THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

Camera-Based Perception under Domain Shifts

Efficient training and multi-view fusion for mobile robots in internal logistics

Erik Brorsson

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2025

Camera-Based Perception under Domain Shifts

Efficient training and multi-view fusion for mobile robots in internal logistics

Erik Brorsson

Acknowledgements, dedications, and similar personal statements in this thesis, reflect the author's own views.

Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000

Printed by Chalmers Digital Printing
Gothenburg, Sweden, September 2025

Camera-Based Perception under Domain Shifts

Efficient training and multi-view fusion for mobile robots in internal logistics

Erik Brorsson
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Autonomous Mobile Robots (AMRs) have emerged as a promising solution for efficient material handling in internal logistics. However, deployment in dynamic, human-shared environments presents major challenges. A core difficulty lies in perception, which must reliably interpret complex scenes to enable safe navigation. This thesis investigates camera-based perception for AMRs, motivated by the availability, low cost, and versatility of cameras. A proposed reference AMR system combining onboard and infrastructure-mounted sensors provides the foundation for exploring key challenges. Two main problems are addressed: the heavy demand for labeled training data in deep learning pipelines, and the difficulty of fusing information from multiple cameras while ensuring robustness to changes in sensor configuration.

To tackle these challenges, the thesis explores unsupervised domain adaptation (UDA) as a unifying strategy. UDA enables models to transfer knowledge from labeled simulated datasets to unlabeled real-world data and to adapt across different multi-camera setups without requiring additional labels. Building on this principle, we propose methods that exploit reliable pseudo-labels and data augmentation to mitigate domain shifts.

The methods are evaluated on benchmarks for semantic segmentation and multi-view pedestrian detection, showing improved performance under domain shifts without extra labeled data. This enables perception systems to rely more on simulated datasets and to adapt more readily to new scenarios, reducing annotation costs. By advancing both monocular and multi-view perception and proposing a reference architecture, this work supports scalable camera-based AMR systems and takes a step toward their widespread deployment.

Keywords: Computer vision, unsupervised domain adaptation, autonomous mobile robots, internal logistics.

To Irma.

List of Publications

This thesis is based on the following publications:

[A] **Erik Brorsson**, Kristian Ceder, Ze Zhang, Sabino Francesco Roselli, Endre Erős, Martin Dahl, Beatrice Alenljung, Jessica Lindblom, Thanh Bui, Emmanuel Dean, Lennart Svensson, Kristofer Bengtsson, Per-Lage Götvall, Knut Åkesson, "Infrastructure-based Autonomous Mobile Robots for Internal Logistics - Challenges and Future Perspectives". Submitted for possible journal publication.

[B] **Erik Brorsson**, Knut Åkesson, Lennart Svensson, Kristofer Bengtsson, "ECAP: Extensive Cut-and-Paste Augmentation for Unsupervised Domain Adaptive Semantic Segmentation". Published in proceedings of *IEEE International Conference on Image Processing (ICIP)*, pp. 610–616, Oct. 2024.

[C] **Erik Brorsson**, Lennart Svensson, Kristofer Bengtsson, Knut Åkesson, "MVUDA: Unsupervised Domain Adaptation for Multi-view Pedestrian Detection". Submitted for possible journal publication.

Contents

Acknowledgments

I would like to thank my colleagues and managers at AB Volvo for supporting my work on the industrial side. In particular, I am grateful to Per-Lage Götvall, who introduced me to ongoing research on automating internal logistics at AB Volvo, which later became a central part of this thesis. I am also thankful for the many fruitful interactions with fellow PhD students at Chalmers and within the WASP community. Furthermore, I would like to thank my supervisors Knut Åkesson, Kristofer Bengtsson, and Lennart Svensson for their continuous guidance and support throughout this project.

Finally, I am especially grateful to my partner, Irma Björnwall. Her steady support and understanding throughout these past years have been invaluable, both for this thesis and for me personally.

This work was supported by AB Volvo, the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, and the Vinnova funded projects SMILE IV (2023-00789) and AIHURO (2022-03012). The experiments were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) and the Swedish National Infrastructure for Computing (SNIC) at Chalmers Centre for Computational Science and Engineering (C3SE) partially funded by the Swedish Research Council through grant agreements no. 2022-06725 and no. 2018-05973.

Acronyms

ML:	Machine Learning
CNN:	Convolutional Neural Network
FCN:	Fully Convolutional Network
MLP:	Multilayer Perceptron
ViT:	Vision Transformer
UDA:	Unsupervised Domain Adaptation

AGV:	Automated Guided Vehicle
AMR:	Autonomous Mobile Robot
SLAM:	Simultaneous Localization and Mapping
UWB:	Ultra-Wideband
UX:	User Experience
NMS:	Non-Maximum Suppression
BEV:	Bird's-Eye-View
RAIL:	**R**eference **A**rchitecture for **I**nfrastructure-based AMR Systems in Internal **L**ogistics

Part I

Overview

CHAPTER 1

Introduction

Automation is reshaping industries by increasing productivity, improving safety, and reducing costs. Among its many applications, internal logistics stands out as a critical area in manufacturing, where efficient material transport directly affects production flow. In highly structured settings, Automated Guided Vehicles (AGVs) have demonstrated the value of automated transport by navigating along fixed routes. However, AGVs are less suited to dynamic environments where obstacles, layouts, and human activity change frequently, since they cannot readily adapt to new scenarios [1]. Variability is particularly pronounced in many real-world factories. For example, at Volvo Trucks' final assembly plants, the production process must accommodate a broad range of vehicle types—including battery-electric, internal combustion engine, and hydrogen-powered trucks—all assembled on the same production line. This diversity significantly increases system complexity and demands flexible material handling. Consequently, automation remains challenging, reinforcing the industry's reliance on human workers.

Driven by the need for more flexible material handling, Autonomous Mobile Robots (AMRs) have emerged as a promising solution [1]. Unlike AGVs, AMRs rely on onboard sensing and decentralized decision-making to navigate

freely in the environment, dynamically avoiding obstacles and collisions. To operate efficiently, they must perform three fundamental tasks: (i) estimate the state of the environment, (ii) localize themselves within it, and (iii) decide on subsequent actions.

In human-shared environments, such as Volvo Trucks' final assembly plants, environment perception (task i) may involve distinguishing obstacles from traversable areas, detecting and predicting the motion of dynamic agents (e.g., pedestrians and forklifts), and interpreting contextual cues such as floor markings or hand signals. Each of these tasks places heavy demands on the robot's vision system. Decision-making (task iii) requires sophisticated planning algorithms that become increasingly complex in unpredictable, dynamic environments.

Current AMR systems typically employ 3D cameras or LiDAR [1] to capture data about their surroundings, combined with powerful onboard computing to process this information. Although advances in sensing and decision-making algorithms enable these platforms to address the required tasks in principle, significant challenges remain. Perception and localization often degrade in highly dynamic environments with frequent changes and occlusions, while reliance on computationally intensive algorithms increases platform cost and complexity.

To advance the development of AMR systems, this thesis investigates how an overarching infrastructure of ceiling-mounted cameras, on-premise cloud servers, and wireless connectivity can be used to enhance system capabilities. The focus is on environment perception that leverages both onboard and infrastructure-mounted cameras. While onboard cameras excel at perceiving the immediate surroundings of the AMRs, ceiling-mounted cameras can provide reliable perception in occluded or distant areas where onboard sensors struggle. On-premise cloud computing enables more efficient processing of computationally demanding algorithms, while wireless communication allows AMRs to offload tasks to centralized resources. This reduces the hardware requirements of the AMR platform, which can simplify its design and lower costs.

While autonomous systems can employ a wide range of sensor modalities, this thesis focuses exclusively on camera-based perception. This choice is motivated by the widespread availability, low cost, and flexibility of RGB cameras in industrial settings, as well as the rapid advancement of Machine

Learning (ML) methods tailored to visual input. The narrower scope enables a deeper study of the algorithms, data challenges, and architectures specific to camera-based systems.

The scope is further limited to instantaneous environment perception: deriving the 3D layout of the environment and current locations of dynamic objects from a single set of camera frames. While tasks involving temporal reasoning, such as object tracking and motion forecasting, are vital for sophisticated decision making, they typically build on these per-frame detections. Instantaneous perception thus forms the foundation of modern perception pipelines.

State-of-the-art perception algorithms rely on deep learning models, such as convolutional neural networks (CNNs) [2] and vision transformers (ViTs) [3], and require large amounts of annotated training data. Unfortunately, collecting and annotating real-world data is often time-consuming and costly [4]. Therefore, heavy reliance on labeled datasets is a major bottleneck for deploying advanced perception methods at scale. One research focus of this thesis is to reduce this dependence through training strategies that effectively leverage inexpensive simulated data and unlabeled real-world data. A central challenge is how to bridge the domain gap between simulated and real-world environments, while also learning robust representations from data that lack labels.

In infrastructure-assisted AMR systems, perception can be based on a combination of onboard and ceiling-mounted cameras. As the number of cameras increases, overlapping fields of view provide richer information but also introduce new algorithmic challenges. A central question becomes how to fuse complementary viewpoints in a way that exploits the full potential of multi-camera setups. This defines the second major focus of the thesis. Special emphasis is placed on approaches that can generalize across different camera configurations, since installations often vary in the number, placement, and types of cameras.

In summary, the scientific contribution of this thesis lies in advancing camera-based perception methods through data-efficient training and multi-camera fusion, while the industrial contribution lies in supporting the deployment of high-performing and cost-efficient AMR systems that address practical challenges in internal logistics.

1.1 Research questions

This thesis work investigates the following research questions:

RQ 1 *What are the key challenges in designing reliable camera-based perception systems for AMRs in internal logistics?*

This question maps out the technical challenges of building robust perception systems for AMRs in dynamic, human-shared environments. Chapter 2 grounds the discussion in an industrial use case at Volvo Trucks and the infrastructure-based reference architecture for AMR systems presented in Paper A. Chapter 3 continues by outlining fundamental methods in modern ML-based perception pipelines and highlights persistent challenges. Together, these results provide a foundation for addressing the more focused questions that follow.

RQ 2 *How can ML models for camera-based perception be trained efficiently without relying on large-scale labeled real-world datasets?*

Given the high cost of manual data annotation, it is important to investigate methods that reduce the need for large labeled datasets. Chapter 4 addresses this question by exploring techniques such as semi-supervised learning, self-supervised learning, and unsupervised domain adaptation to enable training with fewer annotations while maintaining model performance. Both Paper B and Paper C contribute with machine learning algorithms dedicated to learning from unlabeled data.

RQ 3 *How can ML methods be designed to leverage multiple cameras to increase performance and robustness in AMR perception?*

When using multiple cameras, either onboard the robot or mounted on infrastructure, a central challenge is how to align and integrate these diverse inputs into a comprehensive and accurate understanding of the environment. Chapter 5 investigates methods for multi-camera fusion, focusing on ML approaches that learn fusion strategies from data. Particular attention is given to generalizability: since camera setups may vary between installations, it is critical that the learned strategy transfers across different configurations. Paper C addresses this problem through a case study on multi-camera pedestrian detection with calibrated, static cameras.

1.2 Research approach

This thesis follows a design-science research (DSR) methodology [5], which focuses on creating and rigorously evaluating artifacts that address relevant real-world problems. In DSR, research is structured around three interconnected cycles: the relevance cycle, which engages with the application domain to identify and refine problems; the rigor cycle, which draws on and contributes to the existing scientific knowledge base; and the design cycle, which iteratively builds and evaluates artifacts.

This thesis follows these principles through a three-step approach:

1. Identify practical challenges from real-world deployments (Relevance Cycle) – In the industrial application at Volvo Trucks, AMR operations are observed in practice to pinpoint key perception challenges. In Paper A, this process leads to the RAIL reference architecture for infrastructure-based AMR systems, developed and tested within Volvo's assembly plant. Through literature review and user feedback, we identify the core perception challenges that frame RQ 1. These real-world insights then inform the next phase of our work.

2. Develop methods through academic rigor (Design + Rigor Cycles) – Guided by the identified challenges and grounded in computer vision and machine learning theory, we design and benchmark new perception methods against state-of-the-art on publicly available datasets. Papers B and C develop two machine-learning artifacts that directly address RQ 2 and RQ 3. Through comprehensive quantitative evaluation, our methods demonstrate practical utility by solving relevant problems more efficiently than existing approaches. Through academic rigor, our work also contributes to the existing scientific knowledge base.

3. Return to industrial context for evaluation (Relevance Cycle) – Based on academic findings, the last phase of our approach, which has not yet been executed, is to reapply our methods in the original industrial context. The purpose is to assess whether laboratory improvements translate to operational performance, ensuring both scientific contribution and practical impact.

Our approach integrates both inductive and deductive reasoning. Induction arises from empirical observations: through quantitative evaluation on

dedicated benchmarks we seek generalizable conclusions regarding each artifact's strengths and limitations. Deduction shapes our hypotheses, drawing on established theory in computer vision and machine learning. Yet, in the complex, data-driven terrain of modern machine learning, definitive mathematical proofs are rare; instead, rigorous experimental validation becomes the primary arbiter of an artifact's efficacy. By grounding our work in design-science principles and substantiating our claims with reproducible quantitative experiments and targeted qualitative feedback, this thesis advances state-of-the-art methods for camera-based perception in internal-logistics AMR systems.

1.3 Outline

This thesis constitutes two parts. Part I presents an overview of the research, and Part II includes the appended papers. Part I is arranged in seven chapters. Chapter 1 presents the introduction to this thesis. Chapter 2 presents the industrial use case and a reference architecture for AMR systems (Paper A). Chapter 3 reviews computer vision fundamentals, focusing on modern machine learning methods and existing challenges. Chapter 4 examines learning from limited labeled data and presents a case study on semantic segmentation (Paper B). Chapter 5 explores multi-camera perception with a case study on pedestrian detection (Paper C). Chapter 6 summarizes the included papers. Finally, Chapter 7 concludes the thesis by addressing the research questions, highlighting contributions, and outlining future work.

CHAPTER 2

Infrastructure-based AMR System for Automated Internal Logistics

This chapter begins with outlining the industrial demand for automated internal logistics. Thereafter, a reference architecture for AMR systems is described, followed by an industrial implementation. The chapter concludes by summarizing existing challenges in automating internal logistics.

2.1 Industrial demand

At Volvo Trucks' final assembly plants, both battery-electric, internal combustion engine, and hydrogen-powered trucks, are assembled on a single production line. To assemble these trucks, around 25 000 unique parts are used. Since the number of unique parts is so large, it is infeasible to store all parts in direct connection to the assembly line. Instead, the process relies on a complex internal logistics solution that enables just-in-time delivery of materials to the assembly line. Moreover, since the produced trucks are tailored to the customer demands, the demand for different parts vary over time, which requires flexibility in logistics.

The current logistics situation is based mainly on manually driven vehicles, such as forklifts and tugger trains. Autonomous Guided Vehicles (AGVs) are also used for certain transportation, but since the AGVs follow fixed routes, they can can only accommodate parts of the logistics flow. Specifically the part that is expected to remain the same for a considerable time, since changing the routes of the AGVs is cumbersome. Meanwhile, the human workforce is tasked to deal with the flexible part of the logistics.

As an alternative to AGVs, Autonomous Mobile Robots (AMRs) have emerged as a flexible solution for automated material transport, enabling free navigation beyond fixed routes. When introduced into complex industrial environments, these robots must navigate safely in traffic shared with manually driven vehicles and pedestrians. To achieve this, AMRs typically rely on onboard sensors to localize themselves and detect obstacles in real-time for collision-free navigation [1]. However, onboard sensors have limited range and are susceptible to occlusions, complicating perception in cluttered and crowded industrial environments. Localization, which is often based on simultaneous localization and mapping (SLAM) [6], is also challenging in highly dynamic environments where moving objects interfere with the mapping process. Together, these issues reduce the robustness and reliability of AMR systems, hindering their large-scale deployment in industry.

2.2 Reference architecture

To address the limitations of current AMR systems, Paper A proposes RAIL: *a **R**eference **A**rchitecture for **I**nfrastructure-based AMR Systems in Internal **L**ogistics*, designed to meet the demand for flexible material transport in dynamic factory environments. RAIL generalizes traditional onboard-only AMR stacks by integrating infrastructure-mounted sensing and on-premise cloud computing to enable collaborative perception, centralized coordination, and scalable computational resources. Figure 2.1 (from Paper A) illustrates the high-level structure of RAIL.

Overall framework

In RAIL, AMRs and external sensors (e.g., ceiling-mounted cameras) transmit sensor data or extracted features to an on-premise cloud, serving as a central

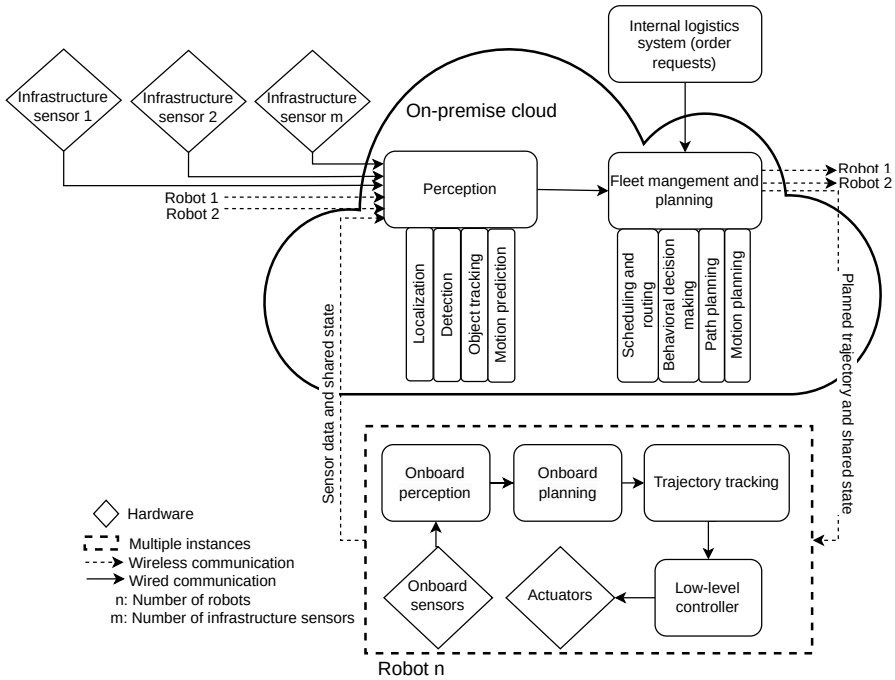

Figure 2.1: RAIL: *a **R**eference **A**rchitecture for **I**nfrastructure-based AMR Systems in Internal **L**ogistics.*

hub for global perception and fleet coordination. RAIL adopts a modular software architecture that separates the system into dedicated perception and decision-making components. These components, which are further described in subsequent sections, are responsible for estimating the states of the robots and the state of the environment, and deciding the next action for each robot.

Naturally, the action for each AMR is based on the current transportation requests, which are provided from the factory's internal logistics system. The actions computed for each AMR, generally described by motion plans in the form of trajectories, are communicated wirelessly to the fleet. These motion plans are then executed on each AMR using a local trajectory tracker and low-level controllers.

In case of infrastructure disruptions, such as intermittent connectivity or

camera failure, the AMRs may also deploy onboard perception and planning modules. To facilitate operations under these conditions, additional information such as robot states or environment maps may also be shared between the AMRs and the on-premise cloud. In Figure 2.1, such information is simply denoted by the *shared state*.

Perception

The responsibility of the perception system is divided into two key tasks: estimating the state of the robots (localization) and estimating the state of the environment (environment perception).

