logistics

MDPI

Article

An Integrated Vision–Mobile Fusion Framework for Real-Time Smart Parking Navigation

Oleksandr Laptiev [1,*], Ananthakrishnan Thuruthel Murali [1], Nathalie Saab [1], Nihad Soltanov [1] and Agnė Paulauskaitė-Taraševičienė [1,2]

[1] Faculty of Informatics, Kaunas University of Technology, 51368 Kaunas, Lithuania; anathu@ktu.lt (A.T.M.); natsaa@ktu.lt (N.S.); nihsol@ktu.lt (N.S.); agne.paulauskaite-taraseviciene@ktu.lt (A.P.-T.)
[2] Artificial Intelligence Excellence Centre, Kaunas University of Technology, 51423 Kaunas, Lithuania
* Correspondence: olelap@ktu.lt

Abstract

Background: Efficient parking navigation in large and dynamic parking areas requires systems that can adapt to real-time conditions and provide precise vehicle localization. *Methods*: This paper presents a smart car parking navigation module that integrates camera-based vehicle perception, homography-based ground-plane localization, mobile GNSS positioning, and dynamic route planning into a unified framework. Instance segmentation (YOLOv8n-seg) is used to detect vehicles and extract ground-contact regions, which are associated with parking slots defined in a GeoJSON-based site model. Mobile GNSS data are fused with visual observations via spatio-temporal proximity scoring to enable robust user–vehicle matching without optical identification. An A* routing algorithm dynamically computes and updates navigation paths, adapting to lane obstructions and slot availability in real time. *Results*: Experimental evaluation on a real six-camera parking facility shows that the proposed segmentation-based localization reduces mean error from 0.732 m to 0.283 m (61.3% improvement), with the 95th-percentile error dropping from 1.892 m to 0.908 m, and outperforming the bounding-box baseline in 85.3% of detections. *Conclusions*: These results demonstrate that sub-meter vehicle localization and reliable user–vehicle association are achievable using standard surveillance cameras without specialized infrastructure, offering a scalable and cost-effective solution for intelligent parking navigation.

Keywords: smart parking; parking navigation; vehicle detection; homography; mobile data fusion; indoor navigation

Academic Editors: Mladen Krstić, Željko Stević and Snežana Tadić

Received: 31 January 2026
Revised: 7 March 2026
Accepted: 7 April 2026
Published: 9 April 2026

1. Introduction

The rapid growth of urban mobility has significantly increased the demand for efficient parking management systems, particularly in large and densely populated areas [1–3]. Drivers often spend a considerable amount of time searching for available or assigned parking spaces, leading to frustration, unnecessary traffic obstruction, and increased fuel consumption. Traditional parking guidance solutions (typically based on static signage, basic occupancy indicators, or fixed navigation routes) are increasingly inadequate in such environments. This limitation is especially evident in parking facilities where occupancy levels, driver behavior, and vehicle flows change continuously throughout the day. As a result, there is a growing need for intelligent parking navigation systems that guide users in real time while dynamically adapting to changing conditions within parking complexes.

An evolution in parking management from sensor-based systems [2,3] to IOT solutions [4] to vision-based approaches [1,5] has been observed. However, existing solutions based on vision focus more on detection and occupancy classification [6–8] without providing navigation guidance or user interaction. Meanwhile, high-precision solutions based on LiDAR or RGB-D [9] for systems requiring wide-deployment remain too costly.

Current smart parking research has increasingly focused on combining multiple technologies, including connected vehicle systems, indoor navigation, and artificial intelligence. A particularly promising direction involves driver-facing solutions that not only guide users to assigned parking spaces but also adapt routing decisions based on the current state of the parking environment. By integrating interactive maps with real-time path generation, such systems can provide clear, step-by-step visual guidance from the entry point to the target parking space. However, routes generated using static calculations quickly become ineffective in highly dynamic micro-environments, where unpredictable vehicle movements, temporary blockages, and rapidly changing slot availability are common. Consequently, effective parking navigation requires predictive and adaptive routing strategies [10–12].

Artificial intelligence (AI) provides an opportunity to significantly enhance the responsiveness and efficiency of parking navigation systems. AI-driven routing can account for multiple factors, including real-time occupancy, internal traffic flow, and anticipated availability. This enables the system to compute not only the shortest route but also the most efficient, context-aware route. Furthermore, AI-based adaptation can improve the overall user experience by reducing congestion during peak periods, when even minor inefficiencies may lead to substantial delays.

Through deliberate methodological design, the proposed framework addresses these gaps. Instance segmentation solves the known inaccuracy of axis-aligned boxes under oblique camera view through precise ground-contact estimation replacing bounding-box detection. Holography-based projection eliminates the need for specialized sensors through the conversion of image space detections into metric coordinates using only cameras. Moreover, mobile GNSS fusion proposes another advantage by enabling user–vehicle associations without relying on optical identification, which might be unreliable in different scenarios like bad weather. Finally, the adaptive A*-based routing algorithm provides dynamic path planning that changes based on real-time updates; a capability does not present in existing perception-only systems.

In this paper, we investigate the design and feasibility of a smart parking navigation module that integrates real-time navigation, AI-driven route adaptation, and hybrid communication mechanisms. The research is driven by challenges observed in large-scale parking facilities, such as truck parking lots, where existing modern solutions often fail to provide accurate guidance and real-time decision support. The aim is to create a scalable, comprehensive smart parking lot architecture that improves the driver's experience and optimizes traffic flow in parking lots. The results of this study demonstrate that through commodity cameras, sub-meter localization is achievable, providing a scalable baseline for future research in multi-facility deployment and autonomous parking systems. The remainder of this paper is organized as follows. Section 2 reviews related work on vision-based parking systems, localization methods, and mobile-assisted navigation. Section 3 describes the dataset, including the geospatial parking layout, multi-camera video recordings, and synthetic mobile positioning data. Section 4 presents the system pipeline and experimental methodology, including model selection and geometric calibration. Section 5 details the proposed system components and experimental results. Section 6 concludes the paper with a summary of contributions, limitations, and directions for future research.

2. Related Works

Intelligent parking systems have emerged as a prominent research focal point in recent years, driven by advancements in image processing and mobile positioning technologies. The existing literature predominantly addresses isolated sub-problems, including multi-camera perception [13], IoT-enabled parking management, parking space detection [14], and occupancy estimation, as synthesized in several comprehensive survey studies [6,10,15,16]. However, there remains a significant research gap regarding systems that jointly integrate user–vehicle associations and in-lot navigation within a unified operational workflow.

Recent progress in this domain has been largely propelled by deep learning-based methodologies, particularly Transformer architectures and hybrid models, which offer enhanced detection accuracy and robustness in complex environments [7–9,17]. Specifically, modern YOLO-based architectures have been widely explored for smart parking applications due to their high performance and efficient deployment capabilities on edge and cloud platforms [9,18,19]. These approaches achieve high precision in vehicle and parking spot detection while reducing reliance on dedicated sensor infrastructures. For instance, lightweight YOLO variants have been employed to identify vehicles and vacant spots, utilizing nearest-neighbor searches for candidate selection [20]. Other studies combine YOLO-based detection with CNN-driven homography estimation to generate bird's-eye-view representations for clustering-based parking-slot extraction, achieving occupancy classification accuracies of up to 90% [4]. Despite these advancements, such systems remain primarily focused on perception and lack integrated mobile user interaction or real-time navigation guidance. Furthermore, a common limitation is the reliance on axis-aligned bounding boxes, which fail to accurately represent vehicle orientation relative to parking slots. Moreover, most deep learning-based approaches rely on axis-aligned bounding boxes, which do not accurately capture vehicle orientation relative to parking slots.

To overcome the limitations of manual annotation, label-free approaches have been proposed for parking space detection. Deformable DETR-based vehicle tracking has been used to infer parking space locations from vehicle motion patterns, while also enabling robust multi-scale object detection and improved handling of geometric variability in complex parking environments [17]. Another parking occupancy detection system has been proposed using a one-stage RetinaNet detector combined with region-based CNNs, aiming to replace sensor-based solutions and manual video segmentation while maintaining reasonable detection accuracy [6]. Additionally, some studies [16] analyze zenith-view imagery to semi-automatically extract parking spaces using classical image processing techniques and estimate real-time occupancy with R-CNNs, achieving accuracies exceeding 98%. While effective, these approaches typically rely on scene-specific calibration and focus on parking space detection rather than user-centric navigation or localization.

