Journal of Robotics, Networking and Artificial Life
Vol. 11(2); December (2025), pp. 89-97
ISSN (Online): 2352-6386; ISSN (Print): 2405-9021
https://alife-robotics.org/jrnal.html

Research Article

Comparative Analysis of LiDAR-Based and Depth-Assisted Image-Based Tree Mapping for Autonomous Forestry Robots

M.A Munjer*, Tan Chi Jie, Eiji Hayashi
Faculty of Computer Science and System Engineering, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka-City, Fukuoka, 820-8502, Japan
Email: munjer.abul-md690@mail.kyutech.jp, tan.jie-chi339@mail.kyutech.jp, haya@ics.kyutech.ac.jp
*Corresponding Author

ARTICLE INFO

Article History

Received 25 November, 2024
Accepted 30 May, 2025

Keywords

LiDAR
DBSCAN
FastSLAM
Eccentricity

ABSTRACT

This study presents a comparative analysis of two real-time tree localization and mapping approaches in forestry environments: an eccentricity-based LiDAR mapping method (EB-LiDAR) and a depth-assisted image-based mapping method (DAIM). EB-LiDAR extracts tree trunk clusters from LiDAR data using DBSCAN clustering and PCA-based eccentricity filtering whereas the DAIM approach integrates YOLO-based visual trunk detection with depth clustering, estimating tree positions through arc-based geometric fitting. Both methods are deployed on a multi-sensor mobile robot platform where LiDAR and IMU data are fused via an Extended Kalman Filter (EKF) to ensure robust odometry. Tree positions are refined into a global frame using SLAM-based transformations and Kalman filtering. Performance evaluation conducted in real-world experiments demonstrates that EB-LiDAR achieves an average positional error of 7.13%, whereas DAIM improves accuracy with a lower average positional error of 4.24%. Experimental results demonstrate that DAIM outperforms EB-LiDAR in positional accuracy, while EB-LiDAR provides higher robustness in detecting all physical tree instances. These comparative results highlight the strengths and limitations of each method, offering valuable insights for autonomous forestry navigation and mapping under varying environmental conditions.

1. Introduction

The growing shortage of labor in various industries, particularly in forestry, has driven the exploration of autonomous mobile robots as potential substitutes for human workers [1]. Among the various techniques developed for Simultaneous Localization and Mapping (SLAM), the FastSLAM algorithm [2], which utilizes particle filters, has shown strong capability in estimating a robot's trajectory while constructing an environmental map simultaneously. This method is especially effective in environments characterized by dynamic elements or partial observability, where traditional SLAM approaches may encounter challenges related to computational demands or data association.

In the context of forestry robotics, prior studies [3] have used Online SLAM to estimate tree positions by identifying cluster centres in LiDAR point-cloud data. However, due to limitations in LiDAR's field of view and resolution, particularly in dense vegetation or close-range scenarios, tree trunks are often only partially captured. As noted in [4], such partial observations result in inaccurate tree centre estimation. To address this, this work proposed an eccentricity-based LiDAR mapping (EB-LiDAR) method, which isolates vertically elongated clusters from 3D point clouds using PCA-based shape filtering. Building upon this, the present study introduces a second, complementary approach—Depth-Assisted Image-Based Mapping (DAIM)—which integrates RGB image detection and aligned depth data. Tree trunks are detected using a YOLO-based object detector, and the corresponding depth region is analysed to estimate tree positions using arc-based geometric fitting. This study enables comparative evaluation of both purely geometric and vision-assisted tree mapping pipelines. The system integrates LiDAR, RGB-D camera, and IMU data through ROS2 [5] and Open3D [6], supported by an Extended Kalman Filter (EKF) for pose estimation. Both methods are tested in same outdoor environments, and tree localization performance is analysed in terms of spatial accuracy and consistency. By comparatively evaluating geometric and semantic sensing strategies, this research highlights their complementary roles in achieving tree mapping for autonomous robots in forest environments.

2. System overview

The proposed forestry mapping system integrates a multi-sensor platform with computational components to

enable autonomous navigation and real-time environmental mapping. Figure 1 illustrates the overall system architecture developed in this research. The system is built upon a differential-drive mobile robot equipped with three primary sensors. Firstly, a Velodyne LiDAR sensor, responsible for capturing high-density 3D point cloud data essential for detecting and localizing tree clusters. On the other hand, another primary sensor is Intel RealSense Depth Camera, providing complementary RGB and depth information to enhance scene understanding and perception robustness. Also, an XSense IMU sensor, delivering orientation, angular velocity, and linear acceleration measurements to improve localization accuracy and maintain robot stability across uneven terrains. A ROS2-powered onboard processor handles real-time control logic, sensor data fusion, and tree mapping operations. LiDAR and camera inputs are utilized for identifying trees and monitoring the environment, while the IMU contributes to motion stability over uneven surfaces.

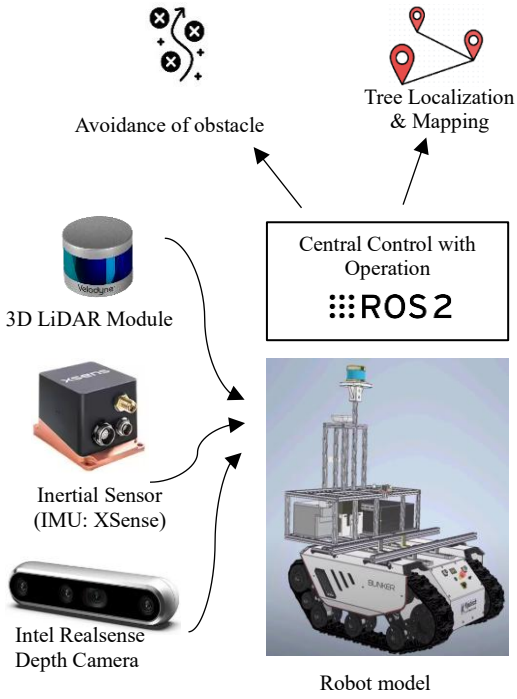


Figure 1 Hardware setup and multi-sensor integration scheme for autonomous tree mapping.

