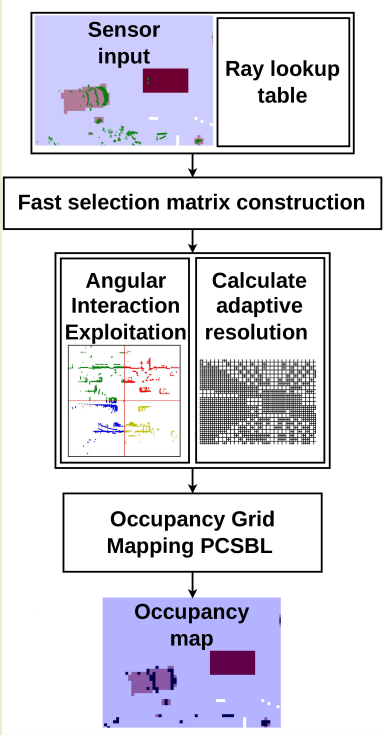
Accelerated Pattern-Coupled Sparse Bayesian Learning for Automotive Occupancy Mapping

Frank Harraway, Peiyuan Zhai, *Graduate Student Member, IEEE*, Geethu Joseph, *Senior Member, IEEE*, and Ashish Pandharipande, *Senior Member, IEEE*

***Abstract*—An occupancy grid map (OGM) is used in automotive driving applications to model the vehicle surroundings using data from sensors on vehicles like light detection and ranging (LiDAR), radar, or their fusion. In state-of-the-art work, pattern-coupled sparse Bayesian learning (PCSBL) was used to estimate the OGM by leveraging spatial dependencies in the map when either a single sensor modality was used or when fusion of multiple sensor modalities was employed. The PCSBL method, however, has high computational complexity, making real-time implementation challenging for large-sized grid maps. To address this limitation, we propose several methods to improve the computational efficiency of PCSBL while maintaining mapping accuracy. First, we utilize a precomputed lookup table to accelerate selection matrix construction. Second, we implement adaptive resolution reduction based on sensor measurements, lowering problem dimensionality where coarse resolution suffices. Third, we develop two novel methods that exploit the narrow angular interactions between measurements and the map regions to enhance computational efficiency. The first method partitions measurements into spatially disjoint submaps that enable parallel processing. The second method exploits the angular structure to impose a block structure on the selection matrix, reducing the computational overhead of matrix operations. Experiments on the nuScenes and RADIATE public datasets show that the presented methods reduce runtime by at least an order of magnitude compared with the benchmark PCSBL and fusion-based PCSBL methods while preserving detection accuracy.**



***Index Terms*— Adaptive resolution, light detection and ranging (LiDAR) point clouds, nuScenes dataset, occupancy grid mapping (OGM), radar point clouds, RADIATE dataset, sparse Bayesian learning.**

I. Introduction

AN OCCUPANCY grid map (OGM) is a discrete representation of the environment around a vehicle, with each grid cell of the map containing occupancy values. An OGM is built by processing data from sensors like light detection and ranging (LiDAR) mounted on the ego vehicle and used to support perception tasks like free-space detection, object tracking, and path planning in autonomous driving and advanced driver assistance systems [1], [2], [3].

A widely used model-based approach in OGMs is the inverse sensor model (ISM) [1], which assumes that each cell in the occupancy map is independent and uses ray casting principles to inform occupancy states. This independence assumption enables fast processing, with ray casting being the most computationally intensive component of the algorithm. Various techniques have been developed to further accelerate ray casting. Hierarchical structures like octrees and kd-trees reduce the number of cells traversed during computation [4], [5], [6], while approximate table-driven

Received 19 August 2025; accepted 23 September 2025. Date of publication 8 October 2025; date of current version 14 November 2025. This work was supported by the Rijksdienst voor Ondernemend Nederland under Grant PPS2207. An earlier version of this paper will be presented in part at the IEEE Sensors Conference, October 2025, Vancouver, Canada. The associate editor coordinating the review of this article and approving it for publication was Prof. Pierluigi Salvo Rossi. *(Corresponding author: Geethu Joseph.)*

Frank Harraway, Peiyuan Zhai, and Geethu Joseph are with the Signal Processing Systems Group, Delft University of Technology, 2628 Delft, The Netherlands (e-mail: fharraway@tudelft.nl; p.zhai@tudelft.nl; g.joseph@tudelft.nl).

Ashish Pandharipande is with NXP Semiconductors, 5656 Eindhoven, The Netherlands (e-mail: ashish.pandharipande@nxp.com).

Digital Object Identifier 10.1109/JSEN.2025.3617133

methods from localization applications provide constant-time distance estimates at the expense of quantization error [7]. However, ISM's independence assumption, while computationally efficient, suffers from conflicts caused by inconsistent measurements and fails to exploit spatial structure in OGMs.

An alternative approach employs kernel-based methods, such as Gaussian process OGMs [8], which capture spatial dependencies. Early implementations were computationally intensive. Subsequent work in [9] addressed this by introducing test-data octrees, among other optimizations, to reduce computational load. Test-data octrees differ from traditional octrees (3-D) and quadtrees (2-D) used in OGMs for storage compression [10], [11], [12], [13]—they dynamically decrease resolution as a preprocessing step based on sensor measurements, thereby reducing computational load. Building on these computational improvements, the Bayesian generalized kernel (BGK) [14], [15] method further accelerates Gaussian process OGMs by incorporating optimizations from [9] and [16]. Despite these advances, Gaussian process OGMs still do not effectively handle the inherent sparsity in occupancy maps. A sparsity-aware OGM model was introduced in [3], where pattern-coupled sparse Bayesian learning (PCSBL) [17] was applied to estimate block-sparse occupancy maps by exploiting spatial structure using a single sensor modality. This work was extended to multimodal fusion in [18]. While single-modality PCSBL for OGM outperforms Gaussian process OGMs and ISM in accuracy, its high computational complexity limits its application to large OGMs, with the multimodal fusion version being even more computationally expensive. Therefore, this work aims to reduce the computational complexity of the PCSBL methods for OGM.

Furthermore, deep learning models have also been investigated in the literature. By leveraging large-scale autonomous driving sensor datasets [19], [20], [21], deep learning models can be trained to infer OGMs with classification or semantic information through learnable networks. These networks can be designed to use either single-modality [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32] or multimodal sensor input. The computational complexity is then dependent on the depth, layer sizes, and types of layers. However, they typically require large annotated datasets, face challenges in generalization, and often necessitate retraining when datasets or sensor types change. Moreover, they frequently struggle with generating occupancy maps of varying sizes and resolutions, estimating confidence in the maps, and handling sensor unreliability. Some studies have partially addressed these issues. For example, multiscale maps are generated using coarse-to-fine query structures [33], [34], [35], though they still require full-size feature computation. Thus, we focus on model-based methods that do not need annotated data, aiming to improve the computational efficiency of state-of-the-art PCSBL.

In this work, we reduce the computational complexity of PCSBL [3] for OGMs for LiDAR, radar, and the fusion of both modalities [18]. Our main contributions are as follows.

1) *Acceleration Techniques:* We present three acceleration techniques for PCSBL:
   a) *Ray Lookup Table:* We reduce the computation required to construct the selection matrix. Using K–d trees, precomputed ray traversals are accessed using a nearest neighbor search, which are then used to efficiently compute the sparse selection matrix.
   b) *Test-Data Quadtrees:* We extend the use of test-data quadtrees from [9] to PCSBL. These quadtrees provide a scene-dependent adaptive resolution, reducing the number of unknowns that need to be solved. To handle the resulting inconsistent cell sizes, we modify PCSBL to ensure compatibility.
   c) *Measurement Model Exploitation:* We leverage the unique structure of the model, where each measurement influences only a narrow angular map region. We introduce two methods: the partition and overlap (PO) method, which splits measurements into submaps for parallel processing, and the region-based cell permutation (CP) method, which reorganizes the measurement matrix into a block structure for faster computations.
2) *Evaluation on Real-World Datasets:* We evaluate our acceleration techniques on the nuScenes and RADIATE public datasets [20], [36] by comparing them with benchmark PCSBL [3], [18], BGK [14], and ISM [37]. For single-modality PCSBL, run time is reduced from 12.9 to 0.47 s (27× speedup), while multimodal fusion was accelerated from 14.6 to 0.7 s (21× speedup), with no reduction in map accuracy.

This work extends our conference paper [38] in several directions. We introduce two new acceleration techniques, a ray lookup table that accelerates selection matrix construction and an adaptive quadtree resolution technique from [9] to the PCSBL algorithm. We also extend the measurement model exploitation techniques for PCSBL in [38] from LiDAR to radar and multimodal fusion PCSBL [18]. To evaluate these radar and fusion extensions, we expand our evaluation to include the RADIATE dataset in addition to nuScenes. Finally, we present more results for the radar and multimodal fusion PCSBL techniques and present time complexity analysis for varying occupancy grid resolutions.

