optical axes [50]

In order to better analyze the image formation of frames taken from two different cameras, a brief overview of the basic perspective formation model of a single image is provided.

The 3D object point $P$ is denoted by $\mathbf{X} = (X, Y, Z)$, image 2D points by $p_i$ $\boldsymbol{x} = (x, y)$. The goal is therefore to determine the geometric relationships between the coordinates of a point in 3D space and the coordinates of the corresponding point in 2D in the image, expressed in pixels.

In general, there will be 4 reference systems:

- The world system $\boldsymbol{R}_0$ (inertial, fixed);
- A system attached to the camera $\boldsymbol{R}_c$;
- A system on the image plane $\boldsymbol{R}_i$;
- A system on the image plane in pixels $\boldsymbol{R}_{pix}$.

The transition from $\boldsymbol{R}_0$ to $\boldsymbol{R}_c$ occurs by using the corresponding rotation-translation matrix between the two reference systems.

The perspective projection model adopted for the transition from $\boldsymbol{R}_c$ to $\boldsymbol{R}_i$ coincides with the central projection model, which involves the following simplifying assumptions in the image formation geometry:

- The image plane coincides with the focal plane;

- To avoid the inversion of the image (which would be upside down), the front focal plane is considered as the image plane.

The $\boldsymbol{R}_c$ system (right-handed) is defined as follows:

- The origin coincides with the center of the lens;
- The z-axis is directed along the optical axis.

The image plane is located at a distance $f$ from the origin of $\boldsymbol{R}_c$ along the $z_c$ axis.

The $\boldsymbol{R}_i$ system has its origin at the intersection point of the optical axis with the image plane, and the $x_i$ and $y_i$ axes are parallel to the respective axes of $\boldsymbol{R}_c$ (Figure 4.4).

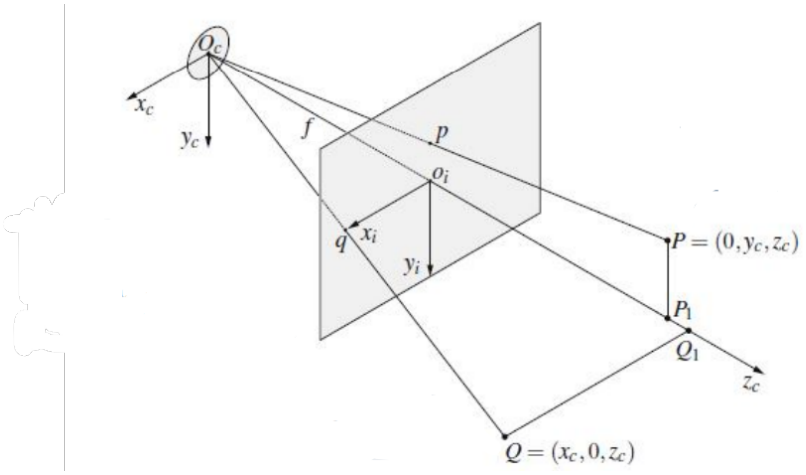



Figure 4.4: Reference System [50]

From the similarity of the triangles $P - P_1 - O_c$ and $p - o_i - O_c$, it is obtained:

$$y_i = f\frac{y_c}{z_c}$$

From the similarity of the triangles $Q - Q_1 - O_c$ and $q - o_i - O_c$, it is obtained:

$$x_i = f\frac{x_c}{z_c}$$

These relationships can be expressed in the form of a matrix equation for the projection as:

$$s\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

where $s$ is a scaling factor.

The coordinates $x_i$,$y_i$, and the scaling factor $s$ can be uniquely determined from the knowledge of $x_c$, $y_c$, and $z_c$, referring to forward projection.

On the other hand, it is not possible to uniquely determine the coordinates $x_c$, $y_c$, and $z_c$ in the camera's reference system if only $x_i$ and $y_i$ are known but not the scaling factor $s$. In this case, it is known as backward projection, for which a unique solution cannot be computed with only one camera.

Forward projection is completed by calculating the position on the image plane itself, expressed in pixels. The transformation from image coordinates $(x_i, y_i)$ to index coordinates $(u, v)$ is carried out under the assumption that the digital image was obtained as output from the image sensor (through an A/D converter), where each pixel corresponds to a particular element of the sensor, which is rectangular in shape:

- $D_x$ and $D_y$ are the dimensions of the rectangle corresponding to a single pixel;
- The origin of $R_i$ is at the pixel coordinates $(u_0, v_0)$.

The following relationships are obtained:

$$\frac{x_i}{D_x} = u - u_0$$

$$\frac{y_i}{D_y} = v - v_0$$

Finally, the coordinates in pixels are derived as:

$$u = u_0 + \frac{x_i}{D_x}$$
$$v = v_0 + \frac{y_i}{D_y}$$

The obtained transformation can be written in matrix form as:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{D_x} & 0 & u_0 \\ 0 & \frac{1}{D_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

The overall transformation is then given by:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{D_x} & 0 & u_0 \\ 0 & \frac{1}{D_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{D_x} & 0 & u_0 & 0 \\ 0 & \frac{f}{D_y} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

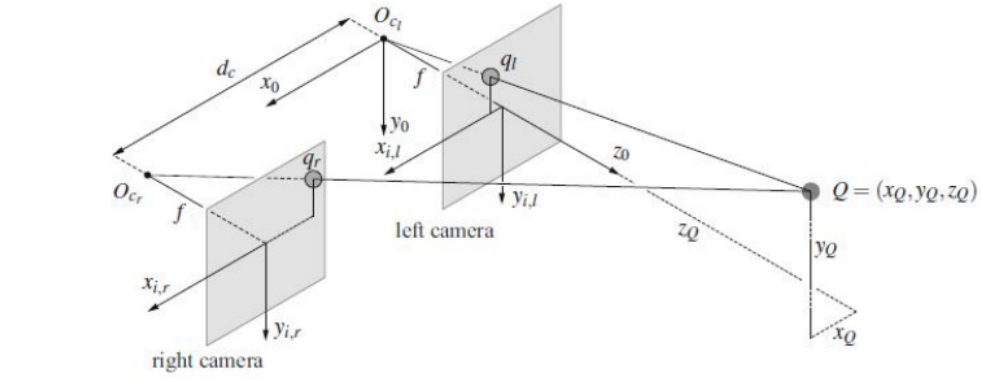



Figure 4.5: Backward projection with stereocamera [50]

The matrix found, which expresses the transformation from the camera's reference system coordinates to the image coordinates in pixels, represents the perspective projection and can be rewritten as:

$$\mathbf{P} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

with $f_x = \frac{f}{D_x}$, $f_y = \frac{f}{D_y}$ and $\mathbf{P}$= $[\mathbf{K}_{3\times3}|0]$, where $\mathbf{K}$ is the camera calibration matrix.

The backward projection problem has no solution using just one camera. However, in general, it is possible to reconstruct the structure of any objects by utilizing stereoscopic vision, which is the process of obtaining depth information from a pair of images taken by two cameras looking at the same scene simultaneously from different relative positions at a fixed distance.

In backward projection process, w+hen computing the coordinates of a generic point $Q = (x_Q, y_Q, z_Q)$ in a fixed system, the starting point is the respective points $q_r$ and $q_l$ in the images of the two lenses (Figure 4.5). The axes of the reference system are jointly with the two cameras, they have the same directions and their optical axes are parallel. By observing the projections from above (Figure 4.6), it is obtained:

$$\frac{z_Q}{f} = \frac{x_Q}{x_{i,l}}, \frac{z_Q}{f} = \frac{x_Q - d_c}{x_{i,r}}$$

where $d_c$ is the distance between the origins of the two planes, $O_{c,l}$ and $O_{c,r}$. And then:

$$z_Q = \frac{f d_c}{x_{i,l} - x_{i,r}}, x_Q = \frac{x_{i,l}}{f} z_Q$$

referring to $x_{i,l}$ and $x_{i,r}$ as the x-axes of the two reference systems. From the lateral

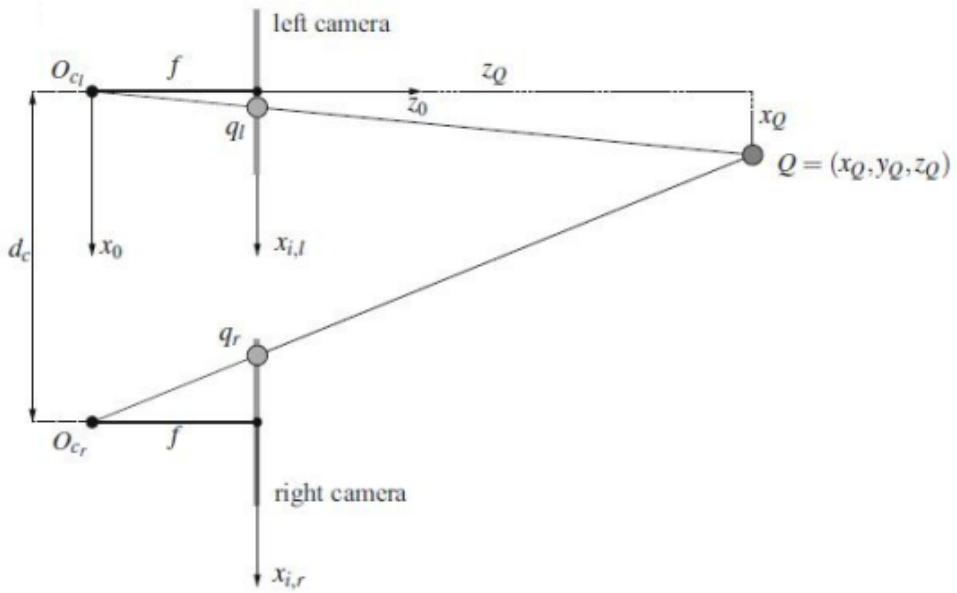



Figure 4.6: Backward projection view from above [50]

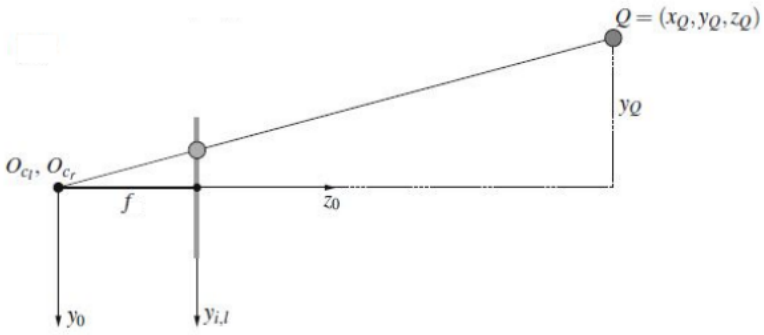



Figure 4.7: Backward projection view from the side [50]

observation (Figure 4.7), it is obtained:

$$\frac{z_Q}{f} = \frac{y_Q}{y_{i,l}}$$

And then:

$$y_Q = \frac{y_{i,l}}{f} z_Q$$

Stereoscopic vision involves several challenges, including correspondence and 3D reconstruction. Correspondence refers to identifying points in the two images, where the 2D pixel coordinates represent the same 3D points. 3D reconstruction involves using disparity to generate a depth map.

Stereocameras are widely used in Computer Vision applications, to perform Object Detection, distance measurements and 3D mapping, making them valuable tools in the development of Autonomous Vehicles.

When choosing a suitable camera for the work, certain characteristics are essen-

tial to optimize sensors usage for the specific task and platforms used. Among the connection protocols adopted by the cameras, the Gigabit Multimedia Serial Link 2 (GMSL2) [51] guarantees high-speed data transfer with low latency, tailored for Automotive and Computer Vision applications. GMLS2 is based on packet, full duplex bidirectional architecture and allows long distance transmissions. Another significant aspect of GMSL2 is its use of SerDes (Serializer and Deserializer), a functional block designed for high-speed chip-to-chip communication that reduces the number of I/O interconnects. It ensures an higher distance of transmission, up to tens of meters, and higher data rates in terms of bandwidth. GMSL2 increases reliability and redundancy by minimizing the effects of external electromagnetic interference through spread-spectrum technology. What further guarantees the widespread adoption of GMLS2 is its compatibility with a variety of sensors and devices, making it well-suited for Automotive systems that require the the simultaneous management of multiple sensors.

Additionally, the type of connector affects camera performance. The FAKRA connector, shown in Figure 4.8, is specifically designed to meet the demands of the automotive industry, enhancing performance in terms of frequency and safety. It enables the transmission of large data volumes from multiple cameras in real time. Thanks to its push-on mechanism and secure plug-and-jack, the FAKRA connector provides greater reliability, preventing cable disconnection during motion.

Figure 4.8: FAKRA connector [52]

The camera selected for this work is Leopard Imaging's 3D stereocamera, specifically the Leopard Imaging Hawk 3D Depth Camera [53], depicted in Figure 4.9.

Figure 4.9: Leopard Imaging Hawk Camera [53]

One of the key reasons behind this choice is their improved resolution, which refers to the amount of detail the camera can capture in an image. Additionally, Leopard Imaging collaborates with NVIDIA to provide cost-effective 3D stereo imaging solutions, combining high quality with low latency on such platforms. Finally, the camera's connection protocol is based on GMSL2 and it utilizes FAKRA connectors, both of which optimize its behavior in Automotive applications.