Localization for AMRs is traditionally achieved by SLAM algorithms based on onboard sensor data, which aims to build a map of the environment and localize the robot within this map simultaneously. While universally applicable, SLAM is difficult in dynamic environments. RAIL lends itself to complementing SLAM with infrastructure based localization, using technologies such as Ultra-Wideband (UWB) or ceiling-mounted cameras.

Environment perception is further divided into detection, tracking and motion prediction.

- Detection: identifies and classifies obstacles and agents (humans, forklifts, tugger trains, pallets).
- Object tracking: associates detections over time to understand how agents move in the environment, also enabling velocity estimation.
- Motion prediction: forecasts the future movement of dynamic agents, so that the planning module can anticipate and avoid potential conflicts.

Each of these perception components can benefit from increased computational budget on the cloud and from varying data sources; both onboard the robot and mounted on infrastructure.

Decision making

Based on the perception results, the decision-making module is designed to provide a long-term route to the goal destination, make short-to-mid-term decisions to progress towards the goal, and compute a feasible trajectory that can be followed to achieve the desired behavior. This is managed by four

modules: scheduling and routing, behavioral decision making, path planning, and motion planning.

- Scheduling and routing: assigns incoming transport requests to available AMRs and computes long-term routes in a graph-based abstraction of the environment, respecting time windows and mitigating traffic congestion.
- Behavioral decision making: chooses among discrete maneuvers (e.g., continue on path, yield, or overtake) based on local traffic context and global objectives.
- Path planning: generates collision-free paths over a mid-term horizon based on full-warehouse maps with static obstacles.
- Motion planning: converts a planned path into time-parameterized trajectories, respecting both the robot's dynamics and the movement of other, nearby dynamic objects.

In RAIL, the entire fleet management and planning stack can be executed on the on-premise cloud, which allows for global reasoning of the entire fleet and environmental state. Again, since many of these sub-tasks are computationally demanding, they benefit from increased computational resources on the cloud.

Onboard autonomy and fail-safe

While the architecture is designed to take full advantage of cloud-based computations, it also accommodates onboard autonomy where needed. In certain environments, onboard sensing may be the only viable option for obtaining accurate, close-range information. Furthermore, onboard capabilities ensure robustness to intermittent connectivity or infrastructure failure. Depending on the specific application and reliability requirements, the onboard stack may range from minimal functionality (e.g., line following) to fully developed autonomous navigation.

Key benefits of RAIL

- Robustness: infrastructure sensors fill perception blind spots to facilitate both localization of the robots and environment perception.

- Scalability: on-premise cloud offers scalable computational resources that are unconstrained by the AMR platform.
- Flexibility: the modular software design allows for updating parts of the stack. Also, the software stack can evolve more or less independently from the AMR hardware since the majority of the algorithms are executed in the cloud.
- Cost efficiency: by offloading computationally demanding algorithms, AMRs can be based on simpler hardware, reducing per-unit cost.

2.3 Industrial evaluation

Paper A presents and evaluates the system in the Volvo Trucks final assembly plant in Tuve, Gothenburg. In the deployment, six robots were transporting mufflers from a pre-assembly area to the assembly line. The 150-meter drive involved navigating in a dynamic environment shared with AGVs, manually operated vehicles and pedestrians, requiring the robots to respond to a changing traffic situation. The cycle time was around seven minutes and up to 130 transportation tasks were completed each day of operation. The remainder of this section describes the implemented details and key takeaways from a User Experience (UX) evaluation.

Overall framework

The architecture of this implementation is shown in Figure 2.2.

The core design principle was to minimize robot complexity and cost by removing expensive onboard sensors and computational units. Localization, perception, and planning capabilities were offloaded to a centralized infrastructure consisting of a local computation cluster and ceiling-mounted cameras. These cameras provide a top-down view of the entire robot-operating area, supporting both robot localization and environmental perception.

The infrastructure cameras stream images over Ethernet to the on-premise cloud. Based on these images, the localization module estimates the position of all AMRs, and the perception module estimates the location of any obstacles in the environment. Based on these results, alongside the transportation requests and a predefined road network, the planning module assigns tasks and

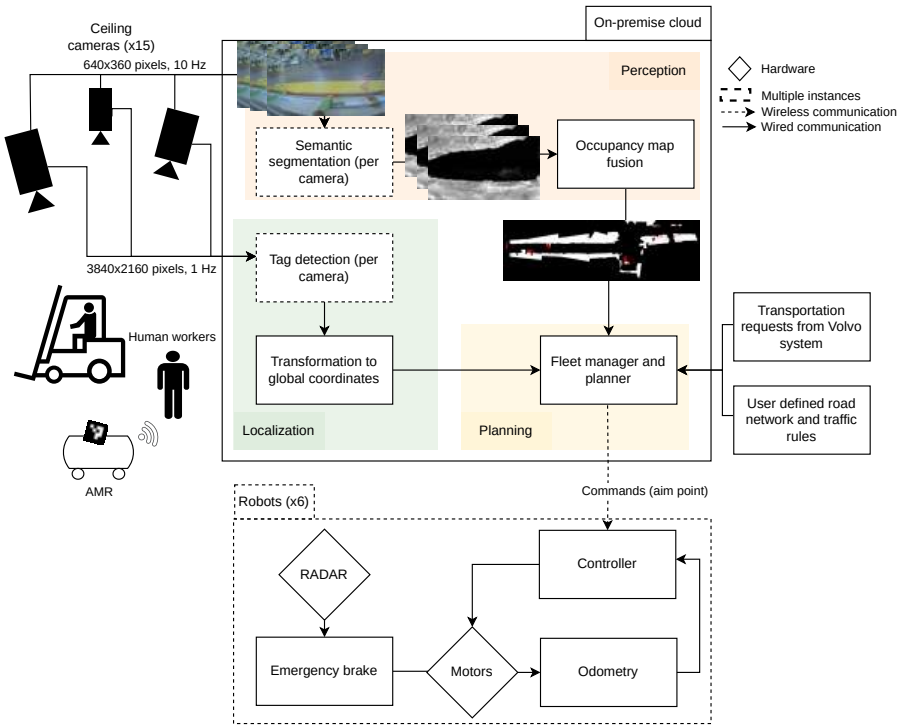

Figure 2.2: System architecture of the industrial implementation at Volvo.

nominal paths to all robots. It then computes *aim points* for all AMRs that are transmitted via Wi-Fi to the robots. The robots execute these plans using low-level controllers and rely on motor odometry for feedback. For safety, an onboard short-range RADAR and a bumper stop activate emergency braking.

Perception

Localization is based on detecting ArUco [7] markers attached to each AMR. Detections from calibrated ceiling cameras are transformed into 3D coordinates, which are fused with motor odometry onboard the robots to provide accurate real-time localization. Environmental perception is achieved through an occupancy map that discretizes the floor into 5×5 cm grid cells, each representing the occupancy status. To this end, binary semantic segmentation (pixel-wise classification of obstacle vs. free space) is first performed separately for each camera to detect obstacles. To establish the correspondence between image-view detections and the occupancy map, the resulting binary masks are projected into a common coordinate frame using camera calibrations. Since the cameras have partly overlapping field-of-view, the transformed masks are then fused into a single occupancy map. Due to budget constraints, the system does not include object tracking or motion prediction, which limits proactive planning but still enables basic obstacle avoidance.

Decision making

The system supports a single transportation task: moving mufflers from a pickup station to the assembly line and returning. Scheduling follows a simple queue-based mechanism, where idle robots wait at the depot until dispatched. Execution is managed by a finite-state machine (FSM) with states defined by application-specific conditions (e.g., kitting status, path clearance, intersection priority). Path planning relies on the A-star algorithm operating over a manually defined road network, while motion control uses a look-ahead point and speed ramp profiles with heading regulation by feedback control. The behavior is intentionally kept simple: robots stop or yield when encountering obstacles rather than re-planning around them. Consequently, human workers are expected to keep paths clear.

Evaluation

Despite the simple design of the perception and decision-making modules, this AMR system has been delivering material within the Volvo plant for almost a year. Drawing from observations and interviews with factory personnel who interact closely with the robots, several insights emerged.

The localization approach based on ceiling-mounted cameras combined with fiducial markers on the robots deliver sufficient precision for industrial use. Ceiling cameras also enable reliable obstacle detections far away from the robots and in densely populated areas, where onboard sensors are susceptible to occlusion. However, ceiling cameras alone cannot guarantee reliable perception in the AMR's immediate vicinity, since the robot may be far away from all cameras. Additionally, object detection from a top-down perspective can be challenging for small objects or items whose appearance blends with the factory floor. These observations suggest that combining onboard and infrastructure-based sensing is a promising direction.

From a behavioral standpoint, forklift drivers reported that the robots' movements can be unpredictable at times, complicating navigation in an already busy and complex traffic environment. Furthermore, instances of traffic deadlock caused by robots obstructing passage were observed. Resolving such cases smoothly requires richer perception beyond a simple occupancy map, including dynamic agent detection and trajectory forecasting, alongside more advanced decision making capabilities. To complement this, clear communication channels between robots and humans, potentially through auditory or visual signals, may also improve coordination and safety.

In summary, while the system demonstrates operational viability, these findings highlight important areas for improvement in perception, behavioral planning, and human-robot interaction to better support safe and efficient co-existence in shared industrial environments.

2.4 Challenges in automating internal logistics

The limitations observed in the industrial evaluation reflect broader challenges in deploying infrastructure-based AMR systems at scale. As identified in Paper A, these challenges span the entire software stack, from localization and perception to fleet management and planning. In addition to these core modules, there are system-level concerns such as network reliability, low-latency

data processing in an on-premise cloud, seamless integration of heterogeneous components, and rigorous safety, security, and privacy assurances. Many subsystems also depend on machine learning, which introduces further hurdles in data collection, model training, validation, and deployment.

Focusing on the core technologies, perception remains fundamentally difficult: cameras and other sensors provide only partial, noisy observations, with occlusions frequently obscuring critical information. The unpredictable behavior of human workers and vehicles introduces additional uncertainty, making it difficult to anticipate how the environment will evolve. Yet anticipating future states is essential for proactive decision making. Fleet management and planning add further complexity. At scale, task allocation and path planning quickly become computationally demanding, while dynamic environments require continuous re-planning to accommodate unexpected events. Designing planners that produce safe and efficient robot behavior in real time, under uncertainty and limited compute budgets, remains a fundamental challenge.

While many aspects of infrastructure-based AMR systems warrant further research, this thesis is dedicated to the perception system. As stated already in Chapter 1, the scope is further limited to perceiving the current state of the environment, excluding object tracking and motion prediction.

CHAPTER 3

Camera-based Perception for Autonomous Mobile Robots

This chapter provides foundational knowledge in modern camera-based perception. It begins with an overview of how deep learning methods, specifically convolutional neural networks and vision transformers, are used to extract meaningful information from images. These concepts are then applied to semantic segmentation and object detection, two fundamental tasks in image interpretation. The discussion then turns to the challenges of interpreting 3D environments using camera-only perception systems. The chapter concludes with an outline of the machine learning life cycle and a summary of key challenges in camera-based perception.

3.1 Deep neural networks for image processing

Computer vision concerns the automatic extraction of meaningful information from visual data such as images or video. In autonomous systems, it enables perception and interpretation of the environment, which are fundamental requirements for navigation, interaction, and decision-making. Traditionally, computer vision relied on algorithmic approaches in which domain experts manually defined features and rules for interpreting visual input. Handcrafted

descriptors such as SIFT [8], HOG [9], Gabor filters [10], and Local Binary Patterns [11] were widely used. Although these methods achieved success in specific tasks, they often struggled to generalize to new or unstructured conditions [12].

The introduction of machine learning, and more recently deep learning, has transformed the field of computer vision [13]. Instead of explicitly programming rules for visual interpretation, models are trained directly on data and learn to extract and combine features relevant to the task, such as detecting pedestrians, identifying drivable surfaces, or recognizing hand gestures. This data-driven approach generally outperforms traditional handcrafted pipelines [13], provided the training data is representative. As a result, machine learning has become the dominant paradigm in modern perception systems.

Among the most influential deep learning models in computer vision are Convolutional Neural Networks (CNNs) [2] and, more recently, Vision Transformers (ViTs) [3]. These models are widely used in applications ranging from object detection and image segmentation to action recognition and visual tracking.

Convolutional neural network

Unlike traditional fully connected neural networks, where every input is connected to every neuron in the next layer, CNNs introduce the concept of local connectivity and shared weights. This is achieved through the convolution operation, in which small, trainable filters are applied to local regions of the input image. These filters scan across the image to detect patterns such as edges, textures, or corners. The key intuition is that combining local features provides a strong basis for image recognition [14]. Moreover, because the same filter is reused across all positions in the image, CNNs naturally achieve translation invariance—the ability to recognize a feature regardless of where it appears. This parameter sharing also dramatically reduces the number of learnable weights compared to fully connected networks, making CNNs both more efficient and less prone to overfitting [14].

LeNet [14] was among the first successful applications of CNNs in computer vision, originally developed for handwritten digit recognition. By applying multiple convolutional filters in parallel, forming a convolutional layer, they extract rich feature maps from the image. By sequentially applying such convolutional layers, the network learns increasingly abstract and complex repre-

sentations [14]. Subsequent research established that the depth of the network is critical for performance [15]. With the advent of powerful GPUs and large-scale datasets, deep CNNs such as AlexNet [15] achieved breakthroughs in the ImageNet [16] challenge, marking a turning point for computer vision.

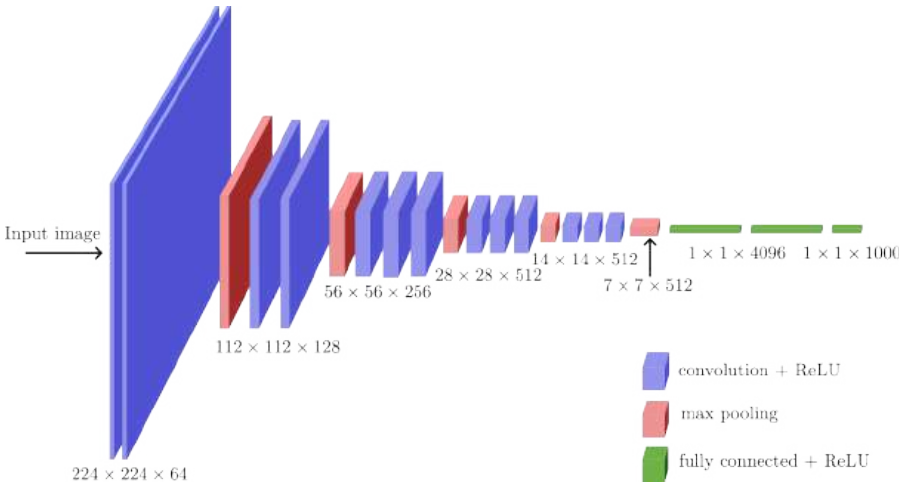

Figure 3.1: Illustration of the VGG16 [17] architecture where each block represents the output of a convolutional, max pooling, or fully connected, layer. The numbers below each block describe the spatial dimension and the number of the channels of the produced feature map. The illustration is taken from [18].

In addition to convolutional layers, modern CNNs include non-linear activation functions (e.g., ReLU) that enable the model to learn complex relationships, and pooling layers that reduce the spatial resolution of the feature maps. A representative architecture for image classification is the VGG16 [17] model, illustrated in Figure 3.1. Here, the network summarizes the global information of the image through many subsequent spatial reductions with pooling layers. As we will see in Section 3.2, CNNs have later been adopted to dense predictions tasks, such as semantic segmentation and object detection. Their ability to learn from raw image data and automatically extract relevant features, alongside computationally efficiency and mature implementation frameworks, has made them a cornerstone of modern perception systems. One of the main strengths with the convolution is the spatial invariance, allowing to extract

similar features across the entire image efficiently. Although a core strength, the local nature of the convolutional operation also presents a challenge in capturing long-range/global information.

Vision transformer

Transformers were originally introduced for machine translation [19], where they achieved state-of-the-art results by modeling sequences of text using the attention mechanisms. At the core of the transformer architecture is scaled dot-product attention. At a fundamental level, this mechanism enables the model to dynamically weigh the importance of different input elements relative to one another, allowing the model to learn the relations between the elements (e.g., words) of the input sequence. To further enhance this capability, transformers employ multi-head self-attention, which allows the model to capture different types of relationships simultaneously.

The self-attention mechanism has since been widely adopted in vision tasks. Most notably, the transformer encoder has been adopted to image data, giving rise to Vision Transformers (ViT) [20] as an alternative to CNNs for feature extraction. As discussed in the next section, the transformer decoder also plays a key role in several vision tasks.

The ViT architecture, illustrated in Figure 3.2, requires a fundamentally different way of representing images than CNNs. Instead of processing a 2D pixel grid, the image is divided into a sequence of non-overlapping patches, which are then flattened and projected into a higher-dimensional space using a linear embedding. These patch embeddings, along with a special classification token and positional encodings, are input to a standard transformer encoder composed of layers of multi-head self-attention and feedforward networks.

The self-attention mechanism allows each patch to attend to every other patch in the image, enabling the model to learn global relationships from the outset. This contrasts with CNNs, where global context is built gradually through stacking many local operations. As a result, Vision Transformers can be more effective in capturing global context [21], which is valuable in tasks involving multiple interacting objects or understanding the overall structure of a scene.

Despite their advantages, ViTs also present certain challenges. One of the most notable is their high data requirement. Since transformers lack the inductive biases inherent in convolutional models (such as locality and translation

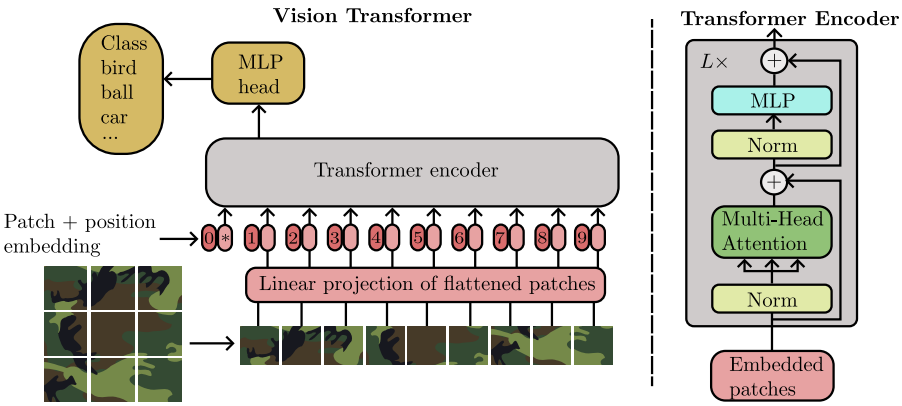

Figure 3.2: Left: Vision Transformer (ViT) architecture [20]. The input image is divided into fixed-size patches, which are linearly embedded and combined with positional embeddings before being passed to the transformer encoder. A learnable classification token (*) is prepended to the sequence, and its output representation is processed by a multilayer perceptron (MLP) to produce class predictions.
Right: Structure of the transformer encoder, consisting of multi-head attention and MLP blocks.

invariance), they typically need large training datasets to generalize well [20]. Furthermore, their computational cost can be higher since the self-attention mechanism scales quadratically with the number of image patches [20]. These limitations led to the development of more efficient transformer variants, such as the Swin transformer [22], which performs self-attention within local windows with linear computational complexity.