More recent research has investigated camera-based parking perception under challenging outdoor conditions, emphasizing robustness to varying illumination and environmental factors [7,21]. Advanced solutions using multi-camera or specialized camera configurations have also been proposed to improve spatial accuracy and scene understanding. For instance, vehicle pose estimation can be achieved using ceiling-mounted cameras calibrated to a common coordinate frame, where point-cloud registration techniques such as fast-GICP enable centimeter-level localization accuracy [9]. Another line of work addresses parking occupancy detection using monocular camera data combined with homography-based inverse perspective mapping and temporal difference analysis to identify space usage over time [19]. In parallel, parking area type classification can be explored using bird's-eye-view representations generated from, for example, YOLO model-based polygon detections and surround-view camera systems [20].

Recent survey and review studies have systematically analyzed smart parking systems and vision-based parking management pipelines, focusing on parking space detection, occupancy classification, vehicle counting, datasets, benchmarking practices, and system architectures [1,3,18,22–25]. These works consistently highlight challenges related to robustness, dataset dependency, and static camera assumptions, while emphasizing the need for more automated and scalable solutions. However, user-centric functionalities, such as user–vehicle matching, mobile user integration, and navigation guidance, remain largely unaddressed.

Table 1 provides a qualitative comparison of representative smart parking systems, highlighting differences in localization methods, navigation capabilities, user–vehicle matching, and key limitations. The comparison illustrates that most existing approaches address isolated functionalities, whereas the proposed system integrates localization, navigation, and user-centric features within a single framework.

Table 1. Qualitative comparison of smart parking systems.

System Category	Localization	Navigation	U–V Matching	Limitation
Vision-based on-street systems, [7,26].	Image-based	No	No	Limited user guidance
Fixed-camera homography systems, [19].	BEV/Homography	No	No	Vision-only, no mobile integration
Label-free vision systems, [17,27].	Image-based	No	No	No navigation support
RGB-D infrastructure systems, [9].	3D point-cloud	No	No	High infrastructure cost & deployment complexity
Camera-based guidance systems, [28].	IPM/Perspective transform	Yes	No	Guidance included, limited user association
Mobile app allocation & indoor navigation systems, [29].	Mobile positioning	Yes	No	Mobile guidance; no camera-based localization
Proposed system (this study)	Homography + Mobile fusion	Yes	Yes	—

Existing research has primarily addressed individual components of smart parking systems, including parking space discovery, occupancy detection, IoT-based management, vehicle localization, and mobile-assisted positioning. These approaches are often evaluated independently, and comparatively fewer works combine camera-based localization with mobile-assisted user association and in-lot navigation in one operational workflow. Motivated by this gap, this work presents an integrated parking navigation framework that couples visual localization with mobile-assisted user–vehicle matching and camera-assisted navigation within a single architecture.

3. Dataset

The study utilized three categories of data, which are geospatial data representing the parking layout, multi camera video recording of real parking activity and synthetic mobile positioning data, which is generated for controlled experimental evaluation.

The geospatial and video data were collected from a real parking facility, which is located in Santaka Valley, Kaunas, Lithuania. The recordings were provided by the parking administrator and reflect real operating conditions.

The mobile datasets were synthetically generated to emulate a realistic GNSS measurement while ensuring synchronization with a video stream. This controlled generation also enables reproduceable evaluation of the mobile visual data fusion model.

3.1. Dataset Preparation from Parking Lot Using QGIS

To create the dataset representing the parking lot, a geospatial area of the parking lot was constructed in the form of a GeoJSON file. The dataset creation process was done by integrating the architectural plan of the parking lot with georeferencing and digitization tools, which are available in QGIS (version 3.44.4). The workflow consisted mainly of three steps, which were first importing the parking-lot blueprint, second georeferencing the plan, and then finally manually annotating the spatial features for export. Figure 1 shows an overlay display of a construction plan and a satellite image of a parking lot located in Santaka Valley (Kaunas, Lithuania).

Figure 1. Map overlaying with QGIS: (red bounding) parking boundaries, (blue bounding) boundaries of each parking space.

Once the original parking-lot plan was imported into QGIS as a raster layer, it was aligned with a real-world spatial reference using control points, which were visible in both the plane and the satellite imagery, as the blueprint lacked coordinate information. These ground control points were used to perform the georeferencing, hence enabling the QGIS to correctly position the parking-lot layout within the known coordinate system.

After the plan was aligned spatially, a vector layer was created to annotate the relevant structural elements of the parking lot with labels, indicating different types of positions, such as general, disabled, and free with a Boolean value. The polygon and polyline features were manually digitized to represent parking boundaries with utmost care to ensure that the digitized geometry accurately followed the scale reference plan and maintained the topological consistency.

In the final step, which is digitization, the exported vector layers in the GeoJSON file were cross-checked with GeoJSON.io to ensure that the annotated geometry matches with the map box, which is implemented in the application. The GeoJSON format was chosen for its compatibility with modern mapping libraries, web-based visualization tools, and downstream processing pipelines used in homography estimation and navigation modeling. The resulting GeoJSON dataset will serve as the ground-truth spatial representation of the parking lot and forms the basis for feature correspondence generation in the homography computation stage. Figure 2 shows the mapping of the finished GeoJSON onto a satellite image of the parking lot.

3.2. Video Dataset Description

The parking lot is monitored by six fixed video cameras mounted at elevated positions to provide continuous visual coverage of the area. These cameras were positioned to cover

different functional zones of the parking area, including entry and exit lanes, central driving aisles, and multiple groups of parking slots.

Figure 2. Vehicle detection and parking space segmentation overlay.

The parking lot administrator provided recorded video streams from all six cameras. The recordings were captured during regular operating hours and reflect realistic traffic conditions, including periods of low and high vehicle density. The video material includes a wide range of real-world scenarios, such as vehicle arrivals and departures, temporary stops, pedestrian movement, and varying lighting conditions due to weather and time of day. Each video sequence is stored in a standard digital format with a fixed frame rate and resolution, ensuring temporal consistency across the dataset.

3.3. Synthetic Mobile Data Generation

To enable the experimental evaluation of the mobile–visual data fusion framework without relying on real smartphone logs, a synthetic mobile dataset is generated from video-based vehicle observations. The objective of this stage is to emulate realistic mobile GNSS measurements, including spatial noise, temporal sampling, and user identity assignment, while preserving full synchronization with the video stream.

A small synthetic user dataset was generated to support the simulation of vehicle–driver assignments, parking-slot assignments, and parking validation. Since the prototype system does not depend on real data, the dataset was generated manually using static predefined values. The dataset is in JSON format, where each record includes a unique identifier, basic profile information, a predefined parking slot assigned to the parking area, and a description of the vehicle associated with this user. The main components for each record in detail are as follows:

- *User information* contains a unique identifier, name, and email address representing typical driver profiles.
- *Assigned parking slot* per user to validate the logic of slot allocation and parking correctness.
- *Vehicle information* contains license plate value, color, make, model, and year of the vehicle.

This synthetic dataset provides a simple yet sufficient foundation for testing the system's behavior and different functionalities.

The video frame rate is denoted by f, where f represents the number of frames per second, and t_0 denotes the absolute timestamp of the first video frame. To approximate a realistic smartphone reporting rate, only every 10th frame is used for mobile data emulation. The timestamp of the sampled frame with index k is computed as

$$t_k = t_0 + \frac{k}{f}. \tag{1}$$

For each timestamp t_k, all vehicle detections available in the corresponding frame are used as input for mobile user simulation. Each detected vehicle is, over time, associated with a world-space track based on coordinate proximity. User identities are assigned at the track level to ensure temporal consistency of the simulated mobile trajectories. The primary identity assignment follows a slot-based approach. A predefined mapping, slot_to_user, $s \mapsto u_s$, is used to associate parking slots with corresponding user identifiers u_s. If a vehicle track remains within the same parking slot s for at least N_{dwell} consecutive sampled frames while being inside the parking area, then the corresponding user identifier u_s is permanently assigned to the track:

$$\text{if } s_k = \text{s} \forall \text{k} \in [k_0, k_0 + N_{\text{dwell}} - 1] \Rightarrow user\_id = u_s. \tag{2}$$

Tracks that never satisfy the parking-slot condition but are observed inside the parking area at least once are handled by a secondary fallback mechanism. Each such track is assigned as a unique unused user identifier from the available user pool. If the pool is exhausted, additional synthetic identifiers of the form $USRxxx$ are generated. The resulting mapping $track\_id \rightarrow user\_id$ is then propagated to all detections belonging to the same track. This hybrid strategy ensures that all mobile users are represented by continuous, physically consistent trajectories.