The robot is capable of autonomous obstacle avoidance and spatially registering detected tree locations into a consistent global map framework. Designed for scalability and robustness, this architecture allows autonomous operation in forestry environments. The resulting maps display the robot's traveled trajectory alongside detected tree positions, offering actionable data for forestry operations and supporting advancements in field robotics autonomy.

3. Methodology

3.1. Sequential Pipeline for Tree Mapping

The system follows a structured pipeline for real-time tree detection and mapping, as outlined in Figure 2. Initially, sensor data from the 3D LiDAR, RGB-D camera, and IMU are collected. LiDAR and IMU data undergo combined processing to estimate robust robot odometry through ICP and EKF fusion, as well as geometric analysis of tree trunks via the EB-LiDAR method described in Section 3.3. On the other hand, RGB-D data are processed independently, beginning with YOLO-based semantic detection followed by depth-point clustering and geometric center estimation (DAIM method, Section 3.4). Both approaches independently yield tree positions, which are subsequently transformed into the global frame and refined using Kalman filtering.

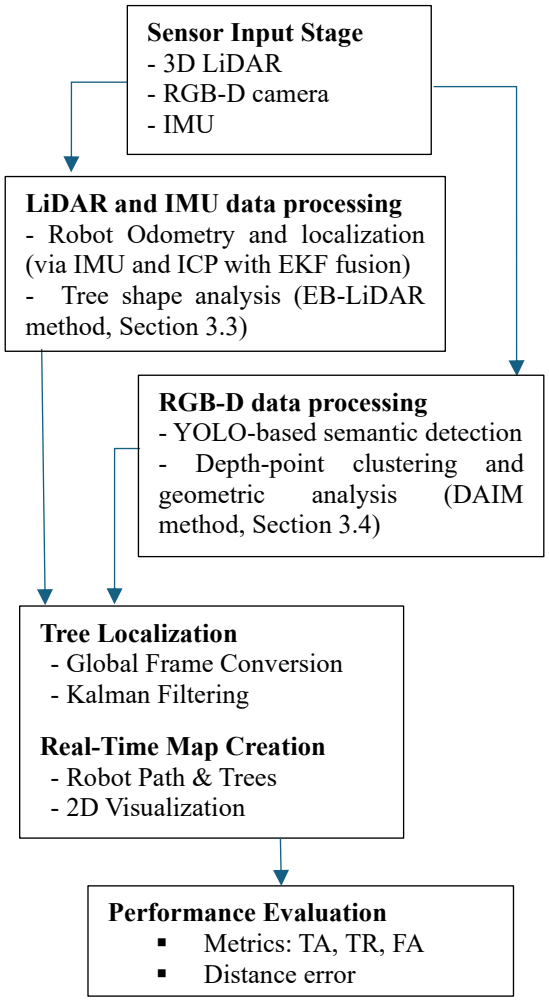


Figure 2 Process Flow and Sensor Integration for Real-Time Tree Mapping.

The final result is a real-time, continuously updated 2D map visualizing the robot's path and detected tree locations. As it is difficult to align the robot's starting position with the precise coordinates of the trees, the ground truth distances between trees were measured manually using a

measuring tape in the field. System performance is evaluated through metrics—True Accepts (TA), True Rejects (TR), False Accepts (FA), and False Rejects (FR) and experimental assessments using distance error analysis, as detailed in subsequent sections.

3.2. *LiDAR Data Capture and Noise Reduction*

The process begins with capturing three-dimensional spatial data using a Velodyne LiDAR mounted on the robot platform. This sensor emits laser pulses and captures their reflections from surrounding objects [7], resulting in a dataset P defined as in Eq. (1)

$$P = \{p_i \mid p_i = (x_i, y_i, z_i) \in \mathbb{R}^3, i=1...,N\} \quad (1)$$

where p_i represents a point with 3D coordinates and N is the total number of points. Since raw LiDAR data contains noise from environmental clutter, a two-step preprocessing routine is applied, as Pass-Through Filtering and Voxel Grid Down-Sampling. Points are filtered based on spatial ranges along each axis. In pass-through filtering , a point p_i is retained only if $\beta_x > x_i > 0, \beta_y > y_i > -\beta_y$, $\beta_z > z_i > -\beta_z$ here β_z , β_y and β_x are axis-specific space thresholds. For the voxel grid down-sampling, a voxel grid filter averages points within small 3D cells (voxels), producing a new dataset P′ can be represented by as in Eq. (2)

$$P' = \left\{ \frac{1}{|V|} \sum_{p_i \in V} p_i \mid V \subset P, \ |V| > 0 \right\} \quad (2)$$

where V represents the points contained within each voxel. This preprocessing pipeline reduces noise while preserving key structural features, enabling efficient downstream processing.

3.3. *Eccentricity-Based LiDAR Mapping (EB-LiDAR)*

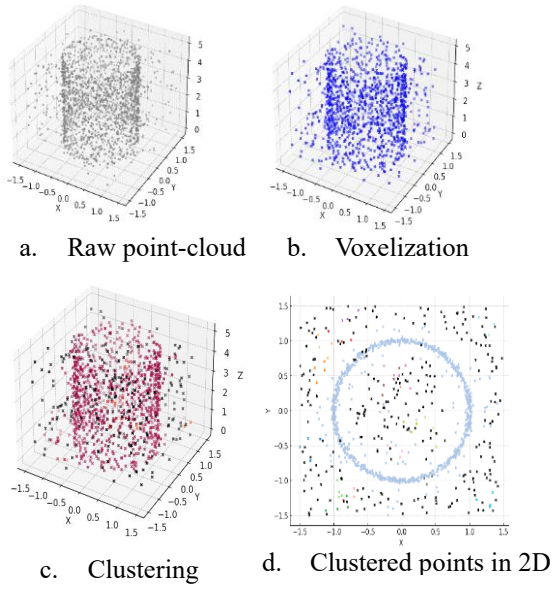
With the filtered point cloud P′, clusters indicative of tree trunks are extracted using the DBSCAN algorithm [8]. Clustering is based on ϵ is maximum neighbour-hood distance and minPts is minimum number of points per cluster. Mathematically, clustering yields as $\text{DBSCAN}(P', \epsilon, \text{minPts}) \rightarrow C = \{C_1, C_2, \ldots, C_k\}$ where C_k represents a detected cluster.

