Fisheye Camera and LiDAR Fusion for Autonomous Following Vehicles

Jiahui Wan
School of Automotive Studies
Tongji University
Shanghai, China
2332936@tongji.edu.cn

Guangqiang Wu
School of Automotive Studies
Tongji University
Shanghai, China
wuguangqiang@tongji.edu.cn

***Abstract*—This paper presents a method that fuses LiDAR and fisheye camera data, aiming to address the current gap in research on the integration of these two sensors, with application to autonomous following vehicles. First, a joint calibration technique is employed to spatially align the LiDAR and fisheye camera, ensuring spatial consistency between the two sensors. Based on this, a target localization algorithm is developed through theoretical analysis. Experimental results show that the algorithm maintains a relative squared error within 10% across all tested distances and angles. Furthermore, even in complex scenarios involving foreground occlusions or background interference, the algorithm still achieves high localization accuracy, with results closely matching the ground truth. Additionally, compared to a pinhole camera, the fisheye camera provides a significantly wider field of view, further enhancing its applicability in dynamic and cluttered environments.**

Keywords—LiDAR-Fisheye Fusion, Target Localization, Autonomous Following Vehicle

I. Introduction

With the continuous advancement of technology, autonomous vehicles have gained significant attention, with many companies developing systems to assist humans in various tasks [1]. Among the core components, the perception module is crucial for interpreting multisensory data to support decision-making and planning [2]. Cameras and LiDAR are essential sensors in perception tasks, contributing to object detection and mapping [3]. While most research focuses on fusion between pinhole cameras and LiDAR, studies on the fusion of fisheye cameras and LiDAR remain limited. Fisheye cameras, offering low cost and wide field of view (FOV) [4], are increasingly important for intelligent vehicle applications, warranting further exploration.

This paper explores the fusion of fisheye camera and LiDAR in an autonomous following scenario, where the vehicle tracks a person's movement trajectory. For example, as illustrated in Fig. 1, an autonomous cleaning vehicle follows a person cleaning the road with a water jet and adjusts its motion by continuously tracking the target person's position. Therefore, obtaining the real-time 3D coordinates of the target person is crucial, and fusing image and point cloud data enables accurate real-time 3D localization of the target.

The key contributions of this paper are: (1) Fusion of fisheye camera and LiDAR data to achieve accurate spatial localization of the target person. (2) Application of this fusion method in a real-world autonomous following vehicle, validated through field experiments.

Fig. 1. One of the Applications of the Autonomous Following Vehicle — a Person-Guided Autonomous Cleaning Vehicle

II. Related Work

This paper depicts an outline of sensor applications and related research in fusion technologies.

A. Camera-Only

Pinhole cameras are widely used in autonomous systems for target detection and classification, offering features like color and texture [5], [6]. End-to-end models estimate depth and BEV features for semantic segmentation [7]. Yinhao Li et al. [8] improved depth estimation using voxel pooling. And Tesla's FSD processes 2300 frames per second for road safety [9]. However, fisheye camera applications are less explored. Benchmarks like F2BEV [10] and fisheye BEV segmentation [11] exist, but deep learning-based depth estimation is hardware-intensive and less suitable for compact systems like cleaning vehicles.

B. LiDAR-Only

LiDAR provides precise depth and environmental profiling [12], [13]. Early segmentation methods [14], and advanced models like DipG-Seg [15] and PointNet [16], improved 3D perception. However, lacking color and texture limits standalone performance.

C. Pinhole Camera and LiDAR Fusion

Combining pinhole camera texture with LiDAR depth enhances perception. Post-processing [17] and low-level geometric fusion [18] improve detection and distance estimation. Yet, integrating such fusion with fisheye cameras remains underexplored.

D. Fisheye Camera and LiDAR Fusion

This fusion remains limited in research. Yet, the fisheye camera's wide FOV and low cost, paired with LiDAR's range and accuracy, make this fusion promising for autonomous driving, especially in road cleaning vehicles.

III. Proposed Approach

A. Joint Calibration of LiDAR and Fisheye Camera

1) Intrinsic Calibration:

The Kannala-Brandt model [19] is used to perform intrinsic calibration of the fisheye camera (Fig. 2), aiming to map 3D coordinates of a point $P\ (x_c, y_c, z_c)$ in the fisheye camera coordinate system to its 2D pixel coordinates.