Camera-derived geographic coordinates represent near-ground-truth positions and therefore must be degraded to reflect realistic mobile GNSS accuracy. For each mobile observation with a true geographic position $(\lambda,\ \phi)$, a random positioning error is applied.

First, the accuracy radius $r \sim \mathcal{U}\{1, 2, 3, 4, 5\}$ meters is sampled independently for each observation. A random orientation $\theta \sim \mathcal{U}(0, 2\pi)$ defines the displacement direction. The metric offset is computed according to the formula:

$$\Delta x = r\cos\theta, \Delta y = r\sin\theta, \tag{3}$$

where Δx and Δy represent the horizontal displacement components in the local metric coordinate system.

Using the local metric-to-degree conversion at the latitude ϕ, $(m_\phi = 111,320, m_\lambda = 111,320\cos\phi)$, the angular displacement is calculated using the following formulas:

$$\Delta\phi = \frac{\Delta y}{m_\phi}, \Delta\lambda = \frac{\Delta x}{m_\lambda}. \tag{4}$$

The perturbed mobile position is then obtained by applying this displacement to the original coordinates, resulting in

$$\lambda' = \lambda + \Delta\lambda, \phi' = \phi + \Delta\phi. \tag{5}$$

The sampled radius r is stored explicitly as the reported mobile positioning accuracy $\sigma = r$. This procedure guarantees that each synthetic mobile measurement exhibits a controlled isotropic spatial error with known magnitude.

After temporal sampling, identity assignment, and noise injection, each synthetic mobile measurement is represented below:

$$m_i = \{u_i, \lambda_i', \phi_i', t_i, \sigma_i\}, \tag{6}$$

where u_i is the assigned user identifier, (λ_i', ϕ_i') is the noisy geographic coordinate, t_i is the synchronized timestamp, and σ_i is the mobile positioning accuracy in meters.

All high-precision internal attributes used during generation (ground-plane coordinates, bounding boxes, slot identifiers) are removed from the exported dataset. The

resulting synthetic mobile stream faithfully replicates the statistical behavior of real smartphone GNSS measurements and is fully synchronized with the video-based vehicle observations, making it suitable for controlled experimental evaluation of mobile–visual data fusion algorithms.

4. System Pipeline and Experimental Methodology

The proposed smart parking navigation system will follow a modular processing pipeline that transforms raw video streams and mobile positioning data into actionable navigation guidance. Figure 3 shows the complete flow across all system modules, from video cameras to the mobile app interface. The pipeline consists of seven sequential modules, each built on the outputs of the previous one.

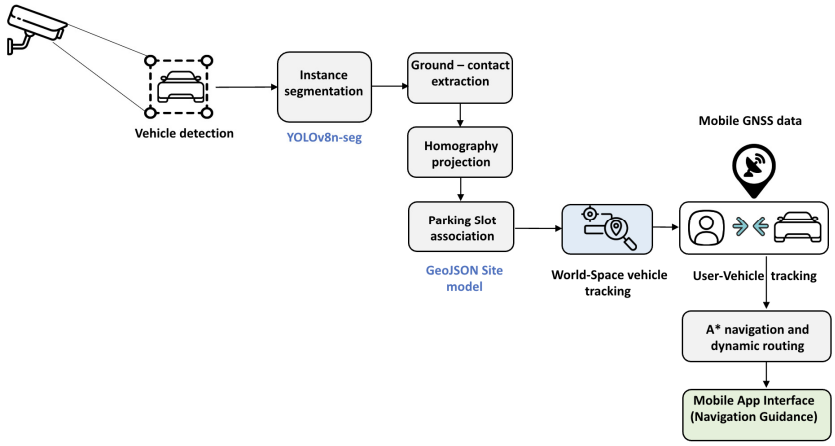

Figure 3. Proposed smart parking navigation system pipeline.

The first module represents the video camera, where the video stream acquisition process occurs. In this module, the system receives continuous video feeds from fixed cameras across the parking lot covering different areas. Video frames are then extracted at the native camera frame rate and passed to the next component, the detection module.

The second module receives video frames and processes them with a YOLOv8n-seg instance segmentation model. The model's output is a set of detections, each consisting of a bounding box, a pixel-wise segmentation mask, a class label, and a confidence score. For subsequent processing, only classes belonging to vehicle-related classes such as cars, motorcycles, buses, and trucks are preserved. Hence, the generated segmented masks provide precise boundaries, which are necessary for the accurate ground-plane localization that will be implemented in the next module.

The third module estimates the ground-contact region of the detected vehicle by extracting the lowest foreground pixels from the generated segmentation mask along each image column. Moving on, these image-space contact points are then transformed into metric world coordinates using a pre-calibrated inverse homography matrix. The output of this module will contain the geographic position as the longitude and latitude of each vehicle present in the parking lot coordinate

The fourth module validates the generated world-space vehicle position if it is within the parking site boundary polygon. If the vehicle is within the boundary, another test, a point-in-polygon test, is performed against all the predefined slot polygons stored inside the GeoJSON site model to determine if the vehicle is located within a valid spot to assign a unique slot identifier. Moreover, car positioning classification, whether correctly parked or poorly parked, is done through parking quality evaluation by computing the fraction of ground-contact points within the assigned slot boundary.

The fifth module links detected vehicles across consecutive frames to form temporally consistent tracks. This is done by minimizing the Euclidean distance between existing track positions and newly detected ones while constrained by a maximum gating distance and slot-consistency behavior. Also, tracks that remain inactive or unobserved beyond a pre-defined timeout threshold are removed from the tracker.

The sixth module uses data that is received from the user's phone, such as mobile positioning data combining user identifiers, GNSS coordinates, timestamps, and reported accuracy. The mobile coordinates are then transformed into the same local ground-plane coordinate system used previously for the camera-based localization. While accounting for spatial distance normalized by GPS accuracy and temporal offset, the user–vehicle association is done using a spatio-temporal proximity score. In addition, a delayed refinement process re-evaluates the association done after a vehicle stabilizes in a slot to correct any early matching errors caused by GPS noise or latency.

The seventh and last module of the system operates upon successful user–vehicle matching in which the A* search algorithm finds the optimal path from the vehicle's current position to the associated parking slot. This module continuously monitors incoming detection data and dynamically responds by recalculating to find the shortest path, taking into consideration any lane obstructions or slot availability changes.

The outputs of this pipeline include annotated video frames displaying vehicle detections with color-coded parking status, slot occupancy detections, user–vehicle association, and detailed navigation guidance provided through the user's mobile application.

4.1. Evaluation and Comparison of Models in an Experimental Environment

To find the most optimal model for our architecture, we evaluated multiple object detection and segmentation models, which include variants of YOLOv8, YOLO11 [5] and a Faster R-CNN (X101-FPN) [18], on our video dataset for car detection. The comparison focused on FPS (frames per second), latency, detection persistence, miss rate, average confidence, average jitter, segmentation capability, model load time and GPU memory usage. The table (Table 2) below shows the evaluation of the models on the video dataset.

Table 2. Performance comparison of deep learning models for car detection and segmentation.

Model	FPS	Latency	Persistence	Miss	Avg Confidence	Avg Jitter	Load	Peak GPU Alloc (MB)
yolov8n	25.77	38.80	96.59	3.41	0.85	43.87	25.24	836.14
yolov8n-seg	22.28	44.69	96.92	3.08	0.89	8.14	29.38	868.48
yolov8x	14.79	67.60	99.52	0.48	0.92	71.14	166.02	1168.90
yolov8x-seg	15.28	65.46	99.65	0.35	0.92	44.72	344.51	1191.09
yolo11n	22.52	44.40	97.62	2.38	0.86	21.19	32.69	609.40
yolo11n-seg	20.11	49.74	96.88	3.12	0.84	35.96	35.09	641.40
yolo11x	16.48	60.67	99.45	0.55	0.91	18.17	158.43	906.93
yolo11x-seg	13.49	74.11	99.45	0.55	0.91	18.17	171.18	1200.35
FasterRCNN _X101_FPN	3.45	289.79	99.94	0.06	0.99	54.06	1539.02	2732.85

Our application requires a pixel-level segmentation mask for accurate vehicle boundary extraction and precise spatial analysis. This was one of the major reasons for eliminating Faster R-CNN (X101-FPN), YOLOv8n, YOLOv8x, YOLO11n, and YOLO11X, despite them offering higher FPS or lower latency, as the models could not provide the semantic segmentation that is necessary for our specific case. This reduced the model pool to four models, YOLOv8n-seg, YOLOv8x-seg, YOLO11n-seg and YOLO11x-seg.

