

BEVFastLine: Single Shot Fast BEV Line Detection for Automated Parking Applications

Praveen Narasappareddygar¹^a, Venkatesh Moorthi Karunamoorthy¹^b, Shubham Sonarghare²^c,
Ganesh Sistu²^d and Prasad Deshpande²^e

¹Valeo India Pvt. Ltd., Chennai, India

²Valeo Vision Systems, Tuam, Ireland
{firstname.lastname}@valeo.com

Keywords: Road Marking Detection, Autonomous Parking, Multi Camera Line Detection, Birds Eye View.

Abstract: In autonomous parking scenarios, accurate near-field environmental perception is crucial for smooth operations. Parking line detection, unlike the well-understood lane detection, poses unique challenges due to its lack of spatial consistency in orientation, location, and varied appearances in color, pattern, and background surfaces. Consequently, state-of-the-art models for lane detection, which rely on anchors and offsets, are not directly applicable. This paper introduces BEVFastLine, a novel end-to-end line marking detection architecture in Birds Eye View (BEV) space, designed for 360° multi-camera perception applications. BEVFastLine integrates our single-shot line detection methodology with advanced Inverse Perspective Mapping (IPM) techniques, notably our fast splatting technique, to efficiently detect line markings in varied spatial contexts. This approach is suitable for real-time hardware in Level-3 automated vehicles. BEVFastLine accurately localizes parking lines in BEV space with up to 10 cm precision. Our methods, including the 4X faster Fast Splat and single-shot detection, surpass LSS and OFT in accuracy, achieving 80.1% precision, 90% recall, and nearly doubling the performance of BEV-based segmentation and polyline models. This streamlined solution is highly effective in complex, dynamic parking environments, offering high precision localization within 10 meters around the ego vehicle.


1 INTRODUCTION


In the domain of autonomous vehicles, the emergence of autonomous parking systems marks a significant stride towards enhanced efficiency, safety, and user convenience. Unlike other driving scenarios such as highways and urban roads, parking areas present complex challenges due to their confined nature and the presence of various static and dynamic obstacles. Central to navigating these challenges are the perception task of line detection, which is critical for accurate vehicle positioning and safe maneuvering within parking environments.


In the spectrum of perception tasks, two pivotal tasks—line detection and lane identification—often become conflated. Line detection, regarded as a


lower-level task, is principally devoted to discerning physical markings within the drivable terrain, employing either geometric or learning-based methodologies. Traditionally, geometric modeling algorithms like the Hough Transform (Hough, 1962) and Canny edge (Canny, 1986) detection have been utilized to pinpoint pixels in camera imagery corresponding to road lines, culminating in outputs such as binary images or geometric parameters encapsulating these lines. However, a contemporary shift is evident towards adopting learning-based algorithms, marking a progressive trend in this domain (Zakaria et al., 2023). Contrarily, lane identification, a higher-tier task, endeavors to decipher the semantic implication of the detected lines. This includes distinguishing which lines demarcate the current driving lane, adjacent lanes, or other classifications of lanes such as oncoming lanes or bike lanes.


Line landmarks, exhibited as various line types on the parking terrain, serve as essential navigational aids, guiding vehicles into designated spaces. Their role is paramount in demarcating drivable zones,

^a <https://orcid.org/0009-0000-2009-2182>

^b <https://orcid.org/0009-0007-2445-5754>

^c <https://orcid.org/0009-0008-4921-1526>

^d <https://orcid.org/0009-0003-1683-9257>

^e <https://orcid.org/0009-0008-9954-680X>

delineating parking slot boundaries, and ensuring safe vehicular maneuvering without collisions or encroachments on neighboring spaces (Lee and Park, 2021). Nonetheless, the detection of these line landmarks presents a challenge due to their elongated form and diverse visual appearances, which often render them indiscernible from other similar patterns in the environment. Such ambiguity poses a challenge to perception algorithms and complicates the data annotation process, potentially resulting in discrepancies in the training data for deep learning models. In this work, the focus is on line identification for parking systems, given the critical role of precise navigation especially in densely populated parking environments. Accurate detection of spatial parameters is pivotal for optimal space utilization, enabling the vehicle to align itself accurately within a parking spot's bounds. This accuracy becomes particularly vital during complex parking maneuvers, like parallel parking, where navigating through tight spaces with minimal margin for error is imperative.

Traditional line detection techniques, such as those outlined in Wang et al. (2018) (Wang et al., 2018), typically involve a two-stage process. The first stage encompasses either semantic segmentation of lines or direct regression of polynomial line coefficients in pixel space. The second stage involves projecting these lines into Birds Eye View (BEV) space, often based on a flat-world assumption. However, this methodology harbors several limitations, including a lack of geographical awareness of road surface variations, such as ramps and slopes.

Furthermore, BEV-based lane detection models, which are predominantly designed for lanes with spatial consistency, cannot be directly applied to line detection in parking scenarios. This is due to their reliance on spatial anchors (Garnett et al., 2019) and offsets (Efrat et al., 2020), which are less effective for lines that exhibit a diverse spatial spread, as opposed to the relatively uniform lanes encountered in driving contexts. Figure 1 highlights the distribution of lanes in two prominent public datasets, KITTI and TuSimple (Wen et al., 2021), in pixel space, and contrasts it with the distribution of parking lines in our dataset within a 25-meter range around the ego vehicle, represented in yellow color. This stark contrast underscores the necessity for a novel approach in line detection specifically tailored to the complexities of parking environments. The heatmap of lines is presented in the Bird's Eye View (BEV) space, rather than in the pixel space. This approach is chosen because in fisheye images, lines tend to be distributed throughout, leading to clutter. Visualizing them in the BEV space, which offers a top-view perspective in

Euclidean space, simplifies the analysis and enhances clarity.