The clustering process is illustrated in Figure 3, showing the progression from the noisy raw point-cloud, through voxelization, to clustering the points using DBSCAN, and finally to a 2D projection for analysis. To distinguish true tree trunks from other objects, each cluster's shape is evaluated by calculating its eccentricity by Principal Component Analysis (PCA) [9]. For each cluster C_K, the covariance matrix Σ is computed by Eq. (3), where $\bar{p}$ is the cluster centroid.

$$\Sigma = \frac{1}{|C_K|} \sum_{p_i \in C_k} (p_i - \bar{p})(p_i - \bar{p})^T \quad (3)$$

The covariance matrix Σ is decomposed to obtain two principal eigenvalues (λ_1, λ_2), where $\lambda_1 > \lambda_2$, and the eccentricity e is calculated using the ratio shown Eq. (4).

$$e = \frac{\lambda_1}{\lambda_2} \quad (4)$$



a. Raw point-cloud b. Voxelization

c. Clustering d. Clustered points in 2D

Figure 3 Point Cloud Clustering Pipeline.

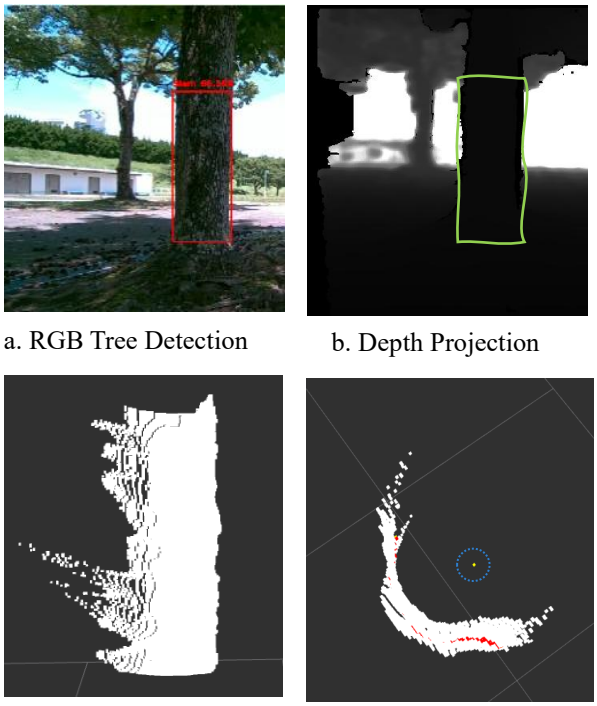
Clusters exhibiting eccentricity values between 10 and 20 are considered as potential tree candidates, as they reflect the elongated shape of partial tree trunks, while irregular or noisy groups are discarded as non-relevant structures.

3.4. *Depth-Assisted Image-Based Mapping (DAIM)*

The Depth-Assisted Image-Based Mapping (DAIM) approach integrates RGB image detection with depth information to localize tree trunks and estimate their geometric properties. Unlike traditional point cloud-only methods, DAIM begins with semantic detection of tree trunks using an image-based object detector YOLO [10] and then utilizes the corresponding region in the aligned depth map for precise 3D localization.

Figure 4 illustrates the processing pipeline for estimating the centre of a detected tree trunk using depth data. At any given instance, an RGB image of the forest scene is captured, and a tree trunk is detected using a YOLO-based object detector. A bounding box is then drawn to localize the trunk region with high confidence. The aligned depth region corresponding to the bounding box is extracted to isolate the depth values of the detected trunk (highlighted with a green box). These extracted depth values are converted into a 3D point cloud through clustering, forming a partial cylindrical surface of the tree trunk. From this 3D cluster, an arc segment near the base of the tree is selected, and three boundary points are extracted for geometric analysis. The arc is then fitted, and the center (yellow dot) and estimated radius (dashed blue circle) are calculated. To estimate the centre of a cylindrical tree trunk from a depth cluster obtained via the DAIM method, a geometric approach has been adopted using three non-collinear points $A(x_1,y_1)$ $C(x_2,y_2)$ $B(x_3,y_3)$ on the detected tree boundary as depicted in Figure 5. These points are selected from the base region of the 3D cluster, ideally forming a visible arc of the circular tree

cross-section. The algorithm involves computing the perpendicular bisectors of two-line segments AB and AC and finding their intersection point, which gives the centre $O(x_0,y_0)$ of the estimated circular trunk.



a. RGB Tree Detection b. Depth Projection

c. Cluster Extraction d. Center Estimation

Figure 4 Process of center estimation of depth points.

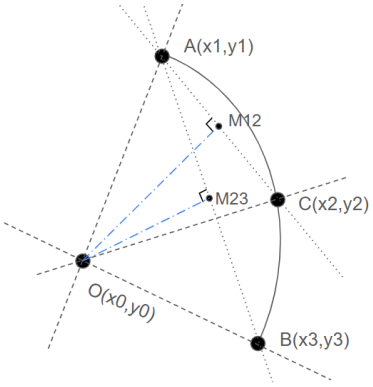


Figure 5 Tree Center Localization Based on Arc Geometry from Depth Points.

The steps begin with mid-points of segments AB and AC given by $M_{12} = \left((x_1 + x_2)/2, (y_1 + y_2)/2\right)$ and $M_{23} = \left((x_2 + x_3)/2, (y_2 + y_3)/2\right)$ respectively. Using these midpoints, the equations of the perpendicular bisectors can be written as Eq. (5) and Eq. (6)

$$y - y_{M_{12}} = m_{12}^{*}\,(x - x_{M_{12}}) \tag{5}$$

$$y - y_{M_{23}} = m_{23}^{*}\,(x - x_{M_{23}}) \tag{6}$$

here $m_{23}^{*} = -1/m_{23}$ and $m_{12}^{*} = -1/m_{12}$ represents the slopes of the perpendicular bisectors of segments AB and AC, respectively, where $m_{12}\ and\ m_{23}$ the slopes of the original segments. Solving Eq. (5) and Eq. (6) gives the center $O(x_0,y_0)$ represented by Eq. (7) and Eq. (8).

$$x_0 = \frac{(y_{M_{23}} - y_{M_{12}} + m_{12}^{*}\,x_{M_{12}} - m_{23}^{*}\,x_{M_{23}})}{/(m_{12}^{*} - m_{23}^{*})} \tag{7}$$

$$y_0 = m_{12}^{*}\left(x_0 - x_{M_{12}}\right) + y_{M_{12}} \tag{8}$$

3.5. *Sensor Fusion for Odometry Estimation*

Accurate and drift-resistant odometry is essential for reliable tree mapping and robot localization in unstructured outdoor environments. In this study, a sensor fusion strategy combining LiDAR, IMU, and wheel encoder data was adopted to improve the quality of pose estimation.

