Visual-Marker-Based Localization for Flat-Variation Scene

Bohuan Xue, *Graduate Student Member, IEEE*, Xiaoyang Yan, Jin Wu, *Member, IEEE*, Jintao Cheng, Jianhao Jiao, *Member, IEEE*, Haoxuan Jiang, Rui Fan, *Senior Member, IEEE*, Ming Liu, *Senior Member, IEEE*, and Chengxi Zhang, *Member, IEEE*

***Abstract*— Localization, an indispensable component in robotics and automation, encounters difficulties arising from appearance variations, resulting in inaccurate data associations. Semantic-based positioning mitigates these challenges by filtering out invalid data, such as moving vehicles and worn road markings. Building on this insight, we introduce a robust semantic visual localization system, which has been successfully deployed in real-world settings. The system uses neural networks to extract road markers and associate data with a semantic map. To enhance system reliability, we use several data filters. These filters remove images that are prone to misrecognition or poorly processed by neural networks. We propose two techniques for vehicle state estimation. The first, using the inverse perspective mapping (IPM) matrix, directly determines the vehicle's central pose. The second technique derives the camera pose using the perspective-n-points (PnP) method and leverages external parameters to infer the vehicle's central state. The wheel encoder, with its robust anti-noise capability, offers odometry in the absence of semantic information, enhancing the system's resilience. The crux of our approach lies in distinct semantic strategies: using lane lines for orientation and road markers exclusively for translation estimation. We also detail an automatic construction method for the semantic map, enhancing the system's practicality. Experimental results indicate that the IPM method outperforms the PnP approach, leading to notably improved positioning accuracy. In addition, the error distribution of the IPM method more closely aligns with a normal distribution compared with that of the PnP approach.**

***Index Terms*— Inverse perspective mapping (IPM), normal distribution, perspective-n-points (PnP), semantic visual localization.**

Manuscript received 30 December 2023; accepted 2 February 2024. Date of publication 1 March 2024; date of current version 15 March 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62233013, in part by the Science and Technology Commission of Shanghai Municipal under Grant 22511104500, in part by the Fundamental Research Funds for the Central Universities, and in part by the Xiaomi Young Talents Program. The Associate Editor coordinating the review process was Dr. Lihui Peng. *(Corresponding author: Chengxi Zhang.)*

Bohuan Xue is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, SAR, China (e-mail: bxueaa@connect.ust.hk).

Xiaoyang Yan and Jin Wu are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong, SAR, China (e-mail: xyanaq@connect.ust.hk; jin_wu_uestc@hotmail.com).

Jintao Cheng is with the School of Electronics and Information Engineering, South China Normal University, Foshan 528200, China (e-mail: chengjt_alex@outlook.com).

Jianhao Jiao is with the Department of Computer Science, University College London, WC1E 6BT London, U.K. (e-mail: jiaojh1994@gmail.com).

Haoxuan Jiang and Ming Liu are with the Robotics and Autonomous Systems Thrust in Systems Hub, Hong Kong University of Science and Technology (Guangzhou), Nansha, Guangzhou, Guangdong 511453, China (e-mail: hjiangax@connect.hkust-gz.edu.cn; eelium@hkust-gz.edu.cn).

Rui Fan is with the College of Electronics and Information Engineering, Shanghai Research Institute for Intelligent Autonomous Systems, the State Key Laboratory of Intelligent Autonomous Systems, and the Frontiers Science Center for Intelligent Autonomous Systems, Tongji University, Shanghai 201804, China (e-mail: rui.fan@ieee.org).

Chengxi Zhang is with the School of Internet of Things, Jiangnan University, Wuxi 214122, China (e-mail: dongfangxy@163.com).

Digital Object Identifier 10.1109/TIM.2024.3372231

I. Introduction

A. Motivation

SHIPPING remains a linchpin of international trade, serving as the most cost-effective transportation mode [1]. Specifically within port operations, unmanned ground vehicles (UGVs) stand to substantially enhance efficiency in container transport [2], with a particular focus on addressing the unique logistical challenges of these environments. The advent of COVID-19 has exacerbated the existing labor shortages in port operations, with a notable decline in available dockers and truck drivers [3]. This period has also seen a surge in aggressive driving behaviors [4], further underscoring the need for the development and deployment of UGVs specifically designed for the complex and demanding conditions of ports.