3.2 Image analysis – applications

From the industrial application described in Chapter 2, we saw that detecting the free, drivable space, alongside the location of different objects, such as forklifts and humans, is critical for downstream planning of the AMRs movements. Starting from the camera images, detecting free vs. occupied space can naturally be formulated as a (binary) semantic segmentation problem, meaning we aim to classify each pixel of the image as either occupied or free. The general formulation of semantic segmentation that consider multiple classes,

which is the topic of Paper B, will be discussed in the first part of this section. The second part of this section is consequently dedicated to object detection, which aims to localize objects of interest in an image. This forms the basis of Paper C, which considers object detection in 3D. However, the focus of this section is image understanding, which will be extended to 3D understanding in the next section.

Before we continue, it is noteworthy that multiple extensions of object detection and semantic segmentation exist. For example, instance segmentation essentially combines the two by producing pixel-level masks for each detected object in the image. While such detailed information could be relevant in the considered industrial application, such extensions are not the primary focus of this thesis. Instead, we focus on the two fundamental tasks: semantic segmentation and object detection.

Semantic segmentation

Semantic segmentation aims to produce per-pixel classifications. This is typically achieved by having the model predict a categorical probability distribution over a predefined set of classes for each pixel. Each pixel is then assigned the class with the highest predicted score. The most successful models for semantic segmentation today are based on CNNs or ViTs, with comprehensive reviews provided by [23] and [21]. Figure 3.3 shows an example prediction from the MIC segmentation model [24], which will be discussed further in Chapter 4. In the figure, semantic classes are color-coded to aid visual interpretation.

Figure 3.3: Image from the Cityscapes [25] dataset and semantic segmentation predictions of the method ECAP (Paper B).

The first major breakthrough in applying CNNs to semantic segmentation came with the introduction of Fully Convolutional Networks (FCNs) [26].

These networks rely on the convolutional operation all the way from extracting meaningful features from the raw images to predicting the pixel-level class. A key challenge for FCNs is capturing global context while preserving high spatial resolution. This is difficult with the original CNN architectures designed for image classification, as pooling layers progressively reduce feature map resolution. Several innovations have since been introduced to adapt CNNs to dense prediction tasks.

A common approach is the encoder-decoder structure, with U-Net [27] as one of the most influential approaches, illustrated in Figure 3.4. In these models, the encoder gradually reduces the spatial resolution to capture high-level semantic information and long-range context, while the decoder progressively reconstructs a high-resolution segmentation map. Skip connections bridge encoder and decoder stages, combining high-resolution coarse features with low-resolution rich semantic features, thereby enhancing the localization precision. Another key development is the use of dilated (or atrous) convolutions, which increase the receptive field without sacrificing spatial resolution or increasing the number of parameters. To further improve global context modeling, multi-scale pyramid modules have been proposed. For instance, DeepLabv2 [28], which forms the basis for our experiments in Chapter 4, employs parallel atrous convolutions with varying dilation rates to capture information at multiple spatial scales.

Recently, ViTs have been adapted for dense prediction tasks [21]. In its original formulation, the ViT [20] faces two key limitations: high computational cost due to global self-attention scaling quadratically with image size, and low-resolution, single-scale feature outputs, which hinder spatial precision in pixel-wise tasks.

To address these issues, Pyramid Vision Transformer (PVT) [29] introduces hierarchical feature maps in the transformer encoder, enabling both high-resolution coarse and low-resolution fine-grained feature extraction, inspired by the preceding development in CNNs. It reduces the patch size from 16×16 (in ViT [20]) to 4×4 and replaces Multi-Head Self-Attention (MHSA) with Spatial Reduction Attention (SRA) to lower the computational burden. PVT v2 [30] further enhances efficiency by refining the attention mechanism and backbone structure.

Based on the hierarchical feature maps, pixel-level classification can be achieved with a lightweight decoder. For example, SEGFormer [31] uses a

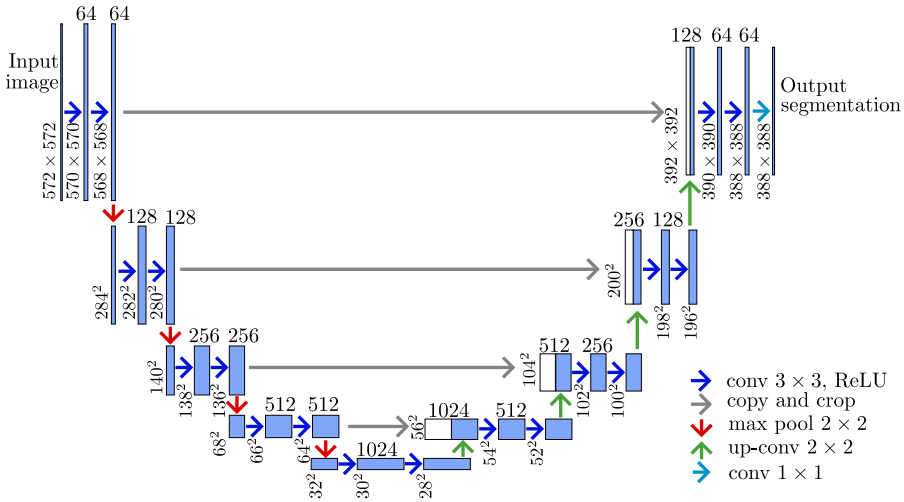

Figure 3.4: Illustration of U-Net [27]. Each block represents the output (feature map) of one of the basic operations. The numbers to the left of each block defines its spatial dimension, and the number above it describes the number of channels.

multilayer perceptron (MLP) as decoder, while other works such as DAFormer [32], which forms the basis of our experiments in Chapter 4, deploys a decoder consisting of a few convolutional layers.

Object detection

In its basic formulation, object detection aims to predict bounding boxes around all objects of interest in an image. Modern detectors build on recent advances in CNNs and ViTs, typically starting with feature extraction using one of these architectures. As in semantic segmentation, it is crucial that extracted features maintain high spatial resolution to ensure precise localization. Based on these features, an object detection head predicts bounding boxes and class labels. Two dominant approaches are anchor-based and transformer-based methods [33].

In anchor-based approaches, the network predicts classification scores and additional attributes over a large set of predefined anchor boxes, with varying sizes and aspect ratios. The model then predicts offsets to adjust the anchors,

enabling more accurate localization. Notable representatives of this class include Faster R-CNN [34], SSD [35], and YOLO [36]. However, these methods typically lack a built-in mechanism for avoiding redundant detections, often producing many overlapping boxes. Therefore, post-processing using Non-Maximum Suppression (NMS) is essential, as illustrated in Figure 3.5.

Figure 3.5: Illustrative example of non-maximum suppression (NMS) in object detection. Anchor-based object detectors may produce overlapping bounding boxes (left), making post-processing with NMS necessary to remove redundant boxes (right). The image is taken from the Wildtrack [37] dataset.

A common class of NMS algorithms is based on greedy selection. They work by first selecting the detection with the highest confidence score, then removing all other detections that overlap significantly with this box. This process repeats for the next highest-scoring box until all detections are processed. Typically, the NMS method is not learned but rather based on heuristics, such as the expected minimum distance between separate objects. Naturally, this process risks suppressing true positives and deteriorating the results, which has motivated research into alternative approaches.

A recent and significant development is the emergence of transformer-based object detectors, which offer a fundamentally different formulation. Rather than generating a dense set of candidate boxes followed by NMS, transformer-based methods treat detection as a set prediction problem with a one-to-one correspondence to ground truth boxes during training. As a result, the model directly predicts a fixed-size set of bounding boxes and learns to suppress duplicates inherently, which eliminates the need for NMS altogether.

A key enabler of these methods is the transformer decoder for object detection, first introduced in DEtection TRansformer (DETR) [38]. As illustrated in Figure 3.6, DETR combines a CNN backbone (e.g., ResNet101 [39]) for fea-

ture extraction with a transformer encoder-decoder architecture. The encoder processes the image features (flattened into a sequence) and adds positional embeddings to preserve spatial information. The decoder uses a set of learnable object queries, which attend globally to the encoder's output to identify objects in the image. Through decoder self-attention, these queries are able to jointly reason about object presence and avoid duplication.

To reduce the computational complexity of the global attention between object queries and encoded image features, Deformable DETR [40] replaced global attention with sparse, deformable attention, allowing each query to focus on a small set of image patches, thus reducing computational cost. In Chapter 5, we will see that such attention-based mechanisms underpin many multi-view detection methods, where image features from multiple cameras must be effectively aggregated.

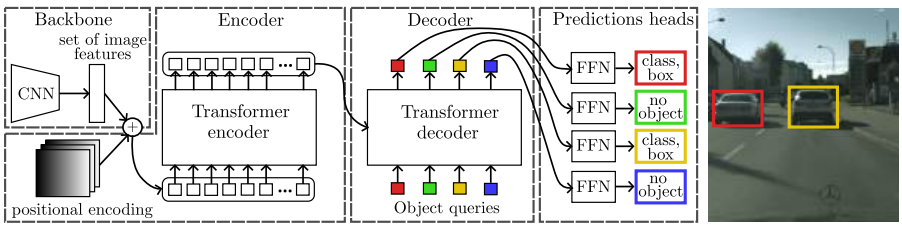

Figure 3.6: Illustration of DEtection Transformer (DETR) [38] with a sample image from the Cityscapes [25] dataset.

3.3 Beyond image understanding: 3D perception

While object detection and semantic segmentation provide a foundation for camera-based perception, they operate primarily in the 2D image plane. For autonomous mobile robots (AMRs) to navigate safely and efficiently, they must also understand the 3D structure of their surroundings. This spatial understanding—identifying where objects are located in three-dimensional space and how they relate to the robot—is critical for collision avoidance, path planning, and interaction with dynamic environments.

To understand how monocular cameras can be used for this task, this section presents the pinhole camera model, arguably the most common model used in computer vision, and how it can be used to infer 3D information from images.

Pinhole camera model

The pinhole camera model [41] is a mathematical abstraction of an idealized pinhole camera, typically conceptualized as a closed box with a single tiny aperture. Through this aperture, light rays from the environment pass and intersect with a planar photosensitive surface, referred to as the image plane. The model defines a unique mapping between a point in 3D space and a point on the image plane: a line is drawn from the 3D point through the optical center (pinhole), and its intersection with the image plane defines the corresponding image point. Every visible 3D point in the scene is thus mapped to a unique location on the image, capturing the effect of perspective projection—a foundational concept in camera-based perception. This section details the mathematical formulation of the pinhole camera model and its role in linking image coordinates with real-world coordinates via intrinsic and extrinsic calibrations.

Coordinate frames and rigid transformation

To model the mapping between 3D world points and image coordinates, we define three coordinate frames:

- World frame: A 3D point is expressed by Euclidean coordinates as $\mathbf{X}_w = [X, Y, Z]^\mathsf{T}$.
- Camera frame: The same point is represented relative to the camera as $\mathbf{X}_c = [X_c, Y_c, Z_c]^\mathsf{T}$. The origin of this coordinate frame is at the optical center of the camera. By convention, the Z-axis points forward along the optical axis, while the X and Y axes align with image columns and rows.
- Image frame: Defined in pixel coordinates on the image plane as $\mathbf{p} = [u, v]^\mathsf{T}$.

The transformation between the world frame and the camera frame is expressed as

$$\mathbf{X}_c = \mathbf{R}\,\mathbf{X}_w + \mathbf{t}, \tag{3.1}$$

where $\mathbf{R} \in \mathbb{R}^{3\times3}$ is a rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is a translation vector that describes the displacement between the two coordinate frames.

Thereafter, normalized image-plane coordinates are obtained by perspective division:

$$x = \frac{X_c}{Z_c}, \quad y = \frac{Y_c}{Z_c}. \tag{3.2}$$

To convert (x, y) into pixel coordinates (u, v), the intrinsic camera matrix $\mathbf{K}$ is applied through

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.3}$$

where f_x and f_y are the focal lengths (in pixels) and (c_x, c_y) is the principal point. Putting it all together, the full forward projection is expressed as

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}] \begin{bmatrix} \mathbf{X}_w \\ 1 \end{bmatrix}, \tag{3.4}$$

where s is an arbitrary scale factor.

Calibration methods

To apply the pinhole camera model in practice, the parameters $\mathbf{K}$, $\mathbf{R}$, and $\mathbf{t}$ must be determined through camera calibration. A widely used method is proposed by [42], which estimates both intrinsic and extrinsic parameters from multiple images of a planar calibration object with known geometry, such as a checkerboard. The method begins with a closed-form solution and refines it through nonlinear optimization by minimizing reprojection error—the discrepancy between observed image points and those predicted by the model.

Real cameras exhibit lens distortion, mainly radial and tangential distortion, which can be estimated during calibration and corrected for. Once intrinsic parameters, extrinsic pose, and distortion coefficients are available, the pinhole model provides a closed-form mapping between 3D scene geometry and image measurements. However, this idealized model disregards multiple aspects of real cameras. For example, the finite aperture size causes light from a 3D point to be distributed over an area rather than focused to a single point, and pixel intensities represent light accumulated over finite sensor areas. Despite this, the pinhole camera model provides sufficiently accurate results in many real-world applications.

Inferring 3D information

While (3.4) relates each world coordinate to a unique image point, the reverse is not true. Instead, a given image point may correspond to any 3D point along a ray extending from the camera's optical center, a property reflected by the arbitrary scale factor s. As such, recovering a 3D point from its image coordinate requires knowledge of depth, since the value of Z_c is lost in projection. Unfortunately, depth information is not directly available in RGB images, making inferring 3D properties from monocular images alone an inherently ill-posed problem. Despite this challenge, recent advances in machine learning have demonstrated impressive results. The problem can also be alleviated by leveraging prior knowledge, such as planar constraints, or by applying multi-view geometry. Figure 3.7 illustrates three common setups for inferring 3D information from cameras.

The first, and most direct, approach to infer the depth of an image is based on monocular depth estimation. Here, deep machine learning has enabled the development of methods that can predict per-pixel depth maps from monocular inputs by learning statistical priors from large datasets [43]. Based on an estimated depth Z_c of a pixel with coordinates (u, v), the corresponding 3D coordinate in the camera frame X_c and world frame X_w can be computed by

$$\mathbf{X}_c = Z_c \mathbf{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, \quad \mathbf{X}_w = \mathbf{R}^\mathsf{T}(\mathbf{X}_c - \mathbf{t}). \tag{3.5}$$

Although recent advances in machine learning have led to models with impressive performance, monocular depth estimation remains a challenging task, particularly in complex scenes with frequent occlusions [43].

The second approach uses planar constraints to resolve the depth ambiguity. This is viable when for example, the object of interest is known to move on the ground which can be accurately approximated by a plane in 3D. If a pixel point $\mathbf{p} = [u, v, 1]^\mathsf{T}$ corresponds to a point on a plane with normal vector $\mathbf{n}$ and an offset d_0 (in the camera frame), it back-projects along the ray $s\mathbf{d} = s\mathbf{K}^{-1}\mathbf{p}$. The intersection with the plane satisfies $s\mathbf{n}^\mathsf{T}\mathbf{d} + d_0 = 0$. From these expressions, the depth s can be solved as

$$s = -\frac{d_0}{\mathbf{n}^\mathsf{T}\mathbf{d}}. \tag{3.6}$$

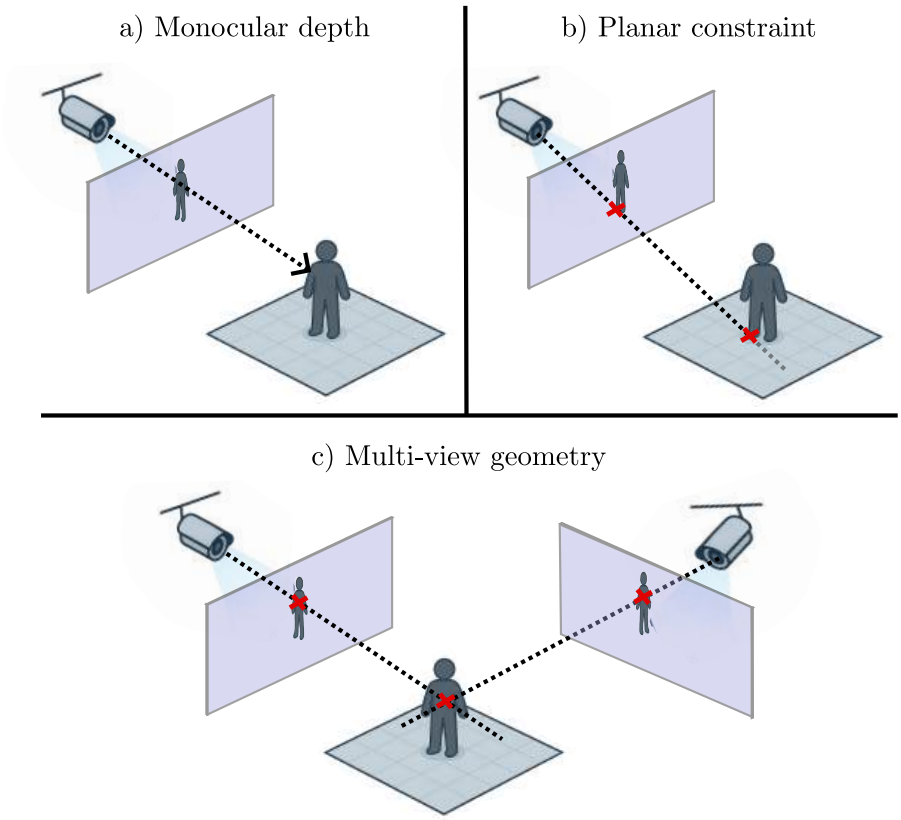

Figure 3.7: Different paradigms for inferring 3D structure with cameras: a) learning based methods can infer the depth of arbitrary pixels, b) and c) use planar and multi-view constraints to resolve the depth ambiguity.

Once the depth is known, (3.5) can be used to transform the pixel coordinate to a 3D point in the world frame.

The third approach leverages multi-view geometry, where multiple overlapping camera views are used to triangulate 3D points. If the same object is detected in at least two images, the corresponding 3D point must satisfy the projection (3.4) of both cameras. Due to noise in detections and calibration inaccuracies, two or more cameras form an overdetermined system with no exact solution. The 3D point $\mathbf{X}_w$ is typically estimated by minimizing the

reprojection error across views.

Many modern computer vision systems, including the AMR system presented in Chapter 2, incorporate overlapping camera views and operate in environments where the ground can be approximated as a plane. These systems are therefore naturally suited to all three approaches described above.

In Chapter 5, dedicated to multi-view 3D perception, we will see that modern learning-based methods often integrate these principles to create more robust systems. For example, monocular depth estimation may be applied independently in each view, and associations across views may be learned implicitly, allowing for enhanced accuracy.

3.4 Machine learning life-cycle

The development of machine learning (ML) systems for computer vision is an iterative and multi-stage process that extends well beyond model design. The supervised learning paradigm remains the most widely adopted approach, and thus shapes the structure and requirements of most modern vision-based ML pipelines. Figure 3.8 illustrates the main components of the ML life-cycle, including data collection, annotation, model training, deployment, and performance monitoring.

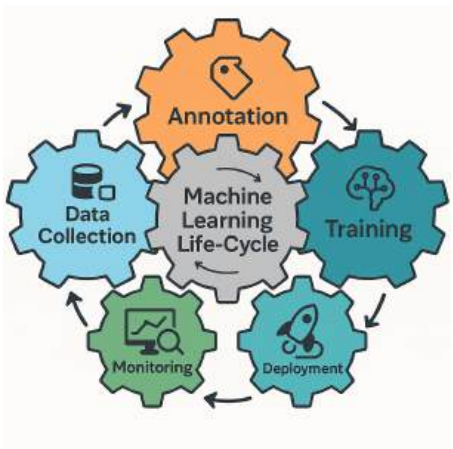

Figure 3.8: Machine learning life-cycle.