II. PCSBL Preliminaries and Complexity Analysis

This section introduces the state-of-the-art PCSBL algorithm for OGM in [3], the extension to PCSBL fusion of multiple modalities in [18], and the computational challenges of the PCSBL methods.

A. Single-Modality OGM PCSBL

We consider the problem of occupancy grid mapping that aims to estimate the occupancy state of a vehicle's surrounding environment from M LiDAR or radar sensor measurements (reflection points). To this end, the environment is discretized into a 2-D grid with N cells, each modeled as a binary random

variable. The collection of all cell occupancies is represented by the vector $\boldsymbol{x} \in \{0,1\}^N$.

1) Sensor Measurement Model: PCSBL for OGMs adopts a linear sensor measurement model in [3], where each measurement contributes free-space and occupied information. For LiDAR measurements, we trace a sensor ray to identify an occupied cell at each reflection point and mark preceding cells along the sensor ray as free. For radar measurements, due to the increased sparsity of measurements, an angular sector up to the reflection point is considered free, while cells within the sector boundary are considered occupied. For both sensors, these occupancy states are encoded in the selection matrix $\boldsymbol{A}$ and measurement vector $\boldsymbol{y}$. Each reflection point contributes one row for occupied cells and one row for free cells in the selection matrix, and a 1 for occupied and 0 for free cells in the measurement vector. Despite these sensor-specific differences, the reconstruction problem can be expressed in a unified linear form for single-modality maps from M measurements

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{n} \tag{1}$$

where the measurements are encoded into the selection matrix $\boldsymbol{A} \in \{0,1\}^{2M \times N}$ and the measurement vector $\boldsymbol{y} \in \{0,1\}^{2M}$. The additive noise term $\boldsymbol{n}$ is modeled as Gaussian with $\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$, where σ^2 is the unknown noise variance. PCSBL aims to reconstruct the unknown map $\boldsymbol{x}$ from $\boldsymbol{y}$ and $\boldsymbol{A}$.

2) Single-Modality PCSBL: The PCSBL algorithm for a single modality (LiDAR or radar) imposes a two-layer hierarchical prior distribution on the unknown $\boldsymbol{x}$. In the first layer, $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{D}^{-1})$ is modeled as a zero-mean Gaussian vector with a diagonal precision matrix $\boldsymbol{D}$. To exploit the block structure, its nth diagonal element is

$$D[n,n] = \alpha[n] + \beta \sum_{m \in \mathcal{L}_n} \alpha[m] \tag{2}$$

where $\alpha[n]$ is the nth cell precision hyperparameter, $\beta \in [0,1]$ is the coupling parameter, and $\mathcal{L}_n$ is the neighbor set of the nth cell. When the precision $\alpha[n]$ of the nth cell is high, the occupancy value $x[n]$ is likely to be close to zero, indicating it is a free cell. Since the precisions are coupled via β, they are spatially correlated, encouraging adjacent cells to agree on the occupancy state. The second layer assigns Gamma hyperpriors

$$\alpha[n] \sim \text{Gamma}(a,b) \quad \text{and} \quad \sigma^{-2} \sim \text{Gamma}(c,d). \tag{3}$$

To estimate $\boldsymbol{x}$, the unknown hyperparameters $\boldsymbol{\alpha}$ and σ^2 are estimated iteratively through the expectation–maximization algorithm [3], [17]. In the tth iteration, given the current hyperparameters $\boldsymbol{\alpha}^{(t)}$ and $(\sigma^{-2})^{(t)}$, the expectation (E) step computes the posterior mean $\hat{\boldsymbol{\mu}}$ and covariance $\hat{\boldsymbol{\Phi}}$ of $\boldsymbol{x}$ as

$$\hat{\boldsymbol{\Phi}} = \left(\left(\sigma^{-2}\right)^{(t)} \boldsymbol{A}^\top \boldsymbol{A} + \boldsymbol{D} \right)^{-1}, \quad \text{and} \quad \hat{\boldsymbol{\mu}} = \left(\sigma^{-2}\right)^{(t)} \hat{\boldsymbol{\Phi}}\, \boldsymbol{A}^\top \boldsymbol{y} \tag{4}$$

where $\boldsymbol{D}$ is computed via (2) with $\boldsymbol{\alpha} = \boldsymbol{\alpha}^{(t)}$. Then, the maximization (M) step updates the hyperparameters to maximize the model likelihood given the measurements

$$\alpha^{(t+1)}[n] = \frac{a}{0.5\,\hat{v}^{(t)}[n] + \beta \sum_{m \in \mathcal{L}_n} \hat{v}^{(t)}[m] + b} \tag{5}$$

$$\left(\sigma^2\right)^{(t+1)} = \frac{1}{M + 2c} \left(\left\| \boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{\mu}}^{(t)} \right\|_2^2 + \left(\sigma^{-2}\right)^{(t)} \sum_n \hat{\Phi}^{(t)}[n,n]\; D[n,n] + 2d \right) \tag{6}$$

where $\hat{v}^{(t)}[m] = \hat{\Phi}[m,m] + \hat{\mu}^2[m]$. Iterations continue until convergence; the final map binary OGM $\hat{\boldsymbol{x}}$ is obtained by thresholding the real-valued vector $\hat{\boldsymbol{\mu}}$.

Algorithm 1 Single-Modality PCSBL

Input: Selection matrix $\boldsymbol{A}$, measurement vector $\boldsymbol{y}$
Parameters: coupling β, Gamma parameters $a, b, c, d > 0$
threshold τ, max iterations T
Output: Binary occupancy map $\hat{\boldsymbol{x}}$
Initialization: $t \leftarrow 0$, $\boldsymbol{\alpha}^{(0)} \leftarrow \boldsymbol{1}$, $(\sigma^2)^{(0)} \leftarrow 0.5$
Precompute $\boldsymbol{A}^\top \boldsymbol{A}$ and $\boldsymbol{A}^\top \boldsymbol{y}$
repeat
  Update $\hat{\boldsymbol{\mu}}^{(t)}$ and $\hat{\boldsymbol{\Phi}}^{(t)}$ via (4) ▷ E-step
  Compute $\boldsymbol{D}^{(t)}$ via (2) with $\boldsymbol{\alpha} = \boldsymbol{\alpha}^{(t)}$
  Update $\boldsymbol{\alpha}^{(t+1)}$, $(\sigma^2)^{(t+1)}$ via (5) and (6) ▷ M-step
  $t \leftarrow t + 1$
until convergence or $t > T$
Compute $\hat{\boldsymbol{x}}$ by thresholding using τ

B. Fusion OGM PCSBL

Building on the single-modality PCSBL framework, this section describes the fusion of LiDAR and radar measurements using PCSBL-based common sparse (CS) and common innovative sparse (CIS) approaches [18].

1) Fusion Measurement Model: PCSBL-based fusion approaches, CS, and CIS rely on two separate measurement models. Let the measurement matrix and vector are denoted by $\boldsymbol{A}_\text{L} \in \{0,1\}^{2M_\text{L} \times N}$ and $\boldsymbol{y}_\text{L} \in \{0,1\}^{2M_\text{L}}$ for LiDAR, and $\boldsymbol{A}_\text{R} \in \{0,1\}^{2M_\text{R} \times N}$ and $\boldsymbol{y}_\text{R} \in \{0,1\}^{2M_\text{R}}$ for radar, where M_L and M_R are the number of LiDAR and radar measurements, respectively.

The CS model assumes the LiDAR $\boldsymbol{x}_\text{L}$ and radar maps $\boldsymbol{x}_\text{R}$ correspond to the same map $\boldsymbol{x}$, leading to a model that concatenates the LiDAR and radar measurements. With $\boldsymbol{x} = \boldsymbol{x}_\text{L} = \boldsymbol{x}_\text{R}$ in (1), we arrive at

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_\text{L} \\ \boldsymbol{y}_\text{R} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_\text{L} \\ \boldsymbol{A}_\text{R} \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} \boldsymbol{n}_\text{L} \\ \boldsymbol{n}_\text{R} \end{bmatrix} = \boldsymbol{A}_\text{CS}\boldsymbol{x} + \boldsymbol{n}_\text{CS} \tag{7}$$

where $\boldsymbol{n}_\text{L}$ and $\boldsymbol{n}_\text{R}$ are LiDAR and radar noise with variances σ_L^2 and σ_R^2, respectively. In contrast, the CIS model accounts for unreliable sensors or misalignment between the sensors that could lead to significant inconsistencies between $\boldsymbol{x}_\text{L}$ and $\boldsymbol{x}_\text{R}$. Therefore, CIS introduces error variables

$$\boldsymbol{x}_\text{L} = \boldsymbol{x}_\text{c} + \boldsymbol{x}_{\Delta\text{L}}, \quad \text{and} \quad \boldsymbol{x}_\text{R} = \boldsymbol{x}_\text{c} + \boldsymbol{x}_{\Delta\text{R}}$$

where $\boldsymbol{x}_\text{c}$ is the common part of the signal, and $\boldsymbol{x}_{\Delta\text{L}}$ and $\boldsymbol{x}_{\Delta\text{R}}$ are the error collectors for LiDAR and radar, respectively.