This scenario presents a significant challenge for autonomous driving in large-scale, appearance-changing environments [2], wherein determining the precise position and orientation of the UGV becomes crucial.

In the field of robotics, localization schemes can generally be divided into two types: map-based localization and nonmap-based localization. The latter is also called simultaneous localization and mapping (SLAM). In practice, to maintain the consistency of positioning results, it is often necessary for nonmap localization methods to provide the correct initial coordinate system.

In extreme scenarios such as sensor failure or localization failure in industrial applications, multisensor fusion based on the Kalman filter (KF) can sometimes offer greater robustness compared with the tight coupling of multiple sensors. Therefore, it is also crucial that the algorithm can output accurate results accompanied by a measure of uncertainty. The system also needs to have a strong self-checking ability to eliminate the wrong data association, thus increasing the system's robustness.

The primary sensors for the localization system include LiDAR, cameras, and global navigation satellite system (GNSS). In scenarios such as traffic jams, the functionality of LiDAR becomes limited, potentially leading to system failures. Moreover, the GNSS signal may require enhancements in certain areas to provide effective positioning results.

Although cameras may not perform optimally during nighttime, this limitation can be mitigated by ensuring sufficient lighting. Given that illuminating the surroundings is a feasible solution in most operational scenarios, a camera-based robust positioning system can be swiftly deployed for industrial applications.

B. Challenges

Our foremost priority in UGV localization is to achieve a harmonious balance between precise localization results and optimal real-time performance. In scenarios where the localization accuracy falls below the requisite threshold or the system encounters substantial delays, it fails to furnish reliable feedback data, which are vital for the UGV's ensuing planning and control processes. Based on these two objectives, the main challenges are as follows.

1) Appearance Changing: Methods using feature points, as delineated in [5] and [6], may encounter performance degradation in dynamic environments. Upon halting, the UGV inadvertently captures features of moving objects, incorporating them into camera pose estimation, a factor that can potentially destabilize the system. As the UGV enters expansive areas characterized mainly by the ground as a notable feature, GNSSs generally operate optimally, barring the presence of overhead metal obstructions. However, the presence of structures such as rail cranes along the port's seaside can introduce interference with the GNSS signal. LiDAR-based methods [7] may encounter difficulties due to insufficient constraints in such environments. Although solutions integrating LiDAR and GNSS have been proposed as in [8] and [9], vehicle congestion beneath the gantry cranes in port scenarios might lead to simultaneous failures of both LiDAR and GNSS. In most scenarios, marker-based methods that use ground features remain viable, using cameras to facilitate successful operation. However, dynamic environments encompass not only the observed scenes but also potential alterations to the markers upon which the method relies. Such markers, encompassing varieties such as rhombus or AprilTags, may be compromised due to staining in industrial settings.

2) Data Association: In map-based localization methodologies, a critical step is ensuring precise data association between the sensor data and the map. This task faces significant challenges in maintaining accurate data associations, given that any inaccuracies can severely compromise the system, resulting in incorrect location estimations. Within the context of SLAM techniques, loop closure functions as a vital mechanism to mitigate cumulative errors, necessitating meticulous data association between sensor inputs and map details. Consequently, it is imperative to develop a sophisticated mechanism capable of identifying outliers during data association, thereby enhancing the system's reliability and precision. Moreover, minor fluctuations in the camera's internal parameters and distortion coefficients can alter the pixel positions of features within an image, necessitating a system equipped with robust noise resistance capabilities.

3) Calculate Resource Utilization: The central processing unit (CPU) allocates time slots for the task of visual localization and processes data from various other sensors. In addition, tasks pertaining to planning, control, and other auxiliary processes consume a significant portion of the CPU's processing time. A more efficient method implies reduced occupation of CPU time slots, thereby facilitating the provision of more real-time data for subsequent processes reliant on the outcomes of visual localization. The graphics processing unit (GPU) can be exclusively allocated to the visual module, obviating the need to account for potential allocation of GPU computing resources to other processes. Therefore, our primary focus is centered on enhancing the computational efficiency of the CPU.