The specific formulas are as follows:

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \frac{x_c}{z_c} \\ \frac{y_c}{z_c} \end{bmatrix} \tag{1}$$

$$\theta = \arctan(\sqrt{a^2 + b^2}) \tag{2}$$

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8) \tag{3}$$

$$\begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} \frac{\theta_d}{\sqrt{a^2+b^2}} a \\ \frac{\theta_d}{\sqrt{a^2+b^2}} b \end{bmatrix} \tag{4}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c \\ d \\ 1 \end{bmatrix} \tag{5}$$

where k_1, k_2, k_3, k_4 are the distortion coefficients; f_x, f_y are the focal lengths; c_x, c_y are the coordinates of the fisheye camera's principal point.

2) Extrinsic Calibration:

Extrinsic calibration estimates the rotation matrix R and translation vector T that transform LiDAR coordinates (x_l, y_l, z_l) into fisheye camera coordinates (x_c, y_c, z_c), expressed as:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R \begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} + T \tag{6}$$

B. Localization of the Target Person's 3D Coordinates

1) Point Cloud Projection and ROI Filtering:

We use ROS's Approximate Time Synchronizer to align timestamps between LiDAR and fisheye camera data. For object detection, the YOLO series model [20] is applied to fisheye images, and the target person is selected from multiple detections using HSV and ORB feature matching. Let ROI denote the region of interest from detection, as shown in Fig. 3.

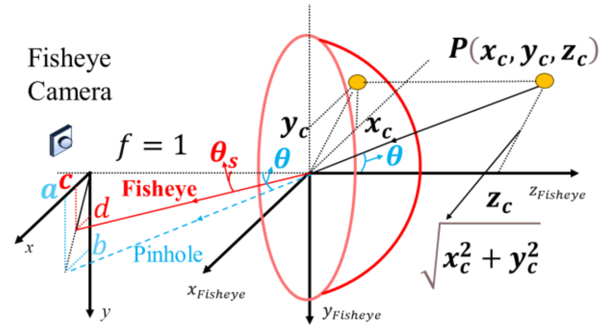

Fig. 2. Fisheye Camera Model

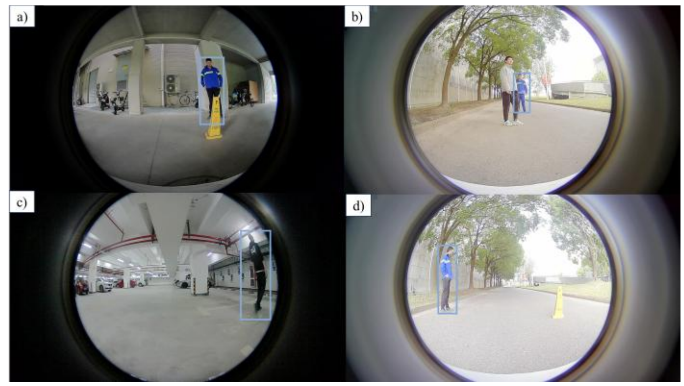

Fig. 3. ROI Illustration in Fisheye Camera View

Due to possible inclusion of non-target objects in the detection box, issues such as foreground occlusion and background interference may occur. Projecting the original point cloud onto the ROI (Fig. 3) produces results as in Fig. 4, revealing such interference.

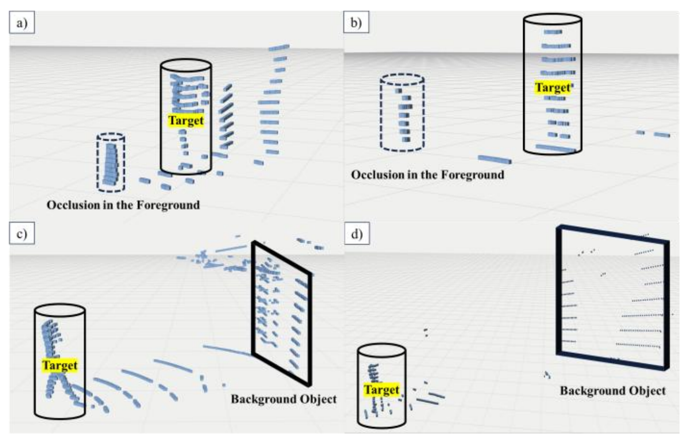

Fig. 4. Projection of Raw Point Cloud onto ROI in Fig. 3

To mitigate the errors introduced by the aforementioned issues, we shrink the bounding box edges inward by a distance Δd with the updated ROI' defined as:

$$ROI' = [x_{\min} + \Delta d, y_{\min}, x_{\max} - \Delta d, y_{\max}] \tag{7}$$

where $(x_{\min}, y_{\min})$ and $(x_{\max}, y_{\max})$ represent the top-left and bottom-right coordinates of the original bounding box ROI, respectively.

From Fig. 5, Δd is set to $0.15 \cdot (x_{\max} - x_{\min})$. In (a), the number of point clouds associated with occluding objects decreases from 273 to 176, a 35% reduction. Similarly, in (b), the number of background wall point clouds decreases from 1170 to 783, a 33% reduction.