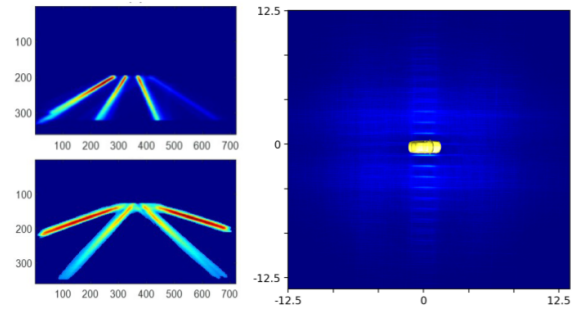


Figure 1: Left: Lanes Heatmap for KITTI and TuSimple datasets in Pixel Space. Right: Parking Lines Heatmap in our dataset in BEV Space with 25 meters around the ego vehicle.

Addressing the challenges associated with line detection necessitates the development of comprehensive and robust detection strategies. These strategies should account for the unique characteristics of line landmarks, requiring the design of context-aware algorithms capable of discerning subtle differences in visual features. Furthermore, the algorithms need to be efficient and real-time to accommodate the dynamic nature of parking environments and ensure timely responses for real-time hardware in Level-3 automated vehicles.

To this end, our proposed methods aim to address these challenges, and our primary contributions are summarized as follows:

- We introduce a novel single shot deep detector for parking lines, proficient in capturing the geometric intricacies of parking boundaries. This system inherently exhibits resilience to challenging conditions within parking environments.
- We introduced an alternative to the LSS method's cumsum trick for splatting the features to BEV space, utilizing one-hot sparse encoding and matrix multiplication for efficient points data processing, thereby reducing computational complexity and memory usage. The splatting technique works with any Inverse Perspective Mapping based BEV projection models and it was demonstrated with OFT BEV model as well.
- Our approach demonstrates its speed, efficiency, and precision via rigorous testing with the large scale surround view perception dataset for parking line detection.

2 RELATED WORK

We delve into prior studies that leverage surround-view cameras, examine techniques associated with processing surround-view images, and provide a concise overview of methods used for line landmark detection, as outlined below.

2.1 Surround-View Camera System

Owing to its large field-of-view (Kumar et al., 2023) of 190° , the surround-view camera system has gained significant popularity in the realm of near-field visual perception (Kumar et al., 2021). A near-field perception involves accurate localization within 10-15 meters around the ego vehicle. A typical such system (figure 2) in autonomous parking frameworks (Kumar et al., 2020; Yahiaoui et al., 2019), integrates four fisheye cameras positioned on the ego-vehicle. Such a configuration ensures a comprehensive capture of objects in close proximity, facilitating dependable visual perception. Our dataset, while derived from the comprehensive Woodscape collection established by (Yogamani et al., 2021), includes a subset of data not previously released, reserved for commercial use due to stringent data protection policies and privacy considerations in line with EU GDPR and Chinese data laws.

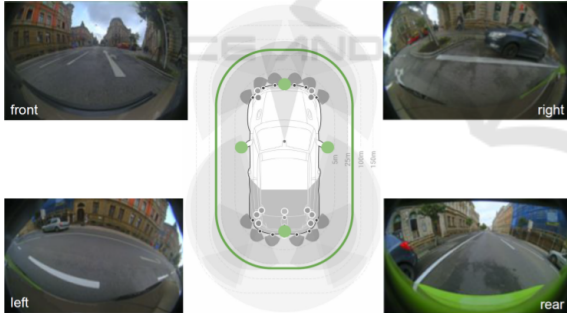


Figure 2: Data capture setup with 4 fisheye surround view cameras covering 360° around the vehicle.

2.2 Orthographic Feature Transform (OFT)

The Orthographic Feature Transform (OFT) method reprojects image-based features into an orthographic 3D space, enhancing spatial configuration understanding (Roddick et al., 2018). This technique, though primarily developed for 3D object detection, holds significant promise for applications like lines and curb detection due to its unique feature processing approach.

In the initial phase, a Convolutional Neural Network (CNN) serves as the front-end feature extractor, processing the input image to identify various features, mathematically represented as a feature map $f(u, v)$, where (u, v) coordinates correspond to the image plane. The OFT populates a 3D voxel feature map $g(x, y, z)$, with each voxel in g corresponding to a 3D space region and associated with a specific 2D image feature map area. The mapping from a voxel to the image plane is approximated by a bounding box, calculated based on intrinsic camera parameters and the voxel's 3D world coordinates. The relationship is expressed as:

$$u_1 = f \cdot \frac{(x - 0.5r)}{\left(z + 0.5 \frac{x}{|x|} r\right)} + c_u \quad (1)$$

$$v_1 = f \cdot \frac{(y - 0.5r)}{\left(z + 0.5 \frac{y}{|y|} r\right)} + c_v \quad (2)$$

$$u_2 = f \cdot \frac{(x + 0.5r)}{\left(z - 0.5 \frac{x}{|x|} r\right)} + c_u \quad (3)$$

$$v_2 = f \cdot \frac{(y + 0.5r)}{\left(z - 0.5 \frac{y}{|y|} r\right)} + c_v \quad (4)$$

where r represents the voxel size, and $[u_1, u_2]$ and $[v_1, v_2]$ define the corresponding voxel's bounding box in the image plane.

Features within the defined bounding box on the image are aggregated (typically by average pooling) to assign a feature vector to the voxel, represented as:

$$g(x, y, z) = \frac{1}{(u_2 - u_1)(v_2 - v_1)} \sum_{u=u_1}^{u_2} \sum_{v=v_1}^{v_2} f(u, v) \quad (5)$$

with the summation over $[u_1, u_2]$ and $[v_1, v_2]$.

The 3D voxel feature map g undergoes transformation into an orthographic feature map $h(x, z)$, collapsing the 3D map down to a 2D bird's-eye view representation, removing the original image's perspective effects. This process might involve summing features along the vertical axis or employing more complex transformation matrices, such as:

$$h(x, z) = \sum_{y=y_0}^{y_0+H} W(y)g(x, y, z) \quad (6)$$

where $W(y)$ are learned weight matrices, and the summation occurs over the voxel map's height y (Roddick et al., 2018).