4) Output Uncertainty: Given that the final result may be integrated with positioning data from other sensors, it is essential to provide a reliable or easily adjustable measure of uncertainty.

C. Related Work

In the quest for technological advancements in UGV localization, visual localization methods have emerged as a focal point due to their intuitive perception of the environment. These methods are commonly distinguished by whether or not they use a preexisting map.

1) Visual Localization Without a Prior Map: This challenge is akin to the SLAM problem, as delineated in [10]. A single camera is incapable of capturing depth information, which consequently impedes the acquisition of accurate metric scales necessary for positioning results [11], [12]. Typically, stereo cameras are used to mitigate the metric scale issue, as illustrated in studies such as [13], [14], [15], and [16]. However, the accuracy of the stereo system is constrained by the baseline length, the distance between the two cameras [17]. Moreover, additional calibration work is required to determine the external parameters between the cameras, which is critical for accurate depth estimation. This limitation primarily affects the system's ability to accurately determine depth information, which is derived from the disparity between the corresponding points in the images captured by the two cameras. To address the aforementioned challenges, several studies, such as [18] and [19], have proposed the use of RGB-D cameras for visual localization.

While RGB-D cameras offer an effective solution for visual localization, they are prone to infrared interference in bright outdoor environments, leading to inaccuracies in depth information. Moreover, the limited range of their depth detection can affect their performance in expansive or complex terrains. These challenges have propelled researchers to explore alternative technical solutions, particularly methods that combine different sensors. This strategy of integrating multiple sensors, commonly referred to as "sensor fusion," aims to leverage the strengths of each sensor to compensate for their individual limitations.

With an inertial measurement unit (IMU), a monocular camera can form a basic sensor suite for six-degree-of-freedom (6-DOF) state estimation [5]. Mainstream visual–inertial (VI) sensor fusion approaches include the KF [20], [21], [22] and graph-optimization-based algorithms [5], [23]. Similar to the stereo approach, the external parameters between the camera and the IMU require calibration. This process, including the