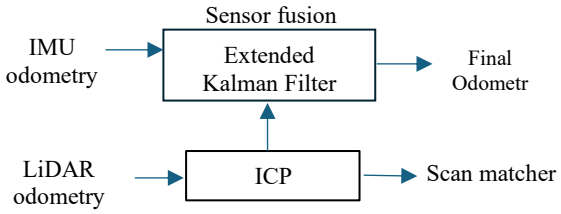


Figure 6 Sensor fusion pipeline for generating odometry

The odometry evaluation was conducted in a natural outdoor test area, measuring approximately 25 × 35 meters, with natural vegetation, scattered trees, and uneven terrain. As shown in Figure 6, the robot's final odometry was computed using an Extended Kalman Filter (EKF), which integrates odometry from a LiDAR-based scan matcher using the Iterative Closest Point algorithm (ICP) algorithm [11] and data from inertial measurements from the IMU.

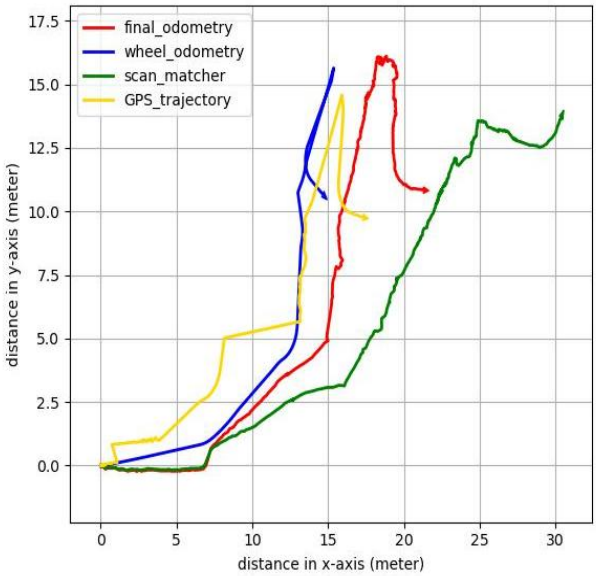


Figure 7 Robot path comparison based on different odometry sources.

In addition to the fused EKF output, wheel odometry and LiDAR scan matcher odometry are presented individually in Figure 7 as baseline references to illustrate the benefits of sensor fusion, although they were not used as standalone navigation sources. The final odometry (red) represents the fused EKF output, the scan matcher (green) corresponds to ICP-based LiDAR odometry, the wheel odometry (blue) reflects the trajectory estimated using only wheel encoders, and the GPS (yellow) indicates the reference trajectory recorded using an Emlid Reach RS with RTK GPS system. The final odometry aligns closely with the GPS path, indicating strong localization performance, particularly in sections where GPS data is available. The wheel odometry alone shows significant drift, especially during rotations and longer runs, while the scan matcher suffers from occasional misalignment due to partial visibility of features in the forested environment. This analysis confirms the advantage of multi-sensor fusion in outdoor field robotics, particularly in environments where no single sensor provides complete reliability. The final odometry output, driven by the EKF, was therefore used as the primary localization source for the tree mapping pipeline in subsequent sections.

3.6. Simultaneous Localization, Mapping, and Tree Position Refinement

Localization and tree mapping are performed simultaneously using the FastSLAM algorithm [12]. FastSLAM preserves information of many particles, each comprising a robot state hypothesis $S_t^{(k)}$ and a corresponding environmental map expressed as in Eq. (9)

$$S_t^{(k)} = \left\{state_t^{(k)}, map_t^{(k)}\right\} \quad (9)$$

As shown in Eq. (10), the robot's pose x_t is predicted using a motion model defined by a control function f(x,u), with ω_t capturing the uncertainty from system noise.

$$x_t = f(x_{t-1}, u_t) + \omega_t \quad (10)$$

Upon detecting landmarks (trees), observations are incorporated via the measurement model can be calculated as in Eq. (11).

$$z_t = h(x_t, t_i) + v_t \quad (11)$$

Here $h(x_t, t_i)$ maps the robot and tree states to predicted measurements, and v_t represents sensor noise. Particle weights are adjusted according to the consistency between predicted and observed landmarks. Initially detected tree positions are expressed relative to the LiDAR sensor frame as in Eq. (12) and to align them with the global odometry frame, a transformation sequence is applied represented in Eq. (13).

$$T_{velodyne} = \{t_i \mid t_i = (x_i, y_i, z_i) \in \mathbb{R}^3, \quad i=1...,M\} \quad (12)$$

$$T_{global\_odom} = T_{velodyne} \cdot T_{base\_link} \cdot T_{odom} \quad (13)$$

where T_{base_link} maps T_{base_link} the LiDAR frame to the robot base, and T_{odom} maps the base to the global frame. Transformed tree coordinates $t_i' = (x_i', y_i', z_i')$ are thus expressed in a consistent global frame for mapping and analysis. When the tree positions are transformed into the universal reference frame, a Kalman Filter [13] is applied to further refine these positions by mitigating measurement noise and sensor drift. Although global transformation aligns the tree positions spatially, minor fluctuations may still persist due to sensor inaccuracies. The Kalman Filter continuously updates the tree estimates over time, providing a more stable and reliable mapping output [14]. The final stage integrates the robot's path R_t and the refined tree positions T_{global} into a dynamic 2D map can be expressed as in Eq. (14) and Eq. (15).

$$R_t = \{r_j | r_j = (x_j, y_j) \in R^2, \ j = 1,.,T\} \quad (14)$$

$$T_{global} = \{t_i' | t_i' = (x_i', y_i',) \epsilon, i = 1, \dots, M\} \quad (15)$$

4. Experimental Results

The proposed tree detection and mapping algorithm was validated through experimental field tests. To assess the system's performance, key evaluation metrics—True Accepts (TA), True Rejects (TR), False Accepts (FA), and False Rejects (FR)—were calculated based on the methodology outlined in [15]. In this context, TA represents the number of trees correctly detected, TR denotes the correct identification of non-tree objects, FA indicates the number of non-tree items incorrectly classified as trees, and FR accounts for actual trees that were not detected.