Training – Supervised Learning Supervised learning assumes access to a dataset composed of input–output pairs, where the input is typically an image and the output is a structured label, such as a bounding box or a pixel-wise semantic category. During training, the model, typically consisting of a feature extractor g and a task-specific prediction head f, takes an input image x and produces a prediction $\hat{y} = f \circ g\,(x)$. This prediction is compared to the corresponding label y via a loss function $\mathcal{L}(\hat{y}, y)$. The model's parameters (in both g and f) are optimized through backpropagation (based on gradient descent) to minimize the loss over the training dataset. The goal is to learn a mapping from inputs to outputs that generalizes well to unseen data.

This setup introduces several assumptions and requirements. First, the training data must accurately represent the conditions under which the model will be deployed. Second, labeling must be consistent and precise to prevent systematic biases. Third, training, validation, and test data must be clearly separated to enable objective evaluation. While conceptually straightforward, these requirements become increasingly complex in large-scale or safety-critical systems.

Data Collection and Annotation Data collection is the first major step in the ML life-cycle and directly determines the system's potential performance. In computer vision, this involves capturing images or videos that reflect the environments, scenarios, and tasks the model is intended to handle. An essential consideration is data coverage: ensuring that the dataset spans the full spectrum of conditions the model may encounter. Insufficient diversity in the data can lead to poor generalization and failure in deployment. For example, if the dataset underrepresents certain object classes, backgrounds, or viewpoints, the model is likely to perform poorly on those cases. Therefore, sufficient data collection is crucial for model performance and is often a time consuming and costly endeavor.

Once data is collected, it must be annotated with task-specific labels. In object detection, this typically involves drawing bounding boxes and assigning class labels; in semantic segmentation, pixel-wise annotations are required. These tasks are labor-intensive and often require manual inspection to ensure quality and consistency. Beyond initial training, labeled data is also essential for model testing and validation. A well-constructed test set must remain completely isolated from the training process and should include both common and rare cases, ideally capturing edge scenarios that challenge the model's

robustness.

Due to the high cost of data collection and annotation, simulation and synthetic data have emerged as powerful tools to supplement or replace parts of the data pipeline. Simulated environments can produce vast amounts of labeled data with perfect ground truth, and allow for systematic variation of conditions such as lighting, weather, object configuration, and dynamic interactions. Synthetic data also enables the generation of rare events, such as near-collisions or sensor failures, which are difficult or dangerous to capture in reality.

Despite these advantages, synthetic data introduces the challenge of the reality gap: models trained on simulated images often struggle to generalize to real-world data due to subtle but impactful differences in appearance, noise, and dynamics. However, with carefully designed training strategies, this gap can be bridged. As explored further in Chapter 4, synthetic data can be a valuable asset for real-world applications.

Deployment and Monitoring Deployment is a major milestone in the ML life-cycle, but it does not mark the end of system development. In dynamic environments, model performance may degrade over time due to changing conditions, sensor drift, or unanticipated scenarios. Consequently, continuous monitoring and periodic retraining on newly collected data are often necessary to maintain system reliability.

A key challenge in deploying ML models in dynamic environments is the issue of uncertainty. Uncertainty in ML models can be broadly categorized into two types [44]: aleatoric (data-related) uncertainty, which arises from inherent noise or ambiguity in the data (e.g., occlusions, motion blur), and epistemic (model-related) uncertainty, which reflects the model's lack of knowledge due to insufficient or unrepresentative training data. Reliable uncertainty estimation is essential for building systems that can reason about their own limitations. For instance, high-uncertainty predictions can be flagged for human intervention, ignored, or used to trigger re-planning in a robotic system. Moreover, uncertainty estimates can guide data collection by highlighting areas where the model lacks sufficient training data.

3.5 Challenges in camera-based perception

This chapter has introduced fundamental components of machine learning for computer vision. It began by outlining the importance of CNNs and ViTs for automatic feature extraction, followed by a discussion of object detection and semantic segmentation as two core image analysis tasks. We then explored how 3D information can be inferred from 2D images. Finally, we examined the machine learning life-cycle and highlighted practical challenges. Based on this foundation, the key challenges in camera-based perception that motivate the investigations in the following chapters are now summarized.

A central challenge is the heavy reliance on large labeled datasets, inherent to the supervised learning paradigm. Real-world data collection is often costly, time-consuming, and difficult, especially in safety-critical domains. While simulation offers a scalable alternative, it introduces its own challenges. Differences in appearance and behavior between synthetic and real-world data hamper model generalization. These limitations underline the need for machine learning methods that relax data requirements and can leverage simulated or unlabeled real-world data to cut collection and annotation costs. This motivates the formulation of RQ2: *How can machine learning models be trained on datasets with few labels for computer vision tasks?* Chapter 4 addresses this question by investigating methods that reduce the reliance on labeled data.

Another critical challenge lies in uncertainty estimation. Both aleatoric and epistemic uncertainty affect the trustworthiness of predictions. At deployment, uncertainty estimates support decision-making by identifying unreliable outputs. During training, they highlight uncertain cases that may benefit from additional supervision. This is particularly important in self-supervised settings, where inaccurate predictions could reinforce errors. This issue is further discussed in the next chapter.

Finally, while monocular camera systems perform well in tasks like object detection and segmentation, they are fundamentally limited in recovering 3D structure due to depth ambiguity, occlusion, and scale uncertainty. As detailed in Section 3.3, multi-camera systems can alleviate these limitations by enabling geometric reasoning across views. This leads to **RQ3**: *How to design ML methods that leverage multiple cameras for increased performance and robustness in computer vision tasks?* Chapter 5 explores ML methods designed for 3D reasoning with multi-camera systems.

CHAPTER 4

Learning with Limited Labeled Data

Since creating real-world labeled datasets is time consuming and costly, there is a great incentive to reduce machine learning methods' dependence on such datasets. In doing so, the data collection and annotation step can be simplified, allowing for faster and more cost effective development of machine learning models in industry and society. This chapter starts with outlining various methods that address this problem. Next, specific focus is given to a specific problem formulation known as Unsupervised Domain Adaptation (UDA). Thereafter, a class of methods known as self-training is investigated. Finally, a case study on unsupervised domain adaptive semantic segmentation is presented.

4.1 Approaches

Transfer learning is one of the most widely used techniques to reduce reliance on labeled data. The principle is straightforward: a model trained on one task often learns features that are useful for another. For example, image classification models trained on large-scale datasets like ImageNet [16] are commonly reused for downstream tasks through fine-tuning. In this process, the feature

extraction backbone is initialized with pre-trained weights, allowing the model to adapt to a new task using fewer labeled examples. This reuse of learned representations significantly reduces the need for large annotated datasets in subsequent tasks [45].

More recently, the emergence of foundation vision models has further advanced this trend by providing feature representations that generalize better across multiple domains and applications. These models typically employ massive computational resources and Vision Transformers (ViTs) with billions of parameters, and are often pre-trained using large-scale image or image-text datasets. Inspired by the success of large language models, which leverage masked language modeling [46] to learn to reconstruct missing parts of text in a self-supervised manner, similar methods have been proposed in computer vision. For example, iBOT [47] learns to recover information in masked image patches through masked image modeling, and CLIP [48] uses contrastive learning to predict which caption goes with which image using text-image pairs collected from the web. Meanwhile, the size of task-specific datasets is also increasing. Segment Anything [49] was trained on over one billion segmentation masks. While foundation models provide powerful starting points, they are not universal solutions: many real-world applications still require domain-specific fine-tuning and curated datasets to achieve high performance.

Given that tailored datasets remain necessary, several strategies aim to reduce the cost and effort of annotation. In semi-supervised learning, models are trained using both labeled and unlabeled data drawn from the same distribution [50]. A typical scenario involves collecting a large dataset (e.g., from cameras) and labeling only a small portion, while dedicated training methods leverage both labeled and unlabeled samples. Complementary to this, active learning can be used to identify and annotate the most informative or uncertain examples, maximizing annotation efficiency [51]. Few-shot learning [52] represents another strategy, enabling models to adapt to new tasks with very limited supervision, which is especially valuable in domains where labeled data is inherently scarce or costly to obtain.

In contrast to these methods, Unsupervised Domain Adaptation (UDA) aims to train models without any labeled data from the deployment (target) domain [4], making it particularly attractive for real-world applications where annotation is expensive or infeasible. Instead, UDA relies on labeled data from a related domain (the source domain) and unlabeled data from the tar-

get domain. This differs from semi-supervised learning, where labeled and unlabeled data are drawn from the same distribution (domain). A common application of UDA is bridging the domain gap between simulated and real-world environments by using labeled simulator data together with unlabeled real-world data. The goal is to train a model that performs well on the target domain by leveraging both datasets. Variants such as source-free UDA [53], which removes access to the source dataset, and domain generalization [54], which trains models to generalize without any target domain data, extend this idea. While these are promising directions, this thesis focuses specifically on UDA.

4.2 Unsupervised domain adaptation

In Unsupervised Domain Adaptation (UDA), a dataset $\mathcal{D}^S$ from the source domain including N^S image-label pairs is available $\mathcal{D}^S = \{x^{S,k}, y^{S,k}\}_{k=1}^{N^S}$, where $x^{S,k}$ denotes an image from the source domain and $y^{S,k}$ denotes the corresponding label. Additionally, a set of N^T unlabeled target domain images are available $\mathcal{D}^T = \{x^{T,k}\}_{k=1}^{N^T}$. It is assumed that the two domains are related, although, yet statistically different. That is, $x^{S,k}$ and $x^{T,k}$ are sampled from different underlying distributions. The challenge lies in training the model on the datasets $\mathcal{D}^S$ and $\mathcal{D}^T$ to achieve satisfactory performance on new test images from the target domain. Naturally, the larger the gap between the two domains, the more challenging it is to learn a model that can generalize across the domains.

UDA methods have been developed for various computer vision tasks, including image classification, segmentation and object detection. Simulation-to-real (sim-to-real) adaptation, which is illustrated in Figure 4.1, is among the most common UDA settings. The reason is that labels can often be easily collected in simulation, and by using UDA, no real-world labels are required. Typically, the appearance of simulated images is very different from real-world images, resulting in a significant domain gap. However, advancements in generative AI, such as diffusion models [56] and generative adversarial networks (GANs) [57], and reconstruction techniques, such as Neural Radiance Fields (NeRFs) [58] and 3D Gaussian Splatting [59], are narrowing the gap between simulated and real-world domains. While these improvements help, residual discrepancies remain, which continues to motivate the development of UDA

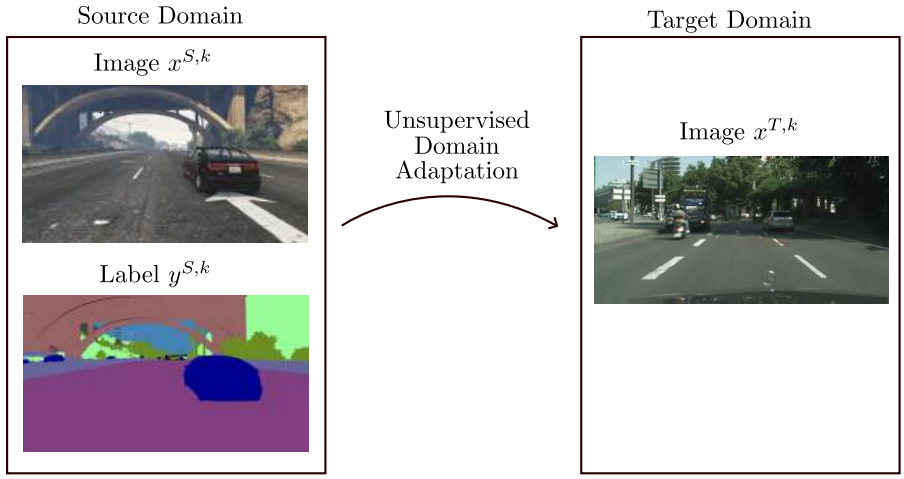

Figure 4.1: Example UDA application: semantic segmentation using the labeled simulated dataset GTA V [55] as source domain and real-world images from Cityscapes [25] as target domain.

methods. Importantly, UDA also extends beyond sim-to-real scenarios. Differences in camera hardware or environmental factors may create domain shifts in real-world datasets.

The majority of UDA methods can broadly be categorized as based on either *adversarial training* or *self-training*. In adversarial training, a domain discriminator provides supervision in a GAN framework to align domain invariant input, features, or output. A notable example is CyCADA [60], which applies adversarial training in both input and feature space. In their approach, a style-transfer network transforms the source images to mimic the appearance of the target images. The style-transfer network is trained to *fool* the domain discriminator about the domain, while the domain discriminator is trained to accurately determine from which domain the input image originates from. If implemented correctly, the style-transfer network can accurately change the appearance of the source images to be more *target-like*, and thus reduce the domain gap. Similarly, the model used for the end task (e.g., image classification, object detection, or semantic segmentation) may be trained to produce domain invariant features. Lastly, invariance may also be enforced in the output [61]. Here, the assumption is that the marginal distribution of the labels

is similar across domains, only the appearance of the images are different. Therefore, enforcing that e.g., object frequency and shapes predicted by the model, to be similar across domains can improve generalization.

Self-training, on the other hand, involves letting the model supervise itself. I.e., typically it makes predictions on the target domain samples, and then uses these predictions later during training. To make this approach feasible, multiple training tricks are used, which are further described in the next section. As self-training is the focus of this thesis, we dedicate the next section to this topic. For a comprehensive discussion around other UDA methods, the reader is referred to the recent surveys [4], [62].

4.3 Self-training

Self-training is a method aimed at utilizing unlabeled data during training, thus reducing the requirement on large, labeled datasets for supervised training. The core idea of self-training is to use the model's own predictions to generate artificial labels for unlabeled target data, enabling subsequent training of the model on this target data using standard supervised techniques. Due to its general formulation, self-training has been applied across various computer vision tasks, including image classification [63]–[66], semantic segmentation [67]–[69], and object detection [24], [70], achieving success in both unsupervised domain adaptation (UDA) [24], [67], [71] and semi-supervised learning [63], [65], [72]. The artificial label can be either soft, meaning it is represented as a probability distribution output by the model [64]; or hard, where the probabilistic output is reduced to a single label, typically the most likely class. Such hard labels are commonly called pseudo-labels [72], and have been the dominating approach in recent years [24], [32], [65], [67]–[69], [71].

Although self-training has gained traction due to its broad applicability and often strong performance, it also presents inherent limitations. Notably, it suffers from a chicken-and-egg problem: high-quality pseudo-labels are essential for effective adaptation, yet such pseudo-labels can only be expected **after** successful adaptation has occurred. In effective implementations, the quality of pseudo-labels and the model's performance improve progressively during training. These methods enhance training effectiveness mainly through improvements in pseudo-label generation, management of erroneous pseudo-labels, and data augmentation. The remainder of this section discusses these three critical aspects.

Pseudo-label generation

Modern computer vision models typically output uncertain, rather than deterministic, predictions. For example, in image classification and segmentation, the model commonly outputs a categorical distribution over classes for the whole image or each pixel, respectively. For object detection, the output may indicate the probability that a bounding box contains an object. Since the true label corresponds to a specific class or bounding box, post-processing is necessary to convert these predictions into pseudo-labels. Formally, pseudo-label creation can be expressed as

$$\tilde{y}^T = h(f_\phi(x^T)), \tag{4.1}$$

where x^T is an unlabeled target sample, f_ϕ is the model, h is the post-processing function, and $\tilde{y}^T$ is the generated pseudo-label.

For image classification and semantic segmentation, labels are typically in one-hot format, while model outputs are probability mass functions. By selecting the most likely class for the image or each pixel, a pseudo-label is produced.

For object detection, pseudo-labeling often requires extra post-processing. As described in the previous chapter, many detectors use non-maximum suppression (NMS) to extract a set of predicted objects from model outputs. This process depends on several handcrafted thresholds, such as confidence and NMS-distance thresholds. The quality of the pseudo-labels is therefore sensitive to these hyperparameters, which can be particularly challenging to tune in UDA scenarios where the model's confidence on the target domain is initially unknown.

Because self-training effectiveness depends heavily on pseudo-label quality, numerous techniques have been proposed to improve pseudo-label generation. A common approach involves averaging multiple predictions—for example, multiple stochastic forward passes [73], multiple forward passes over augmented inputs [74] or model ensembles [75]. Inspired by these methods, the prevalent approach today is the mean teacher [63]. In this setup, a teacher model generates pseudo-labels, while a student model trains on them using supervised loss. The teacher has the same architecture as the student but updates its parameters ϕ as an exponential moving average (EMA) of the

student's parameters θ:

$$\phi_{t+1} \leftarrow \alpha\phi_t + (1-\alpha)\theta_t, \tag{4.2}$$

where $\alpha \in [0, 1]$ controls the update rate and t denotes the training step. The teacher acts as a temporal ensemble of the student, stabilizing training since it evolves more slowly and generally provides better predictions.

The teacher may additionally have privileged knowledge unavailable to the student. For instance, the student may operate on low-resolution images for increased computational efficiency while the teacher generates pseudo-labels from high-resolution images to improve quality. In more complicated setups, the student may consider separate images from a single camera, while the teacher takes video input from a single or multiple cameras. The pseudo-labeling pipeline can also incorporate complex components, such as multi-object tracking algorithms or foundation vision models to guide the labeling process.

However, a key characteristic of self-training is the closed feedback loop between the model and pseudo-label generation: as the model improves, so do the pseudo-labels. This loop may not exist if the teacher and student differ substantially—for example, using entirely different input modalities or independent pseudo-labeling methods. In such cases, the process is better described as knowledge distillation or label transfer rather than self-training. Nonetheless, if possible, providing the teacher with privileged information can greatly benefit self-training. This topic will be revisited in Chapter 5 when discussing multi-view methods, illustrating how teacher and student models may access different data sources.

Managing erroneous pseudo-label

Regardless of the methods used to generate pseudo-labels, it is inherent in the self-training problem formulation that pseudo-labels contain errors. If the pseudo-labels were already perfect, there would be no need to further train the model on unlabeled data. Therefore, handling erroneous pseudo-labels is a core challenge in self-training.

Before delving into different methods for managing this noise, it is important to understand its characteristics and where it comes from. As mentioned in Section 3.4, uncertainty in machine learning models' predictions can be clas-

sified as aleatoric and epistemic. Aleatoric uncertainty, which is irreducible, is inherent in the data and problem setup. For example, it can arise due to inaccurate source labels, and the ambiguity of object boundaries due to limited image resolution. Meanwhile, epistemic uncertainty relates to model deficiencies and insufficient training. In the UDA setting, the epistemic uncertainty is initially large since the target data is out-of-distribution, which significantly affects pseudo-label reliability.

To allow for further investigation, the label errors may be classified as either structured or unstructured [76]. Unstructured label noise refers to (seemingly) random, input-agnostic flips that are equally likely across all classes. While such noise naturally degrades performance, it doesn't necessarily change the model's overall behavior considerably. In contrast, structured noise are concentrated in particular regions of the input/feature space and follow nonuniform class-flip patterns. In a UDA scenario this means that certain types of target-domain examples—say, pedestrians in heavy shadows or unusual poses—are not only more likely to be mislabeled, but are systematically mapped to specific wrong classes. Such bias exacerbates the domain gap and must be treated carefully to enable successful adaptation. Figure 4.2 shows an example of structured noise in pseudo-labels for semantic segmentation, where fences were confused with trains.