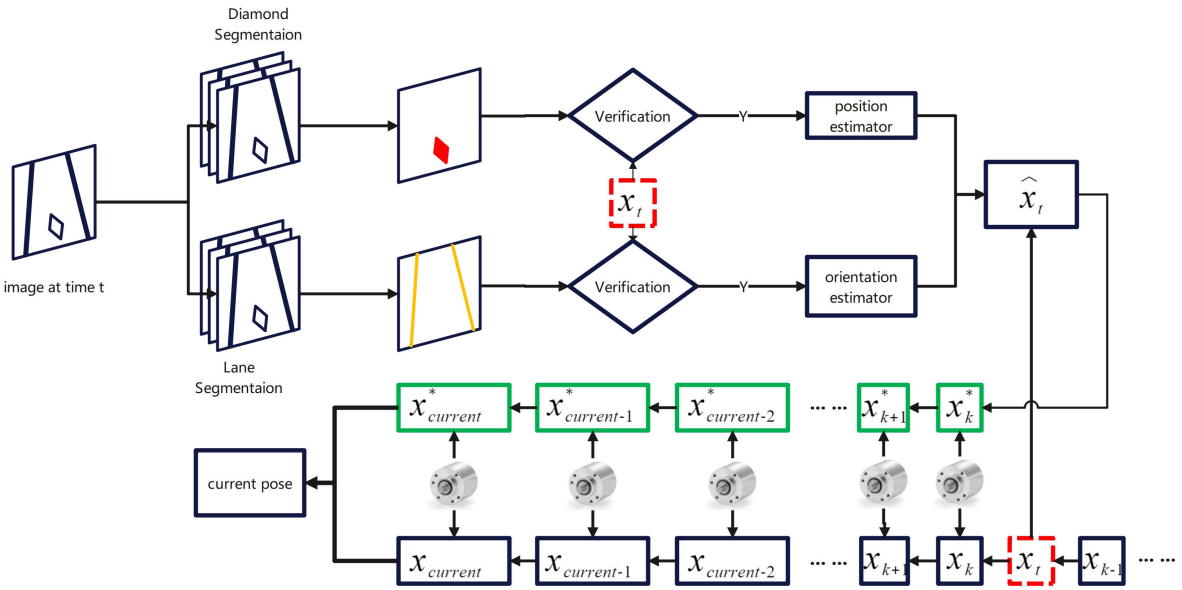


Fig. 1. Block diagram illustrating the complete pipeline of our proposed localization system. The system begins with an initial pose $\boldsymbol{x}$ and estimates vehicle status using wheel odometry, as explained in Section II-F. When image data are available, they undergo processing through two distinct networks, enabling estimation of both orientation and position, as detailed in Section II-D.

necessary IMU parameter adjustments, often contributes to increased system instability. In addition, these methods are unable to adapt to varying scenes.

In addition, there exist other multisensor fusion frameworks. For instance, V-LOAM [24] stands as a state-of-the-art (SOTA) method in the realm of camera and LiDAR fusion, consistently ranking at the forefront of the KITTI dataset leaderboard. The approach in Yin et al. [25] introduces an online photometric calibration module to mitigate photometric disturbances in real-world applications, thereby enhancing the overall system's localization robustness. Distinct from others, the Lidar–inertial–visual SLAM system [26], [27], notably R^{3}LIVE [28], serves not only as a LiDAR–inertial–visual state estimator but also possesses the capability to reconstruct the radiance map dynamically. GVINS [29] exemplifies a tightly coupled GNSS–visual–inertial system. The framework can operate effectively in larger scale environments, benefiting from the assistance of the GNSS.

These multisensor fusion approaches presuppose the reliable functioning of all the sensors. However, sensor failures are not uncommon in practical applications. Moreover, these methods struggle to address scenarios involving moving objects within the environment.

While traditional approaches have been foundational to visual localization, the emergence of deep learning has significantly transformed the field through the introduction of novel techniques. A notable representative is iMAP [30], which is the pioneering work that integrates NeRF [31] as the map representation within an SLAM framework. iMAP optimizes the camera pose through backpropagation using the photometric loss derived from NeRF. However, it uses a single multilayer perceptron (MLP), which is not well-suited for large-scale environments. NICE-SLAM [32] overcomes this limitation by adopting multiple MLPs in place of a single one, using a hierarchical scene representation for more efficient management of expansive environments. Despite this improvement, the geometric structure representation remains imperfect. Vox-Fusion mitigates this by storing voxel grids in an octree and leveraging signed distance fields (SDFs) for a more accurate geometric fit of the scene. However, Vox-Fusion [33] compromises its capacity for synthesizing new viewpoints. A significant limitation of these NeRF-based techniques is their dependency on depth information, hindering their direct use with standard monocular cameras. Orbeez-SLAM [34], which combines the ORB-SLAM2 [14] framework with instant-ngp [35], operates independently of depth information and is capable of pretraining-free operation in novel scenes. However, its memory utilization increases with map size, and it struggles in environments with dynamic objects. GO-SLAM [36] incorporates loop closure detection and global bundle adjustment (BA) in NeRF-based SLAM to mitigate cumulative errors. However, its substantial GPU memory requirements (exceeding 16 GB) and low operating frequency (below 10 frames/s) present challenges for industrial deployment.

Apart from SLAM methodologies using NeRF, a notable contemporary advancement is DROID-SLAM [37]. Using an advanced dense optical flow estimation architecture and iterative updates through a dense BA layer, it facilitates robust visual SLAM. This technique has laid the groundwork for later developments [38], [39]. PVO [39] bolsters resilience in dynamic environments through the integration of three modules: image panoptic segmentation, panoptic-enhanced VO module, and VO-enhanced VPS module.

Nevertheless, these learning-based methods exhibit a significant limitation: their extensive GPU memory requirements impede implementation on industrial robotic systems. Moreover, they do not incorporate a framework for quantifying uncertainty, an essential factor for integrating data with onboard sensors. The efficacy of these algorithms is also

highly contingent on the accuracy of camera intrinsic calibration, potentially undermining their reliability for enduring operations across vast settings.