The extra-large models YOLOv8x-seg and YOLO11x-seg showed higher detection performance with persistence rates of 99.95% and 99.45% respectively, with the lowest miss rates (0.35% and 0.55%) and highest confidence scores (0.92 and 0.91), but this 2–3% improvement came with substantial computational costs: significantly lower FPS and much higher GPU memory consumption. Because of our deployment constraints and real-time performance requirements, these models were deemed too resource-intensive for our use case.

Between the two nano segmentation models, YOLOv8n-seg was selected as the primary model. While YOLO11n-seg gave a 26% lower GPU memory usage (641 MB vs. 868 MB), YOLOv8n-seg was able to outperform it across major metrics, such as by having 10% higher FPS, 5 ms lower latency, better average confidence and, most notably, 4.4× lower jitter. The dramatically lower jitter in YOLOv8n-seg ensures a more stable and consistent segmentation mask across the frames, which is a crucial element for attaining reliable vehicle tracking and spatial analysis. The extra 227 MB of GPU is a reasonable trade-off for the significant gains in speed, accuracy and prediction stability, making YOLOv8n-seg the ideal balance between performance and resource efficiency for our deployment scenarios.

4.2. Geometric Modeling, Site Representation, and Camera Calibration

The proposed system relies on an accurate geometric correspondence between the camera image plane and the real-world ground plane of the parking site. This correspondence is established through a combination of geographic site modeling, planar projective transformation (homography), and an optional non-linear mesh-based correction. Together, these components form a unified geometric framework that ensures consistency between spatial data sources and visual observations.

The geometry of the parking site is specified using a GeoJSON representation. The parking area is defined by a top-level polygon describing the site boundary, while individual parking slots are represented as multipolygons, each associated with a unique *slot_id*. All geographic coordinates, provided in longitude λ and latitude ϕ, are converted into a local Cartesian coordinate system expressed in meters. This transformation is performed using an equirectangular approximation centered at a geographic reference point $(\lambda_0,\ \phi_0)$ derived from the following site boundary:

$$x = (\lambda - \lambda_0) \cdot 111{,}320 \times \cos(\phi_0), y = (\phi - \phi_0) \times 111{,}320. \tag{7}$$

The resulting local ground-plane coordinate system $(x,\ y)$ serves as a common spatial reference for all subsequent computations, including vehicle localization, slot association, mobile user positioning, and world-space tracking.

To establish a mapping between the image plane and the physical ground plane, a planar projective transformation is employed. This transformation is represented by a 3×3 homography matrix H, which maps world coordinates $(x,\ y)$ to image coordinates $(u,\ v)$ in homogeneous form:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}. \tag{8}$$

The inverse transformation projects image measurements back to the ground plane:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim H^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}. \tag{9}$$

The homography is applied consistently throughout the system to project parking boundaries and slot geometries into the image for visualization, to convert detected vehicle image positions into metric ground coordinates, and to compute geographic positions for display. Figure 4 illustrates the described method.

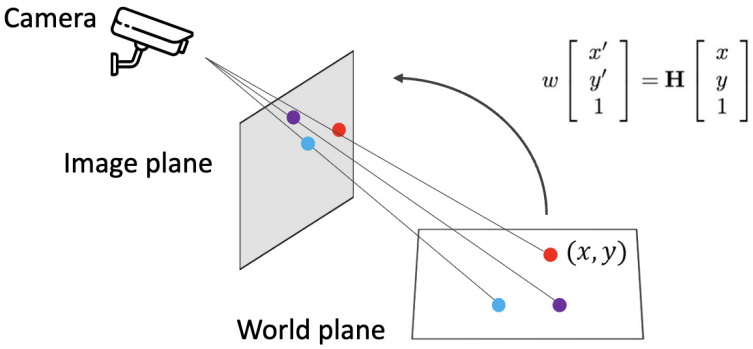

Figure 4. Illustration of planar homography between the ground plane and the camera image.

The estimation of the homography is performed using a dedicated interactive calibration tool. The tool operates either on a still image or on the first frame of a video sequence. For calibration, a control parking slot is selected, and its known world-space polygon $\{(x_i, y_i)\}$ is paired with a set of draggable control handles $\{(u_i, v_i)\}$ in the image. By manually aligning these handles with the visible slot corners, a set of corresponding world–image point pairs are obtained:

$$(x_i, y_i) \leftrightarrow (u_i, v_i), i = 1, \ldots, N,\ N = 4, \tag{10}$$

The homography matrix H is obtained as the solution of the following optimization problem:

$$H = \arg\min_{H} \sum_{i=1}^{N} \| \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} - H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \|. \tag{11}$$

The estimation is performed using a robust RANSAC-based procedure to suppress the influence of potential outliers among the manually selected correspondences.

The resulting transformation matrix H, together with its inverse H^{-1} and the geographic reference origin $(\lambda_0,\ \phi_0)$ is stored in a YAML configuration file and used by the runtime system. Figure 5 shows a tool for creating a homography matrix based on a camera image and GeoJSON. First, the user must specify four green points that correspond to a parking space in the camera image. Since many street cameras have a fisheye effect and other defects, the user can use a blue grid to correct the image.

Although a single homography provides an accurate approximation for planar perspective projection, real camera systems frequently exhibit non-linear distortions that cannot be fully compensated by a projective model alone. To address this limitation, the calibration tool supports an additional mesh-based warp refinement. A regular control grid $\mathcal{G}_s = \{(u_{ij}, v_{ij})\}$ is superimposed over the image, and the user may interactively displace individual grid nodes to define a deformed grid $\mathcal{G}_d = \left\{\left(u'_{ij}, v'_{ij}\right)\right\}$.

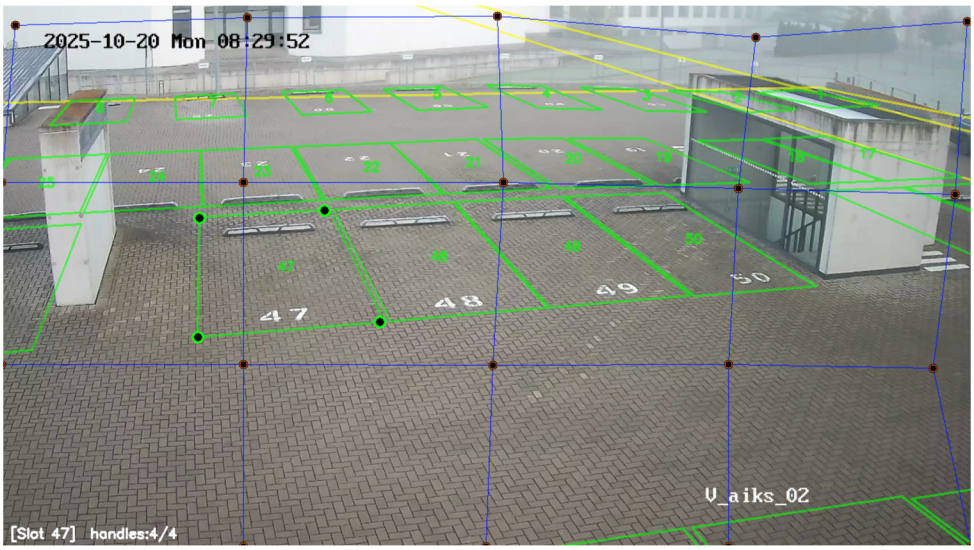

Figure 5. Screenshots of the interactive homography calibration tool with draggable control handle: (green polygons) parking slot boundaries with numbered identifiers and draggable corner handles, (blue grid) lens distortion correction overlay, (yellow polygons) parking boundaries, (bottom-left status bar) currently selected slot and handle count.

For any image point $(u,\ v)$, the displacement is computed by bilinear interpolation of the surrounding grid node offsets:

$$\Delta(u,v) = \text{interp}(\mathcal{G}_d - \mathcal{G}_s),\ (u',v') = (u,v) + \Delta(u,v). \tag{12}$$

During runtime, this displacement field is applied either in forward or inverse form to refine pixel coordinates prior to or after homography projection. Slots that require such non-linear correction are explicitly flagged, allowing the system to apply the mesh-based warp only where necessary, maintaining standard homographic projection elsewhere.