The environment perception is facilitated by the 121.5° horizontal and 147.5° diagonal FOV and by the two Global shutters that capture the entire frame simultaneously. Additionally, the camera's high resistance to water, dust, and humidity makes it suitable for both outdoor and indoor applications (Table 4.2).

| | |
|---|---|
| Use environment | Indoor/Outdoor |
| Baseline | 150mm |
| Video Output | 1200P @ 60 fps with output resolution side-by-side 2 × (1920×1200) |
| Power Supply Range | 9 ∼ 19 VDC |
| Inertial Measurement Unit (IMU) | BMI088 |
| Serializer | Maxim GMSL2 |
| Optical Format | 1/2.6" |
| Resolution | 1920 (H) × 1200 (V) (active pixels) |
| Output Format | 10-bit RAW |
| Color / Mono | Color |
| Distance range | 1.0 ∼ 8.0 m |
| Shutter | Global |
| Interface | FAKRA |

Table 4.2: Hawk stereocamera technical specification

### 4D RADAR

When selecting a RADAR sensor, several key features should be taken into consideration to ensure optimal performance and compatibility with specific needs:

- Frequency range: higher frequencies generally provide better resolution and accuracy, especially in detecting small objects;
- Detection range: it refers to the maximum and minimum distances the sensor can detect;
- Resolution: it is sensor's ability to distinguish between closely spaced objects;
- Field of View (FOV): it indicates how wide the sensor can perceive;
- Object Detection capabilities: some RADAR sensors can classify detected objects;
- Latency: it evaluates the time required for the sensor to process and transmit data;
- Data output format: it takes into account the data format provided by the RADAR and its compatibility with existing systems of frameworks, such as ROS or CAN bus.

Considering these characteristics, the selected RADAR for this project is the Continental ARS 548 RDI, a 4D RADAR (Figure 4.10) that stands out from traditional 2D or 3D RADAR systems by adding an extra layer of information: it not only detects the range, velocity and angle of objects but also determines their height. Moreover, it can identify objects in a wide range and differentiate between small, medium and large objects, even in difficult weather conditions. The 4D capability enhances Object Detection, allowing classification as cars, pedestrians or cyclists. This 4D RADAR is versatile, suitable for both highway driving and urban environments, as it provides effective long-range and short-range detection (Table 4.3).

Figure 4.10: Continental 4D RADAR [54]

The ARS 548 RDI sensor independently measures the distance, speed (Doppler's principle) and angle of objects in a single measurement cycle. This is achieved through pulse compression with a new frequency modulation that avoids the limitations of both classical Pulse-Doppler and Frequency Modulated Continuous Wave (FMCW). As a result, it provides a better overall Signal to Noise Ratio (SNR) and allows for easy separation of range, velocity and angular information in the received signals (Figure 4.11).

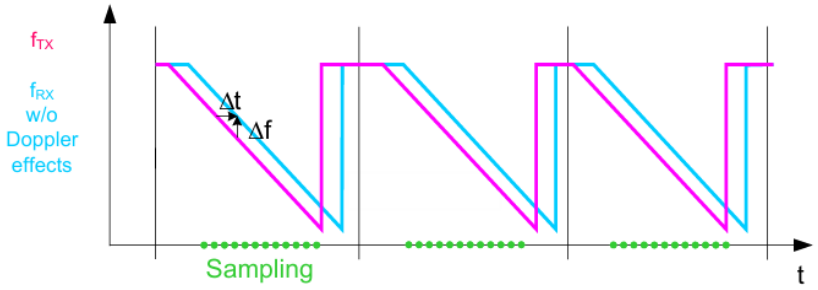



Figure 4.11: RADAR principle [55]

A key advantage of this RADAR principle is that it is resistant to interference from other electromagnetic emissions, preventing the appearance of ghost targets. The antenna system, illustrated in Figure 4.12, utilizes digital beamforming. The example consists of one transmitting antenna (TX) and 4 receiving antennas (RX), which pick up reflections from the same target. These reflections do not differ in amplitude, but vary in phase due to the slightly different distances from each receiving antenna to the target, depending on the azimuth angle $\alpha_{AZ}$.

| Measuring performance | To natural targets |
|---|---|
| Distance range | 0.2 - 301 m |
| Azimuth angle augmentation | ±60° |
| Elevation angle augmentation | ±4° ... ±14° |
| Antenna channels | 16 × RX and 12 × TX = 192 virtual antennas |

Table 4.3: Continental 4D RADAR ARS 548 RDI

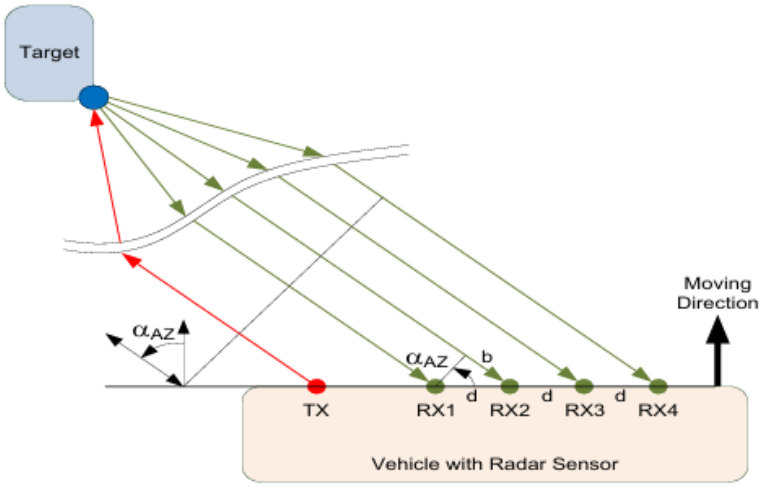



Figure 4.12: Antenna principle [55]

The relationship between phase and angle can be expressed as follows:

$$\alpha_{AZ} = \arcsin \frac{b}{d}$$

using

$$b = (\psi_{RX1} - \psi_{RX2}) \cdot \frac{\lambda}{2\pi}$$

with $\psi_{RXn}$ = received phase of $RX$ beam n, $d$ is the horizontal distance between each pair of the receiving antennas $RXn$ and $\lambda$ is the wavelength of the received signal. The wavelength is the physical distance between two consecutive peaks of a wave and depends on the signal frequency: $\lambda = \frac{c}{f}$ where $c$ is the speed of light and $f$ is the signal frequency.

The ARS 548 RDI sensors employs 12 TX and 16 RX antennas, giving 192 virtual antenna channels. By using complex beamforming on the received RADAR data, the sensor is able to accurately determine the four detection dimensions: range, speed, azimuth and elevation angle.

The RADAR sensor outputs data via Automotive Ethernet, specifically using the BroadR-Reach (100BASE-T1) protocol, which is designed for high-speed communication in automotive environment [56]. BroadR-Reach enables data transmission over a single twisted pair cable, reducing weight and complexity while maintain-

ing reliability and robustness against interference. This is essential for the high-bandwidth, low-latency requirements of modern automotive senors. BroadR-Reach operates at 100 Mbps (Fast Ethernet speeds), sufficient for transmitting data from sensors.

To enable compatibility with standard Ethernet analysis tools, the Technica Engineering 100BASE-T1 MediaConverter_NXP (shown in Figure 4.13) is used to bridge Automotive Ethernet to conventional Ethernet.


Figure 4.13: 100BASE-T1 MediaConverter_NXP [57]

This media converter translates data frames from the 100BASE-T1 protocol to 100BASE-TX (Fast Ethernet) [57].

The conversion process occurs at the physical layer, ensuring that data frames are transmitted with minimal modification and preserving the integrity of the original signal. The Technica Engineering MediaConverter introduces a constant delay of $2.0\mu s$, ensuring that time-sensitive data retains its precision during the transition from the Automotive to the Standard Ethernet domain. This low latency is crucial for real-time applications, where even small delays can affect the performance of ADAS and other critical systems.

## 4.3 Software

### 4.3.1 ROS2 Framework

The Robot Operating System (ROS) is a set of software libraries and tools for building robot applications. From drivers to state-of-the-art algorithms, ROS has the necessary open-source tools for developing robotic projects. ROS2 [58] is a middleware based on an anonymous publisher-subscriber mechanism that allows message passing between different ROS processes (Figure 4.14). At the core of any ROS2

system is the ROS graph, which is the network of nodes in a ROS system and the connections between them through which they communicate. ROS client libraries allow communication between nodes written in different programming languages. There is a core ROS Client Library (RCL) that implements common functionality needed by ROS APIs of different languages. This makes language-specific client libraries easier to write and have more consistent behavior. There are two main client libraries:

- rclcpp: library for C++;
- rclpy: library for Python.


Figure 4.14: ROS2 Humble [58]

### Nodes

In ROS2, a node is an element of the ROS graph that uses a ROS client library to communicate with other nodes. Nodes can be in the same process, in different processes, or on different machines, and connections between them are established through a distributed discovery process. Each node in ROS should be responsible for a single purpose, for example: a node for controlling wheel motors, a node for reading data from cameras, etc. Communication between nodes occurs via the publisher-subscriber model, i.e., each node can send and receive messages from other nodes by publishing and subscribing to topics. Finally, a node can also provide and use services and actions. A communication example is shown in Figure 4.15.

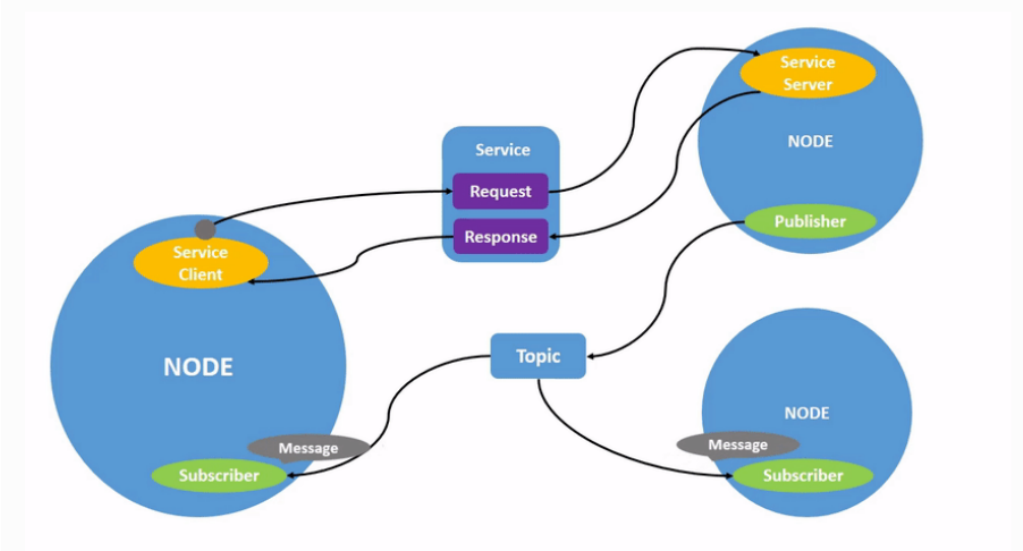



Figure 4.15: Communication between ROS2 nodes [59]

### Topics

ROS2 breaks down complex systems into many modular nodes. Topics are a vital element of the ROS graph that acts as a bus for the nodes to exchange messages and are one of the main ways in which communications occur between different parts of the system. A node can publish data to any number of topics and simultaneously subscribe to any number of topics.

### Services

Services are another communication method for ROS graph nodes. They are based on a call-and-response model, unlike nodes that use a publisher-subscriber model. There can be many clients requesting and using a service, but there can be only one server providing the requested service. Unlike topics that allow nodes to subscribe to data streams and get continuous updates, services provide data only when specifically requested by a client.

### Parameters

A parameter is a configuration value of a node. A node can store parameters such as integers, floats, booleans, strings, and lists. It is possible to configure parameter values from the command line, or to save parameter settings to a file for reloading in a future session.

### Actions

Actions are one of the types of communication in ROS2 and are intended for long-lasting tasks. Their functionality is similar to services, except that actions can be undone. They also provide constant feedback, as opposed to services that return a single response. They are based on topics and services and consist of three parts: an objective, feedback and a result. An action client node sends a goal to a action server node which recognizes the goal and returns a feedback stream and a result (Figure 4.16).

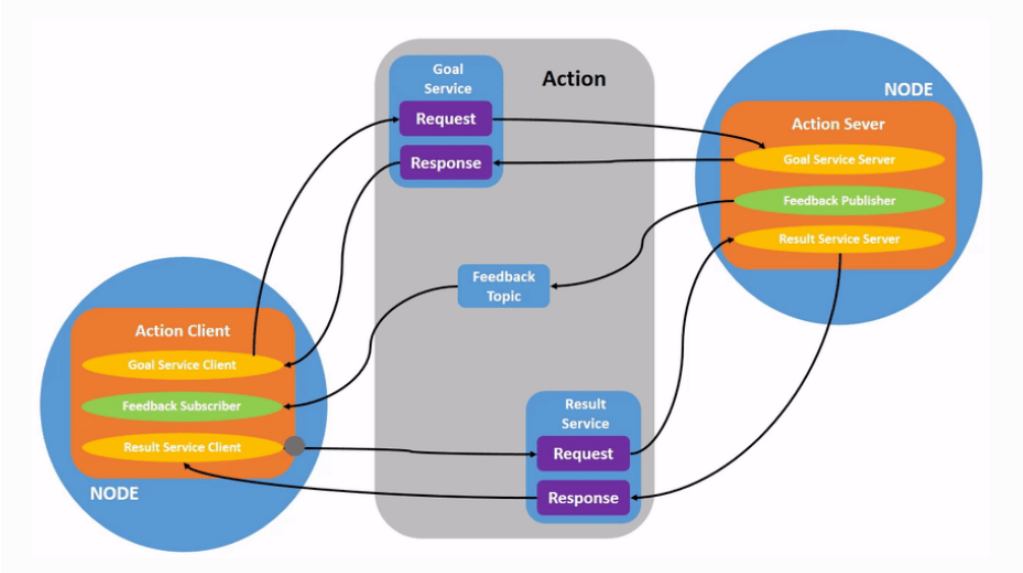



Figure 4.16: ROS2 actions [60]

### RViz

RViz is a data visualization tool that, by subscribing to topics, displays the data it receives based on the type of message (e.g., LaserScan). In Figure 4.17, it can be seen the three axes representing the origin of the sensor and the red semicircle representing the data coming from the laser.