Our advanced system necessitates a mere 3 GB of GPU memory and exhibits greater tolerance to the variations in camera intrinsic precision, rendering it a superior choice for industrial deployment.

2) Visual Localization With a Prior Map: A pre-established map can furnish the localization system with vital information, significantly enhancing its accuracy and robustness [40], [41]. Huang et al. [42] introduced GMMLoc, a cross-modality method capable of tracking a camera within a pre-established map. Ye et al. [43] presented direct sparse localization (DSL), a method that uses a 3-D surfel-based map to enhance the camera's localization accuracy and robustness. However, its reliance on point cloud maps restricts its applicability in various scenarios. Huang et al. [44] used SDFs as the prior map for metric monocular visual localization. Nevertheless, these three methodologies face challenges in dynamic environments and necessitate significant memory for map storage. Compared with our proposed method, these three approaches necessitate considerable CPU power. To address the issues of memory consumption and scene variations, Yu et al. [2] developed a visual localization system suitable for large-scale, appearance-altering environments. However, due to its reliance on graph-optimization-based methods for positioning, it demands significant computational resources. Qin et al. [45] proposed a lightweight semantic map for visual localization, albeit reliant on a cloud map server and only partially leveraging the characteristics of various semantics. The aforementioned systems fail to provide the necessary descriptions of uncertainty and lack strategies for evaluating the accuracy of data correlations. Furthermore, point-matching-based methods for 6-DOF camera pose estimation exist, such as QPEP [46]. However, these are not comprehensive systems and they lack mechanisms for managing situations with erroneous data correlations.

In this article, we use semantic maps to compare the localization techniques of the inverse perspective mapping (IPM) matrix method and the perspective-n-point (PnP) method, particularly in scenarios with limited available corner pixels. Unlike the approach described in [45], our method achieves map localization without the need for a dedicated server for map storage. Instead, it uses only four corners, resulting in reduced costs and optimized computational resource utilization.

D. Contributions

To address the localization challenges faced by UGVs, we propose a robust visual-based localization system. While inspired by our previous work [2] on graph-based optimization for UGV pose and extrinsics estimation, our current approach introduces a marker-based correction mechanism. The significant contributions of this work include the below.

1) A proven and robust localization system suitable for environments with changing appearances. It uses an IPM matrix to directly obtain the UGV pose. This method offers enhanced computational efficiency and simplifies the process of introducing a covariance-based uncertainty system. Once this uncertainty is incorporated, our visual system can integrate smoothly with other positioning techniques. Using this uncertainty description, our visual system seamlessly integrates with other positioning methods. Besides QPEP, there is a notable absence of open-source SOTA solutions that provide an uncertainty description capability.
2) An innovative method for extracting polygon vertices is introduced. Specifically, it can efficiently identify diamond-shaped vertices without requiring extra hyperparameters. This algorithm offers potential adaptability for extracting vertices from other convex polygons.
3) Visual Feature-to-Map Matching Mechanism: We have implemented an advanced matching strategy between visual features and maps. The inclusion of a labeled filtering mechanism ensures the robustness of data association. In comparison to related work, there is a lack of analogous methods designed to ensure the exclusion of mismatched data.
4) Our system autonomously generates comprehensive high-precision maps vital for real-time localization. We have demonstrated that its positioning accuracy is comparable to maps created by obtaining marker coordinates using a total station, which further minimizes manual mapping efforts.
5) A novel calibration strategy is introduced, integrating surveying data for camera-to-vehicle alignment. This strategy focuses on improving calibration accuracy while reducing reliance on the precision of camera intrinsic. Moreover, the calibration results provided by this strategy are durable, eliminating the need for recalibration before each system operation.
6) The experimental results confirm the superior performance of the IPM method over the PnP approach, especially with limited matching points. Further data analysis indicates that our IPM technique favors a Gaussian distribution for error profiling, more so than the PnP method. This characteristic ensures a more accurate data representation for multisensor fusion.