On this basis, the LiDAR point cloud is subjected to projection transformation and filtering. The filtering process is described as follows:

$$P_{projected} = \{(X_i, Y_i, Z_i) \mid f(X_i, Y_i, Z_i) = p_i, p_i \in ROI'\} \tag{8}$$

Here, $P_{projected}$ is the set of all original 3D points that belong to the ROI' region., and $f(X_i, Y_i, Z_i) = p_i$ indicates that the 3D point (X_i, Y_i, Z_i) is mapped to the 2D pixel coordinates $p_i = (u_i, v_i)$ through the function f which is detailed in (1) to (5). ROI' is the adjusted target detection region after boundary refinement.

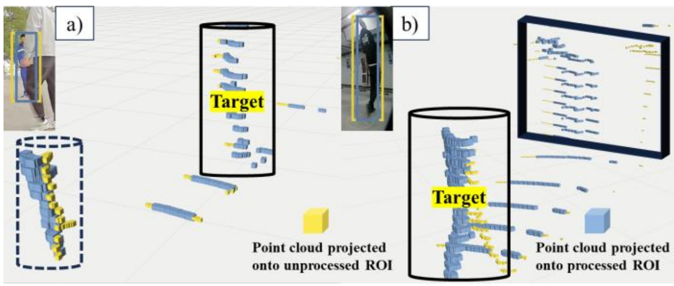

Fig. 5. Projection Results Before and After ROI Boundary Adjustment

2) Proof of Geometric Relationships:

When point $P_{projected}$ is transformed into the fisheye camera coordinate system using the extrinsic parameters, the transformed point $P'_{projected}$ is computed as:

$$P'_{projected} = RP_{projected} + T \tag{9}$$

Based on (1)-(5), it can be proven that in the fisheye camera system's xoz-plane, $P'_{projected}$ is enclosed by two rays originating from the origin (Fig. 6), and the expressions for these two rays are as follows:

$$z = \min\left(\frac{1}{a'_{1i}}\right)x \tag{10}$$

$$z = \min\left(\frac{1}{a'_{2i}}\right)x \tag{11}$$

where a'_{1i} and a'_{2i} are the parameters corresponding to the i-th pixel point on the left and right boundary segments J_1 and J_2 of ROI', respectively.

In practical applications, directly solving for these two boundaries is unnecessary. The above derivation only provides the theoretical basis for the subsequent localization algorithm.

3) Precise Localization of the Target Person:

Based on the theoretical analysis, this paper proposes a precise localization method. In the fisheye camera coordinate system, the xoz-plane is divided into concentric annular layers with a fixed interval Δr, starting from the fisheye camera origin, as shown in Fig. 7.

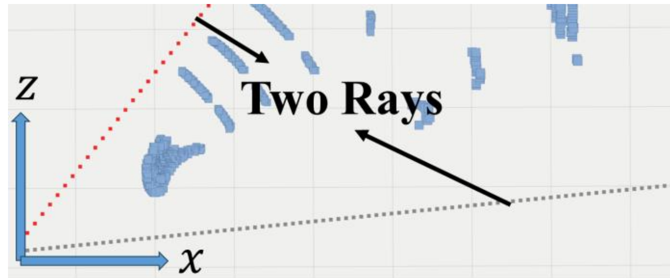

Fig. 6. Two Boundary Rays

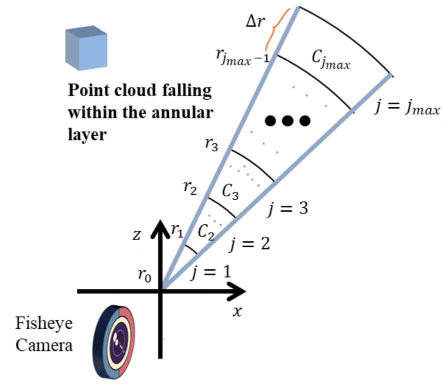

Fig. 7. Schematic of the Annular Division Method

For ease of description, this paper defines the z-axis of the camera coordinate system as pointing straight ahead along the optical axis, the x-axis as parallel to the imaging plane and horizontally to the right, and the y-axis as perpendicular to the $x-z$ plane and vertically upward.

