ARTICLE INFO
Article ID: 12-09-02-0016

doi:10.4271/12-09-02-0016

Toward Enhanced Goods Tracking in Commercial Vehicle Interiors through AI and Sensor Integrations

Turgay Aslandere,[1] Mathijs Lens,[2] Jörg Christian Kirchhof,[3] Pieter Robberechts,[2] Marcel Grein,[1] Wannes Meert,[2] Patrick Vandewalle,[2] Jesse Davis,[2] Bernhard Rumpe,[3] and Toon Goedemé[2]

[1]Ford Motor Company, Germany
[2]KU Leuven, Belgium
[3]RWTH Aachen University, Germany

Abstract

The efficient tracking and management of goods within light commercial vehicles (LCVs) is crucial for various industries, particularly craftsmen and parcel delivery services. This article explores the integration of artificial intelligence (AI) and sensor technologies to enhance item tracking and optimize logistical operations in LCVs. Two technological approaches are examined: a Bluetooth-based tracking system and a camera-based parcel identification framework. The Bluetooth-based solution is designed primarily for craftsmen. It employs Bluetooth tags, vehicle connectivity gateways (VCGs), and a centralized server to provide real-time inventory monitoring and prevent tool misplacement. The camera-based system is aimed at parcel carriers. It utilizes AI-driven object detection and pose estimation to localize and identify parcels within the vehicle. Experimental evaluations show that Bluetooth tracking ensures reliability in tool management and the AI-based vision system holds promise for future scalability in parcel logistics. The findings underscore the need for adaptive tracking methodologies to improve efficiency, reduce operational costs, and support the digital transformation of commercial vehicle ecosystems.

[ALK+25] T. Aslandere, M. Lens, J. C. Kirchhof, P. Robberechts, M. Grein, W. Meert, P. Vandewalle, J. Davis, B. Rumpe, T. Goedemé: Toward Enhanced Goods Tracking in Commercial Vehicle Interiors through AI and Sensor Integrations. In: SAE International Journal of Connected and Automated Vehicles, Volume 9(2), DOI 10.4271/12-09-02-0016, SAE International, Dec. 2025.

History

Received: 01 Apr 2025
Revised: 07 Jul 2025
Accepted: 04 Dec 2025
e-Available: 23 Dec 2025

Keywords

AI, Sensors, Commercial Vehicles, Vehicle Services, Commercial Vehicle Automation

Citation

Aslandere, T., Lens, M., Kirchhof, J., Robberechts, P. et al., "Toward Enhanced Goods Tracking in Commercial Vehicle Interiors through AI and Sensor Integrations," *SAE Int. J. of CAV* 9(2):2026, doi:10.4271/12-09-02-0016.

ISSN: 2574-0741
e-ISSN: 2574-075X

1. Introduction

Commercial vehicles are defined as vehicles used by a business to transport goods or people on public roads. One prominent subclass of commercial vehicles is light commercial vehicles (LCVs), which are often defined as having a maximum gross weight of 3.5 tons. LCVs play a critical role in a wide array of domains such as craftsman and parcel deliveries. Given their pivotal role, optimizing their use in any way can prove beneficial.

The advent of connected vehicles provides one way to do this, as it has introduced efficiencies in mobility-related processes [1]. Integrating intelligent routing with telematic services facilitates reductions in distance traveled, time spent on the road, and can even improve driver behavior toward more efficient practices [2]. Numerous automotive manufacturers currently provide smart vehicle services [3] and/or telematic services, available through various models, such as subscription-based or complimentary offerings. In addition, external service providers have penetrated the market, combining vehicles with proprietary smart devices to provide telematic services [4].

However, LCVs serve as more than just transportation apparatuses. For craftsmen, the LCV fulfills multiple critical business functions [5] by serving as a storage for power tools and inventory, a mobile workshop, and an office where invoices are prepared and documentation is organized. In addition, the vehicle doubles as a break room where the operators spend their lunch breaks. In parallel, a parcel carrier's LCV primarily serves to convey parcels. For this task, it is crucial to store the parcels in a systematic manner, which requires selecting and arranging parcels at each delivery point for efficient last-mile delivery. Furthermore, vehicle storage capacity must be adaptable during a shift to accommodate delivered parcels and any unforeseen parcel pickups.

Consequently, the development of smart applications and services that improve user efficiency beyond mobility is a crucial step in vehicle digitization. This article examines methods to enhance efficiency through artificial intelligence to help optimize logistical operations in LCVs via in-vehicle item tracking. Concretely, we consider two specific approaches, each targeting a different use. First, we describe a Bluetooth-based strategy for managing tools within a craftsman's LCV. Second, we describe a camera-based system that uses computer vision techniques to identify parcels within a delivery context. We describe each approach in detail and empirically evaluate the potential of the systems.

The body of research within the LCV domain remains relatively sparse. Millo et al. [6] introduced a techno-economic evaluation framework to assess commercial vehicle concepts, focusing on the total cost of ownership and the payback period as key factors for fleet operators. Their model accounts for variations in transport tasks, vehicle size, and powertrain technology, allowing a systematic comparison of these variables. Their framework supports strategic decisions by considering payload capacity, volumetric load, driving range, vehicle cost, and payback period. Their work does not include specific use cases, such as parcel delivery.

Perboli and Rosano [7] examined the role of freight transport and parcel delivery in urban areas, especially last-mile delivery. The study aims to identify key actors, analyze their business models, and explore the integration of traditional and green logistics. They also introduce a simulation optimization tool to evaluate mixed-fleet policies in urban delivery.

Van Duin et al. [8] addressed the rapid growth of e-commerce and the resulting fierce competition among parcel delivery service providers, emphasizing the need for innovation to maintain a competitive edge. They highlight that "last-mile delivery," often conducted with large LCVs delivering single parcels to doorsteps, is the most expensive part of logistics. The literature suggests that parcel lockers offer significant cost-savings potential. The article includes a review of the literature on parcel lockers, describes three analysis methods, and presents the findings of a case study.

Figenbaum [5] examined the use of electronic travel logs to analyze the travel behaviors of Norwegian craftsmen and service companies. These logs were collected from devices installed on the vehicles used by the craftsmen. These people rely on motorized transportation to transport personnel, tools, and materials to work sites. The research of Figenbaum evaluates the travel patterns of craftsmen. They investigated how travel patterns can become more sustainable by transitioning from diesel-powered utility vehicles to battery electric utility vehicles.

Craftsmen and parcel carriers constitute the primary customer segments of LCVs, each possessing distinct requirements for their vehicles based on their occupational procedures and unique characteristics [5]. Despite these differences, both groups of users share a common need for an item-tracking application. An item-tracking application is a system that enables real-time monitoring of tagged items within the loading area of a commercial vehicle.

At the initial level, tagged items can be associated with a specific vehicle, capturing binary loading status, as well as the location and time of loading and unloading. At a more advanced level, items can also be spatially located within the loading area, with actuators indicating the precise position of the required items. For craftsmen, the primary requirement is the tracking of power tools. Business owners benefit from real-time oversight of their tool inventory, facilitating efficient coordination and enhancing the perceived responsibility of employees for the tools they use.

Craftsmen can also utilize real-time tracking to ensure all necessary tools are loaded, preventing inadvertent omissions from storage or tools being left behind at construction sites or stolen. Material tracking, which

includes monitoring fill levels, assists craftsmen in adequately preparing for daily tasks. Tool and material tracking can enable stock optimization, optimizing the quantity of tools and materials, which minimizes the required fixed resources. However, this process is challenging as these items are frequently stored in containers, rendering it impractical to monitor them using cameras.

In contrast, parcel carriers require a parcel tracking system to streamline the loading and unloading processes, which are characterized by stringent temporal demands. During the loading phase, the parcel carriers manually scan the parcels using barcode devices and strategically arrange them within the vehicle. This process is a well-studied process [9], and several systems are proposed in the literature [10, 11]. The spatial configuration of parcels is determined by the delivery sequence and the couriers' individual methodologies, which is seen as a challenge due to the high number of possible configurations and optimization requirements. In the delivery process, the retrieval of specific packages can be hindered by suboptimal vehicle organization, thus increasing operational stress and costs. While barcode and QR code systems are effective in many scenarios, they depend on direct line-of-sight and proper orientation. Our CNN-based vision system addresses these limitations by enabling hands-free identification even when codes are occluded or parcels are arbitrarily oriented.

Consequently, for both use cases, there is a compelling need for innovative methodologies to enhance item tracking, inventory optimization, and parcel organization and retrieval, thereby easing the logistical challenges faced by craftsmen and delivery personnel.

The task of recognizing items and packages within a delivery vehicle presents considerable challenges. A limited number of brands (e.g., Zalando, Amazon) account for a substantial volume of parcels, leading to a prevalence of visually similar items. This is also valid for similar tools, e.g., hammers or screw drivers from the same brand. It is impractical to train the system for every parcel configuration; hence, it must be able to reliably detect and identify new, unfamiliar parcels and tools. Additional challenges arise from the limitations of image sensors in vehicular environments, including restricted resolution, limited field of view, optical distortions, and variable lighting conditions, all of which complicate accurate object detection and localization.

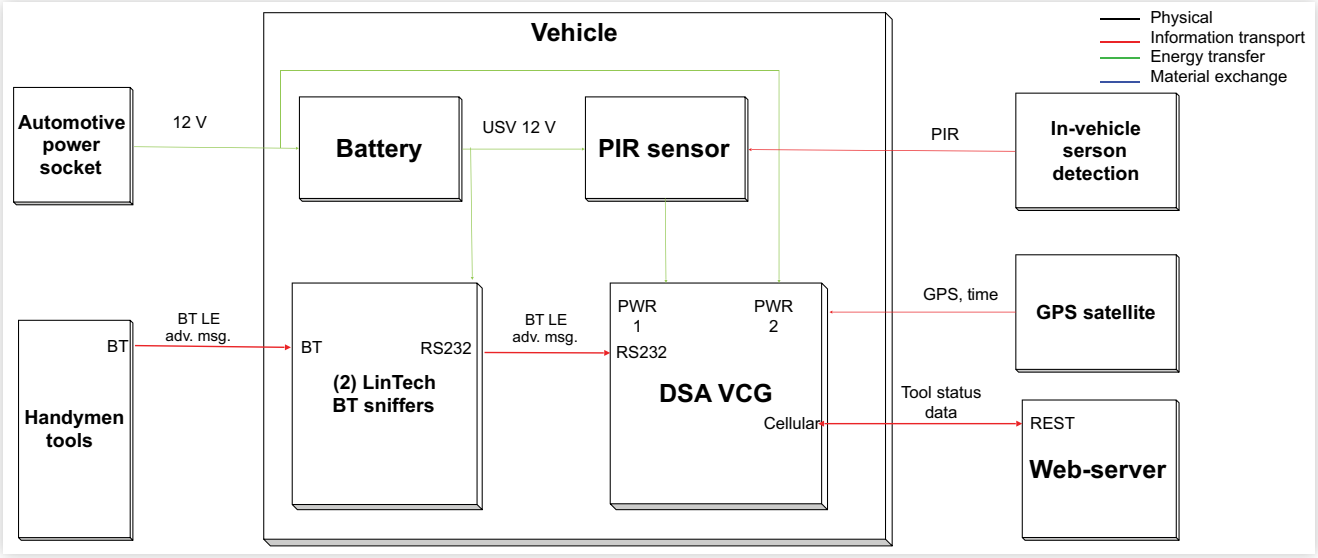
2. Method 1: Bluetooth-Based Setup

2.1. Tool Tracking System Overview

The tool tracking system is composed of three main components (Figure 1). These are Bluetooth tags, DSA VCG, and server. Together, these components create an integrated system for efficiently tracking tools, ensuring their availability, and optimizing their usage. We describe these in the following:

1. **Bluetooth tags**: These are small devices attached to the tools. Their primary function is to report the presence and possibly the location of a tool by periodically emitting radio signals.
2. **DSA VCG**: This is a computing unit installed in a craftsman's vehicle or warehouse. The DSA VCG's role is to receive signals from the Bluetooth tags, process this information, and then forward the processed signals to a centralized server.