In the context of lines detection, OFT's transformation of complex 3D information into a simplified

2D orthographic representation is invaluable. The orthographic view provides a consistent scale for features, crucial for detecting lines and curbs that appear distorted in perspective images. The OFT ensures well-represented features of lines and curbs through feature aggregation in the voxel and orthographic maps. This representation highlights the consistent characteristics of lines and curbs, simplifying detection. Furthermore, the orthographic map enhances spatial reasoning, as the relative positioning and orientation of lines and curbs are easily interpreted in orthographic space. This facilitation is particularly beneficial in complex scenarios like intersections or curved roads. Lastly, the orthographic feature map can integrate with further neural network stages specifically designed for lines and curbs detection, leading to more accurate and reliable detections in autonomous driving systems.

2.3 Lift, Splat, Shoot (LSS)

In the domain of autonomous vehicle perception, the “Lift, Splat, Shoot” (LSS) framework represents a significant advancement, addressing the challenge of integrating multi-camera data into a coherent bird’s-eye-view (BEV) representation (Phillon and Fidler, 2020). The LSS model adheres to crucial symmetries—translation equivariance, permutation invariance of camera inputs, and ego-frame isometry equivariance—facilitating a unified scene interpretation in the ego vehicle frame. This approach overcomes limitations of single-image detectors that cannot fully exploit the multi-viewpoint sensory information.

The LSS process begins with the “Lift” stage, where images are transformed into a shared 3D space without the reliance on depth sensors, generating a latent depth distribution for each pixel across a range of discrete depths. This results in a comprehensive point cloud that encapsulates the scene from all cameras in a frustum-shaped structure. The subsequent “Splat” stage involves an efficient pillar pooling technique that transforms the point cloud into a 2D BEV grid. This grid is then processed by a conventional CNN, enabling BEV semantic segmentation or motion planning. A key innovation in this stage is the use of a cumulative sum operation that optimizes the pooling step, significantly enhancing the model’s efficiency.

Finally, the “Shoot” stage of the LSS model is tailored for motion planning. By evaluating various trajectories on the inferred cost map, the model selects the trajectory with the minimum associated cost. This approach allows for end-to-end differentiability,

where the model learns to fuse camera inputs and optimizes for motion planning objectives concurrently. The LSS model’s capability to learn data-driven fusion across cameras and its backpropagation-friendly architecture enables a level of adaptability and learning potential not seen in previous models, which is empirically demonstrated to be effective in fusing information and aiding motion planning in autonomous driving tasks.

2.4 Line Landmarks Detection

Line segment detection has evolved significantly from traditional image gradient-based methods to more sophisticated learning-based techniques. Initially, methods relied on image gradients, with early approaches thresholding gradient magnitude to identify strong edges and aligned pixel sets. These techniques, while fast and accurate, often faltered in noisy or low-illumination conditions.

In contrast, recent learning-based methods have showcased remarkable advancements. Huang et al. (Huang et al., 2018) and (Huang et al., 2020) introduced end-to-end trainable CNNs, enhancing detection in man-made environments and developing the TP-LSD model for real-time detection. Xu et al. (Xu et al., 2021) leveraged Transformers for line segment detection, significantly refining the process. Lin et al. (Lin et al., 2020) merged classical techniques with deep learning, while Gao et al. (Gao et al., 2022) focused on camera pose refinement through point and line optimization. However, these developments have not directly addressed the unique challenges of parking line detection in BEV space using fisheye cameras.

The ULSD method, as proposed by Li et al. (Li et al., 2021), offers a significant advancement in line segment detection across various camera types. This end-to-end network leverages the Bézier curve model to achieve a model-free representation of line segments, which is particularly beneficial for images with distortion, such as those from fisheye or spherical cameras. The approach is independent of camera distortion parameters, simplifying the detection process for both distorted and undistorted images. Despite these advantages, the paper does not extensively address computational efficiency or real-time processing capabilities, which are critical for deployment in autonomous systems. Further, while the method performs well on the constructed datasets, its adaptability to a broader range of real-world scenarios remains to be thoroughly assessed.

An important aspect to note is that these methods operate in the pixel domain, and the line detections

must be projected into the metric (BEV) space for vehicle navigation planning. This projection poses a significant challenge as it typically assumes a flat ground surface, not accounting for varying road conditions like slopes or ramps. Such an assumption can lead to unacceptable localization errors, particularly in precise navigation applications such as parking in tight spaces. This underscores the need for advanced methodologies capable of accurately interpreting and adapting to the complex topography within the learning models. Though BEV perception using both IPM (Inverse Perspective Mapping) and Transformer-based methods has made strides in lane detection, these techniques often rely on anchors and offsets. This reliance limits their applicability to spatially consistent patterns like lanes, and less so to more diverse, spatially occurring categories such as parking lines or curbs.

3 METHODS

Our architecture, BEVFastLine, comprises three key stages: an image encoder, a fast splatting technique for feature projection into BEV space, and a BEV encoder followed by a Single Shot Line Detection network. (Figure 3).

3.1 LIFT Features

Images captured by four Surround View System (SVS) cameras undergo processing via an EfficientNet B0 encoder, which facilitates the extraction of embeddings. This approach aligns with the methodology employed in the Lift-Splat-Shoot (LSS) system (Phillion and Fidler, 2020). The images processed have a resolution of 1024x856 pixels. The EfficientNet-B0 encoder subsequently produces feature maps with a resolution of 64x54x512. These image features are further processed by a depth net convolutional network, which is designed to extract 17 depth features maintaining the same spatial resolution. Additionally, 64 context features are extracted at the same resolution. The depth features from each camera extend over a spatial range of 10 meters, effectively covering a 10-meter radius around the central vehicle.

3.2 FastSplat over Splatting

In the Lift, Splat, Shoot (LSS) approach, the "cumulative sum trick" is a central mechanism employed to efficiently splat the features from an entire sensor rig to BEV. This trick operates by sorting all points based

on their bin id (grid location in BEV Space), executing a cumulative sum over all features, and then subtracting the cumulative sum values at the boundaries of the bin sections. This method was designed to enhance memory efficiency, which would otherwise be used for padding during sum pooling across pillars (see Figure 4). The cumulative sum operation is central to this approach, effectively aggregating data for subsequent algorithmic steps. However, it is not without drawbacks. The necessity for sorting and the scatter operation for summing are both computationally intensive and not optimally compatible with neural accelerators, which typically expedite inference by distributing vector algebraic operations.