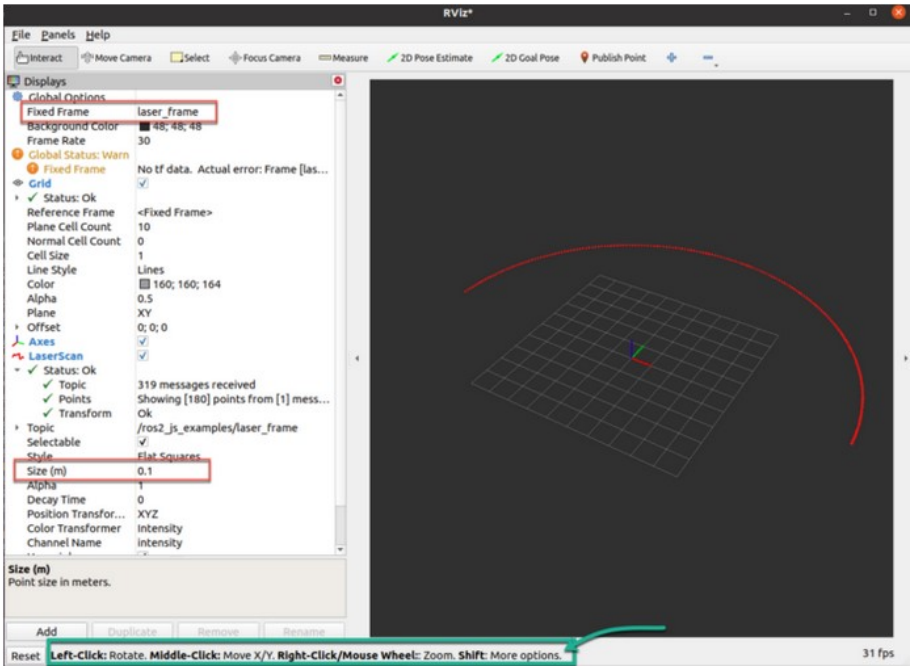



Figure 4.17: RViz capture

### 4.3.2 NVIDIA Isaac ROS

NVIDIA Isaac ROS [61], directly developed by NVIDIA, is a collection of ROS2 packages optimized for Jetson and other NVIDIA platforms, including AI models, leveraging the power of the boards themselves.


Figure 4.18: NVIDIA Isaac ROS [61]

The NVIDIA Isaac ROS collection is built on ROS2 Humble framework, that introduces new hardware acceleration features aimed at increasing performance, especially in AI and Computer Vision tasks. Among the hardware acceleration features implemented, it includes type adaptation and type negotiation.

Type adaptation (REP-2007) [62] is used to optimize ROS node communication by working wirh formats better suited to the underlying hardware. Its main priority

is to clearly communicate the described feature and the order of custom type and ROS type arguments. If a node uses an adapted type, it can publish and receive the adapted type, offering functions to convert between the adapted type and standard ROS types, such as sensor_msgs or geometry_msgs, to the adapted type and vice-versa.

Type negotiation (REP-2009) [63] allows ROS nodes to process graphs and notify of new supported message types. The messages sent over topics can be classified into ROS Message Type and ROS Adapted Type. The latter are custom types defined through type adaptation, converted to and from ROS Message types before transmission. Both regular publisher and subscriber can be created using either ROS Message Types or a ROS Adapted Types. However, negotiating publisher and subscriber are components with preferences for the message types they support. These publishers and subscribers require a list of supported messages types and a dedicated topic to negotiate the selected message types. Type negotiation allows nodes to use the same publisher to handle different message types at the same time. Moreover, it activates only the necessary publishers and subscribers, enabling them to wait if additional information is needed to express their preferences.

One of the most widely used packages in the NVIDIA Isaac ROS repository, which significantly boost task performance, is the Isaac ROS NITROS [64]: it offers an alternative and more efficient method for exchanging messages between nodes, without using the traditional ROS2 communication, eliminating unnecessary GPU-to-CPU copies, reducing software and CPU overhead and saving precious computational resources.

For managing camera devices, NVIDIA has developed the Isaac ROS Argus Camera [65] module based on the Libargus library Camera API [66]. This package processes sensor input to produce images using dedicated hardware engines, accelerating the process and utilizing the full memory bandwidth of the Jetson platforms. It uses Gigabit Multimedia Serial Link (GMSL) or Camera Serial Interface (CSI) to deliver raw data directly to GPU-accelerated memory, while the ISP hardware processes them into a GPU-accelerated output image topic. The main advantage of this module is its ability to process sensor data into ROS2 output topics without the CPU handling a single pixel.

Isaac ROS Argus Camera supports several features for sensor capture and processing, including Auto-White-Balance (AWB), Noise Reduction and Auto-Exposure (AE).

As shown in Figure 4.19, a typical camera driver requires two CPU memory copies for each pixel: the first from the I/O interface with CPU, to make the image accessible to other applications, and the second from the driver to publish the image in ROS.

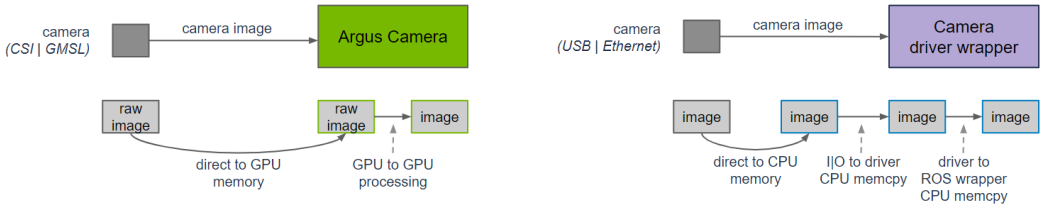



Figure 4.19: Differences between Argus Camera and a generic Camera driver wrapper [65]

## 4.4 Methods

This section outlines the methodology used for the development of the Perception Pipeline on Micromobility vehicle. The system architecture consists of an industrial robot equipped with an NVIDIA Jetson AGX Orin, a stereocamera, and a 4D RADAR sensor. The detection approach for VRUs is first described, followed by the system setup, detailing the connections between sensors and the configuration of the software environment. The pipeline for the stereocamera is explained, covering Image Processing and Object Detection. The integration of the 4D radar sensor and the Data Fusion process, which combines information from both sensors to improve detection, is also discussed.

### 4.4.1 General Architecture

The general architecture consists of a rover, which serves as the base where the sensors are mounted. The core component is the NVIDIA Jetson AGX Orin platform, which is the central processing unit for the system and is connected to various sensors that provide the necessary data for navigation and perception.

A 4D RADAR sensor is mounted on the front of the rover, facilitating forward-facing Object Detection and range estimation. Positioned directly above the RADAR is the stereocamera, which contributes to enhanced perception. This arrangement ensures that both sensors maintain an optimal FOV for detecting obstacles and collecting environmental data.

The 4D RADAR sensor is connected to the Orin via Ethernet, providing high-resolution data for robust environmental sensing. Additionally, the stereocamera, which utilizes GMLS2, is directly linked to the system.

Together, the RADAR and the stereocamera enable the system to achieve precise Obstacle Detection and situational awareness, leveraging the computational power of the Jetson AGX Orin.

### 4.4.2 Vulnerable Road Users Detection

When evaluating the performance of Autonomous Vehicles, it is essential to consider the need for real-time operations, as these vehicles must make decisions based on the data they perceive from their environment. Any delay in receiving this information inevitably results in delays in both control decisions and planning. Moreover, the computing platforms used in such vehicles have a fixed computational budget, typically measured in terms of heterogeneous processing power (TOPS) and throughput.

One of the most challenging tasks of Object Detection is the accurate identification of VRUs, a category that includes cyclists, pedestrians and other at-risk individuals in road environment. These users are particularly difficult to detect due to their smaller size compared to other road vehicles, but improving their recognition is critical for reducing traffic-related fatalities.

To accomplish this task, it is used a combination of stereocamera from Leopard Imaging and 4D RADAR from Continental. The objective is to maximize the performance of each sensor, leveraging on their key points, and reduce their weak points by combining different kind of sensors.

The next part of this work focuses on an analysis of the State-of-Art of Neural Networks models for Object Detection in urban scenarios that optimize the ratio between accuracy and computational resources on ARM architectures, which are commonly used in Autonomous Vehicle platforms.

#### Neural Networks Comparison

Among the examined papers, the work presented in [67] and [68], a solution is proposed for real-time pedestrians and priority sign detection, by using two cameras, on NVIDIA Jetson platform. The Object Detection was executed with a lightweight, custom trained Convolutional Neural Network (CNN): the Single Shot Detector (SSD) MobileNet (Figure 4.20). It is a technique based on a Forward Convolutional Network (FCN) that generates a collection of fixed-size bounding boxes and a score for the presence of object class instances in those boxes, which come behind by a non-maximal suppression step to create the final detection. The score contains confidence values for the presence of each object categories. MobileNet is a class of well-organized models based on a simplified architecture that uses depth-separable convolutions to build lightweight Deep Neural Networks (DNN) that decomposes standard convolution into:

- Depth convolution: applies one filter to each input channel;
- Point convolution: $1 \times 1$ convolution to combine the outputs of the depth convolution.

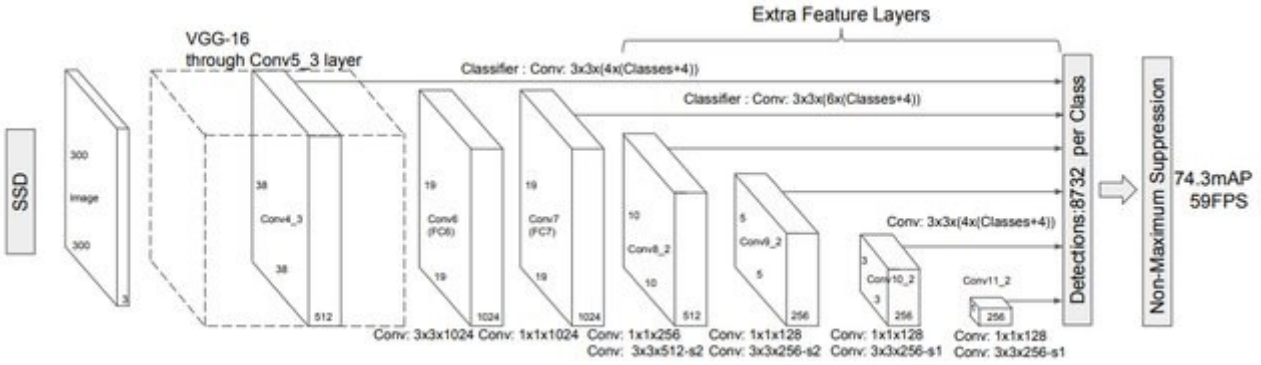



Figure 4.20: SSD-MobileNet architecture [67]

This factorization drastically reduces the computation and model size. According to the validation process, in the testing phase, the model is tried both in day and night situations, pointing out an accuracy surpassing 90% and an inference speed of 8.7 Frame per Seconds (FPS).

Another relevant solution makes use of Transfer Learning on YOLOv3, at different layers, suggested in [69]. Transfer Learning is a Machine Learning (ML) technique where a pre-trained is adapted for a related task through fine-tuning. This approach eliminates the need to train a new model from scratch, which can be both time-consuming and requires substantial amounts of data and computational resources [70]. Depending on the layers of the network on which Transfer Learning was carried out, it produced different results: they showed the worst performance when most of the layers were re-trained. The best metric values were reached with the TL model #4, with the last 6 layers re-trained.

A third solution, proposed in [71], develops a deep-learning model with Task-Specific Bounding Box Regressors (TSBBR) and conditional back-propagation mechanism for detection of objects in motion. This model reaches an accuracy of 86.54%, even with objects as small as $13 \times 13$ pixels. One of the purposes of the TSBBR-model is to detect various scales of object at the same time, by separating the small and large objects in the database, and then training the corresponding configuration of architectures independently.