FIGURE 1 Overall system architecture including software and hardware components.



Essentially, it acts as a bridge between the physical tools and the digital tracking system.

3. **Server**: The server receives data from one or more DSA VCG units. It performs further processing of this data, stores it in a database, and prepares it for user interaction, often through graphical interfaces. This enables users to track and manage their tools effectively, providing insights into tool location and usage.

The Bluetooth tags in the car are mostly provided by Bosch (Bosch Professional GCC 30-4). Bluetooth tags send messages that a receiving antenna can utilize to detect if a beacon is within reach. The intervals between two messages are alternately 8 and 16 s. We attached two Bluetooth antennas (Lintech BLE Sniffer) to the DSA VCG via the RS232 port. The sniffers provide the DSA VCG with all Bluetooth messages within its detection area. To avoid processing a large number of messages that do not belong to our system, the DSA VCG filters all incoming messages to only process those that use a predefined service universally unique identifier (UUID).

After filtering the messages, the DSA VCG calculates which tools are currently in the vehicle based on the messages it received. The concrete processing steps for this are described later in Section 2.2 of this report. After deciding which tools are in the vehicle, the DSA VCG informs the server about the current state of the trunk, how the trunk content was modified since the last message to the server, and about the status of the vehicle (e.g., speed and position).

The server is based on MontiGem [12] and consists of three components that are distributed using different Docker containers: backend, database, and frontend (Figure 2). The backend handles all message processing and database updates. The database is provided in two different containers. One database is shared between all companies. It includes the information to which certain DSA VCGs belong and which companies exist. Additionally, each company also has its own database in which the information about the company is stored (i.e., tools, vehicles, user accounts, etc.). A more detailed description of the data structure can be found in Section 2.3. The frontend component provides a web app to users. After logging in, users can see an overview of all of their tools and vehicles. Location data and maps are provided externally by Microsoft Azure.

Our in-vehicle system uses motion data from a passive infrared (PIR) sensor, Bluetooth data from beacons, and location data from the global positioning system (GPS). This data is used to detect loading and unloading of the loading area in different scenarios.

2.1.1. Loading Scenario The first scenario is loading a vehicle, Figure 3(a). This can happen, for example, in the morning when craftsmen are loading tools from a locked warehouse into the vehicle, or in the evening when tools are to be transported back to the company site when leaving a construction site.

In this scenario, the system may initially be switched off. As soon as the PIR sensor detects movement, the system is powered up. Once the DSA VCG is booted, the Bluetooth sniffers can detect Bluetooth beacons in their environment. If it is detected that there are Bluetooth beacons in the vehicle that were not present when the system was last switched off, these tools are marked as newly added. Additionally, the system tracks the current location via GPS to inform the server where the tools are loaded.

To save power, the system is switched off as soon as no movement has been detected for a certain (adjustable) period of time and the system has not been supplied

FIGURE 2 Web server system architecture.

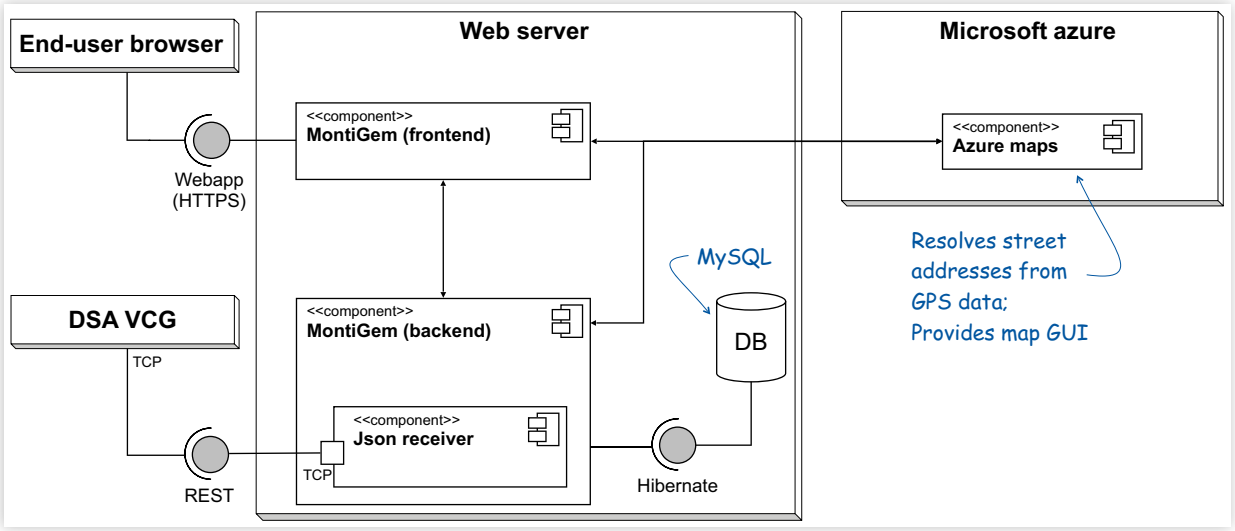
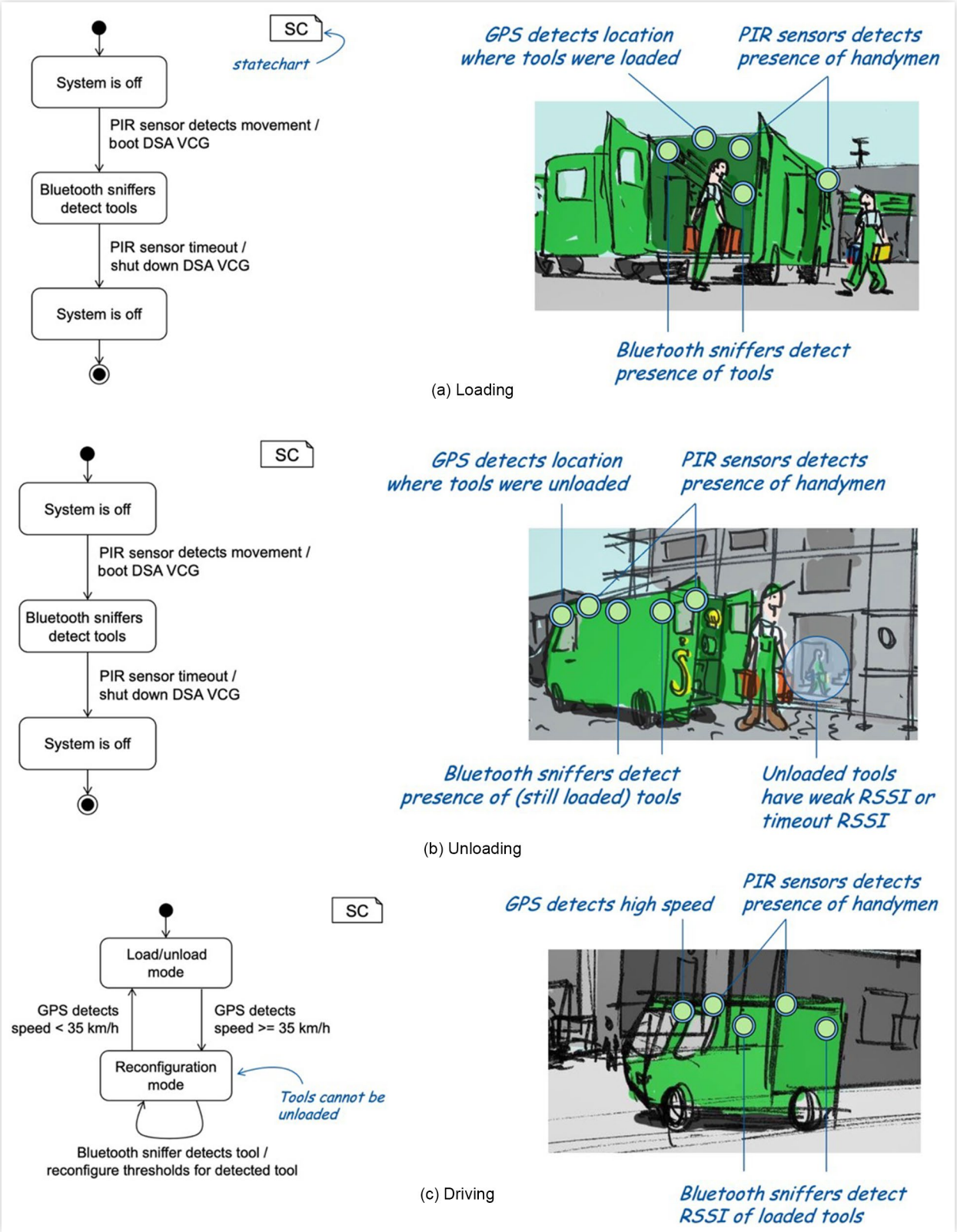


FIGURE 3 Tool tracking scenarios described using state charts.



with external power. In its prototypical implementation, the system is supplied with power via the vehicle's cigarette lighter. In our tests, we found that some vehicles supply the cigarette lighter with power permanently, while other vehicles only supply power when the ignition is switched on. Consequently, it is essential to implement measures that avert the system from depleting the car battery's charge completely, which would consequently inhibit the vehicle's ability to start. In our specific situation, we addressed this issue by incorporating a voltage monitoring device.

2.1.2. Unloading Scenario In the second scenario, the craftsman is at a construction site and takes the tools he needs for his work from the loading area Figure 3(b). In this case, if the system is not already running, it will be restarted by motion detected by the PIR sensor. In a similar way to the first scenario, the system also recognizes that a tool is no longer in the vehicle on the basis of the (un)received Bluetooth data. The details of the processing steps for these operations are described in the next section. In a similar way to the first scenario, the system also recognizes that a tool is no longer in the vehicle on the basis of the (un) received Bluetooth data. The details of the processing steps for these operations are described in the next section. Basically, the detection of a missing tool is based on the fact that the Bluetooth messages of an unloaded tool are either not received at all or only with a very weak received signal strength indicator (RSSI) value.