In contrast to the LSS framework, our FastSplat method, depicted in Figure 4, adopts a distinctive approach by forgoing the cumulative sum trick. Instead, it utilizes a one-hot sparse encoding technique for identifying the bin id of each point. In this method, each bin id is represented by a unique one-hot encoded vector, marked by a single '1' in its corresponding position, with all other elements set to '0'. This efficient encoding simplifies the differentiation between various bin ids.

The process then involves matrix multiplication of these one-hot encoded points with the camera features to compute the Bird's Eye View (BEV) features. FastSplat offers a 4X speed in splatting due to the two primary advantages:

- It obviates the need for a sorting operation, which typically has a computational complexity of $O(n \log n)$, where 'n' is the number of grid cells in BEV space.
- The method replaces the cumsum trick, which at its core is a scatter operation involving summation based on an index vector. This operation is not a standard vector algebraic process, making it unsuitable for neural engines on edge devices. By modeling the process as generalized matrix multiplication, FastSplat aligns well with the capabilities of neural accelerators, which are essentially matrix multiplication engines.

The Fast Split module receives its inputs from the Lift Module and Camera Frustums, with a significant modification in the approach to camera geometry. Unlike the method employed by Phillion et al. (2020) (Phillion and Fidler, 2020), which utilizes pinhole camera geometry for frustum creation, our approach adapts the camera frustums to fisheye geometry. This adaptation is achieved using a polynomial distortion camera model, a technique comprehensively described by Usenko et al. (2018).

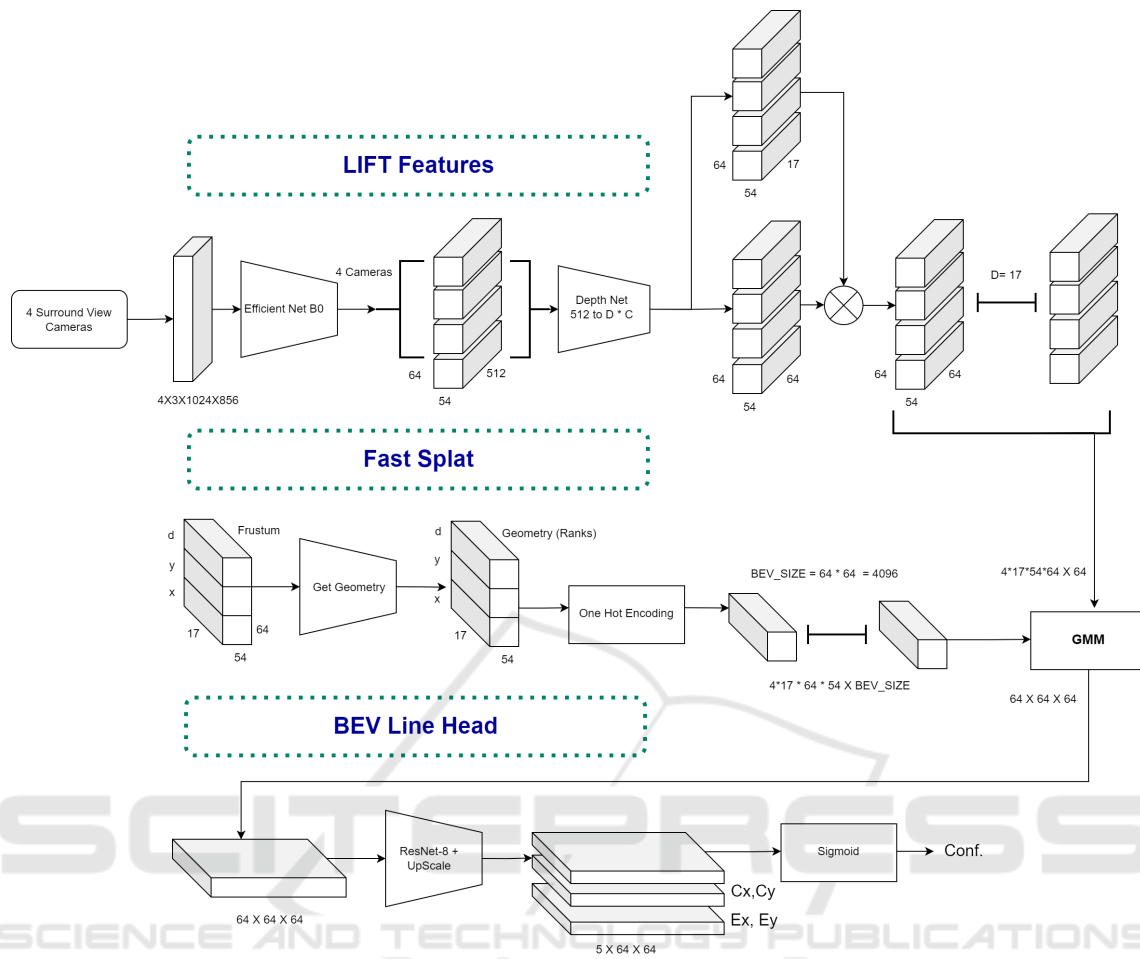


Figure 3: Illustration of the Proposed Method Featuring Fast Lifting and Single Shot Line Detection Head. The notation used in the method includes D , representing the Number of Depths; C , indicating the Number of Cameras; C_x and C_y , which denote the Center of the Line; and E_x and E_y , signifying the Endpoints of the Line in the X and Y directions, respectively.