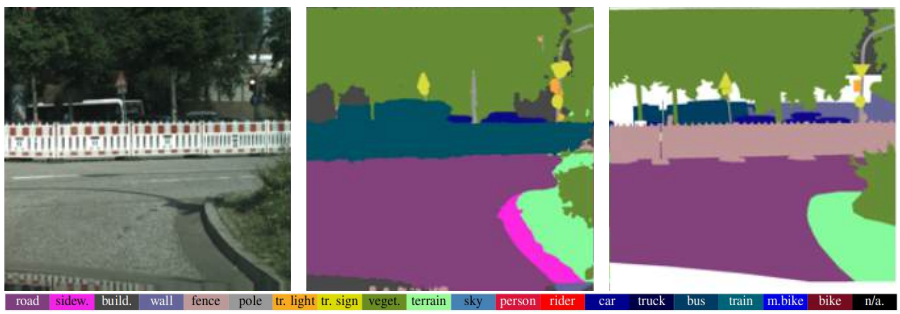

Figure 4.2: An image (left) from Cityscapes [25] for which the pseudo-label (center) differs significantly from the label (right).

Various techniques have been proposed to deal with the noise in the pseudo-labels explicitly. For example, it is common to filter pseudo-labels based on predicted confidence, and only use those that exhibit high enough confidence [77]. Alternatively, a weighted loss function may be used to assign

higher importance to confident pseudo-labels during training [74]. Since predicted confidence is often inaccurate for data that is out of distribution, which is the case for the target domain, a challenge is also to accurately quantify the uncertainty of these labels. To this end, formal methods for uncertainty quantification in deep learning, of which [78] provides a review, may be needed to accurately predict the reliability of the pseudo-labels.

Moreover, an inherent problem with focusing training on reliable pseudo-labels is that these typically correspond to particularly easy training samples. Excessive training on such easy samples may not lead to improved performance on difficult samples. Furthermore, in multi-class problems, the difficulty often varies across the classes. Therefore, confident pseudo-labels often originate from a set of *easy-to-adapt* classes. Focusing training on these may prohibit efficient learning of the *hard-to-adapt* classes. Many methods therefore propose learning techniques dedicated for maintaining class-balance during training, e.g., by using class-wise confidence thresholds when generating/filtering pseudo-labels [74], [77], [79]. It is also common to use curriculum learning that successively increase the difficulty of the training examples [80].

In the end, this situation is a clearer version of the original chicken-and-egg problem: we can only create accurate pseudo-labels for the easy samples, which the model already handles well. The hard examples, where the model needs the most help, do not get reliable labels. Because of this, self-training risks becoming a cycle where the model keeps improving on easy cases but does not get better at the difficult ones. To overcome this, it is important to find ways to provide reliable guidance for the hard examples so that the model can learn and adapt to all levels of difficulty.

Data augmentation

Data augmentation has been used extensively in a related class of methods is called *consistency regularization* [81], where the idea is that the model's predictions on multiple augmented variants of the unlabeled target data should be similar. Later, self-training methods based on pseudo-labeling adopted similar techniques. The pseudo-labels are typically generated from predictions on unaugmented samples, which are then used to supervise the model's prediction on a strongly augmented variant of the sample [65], [67]. Not only can data augmentation improve the model's generalization capability by generating diverse data, but it can also transform *easy* examples to *hard* examples. Hence,

data augmentation can act as a means of mitigating the inherent problem of pseudo-labeling methods; that accurate pseudo-labels are typically only available for particularly easy examples. Instead, data augmentation can enable efficient training on hard examples with accurate pseudo-labels.

Common data augmentation methods include geometric transformations, such as image cropping and flipping [82], alongside visual augmentations, such as image blurring and color jittering [65]. Recently, masking out certain regions of the image have been used as a strong augmentation technique to enhance the learning of context [24]. Beyond augmenting the target data samples, data augmentation has also been used as means to bridge the two domains through mixing, in particular for the task of semantic segmentation. Here, many methods employ the data augmentation proposed by DACS [67], which is illustrated in Figure 4.3. DACS first selects half of the classes in the source image x^S, as indicated by the selection mask q^M, and then cut-and-pastes the corresponding pixels onto the target image x^T. The same is done to the labels y^S and $\tilde{y}^T$, resulting in a mixed image x^M and label y^M that include information from both domains. Training on such mixed samples can hinder the network from learning the two domains separately. Instead, the network is forced to treat them jointly, which promotes learning domain invariant features.

4.4 Case study 1 – semantic segmentation

Paper B is focused on semantic segmentation of images captured by a monocular camera mounted on a car. In the AMR system presented in Chapter 2, similar methods can help the robot perceive its environment independently, without relying on infrastructure. As explained in Chapter 3, modern semantic segmentation models are based on CNNs or ViTs and require large labeled datasets for training. Since it is time-consuming and expensive to create the required pixel-level annotations, research into semantic segmentation methods with reduced data demand is motivated. To this end, Paper B studies Unsupervised Domain Adaptive (UDA) semantic segmentation.

One of the most widely used benchmarks for UDA semantic segmentation is GTA → Cityscapes. As indicated by the arrow, the GTA [55] dataset serves as the labeled source domain and the Cityscapes [25] dataset serves as the unlabeled target domain. Notably, this is a *sim-to-real* adaptation benchmark

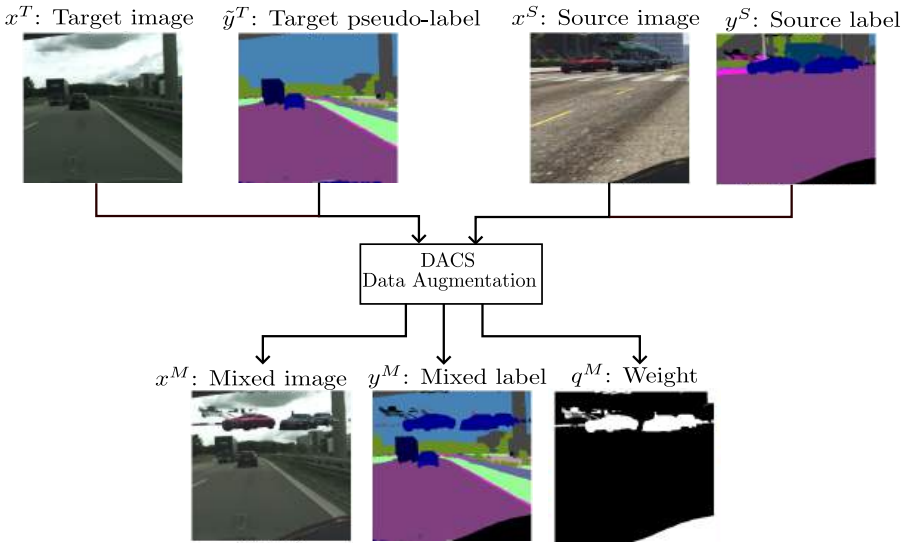

Figure 4.3: Illustration of the domain-mixing data augmentation introduced by DACS [67].

as GTA is simulated while Cityscapes constitutes real-world data. Example images from this benchmark are shown in Figure 4.1. Table 4.1 illustrates the performance of two popular architectures: the CNN-based DeepLabV2 [28] and the transformer-based SegFormer [31]. Both struggle significantly with the domain gap between simulated and real-world data, as indicated by the large performance drop from the oracle (trained with target labels) to source-only (trained only with GTA). This motivates research on UDA methods.

Table 4.1: Performance in mIoU on the Cityscapes validation set after training on the GTA (Source only) or Cityscapes training set (Oracle). These results are taken from [32].

Architecture	Source only	Oracle
DeepLabV2 [28]	34.3	72.1
SegFormer [31]	45.6	76.4

Several recent methods, such as DACS [67], DAFormer [32], HRDA [69], and MIC [24], build on self-training and employ cut-and-paste augmentation

to help overcome the domain gap. The training pipeline, first introduced by DACS and illustrated in Figure 4.4, operates in two branches. In the target branch, a teacher model f_ϕ processes the target image x^T to create a pseudo-label $\tilde{y}^T$. These are then combined with a source image x^S and the associated label y^S in the DACS augmentation module, which was introduced in the previous section, to create a mixed training sample consisting of a mixed image x^M, mixed label y^M, and weight q^M. The student f_θ processes the mixed image and its prediction is compared against the mixed pseudo-label under the target-domain loss $\mathcal{L}^T$. In parallel, the student is supervised on labeled source data using a source-domain loss $\mathcal{L}^S$. Finally, the two losses are combined, and the student parameters are updated by backpropagation, while the teacher parameters is updated as an exponential moving average of the student.

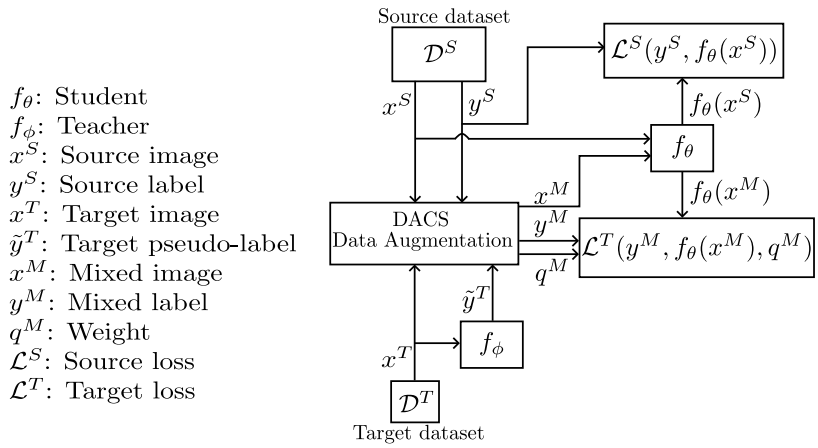

Figure 4.4: Illustration of the DACS [67] self-training framework, which has been used by many subsequent works and forms the basis for the method presented in Paper B.

To mitigate noise in pseudo-labels, $\mathcal{L}^T$ is a weighted cross-entropy loss that manage the influence of each pixel by the weight q^M, which is also produced by the DACS augmentation module. Pixels from the source have weight 1, while target pixels are weighted by the fraction of pixels exceeding a confidence threshold τ. As a result, the loss will be dominated by source content early in training, when confidence on the target is low. As pseudo-label confi-

dence increases, the model increasingly uses target data. However, this scheme does not account for variations within individual target images, meaning that uncertain and erroneous regions in an otherwise confident pseudo-label still makes undesirable contributions to the loss function.

Paper B highlights that this framework is sensitive to pseudo-label errors. Experiments with DAFormer on GTA → Cityscapes demonstrate that about 38 % of the total loss during the last 50 iterations originates from erroneous pseudo-labels. To measure this effect, we introduce DAFormer (denoise), which uses the true target labels to zero out loss on incorrect pseudo-label pixels. Experiments on GTA → Cityscapes verify that DAFormer performs significantly worse than DAFormer (denoise), with an absolute performance drop of 4 mIoU, demonstrating the negative impact of pseudo-label noise.

By further analyzing the pseudo-labels of state-of-the-art methods, it becomes evident that the quality varies significantly over the samples in the target dataset. Importantly, both high and low quality pseudo-labels exist for all classes. Figure 4.5 illustrates two examples of the class rider: one misclassified as a car and one accurately segmented. This suggests opportunities to improve training by emphasizing high-quality pseudo-labels. The main challenge that accurate pseudo-labels mostly come from easy samples remains, but these easy samples cover all classes, which simplifies the problem.

To leverage the accurate pseudo-labels across all classes, Paper B extends the self-training framework proposed by DACS with a novel data augmentation method. The proposed method ECAP builds a memory bank of pseudo-labeled samples during training that is sorted based on predicted confidence. The DACS data augmentation is then modified by injecting (cut-and-pasting) content from high-confidence samples of the memory bank. Table 4.2 shows that the samples from the memory bank are significantly more accurate than samples drawn at random from the target dataset, confirming that predicted confidence correlates well with the accuracy of the pseudo-label. Additionally, our aggressive data augmentation technique ensures that the samples in the memory bank are not too easy for the model, and prevents overfitting.

Table 4.2: Accuracy (in %) of pseudo-labels for pixels originating from the ECAP memory bank and the sampled target image.

	Road	Sidew.	Build.	...	Train	M.bike	Bike
Memory bank	94.3	67.3	89.9		96.1	80.7	76.3
Target image	91.9	60.5	90.4		55.0	37.5	57.4

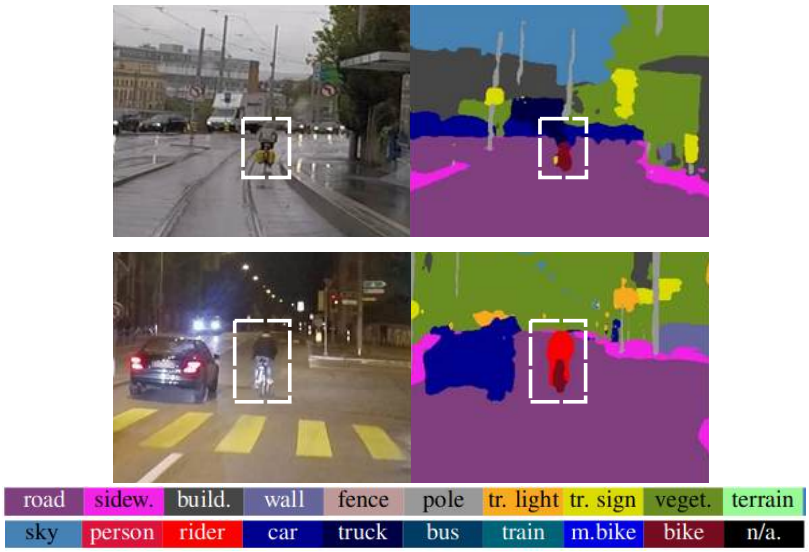

Figure 4.5: Left column: Two images containing a *rider*. Right column: the predicted pseudo-label by ECAP (Paper B). The rider is indicated by a white rectangle in all images. In the top row, the rider has been largely misclassified as a car or truck, while in the bottom row, the rider is correctly classified and relatively accurately segmented.

The proposed data augmentation was integrated with several recent UDA methods and evaluated on the GTA → Cityscapes benchmark. As shown in Table 4.3, ECAP consistently improves performance across methods, with the largest gains observed for the lower-performing ones. These results highlight the importance of managing pseudo-label uncertainty in self-training approaches. Moreover, as discussed in previous section, data augmentation enables transforming easy examples with accurate pseudo-labels into hard, informative training examples, resulting in effective training.

In ECAP, predicted confidence was used as a proxy for certainty of the predictions, which was demonstrated to correlate well with pseudo-label accuracy. However, machine learning models often produce confidence scores that do not match true uncertainty, especially for out-of-distribution samples (i.e. target data in UDA). Because of this, the ECAP memory bank is sometimes polluted with erroneous pseudo-labels, highlighting the need for more accurate

Table 4.3: Performance of different UDA methods without and with ECAP (mIoU in %).

Network	UDA Method	w/o ECAP	w/ ECAP	diff
DeepLabV2	DACS	53.9	58.3	+4.4
DeepLabV2	DAFormer	56.0	61.2	+5.2
DeepLabV2	HRDA	63.0	65.6	+2.6
DeepLabV2	MIC	64.2	66.3	+2.1
DAFormer	DAFormer	68.3	69.1	+0.8
DAFormer	HRDA	73.8	75.0	+1.2
DAFormer	MIC	75.9	76.2	+0.3

and reliable methods for estimating uncertainty.

Another limitation is that cut-and-paste augmentation does not preserve image context, which may prevent the model from learning important contextual cues. For example, the presence of a bike near a person can help classify them as a rider instead of a pedestrian. However, even when the person is riding the bike, the bike may be hidden from the camera view (e.g., behind a fence), so the model must balance using contextual clues with relying on local object appearance. Designing data augmentations that enable learning relevant contextual cues alongside local object appearance is an important challenge for future work.

CHAPTER 5

Multi-view 3D Perception

As the cost of cameras and compute continues to fall, using a rig of multiple cameras for sensing tasks in both robotics and self-driving cars is becoming increasingly common. Multiple cameras with overlapping fields of view increase robustness to occlusion and sensor failure and facilitates inferring 3D structure of the environment. At the same time, multi-view sensing adds complexity because information from overlapping views must be fused. In this chapter we first discuss common applications of multi-view perception and prevalent methods, then investigate the robustness of these methods under domain shifts, and finally present a case study on unsupervised domain adaptation of multi-view pedestrian detection.

5.1 Applications

In the RAIL framework, presented in Chapter 2, multiple infrastructure cameras together with cameras mounted on AMRs may be used for sensing. Typical tasks include estimating the drivable surface and detecting and localizing objects such as pedestrians and forklifts. Unlike the image-space outputs discussed in Chapter 3 (2D detection and semantic segmentation), these tasks

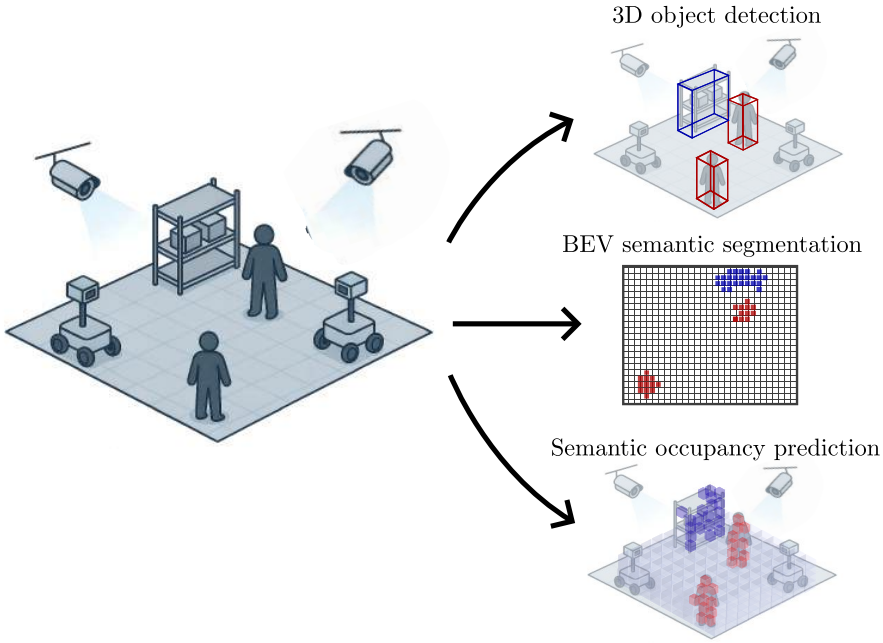

Figure 5.1: Multi-view perception: multiple cameras onboard the robots or mounted to infrastructure are used to extract relevant 3D information, such as 3D object detection, bird's-eye-view (BEV) semantic segmentation, or semantic occupancy prediction.

produce 3D predictions, illustrated in Figure 5.1, which are directly useful for downstream planning and decision making.

Because factory floors are generally near-planar, bird's-eye-view (BEV) representations are particularly attractive. In the industrial implementation described in Paper A, the floor was discretized into a regular grid with 5 cm by 5 cm cells, and the perception system predicted the occupancy of each cell. The multi-class variant of this task, known as BEV semantic segmentation, has been explored extensively in autonomous driving research [83]. In other settings, especially surveillance, multiple static cameras are used for BEV object detection, i.e., estimating object locations as 2D ground-plane coordinates [84].

However, crowded and cluttered factory environments sometimes include

overhangs, cranes, or scaffolding. In such cases a pure BEV representation is insufficient because top-down projections collapse vertical structure: whether a robot can pass under an overhang depends on height. To support such use cases, 3D representations of the environment is necessary. A common approach is 3D object detection [85], where objects are represented by parameterized 3D bounding boxes described by center (x, y, z), dimensions (width, height, length), and heading θ. More recently, semantic occupancy prediction has gained traction wherein multi-view methods predict occupancy and class for small 3D cells called *voxels* to obtain a dense, fine-grained 3D map [86].