Most of the analyzed models are not published or they are not so appropriate for the task of the project, since the latency of the model cannot be adapted to the real time detection.

| Neural Network | Paper | Accuracy | FPS | Disadvantages |
|---|---|---|---|---|
| SSD-MobileNet | "AI on the Road: NVIDIA Jetson Nano-Powered Computer Vision-Based System for Real-Time Pedestrian and Priority Sign Detection" | 92.25% (Day) - 95.15% (Night) | 8.7 | Only Pedestrian and Priority sign detection; private dataset to re-train the model. |
| DNN with TL#4 | "An optimized DNN Model for Real-Time Inferencing on an Embedded Device" | 90% (Precision) - 93.07% (Recall) – 91.56% (F1 score) | 35.082 on NVIDIA Jetson AGX Xavier | Based on YOLOv3. |
| TSBBR | "A deep-learning model with TSBBR and Conditional back propagation for moving Object Detection in ADAS applications" | 86.54% | 67 on 1080Ti; 19.4 on DRIVE-PX2; 8.9 on Jetson TX-2 | It detects only cars. |

Following the results mentioned above, two models are compared: the SSD-MobileNet and the YOLOv8, a heavier Neural Network, with high levels of performance (Figure 4.21).

| Name SO | Microsoft Windows 11 Home |
|---|---|
| Version | 10.0.22631 build 22631 |
| Productor SO | Microsoft Corporation |
| System model | HP Pavilion Laptop 15-cs1xxx |
| System type | PC based on x64 |
| Processor | Intel® Core ™ i7-8565U CPU # 1.80GHz, 1992 Mhz, 4 core |

Table 4.4: Data sheet HP Pavilion

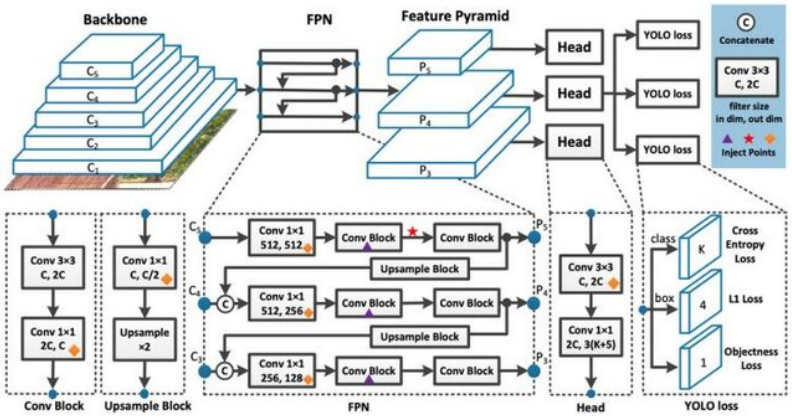


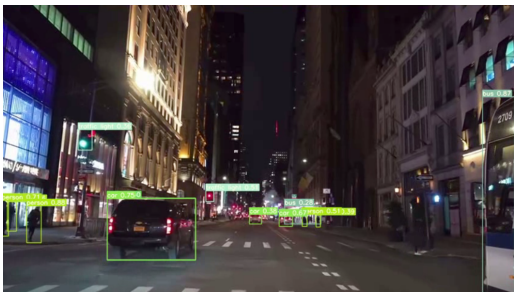

Figure 4.21: YOLOv8 architecture [72]

YOLOv8 is a real-time object detector [72], that stands out from other State-of-Art models by treating Object Detection as a single regression problem, rather than relying on computationally expensive and slower methods, like sliding window. YOLO predicts both the bounding box and class probabilities simultaneously, streamlining the detection process. The improved performance can be attributed to several key innovations:

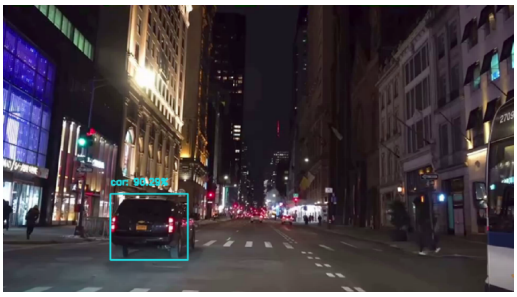
- Spatial Attention mechanism that focuses on some parts of the image;
- Feature Fusion, which improves the accuracy in the identification of small objects by combining high-level semantic features with low-level spatial information;
- Bottlnecks and Spatial Pyramid Pooling Fast (SPPF), which boost detection performance while maintaining high accuracy.

These advancements make YOLOv8 a suitable solution for Object Detection in Automotive field, particularly due to its high accuracy, real-time processing speed and efficiency.

Both the SSD-MobileNet and YOLOv8 are tested on Personal Computer (PC), specifically an HP Pavilion 4.4, and on the NVIDIA Jetson AGX Orin 4.1, to check their performance metrics.

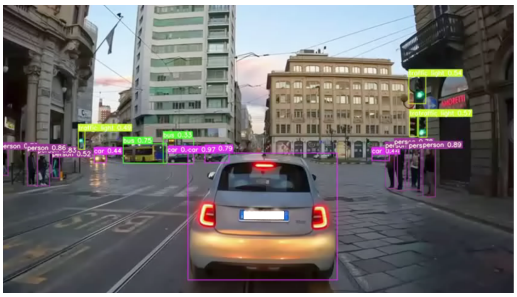


(a) YOLOv8



(b) SSD-MobileNet

Figure 4.22: Night Driving



(a) YOLOv8

(b) SSD-MobileNet

Figure 4.23: City Driving

The two models are evaluated using three different videos, each with specific characteristics: one featuring night-time driving, another with heavy traffic and a third with multiple object classes to detect. This testing aims to assess how the selected models perform under varying conditions.

As shown in Table 4.5, the simulations reveal a significant difference in accuracy between the two chosen models. YOLOv8 achieves a value over 90%, while the SSD-MobileNet only reaches around 20%. Accuracy is computed by analyzing different frames of the videos and measuring the ratio between the true positive (correctly classified objects) and all objects of interest in each frame. Regardless of the metric evidence, the results are visually evident, as the lightweight SSD-MobileNet struggles to detect even simple objects.

(a) YOLOv8

(b) SSD-MobileNet

Figure 4.24: Traffic Road

Another metric considered is the Frame Per Second (FPS): the SSD-MobileNet achieves the highest result, averaging 20 FPS, compared to YOLOv8's average of just 1 FPS. However, this difference is not particularly relevant due to the much lower accuracy of SSD-MobileNet.

| | YOLOV8 | | SSD-MobileNet | |
|---|---|---|---|---|
| Videos | Accuracy | FPS | Accuracy | FPS |
| drive_night.mp4 | 93% | 0.9 | 17% | 14.92 |
| turin_drive.mp4 | 94% | 0.943 | 19% | 22.23 |
| road_traffic.mp4 | 98% | 1.03 | 20% | 18.84 |

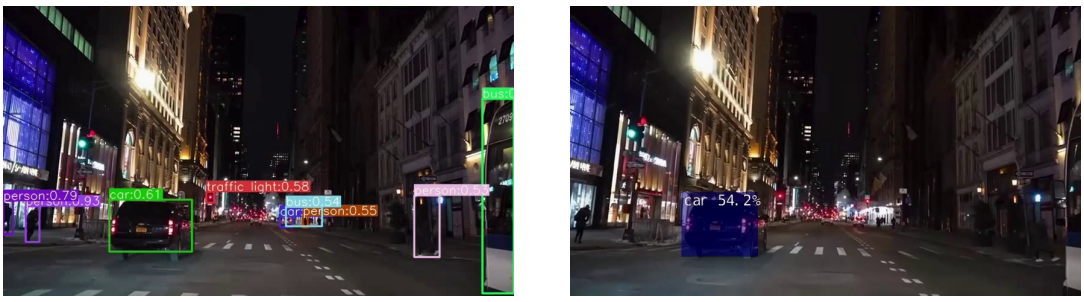
Table 4.5: Comparison between YOLOv8 and SSD-MobileNet in terms of accuracy and FPS on HP Pavilion

According to these metrics, YOLOV8 is the clear choice for deployment on the NVIDIA platform. However, for a comprehensive analysis, both models are tested on the NVIDIA Jetson AGX Orin.

Before testing them on the same videos, the appropriate versions are downloaded for the board: SSD-MobileNet is taken from the "jetson-inference" library and the YOLOv8 model is executed using the TensorRT Engine. The results obtained show the same trend as the tests performed on the PC (Table 4.6).

| | YOLOV8 | | SSD-MobileNet | |
|---|---|---|---|---|
| Videos | Accuracy | FPS | Accuracy | FPS |
| drive_night.mp4 | 87% | 60.77 | 24% | 188 |
| turin_drive.mp4 | 91% | 61.47 | 47% | 139 |
| road_traffic.mp4 | 94% | 63.62 | 23% | 170 |

Table 4.6: Comparison between YOLOv8 and SSD-MobileNet in terms of accuracy and FPS on NVIDIA Jetson AGX Orin




(a) YOLOv8 (b) SSD-MobileNet

Figure 4.25: Night Driving

It is also evident that SSD-MobileNet performs better in simpler scenarios, like in the City Driving video, but it delivers poor results when faced with more complex


(a) YOLOv8


(b) SSD-MobileNet

Figure 4.26: City Driving


(a) YOLOv8

(b) SSD-MobileNet

Figure 4.27: Traffic Road

scenes or night-time frames (Figure 4.27b and Figure 4.25b). Despite its higher FPS, SSD-MobileNet is inadequate for this project due to its low accuracy, which is a critical requirements for the task.

### 4.4.3 System Setup

#### Hardware Setup

The Micromobility vehicle, an adapted overboard platform equipped with two drive wheels and two steering wheels, serves as the foundation for a system capable of advanced AI and Computer Vision tasks (Figure 4.28). This vehicle's built-in battery provides power not only for its drive and steering mechanisms but also for all onboard computational hardware and sensors, enabling them to operate.

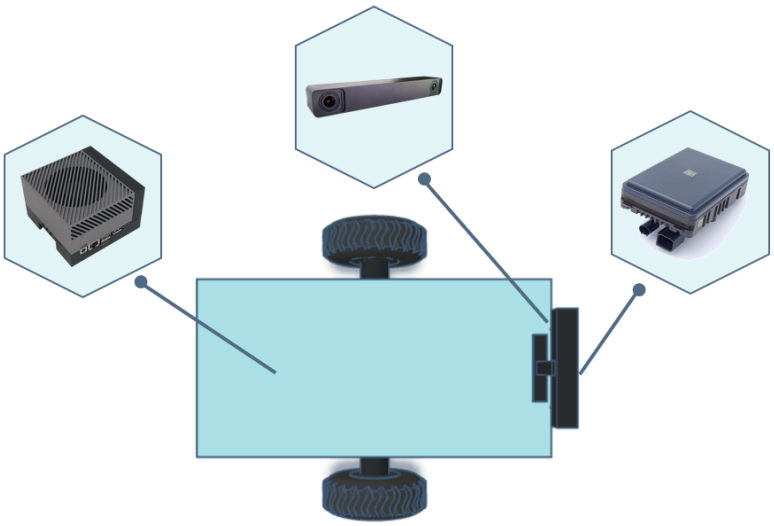


Figure 4.28: Architecture of the Micromobility vehicle

This computational setup is further supported by a tailored power system that ensures energy compatibility across all components. A voltage converter reduces the battery's 36V output to 12V, providing the necessary power for the sensors, including a 4D RADAR and a stereocamera, as well as for the Jetson AGX Orin.

At the heart of the vehicle's processing capabilities is the NVIDIA Jetson AGX Orin, which functions as the sole computing unit, handling all intensive AI computations and sensors data processing.

To ensure the functionality and integration of the stereocamera drivers inside the Isaac ROS framework, the choice of the right version of the NVIDIA suite had to be analyzed. To take full advantage of the features provided by the framework and the compatibility of third-party libraries used within it, the choice fell on JetPack 5.1.1.

The latter was installed via the Software Development Kit (SDK) Manager [73], an NVIDIA product that provides an end-to-end development environment setup solution specifically for the Jetson platforms.

Since the Leopard Imaging Hawk 3D stereocamera uses a GMSL2 transmission protocol, which is not directly supported and integrated into the Jetson boards, in order to use it within this architecture, a Leopard deserializer based on the MAXIM GMSL deserialization technology was used, the E3653-A03 (Figure 4.29) [74]. These deserializers convert the high-bandwidth video streams from multiple camera sensors transmitted over coaxial cables (in a serialized format) into a form that can be handled by the Jetson's CSI (Camera Serial Interface) input. The deserializer adopted is the Maxim MAX96712, specifically designed for GMLS2 camera systems and commonly used in Automotive and Embedded systems, where high-speed video data needs to be transmitted over long distances. The MAX96712 can handle up to 4 GMLS2 camera inputs over coaxial cables, making the system suitable for multi-cameras. Moreover, it supports high data rates up to 6 Gbps per lane (total bandwidth up to 24 Gbps).

Figure 4.29: Stereocamera Deserializer E3653-A03 [74]

### Software Setup

The Jetson AGX Orin is one of the leading products for Micromobility applications, given the computational resources it provides. However, during the integration of the YOLOv8 model, the main challenge encountered was the system's slowness, due to the heavy computational resources required by this specific model.

To overcome these limitations and meet the real-time requirements of the task, the NVIDIA Isaac ROS collection is thoroughly analyzed. This collection offers high-speed communication between nodes and efficient hardware acceleration, enabling a

smoother execution of computationally heavy models, like YOLOv8, while ensuring real-time Object Detection capabilities.