2.1.3. Driving Scenario The RSSI values used by the system to decide whether a tool is inside or outside the vehicle are not static. The system is programmed to dynamically configure itself throughout the duration of the journey, provided that no tools are being loaded or unloaded while the vehicle is in motion, Figure 3(c). This design allows us to place the antennas at different positions in the vehicle—provided that reception of the Bluetooth beacons is possible. This is particularly important for a prototype where the antennas are not permanently installed in the vehicle. When the vehicle reaches a speed of at least 35 km/h, it is considered to be in a driving state. Within this operational mode, the unloading of tools is prohibited. Future implementations may integrate the system directly into the vehicle, allowing for the modification of this threshold. Alternatively, the speed may be assessed using data intrinsic to the vehicle instead of relying on GPS measurements.

As soon as the system is in reconfiguration mode due to a speed greater than 35 km/h, the RSSI values of the received Bluetooth messages are used to adjust the threshold values above which a tool is considered to be loaded or unloaded. If the speed drops again below 35 km/h, the system is switched back to normal loading and unloading mode.

2.2. DSA Vehicle Connectivity Gateway

This section describes the process operations that we perform within the vehicle using the DSA VCG. The VCG is connected via RS232 with two Lintech Bluetooth sniffers. These receive the Bluetooth messages from the beacons, which are attached to the tools of the craftsmen, and forward the messages to the DSA VCG. Power is supplied to the DSA VCG via an external battery, which is charged by the cigarette lighter. Although the VCG also has an internal battery, its power is not sufficient to supply the connected Bluetooth sniffers. A PIR sensor decides when the DSA VCG shall receive power from the external battery. In addition, the DSA VCG is also connected directly to the power source. So, the DSA VCG always receives power when motion is detected or when power is provided via the cigarette lighter. In addition, the DSA communicates via GPS with GPS satellites and via cellular network with our servers. The GPS location is continuously recorded because it may take some time for the system to detect a missing tool, because beacons may only infrequently send messages. If the craftsman starts driving right after unloading a tool, the system shall not report the current location of the vehicle as the location where the tool was unloaded, but instead the recorded location where the vehicle was when the tool was unloaded.

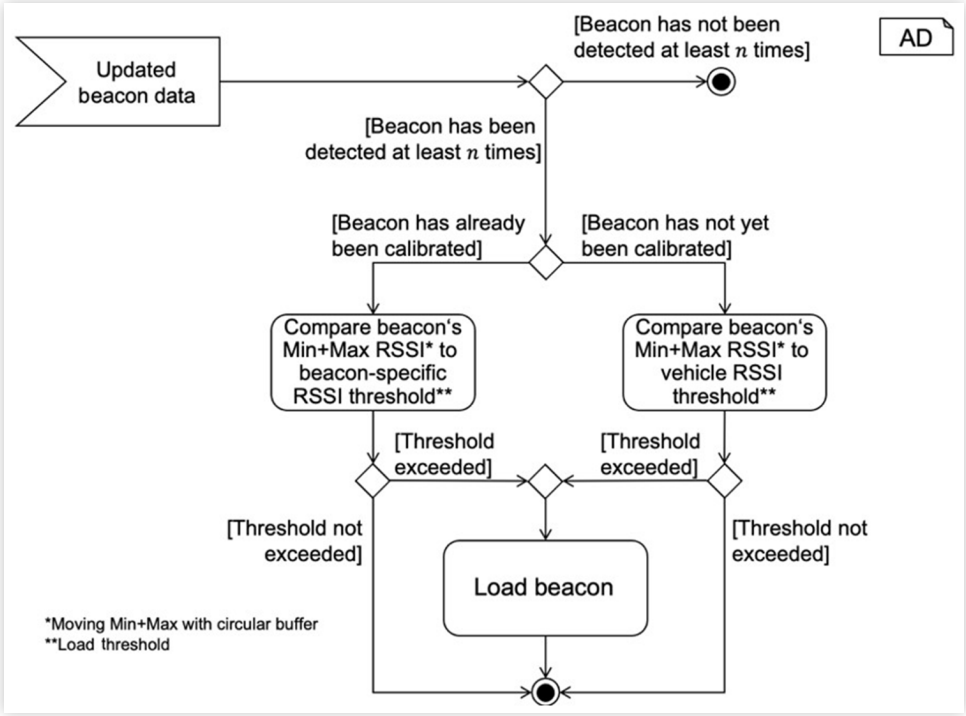
The activity diagram in Figure 4 shows how the system updates its database of beacon data. After collecting the data from our two antennas, we identify which Bluetooth messages refer to (Bosch) beacons by comparing the service UUID of the Bluetooth message to the service UUID of (Bosch) beacons.

Upon identifying messages that function as beacons for processing within our system, we proceed to distinguish between beacons that have undergone prior calibration and those that remain pending calibration (Figure 4).

While the vehicle is driving, the system learns what RSSI values correspond to a tool being in the trunk of the vehicle. These values are later used as thresholds for loading or unloading beacons. In case we do not have that data available, we use general thresholds. These general thresholds are usually very high to prevent incorrectly marking a tool as loaded. Moreover, if two vehicles stand next to each other, the high general threshold prevents both vehicles from considering the same beacon as loaded.

If the currently processed Bluetooth message does not exceed its respective RSSI threshold, we discard the message. If the threshold is exceeded, we use the data to update our database. In general, our processing decides between three states of beacons known to the system: InTrunkBeacons, AllBeacons, and Unloaded. InTrunkBeacons are beacons that are considered within the vehicle. AllBeacons are beacons whose messages are received

FIGURE 4 Activity diagram of beacons describing how beacons change their state.



by the antennas, including those whose RSSI values are too low for them to be considered in the trunk. The tools attached to these beacons are usually close to the vehicle, e.g., in the vehicle parked next to the vehicle. Unloaded beacons are beacons that are not received by the antennas at all and are considered unloaded.

The following describes more precisely how beacons are moved between the different states. To consider a beacon loaded that is currently not loaded, the beacon has to be seen at least n times (n is configurable).

If a beacon has already been detected n times, we compute an aggregated RSSI value from its current and past detections. We store the last m RSSI values for each beacon in a circular buffer and take the sum of the minimum and maximum values in this buffer. This value is then compared against the beacon- and antenna-specific threshold we calculate during the recalibration phase. The thresholds are calculated using the same formula.

If the current value of the aggregated RSSI values exceeds the thresholds, the beacon will be loaded.

Unloading a beacon from the trunk uses the same principle: a beacon is removed when its aggregated RSSI value drops below the threshold. Additional rules include:

1. Tools cannot be unloaded while the vehicle is driving and recalibrating (speed ≥35 km/h).
2. A tool is unloaded if undetected for a set period, which also helps identify tools unloaded while the system was off.

It is also possible to unload beacons based on the frequency with which messages are received. If the connection to a beacon is weak, one might only receive messages from a beacon infrequently. We calculate a running maximum of the times between two receptions of a beacon. This is calculated for every beacon–antenna pair. If no message has been received from a beacon for the running max plus an additional puffer time (16 s in our prototype), we unload the corresponding tool. There are also a minimum and maximum frequency to prevent outliers: In our prototype, beacons are not unloaded based on reception frequency if they have been received within the last 48 s; if a beacon has not been detected within the last 180 s, it will be unloaded regardless of the running maximum. Both of these values (48 and 180) are adjustable.

More information about the beacon handling algorithm can be found in [13].

2.3. Server Architecture

This section elaborates on the backend architecture of our server, which encompasses the internal components and processes that remain inaccessible to end users in their interactions.

This data structure is used as the basis for the MySQL tables created for each craftsman company.

MySQL tables created for each craftsman company. Companies besides their name also have a start and end

of day, and a flag for deactivating unusualTimes detections. If this is set to true, the system will notify the company owners about any unloads that happen outside their business hours, i.e., before startOfDay or after endOfDay. The company has employees which can assume different roles. The roles decide which access rights the employees have within the system. For example, Handymen may not add new vehicles to the system; only administrators have that right.

Containers are our base entity for both storages and vehicles. Exactly one container in the system must be the default container. This container gets assigned all newly added tools before they're first detected by any DSA VCG. In addition to the serial number of the DSA VCG, the containers also have a name and location info. These can be set by the user to identify the container. The location info can be used for storages at job sites that do not have a recognizable address. The optional information that can be added to vehicles is only used as a convenience for the users. It is not actually processed, besides being displayed to the user when viewing information about a specific vehicle.

Each container can have a location. The location contains a GPS longitude/latitude and an address calculated by Azure Maps using the longitude/latitude information. To save costs, the address is only calculated on-demand when a user requests a website where the address for that location needs to be displayed. The meaning of precise/imprecise locations is explained at the end of this section.

The tools encompass, most significantly, a media access control (MAC) address. This address is used to identify the beacon attached to the tool. Users read the MAC address by scanning a 2D code printed on the (Bosch) beacon using a scanner app on their phones and then provide the MAC address to us when adding the tool to the system. The other information is only used to display the tools in the system.

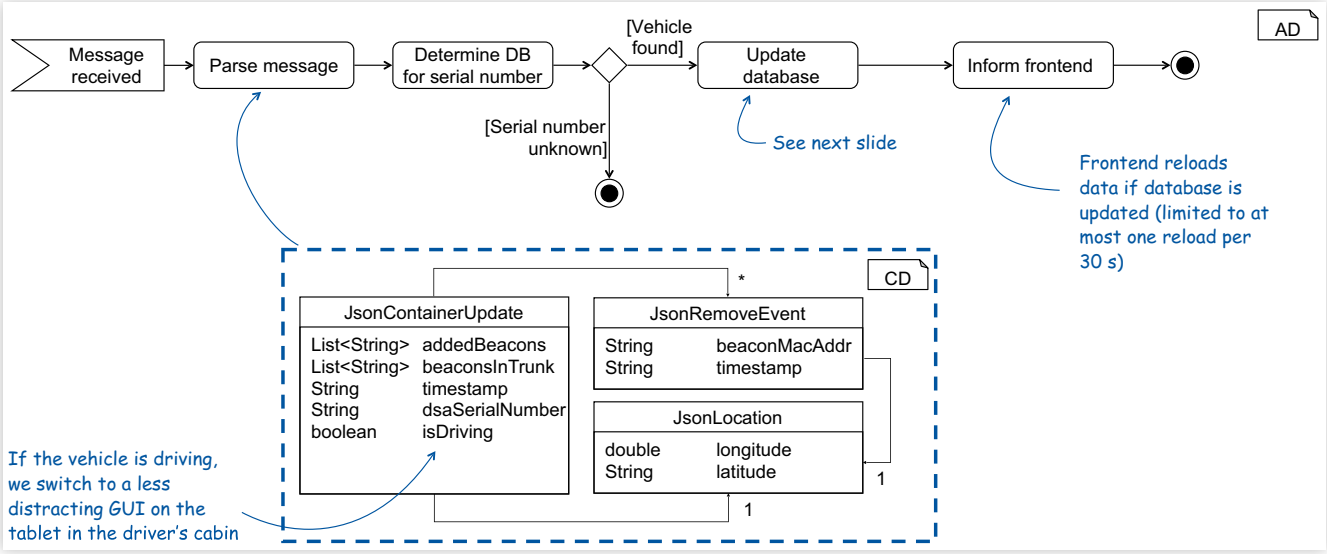
In addition to storing information, the server is also responsible for processing incoming messages, Figure 5, received by the DSA VCGs. Incoming messages are formatted in JavaScript Object Notation (JSON) format. Each message contains a serial number of a DSA VCG that determines to which craftsman company the message applies. Moreover, each message contains the current location and isDriving (i.e., speed ≥35 km/h) of the vehicle and a timestamp when the message was created.