From (1), the CIS measurement model is then formulated as

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_{\mathrm{L}} \\ \boldsymbol{y}_{\mathrm{R}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{\mathrm{L}} & \boldsymbol{A}_{\mathrm{L}} & \boldsymbol{0} \\ \boldsymbol{A}_{\mathrm{R}} & \boldsymbol{0} & \boldsymbol{A}_{\mathrm{R}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{\mathrm{c}} \\ \boldsymbol{x}_{\Delta\mathrm{L}} \\ \boldsymbol{x}_{\Delta\mathrm{R}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{n}_{\mathrm{L}} \\ \boldsymbol{n}_{\mathrm{R}} \end{bmatrix} = \boldsymbol{A}_{\mathrm{CIS}} \boldsymbol{x}_{\mathrm{CIS}} + \boldsymbol{n}_{\mathrm{CIS}} \tag{8}$$

where $\boldsymbol{A}_{\mathrm{CIS}} \in \mathbb{R}^{2(M_{\mathrm{L}}+M_{\mathrm{R}})\times 3N}$ and $\boldsymbol{x}_{\mathrm{CIS}} \in \mathbb{R}^{3N}$. Only $\boldsymbol{x}_{\mathrm{c}}$ is used to estimate the OGM, and the error collectors are discarded.

2) Fusion PCSBL: The fusion methods [18] extend on the [3] method using the measurement models in (7) and (8).

The CS model in (8) allows us to directly apply the PC-SBL algorithm. From the measurement model, we have

$$p(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{\alpha}, \sigma_{\mathrm{L}}^2, \sigma_{\mathrm{R}}^2) = \mathcal{N}(\boldsymbol{A}_{\mathrm{CS}}\boldsymbol{x}, \boldsymbol{C})$$

where the corresponding noise covariance is

$$\boldsymbol{C} = \begin{bmatrix} \sigma_{\mathrm{L}}^2 \boldsymbol{I}_{2M_{\mathrm{L}}} & \boldsymbol{0} \\ \boldsymbol{0} & \sigma_{\mathrm{R}}^2 \boldsymbol{I}_{2M_{\mathrm{R}}} \end{bmatrix}.$$

Then, the posterior mean and covariance of the Gaussian $p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\alpha}, (\sigma_{\mathrm{L}}^2), (\sigma_{\mathrm{R}}^2))$ during the E-step is calculated as

$$\begin{aligned} \hat{\boldsymbol{\mu}}_{\mathrm{CS}} &= \hat{\boldsymbol{\Phi}}_{\mathrm{CS}} \left(\sigma_{\mathrm{L}}^{-2} \boldsymbol{A}_{\mathrm{L}}^{\top} + \sigma_{\mathrm{R}}^{-2} \boldsymbol{A}_{\mathrm{R}}^{\top} \right) \boldsymbol{y}, \\ \hat{\boldsymbol{\Phi}}_{\mathrm{CS}} &= \left(\sigma_{\mathrm{L}}^{-2} \boldsymbol{A}_{\mathrm{L}}^{\top} \boldsymbol{A}_{\mathrm{L}} + \sigma_{\mathrm{R}}^{-2} \boldsymbol{A}_{\mathrm{R}}^{\top} \boldsymbol{A}_{\mathrm{R}} + \boldsymbol{D} \right)^{-1}. \end{aligned} \tag{9}$$

During the M-step, the noise variances for each modality are updated as

$$\begin{aligned} \left(\sigma_{\mathrm{L}}^2\right)^{(t+1)} &= \frac{2d + \left\| \boldsymbol{y}_{\mathrm{L}} - \boldsymbol{A}_{\mathrm{L}} \hat{\boldsymbol{\mu}}_{\mathrm{CS}}^{(t)} \right\|_2^2 + \mathrm{tr}\left(\boldsymbol{A}_{\mathrm{L}}^{\top} \boldsymbol{A}_{\mathrm{L}} \boldsymbol{\Phi}_{\mathrm{CS}}^{(t)} \right)}{2c + 2M_{\mathrm{L}}} \\ \left(\sigma_{\mathrm{R}}^2\right)^{(t+1)} &= \frac{2d + \left\| \boldsymbol{y}_{\mathrm{R}} - \boldsymbol{A}_{\mathrm{R}} \hat{\boldsymbol{\mu}}_{\mathrm{CS}}^{(t)} \right\|_2^2 + \mathrm{tr}\left(\boldsymbol{A}_{\mathrm{R}}^{\top} \boldsymbol{A}_{\mathrm{R}} \hat{\boldsymbol{\Phi}}_{\mathrm{CS}}^{(t)} \right)}{2c + 2M_{\mathrm{R}}} \end{aligned} \tag{10}$$

and $\boldsymbol{\alpha}_{\mathrm{CS}}$ is updated as in single-modality PCSBL in (5).

Similarly, CIS also uses the PCSBL framework with the common component $\boldsymbol{x}_{\mathrm{c}}$ following a Gaussian prior with a coupled precision. However, the LiDAR and radar error components $\boldsymbol{x}_{\Delta\mathrm{L}}$ and $\boldsymbol{x}_{\Delta\mathrm{R}}$ are assumed to follow independent zero-mean Gaussian priors. So, $\boldsymbol{x}_{\mathrm{CIS}} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{D}_{\mathrm{CIS}}^{-1})$ where the precision matrix is given by

$$D_{\mathrm{CIS}}[n,n] = \begin{cases} \alpha_{\mathrm{c}}[n'] + \beta \sum_{m \in \mathcal{L}_{n'}} \alpha_{\mathrm{c}}[m], & \text{if } n' \le N \\ \alpha_{\Delta\mathrm{L}}[n' - N], & \text{if } N < n' \le 2N \\ \alpha_{\Delta\mathrm{R}}[n' - 2N], & \text{if } 2N < n' \le 3N. \end{cases}$$

The entries of the precision vectors $\boldsymbol{\alpha}_{\mathrm{c}}$, $\boldsymbol{\alpha}_{\Delta\mathrm{L}}$ and $\boldsymbol{\alpha}_{\Delta\mathrm{R}}$ follow the Gamma prior in (3), with shape parameter a as a_{c}, a_{L}, and a_{R}, respectively. Then, the posterior mean and covariance of $\boldsymbol{x}_{\mathrm{CIS}}$ in the E-step are calculated as

$$\begin{aligned} \hat{\boldsymbol{\mu}}_{\mathrm{CIS}} &= \hat{\boldsymbol{\Phi}}_{\mathrm{CIS}} \boldsymbol{A}_{\mathrm{CIS}}^{\top} \boldsymbol{C}^{-1} \boldsymbol{y}, \\ \hat{\boldsymbol{\Phi}}_{\mathrm{CIS}} &= \left(\boldsymbol{A}_{\mathrm{CIS}}^{\top} \boldsymbol{C}^{-1} \boldsymbol{A}_{\mathrm{CIS}} + \boldsymbol{D}_{\mathrm{CIS}} \right)^{-1}. \end{aligned} \tag{11}$$

TABLE I
ORDER OF COMPLEXITY COMPARISON

Algorithm	Computational complexity
Single modality	$\mathcal{O}(TN^3 + MN^2)$
CS and CIS fusion	$\mathcal{O}(TN^3 + (M_{\mathrm{L}} + M_{\mathrm{R}})N^2)$
BGK	$\mathcal{O}(N \log(M))$
ISM	$\mathcal{O}(LM)$

The M-step update is given by

$$\begin{aligned} \left(\sigma_{\mathrm{L}}^2\right)^{(t+1)} &= \frac{1}{2c + 2M_{\mathrm{L}}} \left(2d + \left\| \boldsymbol{y}_{\mathrm{L}} - \boldsymbol{A}_{\mathrm{L}} \left(\hat{\boldsymbol{\mu}}_{\mathrm{c}}^{(t)} + \hat{\boldsymbol{\mu}}_{\Delta\mathrm{L}}^{(t)} \right) \right\|_2^2 + \mathrm{tr}\left(\boldsymbol{A}_{\mathrm{CIS}}^{T} \boldsymbol{A}_{\mathrm{CIS}} \hat{\boldsymbol{\Phi}}_{\mathrm{CIS}}^{(t)} \right) \right) \\ \left(\sigma_{\mathrm{R}}^2\right)^{(t+1)} &= \frac{1}{2c + 2M_{\mathrm{R}}} \left(2d + \left\| \boldsymbol{y}_{\mathrm{R}} - \boldsymbol{A}_{\mathrm{R}} \left(\hat{\boldsymbol{\mu}}_{\mathrm{c}}^{(t)} + \hat{\boldsymbol{\mu}}_{\Delta\mathrm{R}}^{(t)} \right) \right\|_2^2 + \mathrm{tr}\left(\boldsymbol{A}_{\mathrm{CIS}}^{T} \boldsymbol{A}_{\mathrm{CIS}} \hat{\boldsymbol{\Phi}}_{\mathrm{CIS}}^{(t)} \right) \right). \end{aligned}$$

and $\boldsymbol{\alpha}_{\mathrm{c}}$ is updated via (5) as in single-modality PCSBL, while the remaining $\boldsymbol{\alpha}_{\Delta\mathrm{L}}$ and $\boldsymbol{\alpha}_{\Delta\mathrm{R}}$ are updated using (5) with $\beta = 0$. For more details on CIS and CS derivations, refer to [18].