Table 1 Parameters and necessary Values.

Parameter	Value
Particle count for FastSLAM (dimensionless)	100
Eccentricity cutoff value (unitless)	20
Voxel resolution for down-sampling (m)	0.01
Neighbourhood radius (ε) for DBSCAN (m)	0.25
Minimum point count per DBSCAN cluster	20
Proximity threshold for cluster association (m)	1.0
Standard deviation limit for cylinder fitting (m)	0.1

Table 1 presents the main parameters and thresholds tuned during experimentation. Notable settings include a voxel size of 0.01 m for point cloud down-sampling, and an eps value of 0.25 m for DBSCAN clustering, defining neighborhood proximity. The voxel size ensures a fine grid for reduced point density, while the clustering parameters enhance detection precision. Additionally, pass-through filters were applied with axis constraints: x (0 to 10 m), y (-10 to 10 m), and z (-0.5 to 0 m), targeting relevant environmental points. An eccentricity threshold of 20 was implemented to support the distinction between tree-like and non-tree objects. Figure 8 provides a comprehensive overview of the robot's navigation through the outdoor test field used for the experiment. The red triangle indicates the robot's dynamically updated position and orientation, while a gray line traces its traveled trajectory. Confirmed tree objects are represented by color-coded circles, visually correlating the robot's environment perception with the mapping output.

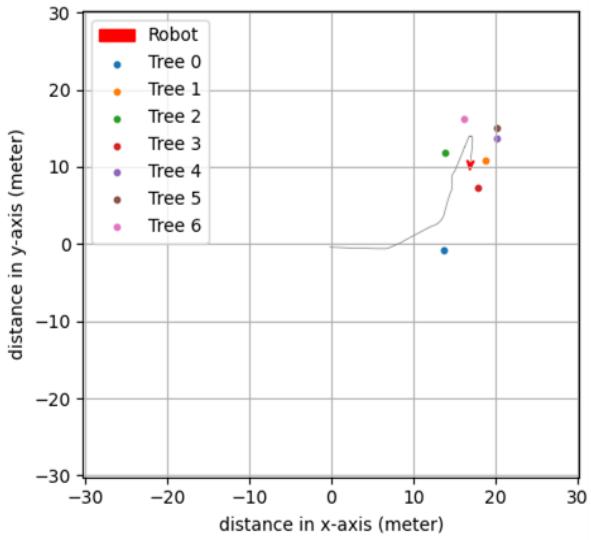


Figure 8 Real time estimated map through experiment using EB-LiDAR.

Detected tree information is further detailed in Figure 9, where each tree is identified by its X and Y coordinates and its corresponding eccentricity value. The spatial distribution of trees shows diverse positioning, ranging from approximately (13.6, -0.78) for "Tree 0" to (20.18, 14.98) for "Tree 5," indicating coverage across a wide area. The eccentricity values, calculated through Principal Component Analysis (PCA), describe the shape characteristics of the detected clusters: higher eccentricity values correspond to elongated, cylindrical forms typical of tree trunks. For instance, "Tree 3" exhibits an eccentricity of 16.46, indicating a highly elongated structure, whereas "Tree 2" shows a lower eccentricity of 10.00, suggesting a less pronounced elongation.

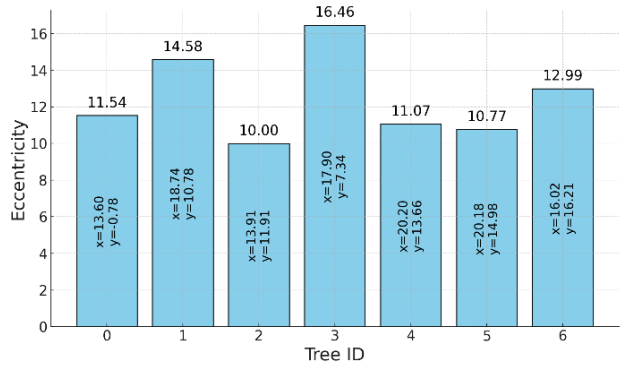


Figure 9 Eccentricity and Position of Detected Tree Clusters for EB-LiDAR.

This information is critical for distinguishing true tree trunks from other vertical structures in the environment, thus supporting more accurate mapping and environmental analysis in forestry applications. A visual depiction of detected tree distributions and inter-tree distances is shown in Figure 10 found from EB-LiDAR method. Labeled trees (Tree 0 to Tree 6) are marked with distinct colors, and connecting lines indicate measured distances between tree pairs. This visualization aids in validating the spatial accuracy of tree localization by comparing calculated distances with expected real-world configurations.

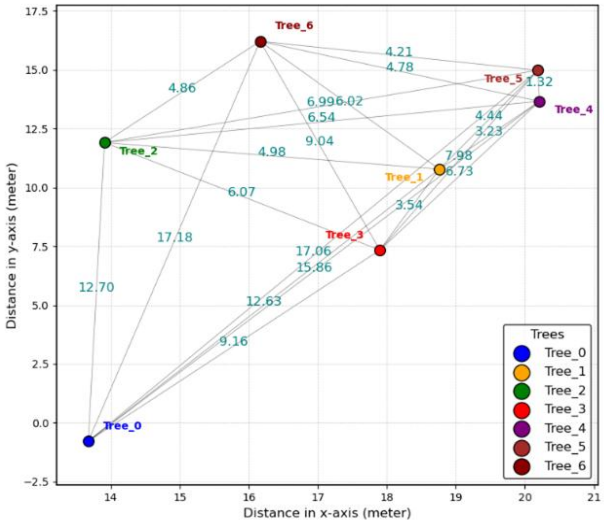


Figure 10 Distance plot of detected tree pairs in EB-LiDAR method.

Further accuracy analysis is illustrated in Figure 11, where a matrix plot displays percentage errors in distance estimations between tree pairs. The observed average error of 7.13% reflects high detection reliability, with minimal error across most pairs. However, slightly higher discrepancies, such as those between Tree 2 and Tree 6 (11.64%) and between Tree 2 and Tree 5 (10.38%), highlight areas where sensor noise, environmental occlusions, or real-time processing constraints may have influenced precision. Several contributing factors to these measurement discrepancies are identified.