The final output of the calibration stage is a unified geometric configuration stored in a YAML file. This file contains the homography matrix H, its inverse H^{-1}, the geographic origin of the local coordinate system $(\lambda_0,\ \phi_0)$, the mesh warp parameters $\mathcal{G}_s, \mathcal{G}_d$, and a per-slot flag indicating whether displacement-based correction should be applied. This integrated calibration representation ensures geometric consistency and metric accuracy across all subsequent stages of vehicle detection, tracking, slot association, and mobile data fusion.

5. Proposed System and Results

5.1. Vehicle Detection, Parking Slot Association, and Visual Feedback

Vehicle detection in the proposed system is performed per frame using a deep learning-based instance segmentation model from the YOLOv8 family. For each input video frame I_t, the model produces a set of detections

$$\mathcal{D}_t = \left\{(B_i, M_i, c_i, p_i)\right\}_{i=1}^{N_t}, \tag{13}$$

where $B_i = (x_1, y_1, x_2, y_2)$ is the bounding box, M_i is the pixel-wise segmentation mask, c_i is the predicted class label, and p_i is the confidence score. Only vehicle-related classes $c_i \in \{\text{car}, \text{motorcycle}, \text{bus}, \text{truck}\}$ are retained for further processing. The use of instance segmentation, instead of bounding boxes alone, allows a more accurate estimation of the physical contact between vehicles and the ground plane.

For each detected vehicle, the ground-plane position is estimated using the segmentation mask whenever available. For each image column u, the lowest foreground pixel is extracted from the mask as

$$v(u) = \max\{v \mid M_i(v, u) = 1\}. \tag{14}$$

The resulting set of image-space ground-contact points is calculated:

$$\mathcal{P}_i^{\text{img}} = \left\{(u_k, v_k)\right\}_{k=1}^{K}. \tag{15}$$

If mesh-based non-linear correction is enabled, each point is first inverse-warped as

$$\left(u'_k, v'_k\right) = W^{-1}(u_k, v_k), \tag{16}$$

where W^{-1} denotes the inverse displacement mesh warp. The corrected image points are then mapped to the local ground plane using the inverse homography:

$$(x_k, y_k, 1)^\top = H^{-1}(u'_k, v'_k, 1)^\top. \tag{17}$$

The final ground-plane vehicle position is obtained as the mean of all projected contact points:

$$x = \frac{1}{K}\sum_{k=1}^{K} x_k,\ y = \frac{1}{K}\sum_{k=1}^{K} y_k. \tag{18}$$

Once the world-space vehicle position $(x,\ y)$ is obtained, a geometric point-in-polygon test is applied to determine whether the vehicle lies inside the parking site boundary polygon Ω:

$$\text{inside\_site}(x, y) = \begin{cases} 1, & (x, y) \in \Omega, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Vehicles located outside the boundary are excluded from parking analysis. For vehicles inside the site, slot association is performed by testing whether $(x,\ y)$ belongs to one of the predefined parking-slot polygons Ω_s:

$$\text{slot\_id}(x, y) = \arg\max_s 1\{(x, y) \in \Omega_s\}. \tag{20}$$

Depending on the calibration configuration, this test is carried out either directly in the world plane or in the image plane $(u,\ v)$ for slots that require displacement-based mesh correction. This assigns a unique *slot_id* to each detected vehicle located in a valid parking slot.

Parking quality is evaluated by analyzing how well the vehicle occupies its assigned slot. When segmentation-based ground-contact points are available, the fraction of contact points that fall inside the assigned slot polygon is computed as

$$r = \frac{1}{K}\sum_{k=1}^{K} 1\{(x_k, y_k) \in \Omega_{slot}\}. \tag{21}$$

If $r < r_{\min}$, where $r_{\min}$ is a predefined threshold, the vehicle is classified as poorly parked.

The complete results of detection, slot association, and parking quality assessment are visualized in the output video stream. Figure 6 shows examples of different use cases. Each vehicle is color-coded according to its status: well, parked inside a slot, poorly parked inside a slot, inside the parking boundary but outside any slot, or outside the parking

area. The assigned *slot_id*, geographic coordinates obtained via the inverse geographic projection and the user identifier are overlaid next to each detected vehicle to ensure full interpretability of the results:

$$(\lambda, \phi) = f^{-1}(x, y), \tag{22}$$

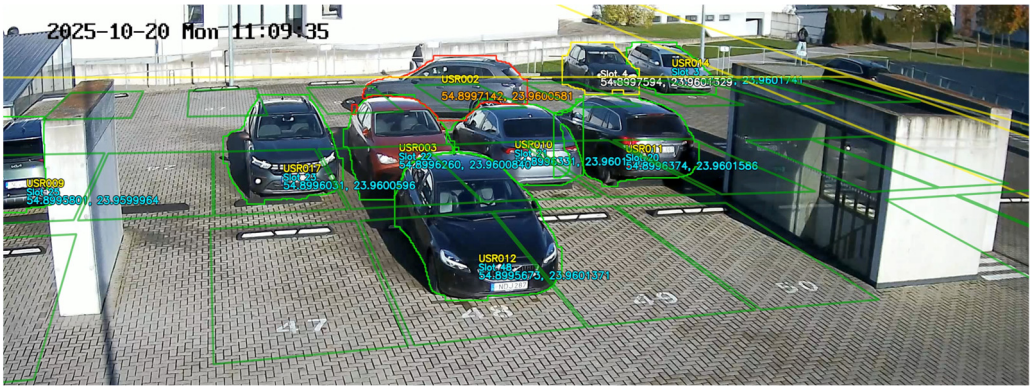

Figure 6. Examples of detected vehicles and parking analysis in three typical cases: (green bounding) correctly parked vehicle inside a slot, (red bounding) vehicle inside the parking boundary but outside any slot, (yellow bounding) poorly parked vehicle with insufficient overlap with the assigned slot.

5.2. Evaluation of Segmentation Based Localization

To validate the effectiveness of the proposed segmentation-based vehicle localization described in Section 5.1, a comparative evaluation was conducted with the traditional bounding box method as a baseline.

5.2.1. Baseline Localization Method

The baseline method is by projecting the bottom center point of the detection bounding box to the world coordinates, which is done by using the same homography transformation. Given a bounding box with corners (x_1, y_1, x_2, y_2), the ground position projected is computed as follows:

$$(\hat{x}, \hat{y}) = H^{-1} \times \left(\frac{x_1 + x_2}{2}, y_{2,}, 1 \right). \tag{23}$$

where y_2 represents the bottom of the edge of the bounding box. This approach assumes that the bottom edge of the bounding box corresponds to the vehicle's ground-contact point. This holds approximately for front-facing vehicles, but degrades when the viewing angle becomes oblique, where the bounding box geometry poorly represents the vehicle's geometry.

5.2.2. Ground Truth Definition for Localization Methods Evaluation

Camera-derived parking-slot geometry is used to define a slot-consistency ground truth for evaluating vehicle localization. For each detection, the estimated ground-contact position is projected into world coordinates and a point-in-polygon test is applied against all parking-slot polygons. If the segmentation-based estimate lies inside a slot, that slot is selected as the reference slot s(i.e., $gt\_slot\_id = s$); detections for which no valid slot can be determined are excluded from the evaluation.

Rather than using a single reference point (e.g., the centroid), the evaluation relies on a depth-inside-slot metric that measures how safely the estimated position lies within the

selected slot polygon. For a position p and the slot polygon P_s, the signed margin to the slot boundary is defined as

$$m(\mathrm{p}, P_s) = \begin{cases} +d(\mathrm{p}, \partial P_s), & \mathrm{p} \in P_s \\ -d(\mathrm{p}, \partial P_s), & \mathrm{p} \notin P_s, \end{cases} \tag{24}$$

where $d(\mathrm{p}, \partial P_s)$ is the Euclidean distance to the nearest polygon edge.

For each slot s, an inradius estimate r_s (maximum achievable inside-margin) is precomputed via random sampling within the polygon. The localization error for an estimate p is then computed as follows:

$$e(\mathrm{p}, s) = \max(0,\ r_s - m(\mathrm{p}, P_s)), \tag{25}$$

so that positions deeper inside the slot (larger positive margin) yield smaller errors, while positions near the boundary or outside the slot are penalized accordingly. For visualization purposes only, the slot centroid is

$$(\overline{x}_s, \overline{y}_s) = \left(\frac{1}{N} \sum_{i=1}^{N} x_i,\ \frac{1}{N} \sum_{i=1}^{N} y_i \right), \tag{26}$$

where (x_i, y_i) are the vertices of the slot polygon and may be displayed as a stable anchor in debug outputs, but they are not used to compute the evaluation metric.