E. Organization

The structure of this article is delineated as follows. Section III-A defines the problem and introduces foundational concepts. The overall system is outlined in Section III-B. Section III-C details the methodology for deriving the IPM matrix of a camera. In addition, it describes the pose relationship between the camera and the vehicle's center. Image processing techniques, encompassing road marker corner extraction and lane line identification, are covered in Section III-D. Section III-E introduces filters designed to eliminate erroneously detected rhombi, thereby enhancing system robustness. Vehicle positioning methodologies are presented in Section III-F, alongside a map layout strategy using rhombus road markers and an automated map generation approach. Section III-G details our automated technique for map construction and the underlying logic for map deployment. The experimental outcomes are shared in Section III. Section IV critically evaluates the strengths and weaknesses of our approach, and Section V concludes the article.

II. Problem Statement and Solutions

A. Problem Statement

The problem of map-based localization is represented as [10]

$$x_k^* = \arg\max_{x_k} P(x_k|Z_k, x_{k-1}, M) \tag{1}$$

where x_k stands for the vehicle's state at time k. It comprises the translation and rotation of the vehicle, denoted as $x = [q_w, q_x, q_y, q_z, t_x, t_y, t_z]^\top$. The set $Z_k = \{z_{k,1}, z_{k,2}, \ldots, z_{k,n}\}$ aggregates all the sensor measurements on the vehicle at time k, with each ith sensor measurement represented as $z_{i,k}$. M refers to the preexisting map data.

To reshape the problem using the maximum a posteriori (MAP) estimation, we use the Bayesian formula

$$x_k^* = \arg\max_{x_k} P(Z_k, x_{k-1}, |x_k, M) P(x_k, M). \tag{2}$$

Assuming the independence of observations between adjacent time sensors, we can rewrite (2) as

$$x_k^* = \arg\max_{x_k} \prod_{i=1}^{n} P(z_{k,i}, x_{k-1}, |x_k, M) P(x_k, M). \tag{3}$$

To apply (3), we must construct models tailored for different sensors. Most vehicle planning and control algorithms assume the vehicle operates on a virtually flat plane. This permits us to simplify x to $x = [\theta, t_x, t_y]$, reducing the state number and hastening subsequent operations.

B. Overview

Our system is based on the assumption that the terrain is relatively flat, a characteristic prevalent in indoor environments and numerous engineered settings, including ports, airports, and industrial zones. This terrain flatness is similarly observed on the majority of urban roads, where ground undulations are typically minimal.

Furthermore, we posit that the system initializes from a known pose—a standard protocol in automated systems, which typically commence operations from a predetermined location. This approach not only streamlines the initiation procedure but also mitigates the risk of global localization errors, thus minimizing the likelihood of system failures.

Fig. 1 illustrates the pipeline of our proposed localization system. Commencing with the initial pose $\hat{\boldsymbol{x}}$, the system adeptly estimates the vehicle's status by harnessing both the camera and the wheel encoder. In scenarios where valid image data are unavailable, chiefly as a result of the camera's limited sampling rate, wheel odometry presents itself as a computationally frugal alternative. This mechanism bridges data voids and perpetually refreshes the vehicle's latest status, guaranteeing seamless transitions and persistent accuracy.

At time t, the segmentation network processes the image to demarcate rhombus and lane lines, depicting them as vertex points and linear stretches. Subsequently, leveraging the positioning data furnished by the odometry, the system facilitates data association between the image and the map, thereby determining the vehicle's state at time t.

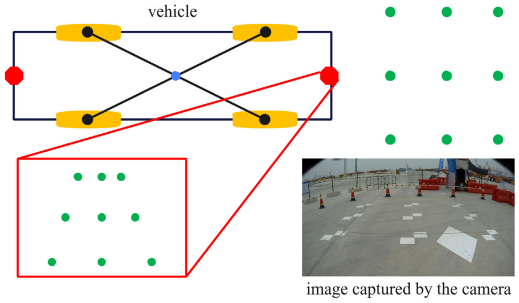


Fig. 2. UGV outline. Black: vehicle body. Yellow: wheels. Red: cameras. Green: ground and image markers. Blue: vehicle body center.