Based on the analysis of the documentation and code, the system architecture is composed of the following NVIDIA Isaac ROS repositories, as shown in Figure 4.30:

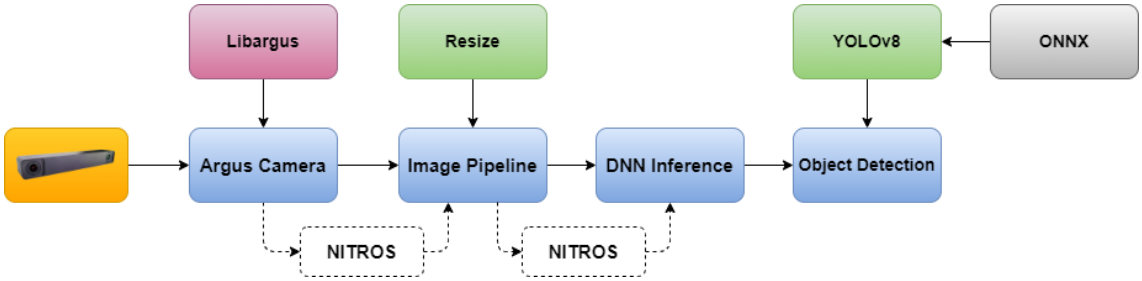



Figure 4.30: Stereocamera pipeline

- Isaac ROS Argus Camera [75], composed of ROS2 packages for managing camera sensors. Built on the Libargus Camera API, it is designed to efficiently manage camera devices on Jetson platforms. This repository leverages dedicated hardware engines to process sensor input, producing images while fully utilizing the Jetson system's memory bandwidth. It operates using Gigabit Multimedia Serial Link (GMSL) or Camera Serial Interface (CSI) to transfer raw data directly to GPU-accelerated memory, with the Image Signal Processor (ISP) hardware converting the data into a GPU-accelerated ROS2 output image topic. A key advantage of Isaac ROS Argus Camera is that it enables sensor data processing without involving the CPU in pixel handling, significantly boosting performance. The module supports a variety of sensor capture and processing features, such as Auto-White-Balance (AWB), Noise Reduction, and Auto-Exposure (AE);

- Isaac ROS NITROS [76] optimizes message formats, using type adaptation and negotiation and speeds up communication between nodes. This repository is a high-performance ROS2 package designed to accelerate the deployment of AI-powered robotics applications on Jetson platforms. It focuses on optimizing Image Processing and perception tasks by directly leveraging the GPU and dedicated hardware accelerators, minimizing CPU involvement. By taking full advantage of the Jetson hardware, Isaac ROS NITROS provides real-time performance for Computer Vision and sensor processing, enabling the development of more responsive and efficient robotic systems. Key features of Isaac ROS NITROS include support for advanced AI models, sensor fusion and efficient data handling, making it ideal for tasks such as Object Detection. With its streamlined integration into the ROS2 ecosystem, the repository allows developers to rapidly deploy and scale their AI-driven applications while

maintaining high performance on edge devices;

- Isaac ROS Object Detection [77] contains ROS2 packages for Object Detection, including the YOLOv8 model. Each package operates within a graph of nodes to generate bounding box detection array with object classes from an input image, that must be resized to match the required resolution of the specific model. This package integrates with the NVIDIA DeepStream SDK and supports pre-trained models like YOLO, Faster R-CNN, and SSD, among others, enabling efficient detection of a wide range of objects. The detected objects are published as ROS2 topics, providing bounding boxes, object labels and confidence scores;

- Isaac ROS Image Pipeline [78] is a ROS2 package designed for Image Processing, which is often required by camera outputs to meet the input specifications of various perception functions, such as cropping, resizing and mirroring;

- Isaac ROS DNN Inference [79] which consists of ROS2 packages specifically designed for performing DNN inference. This repository requires input Tensors, which are generated by a DNN encoder node that converts input images into Tensors. Tensors are multi-dimensional arrays that represent data in structured forms, such as images or feature map, allowing DNNs to efficiently process and analyze complex information like pixel intensities, spatial dimensions and color channel during inference tasks [80].

However, since NVIDIA Isaac ROS does not provide built-in support for 4D RADAR, a custom ROS2 node is integrated into the pipeline, specifically to handle its raw data. This node acts as a specialized wrapper, capturing and interpreting the packets transmitted by the RADAR, converting the raw information into ROS-compatible messages.

However, since NVIDIA Isaac ROS does not provide built-in support for 4D RADAR, a custom ROS2 node has been developed and integrated into the pipeline to manage the raw data produced by the radar. This custom node serves as a specialized interface between the RADAR hardware and the ROS ecosystem. Its primary function is to capture the packets transmitted by the RADAR, which typically include raw sensor data such as velocity, range, and angle information.

Once the raw data is received, the node processes and interprets the information, converting it into ROS-compatible messages, such as sensor_msgs/PointCloud2 or other custom message types suitable for further processing within the ROS environment. This transformation is crucial because it ensures that the RADAR data can be effectively integrated with other sensor inputs, such as the stereocamera and other modules within the system.

### 4.4.4 Stereocamera Pipeline

During the implementations of the stereocamera pipeline, challenges arise due to the discrepancies between the JetPack version required by Leopard drivers and the version required by NVIDIA Isaac ROS, for example concerning differences in Vision Programming Interface (VPI) versions. Furthermore, the directly compatible JetPack version lacks support for YOLOv8, the Neural Network chosen for the perception pipeline. This compatibility issue prevents the use of the latest version of NVIDIA Isaac ROS, which is essential for optimizing performance and utilizing the full capabilities of the stereocamera.

To address this, compatible versions of JetPack are carefully selected and additional packages from other NVIDIA Isaac ROS releases are integrated. Several adjustments are made to ensure the code is functional, enabling both NVIDIA Isaac ROS and YOLOv8 to work within the same environment.

NVIDIA Isaac ROS served as the starting point for the development of custom C++ modules, facilitating the integration of stereocamera, Image Processing, and Object Detection tasks. By building upon the existing capabilities of Isaac ROS, C++ modules were developed to interface with the sensor, handle data processing pipelines for image acquisition and feature extraction, and implement algorithms for Object Detection. These custom modules extended the framework's functionality, allowing for seamless integration and efficient processing of sensory data, ultimately contributing to the system's ability to perform real-time perception tasks in a robust and scalable manner.

#### Image Acquisition

The stereocamera pipeline begins with the acquisition of images, where the camera captures high-resolution visual data from its surroundings. In Computer Vision, understanding and manipulating the geometry of image formation is essential for accurately interpreting 3D structures from 2D images. At the core of this process are camera matrices [81], intrinsic and extrinsic, which serve as fundamental tools that link the 3D world coordinates to the 2D image coordinates captured by the camera.

The intrinsic camera matrix, typically denoted as $K$, contains parameters specific to the camera's internal properties, such as:

- focal length ($f_x$ and $f_y$): defines the camera's zoom level in the horizontal and vertical directions;
- principal point ($c_x$ and $c_y$): the optical center, which is the point on the image plane where the optical axis intersects; it is often close to the center of the

image.

The intrinsic matrix is generally represented as:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

This matrix allow the transformation of 2D pixel coordinates in the image plane to real-world spatial coordinates, helping the system understand the scale and orientation of objects in view.

Alongside $K$, the distortion matrix contains coefficients that model lens distortion, which occur due to imperfections in the lens. By accounting for these distortions, the system can correct raw images, producing geometrically accurate representation of the scene. These corrected images are essential for tasks such as depth estimation, where accurate pixel correspondence between the left and right images of the stereocamera allows the calculation of disparities and distance.

The extrinsic matrix combines rotation ($\mathbf{R}$) and translation ($\mathbf{t}$) matrices, defining the camera's position and orientation relative to a global or reference coordinate system. It enable mapping between the 3D world coordinates and the camera's 3D coordinate system:

$$Extrinsic\,Matrix = [R|t]$$

- $\mathbf{R}$: $3 \times 3$ rotation matrix, describing the camera's orientation;
- $\mathbf{t}$: $3 \times 1$ translation vector, representing the camera's position.

As this code is specifically developed for the stereocamera used in this project, the Isaac ROS Argus Camera uses a YAML file for configuring intrinsic and extrinsic parameters, directly provided by the manufacturer.

The camera data streams are processed using Isaac Argus node and are organized into specific output topics:

- /left/camerainfo
- /left/image_raw
- /right/camerainfo
- /right/image_raw

The "camerainfo" topics refer to the intrinsic and extrinsic characteristics of each camera lens, while "image_raw" contains the raw image data captured by the lenses.

Since the camera manages the two lenses separately, the output of the ROS2 node includes both the image data and camera information for the right and left lenses.

As illustrated in Figure 4.31, raw data streams captured from the stereocamera are processed by the ArgusStereo Node and then output as the topic shown in the diagram.

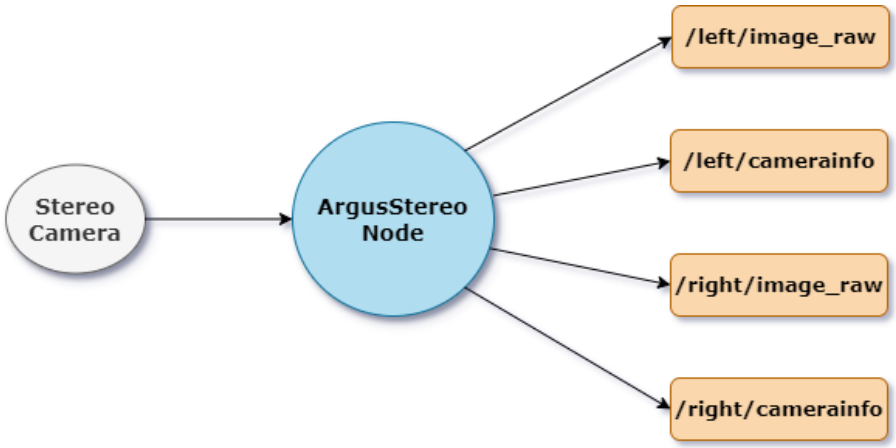



Figure 4.31: ROS2 graph ArgusStereoNode

The graphs of ROS2 nodes are reported in figures as the previous one, including their publishers, subscribers and topics. The standard convention adopted for these diagrams uses light blue circles to represent nodes, while topics are shown with their names are displayed above. The arrows between nodes and topics indicate the direction of communication: incoming arrows represent subscriptions and outgoing arrows represent publishing.

The performance of such topics is analyzed by checking the rate in publishing frames, obtaining an average value of 25 Hz for the right topic (/right/image_raw) and 27 Hz for the left topic (/left/image_raw).

### Rectification and Resize

After obtaining the images, the next step is resizing them. This process adjusts the dimension of the images to meet the specific input requirements of the Object Detection model, reducing their dimensions from 1920×1800 to 640×640 pixels.

Before resizing, the images may undergo rectification. Although this step is not mandatory, it improves the quality of the images and corrects lens distortion, ensuring to deliver accurate visual data for subsequent analysis.

The diagram in Figure 4.32 illustrates the ROS2 processing for the Rectify and Resize stages.

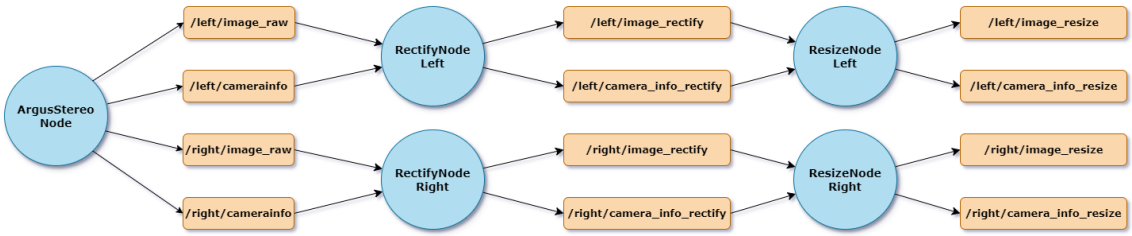



Figure 4.32: ROS2 graph of Rectify and Resize nodes

Each image topic is processed by a RectifyNode, considering separate nodes for the left and right lenses. These nodes rectify the images, adjusting them to compensate for camera distortions. The output of each RectifyNode includes:

- rectified image topics (/left/image_rectify and /right/image_rectify);
- rectified camera information topics (/left/camera_info_rectify and /right/camera_info_rectify).

The rectified images and camera info topics are then sent to ResizeNode instances that resize them as needed by the Neural Network. The final output topics of each ResizeNode are:

- resized image topics (/left/image_resize and /right/image_resize);
- resized camera information topics (/left/camera_info_resize and /right/camera_info_resize).

The topics that publish the rectified and resized images preserve the characteristics of the input topic in terms of average rate, maintaining nearly the same values as before: 25 Hz for the right lens and 27 Hz for the left lens.

### DNN Inference

In the Image Pipeline, DNN Inference is used to analyze and interpret images by running trained Neural Network models on new data. This inference step allows the system to recognize patterns, classify objects, detect features or perform other complex image-related tasks.

In this pipeline, after resizing the images, they are processed through a DNN encoder, which converts them into Tensors, as illustrated in Figure 4.33. This conversion is essential for preparing the data in a format suitable for further processing with Deep Learning frameworks.

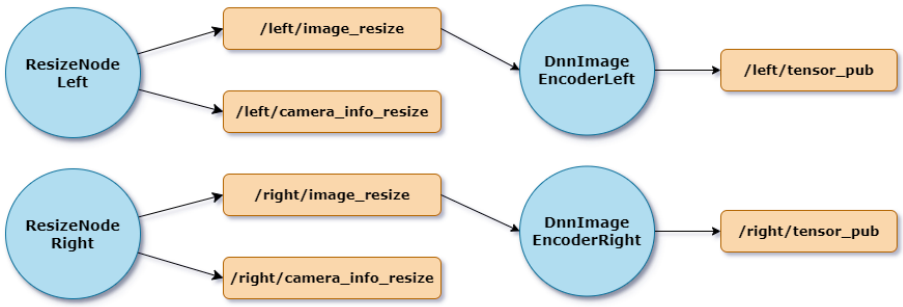



Figure 4.33: ROS2 graph for image encoder nodes

The Tensors are then directed to a node that utilizes TensorRT, as shown in Figure 4.34. This node is optimized for high-performance inference, enabling efficient execution of the Deep Learning model, by leveraging hardware acceleration to increase processing speed and minimize latency.

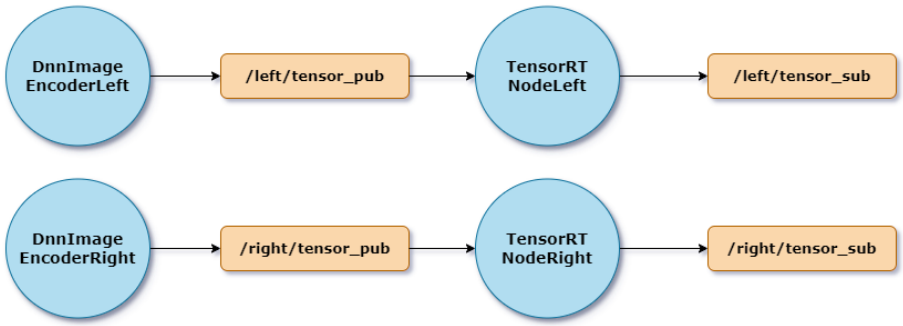



Figure 4.34: ROS2 Graph of TesorRTNode

The diagram in Figure 4.35 illustrates a typical node graph for performing DNN inference on image data.