Regarding the content of the trunk, the messages contain a list of the beacons that were added to the trunk since the last update message (addedBeacons), a list of all beacons that are currently considered in the trunk (beaconsInTrunk), and a list of remove events, each telling the backend where a specific beacon was unloaded.

During the "update database" step of our processing pipeline, the backend decides how to assign tools to vehicles based on the updates received from the vehicles. Since the server backend has global knowledge about the system state, i.e., knows the states of all vehicles and tools instead of only the state of one vehicle and its tools, it can prevent unwanted situations. For example, if two vehicles stand next to each other, this can prevent a tool from being continuously moved between the vehicles. Generally, the rules assume that storages and vehicles do not have overlapping antenna perception areas.

- Containers load a newly added tool if
  - The tool is currently unloaded, or
  - The tool is currently loaded by a storage (i.e., not by another vehicle), or

FIGURE 5 Processing pipeline for incoming messages.



  - The loading container is a storage (i.e., not a vehicle), or
  - The loading container is driving.
- Containers load tools that are "still in the trunk" if
  - The tool is currently unloaded, or
  - The container is driving and does not already have this tool loaded.
- Containers remove tools that are reported as removed if
  - The tool is currently loaded by that container, and
  - The container is a vehicle.

The rules that containers may not unload tools loaded by other containers prevent error situations where a tool is moved from one container to another and where the new owner of the tool then reports having loaded the tool before the previous owner has reported having unloaded it.

The server manages system locations by distinguishing between precise and imprecise data. When craftsmen revisit a job site, their vehicles may not be parked in the exact same spot, and GPS accuracy can vary. Thus, imprecise locations are used to record where tools are unloaded. If a tool is unloaded at a site with an existing imprecise location in the database, it is assigned to that location. Conversely, when tracking vehicle positions for online display, precise locations as reported by the vehicle are utilized.

2.4. User Interfaces

The tool tracking system is designed to serve multiple types of users in different situations.

The most powerful user interface is the desktop version [Figure 6(a)] for the company owner or asset manager. This interface provides a continuous, real-time overview of all company tools. As a result, since the ownership of tools is now traceable, field workers handle equipment more carefully, leading to a considerable reduction in tool losses.

Another key user of the system is the driver of an LCV who needs an overview of the tools in their vehicle's trunk. An in-vehicle display [Figure 6(b)] shows both the tools currently loaded and the unloading locations for any tools that have been removed. The primary goal is to prevent the driver from leaving the job site with any tools left behind. While the vehicle is driving, a reduced UI is shown to reduce distractions.

Upon arriving at a construction site, craftsmen utilize a secondary interface via a smartphone [Figure 6(c)]. Occasionally, they need special tools shared among them. The smartphone app grants access to tools within their own LCV and across all company vehicles, facilitating the location and tracking of special tools throughout the fleet.

The customer journey outlines the routines and requirements of craftsmen. Each morning, craftsmen convene to review the day's schedule and collect necessary special tools. Standard power tools, such as cordless drills and cutters, remain in their vehicles overnight. In contrast, rare and costly tools, like specific measurement instruments, are shared among workers and stored centrally at the company's headquarters after each workday. An in-vehicle interface provides updates on the loading status of both standard and special tools, enabling craftsmen to verify their equipment before departing for the job site. Upon arrival, they unload tools, with the system recording the time, location, and status of each tool to prevent any from being left behind.

Craftsmen occasionally require a special tool during a job. Instead of contacting headquarters or colleagues, they can use a smartphone app to locate the tool. This interface provides access to tools not just in their own vehicle but across the entire company fleet, allowing them to swiftly identify and contact the appropriate person for tool retrieval.

Finally, the company owner can review the load status of the entire fleet at any time. The interface provides up-to-date locations of the tools, and different views allow the owner to see lists of tools or a map with the live location. Furthermore, the user can review detailed information about their tools, such as invoices, maintenance documentation, and maintenance intervals.

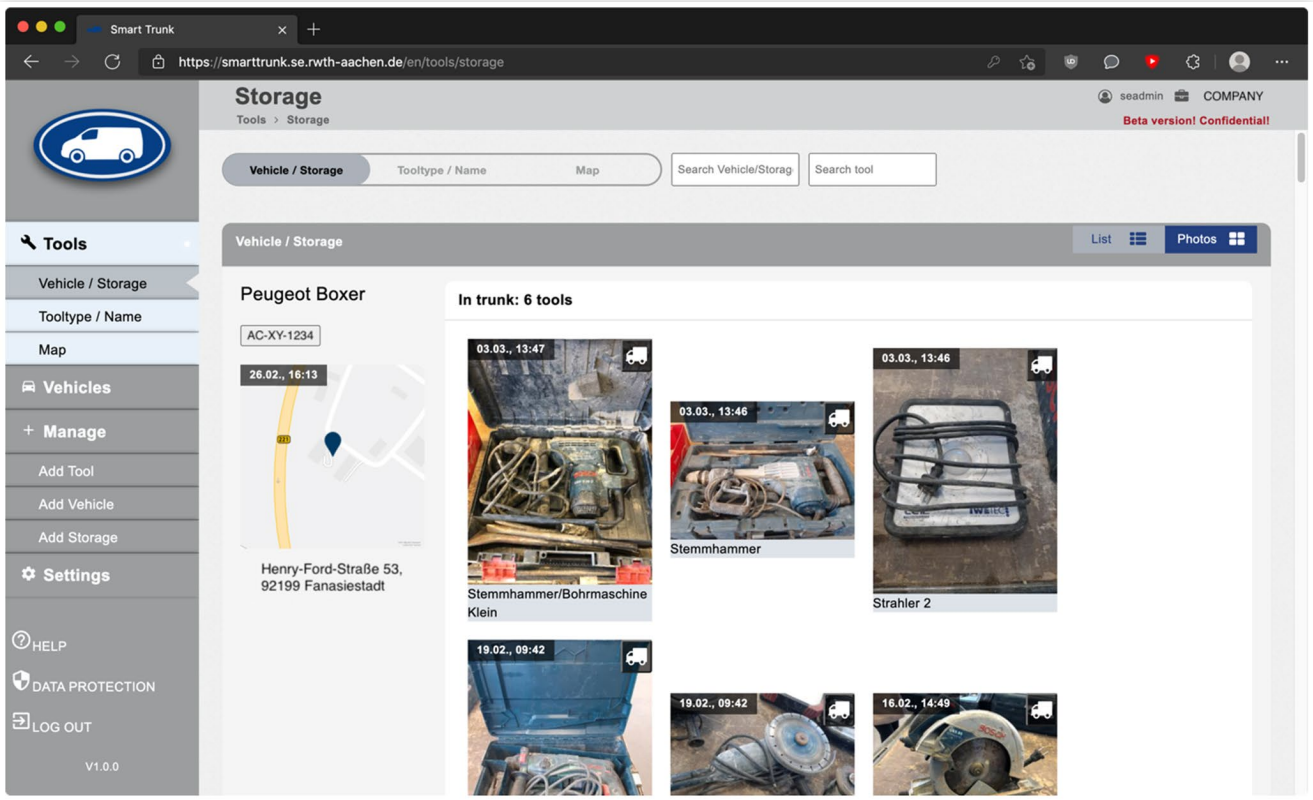
3. Method 2: Camera-Based Setup

We present an AI system for parcel identification using cameras on delivery vehicles. It assists couriers by accurately locating parcels during loading and unloading. Utilizing a YOLO-based architecture [14], it ensures rapid and precise detection, even in complex settings. Object pose estimation helps segment parcels for feature analysis, regardless of the camera angle. Similarity learning distinguishes parcels by unique attributes like size, shape, and color. The system must recognize unknown parcels from limited brand samples, overcoming challenges like sensor limitations and variable lighting, which are addressed by enhancing the vision pipeline with reasoning capabilities.

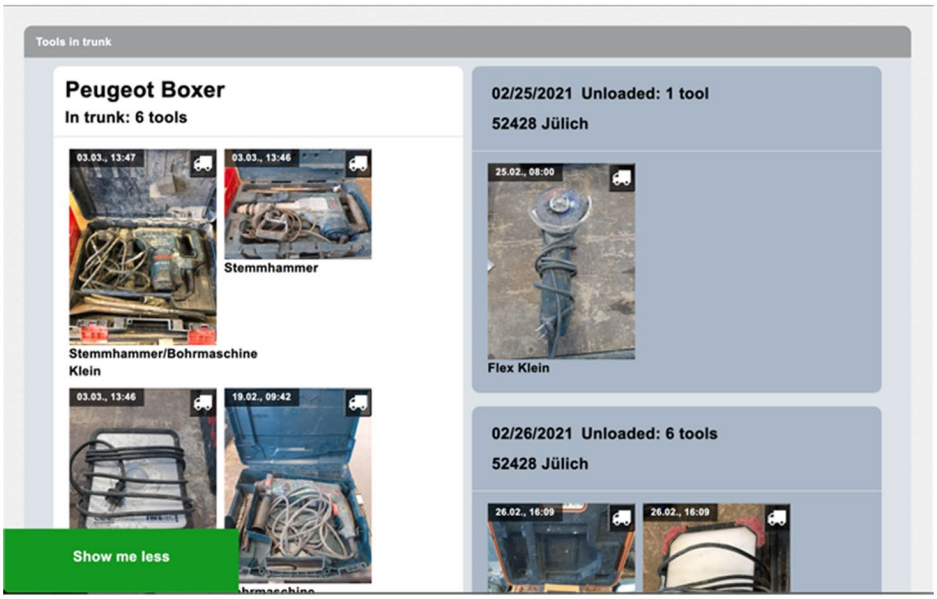
3.1. System Overview

At the sorting center, the parcels are cataloged using an AI-driven system that extracts three key pieces of information: parcel side embeddings, human-interpretable attributes, and parcel dimensions. Each parcel is scanned from multiple angles, and its visual features are encoded into a similarity embedding. In addition, attributes such as brand logos, labels, tape color, and barcodes are

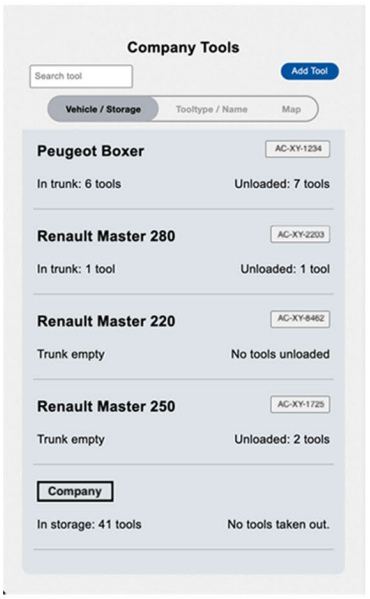
FIGURE 6 Tool tracking user interfaces.



(a) Desktop



(b) In-vehicle display



(c) Smartphone

identified and stored. The parcel's physical dimensions are also recorded to assist in later identification. This catalogue serves as a reference for downstream processing.