The point cloud set $P'_{projected}$ is assigned to corresponding rings, with the j-th annular layer C_j defined as:

$$C_j = \{P_i \mid r_{j-1} \le \sqrt{x_i^2 + z_i^2} \le r_j\} \tag{12}$$

where $P_i = (x_i, y_i, z_i)$ is the i-th point in $P'_{projected}$, r_j and r_{j-1} are the radius of the j-th and the previous annular layer, respectively, and satisfy:

$$r_0 = 0,\ r_j = r_{j-1} + \Delta r \ \ for \ \ 1 \le j \le j_{\max} \tag{13}$$

Next, let $y_{\min,j}$ be the minimum y-value among all points in C_j. Points within a certain distance from $y_{\min,j}$ are selected to form the filtered point cloud set $C_{filter,j}$:

$$C_{filter,j} = \{P_i \in C_j \mid y_{\min,j} + y_0 \le y_i \le y_{\min,j} + y_1\} \tag{14}$$

where $P_i = (x_i, y_i, z_i)$ is the i-th point in C_j, y_0 and y_1 are the lower and upper bounds. This method effectively removes ground points, as shown in Fig. 8.

Next, as illustrated in Fig. 9, we proceed to the merging step:

$$C_{merge,i} = \bigcup_{j=f_i}^{j=f_i+\Delta j} C_{filter,j} \tag{15}$$

where $C_{merge,i}$ refers to the point cloud set of the i-th merged annular layer, and f_i is defined as:

$$f_1 = \arg\min_{j}(C_{filter,j} \neq \varnothing)$$
$$f_i = \arg\min_{j > f_{i-1}+\Delta j}(C_{filter,j} \neq \varnothing), i > 1 \tag{16}$$

where Δj is the number of layers to be merged, calculated as:

$$\Delta j = \left\lfloor \frac{D_{merge}}{\Delta r} \right\rfloor \tag{17}$$

where $\lfloor \bullet \rfloor$ represents the floor operation, and D_{merge} denotes the merging distance, which depends on the size of the target person.

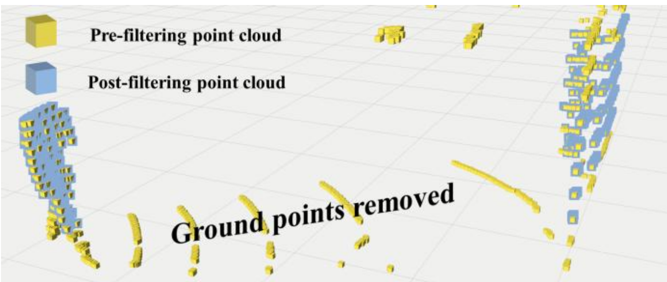

Fig. 8. Schematic Comparison of Point Cloud Before and After Filtering

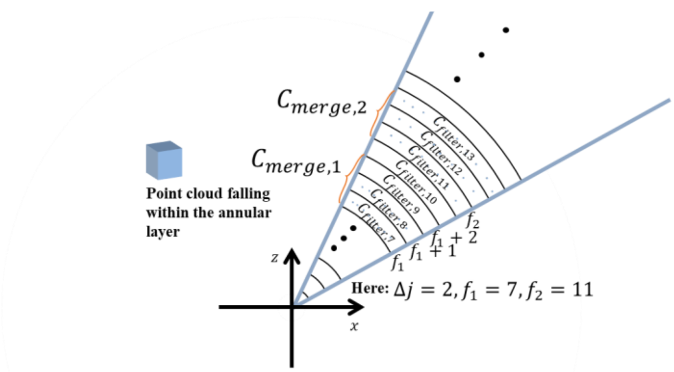

Fig. 9. Demonstration of Annular Layer Merging

Based on all the merged annular layer sets $C_{merge,i}$, the next step is to select the few merged annular layer sets closest to the camera origin for further evaluation:

$$C_{selected,i} = C_{merge,i}, 1 \le i \le \min(3, i_{\max}) \tag{18}$$

To identify the target person from multiple candidate sets, the paper proposes the following selection strategy, which is summarized in Algorithm 1.

Algorithm 1 Locate the position of the target person

Input: All selected point cloud sets $C_{selected,i}$.
Output: 3D coordinates of the target person P_{target} .
Initialisation: Parameters h, M, W

$C_{target} = \varnothing$
if $i_{\max} = 1$ **then**
  $C_{target} = C_{selected,1}$
else
  if $i_{\max} = 3$ **then**
    $i_1 = \arg\max_{i\in\{1,2,3\}}(|C_{selected,i}|)$
    $i_2 = \arg\max_{i\in\{1,2,3\}\setminus\{i_1\}}(|C_{selected,i}|)$
    $C_1 = C_{sekected,i_1}$
    $C_2 = C_{sekected,i_2}$
  end if
  $C_{max} = C_{\arg\max_{i\in\{1,2\}}|C_i|}$
  $C_{min} = C_{\arg\min_{i\in\{1,2\}}|C_i|}$
  if $|C_{max}| \ge h|C_{min}|$ **then**
    $C_{target} = C_{max}$
  else
    $L = MidPlane(ROI')$
    $\Delta_1 = DisWid(C_{max}, L)$
    $\Delta_2 = DisWid(C_{min}, L)$
    if $\Delta_1 \in W \,\&\, \Delta_2 \notin W$ **then**
      $C_{target} = C_{max}$
    else if $\Delta_2 \in W \,\&\, \Delta_1 \notin W$ **then**
      $C_{target} = C_{min}$
    end if
  end if
endif
$P_{target} = median(PlaneCluster(C_{target}))$
return P_{target}