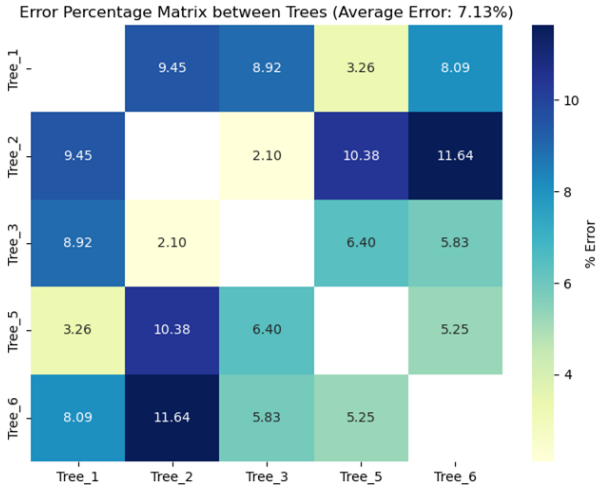


Figure 11 Matrix plot of errors between measured and estimated distance of EB-LiDAR.

LiDAR sensor noise, interference from dense foliage, partial occlusions, and terrain-induced misalignments can all impact point cloud data quality. Additionally, real-time system constraints often prioritize rapid processing over maximal precision. Despite these factors, the error rates observed remain within acceptable limits for general

forestry navigation tasks. To achieve higher precision, further enhancements to the filtering algorithms, sensor calibration, or multimodal sensing integration could be explored.

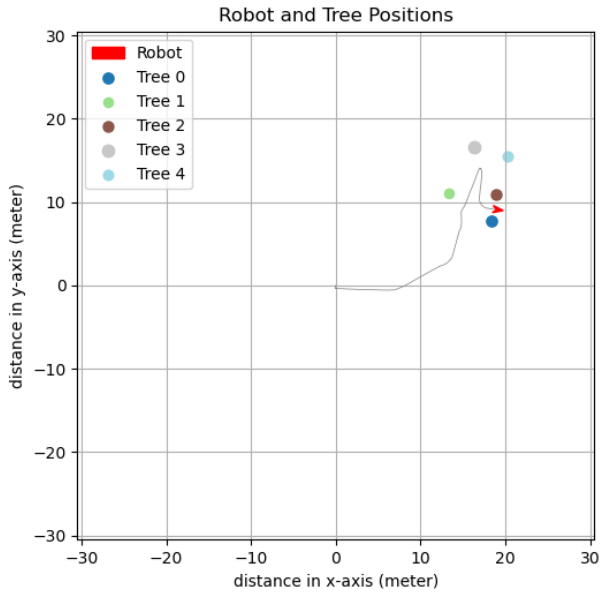


Figure 12 Real time estimated map through experiment using DAIM.

Figure 12 presents the spatial distribution of detected tree positions based on the DAIM method. In this scenario, five trees were identified and labeled from Tree 0 to Tree 4, alongside the robot's navigated trajectory in a 2D mapping space. Each tree is represented by a distinct color, and their positions are plotted with respect to the robot's path, indicated by a grey line and a red triangle denoting the robot's final position. Compared to EB-LiDAR method, DAIM shows fewer false positives and improved localization in certain regions.

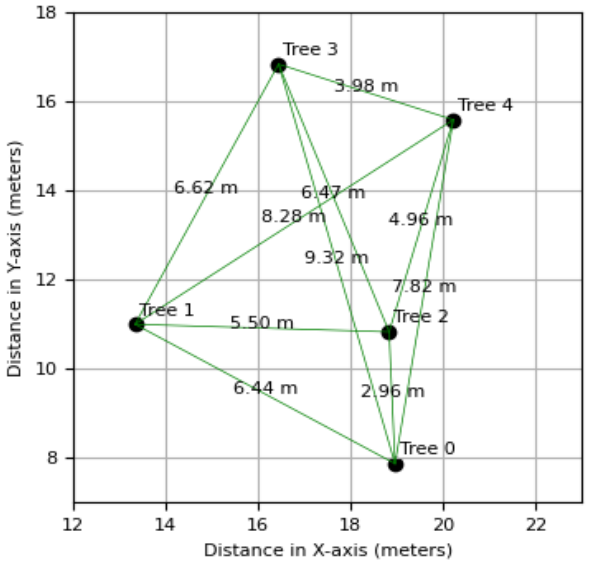


Figure 13 Distance plot of detected tree pairs in DAIM method.

To quantitatively evaluate the spatial accuracy of the DAIM method, an inter-tree distance comparison was performed. Visualization of tree positions and pairwise distances computed from detected trunk centers in DAIM method depicted in Figure 13. Each node represents a detected tree, and the green lines indicate the Euclidean distances between tree pairs. This spatial map aids in analyzing relative tree spacing, crucial for forest structure understanding and autonomous navigation planning.

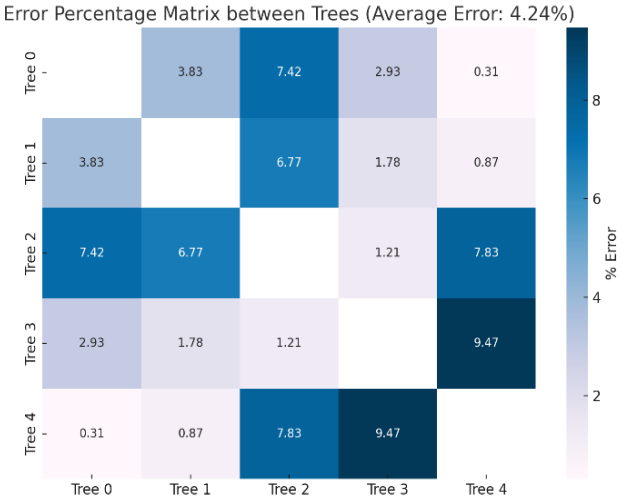


Figure 14 Matrix plot of errors between measured and estimated distance of DAIM.