Later, when parcels are inside the LCV, the system attempts to re-identify them using onboard image sensors. The detection pipeline consists of two primary branches:

1. Re-Identification Branch: This branch generates a similarity embedding for each detected parcel and compares it against the cataloged embeddings from the sorting center. The goal is to find the closest match based on visual appearance.
2. Attribute Extraction Branch: In parallel, a second branch extracts human-interpretable attributes, such as brand information and unique parcel markings, which further refine the identification process.

In the final stage, a reasoning system combines information from both branches, leveraging similarity scores, extracted attributes, and size constraints to match the observed parcel with its corresponding entry in the catalog. This hybrid approach ensures accurate and robust parcel identification, even under challenging conditions such as occlusions or varying orientations within the van.

3.2. Algorithms

The goal is to build a system that can localize and identify parcels robustly using the image sensors in a LCV. The overall architecture of our proposed framework is shown in the figure below. It consists of five modules. The first module focuses on localizing parcels using a single-shot object detection algorithm, YOLOv5. The second module uses a pose detection algorithm to identify the side faces of each parcel. The remaining modules operate on these side faces. We use a pipeline with two branches. The first branch compresses each parcel's side to a similarity embedding. The second branch complements the similarity embedding with visual human-interpretable attributes (e.g., brand, labels, tape color). Finally, the matching module combines the output of the two branches and attempts to match the observed parcel against an indexed catalogue of available parcel information. In the next section, we discuss the implementation of these components, as well as ideas on how to implement the attribute classification network and the matching module.

3.2.1. Real-Time Tracking To efficiently track parcels in real time, our system leverages the YOLOv5 object detection model. YOLOv5 scans the scene for parcels and identifies their locations within the image space. Once a parcel is detected, a bounding box with a margin is applied to isolate the parcel from its surroundings. This cropped image is then fed into the 3D pose estimation model, which determines the parcel's orientation and extracts relevant side faces for further processing.

Using YOLOv5 for initial parcel localization simplifies the pose estimation task by narrowing down the region of interest, reducing computational complexity, and improving accuracy. YOLOv5 [14] is particularly well-suited for this task due to its ability to perform fast and accurate detections while handling multiple object types, such as parcels, bags, and envelopes, within the same scene.

3.2.2. 3D Pose Estimation and Side Extraction In order to extract the faces of each parcel, we need to estimate the 6 degrees of freedom (DoF) of the object. These 6 DoF can be separated into a translation (lateral movement in x, y, z) and a rotation (pitch, yaw, roll). The capability of most deep learning models to estimate the 6 DoF pose depends on 3D information such as the object's CAD model or 3D sensors (lidar) [15–17]. To reduce the overall cost of the solution, only 2D information (images) can be used as input. Therefore, our objective is to determine the 6DoF pose of parcels using a single RGB image.

We propose a convolutional neural network (CNN)–based approach to achieve this. From the 3D bounding cuboid defined by the 6 DoF, each parcel side can then be extracted for further processing. An additional benefit is that we get the relative dimensions of each parcel. This information helps to limit the search space in the smart matching step. To achieve this, a CNN is used. The CNN identifies the pixel coordinates for each vertex of the parcel(s). Next, CNN predicts the relative size of the parcel(s) based on the pixel coordinates of its corners. Once CNN has found these coordinates, along with the size of the parcel, a 3D pose can be estimated using the perspective-n-point algorithm. The perspective n-point algorithm [18] is a well-known method to estimate the 3D pose of an object from a set of 2D image points. In this context, the algorithm takes as input the dimensions of the parcel, as well as a set of 2D pixel coordinates representing the vertices of the parcel in the image.

This algorithm iteratively maps the 2D pixel coordinates to the correct 3D corner. This mapping involves solving a system of equations that relates the 2D and 3D coordinates, using triangulation. This leads to an estimation for the position and orientation of the parcel in the 3D space. The model should be able to detect multiple parcels and be robust to occlusions, changes in background, and lighting conditions. We use the Center Pose model [19] to find the corners and relative dimensions for each parcel.

3.2.3. Parcel Re-Identification After extracting the sides of a parcel, we propose to train a CNN to transform the image of a parcel's side into a descriptive embedding vector—a fingerprint—that can be used to compute a similarity score for pairs of images. Importantly, the representation used should allow recognition beyond the set of parcels used in the training. An established approach to learn such an effective image embedding consists of training a deep CNN with a Siamese network architecture [20] according to the triplet ranking loss [21] (Figures 7 and 8).

FIGURE 7 Parcel tracking system overview.

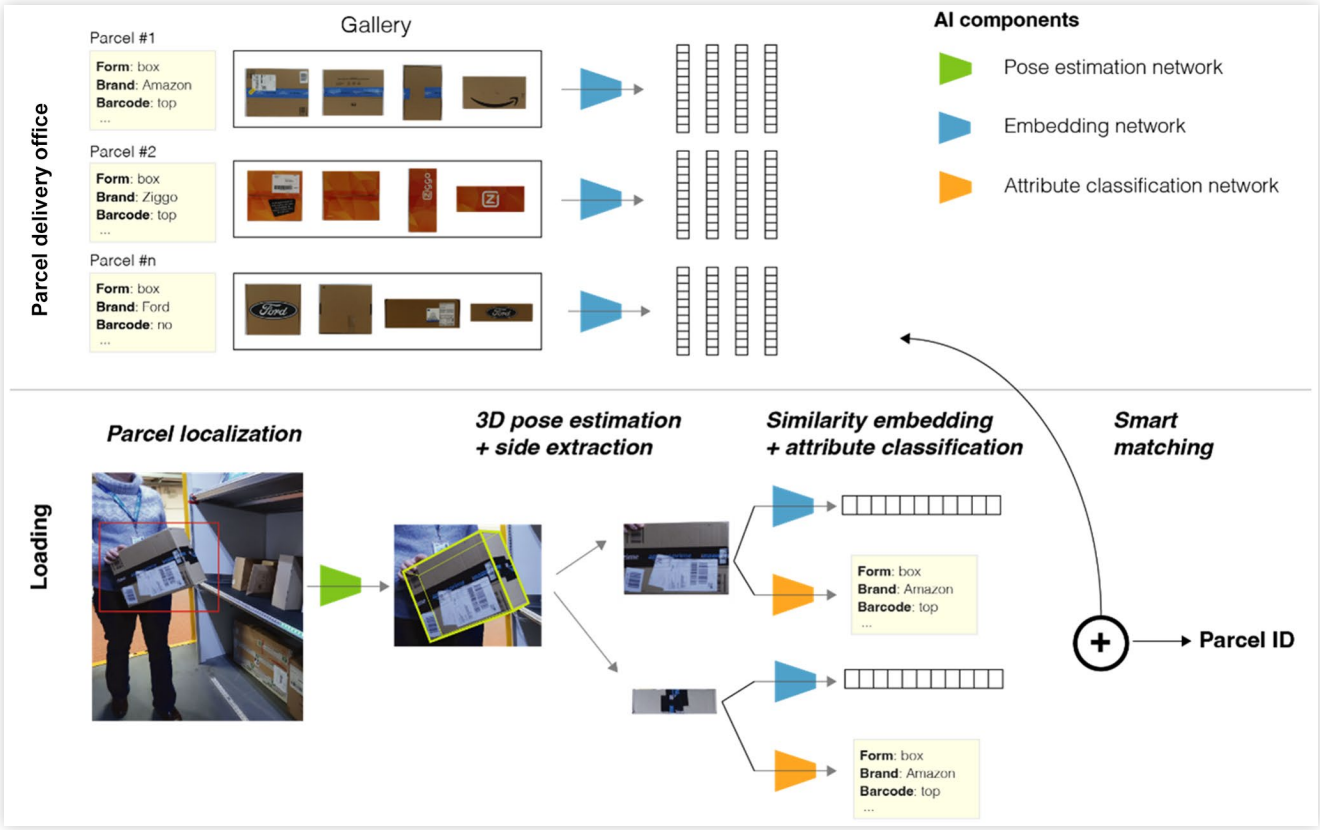
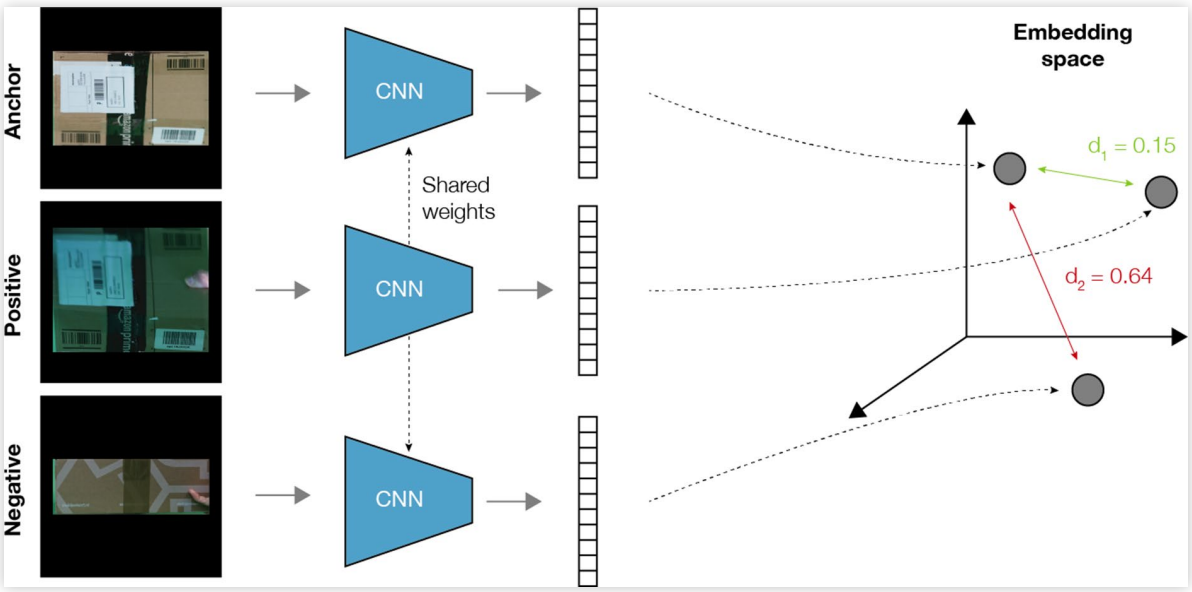


FIGURE 8 Siamese network embeddings.



Classical CNNs modify parameters for image classification, while Siamese neural networks assess feature distances between input images. They employ identical network configurations to produce embedding vectors from the same dataset. The model generates embeddings for novel images and compares them in pairs against a gallery of known images, assigning the highest re-identification probability to the pair with the highest score according to the network.

Preprocessing: We first modify the images so that the CNN becomes rotational and perspective equivariant. As the courier can rotate the parcel in any direction and can move freely within the van, the resulting images of the parcel sides that are extracted from the video stream can have any rotation and perspective. To be able to match multiple images of the same parcel, the model must be insensitive to these transformations. However, the convolution operation is only translation equivariant. We propose to solve this by applying a perspective transform. For computing the homograph matrix, we define the source plane by the four vertices of the detected parcel side, and we define the destination plane as a rectangle with the same aspect ratio as the source plane and a horizontal base. Finally, the resulting rectangular images are rescaled to 300 by 300 pixels, keeping the aspect ratio of the raw image and padding with black pixels (Figure 9).