As discussed in Section 3.3, camera calibration is fundamental for extracting 3D information and for associating detections across multiple cameras, and is generally a pre-requisite for multi-view BEV and 3D perception. In many applications, the cameras can be jointly calibrated before deployment. In surveillance settings, the cameras can be calibrated in relation to a common world coordinate system, which is typically fixed and aligned with the ground plane. For autonomous vehicles an ego-centric world coordinate system that moves with the car is typically used. In this coordinate frame, the cameras are static, which again allows for establishing the camera calibration prior to deployment.

In other applications the relative camera poses change over time, making one-time calibration infeasible. For example, in the RAIL framework cameras attached to robots will move together with the robots, and the combination of onboard and infrastructure cameras results in a time-varying camera setup that necessitates dynamic recalibration. This makes creating a coherent, joint representation of the environment considerably more challenging.

Furthermore, using cameras both onboard the robots and on infrastructure necessitates wireless communication to consolidate the data. This means that multi-view models must consider bandwidth constraints, additional latency, jitter and intermittent connectivity. In contrast, applications that consider multiple cameras onboard a single robot, or multiple infrastructure cameras with wired links avoid these wireless communication issues.

Although the RAIL framework enables joint use of onboard and infrastructure cameras, this section focuses on the simpler case of time-invariant, pre-calibrated camera rigs with wired connectivity. Under these conditions, the image-to-world correspondence from the calibrated pinhole camera model (Section 3.3) applies directly. In RAIL this setup covers multi-view systems

mounted on a single robot as well as networks of fixed infrastructure cameras. Restricting the discussion in this way makes it possible to analyze multi-view architectures in depth and to establish a clear baseline before addressing more complex scenarios such as time-varying rigs and intermittent wireless connections.

5.2 Architectures for multi-view perception

Methods for 3D perception with multiple cameras can be categorized by how information from different views is fused. As illustrated in Figure 5.2, three common paradigms are early, late, and intermediate fusion. In general, all methods consist of three main components: a feature extractor, a prediction head, and a fusion block. In early fusion, the pipeline begins with fusion, followed by a single feature extraction and prediction stage. In late fusion, each camera view is processed independently through feature extraction and prediction, and the resulting image-space predictions are fused at the end of the pipeline. In intermediate fusion, fusion is performed after per-view feature extraction but before a shared prediction head. The remainder of this section presents these three paradigms in more detail.

Early fusion

Early fusion involves transforming the raw sensor data (images) into common coordinate frame where they can be aggregated and processed jointly. In applications where objects move on a known planar surface, e.g., pedestrians moving on the ground, the images may be warped directly to a bird's-eye-view (BEV) [87]. Warping to bird's-eye-view is done by applying a homography to the image, which is a transformation between two images viewing the same planar surface. In essence, they use (3.6) that maps image pixels to 3D locations based on the planar constraint. The information from the different views are thereby spatially aligned. Thereafter, various fusion techniques may be applied, such as concatenation along the channel dimension, or averaging. The attained multi-view image can then be processed by (more-or-less) standard approaches for semantic segmentation or object detection to perform the relevant task. While this approach is conceptually simple and may yield decent results in certain applications, information loss may occur in the trans-

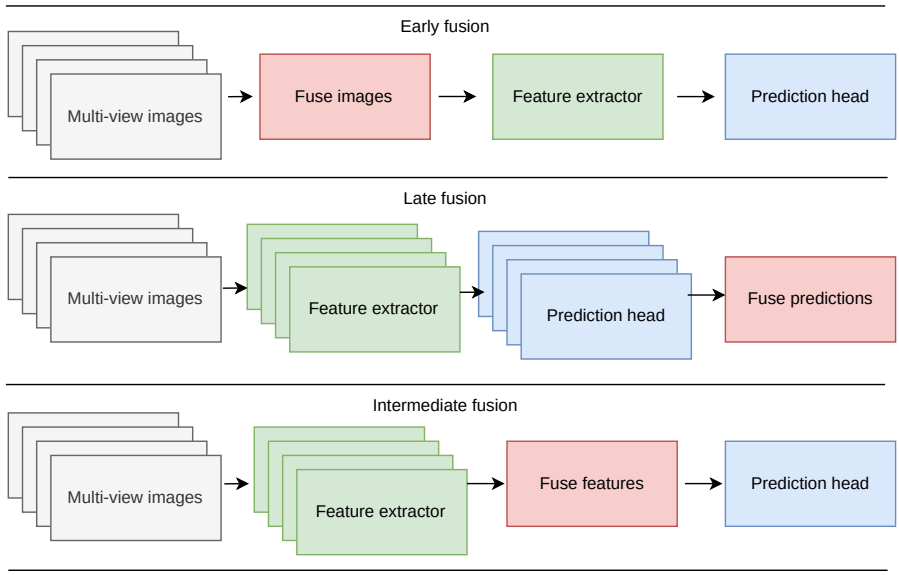

Figure 5.2: Illustration of different paradigms for multi-view perception.

formation since the views are distorted. Moreover, pre-trained backbones for feature extraction and foundation models generally operate on natural images, and may not generalize well to the fused BEV image, which may exclude the use of such models for early fusion.

Another approach, which may be more feasible in non-planar scenes, or unknown planar constraint, is to transform the RGB images into pseudo-LiDAR through dense depth estimates. By predicting the depth of each pixel in the cameras, the pixels can be back-projected to 3D through (3.5) to create a point cloud. This method has been commonly applied to monocular 3D object detection in the autonomous driving community [85]. Naturally, it can be extended to multi-view settings by merging the point clouds created from each view. Recently, DETR3D [88] proposed such a method as a baseline, which was significantly outperformed by their own method based on intermediate fusion. An obvious limitation of the pseudo-LiDAR method is that it depends on monocular accurate depth estimation. Also, it disregards information in the image space that may give additional clues about object location and classes.

Late fusion

In late fusion, a detection or segmentation model is applied the each of the cameras separately. The resulting predictions are then fused in post-processing. This approach is relatively common in the literature of multi-view pedestrian detection, where many methods rely on 2D detections in the images. One class of methods deploy background subtraction algorithms on each view, and then computed likely pedestrian locations based on probabilistic models that consider the foreground in each view [84], [89]. In essence, evidence for the existence of a pedestrian at a given 3D coordinate is collected by projecting the point to each camera view by (3.4), combined with the extracted foreground information and an assumed size of the average pedestrian. To extract more informative information from each view, [90] instead perform instance segmentation. By detecting the feet of the pedestrian, which are assumed to be on the ground, the image-view detections can be transformed to 3D based on a planar constraint through (3.6). Once the detections of each camera have been transformed to 3D, they can be grouped based on e.g., Euclidean proximity.

Similar approaches has been investigated in the setting of autonomous driving. However, in the absence of the planar constraint of objects in the scene, methods here generally rely on monocular 3D detections. For example, DETR3D [88] compare their work to a baseline based on late fusion. Specically, the monocular 3D object detector FOCS3D [91] is applied to each image separately to create 3D bounding box detections. Thereafter, the bounding boxes produced by overlapping cameras are merged and post-processed with non-maximum suppression. GitNet [92] instead consider BEV semantic segmentation. They develop a monocular approach, which extends to multi-view setting by applying their model separately on each view. They then sum the logits (categorical distribution over the considered classes) from multiple views in each location in the BEV plane, before applying a softmax to derive the fused class probabilities.

While the late fusion pipline is conceptually simple, it has inherent limitations as its performance hinges on accurate single view results. Specifically, it suffer from the illposed problem of inferring 3D strucure from a monocular camera. Second, mised detections and false positives may lead to conflicting predictions between views, leading to ambiguities when fusing the predictions from multiple cameras.

Intermediate fusion

Methods based on intermediate fusion are motivated by the limitations of early and late fusion. While early fusion fails to leverage high-performing modern image processing tools, late fusion fails to leverage multiple view and instead is overly dependent on single view results. In contrast, intermediate fusion leverages the strength of modern image encoders to extract rich features from each camera independently. These features, being more abstract than raw pixels but more informative than discrete predictions, offer a flexible and powerful representation space for multi-view integration. By fusing at the feature level, the model can incorporate contextual cues, exploit inter-view redundancy, and learn correspondences across views, which are essential for resolving depth ambiguities and reconstructing 3D spatial relationships.

Multi-view methods based on intermediate fusion may be categorized as either *dense* or *sparse*, depending on the nature of the learned feature representation in 3D.

Dense intermediate fusion

The most common approach for intermediate fusion revolves around building a dense feature space in BEV, although alternative representations such as 3D voxels or multiple orthogonal planes have also been proposed. Such methods generally consist of three steps; (i) processing each view by a conventional feature extraction backbone (CNN or ViT), (ii) transforming image features from all cameras to 3D and fusing them into the chosen dense feature representation, (iii) a task-specific head process the dense features to create the final predictions. In this framework, stage (ii) is particularly difficult due to the challenges involved in inferring 3D information from images. Existing methods can be categorized as either *pushing* or *pulling*.

Pushing methods estimate pixel-wise depth in each camera and use this information to "push" image features into 3D space. In essence, known camera calibration alongside pixel-wise depth estimates allows these methods to transform pixel coordinates to 3D through (3.5), presented in Chapter 3. Although, many methods use a categorical depth distribution rather than a deterministic value, meaning that each pixel is pushed to multiple 3D candidate locations [94]–[98]. Since the depth estimation is performed individually per view, pushing methods face fundamental challenges due to the ill-posed

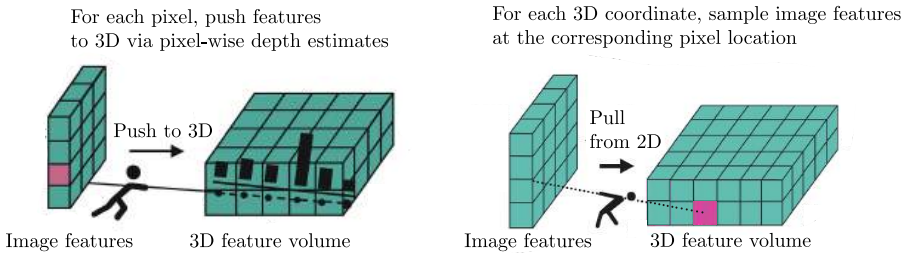

Figure 5.3: Illustration of *pushing* and *pulling* methods for dense intermediate fusion. Figure inspired by [93].

nature of monocular depth estimation. Without stereo or LiDAR cues, the network must infer depth from appearance alone, which introduces significant ambiguity.

Pulling approaches have gained popularity, as they don't depend on monocular depth estimation. Instead, they initialize a 2D [87], [99] or 3D [93], [100] grid in world-coordinates, which is then projected into each view to extract image-view features. This projection, which requires known camera calibration matrices, is described by (3.4). Once the projection of a 3D point in each camera is established, image features can be sampled through bilinear sampling [87], [93], [100]. However, since scene geometry (actual depth) is not considered in the transformation, occlusions in crowded scenes result in 3D points pulling irrelevant information from affected cameras. To increase flexibility, BEVFormer [101] uses deformable attention [40] to predict sampling offsets around such projected image points to learn where to pull features from. Alternatively, BEVSegFormer [102] replace the analytical projection with a learned MLP that maps 3D points to 2D views without explicit camera parameters.

After transforming the image features to 3D based on either pushing or pulling, most methods collapse the vertical dimension to create BEV features. However, methods aimed at more detailed 3D tasks, such as semantic occupancy prediction, may adopt voxel representations [98], [103] or use multiple orthogonal planes [86] from which fine-grained 3D information can be recovered. Regardless of the chosen representation, it is critical the the resulting fused features have a fixed dimension independent of the number of cameras, enabling deployment on camera setups with varying number of cameras. To

this end, most methods apply simple, non-parametric approaches, such as summing [83], [86], [96] or averaging [93], [99] across cameras. While effective, such methods don't enable sophisticated reasoning about which camera provides the most relevant features for specific 3D locations, which could be important in presence of occlusions. To this end, [104] propose a weight prediction branch that learns how to weigh the features from different cameras to emphasize the most informative views.

Finally, the fused 3D-aware features are further processed to derive task-specific predictions. While voxel and multi-plane representations require tailored task-specific heads, this is particularly simple in the BEV representation. Since the BEV representation has similar structure as a 2D feature map extracted from a single RGB image, i.e., constituting a regular 2D grid of feature vectors, standard image processing techniques for object detection and semantic segmentation can be applied. For example, BEVFormer [83] adapt the image detector Deformable DETR [40] for 3D object detection and the image segmentation model Panoptic SegFormer [105] for BEV segmentation.

Sparse intermediate fusion

Another class of intermediate fusion methods, initially designed for 3D object detection, instead rely on a set of sparse 3D object queries. These queries collect and integrate features from all camera views via self- and cross-attention, following the set-based detection paradigm introduced in DETR [38]. The extension to mutli-view 3D object detection was first introduced by DETR3D [88]. Each query predicts an object center point, which is then projected into each view using the known camera calibration matrix. Based on this reference points, the object queries extract image features through bilinear sampling. The mechanism is equivalent to previously presented *pulling* methods, but with the difference that this is done for a sparse set of object queries rather than a dense grid of 3D points. Later, Graph-DETR3D [106] improves this sampling using a graph structure, while Sparse4D [107] adds deformable attention similar to BEVFormer.

PETR [108] generalizes this further by replacing explicit 2D-3D projections and bilinear sampling with global transformer cross-attention between object queries and multi-view image features. Specifically, by encoding 3D positional information directly into image features via calibration-aware embeddings, they can use the standard transformer decoder in DETR [38] and allow object

queries to attend to the multi-view features jointly. Through the interaction, the model can reason about correspondences between different views more explicitly than what is done by most previous methods. However, global cross-attention across multi-view image features is computationally expensive, making it difficult to extend to dense prediction tasks. Still, the subsequent model PETRv2 [109] shows strong performance in BEV segmentation, despite its low spatial resolution caused by sparse queries.

5.3 Generalization

Multi-view 3D perception models built on supervised learning require large, richly annotated datasets for training. Unlike monocular vision tasks such as object detection, multi-view methods demand complex camera rigs with precise temporal and spatial calibration, along with labor-intensive 3D annotation. This makes the process of data collection and labeling significantly more challenging, which in turn increases the importance of generalization. Since intermediate fusion methods currently deliver the highest performance across applications including 3D object detection and BEV semantic segmentation, this section focuses on their generalization capabilities.

Domain shifts in multi-view perception

Domain shifts in multi-view settings can arise from many of the same factors that affect monocular perception, such as lighting variation, changes in object types, scene appearance, and the gap between simulated and real environments. However, multi-view 3D perception also introduces unique challenges due to variability in camera configurations. Both intrinsic and extrinsic camera parameters play a central role in how image features are mapped to world coordinates. For example, the size of an object in the image depends not only on its distance to the camera but also on the camera's intrinsic parameters, and the spatial correspondence between multiple views depends on both intrinsic and extrinsic parameters of the cameras. Variations in these parameters between training and deployment can significantly degrade model performance [110].

In the industrial application described in Chapter 2, it is critical that a perception system trained in one factory can scale to new areas or even to en-

tirely different factories. Because the placement of ceiling-mounted cameras will vary across installations, the model must handle differences in camera configuration. At the same time, environmental conditions such as lighting, object appearance, and scene layout are expected to remain relatively consistent. This makes changes in camera parameters the primary concern for generalization.

Vulnerability of different architectures

For deployment across varying camera setups, it is crucial that the model explicitly incorporates camera intrinsics and extrinsics as inputs. In contrast, methods such as BEVSegFormer [102], which learn image-to-world correspondences from data rather than relying on calibration matrices, are particularly vulnerable to domain shifts when the camera configuration changes. This vulnerability arises because the camera geometry is baked into learned parameters and cannot be easily adjusted to new setups. To address this, most methods use the camera parameters directly when either *pushing* or *pulling* features to 3D, or in creating 3D aware positional embeddings as in PETR [108].

Recent research has investigated the generalization capabilities of intermediate fusion models under various domain shifts. For instance, [111] evaluate both pushing methods [94], [95] and pulling methods [83], along with sparse methods [88], [107], under several types of perturbations. These include environmental changes such as fog, low brightness, and snow, sensor degradations such as motion blur and color quantization, and camera failures such as dropped frames or missing views. They find that all method categories are vulnerable to such shifts, with pushing methods being especially sensitive. The study by [110] has a particular focus on camera extrinsic shifts. It shows that both pushing and pulling methods suffer performance drops when deployed in configurations different from those seen during training.

All these models rely on the quality of image features, which makes them inherently sensitive to changes in image appearance. However, the way in which features are transformed into 3D space may also affect robustness. This is particularly evident for dense methods that includes a dedicated transformation step. Generally, domain shifts can affect these methods in all three stages of the pipeline: (i) image feature extraction, (ii) image-to-world transformation, (iii) prediction head. A detailed analysis of pushing and pulling methods helps

clarify where and how these weaknesses appear.

Pushing methods use pixel-wise depth predictions in combination with camera intrinsics and extrinsics to push image features into 3D space. If the depth estimates are accurate, the resulting BEV features are well aligned with the actual positions of objects in the scene. Theoretically, changing camera positions should not greatly alter the BEV representation if depth is precise. However, inaccurate depth estimates leads to spatial misalignment of BEV features, which may cause issues for the prediction head in step (iii). Unfortunately, depth estimation performance typically degrade under domain shifts, which deteriorates the performance of the entire framework [110]. In effect, pushing methods are sensitive to domain shifts in all three stages listed above.

Pulling methods typically use camera calibration matrices to project 3D reference points into the image plane and sample image features through bilinear interpolation. This makes the image-to-world transformation stage inherently robust to new camera configurations because it does not rely on learned mappings. However, the resulting BEV features are stretched and corrupted by irrelevant features since the transformation doesn't consider the actual scene geometry. These distortions vary with camera placement and introduce distribution shifts in the BEV feature space, causing issues for the prediction head [112]. For example, pulling features into BEV from a camera with a top-down view (e.g., a ceiling camera) results in only to mild stretching. In contrast, features from a side-viewing camera suffer greatly from stretching. As a result, these methods are sensitive to domain shifts in stage (i) and (iii): image feature extraction and BEV prediction head. Advanced variants such as BEVFormer [83] use deformable attention to focus on the most relevant regions in the image, which may help mitigate these geometric distortions. Still, this merely shifts the burden of generalization to the attention mechanism.

5.4 Case study 2 – multi-view pedestrian detection

In Paper C, we consider estimating the 3D location of pedestrians using a set of stationary monocular cameras with overlapping field of view. This problem formulation naturally relates to the industrial use case at Volvo, where stationary cameras mounted inside the factory may be used under the RAIL framework to detect dynamic objects. While dynamic objects of different

classes and shape, such as pedestrians and forklifts, are of interest, Paper C is dedicated only to pedestrians.

Multiple publicly available datasets exist to facilitate research in this area, which are listed in Table 5.1. In these datasets, the stationary cameras are generally mounted at about two meters height and have a side-view of the scene, together monitoring an area of about 400 square meters. The cameras have overlapping field of view, so multiple cameras may be used to detect pedestrians at certain locations. However, since the scenes are crowded (typically between 20 and 40 people) and the cameras are mounted relatively low, occlusions are frequent. Moreover, the ground of the monitored area is relatively flat, implying that the 3D position of pedestrians can be well described by a point on the ground plane. The task is to estimate the location of each pedestrian on this ground plane. A data sample from the Wildtrack [37] dataset is illustrated in Figure 5.4.

Table 5.1: Popular publicly available datasets for multi-view pedestrian detection.