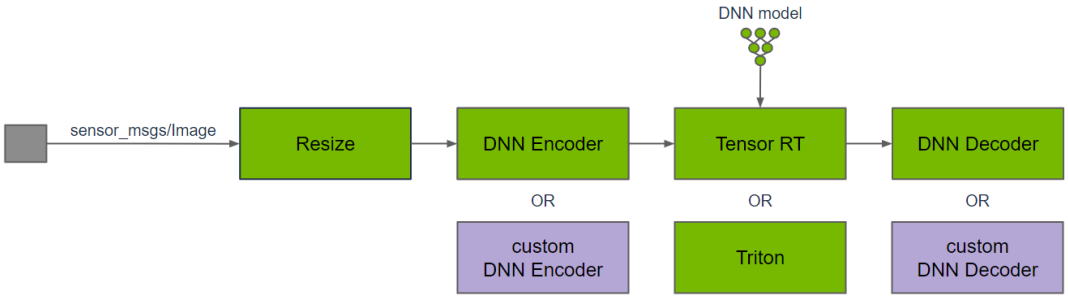



Figure 4.35: DNN Image Pipeline [79]

Initially, the input image is resized to match the DNN's required input resolution and also to reduce image resolution, which improves inference performance. This reduction is crucial because the computation time for DNN inference generally increases with the number of pixels in the image. Since DNN inference operates

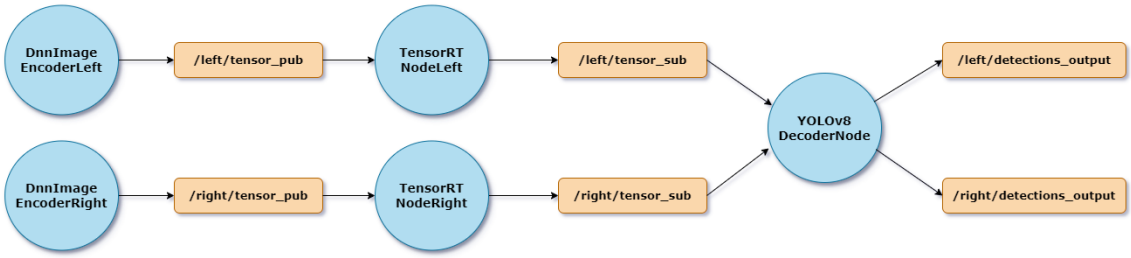



Figure 4.36: ROS2 graph for Object Detection node

on input Tensors, a DNN Encoder node is used to convert the input image into Tensors, applying any necessary pre-processing required by the DNN model. After the inference is complete, a DNN Decoder node converts the output Tensors into results that the application can utilize.

To perform DNN inference, separate ROS nodes for TensorRT and Triton are available. The TensorRT node utilizes TensorRT to deliver high-performance Deep Learning inference by optimizing the DNN model for Jetson and discrete GPUs. While it includes many common operations utilized by DNN models, it may not support newer or custom models. In such cases, the Triton node serves as an alternative, employing the Triton Inference Server, which supports various inference backends, including ONNX Runtime, TensorRT Engine Plan, TensorFlow and PyTorch.

### Object Detection

During the Object Detection phase, the Tensors are analyzed using the selected Neural Network model, YOLOv8, to classify objects and gather information about the scene captured by the stereocamera.

In detail, the output from the TensorRT node is directed to a decoder node, which is responsible for performing Object Detection using the YOLOv8 model. This model analyzes the processed data and identifies objects within the images, generating bounding boxes around detected items (Figure 4.36).

To improve the efficiency of the topics, some of the 80 classes, on which YOLOv8 model has been trained, are removed, considering that the classes of interest for the objects are the VRUs. For this purpose, a filter is applied when the vector of the detection output is created, adding only the elements belonging to the chosen classes. The launch file of the YOLOv8 repository activates the following nodes:

- the encoding node, to transform image output into tensors;
- the TensorRT node;

- the YOLOv8 decoder node, to perform the decodification of tensors to detections output, applying the YOLOv8 model;
- the visualization script, that represents and visualizes the bounding boxes of the identified objects, with the class to which they belong.

Finally, the results of the Object Detection process are visualized, displaying the bounding boxes over the original images. This visualization provides a clear representation of the detected objects, enabling further analysis and decision-making based on the object's classification and location within the environment. An example of the obtained results is shown in Figure 4.37: on the left, a parking lot full of cars is depicted, where one of the tests was performed, and on the right, a frame from the stereocamera showing the detected objects.

(a) Reference image

(b) Object Detection in stereocamera frames

Figure 4.37: Parking with cars

To evaluate the performance of the topics without visualization, a dedicated launch file is developed. Based on the frames analyzed, the accuracy metrics are high, as the network is able to recognize almost all the VRUs within the scene.

#### Challenges in Stereocamera Pipeline Implementation

During the development of the stereocamera pipeline, several challenges were encountered. One major issue was ensuring compatibility between the JetPack version installed on the Jetson AGX Orin and the necessary third-party libraries for the correct functioning of the Isaac modules used within this pipeline. Additionally, integrating the YOLOv8 model posed its own set of difficulties, as it was missing from the installed version of NVIDIA Isaac ROS, necessitating alternative approaches to incorporate the model into the pipeline.

Another significant challenge was building a pipeline capable of handling both lenses of the stereocamera. The process of managing the output from the two lenses proved complex, as it required custom configuration to ensure that the images were processed in real-time, maintaining accuracy and alignment for Object Detection tasks. These issues, combined with the inherent computational demands of the system, added layers of complexity to the development process.

### 4.4.5 4D RADAR Integration

In this thesis, the 4D RADAR sensor is integrated into the perception system, enhancing the vehicle's ability to detect and interpret its surroundings.

A specific local network configuration is required to effectively incorporate the RADAR ARS 548 RDI and to ensure that the RADAR sensor can communicate seamlessly with other components in the vehicle, specifically referring to the NVIDIA Jetson AGX Orin. A unique IP address is assigned to facilitate communication within the network, following the Ethernet protocol. Additionally, adequate power supply configurations must be established, ensuring that the RADAR operates within its specified voltage range. Once the RADAR is mounted within the vehicle, several parameters must be set correctly, including the vehicle's dimensions, the height at which the RADAR is placed and the orientation of the plug connections, as they directly influence the RADAR's FOV and detection capabilities.

A ROS2 driver [82] connecting to the RADAR's network socket is used to capture the incoming data packets and to process the information based on the payload length, extracting key parameters such as distance, velocity, azimuth and elevation angles.

The RADAR output signals are divided into three categories: Sensor Status, Object Interface (OI) and RADAR Detection Interface (RDI).

The Sensor Status message is sent every 50 ms and reflects the current configuration of the sensor, including status signal for dynamic parameters and a global RADAR status which is specially useful for validating that the configuration sent matches the expected settings.

The Object Interface provides a list of detected objects, including their dynamics, dimension and shape. Objects are classified as pedestrians, cars, trucks, motorbikes or bicyclists. The OI implemented in the ARS 548 sensor can output up to 50 tracked objects on the communication bus. Object data contains:

- Object ID;
- Distance with reference to vehicle rear axle (X,Y,Z);
- Relative and absolute acceleration (X,Y);
- Width, length, heading;
- RADAR Cross Section (RCS);
- Object age (in RADAR cycles);
- Classification (car, truck, bike, pedestrians, etc.);
- Dynamic property (moving, stationary);
- Yaw rate.

RADAR Detection Interface (RDI) handles raw data processing, alignment and detection tracking and contains:

- 3D position (distance, azimuth, elevation);
- RADAR Cross Section;
- Doppler speed.

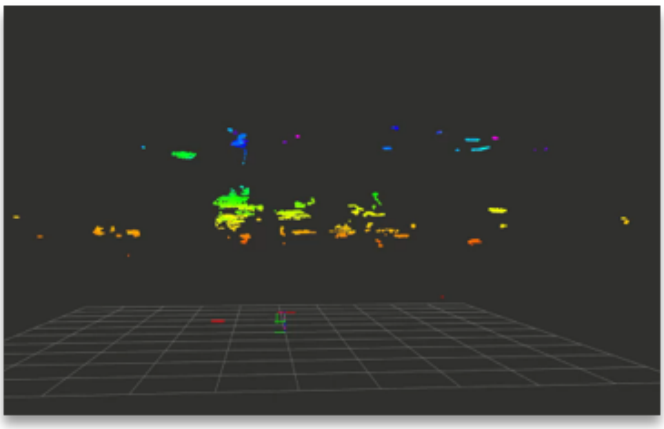
Since the driver used for the RADAR sensor is based on ROS2, it publishes messages on topics that can be visualized using RVIZ.

By using the aforementioned output signals, the first step for the RADAR integration involved characterizing the RADAR's output to understand the types of messages it transmitted and to verify whether the identified objects were consistent with the actual objects present in the environment.

The RADAR was tested under various conditions, with different objects placed in front of it. An initial challenges encountered was the unclear visualization of data through RVIZ, which made it difficult to interpret the messages immediately.

As shown in Figures 4.38, one of the first attempts involved testing the RADAR in a parking, with several cars (Figure 4.38a). The visualization output was completely unclear (Figure 4.38b), as it was impossible to determine which car each point belonged to.

(a) Reference image



(b) RADAR points

Figure 4.38: Parking with cars

As a result, the messages transmitted on the topics were firstly analyzed directly. The testing began with a single object and more objects were gradually introduced to verify the accuracy of spatial position information and Object Classification.

The spatial data was validated by calculating the actual distance to the points detected by the RADAR, resulting in highly accurate estimates, considering the RADAR reference system illustrated in Figure 4.39.

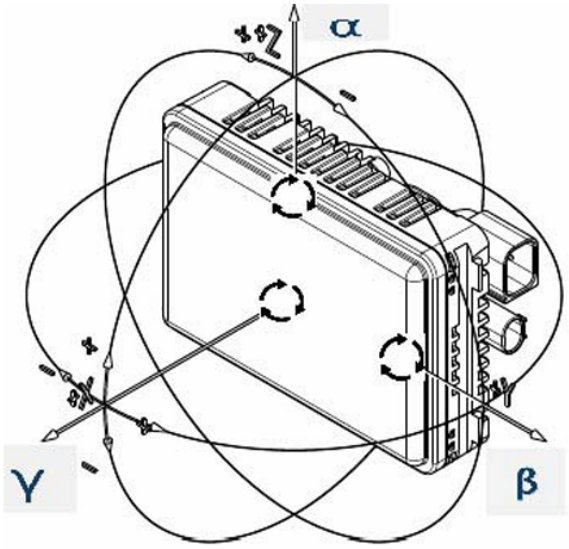
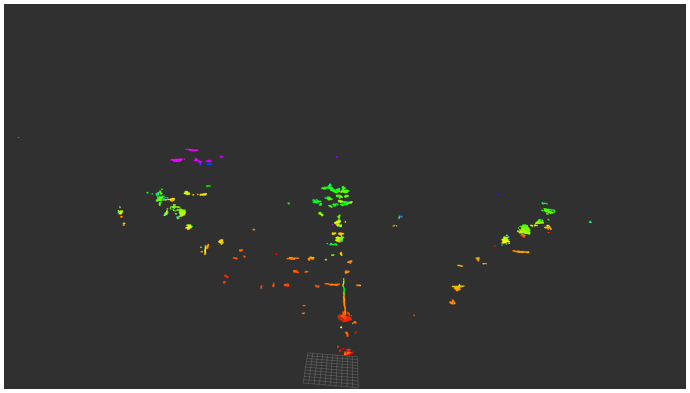



Figure 4.39: RADAR Reference System [55]

For example, in the same parking lot as before, after a clearer interpretation and data analysis, the Micromobility vehicle was turned to face the sidewalk (Figure 4.40a). By measuring one of the points closest to the RADAR, its accuracy was verified, resulting in optimal performance. Regarding the visualization output, a straight line of points was observed, representing the sidewalk, as shown in Figure 4.40b.

(a) Reference image



(b) RADAR points

Figure 4.40: Sidewalk

However, the classification results were not as successful, as all objects were consistently labeled as "hazard", indicating limitations in the RADAR's ability to distinguish between different object types in this particular setup.

### 4.4.6 Data Fusion

Data Fusion refers to the process of integrating information from multiple sources to generate more accurate, reliable or useful insights that could be obtained from a single source. This technique is widely used across various fields such as robotics, Autonomous Vehicles, defense and surveillance. Data Fusion can operate at different levels, including sensor, feature and decision levels, providing a multi-level approach to combining diverse data.

Data Fusion can be categorized into different levels based on the stage at which the data is combined:

- Low-level (Sensor-level) Fusion: it involves merging raw data from multiple sensors before significant processing occurs. The main objective is to improve

SNR or enhance the resolution of measurements, providing cleaner or more precise data for subsequent steps;

- Mid-Level (Feature-level) Fusion: it provides that features, such as edges, shapes or textures are extracted from raw sensor data. These features are then fused to reduce the amount of data while retaining relevant information;

- High-Level (Decision-level) Fusion: it integrates decisions from individual sensors or algorithms, each having already processed its data independently. This level is particular useful when combining information coming from sources that may operate at different scales or units. For example, it can merge insights from a variety of perception sensors like cameras, RADAR and LiDAR.

Data Fusion is widely used in Automotive Systems, integrating data from multiple sensors to improve the vehicle's understanding of its environment. This is essential for enabling ADAS and fully autonomous driving, enhancing decision-making, increasing the accuracy of Object Detection and improving safety and performance under various driving conditions.