C. PCSBL Complexity Considerations

Having introduced the PCSBL algorithm, we next present a complexity analysis for both single and multimodal PCSBL.

1) Single-Modality PCSBL: The main computational challenge in single-modality PCSBL arises from the matrix inversion in the E-step given by (4). The precomputation $\boldsymbol{A}^{\top}\boldsymbol{A}$ requires $\mathcal{O}(MN^2)$ complexity, while the matrix inversion step has $\mathcal{O}(N^3)$ complexity. Since the number of EM iterations repeats the E-step T times, the $\mathcal{O}(N^3)$ matrix inversion complexity is further magnified, making PCSBL computationally expensive for large-scale problems. Additional costs include constructing the selection matrix at $\mathcal{O}(MN)$ and computing ray intersections at $\mathcal{O}(LM)$, where L is the average number of cells intersected per ray.

2) Fusion PCSBL: The CS fusion method retains the same structure and computational characteristics as the single-modality PCSBL. However, it adds overhead from the separate computation of $\boldsymbol{A}_{\mathrm{L}}^{\top}\boldsymbol{A}_{\mathrm{L}}$ and $\boldsymbol{A}_{\mathrm{R}}^{\top}\boldsymbol{A}_{\mathrm{R}}$, as well as from the additional noise update in (10). CIS fusion includes this same overhead and further increases the cost due to a three-times larger selection matrix used for the error collectors. This impacts both the E-step in (11) and the noise update.

The overall computational complexities, along with those of single-modality ISM [1] and BGK [14], are summarized in Table I. As evident from Table I, PCSBL is the most computationally demanding algorithm among model-based mapping methods, despite its superior map accuracy. Therefore, our goal is to optimize the matrix computations in both single- and multimodal PCSBL, enabling fast and scalable occupancy mapping by leveraging the inherent structure of the measurement model, without compromising accuracy.

III. PCSBL COMPUTATIONAL REDUCTION TECHNIQUES

We first discuss the strategies to improve the computational efficiency of single-modality PCSBL-based mapping

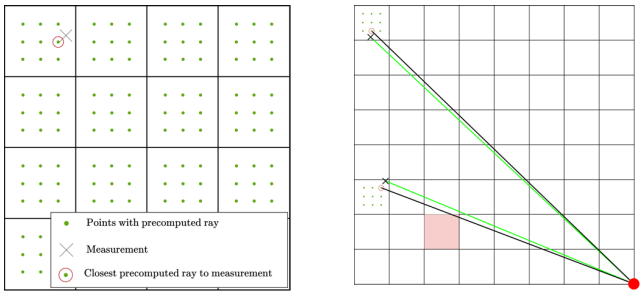


Fig. 1. (a) Precomputed rays sampled at uniformly spaced endpoints, with a spatial tree used to find the closest precomputed point for each measurement. (b) Comparison between the precomputed ray (black line) and the true ray (green line) from the ego center (red dot): red cells highlight incorrectly identified free cells.

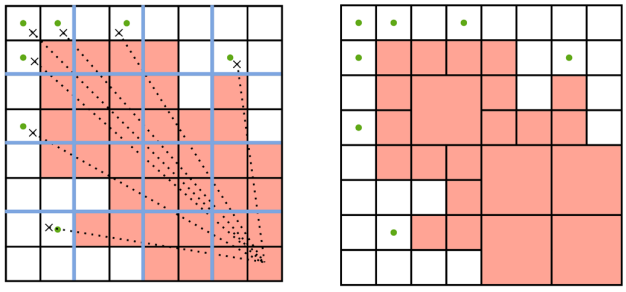


Fig. 2. Adaptive grid resolution using quadtrees refines the grid based on measurements. In (a), cells with measurements (crosses) are occupied (green) and cells along the rays are free (red). Homogeneous higher level nodes (blue grids) in (a) are merged into larger cells, creating the structure in (b).

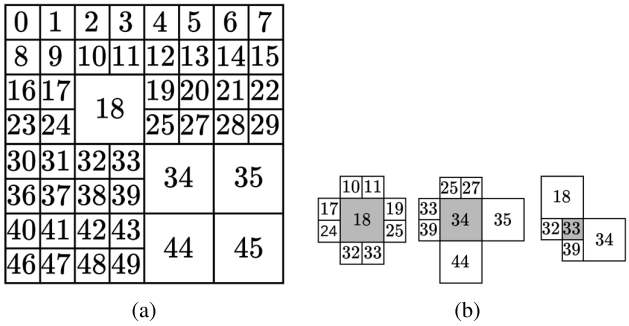


Fig. 3. Indexed pruned quadtree following row-major indexing in (a), with varying pattern coupling structures of cells neighboring selected cells (gray cells) in (b).

algorithms. These strategies operate in two stages. In the first stage, we speed up selection matrix construction using a precomputed ray lookup table combined with compressed sparse row matrices, while reducing signal dimension through extended test-data quadtrees. The second stage applies PCSBL to the reduced model using two acceleration strategies: PO, which parallelizes submap processing, and CP, which permutes measurement matrix columns to exploit spatial structure. Then, we extend these ideas to fusion PCSBL in Section II-B.

A. Ray Lookup Selection Matrix Construction

Building $\boldsymbol{A} \in \{0,1\}^{2M\times N}$ is computationally expensive due to the online ray intersection calculations. To tackle this, we perform the computations offline using a lookup table. Specifically, we generate U reflection points in each of the N grid cells and precompute the ray intersections from the sensor origin to these UN points according to the LiDAR and radar models in (1). The results are stored in a lookup table for these points using a kd-tree [39] for faster access. Then, to construct the measurement matrix $\boldsymbol{A}$ for a given set of reflection points, each reflection point queries its nearest neighbor among the UN precomputed points and retrieves its associated precomputed ray in the lookup table, as illustrated in Fig. 1(a). This is an approximation; Fig. 1(b) shows that it may introduce quantization error where free cells can be identified incorrectly, but the hit cells will always be identified correctly.

This method reduces the computational complexity required to obtain the rays from $\mathcal{O}(LM)$ to $\mathcal{O}(M\log(UN))$ with memory complexity of $\mathcal{O}(UNL)$ where L is the average number of cells intersected by the rays to the precomputed points. For example, an occupancy grid with $N = 16384$ cells and $U = 25$ points per cell requires only 150 MB of memory to store all the necessary intersections.

Furthermore, using measurements to access the precomputed rays, the selection matrix for LiDAR and radar can then be constructed using the compressed sparse row format, reducing the complexity from $\mathcal{O}(MN)$ to $\mathcal{O}(\|\boldsymbol{A}\|_0)$, where $\|\boldsymbol{A}\|_0$ refers to the number of nonzero elements in $\boldsymbol{A}$.

B. Test-Data Quadtrees

The complexity of the PCSBL-based mapping algorithm is further reduced using the adaptive resolution technique of test-data quadtrees [9]. The key idea is to coarsen the representation in areas that are consistently free, based on sensor measurements. First, we query the precomputed ray lookup table to retrieve occupied and free cells along each ray path. Then, we populate the quadtree with occupancy states as shown in Fig. 2(a). Finally, any node whose child cells are all free is pruned, effectively merging multiple free cells into a single larger cell, as demonstrated in Fig. 2(b). This results in variable-sized cells: fine resolution is retained in occupied regions, and larger cells represent free regions. It reduces the effective number of unknowns N in the map vector, improving the algorithm complexity.

With varying resolution of the grid cells, standard row indexing is no longer applicable; however, row-major ordering is maintained for vectorizing the 2-D map to a map vector.

The indexing structure changes to Fig. 3(a). In addition, this technique alters the neighbors of each cell. Thus, the coupling pattern based on the direct neighbors $\mathcal{L}_n$ for the nth cell needs to be modified as illustrated in Fig. 3(b). Furthermore, it is observed that pruned cells are more likely to be free cells. Larger cells at higher pruning levels have more neighbors, which can disproportionately influence free-space information. Thus, only one level of pruning is recommended.