Moving vehicles and vehicles outside parking slots are excluded, since no reliable slot-based reference can be established for them without additional instrumentation and data.

5.2.3. Results of Evaluation of Localization Methods

The evaluation was performed on the video sequence from four fixed surveillance cameras which were monitored from different zones of the parking lot. Table 3 represents the average localization error. The proposed segmentation-based method consistently achieved a lower mean error than the bonding box, which is the baseline for the four camera positions.

Table 3. Combined localization error.

Method	Mean (m)	Std (m)	Median (m)	P95 (m)
Segmentation (proposed)	0.283	0.240	0.233	0.908
BBox Center (baseline)	0.732	0.568	0.568	1.892

The segmentation method achieved a mean error of 0.283 m, compared with 0.732 m for the bounding box baseline, representing a 61.3% reduction in localization error. In pairwise comparisons, the segmentation method outperformed the baseline in 85.3% of detections (23,909 out of 28,031). The 95% error of segmentation across all the cameras is 0.908 m, compared to 1.892 m for the baseline, which indicates that the segmentation not only improves the average but also reduces the worst-case errors.

To analyze performance under different imaging conditions, detections were stratified by distance from the camera, estimated from the normalized bounding box area. Larger bounding boxes indicated that the vehicles were closer to the camera, while smaller boxes indicated that the vehicles were farther away. This enables the analysis of how segmentation quality affects the localization accuracy.

Table 4 represents the distance from the camera. The improvement can be seen more for vehicles in the near distance zone, where larger bounding boxes were able to

provide more pixels for the segmentation, which enabled more accurate extraction of ground-contact points.

Table 4. Localization error by distance.

Distance	Method	Mean (m)	Std (m)
Near	Segmentation (proposed)	0.253	0.167
	BBox Center (baseline)	0.792	0.538
Far	Segmentation (proposed)	0.302	0.278
	BBox Center (baseline)	0.678	0.587

Figure 7 shows that the bounding box can sometimes be too large, which impacts localization results. Calculating vehicle coordinates using their masks is the most appropriate method.

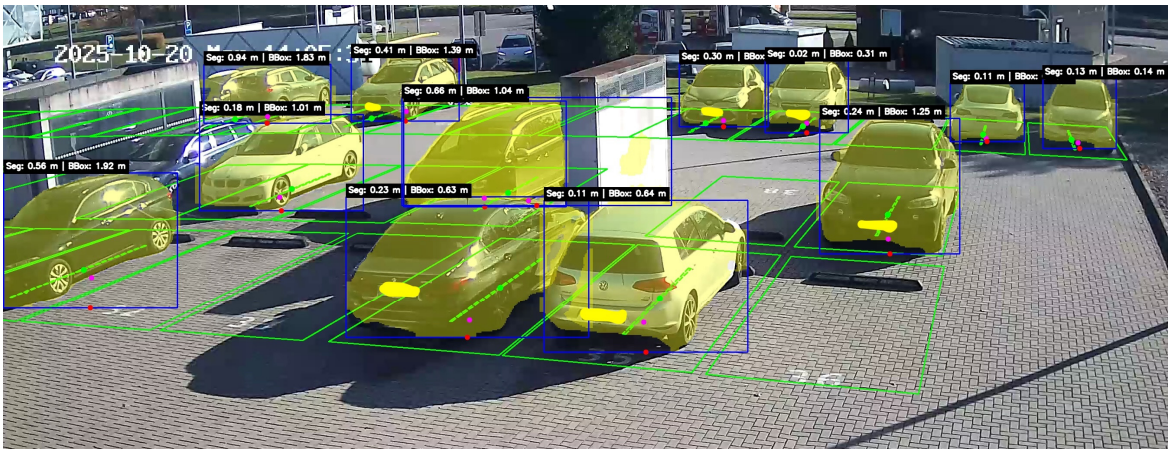

Figure 7. Comparison visualization: (yellow) object masks, (blue) object bounding boxes, (green line) parking space markings, (green dotted line) deepest parking space zone, (red dot) coordinate calculated based on object bbox, (pink dot) coordinate calculated based on object mask.

The evaluation results were able to confirm that the segmentation-based ground contact extracted provides significantly more accurate vehicle localization than the conventional bounding box projection. These improvements can be attributed to two main factors, which are as follows:

- An accurate ground-contact estimation: the segmentation mask can capture the silhouette of the vehicle, which allows the extraction of the corresponding pixels to the ground interface rather than an approximated rectangular boundary.
- Robustness to the viewing geometry: For vehicles that were obliquely viewed, the bounding box bottom edges lay displaced from the true ground-contact points, while the segmentation mask was able to identify them correctly regardless of the vehicle orientation.

This consistent improvement across all four cameras showed that the proposed method was able to generalize well to different camera positions and viewing angles. The stronger improvement for near-distance vehicles is expected, as the larger the objects, the richer the segmentation details for ground-contact estimation.

5.3. Mobile Data Integration and World-Space Tracking

In real outdoor deployments, conventional street-level cameras are strongly affected by adverse environmental conditions such as rain, fog, snow, low illumination, and glare. Under such conditions, reliable license plate recognition becomes unstable or entirely

infeasible. Nevertheless, the requirement to associate each observed vehicle with a particular user remains essential for personalized parking services and navigation. Therefore, instead of relying on optical identification, the proposed system adopts a proximity-based user–vehicle association strategy, in which camera-based vehicle localization is fused with mobile GPS data transmitted by user devices.

The proposed framework integrates two asynchronous data streams: vehicle positions obtained from stationary cameras and mobile user positions obtained from smartphones. Conceptually, this interaction follows the real-world prototype illustrated in the diagrams below, while in experimental evaluation, the mobile stream is emulated using a time-stamped dataset synchronized with the video sequence. Figures 8 and 9 illustrate the overall system prototype and the interaction between the user's mobile device and the server.

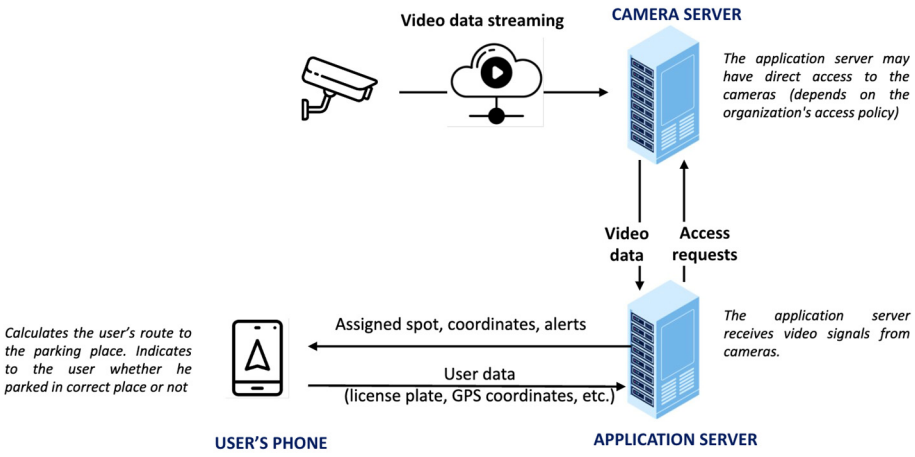

Figure 8. System prototype diagram.

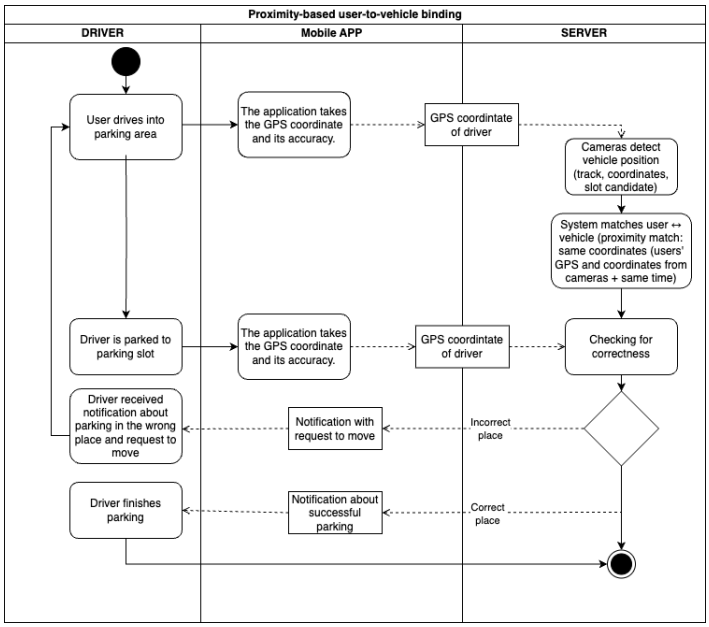

Figure 9. Activity diagram of system prototype.