As a result of this preprocessing step, the only remaining possible variation between a gallery image and a query image is due to 90° rotations, varying light conditions, differences in image quality, and partial occlusions. The first can be addressed by augmenting the gallery with 90° rotations or by normalizing the rotation using PCA. The latter must be "learned" by the Siamese network [20].

Model: The baseline model utilizes the model described by Koch et al. [20], where a Siamese CNN framework with six layers. Rectified linear units (ReLU) are employed in the first layers, followed by sigmoidal units in the final layers. The model comprises convolutional layers with a single channel, filters of variable size, and a stride of 1. Filter counts are multiples of 16, starting at 32 for the first two layers and increasing to 64 in later layers. The output feature maps are activated by a ReLU function, subsequent to a max-pooling operation with dimensions and a stride of 2. A filter map for every layer of the initial twin is denoted as described by Koch et al. [20].

Loss function: The training methodology for the proposed network entails reorganizing the preprocessed sample set into a balanced matrix of both similar and dissimilar image pairs. Each similar pair consists of two unique images of the same parcel, while each dissimilar pair comprises two images from different parcels.

Initially, each training example consists of three unique images: an anchor (i_a), a positive (i_p), and a negative (i_n). The selection ensures that i_a and i_p belong to the same category, while i_n does not. Considering a distance function $d(X, Y)$ in the embedding space, where $X, Y \in \mathcal{D}$, and $E(i)$ denotes the image i's embedding evaluated by the CNN, the loss function to be minimized is given as [20]:

$$L_{\text{enc}} = \max\left(0, d\left(E\left(i_a\right), E\left(i_p\right)\right) - d\left(E\left(i_a\right), E\left(i_n\right)\right) + \alpha\right)$$

where α represents a predetermined margin to be maintained between the two distances.

Ensuring swift convergence necessitates the precise selection of relevant triplets. We adopt a "hard" mining approach for selecting positive examples and utilize a "semi-hard" strategy for identifying negative examples.

Post Processing: The key points and the relative cuboid dimensions are used to find the optimal bounding cuboid. This is achieved using the PnP algorithm. The PnP solver is designed to determine the 3D box based on its eight vertices. It accomplishes this by matching the pixel coordinates of the eight vertices in the image to the real 3D model of the box. If the dimensions of the box are known, a 3D representation can be created that only requires the position and rotation to match the image (6 DoF). The algorithm iteratively searches for the position and rotation by using the 2D points one by one and mapping them to the correct 3D point. When only one point is used, the position is fixed, but the box can still rotate (3 DoF remain). Each additional point decreases the DoF by one, so using a total of four points can solve

FIGURE 9 Perspective transform.

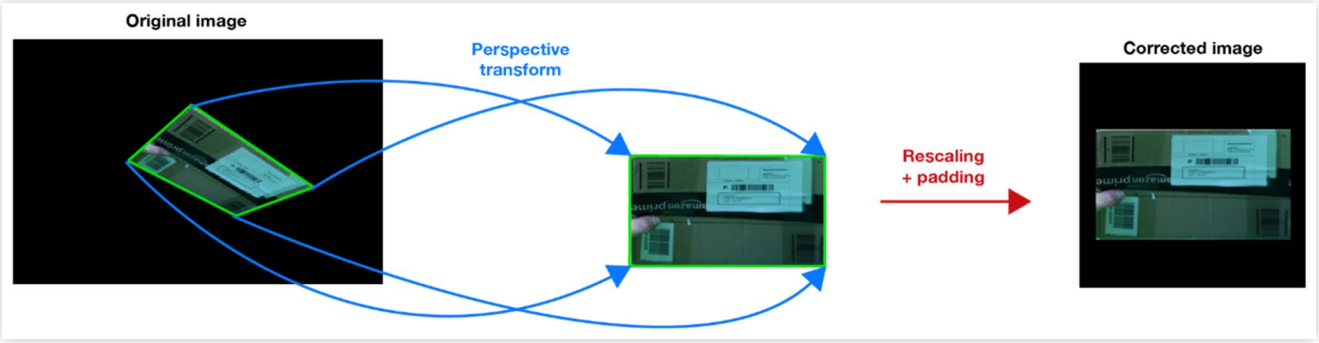
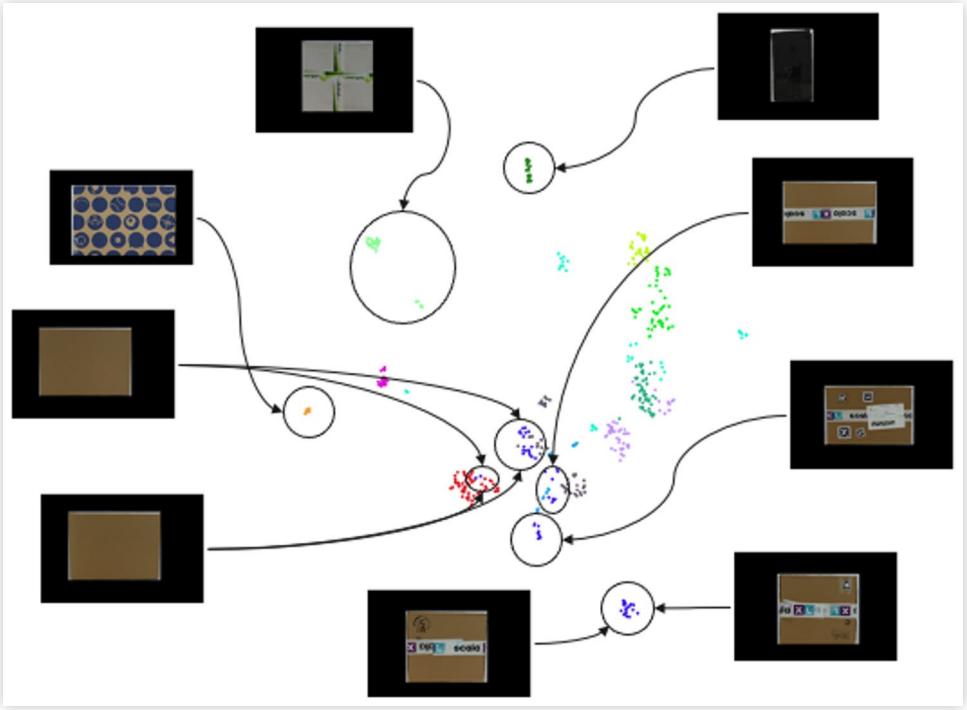


FIGURE 10 Parcel embeddings.



the problem. We use eight points because each point is an estimation with an error margin, and using multiple points helps to find the best combination of all the points and dimensions for accurate results.

Attribute Classification: The output vector generated by the Re-ID network theoretically contains all the visual details of a parcel, making it potentially sufficient to solve the re-identification problem. However, due to the similarity in appearance among parcels, the Re-ID network often struggles to find the correct match. To complement the results of the Re-ID branch, we employ the attribute branch, which utilizes human-interpretable information (attributes) to restrict the search space using logical reasoning. By incorporating this information, we can significantly enhance accuracy. Attributes can include labels, barcodes, text on the parcels, and more. The attribute network takes the same image as input as the Re-ID network. Detecting these attributes is a typical object detection task, which can be accomplished using state-of-the-art neural networks such as YOLO-based models [14], faster R-CNN, and others.

Matching: A naïve approach for solving the parcel matching problem is by selecting a single side of the parcel (e.g., the side with the largest visible face area), generating its embeddings, and matching it with the most similar gallery embedding. This is the traditional approach used in re-identification applications (e.g., person re-identification). However, this naïve deep learning–based approach may fail to re-identify some parcels, as parcels can have a generic look, the gallery could contain many similar parcels, and essential parts could be occluded by other objects. To resolve this, we propose to integrate the deep similarity learning approach with reasoning to (1) correct some of the mistakes made by the deep learning model and (2) perform out-of-distribution detection to avoid incorrect matches. Attribute information helps narrow the search space by either hard filtering out parcels that don't match query attributes or soft scoring them based on attribute similarity. In soft scoring, each attribute is weighted by its reliability, with more accurate ones weighted higher. The final match score combines the attribute score and embedding similarity via a weighted sum (Figure 10).

3.2.4. Model Transparency and Interpretability

At present, the central component of our Re-ID pipeline is a deep convolutional embedding network. Although these networks excel in numerous visual tasks, they are frequently perceived as black boxes, making interpretation challenging. However, providing a clear and comprehensible explanation is crucial for the practical implementation of deep neural networks. This is essential both to establish trust in the model and to assist in model comprehension and debugging. Understanding can be approached by analyzing the network at both the output layer and the intermediate convolutional layers.

A low-dimensional representation of parcel embeddings: At the output layer, we can try to gain insight into the embeddings or feature representations of our image. One way to understand the generated embeddings is to visualize them in a low-dimensional space such as 2D or 3D. This is achievable through methods like PCA (principal

component analysis), which reduce the dimensionality of high-dimensional embedding vectors while largely maintaining their pairwise distances. By visualizing the reduced-dimensional embeddings, we can get an idea of how the network grouping images of the same or similar parcels together. This idea is illustrated in the figure for the embeddings of different sides of 11 parcels. Each dot represents the embedding vector corresponding to a randomly augmented view (i.e., rotated and/or scaled) of the original image. As desired, embeddings of the same (side of) a parcel are typically clustered together in the 2D space.

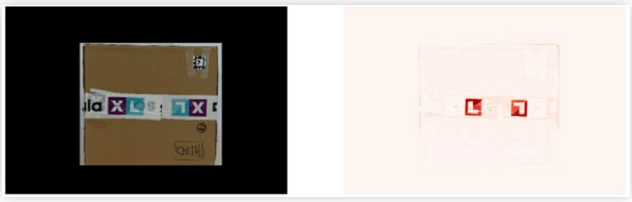
Understanding convolutions through attention: Attention mechanisms allow us to determine which convolutional feature transformations are emphasized, focusing on key regions of the input data for embedding creation. Attention weights facilitate the identification of crucial input areas. Grad-CAM [22] excels in highlighting vital areas of the image, employing backpropagation gradients to elucidate network decisions. Nonetheless, Grad-CAM and related attention methods are mainly tailored for classification tasks, posing challenges when adapted to embedding networks. We address these challenges by proposing an adaptation of the Grad-CAM technique specifically for embedding networks, building on the foundation laid by Chen [23] (Figure 11).