C. PCSBL Matrix Computation Acceleration Techniques

With a fast measurement matrix construction and fewer unknowns, we next apply PCSBL. To speed up matrix operations, we introduce two strategies: PO and CP by exploiting the structure in the measurement matrix $\boldsymbol{A}$.

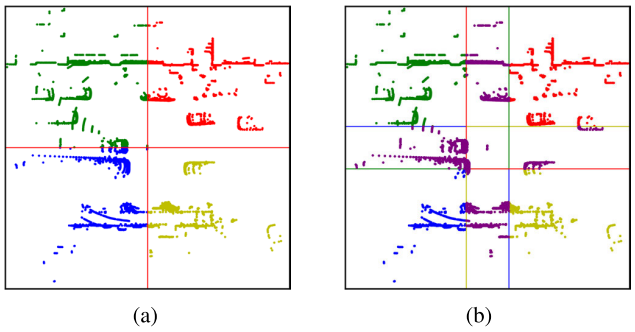


Fig. 4. Partitioning LiDAR measurements into $K = 4$ regions. (a) Nonoverlapping partition. (b) Overlapping partition (overlap in purple).

1) Partition and Overlap: In the measurement model, each LiDAR measurement affects only the grid cells along its sensor ray within a narrow angular sector. One approach is to partition the measurements into K discrete angular regions, as shown in Fig. 4(a) for $K = 4$. Then, the PCSBL algorithm can be applied to each region, recovering an N/K-sized map from M/K LiDAR measurements on average, resulting in the total complexity of $\mathcal{O}(TN^3/K^2 + MN^2/K)$.

By computing K independent maps, the pattern coupling across the region boundaries is lost. To alleviate this, measurements can be partitioned with overlaps, as illustrated in Fig. 4(b). Given the mean $\hat{\boldsymbol{\mu}}_k$ and covariance $\hat{\boldsymbol{\Phi}}_k$ for each region k, the maps are combined via Bayesian fusion, i.e.,

$$\hat{\mu}_{\text{merged}}[n] = \frac{\sum_{r\in\mathcal{R}_n} \frac{\hat{\mu}_r[n]}{\hat{\Phi}[n,n]}}{\sum_{r\in\mathcal{R}_n} \frac{1}{\hat{\Phi}_r[n,n]}}$$

where $\mathcal{R}_n$ denotes the set of regions that overlap at the nth cell. The resulting merged mean map $\hat{\boldsymbol{\mu}}_{\text{merged}}$ is then thresholded to yield the final binary occupancy estimate $\hat{\boldsymbol{x}}$. Overlapping regions require repeated occupancy estimation for shared cells, increasing computational complexity compared with nonoverlapping partitioning.

2) Region-Based CP: In the benchmark PCSBL [3], the measurement model operates on row-indexed grids [see Fig. 5(a)]. As shown in Fig. 5(a), LiDAR rays from distinct angular sectors (e.g., top versus bottom) intersect mutually disjoint sets of cells. Consequently, the selection matrix is structurally split in the middle [dotted line in Fig. 5(d)]; map cell indices are equivalent to selection matrix column indices. Exploiting this property further, the row-based indexing can be permuted to a K-region-based indexing, where each region is defined by an angular sector around the ego center. For $K = 4$, the reindexing shown in Fig. 5(b) produces a four-block structure in $\boldsymbol{A}$ [see Fig. 5(e)], with nonzero entries of the selection matrix perfectly within each block. Consequently, it $\boldsymbol{A}^\top\boldsymbol{A}$ is block-diagonal. The computational complexity is reduced through block matrix operations in several areas: precomputation of $\boldsymbol{A}^\top\boldsymbol{y}$ and $\boldsymbol{A}^\top\boldsymbol{A}$, efficient residual computation for the noise variance update in (6), and most significantly, optimized matrix inversion for posterior covariance and multiplication operations in the mean expectation steps in (4). This achieves the same complexity reduction described in Section II-C.1.

Furthermore, for $K > 4$ in the LiDAR measurement model, the region boundaries can pass through grids, and the LiDAR

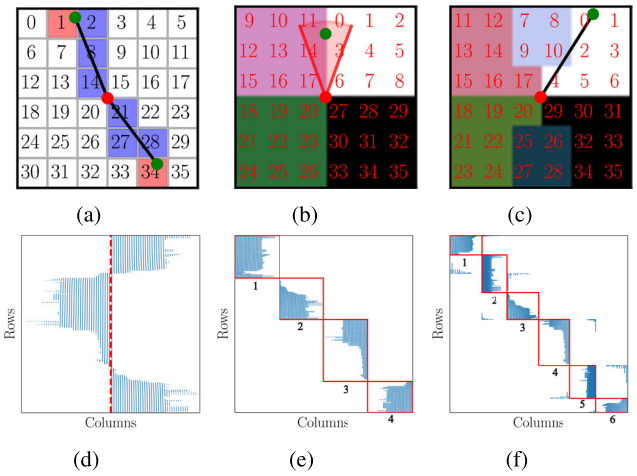


Fig. 5. Grid indexing and selection matrices. Different regional indexing approaches with the ego center (red dot), the reflection point (green dot), LiDAR rays (black line), and radar beam (red) (top). Associated selection matrices where red blocks indicate numbered regions and blue and white boxes represent nonzero and zero matrix values, respectively, (bottom). (a) Indexing: K. (b) Indexing : K. (c) Indexing: K. (d) $\boldsymbol{A}$. (e) $\boldsymbol{A}$. (f) $\boldsymbol{A}$.

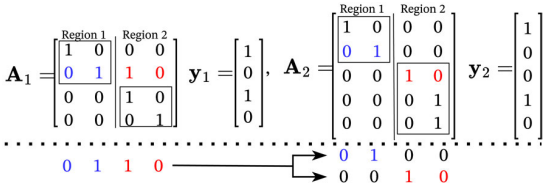


Fig. 6. Splitting measurements spanning multiple regions: the original matrix A_1 (left) contains a row with overlapping elements (red), which is split at region boundaries and reassigned to form the block-structured matrix A_2 (right), adding extra rows (highlighted below the dotted line).

ray may intersect cells belonging to multiple regions. For example, the LiDAR ray in Fig. 5(c) results in a $\boldsymbol{A}$ matrix like in Fig. 5(f) where certain rows have nonzero entries spanning multiple regions. In the case of the radar measurement model, the beamwidth can cause a single beam to intersect multiple regions, as shown in Fig. 5(b). This breaks the block structure and hinders the use of block matrix operations. This can be dealt with by discarding the elements outside the blocks to retain the block structure. Alternatively, we split rows spanning multiple regions to enforce the block structure so that no information is discarded. Fig. 6 shows an example where a row with support falling in two regions is split into two rows.

To summarize, the computational savings from our three accelerations techniques for PCSBL as follows: 1) approximate raycasting with U representative reflection points reduces complexity for sparse selection matrix construction from $\mathcal{O}(MN+LM)$ to $\mathcal{O}(\|\boldsymbol{A}\|_0+M\log(UN))$; 2) quadtrees reduces N; and 3) both PO and CP with K regions lower complexity from $\mathcal{O}(TN^3+MN^2)$ to $\mathcal{O}(TN^3/K^2+MN^2/K)$. PO requires overlap fusion that effectively increases problem size, while CP maintains full coupling across K regions without fusion overhead.

D. Extension of Fusion PCSBL Algorithms

The fast measurement matrix construction with adaptive model dimension reduction from Sections III-A and III-B

applies directly to CS and CIS fusion methods. Similarly, the PO strategy extends easily to multimodal fusion, while CP requires additional considerations, as explained below.

CS fusion retains the same selection matrix structure as in (7), where columns correspond to cells of a single occupancy map, enabling the same region-based CP. The permuted CS selection matrix creates a block diagonal $\boldsymbol{A}_{\mathrm{CS}}^{\top}\boldsymbol{A}_{\mathrm{CS}}$, and the residual and trace calculations in (10) exploit identical structures as single-modality PCSBL-CP. Structurally, applying CP to CS fusion mirrors the single-modality case, utilizing the same block structures in (9) for block inversions and other matrix operations.

For CIS, this approach does not directly apply due to the altered measurement model in (8). With the addition of the error variables, the columns no longer correspond to the cells of a single occupancy map. To restore a block-diagonal structure in $\boldsymbol{A}_{\mathrm{CIS}}^{\top}\boldsymbol{A}_{\mathrm{CIS}}$, we permute $\boldsymbol{x}_{\mathrm{CIS}}$ so that the triplets $\left(x_{\mathrm{c}}[i],\, x_{\Delta\mathrm{L}}[i],\, x_{\Delta\mathrm{R}}[i]\right)$ appear consecutively for $i = 1, \ldots, N$. The permuted vector is given by

$$\left[x_{\mathrm{c}}\,[1]\;x_{\Delta\mathrm{L}}\,[1]\;x_{\Delta\mathrm{R}}\,[1]\;\ldots\;x_{\mathrm{c}}\,[N]\;x_{\Delta\mathrm{L}}\,[N]\;x_{\Delta\mathrm{R}}\,[N]\right]$$

which enforces block-diagonality in $\boldsymbol{A}_{\mathrm{CIS}}^{\top}\boldsymbol{A}_{\mathrm{CIS}}$ by eliminating cross-terms between different columns belonging to separate regions. While the CP method reduces complexity from $\mathcal{O}(N^3 + N^2(M_{\mathrm{L}} + M_{\mathrm{R}}))$ to $\mathcal{O}(N^3/K^2 + N^2(M_{\mathrm{L}} + M_{\mathrm{R}})/K)$ for both CS and CIS approaches, CIS requires a selection matrix with three times more columns than CS due to the error collectors.