It offers numerous advantages, particularly in improving system reliability and performance. By cross-verifying data from multiple sensors, it increases both accuracy and redundancy, ensuring that even if one sensor temporarily fails or provides incorrect data, the system remains functional. This approach enhances Object Detection and Classification, leading to more dependable operation.

Additionally, it creates a comprehensive understanding of the environment, providing a detailed view of objects including their speed, location and trajectory. This results in more precise decision-making, whether for Lane Keeping, Obstacle Avoidance or navigation through complex settings.

In challenging scenarios, such as rain, fog or direct sunlight, the integration of sensors that excel in different environments improves Obstacle Detection. The real-time processing of sensor data is another critical benefit: in autonomous systems, this allows for immediate reactions to changes in road conditions or unexpected obstacles, improving both decision-making and overall safety.

However, Data Fusion also poses several challenges: one significant issue is calibration between different sensors, since misalignment or timing discrepancy can lead to inaccurate results. The sensors often operate at different rates and Data Fusion systems need to synchronize this data in real-time to avoid latency or delays in decision-making. Moreover, the computational demands of fusing large volumes of raw sensor data, especially in real-time, can be substantial, particularly in system with low-latency requirements like Autonomous Vehicles. Dealing with conflicting sensor inputs presents a significant challenge: for instance, if RADAR detects an

object that a camera cannot see, the fusion system must resolve these discrepancies to create a trustworthy understanding of the environment.

In this thesis, the Data Fusion is performed between the outputs of stereocamera and 4D RADAR, enabling the projection of detected camera objects RADAR points.

Since the perception pipeline is a critical component of the perception system for Autonomous Vehicles, the decision to use Data Fusion from both sensors arises from the need to compensate for the limitations each sensor has when used independently. The stereocamera offers a clearer output, providing bounding boxes for detected objects, but it has a narrower FOV compared to RADAR. On the other hand, RADAR's wider FOV enables the detection of more objects and is more reliable in adverse weather conditions.

However, RADAR's detections are less precise and accurate than those of the stereocamera. Therefore, the stereocamera is used to refine and label the objects detected by RADAR, while its distance information is leveraged to enhance object tracking and positioning.

A ROS2 node is developed for performing Data Fusion, that subscribes to multiple ROS2 topics:

- RADAR Object Interface (OI) and RADAR Detection Interface (RDI);
- Detected objects from the stereocamera;
- Stereocamera frames and information.

Several callback functions are implemented to handle the incoming data, such as processing the RADAR target data from both topics, managing detected object data and camera images.

First, image data is received from both lenses of stereocamera, while RADAR information about detected objects is retrieved from the two specific topics. The RADAR targets are transformed into the stereocamera coordinates frame and then projected into 2D camera space. The projection of RADAR targets onto the camera image is used to establish associations between RADAR targets and object detected by the camera, based on the coordinates from both sensors. For each detected object, if a match is found, the nearest RADAR target is selected based on range. Finally, the pixel coordinates of the 2D Bounding Boxes of detected objects are projected in 3D and are published onto RADAR frames, which are visualized through RVIZ, as shown in Figure 4.41.

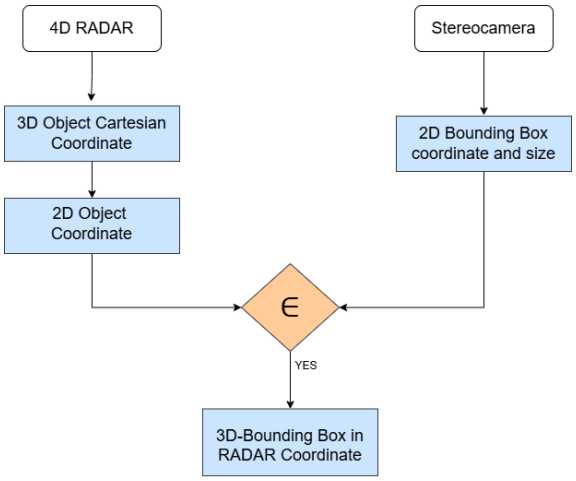
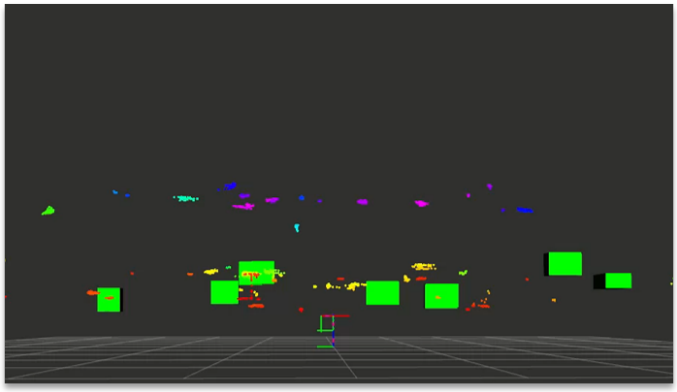



Figure 4.41: Data Fusion Flow Diagram

In addition to the visualization, which is certainly useful for better understanding the RADAR output, Data Fusion was used to classify the objects identified by the RADAR, as no adequate classification is provided. Moreover, the distance information obtained from the RADAR is essential for determining how close an object is to the vehicle, which is necessary for making potential decisions.

In the Figure 4.42, on the left, the usual parking lot with cars is shown, while on the right is an image from the RVIZ viewer displaying the data fusion: in green, there are the 3D bounding boxes of the cars identified by both the radar and the stereocamera.

(a) Reference image



(b) Data Fusion

Figure 4.42: Parking with cars

# Chapter 5

# Results

To validate Object Detection using stereocamera and RADAR, several metrics can be adopted, depending on the goals of the applications.

Detection Accuracy Metrics measure how well the system can correctly detect and classify objects [83]:

- Precision: indicates the fraction of identified objects that are actually correct;

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

- Recall: refers to the system's ability to detect all objects that are actually present;

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

- F1-Score: is the harmonic mean of precision and recall, used to balance the two metrics;

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

In this context, it is also crucial to validate the accuracy of position and distance estimates for the detected objects:

- Mean Distance Error: measures the average error in the distance estimates of objects in RADAR system and the actual distance;

$$Mean\ Distance\ Error = \frac{1}{N} \sum_{i=1}^{N} |d_{RADAR,i} - d_{ground\ truth}|$$

- Root Mean Square Error (RMSE): a common metric for distance estimation, indicating the root of the mean squared error between the estimate and the

actual value.

$$Root\ Mean\ Square\ Error = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(d_{RADAR,i} - d_{ground\ truth})^2}$$

In addition to accuracy, it is important to assess the temporal performance of the system:

- Processing Time per Object, that measures the time required to detect and identify a single object;
- Frame Per Second (FPS), which refers to the number of frames per second the system can process in real-time.

These metrics together provide a comprehensive evaluation of both the detection accuracy and the performance efficiency of the system.

During testing, four distinct scenarios were considered:

- Simple Indoor Scenario: in this setting, there are two cars and a person walking, with a straightforward indoor environment;
- Complex Indoor Scenario: the device is positioned in a corner, limiting its FOV and facing less defined objects (still with two cars and a person walking);
- Outdoor Scenario: this takes place in a parking lot with numerous cars, bicycles and two people walking;
- Outdoor Scenario with low lighting: in the same parking lot as before, but with reduced lighting, a person and a car maneuvering.

The testing process involved three stages:

- Stereocamera Testing: in each of the scenarios, the accuracy of the Object Detection algorithm was assessed. This was done by calculating Precision, Recall and F1-score for each frame to evaluate how accurately the stereocamera detected and classified objects;
- Radar Testing: using the same scenarios, data from the RADAR sensor was recorded with ROS2 bag files [84] from the two specific topics. ROS2 bag is a tool in ROS2 used for recording and storing data from topics and it allows users to capture sensor data, messages and other information during testing or operation. This recorded data can later be played back to simulate the same environment, which is helpful for debugging, analyzing or re-running tests. In these tests, the recordings were used to calculate the detected distance of

objects, which was then compared to the actual distance measured with a laser meter for accuracy verification. The validation of the RADAR employed metrics such as MSE and RMSE, while allowing for a certain tolerance due to the inherent challenge of ensuring that the RADAR was referencing the exact same point measured with the laser meter;

- Overall System Evaluation: finally, the performance of the entire system was evaluated in terms of FPS and processing Time per Object to measure how efficiently the system could handle Object Detection and tracking in real-time across all scenarios.

The only scenario that showed very poor performance was the more complex indoor environment, likely due to the particular angle of the stereocamera, which made it difficult to accurately recognize the objects present. In fact, one of the two cars was never recognized, while the other was identified as a "truck". The pedestrian, however, was consistently tracked, even in frames where it not completely visible. The RADAR, nevertheless, provided reliable measurements for the surrounding objects.

In the remaining tests, both the stereocamera and RADAR demonstrated optimal performance, consistently detecting cars and tracking pedestrians throughout their movements, even when they were obscured or distant from the vehicle, as shown in Table 5.1.

| **Scenario** | **Precision** | **Recall** | **F1-score** | **FPS** | **PTO (ms)** |
|---|---|---|---|---|---|
| Simple Indoor | 99% | 99% | 99% | 32.3 | 30 |
| Complex Indoor | 34% | 21% | 26% | 35.25 | 60 |
| Outdoor | 100% | 99% | 99.5% | 31.5 | 30 |
| Outdoor with low lighting | 100% | 99% | 99.5% | 36.125 | 30 |

Table 5.1: System-wide Metrics

# Chapter 6

# Conclusions and Future Works

The purpose of the thesis was to design and develop a suitable pipeline to perform Object Detection, using Micromobility devices as a testbed with the goal of future transfer learning applications on Macromobility vehicles. This approach was chosen to build a scalable system that could later be adapted to larger and more complex transportation systems.

The use of Micromobility vehicles was justified by several factors. First, the integration of sensors and data acquisition systems on these vehicles is significantly simpler and more cost-effective compared to larger vehicles. The smaller size and relatively straightforward mechanics of Micromobility devices allowed for rapid prototyping and deployment of the pipeline in real-world scenarios. Additionally, the controlled environment in which these tests were conducted provided the opportunity to evaluate the system's performance with greater precision, minimizing external risks and ensuring safer testing conditions.

Both the stereocamera and the 4D RADAR demonstrated distinct advantages and disadvantages during testing. The stereocamera provides high-resolution image frames with low latency, enabling precise Object Detection. The camera offers a wide FOV, allowing the system to capture a larger position of the environment in a single frame. A key aspect is its compatibility with ROS2, making it easier to integrate into existing robotics and autonomous systems. However, it can be affected by environmental factors such as rain, snow or fog, reducing its accuracy and processing stereo images is computationally intensive, especially in real-time applications.

The 4D RADAR excels in providing accurate information about the spatial position, range and speed of detected objects, but its ability to classify objects is limited, making it insufficient for effective detection when used alone. The 4D RADAR provides not only range, speed and azimuth like traditional RADAR, but also measures the elevation of objects, that enables a more detailed and accurate 3D representation of the environment. Moreover, the advanced processing power of the 4D RADAR

enable simultaneous tracking of multiple objects, even in cluttered or high-traffic areas. On the other hand, the complexity and advanced technology of this sensor make it more expensive than traditional RADAR systems, driving up the overall cost of the ADAS.

To overcome the shortcomings of each sensor, a Data Fusion approach was adopted, combining the strengths of both. This fusion allows the system to leverage the stereocamera's detailed visual information and RADAR's robust spatial and speed data, compensating for each sensor's weaknesses and resulting in a more reliable Object Detection system.

Regarding the software used in this work, ROS2 excels in facilitating communication between the various components of the pipeline, ensuring both speed and reliability. The NVIDIA Isaac ROS, employed for the stereocamera pipeline, offers superior performance by leveraging NVIDIA's hardware architecture. However, it presents challenges in integrating with external code and there are significant differences between the versions of its software collections, which complicates development and integration efforts.

While the proposed solution achieved promising results, several avenues for future improvements and further research remain open. One potential enhancement involves exploring alternative sensor configurations: for instance, incorporating monocameras instead of stereocamera could reduce the system's hardware complexity and cost.

Another important area for future work lies in expanding the software frameworks used for the perception system. While ROS2 and NVIDIA Isaac ROS were highly beneficial in this thesis, alternative frameworks, specifically designed for autonomous driving systems, could be explored. These platforms offer robust support for large-scale, real-time ADAS applications and may provide additional modules and optimizations for sensor fusion.

Moreover, as the automotive industry continues to adopt higher levels of autonomy, future iterations of this work could benefit from the inclusion of LiDAR sensors to complement the existing camera and RADAR setup. The addition of LiDAR would offer highly accurate 3D point cloud data, providing another layer of redundancy and robustness to the perception system. The fusion of these diverse data sources could further refine the system's accuracy, especially in complex or cluttered environments.

Lastly, transitioning the developed system from the prototypical stage to real-world testing on public roads remains an essential step. Rigorous testing under diverse conditions, such as varying lighting, weather and traffic scenarios, will help identify areas for optimization and ensure that the system meets safety and performance standards required for deployment in commercial ADAS solutions.

In conclusion, this thesis has laid a solid foundation for a stereocamera and RADAR-based perception system within the ADAS domain. With further exploration of monocamera setups, alternative frameworks, LiDAR integration and real-world validation, the proposed system can continue evolving towards a more mature, efficient and versatile solution for the next generation of ADAS.

# Bibliography


[1] *A Brief History of Autonomous Vehicles – from Renaissance to Reality.* URL: https://www.mobileye.com/blog/history-autonomous-vehicles-renaissance-to-reality/.

[2] Liang B. Zhao J. and Chen Q. "The key technology toward the self-driving car". In: *International Journal of Intelligent Unmanned Systems* (2018).

[3] International Business Machines (IBM). *What is computer vision?* URL: https://www.ibm.com/topics/computer-vision.