Grad-CAM [23] was initially designed for classification tasks to assess the significance of each neuron by utilizing gradient information that reaches the final convolutional layer(s) of a CNN, aiding in decision-making, such as identifying an "Amazon parcel" image. Neurons in these layers detect class-specific semantic information like an Amazon logo or certain tape on a parcel. For class c, the class discriminative map is derived by first calculating the gradient of the pre-softmax score y^c concerning the feature map $A^k \in \mathbb{R}^{u \times v}$ of a convolutional layer, where k indicates the channel index [23]:

$$g_c\left(A^k\right) = \frac{\partial y^c}{\partial A^k}$$

Subsequently, the gradients are averaged to determine the neuron importance weight α_k^c within each channel [23]:

FIGURE 11 Grad-CAM output.



$$\alpha_k^c = \frac{1}{u \cdot v} \sum_i^u \sum_j^v \frac{\partial y^c}{\partial A_{i,j}^k}$$

where (i, j) represents the spatial index and $u \cdot v$ indicates the spatial resolution of the feature map [23]. This weight is termed a grad-weight. Grad-CAM is subsequently constructed as a weighted sum of feature maps, with a ReLU operator applied [22, 23]:

$$L^c = \text{ReLU}\left(\sum_k^K \alpha_k^c A^k\right)$$

ReLU is utilized since our focus is on features that enhance the target class (i.e., pixels whose intensity should be raised to amplify y^c). Consequently, Grad-CAM generates a class-specific heatmap consistent with the dimensions of the feature map.

Embedding networks cannot be directly adapted for Grad-CAM, as they lack per-class scoring during training or testing. To address this issue, we utilize pairwise distances between samples as a differentiable activation for calculating grad-weights. Our approach involves sampling several dissimilar images from a designated anchor image of a parcel to determine visual attention within the anchor. We formally alter the described per-class gradient following [23] by Chen et al.:

$$g\left(A^k\right) = \frac{\partial L_{\text{pair}}}{\partial A^k}$$

Thus, the grad-weights for an anchor image can be calculated as [23]:

$$\alpha_k = \frac{1}{u \cdot v} \sum_i^u \sum_j^v \frac{\partial L_{\text{pair}}}{\partial A_{(x,y)}^k}$$

4. Results

4.1. Tool Tracking

We evaluated the tool tracking system over a period of 11 months with three trade companies from different trades with different vehicle types. A total of 139 different tools were tracked across 22 containers. During the first trial, our main focus was on improving the reliability of the system. As the system still had an increased number of incorrect measurements at this time, we will not consider the measured data from the first test any further here.

In the other two trials, the status of tools was changed from seen to not seen or vice versa, a total of 10,409 times on the server side. The status of tools was changed from loaded to unloaded or vice versa a total of 7242 times. The location of tools was changed a total of 8519 times, and the assignment of a tool to a container 1513 times. A total of 205,482 location changes were recorded for the containers.

We also conduct an assessment of the system using a range of handyman services. The system demonstrated its capacity to monitor missing tools effectively and did not exhibit any malfunctions throughout the process. Although instances of missing signals and GPS signal anomalies were observed, attributed to external influences, these factors did not impact the system's performance, as the system ensures continuous updates of the signals.

This pilot demonstrated a favorable response from the company owner, craftsmen, and administration. The system was perceived as helpful in providing a better overview of tool locations, saving time in tool search, and reducing physical effort. Concerns were raised by the craftsmen regarding the privacy settings of the system. They expressed fears that the tool tracking system could be misused by the company owner. The craftsmen viewed the tool tracking system as a "wolf in sheep's clothing," believing that the company owner might use it to track the workers rather than just the tools. Since the system requires real-time location data of the vehicles to track the tools, this also reveals the real-time location of the craftsmen. In response, we developed an optional location tracking feature. The company owner can choose between tracking only the loading and unloading locations or opting out of location tracking altogether. This option led to an increased acceptance rate among all users.

The potential cost savings were based on reduced tool search and thus less time for calling colleagues to locate special tools. It was estimated that checking for all tools before leaving the construction site takes between 5 and 10 min per day and vehicle. This sums up to 1.75 and 3.5 h/month and vehicle (assumption: 21 working days per month). Additionally, phone calls conducted by the administration were normally made to find special tools, taking an average of 15 min/day, resulting in a time saving of 1.3 h/month and vehicle (assumption: 21 working days per month and 4 vehicles per company). The tool tracking system was also helpful in reducing tool loss, with two tools saved from being forgotten in just 8 weeks of testing. The range of monetary value of the tools was from 100€ to 4000€. Taken together, there was a time-saving potential between 3.15 and 4.8 h/month and vehicle and two prevented tool losses.

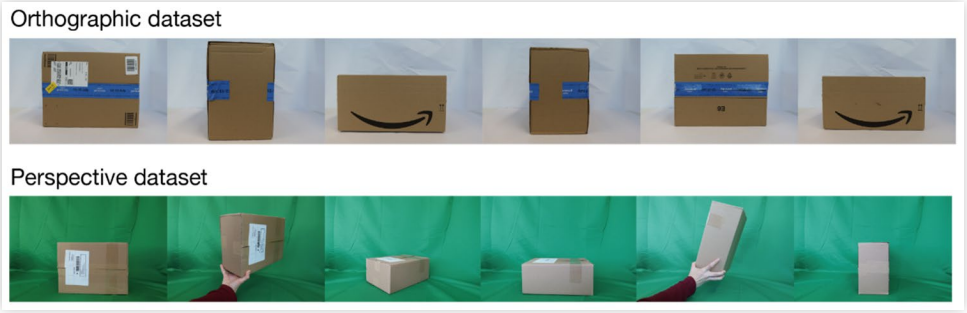
4.2. Parcel Tracking

4.2.1. Dataset We created an image dataset from a diverse set of parcels including the most representative form factors (i.e., boxes, bags, and envelopes) and brands. High-resolution color photos (3648 by 5472 pixels) were taken from multiple sides of each parcel, with the parcel in the center of the image, in order to train and evaluate the machine learning models. The dataset can be broken down into two subsets with different properties. We refer to them as the orthographic and perspective dataset (Figure 12).

The orthographic dataset comprises orthographic views of 2 to 6 sides of 64 different parcels, totaling 408 images, all captured against a white background. To segment the parcel within each image, a YOLO object detector was applied [14], followed by manual post-processing to refine the region of interest.

In contrast, the perspective dataset features perspective views and images of handheld parcels, which often result in partial occlusions, making it more indicative of real-world scenarios. This dataset consists of 725 images representing 59 parcels, with the 3D pose manually annotated for 443 of these images. We trained the pose estimation model on the perspective dataset and a variation of the perspective dataset. We will first discuss the results of the perspective dataset before discussing the results for the variation. All images have a green background to create a simple and controlled environment for early stage model training. However, to better reflect real-world deployment conditions, additional images are needed featuring more complex backgrounds, natural lighting,

FIGURE 12 Orthographic and perspective dataset.



and varied parcel handling. To supplement this, we also considered using synthetic data to increase diversity and robustness, as in Lens et al. [24].

4.2.2. Pose Estimation The network was trained on the perspective dataset, consisting of 243 training images and 43 test images, over 85 epochs. Not all available images were used due to the lack of annotations, as annotating 6D poses is labor-intensive and costly. It is common for object detectors to use intersection over union (IoU) as a metric to evaluate the performance. For the bounding box, this results in the formula:

$$\text{IoU} = \frac{A_{\text{Overlap}}}{A_{\text{Union}}}$$

In IoU 50%, the predicted bounding box or segmentation mask is considered correct if the overlap with the ground truth is at least 50%. Therefore, the higher the IoU 50% value, the better the performance of the algorithm in accurately localizing or segmenting the object of interest. However, centerpose predicts 3D bounding boxes, so we modify the formula and use volume instead of area. This results in:

$$\text{IoU} = \frac{V_{\text{Overlap}}}{V_{\text{Union}}}$$

The average IoU is 59% on the train set and 56% on the test set. The IoU 50% for the train set is 72% and 65% for the test set (Figure 13).

In a later experiment, the green background was replaced with a random background. This adaptation allows the network to handle more complex scenes. However, since the boxes are placed randomly within these images, some results appear visually unnatural.

We retrained centerpose with this random background dataset, initializing it with the weights for cereal boxes from the Objectron dataset [25]. After training for 20 epochs, the network began to overfit, meaning that while it performed well on the training data, its performance on unseen data deteriorated. This suggests that the model became too specialized to the training data and struggled to generalize to new examples.

TABLE 1 Overview of IoU results.

	Avg train IoU (%)	Avg test IoU (%)	IoU50 train (%)	IoU50 test (%)
Perspective dataset	59	56	72	65
Random background	67	50	86	54
Random background (query only)	70	55	88	64

The evaluation results show an average IoU of 67% for training and 50% for testing. The IoU50% scores are 86% and 54%, respectively. Although these percentages may seem low, visual inspection indicates that the results remain acceptable.

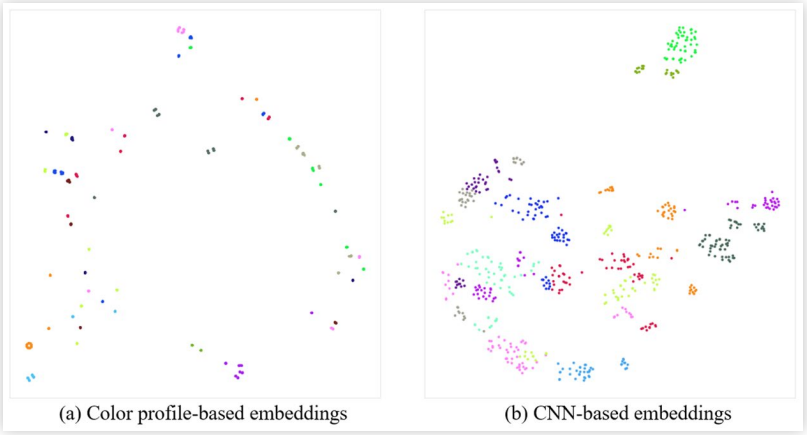
One challenge in the dataset is the high number of parcels aligned parallel to the camera plane, making depth estimation particularly difficult for the network. After removing these images, the evaluation showed a 3% improvement in training performance and a 10% increase in the test set. Table 1 provides an overview of all IoU results. Lens et al. [24] leverage diffusion models to generate realistic, annotated synthetic data. Incorporating this pipeline could enhance performance by pretraining the model on the synthetic dataset before fine-tuning it on real data.

4.2.3. Parcel Re-Identification To evaluate the embedding network, we simulate the parcel re-identification (Re-ID) task. This is a content-based image retrieval (CBIR) problem, where the model must match an image of a parcel (i.e., a query image) with an image of a parcel in the gallery. Therefore, the embeddings are first generated for the query image and each gallery image. Next, the cosine distance between the query and the embedding of each gallery image is computed. The pair with the smallest distance is then regarded as a match. We consider two different evaluation settings:

1. **Parcel Re-ID:** The gallery only contains the top-down view (i.e., the side with the label) while the query can be an image of any side of a parcel in the gallery.
2. **Parcel Side Re-ID:** The gallery contains an orthographic view of each side of the parcels. A query is matched with the correct parcel if the query and the matched gallery image belong to the same parcel (Figure 14).

FIGURE 13 Results pose estimation.

FIGURE 14 Embeddings space of color profile and CNN-based embeddings.



(a) Color profile-based embeddings (b) CNN-based embeddings

To train the embedding network, we randomly split the parcels in the orthographic dataset in a train and test dataset using an 80%–20% ratio. From the training set, we create batches of 100 samples per epoch by randomly sampling images of ten parcels from the training set and applying a random set of augmentations to these images. These augmentations include a horizontal and vertical flip, zooming out up to 50% of the original size, and a rotation around the center.