IV. Performance Evaluation

We evaluate the algorithms using the publicly available nuScenes [20] and RADIATE datasets [36].

A. Experimental Setting

The evaluation metrics are 1) runtime; 2) intersection over bounding box (IoBB) to measure the overlap between the estimated and ground truth obstacles; 3) free-space error (defined as the ratio of false positives in ground truth unoccupied areas); and 4) angular scan normalized mean squared error (AS-NMSE) from [3] that serves as a drivable area metric.

We evaluate the performance of the following accelerated PCSBL variants: 1) CP without quadtree acceleration; 2) PO with two-cell overlap without quadtree acceleration; and 3) quadtree-accelerated CP (QCP), as CP outperforms PO. All our PCSBL variants employ approximate ray lookup for selection matrix construction using $U = 25$, and use $K = 16$ regions for PO and CP accelerations.

The benchmark methods are BGK [14], ISM [37], and baseline PCSBL [3]. Benchmark methods from [3], [14], [18], and [37] use exact ray casting. PCSBL parameters are initialized according to specifications in [3] and [18].

We evaluate our methods across two datasets with different modality focuses. For nuScenes, we test single-modality methods on LiDAR data, comparing CP, PO, and QCP. For RADIATE, we evaluate both radar-only methods, PCSBL, CP, QCP, and fusion-based methods combining LiDAR and radar. The fusion-based methods include CS and CIS [18] along with their CP and QCP accelerated variants (CP-CS, QCP-CS, and CP-CIS). We omit the evaluation results for single-modality LiDAR data on RADIATE, as it employs the same LiDAR sensor type as nuScenes and leads to repetitive results. Meanwhile, we omit radar evaluation on nuScenes due to the known issue of excessively sparse radar point clouds that often contain no points in some of the surrounding objects [3], [40].

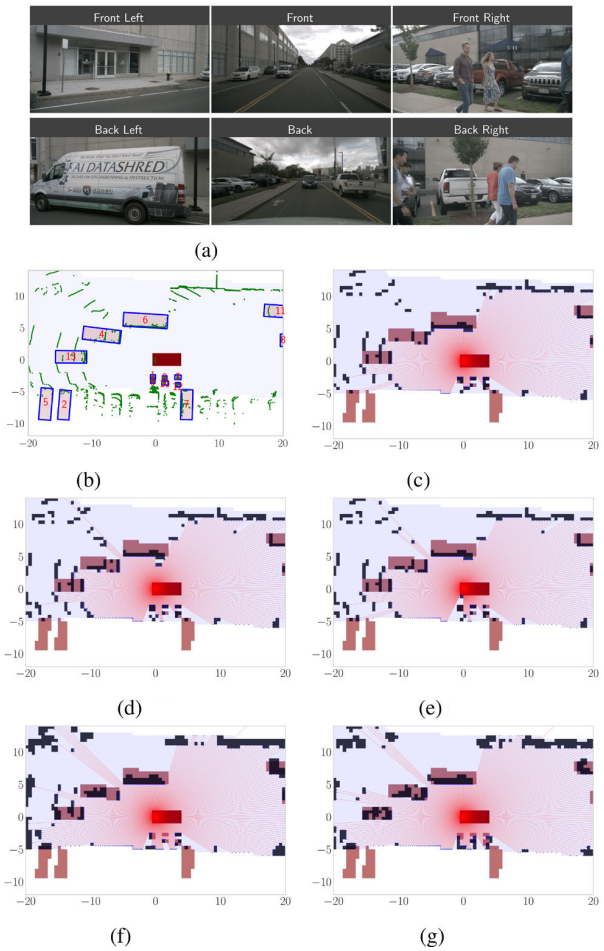


Fig. 7. Occupancy mapping comparison on Scene 204 Frame 0 from nuScenes. (c)–(h) show the ego vehicle (dark red), ground truth boxes (pink), estimated occupancy (black), and road mask (pale blue). The vehicle front faces right. (a) Ego vehicle cameras. (b) LiDAR scan. (c) PCSBL. (d) PO. (e) QCP. (f) BGK. (g) ISM.

B. Comparison on nuScenes Dataset

Fig. 7 and Table II show evaluations on Scene 204 from nuScenes, with $M = 6231$ LiDAR measurements and a map of $N = 2926$ cells. The nuScenes dataset provides road topology information, allowing efficient map construction only in the area of interest; thus, the number of cells in a given scene is dependent on the unique road geometry.

Quantitative results from Table II show that the PCSBL variants run up to 20 times faster than the benchmark PCSBL and achieve the same detection rate while incurring only a marginal increase in free-space error and AS-NMSE. Although BGK and ISM have good IoBB and runtimes, they suffer from poor free-space error and lower AS-NMSE because of inaccurate edge detection. PCSBL outperforms BGK and ISM on all metrics, despite lower IoBB for large objects

TABLE II
RESULTS OF MODEL COMPARISON ON NUSCENES LIDAR SCENE 204 FRAME 0

Metric	ID	Methods					
		PCSBL	PO	CP	QCP	BGK	ISM
IoBB	0	1.000	1.000	1.000	1.000	0.000	0.000
	1	1.000	1.000	1.000	1.000	0.500	0.000
	2	0.000	0.000	0.000	0.000	0.000	0.000
	3	1.000	0.500	1.000	0.500	1.000	0.000
	4	0.188	0.292	0.313	0.271	0.396	0.479
	5	0.027	0.027	0.054	0.054	0.108	0.081
	6	0.279	0.279	0.279	0.279	0.492	0.459
	7	0.028	0.028	0.028	0.028	0.028	0.000
	8	0.500	0.500	0.500	0.500	0.750	0.000
	9	0.500	1.000	0.500	1.000	0.500	0.500
	10	0.500	0.500	0.500	0.500	0.500	1.000
	11	0.348	0.348	0.348	0.348	0.696	0.565
	12	0.750	0.750	0.750	0.750	0.500	0.250
	13	0.333	0.333	0.333	0.333	0.472	0.667
Detected		**13**/14	**13**/14	**13**/14	**13**/14	12/14	8/14
AS-NMSE		**0.244**	0.271	0.274	0.280	0.445	0.394
Free-Space		**0.045**	0.053	0.051	0.048	0.079	0.067
Runtime(s)		8.254	1.420	0.428	**0.411**	0.165	0.178

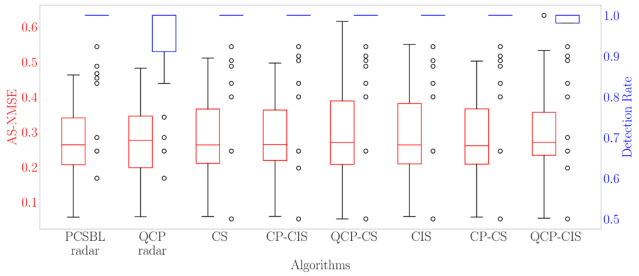


Fig. 8. Object detection rate and AS-NMSE in the ground truth maps over 200 scenes from nuScenes using LiDAR data.

TABLE III
MEAN RUNTIME OVER 200 LIDAR NUSCENES SAMPLES

Model	PCSBL	PO	CP	QCP	BGK	ISM
Time (s)	12.990	1.405	0.672	**0.474**	0.170	0.167

(e.g., targets 4, 6, 11, and 13). The higher IoBB values of BGK and ISM for large objects come at the expense of increased false alarms and failure to detect many pedestrians. PCSBL and its variants are significantly better at pedestrian detection. Meanwhile, BGK misses one pedestrian (target 0), and ISM misses two (target 0 and 3). Finally, the CP method is significantly closer to BGK and ISM in terms of runtime than the benchmark PCSBL.

Furthermore, Fig. 8 and Table III show the statistical performance of the methods averaged over 200 random scenes from nuScenes. Fig. 8 shows that the proposed PCSBL variants achieve similar performance compared with the benchmark PCSBL. It also shows that the BGK and ISM have higher AS-NMSE and lower detection rate compared with the PCSBL and its variants, which is similar to the observations in Scene 204. The results in Table III show that across the three PCSBL algorithms, the proposed methods perform similar to benchmark PCSBL while reducing runtime complexity by more than an order of magnitude.