Each mobile observation is represented as

$$m_i = \{u_i, \lambda_i, \phi_i, t_i, \sigma_i\}, \tag{27}$$

where u_i denotes the user identifier, λ_i and ϕ_i are longitude and latitude, t_i is the timestamp, and σ_i is the reported GPS accuracy in meters. All mobile coordinates are con-

verted into the same local ground-plane coordinate system $(x,\ y)$ used for camera-based vehicle localization:

$$x_i = (\lambda_i - \lambda_0) \times 111{,}320\cos(\phi_0), \qquad y_i = (\phi_i - \phi_0) \times 111{,}320, \tag{28}$$

where (λ_0, ϕ_0) is the geographic reference origin of the parking site. This transformation guarantees metric consistency between visual and mobile measurements.

Each detected vehicle is represented in the tracker by a world-space track

$$\tau_k = \{x_k, y_k, t_k, s_k, \updownarrow_k, \mathcal{H}_k\}, \tag{29}$$

where (x_k, y_k) is the current ground-plane position, t_k is the last update time, s_k is the accumulated parking-presence score, $\updownarrow_k$ is the dominant parking-slot identifier, and $\mathcal{H}_k$ is the history of past world positions. At each frame, a detection at position (x_d, y_d) is matched to an existing track by minimizing the Euclidean distance d_k.

Subject to a gating condition $d_k < d_{\max}$ and to a slot-consistency constraint for slot-stable tracks. If no admissible match exists, a new track is initialized. Tracks that remain unobserved for more than N_{miss} frames are removed from the tracker.

User–vehicle association is driven by spatio-temporal proximity. For each active track without an assigned user, a greedy matching procedure is applied. For a vehicle at position (x_v, y_v) and time t_v, and a mobile user sample at (x_u, y_u) and time t_u, the matching score is defined as

$$S(v,u) = \frac{\sqrt{(x_v - x_u)^2 + (y_v - y_u)^2}}{\sigma_u} + \alpha \ \mid t_v - t_u \mid, \tag{30}$$

where σ_u is the GPS accuracy and α is a temporal weighting coefficient. Only candidates satisfying

$$\mid t_v - t_u \mid < \Delta t_{\max}, \qquad \sqrt{(x_v - x_u)^2 + (y_v - y_u)^2} < \beta\sigma_u \tag{31}$$

are considered valid. The user who minimizes $S(v,u)$ is selected as a provisional match, under the constraint that each user can be associated with at most one active vehicle at a time. If no valid candidate exists, the vehicle remains temporarily unassigned.

To mitigate the effect of GPS noise, delayed updates, and early matching errors, a delayed refinement stage is applied to tracks that exhibit sustained presence inside the parking boundary, stable occupancy of a single slot, and sufficient temporal history. For a temporal window $[t - \Delta T,\ t + \Delta T]$, the mean vehicle position is computed as

$$\overline{x}_v = \frac{1}{N_v}\sum_{i=1}^{N_v} x_i,\ \overline{y}_v = \frac{1}{N_v}\sum_{i=1}^{N_v} y_i, \tag{32}$$

and, analogously, for each user,

$$\overline{x}_u = \frac{1}{N_u}\sum_{j=1}^{N_u} x_j,\ \overline{y}_u = \frac{1}{N_u}\sum_{j=1}^{N_u} y_j, \tag{33}$$

with mean accuracy $\overline{\sigma}_u$ and representative time $\overline{t}_u$. A refined association score is then computed as

$$S_{\text{ref}}(v,u) = \frac{\sqrt{(\overline{x}_v - \overline{x}_u)^2 + (\overline{y}_v - \overline{y}_u)^2}}{\overline{\sigma}_u} + \alpha_{\text{ref}} \mid t - \overline{t}_u \mid . \tag{34}$$

A reassignment is accepted only if

$$S_{\text{ref}}(v, u) < \gamma S_{\text{current}}(v), \tag{35}$$

where $\gamma < 1$ enforces that the new association must be significantly better than the previous one. This criterion prevents frequent identity oscillations.

After greedy matching and optional window-based refinement, the final association $A : \tau_k \rightarrow u_j$ is stored in the track manager and propagated to all detections in the current frame. The updated track state is reused in subsequent frames to ensure tracking continuity, slot stabilization, and consistent mobile–visual data fusion. As a result, the system yields a temporally stable and physically consistent mapping between visually observed vehicles and mobile users, even under conditions of noisy GPS measurements, intermittent mobile updates, and partial visual occlusions. The complete processing chain—from raw detections and mobile data to final user–vehicle association—follows the conceptual logic depicted in the prototype diagrams, while operating entirely on synchronized and emulated data during experimental evaluation.

5.4. Routing Algorithm for Dynamic Parking Navigation

To ensure optimal routing inside the parking area, the system employs an A* search algorithm as its core path-planning module. It is an established best-first search algorithm widely utilized across autonomous navigation modules due to its optimality guarantees when an admissible heuristic is applied [12]. Based on these properties, it was found to be the perfect fit for this dynamic environment represented by the parking area and was implemented on top of the frontend logic.

The parking layout is modeled as a weighted graph $G = (V, E)$. Each node $v \in V$ represents a navigable lane within the closed parking area, and each edge $e \in E$ represents intersections or transitions between lanes. The main goal of the planner is to find the shortest path from the vehicle's entry point to the assigned parking spot for this user.

The following cost function represents how the A* algorithm operates:

$$f(n) = g(n) + h(n). \tag{36}$$

It combines the actual cost $g(n)$ from the start node to the current node n, in addition to a heuristic estimate $h(n)$ of the remaining distance. The heuristic is derived to ensure admissibility and preserve optimality from the geometric distance between nodes. It efficiently prioritizes and chooses the path with the lowest cost.

Most importantly, it allows re-routing based on changes in conditions. This is essential in scenarios where collisions might occur, like a blocked lane, or for real-time updates concerning assigned parking spots. Hence, each time new detection data is received at the frontend, the routing algorithm based on the A* triggers a recalculation and reflects these changes for the user while ensuring to always return the most optimal path based on all the conditions passed.

By combining live dynamic data updates and heuristic-based planning, the system ensures a robust navigation module while optimizing computational efficiency. A highly adaptive replanning capability is added to the system because of this combination, ensuring its usability in real-world unpredictable scenarios.

6. Conclusions

This paper proposes an end-to-end smart parking navigation system that integrates visual vehicle detection, geometric site modeling, mobile positioning, user–vehicle association, and dynamic routing. The created framework addresses key limitations of existing

parking solutions by combining these components into a single coherent architecture capable of operating under real-world conditions.

A key contribution of this study is demonstrating that accurate geopositioning of vehicles can be achieved using standard, widely available surveillance cameras. By applying homography-based ground-plane projection, image-space detections are reliably transformed into metric world coordinates without the need for specialized hardware such as depth sensors, LiDAR, or dedicated positioning infrastructure. This significantly reduces deployment cost and complexity while maintaining sufficient localization accuracy for parking management and navigation tasks.

The experimental evaluation confirmed that the selected YOLOv8n-seg model provides a robust balance between accuracy, stability, and computational efficiency, enabling real-time vehicle tracking and parking-slot association. The fusion of visual localization with mobile GNSS data allows reliable user–vehicle matching even in the presence of noisy or intermittent mobile measurements. Additionally, the integration of an A* routing algorithm enables adaptive, real-time navigation to assigned parking slots, improving traffic flow and user experience within the parking facility. Overall, the results indicate that the proposed system offers a scalable and cost-effective solution for intelligent parking navigation.

Nevertheless, several limitations of the current study should be acknowledged. First, the mobile positioning data used in the experimental evaluation were synthetically generated rather than collected from real smartphone devices, which may not fully capture the variability and edge cases present in real-world GNSS measurements. Second, the system was evaluated on a single parking facility, and its generalizability to environments with substantially different layouts, camera configurations, or lighting conditions remains to be validated. Third, the homography calibration process currently requires manual control point selection, which may introduce inconsistencies and limit scalability across large multi-camera deployments.

These limitations suggest several promising directions for future research. Validation using real mobile positioning data collected from smartphones in live deployments would strengthen the practical credibility of the user–vehicle association module. Extension of the framework to multiple facilities with varying configurations would assess its generalizability and support multi-facility deployment scenarios. Furthermore, developing automated or semi-automated calibration procedures would reduce setup effort and improve consistency. Finally, exploring more advanced tracking and re-identification strategies could further enhance system robustness under challenging conditions such as heavy occlusion, adverse weather, or high vehicle density.