3.3 Single Shot Line Head

Though methods like ULSD offer a solution for direct regression of a parametric line model, they are not ideally suited for integrating into the BEV architectures due to computational issues as outlined below:

- ULSD operates as a two-stage method. Initially, a line proposal network generates potential line candidates. These candidates are then used as control points for a bezier model to compute uniform points. Subsequently, the point indices are employed to extract feature vectors from the encoder block, which are then passed to a binary classifier for line detection.
- Additionally, the concept of junctions and centers in ULSD, and the associated learning process to match them and form line proposals, involve com-

plex operations. These operations are not typically supported by hardware accelerators on embedded systems, posing a challenge for real-time application.

Our proposed model, directly regressing lines in a grid fashion as shown in Figure 3, marks a significant shift in line detection methodologies. By regressing the line center and the lengths extending from it, and employing a classification node for line existence probability, our model aligns with the paradigm shift observed in object detection. Similar to how single-stage, anchor-free algorithms like (Zhou et al., 2019) and (Tian et al., 2020) have provided effective alternatives to the two-stage methods like (Ren et al., 2016), our single-stage line detection approach offers a comparable alternative in the line detection approach. This represents a notable advancement, mirroring the evolution seen in object detection towards more efficient and streamlined models.

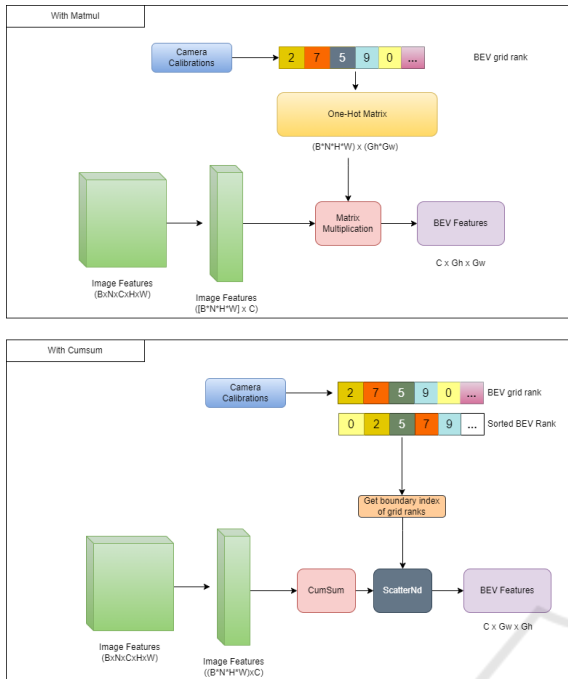


Figure 4: TOP: Fast Splating with Sprase Matrix Multiplication. Bottom: Point Pillars based Splating using Cumulative Sum.

4 EXPERIMENTS

This section presents our methodology and the technical details behind the evaluation of our parking line detection work. We discuss the setup used for training our models, the characteristics of the commercial subset of the dataset utilized, and the specific implementation strategies adopted. This section also details the post-processing techniques that refine the output and underscores the measures ensuring system reliability across varied scenarios.

4.1 Experimental Setup

The experimental setup for evaluating our models was conducted using high-performance computing resources. Specifically, the models were trained on an NVIDIA A100 GPU with 40 GB of memory, which is well-suited for demanding deep learning tasks. We chose a batch size of 4 to balance between computational efficiency and memory constraints.

For training times for the OFT and LSS approaches for an epoch are 1.6 and 1.5 hours respectively. The choice of the optimizer was AdamW, a variant of the Adam optimizer that incorporates a decoupled weight decay regularization, which often leads to better performance in deep learning models.

The learning rate was set to $1e-4$, providing a suitable compromise between convergence speed and stability of the training process. In the interest of focusing on the models' ability to learn from raw data, no augmentations were applied to either the image or BEV representations during training. This decision was made to assess the models' performance under unaltered data conditions, thus ensuring the evaluation of their fundamental feature extraction and generalization capabilities.

4.1.1 Dataset

Our dataset, curated for the development and assessment of our parking line detection system, comprises images from fisheye cameras that encapsulate a 360-degree view of parking environments. The image set incorporates views from Front View (FV), Rear View (RV), Mid-View Right (MVR), and Mid-View Left (MVL) cameras, each capturing at a resolution of 1024x856 pixels. The training subset consists of 12,574 timesteps, translating to 50,296 images across all camera perspectives. For evaluation, the dataset encompasses 3,529 timesteps, totaling 14,116 images. Here, a 'timestep' denotes a singular temporal instance with simultaneous captures from all cameras, reflecting various parking scenarios as shown in Figure 5.

The timestamps are carefully sourced from 602 traces out of which 473 for training and 129 for validation. The dataset offers sequences that showcase continuity in line features, crucial for temporal consistency and algorithmic robustness. Each sequence represents a complete trajectory of the ego vehicle from entering to exiting a parking area. Emphasizing geographic diversity, the dataset includes captures from the Czech Republic (CZE), Ireland (IRL), and the United States (USA), with 147, 416, and 45 traces, respectively. This diversity ensures the adaptability of our system to various parking layouts and conditions.

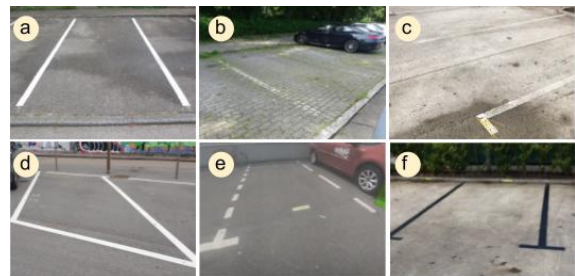


Figure 5: Parking Lines in Our dataset. a) Solid lines , b) Lines on cobble stones, c) Washed out lines, d) Angled lines, e) Dashed lines and f) Colored lines.

Data collection was performed using six different vehicles to affirm the model’s invariance to specific vehicle camera rig configurations. The environmental contexts covered include a spectrum of parking settings: city lots, indoor garages, urban parking structures, and rural parking areas, with respective instances of 5, 61, 491, and 51, thus providing extensive exposure to various parking situations.