We can see from Algorithm 1, the cases are divided into three scenarios: $i_{\max} = 1$, 2 , 3 . The ultimate goal is to determine which $C_{selected,i}$ corresponds to the target's point cloud, and assign it to C_{target} . Then, clustering is performed on C_{target} , and the median of the x , y , and z axis of the clustered point cloud set is taken as the target's 3D coordinates P_{target} . If $C_{target} = \varnothing$, it indicates that no target has been detected in this frame, and thus the frame is discarded and move to the next frame.

Here, h , M and W are parameters related to the laser beam count and target size. $|C|$ represents the number of point clouds in set C .

The function $MidPlane$ generates a central plane L based on the center pixel (u_0, v_0) of ROI' , with the expression as follows:

$$z - \frac{1}{a'_m}x = 0 \tag{19}$$

where a'_m can be solved by inversely using the intrinsic parameters of the fisheye camera.

The function $DisWid$ calculates the lateral width of the point cloud set relative to plane L :

$$\Delta = \max_{i=1,\dots,N}\left(\frac{Ax_i + Bz_i}{\sqrt{A^2+B^2}}\right) - \min_{i=1,\dots,N}\left(\frac{Ax_i + Bz_i}{\sqrt{A^2+B^2}}\right) \tag{20}$$

where $A = -1/a'_m$ and $B = 1$, x_i , z_i are the x , z -coordinate of the i -th point from the point cloud set C_{max} or C_{min} .

The function $PlaneCluster$ performs plane clustering on the point cloud, and the $median$ function calculates the median for each coordinate axis (x, y, z) of the point cloud set.

IV. Experimental Evaluation

To evaluate the localization algorithm, we conducted several experiments, using relative squared error as the performance metric:

$$\text{Relative Squared Error} = \frac{\sqrt{(x_{sf}-x_{gt})^2+(y_{sf}-y_{gt})^2+(z_{sf}-z_{gt})^2}}{\sqrt{x_{gt}^2+y_{gt}^2+z_{gt}^2}} \tag{21}$$

where (x_{sf}, y_{sf}, z_{sf}) is the sensor fusion coordinates, and (x_{gt}, y_{gt}, z_{gt}) is the ground truth.

Single Person Localization: We tested the localization performance of a single target person at various distances and angles, with the relevant visual results shown in Fig. 10, and the relative squared errors shown in Figs. 12 and 13.

The results indicate that, regardless of the test distance or angle, the relative squared error remains below 10%, demonstrating that the localization algorithm consistently provides high-precision results across different distances and angles.

Foreground Occlusion and Background Interference: To simulate complex environments, we tested the algorithm under foreground occlusion and background interference (Fig. 11). Despite these challenges, the algorithm maintained high accuracy, with results closely matching the ground truth.

Comparison between Fisheye Camera and Pinhole Camera: As shown in Fig. 14, under the same conditions using the same physical camera, the fisheye model provides a significantly wider field of view than the pinhole model. Fig. 15 further compares the relative squared error at different angles under identical conditions: the fisheye camera maintains a consistent error below 5%, while the pinhole model has a much narrower effective angular range.