As our evaluation metric, we use the top-k accuracy classification score on the test set. This metric computes the number of times where the gallery parcel that matches with the query is among the top-k gallery parcels with the smallest embedding distance. Effectively, the top-1 accuracy corresponds to the percentage of parcels that would be identified correctly. Values of $k > 1$ give an indication of how the embeddings would perform when used to create a selection of potential matches which can be refined with additional methods.

As a simple baseline for our deep learning embeddings, we use color histograms. They represent one of the first CBIR techniques. The core idea is to take an image, translate it into a color-based histogram, and use these histograms to retrieve images with similar color profiles. To create the color histogram, we compute the histogram for each color channel with 32 equally spaced bins. By concatenating these histograms, we obtain a vector of length 96. Finally, we apply L2 normalization to this vector such that we compare images by the cosine similarity between their color histograms. Using this approach, we obtain a top-1 accuracy of 38% and top-3 accuracy of 57% on the train set in the parcel Re-ID setting with basic augmentations. The key limitation of color profiles is their sole reliance on color, ignoring textures, edges, and image content.

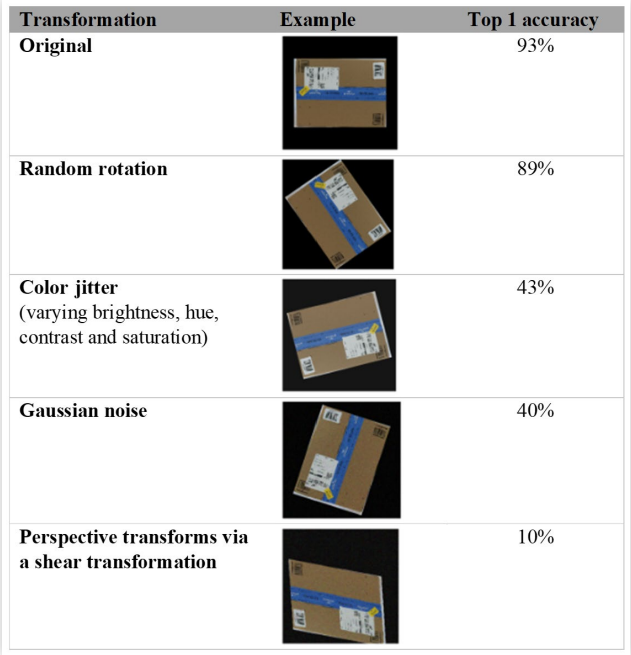
The parcel Re-ID setting is practically the most straightforward setting, as it only requires a top-view picture to be taken in the PDO. However, it can be challenging to visually match two different sides of the same parcel, as they do not necessarily share any visual cues. For example, a logo might be printed on only one side of the parcel. Using basic augmentations in this setting, we obtain a top-1 accuracy of 40% and a top-3 accuracy of 76%. Surprisingly, the top-1 accuracy is only slightly better than the baseline model based on the color profile. Yet, the CNN-based embedding network results in much better embeddings (Figure 15).

The parcel side Re-ID setting is comparatively much easier. With respect to the previous setting, we first only adapt the gallery by adding an orthographic picture of each side of the test parcels. As each query is now a rotated or scaled version of a gallery image, we obtain a high top-1 accuracy of 87% and top-3 accuracy of 93%. However, in a realistic setting, the query images are independently taken from the gallery images in a different environment with varying lighting and video quality and from different angles. To evaluate the effect of these distortions, we applied a set of more advanced augmentations to the query images. As Table 2 illustrates, each of these transformations decreases the performance. The performance of the embedding networks is currently suboptimal. We hypothesize that increasing the amount of training data could significantly enhance their performance.

5. Conclusion

The integration of AI and sensor technologies into LCV presents significant opportunities for optimizing goods tracking, inventory management, and delivery efficiency. This study explored two distinct approaches: a Bluetooth-based tracking system tailored for craftsmen and a camera-based AI solution designed for parcel carriers. The Bluetooth-based system demonstrated high reliability in tool tracking and inventory monitoring, reducing the likelihood of misplaced or stolen equipment. However, its

FIGURE 15 Top-1 accuracy regarding to varying transformations.



applicability remains limited to predefined objects equipped with Bluetooth beacons, making it less suitable for dynamic parcel logistics.

The AI-driven camera-based system showed promise in real-time parcel identification, leveraging object detection, pose estimation, and similarity learning for robust recognition. While this approach enhances the flexibility of tracking diverse package types without requiring prior tagging, challenges such as lighting variations, occlusions, and computational constraints remain. Future advancements in deep learning models and sensor fusion techniques may further improve the accuracy and scalability of vision-based tracking systems.

Overall, this research highlights the necessity of selecting tracking methodologies based on specific use-case requirements. For craftsmen, Bluetooth tracking offers a structured and reliable solution, while AI-based vision systems hold potential for broader logistics applications. Future work should explore hybrid tracking frameworks that combine multiple sensing modalities to maximize accuracy and adaptability in commercial vehicle environments. Building on this, we also see potential in leveraging intelligent loading strategies and support real-time decision-making, such as dynamic routing based on the precise location and delivery order of parcels.

Contact Information

Turgay Aslandere, corresponding author
taslande@ford.com

References

1. Lu, N., Cheng, N., Zhang, N., Shen, X. et al., "Connected Vehicles: Solutions and Challenges," *IEEE Internet of Things Journal* 1, no. 4 (2014): 289-299.
2. Dalla Chiara, G., Krutein, K.F., Ranjbari, A., and Goodchild, A., "Understanding Urban Commercial Vehicle Driver Behaviors and Decision Making," *Transportation Research Record* 2675, no. 9 (2021): 608-619.
3. Tamilvizhi, T., Surendran, R., and Krishnaraj, N., "Cloud Based Smart Vehicle Tracking System," in *2021 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, Southend, UK, 1-6, 2021, IEEE.
4. Cassias, I. and Kun, A.L., *Vehicle Telematics: A Literature Review.* Vol. 54 (Durham, NH: University New Hampshire, 2007).
5. Figenbaum, E., "The Potential for Electric Utility Vehicles in Craftsmen Enterprises," in *European Transport Conference 2016, Association for European Transport (AET)*, Barcelona, Spain, 2016.
6. Millo, F., Cubito, C., Rolando, L., Pautasso, E. et al., "Design and Development of an Hybrid Light Commercial Vehicle," *Energy* 136 (2017): 90-99.
7. Perboli, G. and Rosano, M., "Parcel Delivery in Urban Areas: Opportunities and Threats for the Mix of Traditional and Green Business Models," *Transportation Research Part C: Emerging Technologies* 99 (2019): 19-36.
8. van Duin, J.R., Wiegmans, B.W., van Arem, B., and van Amstel, Y., "From Home Delivery to Parcel Lockers: A Case Study in Amsterdam," *Transportation Research Procedia* 46 (2020): 37-44.
9. Zainudin, J., Samad, H., Miserom, F., and Sabri, S., "Parcel Tracking System Using Barcode Scanner with Verified

TABLE 2 Performance of re-identification systems.

Setting	Test dataset	Top-1 acc (%)	Top-2 acc (%)	Top-3 acc (%)
Parcel re-ID	Orthographic	38	50	57
		40	64	76
Parcel side Re-ID		87	92	93
	Perspective	42	49	57

Notification," *IOP Conference Series: Materials Science and Engineering* 1062 (2021): 012039.

10. Vyshak, T., Varun, S., Unnathi, G., Sujaya, B. et al., "Real-Time Global Parcel Tracking System in the Supply Chain Using RFID and GPS," in *2025 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*, Bangalore, India, 1-5, 2025, IEEE.
11. Clausen, S., Zelenka, C., Schwede, T., and Koch, R., "Parcel Tracking by Detection in Large Camera Networks," in Brox, T., Bruhn, A., and Fritz, M., (eds), *Pattern Recognition. GCPR 2018. Lecture Notes in Computer Science*, 11269, Springer, Cham, doi:https://doi.org/10.1007/978-3-030-12939-2_7.
12. Buschhaus, C., Gerasimov, A., Kirchhof, J.C., Michael, J. et al., "Lessons Learned from Applying Model-Driven Engineering in 5 Domains: The Success Story of the MontiGem Generator Framework," *Science of Computer Programming* 232 (2024): 103033.
13. Kuck, D., Grein, M., Eikelenberg, N.L.W., Pijls, W. et al., Monitoring a vehicle cargo space for objects connected to beacon-transmitting devices. US Patent US12106260B2, Ford Global Technologies, 2024.
14. Jiang, P., Ergu, D., Liu, F., Cai, Y. et al., "A Review of Yolo Algorithm Developments," *Procedia Computer Science* 199 (2022): 1066-1073.
15. Zhao, W., Zhang, S., Guan, Z., Luo, H. et al., "6D Object Pose Estimation via Viewpoint Relation Reasoning," *Neurocomputing* 389 (2020): 9-17.
16. Josifovski, J., Kerzel, M., Pregizer, C., Posniak, L. et al., "Object Detection and Pose Estimation Based on Convolutional Neural Networks Trained with Synthetic Data," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 6269-6276, 2018, IEEE.
17. Su, H., Qi, C.R., Li, Y., and Guibas, J., "Render for CNN: Viewpoint Estimation in Images Using CNNS Trained with Rendered 3D Model Views," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 2686-2694, 2015.
18. Li, S., Xu, C., and Xie, M., "A Robust o(n) Solution to the Perspective-n-Point Problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, no. 7 (2012): 1444-1450.
19. Lin, Y., Tremblay, J., Tyree, S., Vela, P.A. et al., "Single-Stage Keypoint-Based Category-Level Object Pose Estimation from an RGB Image," in *2022 International Conference on Robotics and Automation (ICRA)*, Philadelphia, PA, 1547-1553, IEEE, 2022.
20. Koch, G., Zemel, R., Salakhutdinov, R. et al., "Siamese Neural Networks for One-Shot Image Recognition," in *ICML Deep Learning Workshop*, vol. 2, Lille, France, 1-30, 2015.
21. Hoffer, E. and Ailon, N., "Deep Metric Learning Using Triplet Network," in Feragen, A., Pelillo, M., and Loog, M., (eds), *Similarity-Based Pattern Recognition. SIMBAD 2015. Lecture Notes in Computer Science* (Cham, Switzerland: Springer, 2015), 84-92, doi:https://doi.org/10.1007/978-3-319-24261-3_7.
22. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R. et al., "Grad-Cam: Visual Explanations from Deep Networks via Gradient-Based Localization," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 618-626, 2017.
23. Chen, L., Chen, J., Hajimirsadeghi, H., and Mori, G., "Adapting Grad-Cam for Embedding Networks," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2794-2803, 2020.
24. Lens, M., De Feyter, F., and Goedemé, T., "Making Real Estate Walkthrough Videos Interactive," in *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: VISAPP*, Porto, Portugal, 319-326, 2025.
25. Ahmadyan, A., Zhang, L., Wei, J., Ablavatski, A. et al., "Objectron: A Large Scale Dataset of Object-Centric Videos in the Wild with Pose Annotations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Nashville, TN, 2020.