Figure 14 shows the percentage error matrix, which illustrates the relative deviations in pairwise distances between trees when compared to the ground truth for DAIM method. Diagonal entries have been excluded as they represent zero self-distance. The color gradient in the matrix indicates the magnitude of error, with darker shades representing higher deviation. Most tree pairs show low to moderate percentage errors, with several values under 4%, demonstrating good consistency. The overall average error of 4.24% confirms the reliability of the this approach in estimating tree positions and spatial relationships in the environment. Additionally, due to independent clustering and indexing logic, the tree IDs assigned by each method differ even when referring to the same physical tree. For example, the tree detected as tree_1 in the EB-LiDAR method was labeled tree_2 in the DAIM output, and so on. To ensure fair and consistent evaluation of both methods, a unified ID system has been established based on spatial proximity between matched tree positions as tree IDs 1, 2, 3, 5, and 6 in the EB-LiDAR method correspond respectively to tree IDs 2, 1, 0, 4, and 3 in the DAIM method. Figure 15 illustrates the coordinate-wise comparison of tree positions estimated by the EB-LiDAR and DAIM methods, after aligning corresponding tree IDs based on spatial proximity. Each pair of bars represents the position estimated by the two methods for the same physical tree, labeled according to the unified tree IDs. The percentage annotations within each bar group represent the relative deviation between the EB-LiDAR and DAIM coordinate estimates, calculated as the percentage difference from the EB-LiDAR-based value. Notably, the

X-coordinate deviations range from 0.2% to 5.9%, while Y-coordinate deviations remain relatively low, ranging from 1.0% to 7.7%. These results indicate that both methods produce generally consistent tree localization, with small variations likely due to occlusion, resolution differences, or sensor-specific noise.

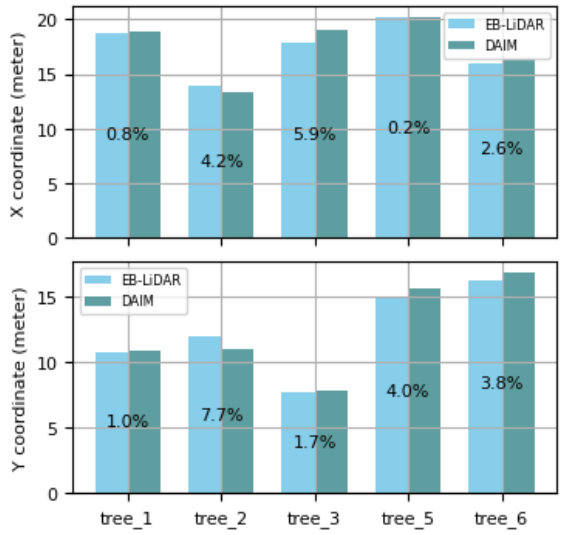


Figure 15 Comparison of Tree Coordinates from EB-LiDAR and DAIM Methods with Percentage Deviations.

The percentage annotations within each bar group represent the relative deviation between the EB-LiDAR and DAIM coordinate estimates, calculated as the percentage difference from the EB-LiDAR-based value. Notably, the X-coordinate deviations range from 0.2% to 5.9%, while Y-coordinate deviations remain relatively low, ranging from 1.0% to 7.7%. These results indicate that both methods produce generally consistent tree localization, with small variations likely due to occlusion, resolution differences, or sensor-specific noise.

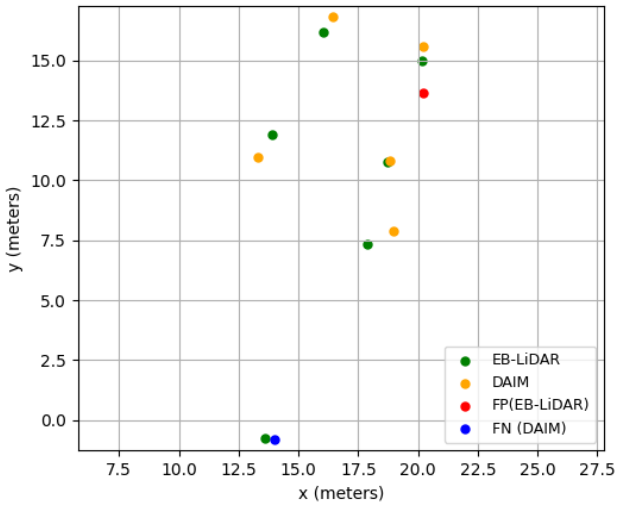


Figure 16 Comparison of Tree Detection Results for EB-LiDAR and DAIM Methods.

Following the quantitative evaluation of localization accuracy using positional error metrics, Figure 16 provides a visual comparison of the detected tree positions between the EB-LiDAR and DAIM methods. The figure plots the estimated tree coordinates from both approaches against each other in a common coordinate frame. Green and orange dots represent the tree positions detected using EB-LiDAR and DAIM, respectively. Additionally, red markers indicate false positives (FP) by EB-LiDAR—non-tree objects misidentified as trees—while blue markers highlight false negatives (FN) in DAIM—actual trees that were not detected. This visualization emphasizes the spatial consistency and discrepancies between the two methods, offering qualitative insight into each method's ability to correctly identify trees in the environment. It also reinforces earlier statistical findings, such as the DAIM method's lower average distance error but slightly higher rate of false negatives. Together, this figure supports a comprehensive comparison that integrates both numerical and spatial perspectives on detection reliability.

Table 2 Detection Statistics for EB-LiDAR and DAIM.

Method	N_{real}	N_{det}	TA	TN	FA	FR
EB-LiDAR	6	7	6	2	1	0
DAIM	6	5	5	3	0	1

Based on the evaluation metrics derived from experimental results in Table 2, EB-LiDAR successfully detected 6 out of 6 actual trees (TA), with 1 non-tree object falsely classified as a tree (FA) and no missed detections (FR). In contrast, DAIM detected 5 of the 6 real trees, missing one (FR), but achieved perfect precision with no false positives. In the experimental area, three non-tree objects were placed to test the false positive rejection, resulting in true negative (TN) values of 2 and 3 for the EB-LiDAR and DAIM methods, respectively. Consequently, after calculating the accuracy metrics, EB-LiDAR achieved an accuracy of 0.89, a precision of 0.86, a recall of 1.00, and an F1 score of 0.92. DAIM matched the overall accuracy at 0.89 but yielded a higher precision of 1.00, a lower recall of 0.83, and a slightly lower F1 score of 0.91. These results highlight the trade-offs between the two approaches—EB-LiDAR offers higher recall, while DAIM provides better precision—indicating that the choice of method can be tailored depending on whether minimizing false positives or false negatives is more critical for the given forestry application.