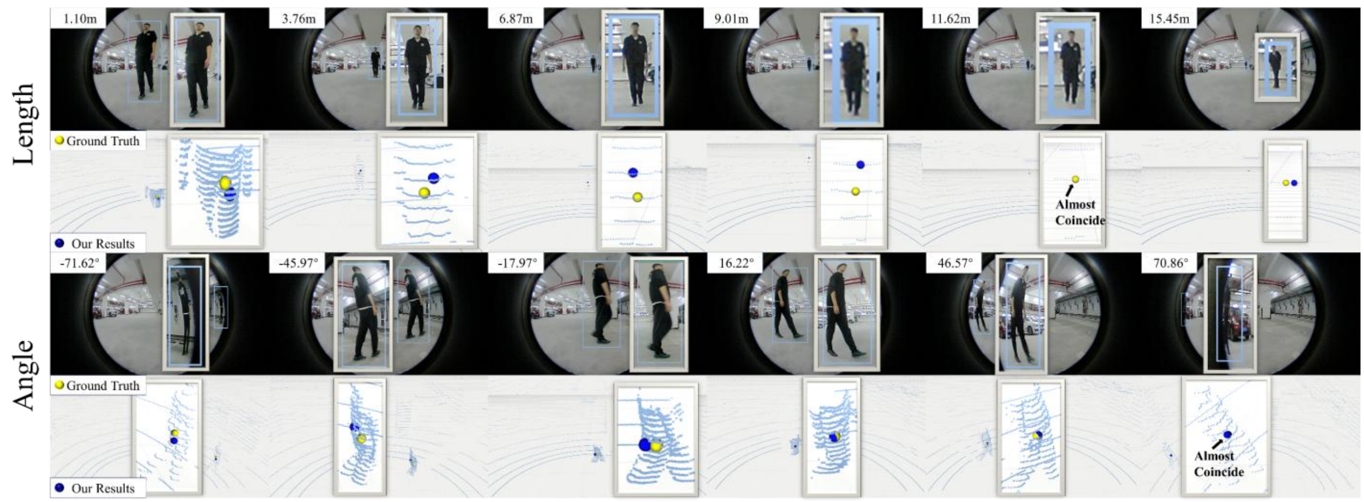

Fig. 10. Localization Performance of the Target Person at Different Distances and Angles. The upper part ("Length") presents the localization results at six different distances, while the lower part ("Angle") displays the localization results at six different angles.

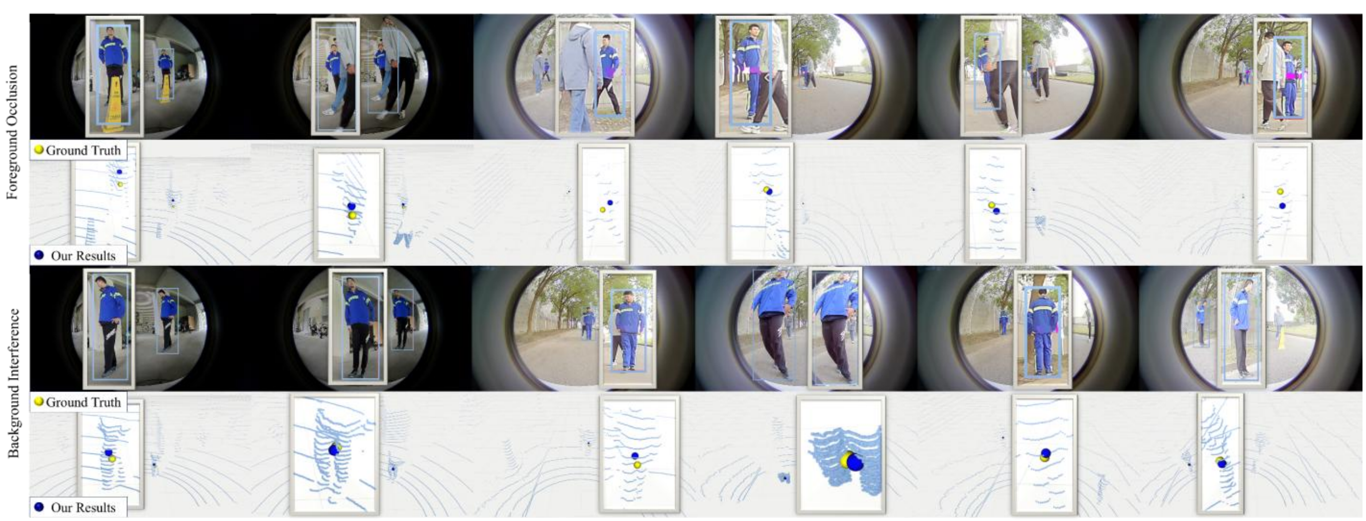

Fig. 11. Test of Foreground Occlusion and Background Interference. The upper part ("Foreground Occlusion") shows the localization results when obstructions or other persons are present in front of the target, while the lower part ("Background Interference") displays the localization results when interfering objects are located behind the target.