Author Contributions: Conceptualization, A.P.-T.; methodology, O.L., A.T.M. and N.S. (Nathalie Saab); software, O.L. and N.S. (Nathalie Saab); validation, O.L. and A.T.M.; formal analysis, N.S. (Nihad Soltanov); investigation, O.L. and A.T.M.; resources, A.P.-T., A.T.M. and O.L.; data curation, A.T.M., O.L. and N.S. (Nathalie Saab); writing—original draft, O.L., A.T.M., N.S. (Nathalie Saab) and N.S. (Nihad Soltanov); writing—review and editing, O.L. and A.P.-T.; visualization, O.L., A.T.M., N.S. (Nathalie Saab) and A.P.-T.; supervision, A.P.-T.; project administration, O.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All the necessary data are cited in the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. de Almeida, P.R.L.; Alves, J.H.; Parpinelli, R.S.; Barddal, J.P. A systematic review on computer vision-based parking lot management applied on public datasets. *Expert Syst. Appl.* **2022**, *198*, 116731. [CrossRef]
2. Mahmud, S.A.; Khan, G.M.; Rahman, M.; Zafar, H. A survey of intelligent car parking system. *J. Appl. Res. Technol.* **2013**, *11*, 714–726. [CrossRef]
3. Diaz Ogás, M.G.; Fabregat, R.; Aciar, S. Survey of smart parking systems. *Appl. Sci.* **2020**, *10*, 3872. [CrossRef]
4. Lee, C.P.; Leng, F.T.J.; Habeeb, R.A.A.; Amanullah, M.A.; ur Rehman, M.H. Edge computing-enabled secure and energy-efficient smart parking: A review. *Microprocess. Microsyst.* **2022**, *93*, 104612. [CrossRef]
5. Da Luza, G.P.; Satoa, G.M.; Gonzaleza, L.F.G.; Borin, J.F. Smart Parking with Pixel-Wise ROI Selection for Vehicle De-tection Using YOLOv8, YOLOv9, YOLOv10, and YOLOv11. *Internet Things* **2025**, *36*, 101858. [CrossRef]
6. Padmasiri, H.; Madurawe, R.; Abeysinghe, C.; Meedeniya, D. Automated vehicle parking occupancy detection in real-time. In Proceedings of the 2020 Moratuwa Engineering Research Conference, Moratuwa, Sri Lanka, 28–30 July 2020.
7. Morell, J.Á.; Luque, G.; Alba, E. On-street parking space localization with deep learning using low-quality images from public cameras. *Internet Things* **2025**, *32*, 101619. [CrossRef]
8. Grbić, R.; Koch, B. Automatic Vision-Based Parking Slot Detection and Occupancy Classification. *Expert Syst. Appl.* **2023**, *225*, 120147. [CrossRef]
9. Cao, B.; He, Y.; Zhuang, H.; Yang, M. Infrastructure-Based Vehicle Localization System for Indoor Parking Lots Using RGB-D Cameras. *J. Shanghai Jiao Tong Univ.* **2023**, *28*, 61–69. [CrossRef]
10. Ma, J.; Wang, Z. Urban Parking Demand Forecasting Using xLSTM-Informer Model. *IEEE Access* **2025**, *13*, 80601–80611. [CrossRef]
11. Hamad, L.; Khan, M.A.; Menouar, H.; Filali, F.; Mohamed, A. Haris: An Advanced Autonomous Mobile Robot for Smart Parking Assistance. In Proceedings of the 2024 IEEE International Conference on Consumer Electronics, Las Vegas, NV, USA, 6–8 January 2024.
12. Zhao, Y. Automatic parking planning control method based on improved A* algorithm. *arXiv* **2024**, arXiv:2406.15429.
13. Ciampi, L.; Gennaro, C.; Carrara, F.; Falchi, F.; Vairo, C.; Amato, G. Multi-camera vehicle counting using edge-AI. *Expert Syst. Appl.* **2022**, *207*, 117929. [CrossRef]
14. Arun, A.; Sangeeth, S.; Manu, G.K.; Safana, F. Vehicle Detection and Parking Space Estimation using Spatial Analysis and Edge Computing for Temporary and Unorganized Lots. In Proceedings of the 2025 8th International Conference on Circuit, Power & Computing Technologies, Kollam, India, 7–8 August 2025.
15. Sharifai, A.G.; Danlami, M.; Bakari, I.B.; Haruna, U.S. Vacant Parking Space Detection Based on DenseNet-121 with Swin Transformer in Smart Cities. In Proceedings of the 2024 1st International Conference on Cyber Security and Computing (CyberComp), Melaka, Malaysia, 6–7 November 2024.
16. Carrera García, J.M.; Recas Piorno, J.; Guijarro Mata-García, M. Expert system design for vacant parking space location using automatic learning and artificial vision. *Multimed. Tools Appl.* **2022**, *81*, 38661–38683. [CrossRef] [PubMed]
17. Nguyen, T.T.; Sartipi, M. Smart Camera Parking System With Auto Parking Spot Detection. In *Computer Vision—ACCV 2024 Workshops: 17th Asian Conference on Computer Vision*; Springer: Singapore, 2024; Volume 15482, pp. 237–246.
18. Wong, G.S.; Goh, K.O.M.; Tee, C.; Md. Sabri, A.Q. Review of Vision-Based Deep Learning Parking Slot Detection on Surround View Images. *Remote Sens.* **2023**, *23*, 6869. [CrossRef]
19. Hu, Z.; Xiao, H.; Zhou, Z.; Li, N. Detection of parking slots occupation by temporal difference of inverse perspective mapping from vehicle-borne monocular camera. *Automob. Eng.* **2021**, *235*, 3119–3126. [CrossRef]
20. Xiao, A.; Doshi, D.; Wang, L.; Gorantla, H.; Heitzmann, T.; Groth, P. Parking Spot Classification based on Surround View Camera System. *Appl. Mach. Learn.* **2023**, *12675*, 147–154.
21. Yuldashev, Y.; Mukhiddinov, M.; Abdusalomov, A.B.; Nasimov, R.; Cho, J. Parking Lot Occupancy Detection with Improved MobileNetV3. *Sensors* **2023**, *23*, 7642. [CrossRef]
22. Ma, Y.; Liu, Y.; Shao, S.; Zhao, J.; Tang, J. Review of Research on Vision-Based Parking Space Detection Method. *Int. J. Web Serv. Res.* **2022**, *19*, 1–25. [CrossRef]
23. Hassoune, K.; Dachry, W.; Moutaouakkil, F.; Medromi, H. Smart parking systems: A survey. In Proceedings of the 2016 11th International Conference on Intelligent Systems: Theories and Applications, Mohammedia, Morocco, 19–20 October 2016.
24. Biyik, C.; Allam, Z.; Pieri, G.; Moroni, D.; O'Fraifer, M.; O'Connell, E.; Olariu, S.; Khalid, M. Smart parking systems: Reviewing the literature, architecture and ways forward. *Smart Cities* **2021**, *4*, 623–642. [CrossRef]
25. Alam, M.R.; Saha, S.; Bostami, M.B.; Islam, M.S.; Aadeeb, M.S.; Islam, A.K.M.M. A survey on IoT driven smart parking management system: Approaches, limitations and future research agenda. *IEEE Access* **2023**, *11*, 119523–119543. [CrossRef]
26. Popereshnyak, S.V.; Chornobryvets, D.V. Computer vision-based approach to parking space availability detection. *Syst. Technol.* **2025**, *69*, 83–91. [CrossRef]

27. de Almeida, P.R.L.; Alves, J.H.; Oliveira, L.S.; Hochuli, A.G.; Fröhlich, J.V.; Krauel, R.A. Vehicle occurrence-based parking space detection. In Proceedings of the 2023 IEEE International Conference on Systems, Man, and Cybernetics, Honolulu, HI, USA, 1–4 October 2023.
28. Liu, B.; Lai, H.; Kan, S.; Chan, C. Camera-Based Smart Parking System Using Perspective Transformation. *Smart Cities* **2023**, *6*, 1167–1184. [CrossRef]
29. Ting, C.P.; Muniandy, M.; Batumalai, C. Smart Parking Mobile App System Allocation and Navigation. *J. Eng. Sci. Technol.* **2024**, *19*, 40–46.