C. Comparison on RADIATE Dataset

We evaluate single-modal and multimodal PCSBL, presenting single-scene results, statistical results over 40 samples, and time complexity as a function of the number of cells.

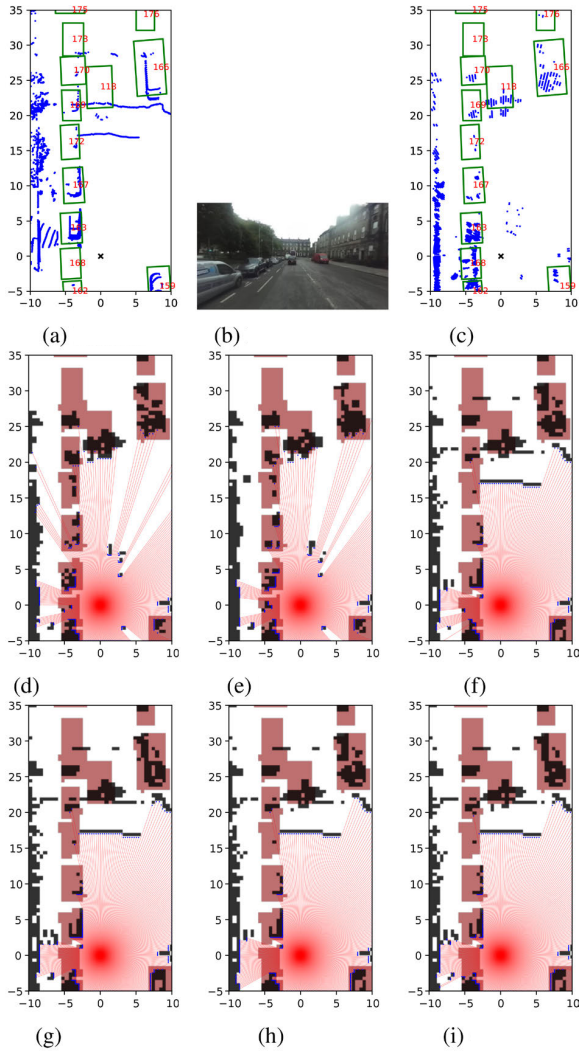


Fig. 9. Occupancy mapping comparison on RADIATE City-3-0, frame 275. Benchmark PCSBL, CS, and CIS are shown alongside their accelerated variants. Ground truth boxes are pink boxes or green outlines. (a) LiDAR. (b) Front camera. (c) Radar. (d) PCSBL radar. (e) QCP radar. (f) CS. (g) QCP-CS. (h) CIS. (i) CP-CIS.

LiDAR and radar modalities from city scene 3-0 are used to compare the accelerated CS and CIS to the benchmark versions in Fig. 9. The scene consists and $M_{\mathrm{L}} = 5702$ LiDAR and $M_{\mathrm{R}} = 1245$ radar measurements. Visually, minimal differences are observed, similar to the nuScenes results. The similar performance is confirmed by the AS-NMSE AND free-space errors in Table IV, and most IoBB values remain the same or very close across PCSBL fusion variants. The acceleration methods on CS fusion decreased the runtime by more than 10 times with CP and 22 times with QCP. Benchmark CIS runs approximately 10 times faster with CP, but is still very slow at 23 s.

Furthermore, Fig. 10 and Table V show the statistical performance of the multimodal and radar PCSBL methods averaged over 40 scenes from RADIATE. It is clear that the accelerations perform closely to the benchmark fusion methods in Fig. 10, in both detection rate and ANMSE.

Fig. 11 analyzes time complexity across varying the number of cells N. Run time performance is evaluated on frame 275

TABLE IV
RESULTS OF RADIATE SCENE CITY-3-0 FRAME 275 FOR SINGLE-MODALITY RADAR AND LIDAR-RADAR FUSION

Metric	ID	Methods							
		PCSBL radar	QCP radar	CS	CP-CS	QCP-CS	CIS	CP-CIS	QCP-CIS
IoBB	118	0.432	0.409	0.375	0.364	0.386	0.375	0.352	0.330
	159	0.105	0.118	0.224	0.237	0.237	0.224	0.224	0.211
	162	0.260	0.260	0.260	0.260	0.260	0.260	0.260	0.260
	163	0.268	0.232	0.286	0.304	0.304	0.286	0.286	0.286
	166	0.402	0.423	0.445	0.423	0.394	0.445	0.409	0.372
	167	0.180	0.197	0.115	0.164	0.098	0.115	0.147	0.049
	168	0.160	0.180	0.100	0.080	0.100	0.060	0.060	0.080
	169	0.298	0.213	0.149	0.213	0.170	0.149	0.213	0.149
	170	0.271	0.288	0.237	0.237	0.237	0.237	0.237	0.237
	172	0.113	0.094	0.019	0.019	0.019	0.019	0.019	0.019
	173	0.000	0.000	0.056	0.056	0.056	0.056	0.056	0.056
	175	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	176	0.200	0.200	0.200	0.200	0.200	0.200	0.200	0.200
Detected		11/13	11/13	**12**/13	**12**/13	**12**/13	**12**/13	**12**/13	**12**/13
AS-NMSE		0.243	0.200	**0.091**	0.092	0.093	0.103	0.094	0.101
Free-Space		**0.055**	0.058	0.068	0.069	0.070	0.068	0.069	0.063
Runtime(s)		7.405	**0.610**	14.989	1.412	**0.669**	210.241	22.621	12.221

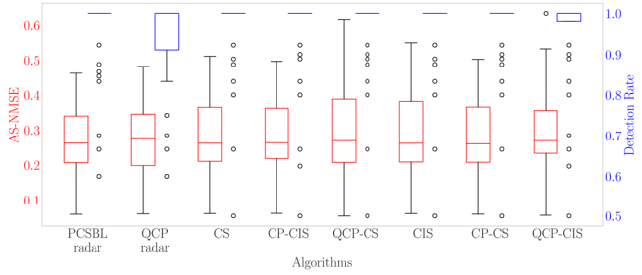


Fig. 10. Object detection rate and AS-NMSE in the ground truth maps over 40 RADIATE samples.

TABLE V
MEAN RUNTIME OVER 40 RADIATE SAMPLES

Model	PCSBL radar	QCP radar	CS	CP-CS	QCP-CS	CIS	CP-CIS	QCP-CIS
Time(s)	6.871	**0.623**	14.593	1.481	**0.709**	208.483	20.884	8.771

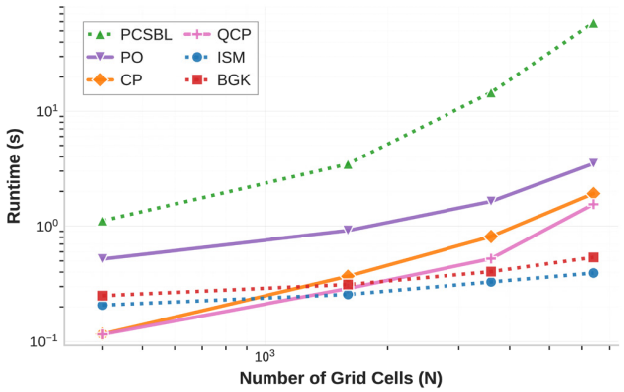


Fig. 11. Time complexity comparison using single-modality LiDAR measurements from RADIATE City-3-0 frame 275 for varying the map size N.

(city-3-0, RADIATE dataset) using LiDAR within a 20×20 m ego vehicle grid, with N varied by adjusting cell size (map resolution). The accelerated methods demonstrate superior performance compared with BGK and ISM when estimating occupancy for relatively small numbers of cells N. However, as N increases, the QCP and CP methods scale less favorably. The CP and PO methods exhibit better scaling behavior with increasing N due to their $\mathcal{O}(N^3/K^2)$ complexity versus $\mathcal{O}(N^3)$ for benchmark PCSBL. While both accelerated methods significantly outperform PCSBL, PO operates more slowly than CP due to additional occupancy calculations and separate selection matrix overhead. To avoid clutter, fusion methods are not shown in this figure. However, our experiments show that accelerated CIS performed comparably to benchmark single-modality PCSBL, while accelerated CS more closely matched accelerated single-modality variants.

V. CONCLUSION

We introduced an approximate raycasting with a sparse matrix construction method to speed up the selection matrix construction, extended test-data quadtrees to PCSBL to accommodate variable map resolution, and presented two computationally efficient PCSBL techniques (PO and CP) that exploit the selection matrix for OGM reconstruction. These accelerations were further extended to the multimodal setting to achieve similar computational savings. Evaluations on the nuScenes and RADIATE datasets showed that these methods maintain the detection accuracy of the benchmark single modality and fusion PCSBL methods, but with significantly lower runtimes. Among the scalable variants, CP is more efficient than PO, which requires additional occupancy calculations. Accelerated single-modality PCSBL and CS fusion achieve subsecond performance at practical grid resolutions, while CIS fusion needs some improvement for real-time operation. Future work could adopt mapping based on polar grids and implement the algorithms in more efficient languages, such as C or C++, and implement PCSBL for OGM in real time.