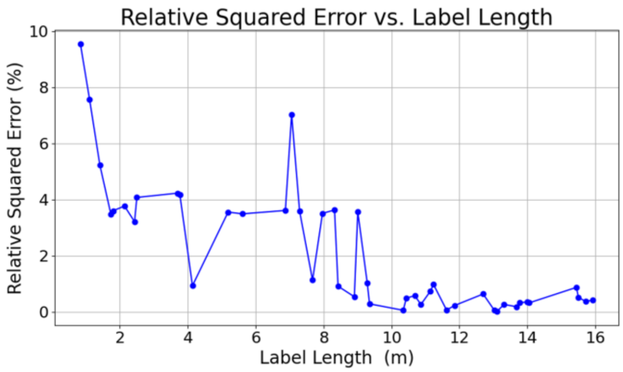

Fig. 12. Relative Squared Error at Different Distances

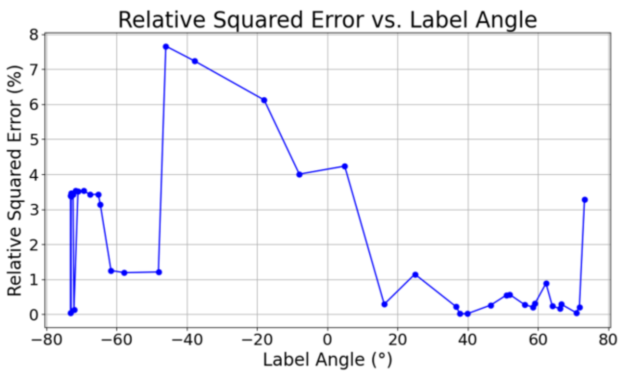

Fig. 13. Relative Squared Error at Different Angles

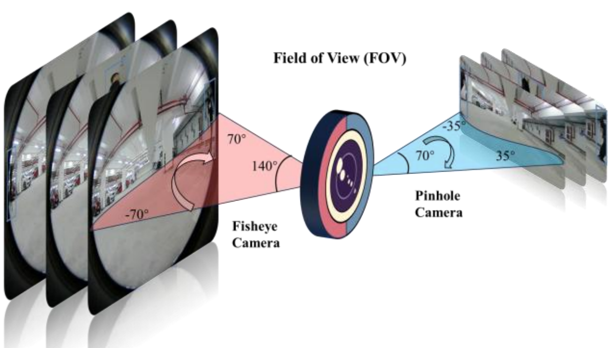

Fig. 14. Field of View: Fisheye Camera vs. Pinhole Camera

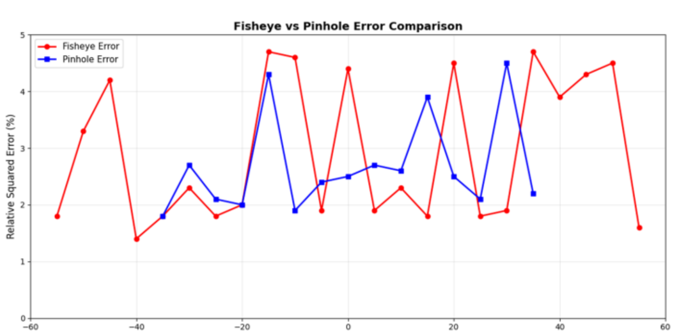

Fig. 15. Relative Squared Error at Different Distances: Fisheye Camera vs. Pinhole Camera

Acknowledgment

Thanks to the editors and all the reviewers for their valuable feedback and recommendations on this paper.

V. Conclusion

This paper presents a localization algorithm based on the fusion of a fisheye camera and LiDAR, which was validated on an autonomous cleaning vehicle. In the experiments, we tested various scenarios, including single-person localization, foreground occlusion, and background interference. The results demonstrate that the algorithm can accurately localize the target at varying distances and angles, and effectively handle occlusions and background interference. Additionally, this paper compares the field of view between the fisheye camera and the pinhole camera, showing that the fisheye model provides a significantly wider coverage.

Overall, the fusion of the fisheye camera and LiDAR performs excellently in localization within complex environments. This approach has significant potential for applications in autonomous driving, robotics, and other intelligent systems, greatly enhancing the system's perception and decision-making capabilities.

References

[1] R. Ayala and T. K. Mohd, "Sensors in Autonomous Vehicles: A Survey," Journal of Autonomous Vehicles and Systems, vol. 1, no. 3, p. 031003, Jul. 2021, doi: 10.1115/1.4052991.

[2] C. Sun et al., "Toward Ensuring Safety for Autonomous Driving Perception: Standardization Progress, Research Advances, and Perspectives," IEEE Trans. Intell. Transport. Syst., vol. 25, no. 5, pp. 3286–3304, May 2024, doi: 10.1109/TITS.2023.3321309.

[3] H. A. Ignatious, H.-E.- Sayed, and M. Khan, "An overview of sensors in Autonomous Vehicles," Procedia Computer Science, vol. 198, pp. 736–741, 2022, doi: 10.1016/j.procs.2021.12.315.

[4] Y. Qian, M. Yang, and J. M. Dolan, "Survey on Fish-Eye Cameras and Their Applications in Intelligent Vehicles," IEEE Trans. Intell. Transport. Syst., vol. 23, no. 12, pp. 22755–22771, Dec. 2022, doi: 10.1109/TITS.2022.3210409.