5. Conclusion

This research proposed and validated a comparative framework for real-time tree detection and mapping using two complementary methods: EB-LiDAR and DAIM. The EB-LiDAR method demonstrated efficient cylindrical object detection through shape-based filtering of LiDAR point clouds, while DAIM provided robust localization through image-based tree detection and depth-supported centre estimation. Both methods were deployed on an autonomous robot equipped with LiDAR, depth camera, and IMU, and evaluated through same real-world field experiments. Experimental results revealed that the EB-LiDAR approach achieved accurate tree localization with an average positional error of 7.13%, whereas DAIM further improved accuracy with a lower average error of

4.24%. Evaluation based on standard detection metrics showed that EB-LiDAR achieved an accuracy of 0.89, precision of 0.86, recall of 1.00, and F1 score of 0.92. In comparison, DAIM recorded the same accuracy of 0.89, but with a higher precision of 1.00, slightly lower recall of 0.83, and an F1 score of 0.91. These results underscore the reliability and complementary nature of the two methods for tree mapping in unstructured forest environments. A comparative analysis also showed that the EB-LiDAR method falsely identified one non-tree object (false positive), while DAIM missed one actual tree (false negative) due to limitations in visual detection. Moreover, although the depth camera provides a denser point cloud than LiDAR, the increased volume of data introduces additional filtering overhead and requires higher processing power, resulting in slower performance in DAIM method. Overall, this study highlights the complementary strengths of geometric and semantic sensing approaches, demonstrating that each method contributes uniquely to robust tree detection under varying environmental conditions. Future work may explore integrating both methods in a hybrid detection pipeline to enhance robustness under varying environmental conditions and to expand capabilities toward semantic classification of forest features.

References

1. C. J. Tan, S. Ogawa, T. Hayashi, T. Janthori, A. Tominaga, and E. Hayashi, “3D Semantic Mapping based on RGB-D Camera and LiDAR Sensor in Beach Environment,” *2024 1st International Conference on Robotics, Engineering, Science, and Technology, RESTCON 2024*, pp. 21–26, 2024.
2. Sebastian Thrun, Michael Montemerlo, Daphne Koller, Ben Wegbreit, Juan Nieto, and Eduardo Nebot, “FastSLAM: An Efficient Solution to the Simultaneous Localization And Mapping Problem with Unknown Data,” *Journal of Machine Learning Research*, pp. 1–48, 2004.
3. Sylvain Geiser, Sakmongkon Chumkamon, Ayumu Tominaga, Takumi Tomokawa, Eiji Hayashi, Online SLAM for Forestry Robot, Journal of Robotics, Networking and Artificial Life, vol-9-2, pp-177-182, 2022-2023.
4. Geiser Sylvain, Chumkamon Sakmongkon, Tominaga Ayumu, Tomokawa Takumi, Jie Tan Chi, and Hayashi Eiji, “Practical Implementation of FastSLAM for Forestry Robot,Proceedings of International Conference on Artificial Life and Robotics,” Feb. 2023, pp. 318–322.
5. S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, “Robot Operating System 2: Design, architecture, and uses in the wild,” *Sci Robot*, vol. 7, no. 66, May 2022.
6. Q.-Y. Zhou, J. Park, and V. Koltun, “Open3D: A Modern Library for 3D Data Processing,” *arXiv:1801.09847*, pp. 1–6, Jan. 2018.
7. Tan Chi Jie, “The BCRobo dataset for Robotic Vision and Autonomous Path Planning in Outdoor Beach Environment,” *ICAROB2023*, pp. 327–332, Feb. 2022.
8. D. Deng, “DBSCAN Clustering Algorithm Based on Density,” in *2020 7th International Forum on Electrical Engineering and Automation (IFEEA)*, Hefei, China, 2020, pp. 949–953.
9. A. Maćkiewicz and W. Ratajczak, “Principal components analysis (PCA),” *Comput Geosci*, vol. 19, no. 3, pp. 303–342, Mar. 1993.
10. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 779–788, Jun. 2015.
11. A. Censi, "An ICP variant using a point-to-line metric, *IEEE International Conference on Robotics and Automation*, pp. 19-25, Pasadena, CA, USA, 2008.
12. M. W. M. Gamini Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, “FastSLAM: a factored solution to the simultaneous localization and mapping problem,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, Jun. 2001.
13. R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Journal of Fluids Engineering, Transactions of the ASME*, vol. 82, no. 1, pp. 35–45, 1960.
14. M. A. Munjer, T. C. Jie, and E. Hayashi, “LiDAR-Enhanced Real-Time Tree Position Mapping for Forestry Robots,” *ICAROB2025*, pp. 318-325, Feb.13-16, 2025.
15. Alireza Baratloo, Mostafa Hosseini, Ahmed Negida, and Gehad El Ashal, “Part 1: Simple Definition and Calculation of Accuracy, Sensitivity and Specificity,” *Emerg (Tehran)*, vol. 3(2), p. 48, 2015.

Authors Introduction

M.A Munjer

He received his Masters of Science in Engineering from the Department of Electrical and Electronic Engineering, RUET, Bangladesh in 2019. He is currently a Doctoral student at Kyushu Institute of Technology and conducts research at Hayashi Laboratory.

Tan Chi Jie

He received his Masters of Science in Engineering from the Department of Computer Science and Systems Engineering, Kyushu Institute of Technology, Japan in 2023. He is currently a Doctoral student at Kyushu Institute of Technology and conducts research at Hayashi Laboratory.

Prof. Eiji Hayashi

Prof. Eiji Hayashi is a professor in the Department of Intelligent and Control Systems at Kyushu Institute of Technology. He received the Ph.D. (Dr. Eng.) degree from Waseda University in 1996. His research interests include Intelligent mechanics, Mechanical systems and Perceptual information processing. He is a member of The Institute of Electrical and Electronics Engineers (IEEE) and The Japan Society of Mechanical Engineers (JSME).