REFERENCES

[1] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, Jun. 1989.
[2] T. Rhemrev et al., "ELLAS: Enhancing LiDAR perception with location-aware scanning profile adaptation," *IEEE Sensors J.*, vol. 25, no. 5, pp. 8766–8775, Mar. 2025.
[3] Ç. Önen, A. Pandharipande, G. Joseph, and N. J. Myers, "Occupancy grid mapping for automotive driving exploiting clustered sparsity," *IEEE Sensors J.*, vol. 24, no. 7, pp. 9240–9250, Apr. 2024.
[4] J. Revelles, C. Ureña, and M. Lastra, "An efficient parametric algorithm for octree traversal," *J. WSCG*, vol. 8, nos. 1–3, pp. 212–219, 2000.
[5] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auto. Robots*, vol. 34, no. 3, pp. 189–206, Apr. 2013.
[6] J. Zhou and D. Wen, "Research on ray tracing algorithm and acceleration techniques using KD-tree," in *Proc. Int. Conf. Intell. Comput. Signal Process.*, Apr. 2021, pp. 107–110.
[7] C. H. Walsh and S. Karaman, "CDDT: Fast approximate 2D ray casting for accelerated localization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 3677–3684.
[8] S. O'Callaghan, Fabio. T. Ramos, and H. Durrant-Whyte, "Contextual occupancy maps using Gaussian processes," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 1054–1060.
[9] J. Wang and B. Englot, "Fast, accurate Gaussian process occupancy maps via test-data octrees and nested Bayesian fusion," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 1003–1010, May 2016.
[10] L. Piotrowski, *Robotic Systems: Advanced Techniques and Applications*. Dordrecht, The Netherlands: Springer, 1992.
[11] G. K. Kraetzschmar, G. P. Gassull, and K. Uhl, "Probabilistic quadtrees for variable-resolution mapping of large environments," *IFAC Proc. Volumes*, vol. 37, no. 8, pp. 675–680, Jul. 2004.

[12] Y. Chen, W. Shuai, and X. Chen, "A probabilistic, variable-resolution and effective quadtree representation for mapping of large environments," in *Proc. Int. Conf. Adv. Robot. (ICAR)*, Jul. 2015, pp. 605–610.
[13] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 4093–4098.
[14] K. Doherty, J. Wang, and B. Englot, "Bayesian generalized kernel inference for occupancy map prediction," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 3118–3124.
[15] K. Doherty, T. Shan, J. Wang, and B. Englot, "Learning-aided 3-D occupancy mapping with Bayesian generalized kernel inference," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 953–966, Aug. 2019.
[16] S. Kim and J. Kim, "Recursive Bayesian updates for occupancy mapping and surface reconstruction," *Proc. Australas. Conf. Robot. Autom.*, Dec. 2014, pp. 354–361.
[17] J. Fang, Y. Shen, H. Li, and P. Wang, "Pattern-coupled sparse Bayesian learning for recovery of block-sparse signals," *IEEE Trans. Signal Process.*, vol. 63, no. 2, pp. 360–372, Jan. 2015.
[18] P. Zhai, G. Joseph, N. J. Myers, Ç. Önen, and A. Pandharipande, "Sparsity-aware occupancy grid mapping for automotive driving using radar-LiDAR fusion," in *Proc. IEEE SENSORS*, Oct. 2024, pp. 1–4.
[19] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.
[20] H. Caesar et al., "NuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11621–11631.
[21] R. Van Kempen, B. Lampe, T. Woopen, and L. Eckstein, "A simulation-based end-to-end learning framework for evidential occupancy grid mapping," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jul. 2021, pp. 934–939.
[22] D. Bauer, L. Kuhnert, and L. Eckstein, "Deep, spatially coherent inverse sensor models with uncertainty incorporation using the evidential framework," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 2490–2495.
[23] R. Cheng, C. Agia, Y. Ren, X. Li, and B. Liu, "S3CNet: A sparse semantic scene completion network for LiDAR point clouds," in *Proc. Conf. Robot. Learn.*, 2020, pp. 2148–2161.
[24] L. Sless, B. E. Shlomo, G. Cohen, and S. Oron, "Road scene understanding by occupancy grid learning from sparse radar clusters using semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 867–875.
[25] R. Weston, S. Cen, P. Newman, and I. Posner, "Probably unknown: Deep inverse sensor modelling radar," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 5446–5452.
[26] J. Wilson et al., "MotionSC: Data set and network for real-time semantic mapping in dynamic environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 8439–8446, Jul. 2022.
[27] X. Yan et al., "Sparse single sweep LiDAR point cloud segmentation via learning contextual shape priors from scene completion," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, vol. 35, no. 4, pp. 3101–3109.
[28] C. B. Rist, D. Emmerichs, M. Enzweiler, and D. M. Gavrila, "Semantic scene completion using local deep implicit functions on LiDAR data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 7205–7218, Oct. 2022.
[29] M. Schreiber, V. Belagiannis, C. Gläser, and K. Dietmayer, "A multi-task recurrent neural network for end-to-end dynamic occupancy grid mapping," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2022, pp. 315–322.
[30] A. Popov, P. Gebhardt, K. Chen, and R. Oldja, "NVRadarNet: Real-time radar obstacle and free space detection for autonomous driving," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 6958–6964.
[31] A.-Q. Cao, A. Dai, and R. De Charette, "PaSCo: Urban 3D panoptic scene completion with uncertainty awareness," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2024, pp. 14554–14564.
[32] F. Ding, X. Wen, Y. Zhu, Y. Li, and C. X. Lu, "Robust 3D occupancy prediction with 4D imaging radar," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2024, pp. 101589–101617.
[33] X. Wang et al., "OpenOccupancy: A large scale benchmark for surrounding semantic occupancy perception," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Dec. 2023, pp. 17850–17859.
[34] X. Y. Tian, T. Jiang, L. Yun, Y. Wang, Y. Wang, and H. Zhao, "Occ3D: A large-scale 3D occupancy prediction benchmark for autonomous driving," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2023, pp. 64318–64330.
[35] J. Zhang, Y. Ding, and Z. L. Liu, "OccFusion: Depth estimation free multi-sensor fusion for 3D occupancy prediction," in *Proc. Asian Conf. Comput. Vis.*, Dec. 2024, pp. 232–249.
[36] M. Sheeny, E. De Pellegrin, S. Mukherjee, A. Ahrabian, S. Wang, and A. Wallace, "RADIATE: A radar dataset for automotive perception in bad weather," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 1–7.
[37] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics* (Intelligent Robotics and Autonomous Agents Series). Cambridge, MA, USA: MIT Press, 2005.
[38] F. Harraway, P. Zhai, G. Joseph, and A. Pandharipande, "Computationally-efficient sparsity-aware occupancy grid mapping for automotive driving," in *Proc. IEEE SENSORS*, Oct. 2025.
[39] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 209–226, Sep. 1977.
[40] S. Muckenhuber, E. Museljic, and G. Stettinger, "Performance evaluation of a state-of-the-art automotive radar and corresponding modeling approaches based on a large labeled dataset," *J. Intell. Transp. Syst.*, vol. 26, no. 6, pp. 655–674, Nov. 2022.

Frank Harraway received the B.Eng. degree in electrical and electronic engineering from the University of Stellenbosch, Stellenbosch, South Africa, in 2023, and the M.Sc. degree in electrical engineering from Delft University of Technology, Delft, The Netherlands, in 2025.

His research interests include robotics, signal processing, and machine learning.

Peiyuan Zhai (Graduate Student Member, IEEE) received the B.Eng. degree in information engineering from Xi'an Jiaotong University, Xi'an, China, in 2020, and the M.S. degree in electrical engineering from Delft University of Technology, Delft, The Netherlands, in 2022, where he is currently pursuing the Ph.D. degree with the Signal Processing Systems Group.

His research interests include automotive perception, multimodal sensing, and sparse Bayesian learning.

Geethu Joseph (Senior Member, IEEE) received the Ph.D. degree in electrical communication engineering from the Indian Institute of Science, Bengaluru, India, in 2019.

She is currently a Tenured Assistant Professor with the Signal Processing Systems Group, Delft University of Technology, Delft, The Netherlands. Her research interests include statistical signal processing, network control, and machine learning.

Ashish Pandharipande (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Iowa, Iowa City, IA, USA, in 2002.

He is currently a Technical Director of NXP Semiconductors, Eindhoven, The Netherlands, and a part-time Full Professor with Eindhoven University of Technology, Eindhoven. His research interests include sensing, machine learning, data analytics, and their applications in domains such as autonomous mobility, smart lighting systems, and cognitive wireless systems.