Dataset	Samples	Cameras	Scenes	Description
Wildtrack [37]	400	7	1	Real-world: ETH university.
MultiviewX [87]	400	6	1	Simulation in Unity.
GMVD [99]	5995	4-6	7	Simulation in Unity (more diverse than MultiviewX).

The state-of-the-art methods for this task are based on intermediate fusion. Specifically, the non-parametric *pulling* method based on bilinear sampling is particularly predominant in the field. A recent method is GMVD [99], which initializes a 2D grid aligned with the ground plane that is then projected into each camera view to sample features produced by the image-view feature extractor (ResNet18 [39]). The acquired BEV features from different cameras are fused by averaging over the camera-channels. Three convolutional layers are applied as prediction head to transform the BEV features into the desired occupancy map, which is a heatmap of likely pedestrian locations. Finally, the occupancy map is subject to thresholding and non-maximum suppression (NMS) to derive pedestrian locations. In considering each location on the heatmap as a candidate detection, the NMS is similar as described i Chapter 3. While GMVD is conceptually simple and achieves high performance in the supervised setting, it unfortunately exhibits limited performance across varying setups in practice.

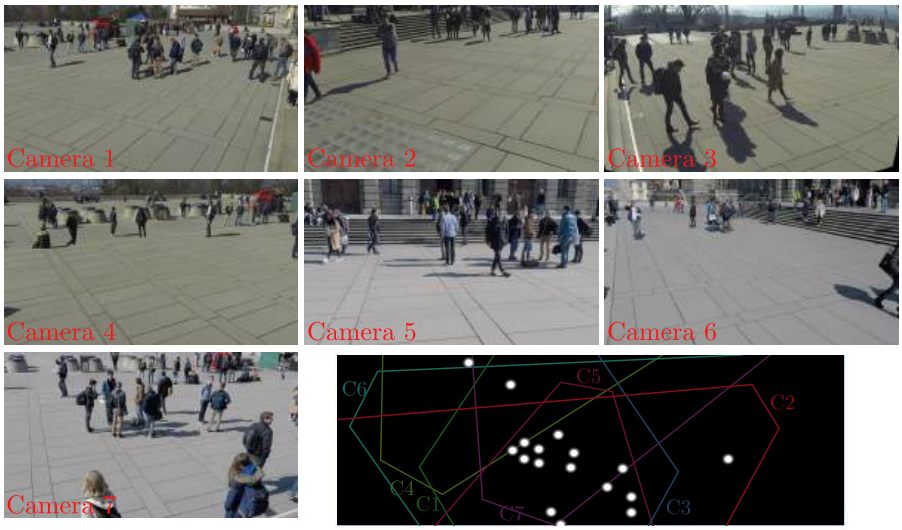

Figure 5.4: A sample from the Wildtrack [37] dataset including images from seven cameras and annotations of pedestrian locations in bird's-eye-view. Each camera's field-of-view is also displayed in the label.

To study this in detail, a few generalization benchmarks based on the mentioned datasets are considered in Paper C. Specifically, Paper C explores sim-to-real adaptation with MultiviewX → Wildtrack, the converse with Wildtrack → MultiviewX, and adaptation across different camera setups with several *real-to-real* and *sim-to-sim* benchmarks. Table 5.2 shows the performance of the *Oracle*, which has been trained on labeled data from the target domain, and the *Source only*, which has been trained on labeled data from the source domain. It also shows the performance of the proposed MVUDA method, which will be described shortly. All three methods are based on the GMVD [99] model and the performance is measured by Multi Object detection Accuracy (MODA), which accounts both for missed detections and false positives. For the full details, the reader is referred to Paper C.

As described in the previous section, generalization problems are expected both in the image-view feature extractor due to appearance changes, and in the BEV prediction head, since the distribution of the BEV features are dependent on the camera extrinsics. Meanwhile, the feature *pulling* is robust

to distribution shifts as it is not trainable. Table 5.2 clearly shows that the method suffers from poor generalizability across different camera setups, both across simulation and real-world environments, but also within simulation and real-world respectively, due to changes in camera configuration.

Table 5.2: Performance in MODA on seven adaptation benchmarks of a *Source only* and *Oracle* model, and the MVUDA adaptation method proposed in Paper C.

Benchmark	Source only	MVUDA	Oracle
MultiviewX → Wildtrack	70.0	85.4	91.3
Wildtrack → MultiviewX	35.7	82.2	90.8
Wildtrack2,4,5,6 → 1,3,5,7	75.0	79.2	83.7
Wildtrack1,3,5,7 → 2,4,5,6	71.6	81.2	86.9
MultiviewX 1,2,6 → 3,4,5	54.0	63.9	74.9
GMVD1 → MultiviewX	70.1	88.9	90.8
GMVD2 → MultiviewX	66.3	88.4	90.8

To mitigate the need for collecting and labeling new data for every new camera installation, Paper C investigates this problem under the UDA paradigm. For example, when studying the benchmark MultiviewX→Wildtrack, labeled data from MultiviewX and unlabeled data from Wildtrack are used for training. To address domain generalization issues in both the image-view feature extractor and the BEV decoder in a unified framework, Paper C proposes a mean teacher self-training framework that allows for training the entire model in an end-to-end fashion on unlabeled target data. The framework, illustrated in Figure 5.5, has many similarities with established self-training methods for monocular UDA presented in Chapter 4. In each iteration of training, the student is trained on a labeled source sample and an pseudo-labeled target sample using an established loss function. Data augmentation is applied to make the task more challenging for the student. Specifically, we apply 3DROM [113] that masks part of the images to artificially introduces occlusions, and Dropout [99] that randomly drops one of the camera views. The pseudo-label is generated by a mean teacher whose weights are an exponential moving average of the student's weights. To generate as accurate pseudo-labels as possible, the mean teacher is fed unaugmented data samples.

As expected from the study conducted in Paper B, erroneous pseudo-labels is an inherent problem with the framework. Compared with semantic seg-

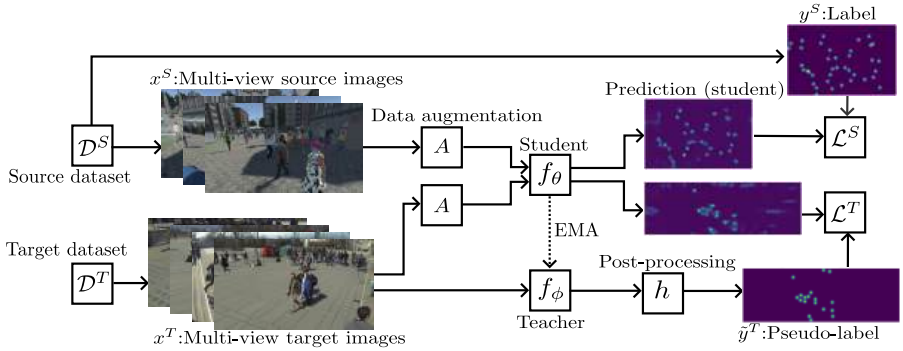

Figure 5.5: Illustration of the self-training framework MVUDA developed in Paper C.

mentation, wherein pseudo-labels are derived from the predictions by simply choosing the most likely class at each pixel location, the object detection problem poses additional challenges due to the reliance on NMS. In applying the NMS to the predicted occupancy map, two different thresholds must be set. One for the minimum score in the occupancy map that can be considered a candidate detection, and one for the distance threshold that determines how close any two detections can be each other. While the second may be regarded as a universal constraint applicable to all domains, the first is based on the model's confidence, which can be greatly affected by domain shifts. Setting the thresholds incorrectly can cause even descent predictions of the model be reduced to completely non-sensical pseudo-labels. Additionally, the thresholds can't be tuned on the target domain due to absence of labels.

To this end, Paper C proposes a simple, yet effective, pseudo-labeling approach to mitigate this issue. The rationale behind the approach is that the spatial variation of scores across the predicted occupancy map is more important than the level of confidence. Specifically, while the overall confidence level may shift significantly between domains, the spatial smoothness of the predictions and the fact that pedestrian locations typically coincide with a local maximum (peak) in the heatmap remains more robust. By altering the NMS method based on this observation, the generated pseudo-labels become more reliable, which greatly enhances the effectiveness of the overall framework.

The results in Table 5.2 show that our UDA framework significantly boosts

the performance compared with *Source only* training. However, the oracle's performance is still unmatched, leaving room for future improvements. Within our proposed framework, more careful treatment of pseudo-label noise, alongside appropriate data augmentation techniques are likely to improve performance further. Another direction is to combine self-training with other methods for UDA, such as domain-invariant feature learning through adversarial training, or broader methods for generalizable machine learning, such as foundation vision models. Moreover, the research field would benefit from investigating how the predominant *pulling* method for feature fusion compares with *pushing* or *sparse* methods, especially under the challenging UDA setting.

CHAPTER 6

Summary of Included Papers

This chapter provides a summary of the included papers.

6.1 Paper A

Erik Brorsson, Kristian Ceder, Ze Zhang, Sabino Francesco Roselli, Endre Erős, Martin Dahl, Beatrice Alenljung, Jessica Lindblom, Thanh Bui, Emmanuel Dean, Lennart Svensson, Kristofer Bengtsson, Per-Lage Götvall, Knut Åkesson
Infrastructure-based Autonomous Mobile Robots for Internal Logistics - Challenges and Future Perspectives
Submitted for possible journal publication.

This paper addresses the development of autonomous mobile robots (AMRs) for automated internal logistics in dynamic indoor environments. For AMRs to operate effectively in such settings, they must be able to perceive their surroundings, localize reliably, and make robust decisions. Most current AMR systems rely on decentralized onboard sensing and decision making, but they face challenges on all fronts. Highly dynamic environments cause frequent

sensor occlusions that hinder perception and localization, while decision making that enables seamless collaboration with human workers remains difficult. To enhance AMR capabilities, this paper investigates the use of infrastructure sensors, wireless communication, and on-premise cloud resources. We propose a reference architecture, RAIL, which enables a wide range of infrastructure- and cloud-based functions, including improved perception through external sensors and computational offloading to the cloud. Through a combination of literature review and evaluation in an industrial deployment, we identify critical open challenges and outline promising directions for future research. These challenges are grouped into four thematic areas: core autonomy and intelligence, infrastructure integration, system-level considerations, and human-centric design. Although the paper covers a broad scope of AMR functionality, one of the central findings concerns perception. We find that combining onboard and infrastructure cameras is a promising approach to achieve robust and comprehensive environment understanding. However, fusing information from diverse sources is challenging, particularly under intermittent connectivity and bandwidth constraints. In addition, training and testing machine learning models require large-scale labeled datasets, which significantly increases costs. Efficient alternatives, such as simulation-based training and testing, represent a promising direction. The conclusions from our work provide a foundation for both academic research and industrial adoption of the next generation of AMR systems.

Contributions: EB decided on the overall layout of the paper, with assistance from KB, KÅ, and LS. EB also contributed to the development and implementation of the robotics system as well as to the writing. KC, ZZ, SFR, EE, BA, JL, TB, KB, and KÅ contributed to the writing, while PG, EE, MD, ED, and KB contributed to the development and implementation of the robotics system.

6.2 Paper B

Erik Brorsson, Knut Åkesson, Lennart Svensson, Kristofer Bengtsson
ECAP: Extensive Cut-and-Paste Augmentation for Unsupervised Domain Adaptive Semantic Segmentation
Published in proceedings of *IEEE International Conference on Image Processing (ICIP)*,
pp. 610–616, Oct. 2024
 DOI: 10.1109/ICIP51287.2024.10647390 .

This paper addresses semantic segmentation in images captured from a car-mounted camera. State-of-the-art approaches based on convolutional neural networks (CNNs) and Vision Transformers (ViTs) achieve strong performance in fully supervised settings. However, since pixel-wise labels are costly to obtain, it is essential that models generalize well. A key challenge is that models are sensitive to domain gaps, which cause significant performance drops when the training data differs from the deployment environment. To mitigate this, we investigate unsupervised domain adaptation (UDA), where a model is trained on a labeled source dataset and adapted to an unlabeled target dataset. This setting is highly relevant in scenarios such as using labeled simulation data alongside unlabeled real-world data. Most UDA methods for semantic segmentation rely on self-training, where pseudo-labels are generated for unlabeled target data and used for supervised adaptation. However, self-training is vulnerable to errors in pseudo-labels, which can degrade performance. In our paper, we design a novel data augmentation method devised to directly address this issue. Specifically, we build a memory bank of confident pseudo-labels which is used through large-scale data augmentation to focus training on such samples. Our method increases performance on two popular sim-to-real benchmarks, thereby reducing the reliance on extensive manual labeling of real-world data. This enhances the practicality of semantic segmentation models for real-world deployment.

Contributions: EB contributed to idea generation, implementation, analysis, and writing, while KÅ, LS, and KB contributed to idea generation, analysis of results, and writing.

6.3 Paper C

Erik Brorsson, Lennart Svensson, Kristofer Bengtsson, Knut Åkesson
MVUDA: Unsupervised Domain Adaptation for Multi-view Pedestrian Detection
Submitted for possible journal publication.

This paper addresses the problem of multi-view pedestrian detection, where multiple stationary cameras capture different perspectives of the same scene to estimate pedestrian positions on the ground plane. The central challenge lies in fusing complementary information from all cameras into a coherent representation of the environment, thereby improving robustness to occlusions and the limitations of single-view detections. State-of-the-art methods employ convolutional neural networks (CNNs) to extract features from each view independently. These features are projected into a bird's-eye-view (BEV) representation, where they are aggregated across views. A prediction head, typically composed of a few convolutional layers, then outputs an occupancy map describing pedestrian locations. Although these methods achieve strong performance in supervised settings, they often overfit to the specific camera configuration from which the training data was collected. As a result, practical deployment is hindered, since collecting and labeling new training data is required for each new setup. To address this limitation, we investigate unsupervised domain adaptation (UDA) for multi-view pedestrian detection. Specifically, we adopt a self-training framework to adapt a detector trained on one camera setup, whether in simulation or in the real world, to a different real-world configuration. We further propose a novel pseudo-labeling strategy that exploits the spatial smoothness of predictions to reduce noise in self-training. This approach substantially improves performance on unseen camera setups without requiring additional labeled data, making multi-view pedestrian detectors more practical and cost-efficient for real-world deployment.

Contributions: EB contributed to idea generation, implementation, analysis, and writing, while KÅ, LS, and KB contributed to idea generation, analysis of results, and writing.

CHAPTER 7

Concluding Remarks and Future Work

This chapter concludes the thesis by addressing the research questions, summarizing both the scientific and industrial contributions, and outlining directions for future work.

7.1 Answering the research questions

This thesis has investigated camera-based perception for Autonomous Mobile Robots (AMRs). Chapter 2 grounded our work in the industrial application and proposed reference architecture for infrastructure-based AMR systems. Chapter 3 presented the computer vision and machine learning fundamentals that camera-based perception systems hinge on, and concluded with describing prevalent challenges. Chapter 4 explored one of these challenges: namely methods that reduce the dependency on labeled data during training, which is an important consideration in real-world applications since labeling data is often expensive and time-consuming. Chapter 5 discussed another fundamental challenge, concerning fusing information from multiple cameras to create a more comprehensive and robust environmental representations. Based on this foundation and the appended papers, the three research questions posed

at the outset are now addressed.

RQ 1 *What are the key challenges in designing reliable camera-based perception systems for AMRs in internal logistics?*

In Paper A, we propose RAIL: *a **R**eference **A**rchitecture for **I**nfrastructure-based AMR Systems in Internal **L**ogistics*, integrating ceiling cameras, on-premise cloud computing, and onboard intelligence. In this architecture, the perception system must estimate the current environment state, such as drivable areas and dynamic object locations, by processing image data from onboard and infrastructure-mounted cameras. This task faces multiple challenges.

Modern computer vision models based on machine learning encounter difficulties throughout the entire model life-cycle. First, extensive data collection is required to cover the operational domain sufficiently. Since most ML models rely on supervised training, this is followed by data annotation, which often involves expensive and time-consuming manual effort. After data collection and annotation, developing high-performing and generalizable models remains an ongoing challenge. In the RAIL framework, multiple cameras on robots or infrastructure may have overlapping fields of view, which raises the additional question of how information can be fused across cameras for more robust and precise perception. Finally, models must be extensively tested and continuously monitored during deployment to ensure reliable operation. In this context, accurately estimating model uncertainty is critical for understanding when the model's predictions can be trusted, which directly affects the safety and reliability of the entire system.

This thesis places particular focus on two of these challenges. First, RQ2 addresses ML training methods that require fewer annotations, thus reducing the need for costly and time-consuming data collection and labeling. This topic is discussed in detail in Chapter 3 and forms the main theme of Papers B and C. Second, how to fuse information from multiple cameras is addressed by RQ3 and Chapter 4, which is central to Paper C.

RQ 2 *How can ML models for camera-based perception be trained efficiently without relying on large-scale labeled real-world datasets?*

Labeling real-world data is prohibitively expensive, which limits the broad use of ML models in computer vision. To alleviate this, simulators offer a promising approach for data collection, labeling, and model testing. However, models trained in simulation often suffer from the sim-to-real gap, which hin-

ders generalization to real-world data. This thesis places specific emphasis on unsupervised domain adaptation (UDA), which enables training models using labeled simulated data and unlabeled real-world data, thus achieving high real-world performance without requiring real-world labels.

Within UDA, self-training has emerged as a promising approach. Effective adaptation is achieved by training on target data using pseudo-labels generated by the model itself. A core challenge is that these pseudo-labels are noisy and often incorrect, limiting adaptation efficiency. Moreover, this label noise is typically structured rather than random, meaning certain classes are regularly confused with others under specific conditions. This structured noise can be reinforced by self-training, worsening the domain gap. In Paper B, we propose a data augmentation method that grounds self-training in confident pseudo-labels to reduce the negative effect of incorrect labels. Paper C further studies pseudo-labeling for multi-view object detection and proposes a robust pseudo-labeling method relying on spatial smoothness in predictions. While these methods mitigate some challenges, detecting erroneous pseudo-labels and reducing their impact remains a key open problem within the self-training paradigm.

RQ 3 *How can ML methods be designed to leverage multiple cameras to increase performance and robustness in AMR perception?*

In Chapter 5, we investigate how multiple cameras can be used for perception. We limit the discussion to camera rigs that are reasonably well calibrated and temporally synchronized. In the RAIL framework, such methods could be applied to process the images from either multiple ceiling cameras on the on-premise cloud, or to process images from multiple cameras onboard the robot.

Recently, deep learning methods that learn how to fuse information from multiple cameras have been successful in tasks such as bird's-eye-view (BEV) segmentation and 3D object detection. These methods fuse image-view features from all cameras, which are extracted by standard image processing backbones, enabling a comprehensive understanding of the multi-view images. However, similar to monocular perception, they generally depend on supervised training. Furthermore, due to the complexity of sensor setups, data collection is typically constrained to one or a few specific configurations. As a result, models often overfit to the training setup, including the cameras' placement.

Through a case study on multi-view pedestrian detection, we investigate an intermediate fusion method based on feature pulling under domain shifts caused by sim-to-real gaps or changes in camera setup. While the transformation of image features to BEV space naturally adapts to new camera calibrations, domain shifts can still arise during image-view feature extraction and within the BEV prediction head, limiting the generalization of these methods across different camera setups. To address this, Paper C demonstrates that self-training techniques originally developed for monocular tasks can be extended to multi-view settings, enabling models to adapt to new camera rigs without labeled data and thereby improving deployment scalability. A key success factor of the framework lies in generating reliable pseudo-labels through careful post-processing, which leverages the spatial smoothness of the predictions.