The dataset further addresses a variety of lighting conditions critical for parking applications: artificial lighting for indoor settings (61 instances), low-light conditions found during dawn or dusk (6 instances), and bright daylight scenarios typical for outdoor parking (511 instances). Street lighting conditions, representing nighttime parking environments, are included in 30 instances. This extensive range of lighting conditions ensures the detection system is thoroughly tested and capable of reliable performance in real-world parking applications, ready for rigorous training and evaluation.

4.1.2 Implementation Details

Our system processes four fisheye images by passing them to an EfficientNet-lite B0 inspired image encoder. This encoder consists of five reduction blocks, reducing spatial dimensions and enhancing feature depth, with a skip connection between the latter stages to preserve context in the output features.

For the Orthographic Feature Transform (OFT) projection, an adaptation for fisheye cameras aligns with the SVS BEV grid using extrinsic parameters. The BEV encoder then refines these features through three reduction and two upsampling layers, culminating in a feature map tailored for line detection.

The lines decoder, with three branches, predicts junction points, center points, and line segmentation. It employs varied loss functions like binary cross entropy and weighted L1 and smooth L1 losses for different prediction types. Post-processing utilizes non-maximum suppression and max pooling on junction maps, followed by matching and correction steps to compute Lines of Interest (LOIs).

This streamlined process, emphasizing computational efficiency and modularity, ensures real-time applicability across diverse computational setups, maintaining precision without excessive computational load. In comparison to the standard semantic segmentation model (LSS), which requires 14.8 TOPS (Trillion Operations Per Second) for a 10 FPS (Frames Per Second) throughput and incurs a post-processing time of 4.8 seconds on a typical Intel Xeon PC equipped with an Nvidia A100 GPU, our proposed BEVFastLine model demonstrates a similar computational demand at 14.81 TOPS. However, it significantly re-

duces post-processing time to just 0.4 seconds. This reduction in processing time translates into decreased CPU cycle requirements and also enhancing overall performance efficiency as detailed in Table 1.

The post-processing phase is only needed for the baseline models (LSS & OFT) to convert the line segmentation output to poly-line representation.

The post-processing phase is pivotal in transforming the segmentation masks produced by the decoder into vectorized line representations. Following binary thresholding to enhance the contrast between line features and the background, the masks are skeletonized to single-pixel thickness, preserving the essential structure of the lines.

Subsequent to skeletonization, any intersecting lines are segmented at junctions to facilitate individual analysis. Connected Component Analysis (CCA) is then utilized to identify and isolate these distinct line segments. At this juncture, we employ Bézier curve fitting to model the geometry of each line segment accurately.

Bézier curves offer a parametric approach to defining curves and are particularly advantageous for their scalability and the ease with which they can model complex shapes. For relatively straight paths or gentle curves, quadratic Bézier curves are fitted using the endpoints and a calculated control point from the segmented lines. For more complex or sharply curving paths, cubic Bézier curves are utilized, requiring two control points in addition to the endpoints to accurately trace the line’s trajectory.

By fitting Bézier curves to the segmented lines, our system generates a smooth and continuous representation of road markings. These vectorized lines are mathematically defined and easily manipulated, which is essential for applications that rely on precise pathing, such as autonomous vehicle navigation systems and advanced mapping services.

The fitting of Bézier curves marks the final step in the line detection process, concluding with the extraction of endpoints for each curve. This comprehensive post-processing strategy ensures that our system’s outputs are not only accurate in terms of spatial positioning but also robust against the variability of real-world driving environments.

5 PERFORMANCE EVALUATION

The effectiveness of our line landmark detection system is assessed quantitatively through precision and recall metrics, while also incorporating a spatial validation method to determine the veracity of detected line endpoints.