[4] M. Galvani. "History and future of driver assistance". In: *IEEE Instrumentation & Measurement Magazine* (2019).

[5] U.S. Department of Transportation. *Preparing for the future of transportation.* URL: https://www.transportation.gov/sites/dot.gov/files/docs/policy-initiatives/automated-vehicles/320711/preparing-future-transportation-automated-vehicle-30.pdf.

[6] Krzysztof Czarnecki. "Operational design domain for automated driving systems". In: *Taxonomy of Basic Terms ", Waterloo Intelligent Systems Engineering (WISE) Lab, University of Waterloo, Canada* 1 (2018).

[7] *SAE Levels of Driving Automation.* URL: https://www.sae.org/blog/sae-j3016-update.

[8] Ammar Mousa Abdulhassan Dr. Bassim Abdulbaqi Jumaa Anwaar Mousa Abdulhassan. "Advanced Driver Assistance System (ADAS): A Review of Systems and Technologies". In: *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* (2019).

[9] European Union. *General Safety Regulation.* URL: https://eur-lex.europa.eu/legal-content/IT/TXT/PDF/?uri=CELEX:02019R2144-20240707..

[10] Prof. Ajay Kumar Srivastava Rohit Bhagat Priyanshu Singh. "A Comprehensive Review on Advanced Driver Assistance Systems (ADAS)". In: *International Journal of Advances in Engineering and Management (IJAEM)* (2023).

[11] *Advanced Driver Assistance Systems (ADAS)*. URL: https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.chipsaway.biz%2Fblog%2Fadvanced-driver-assistance-systems-adas%2F&psig=AOvVaw3RrFLdl1afhJsvrurSDDust=1732180836802000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCNjP7M3K6okDFQAAAAAdAAAAABAE.

[12] Team LocoNav. *LiDAR Technology*. URL: https://loconav.com/blog/lidar/.

[13] Inc Neuvition. *Solid-state LiDAR*. URL: https://www.neuvition.com/media/blog/solid-state-lidar-3.html.

[14] *Understanding Ultrasonic Sensors in Cars*. URL: https://caradas.com/understanding-ultrasonic-sensors-in-cars/.

[15] A Karthikeyan Sowmya Shree B V. "Computer Vision based Advanced Driver Assistance System Algorithms with Optimization Techniques - A Review". In: *IEEE* (2018).

[16] *Micromobility Definition*. URL: https://www.mobilitycoe.org/literature-review/micromobility-definition/.

[17] Connor Klein Nicole DuPuis Jason Griess. *Micromobility in Cities*. URL: https://www.nlc.org/wp-content/uploads/2019/04/CSAR_MicromobilityReport_FINAL.pdf.

[18] Horace Dediu. *The Three Eras of Micromobility*. URL: https://micromobility.io/news/the-three-eras-of-micromobility.

[19] *History of the Bicycle: Origins and Evolution up to the Present Day*. URL: https://www.mammothbikes.com/en/blog/history-of-the-bicycle-origins-and-evolution/a-867.

[20] *The Evolution of Bicycles – An Exploration of their Journey through History and Impact on Society*. URL: https://bicyclepotential.org/blog/the-evolution-of-bicycles-an-exploration-of-their-journey-through-history-and-impact-on-society.

[21] *When Were Electric Bikes Invented?* URL: https://discerningcyclist.com/first-electric-bike/.

[22] *The History of Electric Bikes Written by Benny in News & Features*. URL: https://www.ebikebible.com/e-bike-facts-history-electric-bikes/.

[23] *Electric Scooter History - Invention, Evolution & Innovations*. URL: https://fluidfreeride.com/blogs/news/electric-scooters-history.

[24] Rider Guide. *Ultimate Guide to Electric Scooter*. URL: https://riderguide.com/guides/definitive-guide-electric-scooters/#:~:text=Electric%20scooters%20have%20a%20handful,stem%2C%20suspension%2C%20and%20tires..

[25] *Electric Scooter*. URL: https://www.google.com/url?sa=i&url=https%3A%2F%2Fglobal.razor.com%2Fit%2Fproducts%2Felectric-scooters%2F&psig=AOvVaw2ZfQscSxTtloset1pDuoO8&ust=1732188412861000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCLiyqvDm6okDFQAAAAAdAAAAABAE.

[26] *Electric Skateboard*. URL: https://www.google.com/url?sa=i&url=https%3A%2F%2Fsnowboardingprofiles.com%2Fexpert-tips-for-picking-the-best-electric-skateboard&psig=AOvVaw1nAkI5PPqcxexmmRo5Rnvf&ust=1732187216225000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCMC1oLDi6okDFQAAAAAdAAAAABAE.

[27] *The History and Evolution of the Electric Skateboard*. URL: https://linkyinnovation.com/the-history-and-evolution-of-the-electric-skateboard/.

[28] *The Types of Electric Skateboards*. URL: https://gyroorboard.com/blogs/posts/electric-skateboard-types?srsltid=AfmBOop_l5j7kCzwMGKv4_IQmoPyj2Ff6t50NkmDOty6oRhZzJW82uef.

[29] *How do Electric Skateboards Work?* URL: https://linkyinnovation.com/how-do-electric-skateboards-work/.

[30] *How do Electric Skateboards Work?* URL: https://linkyinnovation.com/how-does-an-electric-skateboard-motor-work/#:~:text=Belt%20Drive%20motors%20are%20the,wheels%20replaced%20by%20a%20motor..

[31] *Understanding the Different Types of Electric Skateboards - Ecomobl Electric Skateboards USA*. URL: https://www.ecomobl.com/blog/understanding-the-different-types-of-electric-skateboards/?srsltid=AfmBOooDM6YjsGrVodOBRx

[32] *How are robots used in transportation?* URL: https://robotnik.eu/how-are-robots-used-in-transport/.

[33] *Mobile manipulators*. URL: https://robotnik.eu/products/mobile-manipulators/.

[34] *Autonomous Mobile Robots*. URL: https://robotnik.eu/products/mobile-robots/.

[35] Gianluca Riccio. *ADAS, self-driving AI, also wants to make e-bikes safer*. URL: https://en.futuroprossimo.it/2021/12/adas-ai-guida-autonoma-ebike/.

[36] Xian-Hong Huang et al. "ADAS E-Bike: Auxiliary ADAS Module For Electric Power-assisted Bicycle". In: *2022 IET International Conference on Engineering Technologies and Applications (IET-ICETA)* (2022).

[37] *ADAS, l'AI a guida autonoma vuol rendere più sicure anche le e-bike*. URL: https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.futuroprossimo.it%2F2021%2F12%2Fadas-ai-guida-autonoma-ebike%2F&psig=AOvVaw1CCwyaKs66AF7Btiust=1732188532407000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCOjCqqTn6okDFQAAAAAdAAAAABAE.

[38] *Dock-Y developed its New Advanced Rider Assist System for Micro Mobility*. URL: https://cbitvb.uk/2023/07/11/dock-y-developed-its-new-advanced-rider-assist-system-for-micro-mobility/.

[39] *Transforming Mobility: The Architecture and Solutions of V2X Communication in Electric Vehicles*. URL: https://si.farnell.com/transforming-mobility-the-architecture-and-solutions-of-v2x-communication-in-electric-vehicles-trc-ar#:~:text=V2X%20enhances%20micro%2Dmobility%20safety,presence%20of%20vulnerable%20road%20users..

[40] *The Future of Micromobility*. URL: https://www.friwo.com/wp-content/uploads/20221004-Friwo-White-paper-EN-Zukunft-Mikromobilitaet.pdf.

[41] *The 2 great challenges of micromobility*. URL: https://alicebiometrics.com/en/the-great-challenges-of-micromobility/..

[42] *How to Overcome Challenges in Micromobility with IoT*. URL: https://mobisoftinfotech.com/resources/blog/iot-in-micro-mobility-industry/..

[43] *Honda CI Micro-mobility technologies*. URL: https://global.honda/en/tech/Honda_CI_Micro-mobility/.

[44] *ALBA Robot*. URL: https://www.alba-robot.com/it.

[45] *Jetson Modules*. URL: https://developer.nvidia.com/embedded/jetson-modules.

[46] *Jetpack SDK*. URL: https://developer.nvidia.com/embedded/jetpacks.

[47] *CUDA Toolkit*. URL: https://developer.nvidia.com/cuda-toolkit.

[48] *VPI - Vision Programming Interface*. URL: https://docs.nvidia.com/vpi/architecture.html.

[49] *NVIDIA Jetson AGX Orin Series*. URL: https://resources.nvidia.com/en-us-jetson-agx-orin-pathfactory-content.

[50] Indri M. "Computer Vision per la robotica - Parte 2". In: *Politecnico di Torino - Corso di Sistemi Robotici* (2023).

[51] *An Overview of GMSL2 Technology in Automotive Camera Systems*. URL: https://www.edge-ai-vision.com/2024/09/an-overview-of-gmsl2-technology-in-automotive-camera-systems/#:~:text=GMSL2%20is%20known%20for%20its,and%20a%20bidirectional%20communication%20path..

[52] *FAKRA Connector*. URL: https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.mouser.it%2Fnew%2Famphenol%2Famphenol-fakra-2-fakra-assemblies%2F&psig=AOvVaw2QOnr3PKJJvRVfkA3gkLoj&ust=1732193111524000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCLDNxKr4

[53] *LI-AR0234CS-STEREO-GMSL2-30*. URL: https://leopardimaging.com/wp-content/uploads/2024/07/LI-AR0234CS-STEREO-GMSL2-30_Datasheet_V1.8.pdf.

[54] *ARS 548 RDI*. URL: https://conti-engineering.com/components/ars-548-rdi/.

[55] *ARS 548 RDI - Short Description*.

[56] *100BASE-T1 Ethernet: the evolution of automotive networking*. URL: https://www.ti.com/lit/wp/szzy009/szzy009.pdf?ts=1730183476675&ref_url=https%253A%252F%252Fwww.google.com%252F.

[57] *100BASE-T1 MEDIACONVERTER NXP*.

[58] *ROS2 HUMBLE*. URL: https://docs.ros.org/en/humble/index.html.

[59] *Understanding services*. URL: https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Services/Understanding-ROS2-Services.html.

[60] *Understanding actions*. URL: https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Actions/Understanding-ROS2-Actions.html.

[61] *NVIDIA Isaac ROS*. URL: https://developer.nvidia.com/isaac/ros.

[62] *Type Adaptation Feature*. URL: https://ros.org/reps/rep-2007.html.

[63] *Type Negotiation Feature*. URL: https://ros.org/reps/rep-2009.html.

[64] *NITROS*. URL: https://nvidia-isaac-ros.github.io/concepts/nitros/index.html.

[65] *Isaac ROS Argus Camera*. URL: https://nvidia-isaac-ros.github.io/repositories_and_packages/isaac_ros_argus_camera/index.html.

[66] *Libargus Camera API*. URL: https://docs.nvidia.com/jetson/l4t-multimedia/group__LibargusAPI.html.

[67] K. Sarvajcz, L. Ari, and J Menyhart. "AI on the Road: NVIDIA Jetson Nano-Powered Computer Vision-Based System for Real-Time Pedestrian and Priority Sign Detection". In: *Applied Sciences* (2024).

[68] M. Narkhede and N. Chopade. "Real-Time Detection of Vulnerable Road Users Using a Lightweight Object Detection Model". In: *International Journal of Intelligent Systems and Applications in Engineering* (2023).

[69] J. Park et al. "Optimized DNN Model for Real-Time Inferencing on an Embedded Device". In: *Sensors* (2023).

[70] *What is transfer learning?* URL: https://www.ibm.com/topics/transfer-learning.

[71] G.-T. Lin, V. Malligere Shivanna, and J Guo. "A Deep-Learning Model with Task-Specific Bounding Box Regressors and Conditional Back-Propagation for Moving Object Detection in ADAS Applications". In: *Sensors* (2020).

[72] *YOLOv8 Architecture: A Deep Dive into its Architecture*. URL: https://yolov8.org/yolov8-architecture/.

[73] *NVIDIA SDK Manager*. URL: https://docs.nvidia.com/sdk-manager/index.html.

[74] *Leopard Imaging E3653-A03*. URL: https://leopardimaging.com/wp-content/uploads/2023/11/E3653-A03_Datasheet.pdf.

[75] *Isaac ROS Argus Camera*. URL: https://github.com/NVIDIA-ISAAC-ROS/isaac_ros_argus_camera.git.

[76] *Isaac ROS NITROS*. URL: https://github.com/NVIDIA-ISAAC-ROS/isaac_ros_nitros.gitt.

[77] *Isaac ROS Object Detection*. URL: https://github.com/NVIDIA-ISAAC-ROS/isaac_ros_object_detection.git.

[78] *Isaac ROS Image Pipeline*. URL: https://github.com/NVIDIA-ISAAC-ROS/isaac_ros_image_pipeline.git.

[79] *Isaac ROS DNN Inference*. URL: https://github.com/NVIDIA-ISAAC-ROS/isaac_ros_dnn_inference.git.

[80] *Tensors in machine learning*. URL: https://medium.com/@nabinadhikari190/tensors-in-machine-learning-2c4f6c336244.

[81] *Dissecting the Camera Matrix, Part 3: The Intrinsic Matrix*. URL: https://ksimek.github.io/2013/08/13/intrinsic/.

[82] *ARS_548_RDI Driver*. URL: https://github.com/robotics-upo/ars548_ros.

[83] Rafael Padilla, Sergio L. Netto, and Eduardo A. B. da Silva. "A Survey on Performance Metrics for Object-Detection Algorithms". In: *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)* (2020).

[84] *Recording and playing back data*. URL: https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Recording-And-Playing-Back-Data/Recording-And-Playing-Back-Data.html.