However, Paper C considers only stationary, well-calibrated cameras connected via wired communication. Fusing data from both onboard and infrastructure cameras introduces additional challenges, including spatio-temporal alignment as the spatial relationship between cameras changes with robot movement. Moreover, wireless communication introduces constraints such as limited bandwidth, latency, and jitter, which must also be managed.

7.2 Scientific and industrial contribution

In Paper A, we propose a reference architecture for infrastructure-based AMR systems and, through a combination of literature review and industrial evaluation, provide a solid foundation for future research. In addition, Paper A identifies critical challenges spanning the entire AMR system, including but not limited to perception, which serve as important directions for both academic and industrial research. Papers B and C build on this foundation by addressing specific perception challenges within the proposed system. Specifically, they present algorithms that advance the state-of-the-art in semantic segmentation and multi-view pedestrian detection under the unsupervised domain adaptation paradigm. Beyond introducing novel methods, the work also provides insights that can inform future research directions. In particular, we emphasize the importance of managing pseudo-label noise and designing effective data augmentation strategies in self-training, and we propose dedicated methods to address these challenges.

The contributions of this thesis also hold significant industrial relevance. Paper A supports the development of robust and flexible AMR systems that align with industrial needs. The method developed for multi-view pedestrian detection in Paper C facilitates deployment in industrial settings by reducing the demand for labeled data. Similarly, the method for unsupervised domain-adaptive semantic segmentation presented in Paper B reduces annotation requirements for this fundamental task. Since semantic segmentation is a core computer vision problem with applications well beyond AMRs, the proposed method and insights may also benefit a broad range of industrial applications.

Together, these contributions demonstrate both scientific advances in computer vision methods and practical progress toward the deployment of high-performing, cost-efficient AMR systems in real-world industrial environments.

7.3 Future work

Although substantial progress has been made in this thesis, particularly in advancing learning with limited labeled data and developing generalizable multi-view methods for camera-based perception, further research is needed.

For learning with limited labeled data, UDA methods based on self-training still lag behind oracle performance. Future work may focus on more rigorous handling of inaccurate pseudo-labels, for example by integrating formal uncertainty quantification methods. While data augmentation has proven crucial, current techniques risk introducing unrealistic artifacts that hinder the learning of important features such as contextual cues. Advances in synthetic data generation, including recent methods such as Gaussian splatting, offer great potential to improve realism and thereby enhance training effectiveness. Future research should seek to combine such advances in synthetic data generation with UDA techniques to maximize their complementary benefits.

For multi-view 3D perception, progress in both model architecture and training methodology is needed to improve generalization. This thesis highlighted vulnerabilities in current approaches, underscoring the need for architectures that effectively leverage multi-view information while remaining robust to domain shifts. While this thesis demonstrated that self-training methods originally designed for monocular detection can be successfully extended to multi-view settings, future research should investigate strategies

that more explicitly capitalize on cross-view consistency to further improve performance. Additional challenges arise when cameras are distributed across robots and infrastructure and connected via wireless communication. In this setting, further research is required to enable robust information sharing under bandwidth constraints, latency, jitter, sensor faults, and spatio-temporal misalignment of data.

While instantaneous environment perception, which is the focus of this thesis, lays the foundation for AMR perception, future research must also consider processing time-series data (i.e., videos) to track environment states over time. Tracking the positions of dynamic agents such as forklifts and pedestrians provides the basis for predicting their future movements, a critical capability for enabling proactive decision making in AMRs and ensuring smooth interactions. More generally, temporal modeling improves state estimation through filtering and increases robustness to occasional misdetections, for example those caused by occlusions. Future work should also investigate how temporal consistency in videos can serve as an auxiliary supervision signal in self-training frameworks.

Finally, foundation vision models represent a paradigm shift. Instead of training models for specific tasks, large foundation models trained on vast datasets offer strong performance across many applications. Although not yet universal solutions, future research can benefit from leveraging these models, for example by using them to guide pseudo-labeling in self-training.

References

[1] G. Fragapane, R. De Koster, F. Sgarbossa, and J. O. Strandhagen, "Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda," *European Journal of Operational Research*, vol. 294, no. 2, pp. 405–426, 2021.

[2] M. Jogin, Mohana, M. S. Madhulika, G. D. Divya, R. K. Meghana, and S. Apoorva, "Feature extraction using convolution neural networks (CNN) and deep learning," in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, 2018, pp. 2319–2323.

[3] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM computing surveys (CSUR)*, vol. 54, no. 10s, pp. 1–41, 2022.

[4] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 5, pp. 1–46, 2020.

[5] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS quarterly*, pp. 75–105, 2004.

[6] H. Taheri and Z. C. Xia, "SLAM; definition and evolution," *Engineering Applications of Artificial Intelligence*, vol. 97, p. 104 032, 2021.

[7] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and R. Medina-Carnicer, "Generation of fiducial marker dictionaries using mixed

integer linear programming," *Pattern recognition*, vol. 51, pp. 481–491, 2016.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, 886–893 vol. 1.

[10] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition," *IEEE Transactions on Image Processing*, vol. 11, no. 4, pp. 467–476, 2002.

[11] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[12] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "PCANet: A simple deep learning baseline for image classification?" *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5017–5032, 2015.

[13] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, "Deep learning vs. traditional computer vision," in *Science and information conference*, Springer, 2019, pp. 128–144.

[14] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, IEEE, 2009, pp. 248–255.

[17] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2015. arXiv: 1409.1556 [cs.CV].

[18] F. Hajamohideen, N. Shaffi, M. Mahmud, K. Subramanian, A. Al Sariri, V. Vimbi, A. Abdesselam, and A. D. N. Initiative, "Four-way classification of Alzheimer's disease using deep Siamese convolutional neural network with triplet-loss function," *Brain Informatics*, vol. 10, no. 1, p. 5, 2023.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 5998–6008.

[20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, *An image is worth 16x16 words: Transformers for image recognition at scale*, 2021. arXiv: 2010.11929 [cs.CV].

[21] H. Thisanke, C. Deshan, K. Chamith, S. Seneviratne, R. Vidanaarachchi, and D. Herath, "Semantic segmentation using vision transformers: A survey," *Engineering Applications of Artificial Intelligence*, vol. 126, p. 106 669, 2023, ISSN: 0952-1976.

[22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin Transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[23] B. Emek Soylu, M. S. Guzel, G. E. Bostanci, F. Ekinci, T. Asuroglu, and K. Acici, "Deep-learning-based approaches for semantic segmentation of natural scene images: A review," *Electronics*, vol. 12, no. 12, p. 2730, 2023.

[24] L. Hoyer, D. Dai, H. Wang, and L. Van Gool, "MIC: Masked image consistency for context-enhanced domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 11 721–11 732.

[25] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3213–3223.

[26] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[27] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[28] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[29] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid Vision Transformer: A versatile backbone for dense prediction without convolutions," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 568–578.

[30] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "PVT v2: Improved baselines with pyramid vision transformer," *Computational visual media*, vol. 8, no. 3, pp. 415–424, 2022.

[31] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, *SegFormer: Simple and efficient design for semantic segmentation with transformers*, 2021. arXiv: `2105.15203 [cs.CV]`.

[32] L. Hoyer, D. Dai, and L. Van Gool, "DAFormer: Improving network architectures and training strategies for domain-adaptive semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 9924–9935.

[33] A. B. Amjoud and M. Amrouch, "Object detection using deep learning, CNNs and vision transformers: A review," *IEEE Access*, vol. 11, pp. 35 479–35 516, 2023.

[34] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[35] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *European conference on computer vision*, Springer, 2016, pp. 21–37.

[36] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[37] T. Chavdarova, P. Baqué, S. Bouquet, A. Maksai, C. Jose, T. Bagautdinov, L. Lettry, P. Fua, L. Van Gool, and F. Fleuret, "Wildtrack: A multi-camera HD dataset for dense unscripted pedestrian detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5030–5039.

[38] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*, Springer, 2020, pp. 213–229.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[40] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, *Deformable DETR: Deformable transformers for end-to-end object detection*, 2021. arXiv: `2010.04159 [cs.CV]`.

[41] P. Sturm, "Pinhole camera model," in *Computer Vision: A Reference Guide*, Springer, 2021, pp. 983–986.

[42] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2002.

[43] Y. Ming, X. Meng, C. Fan, and H. Yu, "Deep learning for monocular depth estimation: A review," *Neurocomputing*, vol. 438, pp. 14–33, 2021.

[44] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods," *Machine learning*, vol. 110, no. 3, pp. 457–506, 2021.

[45] P. Kora, C. P. Ooi, O. Faust, U. Raghavendra, A. Gudigar, W. Y. Chan, K. Meenakshi, K. Swaraja, P. Plawiak, and U. R. Acharya, "Transfer learning techniques for medical image analysis: A review," *Biocybernetics and biomedical engineering*, vol. 42, no. 1, pp. 79–107, 2022.

[46] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[47] J. Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. Yuille, and T. Kong, *iBOT: Image BERT pre-training with online tokenizer*, 2022. arXiv: `2111.07832 [cs.CV]`.

[48] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, PmLR, 2021, pp. 8748–8763.

[49] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 4015–4026.

[50] X. Yang, Z. Song, I. King, and Z. Xu, "A survey on deep semi-supervised learning," *IEEE transactions on knowledge and data engineering*, vol. 35, no. 9, pp. 8934–8954, 2022.

[51] Y. Gal, R. Islam, and Z. Ghahramani, "Deep Bayesian active learning with image data," in *International conference on machine learning*, PMLR, 2017, pp. 1183–1192.

[52] P. Bateni, R. Goyal, V. Masrani, F. Wood, and L. Sigal, "Improved few-shot visual classification," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 14 493–14 502.

[53] J. Li, Z. Yu, Z. Du, L. Zhu, and H. T. Shen, "A comprehensive survey on source-free domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 8, pp. 5743–5762, 2024.

[54] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 4, pp. 4396–4415, 2022.

[55] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *Computer Vision – ECCV 2016*, vol. 9906, 2016, pp. 102–118.

[56] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[57] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[58] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as Neural Radiance Fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[59] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering.," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[60] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "CyCADA: Cycle-consistent adversarial domain adaptation," in *International conference on machine learning*, Pmlr, 2018, pp. 1989–1998.

[61] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker, "Learning to adapt structured output space for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7472–7481.

[62] X. Liu, C. Yoo, F. Xing, H. Oh, G. El Fakhri, J.-W. Kang, J. Woo, *et al.*, "Deep unsupervised domain adaptation: A review of recent advances and perspectives," *APSIPA Transactions on Signal and Information Processing*, vol. 11, no. 1, 2022.

[63] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc., 2017.

[64] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "MixMatch: A holistic approach to semi-supervised learning," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 5049–5059.

[65] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, "FixMatch: Simplifying semi-supervised learning with consistency and confidence," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 596–608.

[66] G. French, M. Mackiewicz, and M. Fisher, *Self-ensembling for visual domain adaptation*, 2018. arXiv: `1706.05208 [cs.CV]`.

[67] W. Tranheden, V. Olsson, J. Pinto, and L. Svensson, "DACS: Domain adaptation via cross-domain mixed sampling," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2021, pp. 1379–1389.

[68] Y. Wang, H. Wang, Y. Shen, J. Fei, W. Li, G. Jin, L. Wu, R. Zhao, and X. Le, "Semi-supervised semantic segmentation using unreliable pseudo-labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 4248–4257.

[69] L. Hoyer, D. Dai, and L. Van Gool, "HRDA: Context-aware high-resolution domain-adaptive semantic segmentation," in *Computer Vision – ECCV 2022*, 2022, pp. 372–391.

[70] Q. Cai, Y. Pan, C.-W. Ngo, X. Tian, L. Duan, and T. Yao, "Exploring object relation in mean teacher for cross-domain detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 457–11 466.

[71] J. Deng, W. Li, Y. Chen, and L. Duan, "Unbiased mean teacher for cross-domain object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4091–4101.

[72] D.-H. Lee *et al.*, "Pseudo-Label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, ICML*, Atlanta, vol. 3, 2013, p. 896.

[73] Q. Zhou, Z. Feng, Q. Gu, J. Pang, G. Cheng, X. Lu, J. Shi, and L. Ma, "Context-aware mixup for domain adaptive semantic segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 2, pp. 804–817, 2023.

[74] N. Araslanov and S. Roth, "Self-supervised augmentation consistency for adapting semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2021, pp. 15 384–15 394.

[75] S. Laine and T. Aila, *Temporal ensembling for semi-supervised learning*, 2017. arXiv: `1610.02242 [cs.NE]`.

[76] S. Thulasidasan, T. Bhattacharya, J. Bilmes, G. Chennupati, and J. Mohd-Yusof, *Combating label noise in deep learning using abstention*, 2019. arXiv: `1905.10964 [stat.ML]`.

[77] Y. Zou, Z. Yu, B. V. Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Sep. 2018, pp. 289–305.

[78] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information fusion*, vol. 76, pp. 243–297, 2021.

[79] K. Mei, C. Zhu, J. Zou, and S. Zhang, "Instance adaptive self-training for unsupervised domain adaptation," in *Computer Vision – ECCV 2020*, Springer, 2020, pp. 415–430.

[80] Q. Lian, F. Lv, L. Duan, and B. Gong, "Constructing self-motivated pyramid curriculums for cross-domain semantic segmentation: A non-adversarial approach," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 6758–6767.

[81] P. Bachman, O. Alsharif, and D. Precup, *Learning with pseudo-ensembles*, 2014. arXiv: `1412.4864 [stat.ML]`.

[82] L. Melas-Kyriazi and A. K. Manrai, "PixMatch: Unsupervised domain adaptation via pixelwise consistency training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 435–12 445.

[83] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai, "BEVFormer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *European conference on computer vision*, Springer, 2022, pp. 1–18.

[84] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua, "Multicamera people tracking with a probabilistic occupancy map," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 267–282, 2007.

[85] X. Ma, W. Ouyang, A. Simonelli, and E. Ricci, "3D object detection from images for autonomous driving: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 5, pp. 3537–3556, 2023.

[86] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Tri-perspective view for vision-based 3D semantic occupancy prediction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 9223–9232.

[87] Y. Hou, L. Zheng, and S. Gould, "Multiview detection with feature perspective transformation," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Cham: Springer International Publishing, 2020, pp. 1–18, ISBN: 978-3-030-58571-6.

[88] Y. Wang, V. C. Guizilini, T. Zhang, Y. Wang, H. Zhao, and J. Solomon, "DETR3D: 3D object detection from multi-view images via 3D-to-2D queries," in *Conference on robot learning*, PMLR, 2022, pp. 180–191.

[89] A. Alahi, L. Jacques, Y. Boursier, and P. Vandergheynst, "Sparsity driven people localization with a heterogeneous network of cameras," *Journal of Mathematical Imaging and Vision*, vol. 41, pp. 39–58, 2011.

[90] R. Qiu, M. Xu, Y. Yan, J. S. Smith, and Y. Ling, "PPM: A boolean optimizer for data association in multi-view pedestrian detection," *Pattern Recognition*, vol. 156, p. 110 807, 2024.

[91] T. Wang, X. Zhu, J. Pang, and D. Lin, "FCOS3D: Fully convolutional one-stage monocular 3d object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 913–922.

[92] S. Gong, X. Ye, X. Tan, J. Wang, E. Ding, Y. Zhou, and X. Bai, "GitNet: Geometric prior-based transformation for birds-eye-view segmentation," in *European conference on computer vision*, Springer, 2022, pp. 396–411.

[93] A. W. Harley, Z. Fang, J. Li, R. Ambrus, and K. Fragkiadaki, "SimpleBEV: What really matters for multi-sensor bev perception?" In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 2759–2765.

[94] Y. Li, Z. Ge, G. Yu, J. Yang, Z. Wang, Y. Shi, J. Sun, and Z. Li, "BEVDepth: Acquisition of reliable depth for multi-view 3D object detection," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, 2023, pp. 1477–1485.

[95] J. Huang, G. Huang, Z. Zhu, Y. Ye, and D. Du, *BEVDet: High-performance multi-camera 3d object detection in bird-eye-view*, 2022. arXiv: `2112.11790 [cs.CV]`.

[96] J. Philion and S. Fidler, "Lift, Splat, Shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3D," in *European conference on computer vision*, Springer, 2020, pp. 194–210.

[97] A. Hu, Z. Murez, N. Mohan, S. Dudas, J. Hawke, V. Badrinarayanan, R. Cipolla, and A. Kendall, "FIERY: Future instance prediction in bird's-eye view from surround monocular cameras," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 273–15 282.

[98] Y. Li, Y. Chen, X. Qi, Z. Li, J. Sun, and J. Jia, "Unifying voxel-based representation with transformer for 3D object detection," *Advances in Neural Information Processing Systems*, vol. 35, pp. 18 442–18 455, 2022.

[99] J. Vora, S. Dutta, K. Jain, S. Karthik, and V. Gandhi, "Bringing generalization to deep multi-view pedestrian detection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 110–119.

[100] S. Aung, H. Park, H. Jung, and J. Cho, "Enhancing multi-view pedestrian detection through generalized 3D feature pulling," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1196–1205.

[101] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Q. Yu, and J. Dai, "BEVFormer: Learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[102] L. Peng, Z. Chen, Z. Fu, P. Liang, and E. Cheng, "BEVSegFormer: Bird's eye view semantic segmentation from arbitrary camera rigs," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5935–5943.

[103] A.-Q. Cao and R. De Charette, "Monoscene: Monocular 3D semantic scene completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3991–4001.

[104] R. Qiu, M. Xu, Y. Ling, J. S. Smith, Y. Yan, and X. Wang, "A deep top-down framework towards generalisable multi-view pedestrian detection," *Neurocomputing*, vol. 607, p. 128 458, 2024.

[105] Z. Li, W. Wang, E. Xie, Z. Yu, A. Anandkumar, J. M. Alvarez, P. Luo, and T. Lu, "Panoptic SegFormer: Delving deeper into panoptic segmentation with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1280–1289.

[106] Z. Chen, Z. Li, S. Zhang, L. Fang, Q. Jiang, and F. Zhao, "Graph-DETR3D: rethinking overlapping regions for multi-view 3D object detection," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 5999–6008.

[107] X. Lin, T. Lin, Z. Pei, L. Huang, and Z. Su, *Sparse4D: Multi-view 3D object detection with sparse spatial-temporal fusion*, 2023. arXiv: 2211.10581 [cs.CV].

[108] Y. Liu, T. Wang, X. Zhang, and J. Sun, "PETR: Position embedding transformation for multi-view 3D object detection," in *European conference on computer vision*, Springer, 2022, pp. 531–548.

[109] Y. Liu, J. Yan, F. Jia, S. Li, A. Gao, T. Wang, and X. Zhang, "PETRv2: A unified framework for 3D perception from multi-camera images," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 3262–3272.

[110] G. Chang, J. Lee, D. Kim, J. Kim, D. Lee, D. Ji, S. Jang, and S. Kim, "Unified domain generalization and adaptation for multi-view 3D object detection," *Advances in Neural Information Processing Systems*, vol. 37, pp. 58 498–58 524, 2024.

[111] S. Xie, L. Kong, W. Zhang, J. Ren, L. Pan, K. Chen, and Z. Liu, *RoboBEV: Towards robust bird's eye view perception under corruptions*, 2023. arXiv: 2304.06719 [cs.CV].

[112] W.-Y. Lee, L. Jovanov, and W. Philips, "Multi-view target transformation for pedestrian detection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 90–99.

[113] R. Qiu, M. Xu, Y. Yan, J. S. Smith, and X. Yang, "3D random occlusion and multi-layer projection for deep multi-camera pedestrian localization," in *European Conference on Computer Vision*, Springer, 2022, pp. 695–710.