[5] S. Campbell et al., "Sensor Technology in Autonomous Vehicle : A review," in 2018 29th Irish Signals and Systems Conference (ISSC), Belfast: IEEE, Jun. 2018, pp. 1–4. doi: 10.1109/ISSC.2018.8585340.

[6] C. Cronin, A. Conway, and J. Walsh, "State-of-the-Art Review of Autonomous Intelligent Vehicles (AIV) Technologies for the Automotive and Manufacturing Industry," in 2019 30th Irish Signals and Systems Conference (ISSC), Maynooth, Ireland: IEEE, Jun. 2019, pp. 1–6. doi: 10.1109/ISSC.2019.8904920.

[7] J. Philion and S. Fidler, "Lift, Splat, Shoot: Encoding Images From Arbitrary Camera Rigs by Implicitly Unprojecting to 3D," Aug. 13, 2020, arXiv: arXiv:2008.05711. Accessed: Nov. 20, 2024. [Online]. Available: http://arxiv.org/abs/2008.05711.

[8] Y. Li et al., "BEVDepth: Acquisition of Reliable Depth for Multi-View 3D Object Detection," AAAI, vol. 37, no. 2, pp. 1477–1485, Jun. 2023, doi: 10.1609/aaai.v37i2.25233.

[9] E. Talpes et al., "Compute Solution for Tesla's Full Self-Driving Computer," IEEE Micro, vol. 40, no. 2, pp. 25–35, Mar. 2020, doi: 10.1109/MM.2020.2975764.

[10] E. U. Samani, F. Tao, H. R. Dasari, S. Ding, and A. G. Banerjee, "F2BEV: Bird's Eye View Generation from Surround-View Fisheye Camera Images for Automated Driving," Aug. 01, 2023, arXiv: arXiv:2303.03651. Accessed: Nov. 20, 2024. [Online]. Available: http://arxiv.org/abs/2303.03651.

[11] S. Yogamani, D. Unger, V. Narayanan, and V. R. Kumar, "FisheyeBEVSeg: Surround View Fisheye Cameras based Bird's-Eye View Segmentation for Autonomous Driving," in 2024 IEEE/CVF

Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA: IEEE, Jun. 2024, pp. 1331–1334. doi: 10.1109/CVPRW63382.2024.00140.

[12] S. Mou, Y. Chang, W. Wang, and D. Zhao, "An Optimal LiDAR Configuration Approach for Self-Driving Cars," May 20, 2018, arXiv: arXiv:1805.07843. Accessed: Nov. 20, 2024. [Online]. Available: http://arxiv.org/abs/1805.07843.

[13] N. Li et al., "A Progress Review on Solid - State LiDAR and Nanophotonics - Based LiDAR Sensors," Laser & Photonics Reviews, vol. 16, no. 11, p. 2100511, Nov. 2022, doi: 10.1002/lpor.202100511.

[14] M. Himmelsbach, F. V. Hundelshausen, and H.-J. Wuensche, "Fast segmentation of 3D point clouds for ground vehicles," in 2010 IEEE Intelligent Vehicles Symposium, La Jolla, CA, USA: IEEE, Jun. 2010, pp. 560–565. doi: 10.1109/IVS.2010.5548059.

[15] H. Wen, S. Liu, Y. Liu, and C. Liu, "DipG-Seg: Fast and Accurate Double Image-Based Pixel-Wise Ground Segmentation," IEEE Trans. Intell. Transport. Syst., vol. 25, no. 6, pp. 5189–5200, Jun. 2024, doi: 10.1109/TITS.2023.3339334.

[16] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI: IEEE, Jul. 2017, pp. 77–85. doi: 10.1109/CVPR.2017.16.

[17] Lee, Jongseo, Mangyu Kim, and Hakil Kim. "Camera and LiDAR sensor fusion for improving object detection." Journal of Broadcast Engineering 24.4 (2019): 580-591.

[18] G. A. Kumar, J. H. Lee, J. Hwang, J. Park, S. H. Youn, and S. Kwon, "LiDAR and Camera Fusion Approach for Object Distance Estimation in Self-Driving Vehicles," Symmetry, vol. 12, no. 2, p. 324, Feb. 2020, doi: 10.3390/sym12020324.

[19] J. Kannala and S. S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 8, pp. 1335–1340, Aug. 2006, doi: 10.1109/TPAMI.2006.153.

[20] M. Hussain, "YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection," Machines, vol. 11, no. 7, p. 677, Jun. 2023, doi: 10.3390/machines11070677.