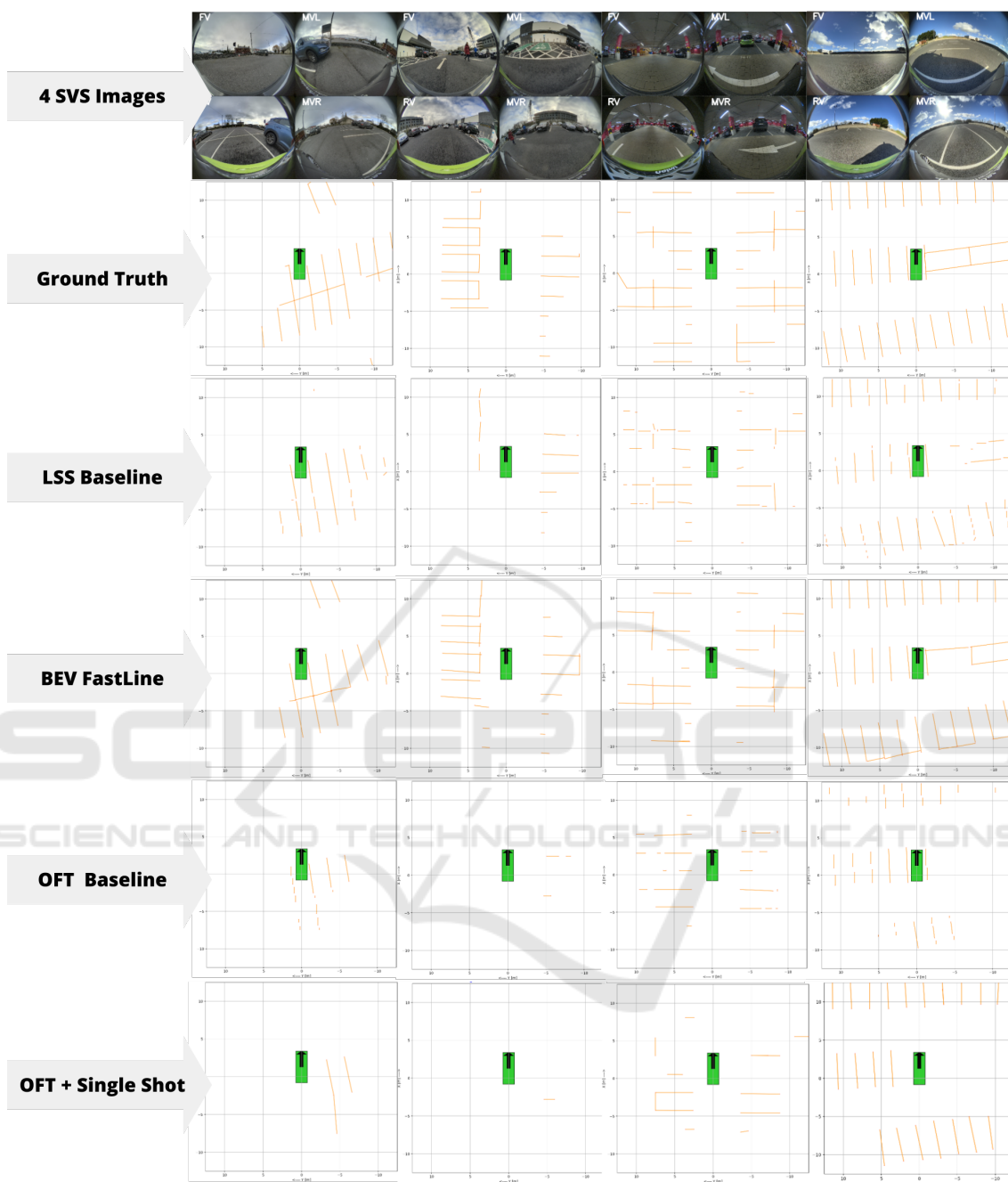


Figure 6: Qualitative Results on four samples from the test set.

5.1 Evaluation Criteria

True positive detections are validated if the predicted line endpoint falls within a circle whose radius is at certain percent of the distance (Circular Radius in 1 from the ego vehicle to the predicted endpoint). This radius scales with distance to account for the relative position of line features, recognizing that distant lines allow for a larger margin of error due to perspective

effects (see Figure 7).

False positives are identified when predicted endpoints do not coincide with any ground truth line within the scaled radius, indicating a detection where no line feature exists. These criteria ensure that our evaluation accounts for both the accuracy of line placement and the needed scaling for positional errors.

Table 1: Comparative Performance Metrics for Line Detection Approaches.

Experiment		Baseline	Proposed	Baseline	Proposed
Metrics	Circular Radius	LSS + Lines Derived	BEV FastLine (LSS)	OFT + Lines Derived	OFT + Single Shot
Precision	20%	45	80.1	28.8	80.2
Recall		46.2	90	12	25.6
Precision	25%	51.1	81.3	33.7	82.4
Recall		52.5	71.1	13.8	26
Precision	30%	55.3	82.3	40.5	83.6
Recall		56.8	71.9	16.7	26.6
Precision	35%	59.6	83.4	47.3	84.8
Recall		61.2	72.6	19.6	27.2

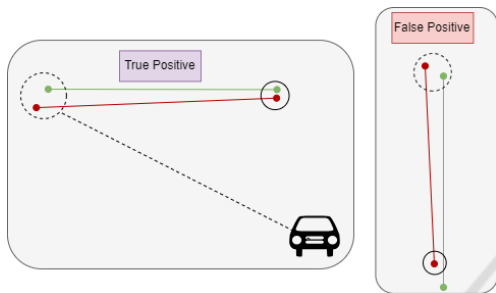


Figure 7: Illustration of line detection evaluation criteria: On the left, a True Positive (TP) where a predicted line’s endpoints are within a set radius. On the right, a False Positive (FP) where any endpoint is outside this radius.

5.2 Precision

Precision is quantified by the ratio of true positive detections to the total number of positive predictions made by the model. The formula is as follows:

$$P = \frac{TP}{TP + FP} \quad (7)$$

Here, TP represents true positives, corroborated by the spatial validation method, and FP indicates false positives, where predicted endpoints fall outside the defined evaluation radius from any ground truth line. High precision reflects the model’s ability to accurately detect lines without triggering excessive false alarms, crucial for reliable autonomous navigation.

5.3 Recall

Recall measures the model’s capacity to detect all pertinent line features present in the data:

$$R = \frac{TP}{TP + FN} \quad (8)$$

FN denotes false negatives, where the model fails to identify an actual line within the evaluation radius. High recall signifies comprehensive detection coverage, vital for the completeness of navigational cues.

This evaluation methodology, encompassing both precision and recall augmented with spatial validation, provides a robust measure of our system’s performance, ensuring that detected lines are precise and consistent with real-world locations relative to the ego vehicle.

6 DISCUSSION

The comparative analysis of line detection methods, as illustrated in Table 1, reveals a promising direction for the proposed BEV FastLine and OFT + SingleShot approaches. Both methods demonstrate significant precision rates, with BEV FastLine achieving 80.1% and OFT + SingleShot reaching 80.2%, which is approximately 2X higher than their respective baselines of 45% and 28.8%. This high precision suggests a reduced rate of false positives, a crucial factor in autonomous navigation where inaccurate line detection could lead to misguidance or safety hazards.

Table 1 underscores the overall 2X improvement (averaged over all metrics) achieved by single-stage direct regression of lines as polylines in BEV space. This approach contrasts with methods that perform segmentation followed by line fitting as a post-processing operation. These improvements are consistent across both architecture styles (LSS, OFT). Figure 6 demonstrates the limitations of segmentation models in fully regressing line structures, even with extensive post-processing.

When considering recall, the BEV FastLine method demonstrates a superior ability to detect relevant line features with a recall of 90%, compared to the OFT + SingleShot method which stands at 25.6%. The higher recall of the BEVFastLine method is due to the fact that LSS itself is superior to OFT and it is more effective in identifying the presence of lines, a quality that could prove crucial in complex driving scenarios where every line marking is significant for vehicle guidance and decision-making processes.

The results indicate that while OFT provides a framework for accurate line prediction when it detects them, its overall line feature detection rate needs enhancement. On the other hand, the BEV FastLine demonstrates an improved balance between detecting lines and maintaining a lower false positive rate.

These observations warrant further investigation into the underlying factors contributing to the performance disparities. It is conceivable that the intrinsic characteristics of the LSS method, such as its focus on learning from a comprehensive point cloud, may afford it a broader detection capability. In contrast, the OFT method's dependence on transforming image features to an orthographic view might limit its sensitivity to certain types of line markings or variances in environmental conditions.

7 CONCLUSIONS

In conclusion, our proposed methodologies introduced in this study provide a significant advancement in line landmark detection for autonomous parking systems. The empirical results underscore the precision of both methods, with the BEV FastLine approach demonstrating a commendable balance between precision and recall. This balance is crucial for real-world applications where accurate line detection is instrumental in safe and reliable vehicle navigation.

The OFT + SingleShot method, also superior in precision, when compared to baseline OFT based segmentation model. The current work lays the foundation for future enhancements in detection rates and suggests that a hybrid approach may yield a more optimal solution. Such improvements are vital for navigating complex environments and ensuring comprehensive line detection coverage.

The Fast Splatting technique introduced in our work requires 4X less computational time when compared to started cumsum operation and also suitable for neural engines on the embedded systems.

Finally, the work delineated in this paper significantly enriches the evolving domain of autonomous vehicle technologies. By highlighting the strengths and areas for development in line landmark detection, it steers future efforts towards creating more sophisticated and robust systems. These systems will be essential in realizing the full potential of autonomous vehicles, ensuring safety, efficiency, and reliability in automated parking and beyond.

ACKNOWLEDGEMENTS

The authors thank Valeo Vision System for their support, resources, and the opportunity to contribute to the broader research community with this work.

REFERENCES

- Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698.
- Efrat, N., Bluvstein, M., Oron, S., Levi, D., Garnett, N., and Shlomo, B. E. (2020). 3d-lanenet+: Anchor free lane detection using a semi-local representation.
- Gao, S., Wan, J., Ping, Y., Zhang, X., Dong, S., Yang, Y., Ning, H., Li, J., and Guo, Y. (2022). Pose refinement with joint optimization of visual points and lines. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2888–2894. IEEE.
- Garnett, N., Cohen, R., Pe'er, T., Lahav, R., and Levi, D. (2019). 3d-lanenet: End-to-end 3d multiple lane detection.
- Hough, P. V. (1962). Method and means for recognizing complex patterns. US Patent 3,069,654.
- Huang, K., Wang, Y., Zhou, Z., Ding, T., Gao, S., and Ma, Y. (2018). Learning to parse wireframes in images of man-made environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 626–635.
- Huang, S., Qin, F., Xiong, P., Ding, N., He, Y., and Liu, X. (2020). Tp-lsd: Tri-points based line segment detector. In *European Conference on Computer Vision*, pages 770–785. Springer.
- Kumar, V. R., Eising, C., Witt, C., and Yogamani, S. (2023). Surround-view fisheye camera perception for automated driving: Overview, survey & challenges. *IEEE Transactions on Intelligent Transportation Systems*.
- Kumar, V. R., Hiremath, S. A., Bach, M., Milz, S., Witt, C., Pinard, C., Yogamani, S., and Mäder, P. (2020). Fisheyedistancenet: Self-supervised scale-aware distance estimation using monocular fisheye camera for autonomous driving. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 574–581. IEEE.
- Kumar, V. R., Yogamani, S., Rashed, H., Sitsu, G., Witt, C., Leang, I., Milz, S., and Mäder, P. (2021). Omnidet: Surround view cameras based multi-task visual perception network for autonomous driving. *IEEE Robotics and Automation Letters*, 6(2):2830–2837.
- Lee, Y. and Park, M. (2021). Around-view-monitoring-based automatic parking system using parking line detection. *Applied Sciences*, 11(24):11905.
- Li, H., Yu, H., Wang, J., Yang, W., Yu, L., and Scherer, S. (2021). Ulsd: Unified line segment detection across pinhole, fisheye, and spherical cameras. *ISPRS Journal of Photogrammetry and Remote Sensing*, 178:187–202.

- Lin, Y., Pinteá, S. L., and van Gemert, J. C. (2020). Deep hough-transform line priors. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 323–340. Springer.
- Phillon, J. and Fidler, S. (2020). Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 194–210. Springer.
- Ren, S., He, K., Girshick, R., and Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks.
- Roddick, T., Kendall, A., and Cipolla, R. (2018). Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*.
- Tian, Z., Shen, C., Chen, H., and He, T. (2020). Fcos: A simple and strong anchor-free object detector.
- Wang, Z., Ren, W., and Qiu, Q. (2018). Lanenet: Real-time lane detection networks for autonomous driving. *arXiv preprint arXiv:1807.01726*.
- Wen, T., Yang, D., Jiang, K., Yu, C., Lin, J., Wijaya, B., and Jiao, X. (2021). Bridging the gap of lane detection performance between different datasets: Unified viewpoint transformation. *IEEE Transactions on Intelligent Transportation Systems*, 22(10):6198–6207.
- Xu, Y., Xu, W., Cheung, D., and Tu, Z. (2021). Line segment detection using transformers without edges. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4257–4266.
- Yahiaoui, M., Rashed, H., Mariotti, L., Sistu, G., Clancy, I., Yahiaoui, L., Kumar, V. R., and Yogamani, S. (2019). Fisheyemodnet: Moving object detection on surround-view cameras for autonomous driving. *arXiv preprint arXiv:1908.11789*.
- Yogamani, S., Hughes, C., Horgan, J., Sistu, G., Varley, P., O’Dea, D., Uricar, M., Milz, S., Simon, M., Amende, K., Witt, C., Rashed, H., Chennupati, S., Nayak, S., Mansoor, S., Perroton, X., and Perez, P. (2021). Woodscape: A multi-task, multi-camera fish-eye dataset for autonomous driving.
- Zakaria, N. J., Shapiai, M. I., Ghani, R. A., Yasin, M., Ibrahim, M. Z., and Wahid, N. (2023). Lane detection in autonomous vehicles: A systematic review. *IEEE Access*.
- Zhou, X., Wang, D., and Krähenbühl, P. (2019). Objects as points.