The selection of rhombus-shaped markers for our system is driven by two primary considerations. First, simple geometric shapes are favored for their detectability and robustness within visual localization frameworks. Second, rhombus configurations are commonplace for pedestrian crosswalk warning lines in numerous countries and are widely accessible due to standardized construction molds. Conversely, triangular markers are rarer and not standardized in their production, potentially complicating deployment and diminishing the feasibility of broad implementation. Therefore, rhombus markers offer a practical and efficient solution for our system's design and application needs.

C. Parameter Calibration

For the autonomous system's planning and control to be effective, it is crucial to identify the vehicle body's center, not just the camera's pose. Hence, we need to calibrate the external parameters from the camera to the vehicle body center. Importantly, this calibration is a one-time procedure; the results can be recurrently used for subsequent vehicle operations, negating the need for recalibration before each session.

Although the total station can yield measurements with remarkable precision—accurate to 0.001 m—it is laborious to measure the vehicle's center body coordinates directly. To navigate this, we define the body center as the connecting line's midpoint between the four tire centers, as illustrated in Fig. 2, where the vehicle body center is interpreted as the intersection of the diagonal lines.

1) Get the IPM Matrix: Each feature point on the ground corresponds to a specific point in the camera image. Assuming a flat ground surface, we leverage the IPM matrix $\boldsymbol{H}$ to define their relationship as $\boldsymbol{H}\boldsymbol{p}^i = \boldsymbol{p}^w$ [47], where $\boldsymbol{p}^i$ denotes the pixel position in the image, and $\boldsymbol{p}^w$ represents the coordinate of the corresponding point on the real-world ground.

Note that we have omitted the transformation involving homogeneous coordinates; this notation will persist without further explanation in Sections II-C and II-E–II-G.

Generally, using more points can enhance the calibration accuracy. To guarantee accurate data association, we annotate the image pixels manually. However, excessive data annotation can often lead to errors in the operation. In practice, we use nine points for IPM matrix calibration, as illustrated in Fig. 2. Since the matrix $\boldsymbol{H}$ possesses 8 DOFs, a minimum

of four pairs of noncollinear points are required; subsequently, we apply the method described in [2] to determine $\boldsymbol{H}$.

It is necessary to describe the uncertainty associated with the matrix $\boldsymbol{H}$. We define the projection error as the lower bound of error. In the calibration process involving n points, each denoted as $\boldsymbol{p}_k^i$ in the image, we determine the error as $\boldsymbol{e}_k = \boldsymbol{H}\boldsymbol{p}_k^i - \boldsymbol{p}_k^w$. Consequently, the covariance of $\boldsymbol{e}_k$ is represented as $\boldsymbol{n}^i$. The term $\boldsymbol{n}^i$ describes the uncertainty in projecting a point from the image to the ground.

After getting the IPM matrix, we can know the pixel coordinate $\boldsymbol{p}_b^i$ in images is

$$\boldsymbol{p}_b^i = \boldsymbol{H}^{-1}\boldsymbol{p}_b^w \tag{4}$$

where $\boldsymbol{p}_b^w$ is the coordinate of the vehicle center we introduced earlier.

2) Calibrate Camera Extrinsic Parameters: To accurately establish the camera's pose in the world, we use a PnP approach that exploits the known correspondences between world points and their projections onto image pixels. However, this method's reliability is compromised under conditions where the camera's intrinsic parameters and distortion coefficients are inaccurately determined or when the PnP solver is provided with a sparse set of points.

To circumvent these challenges, we introduce measurements from a total station, which supplies a precise estimate of the camera's position, denoted by the vector $\boldsymbol{t}$. These additional data transform the problem from a 6-DOF estimation to a more tractable 3-DOF problem, focusing solely on the camera's orientation.

The optimization of the camera's orientation, represented by $\boldsymbol{R}^*$, is formulated as

$$\boldsymbol{R}^* = \arg\max_{\boldsymbol{R}} \frac{1}{2}\sum_{i=1}^{n}\left\|\boldsymbol{u}_i - \frac{1}{s_i}\boldsymbol{K}(\boldsymbol{R}\boldsymbol{P}_i + \boldsymbol{t})\right\|_2^2 \tag{5}$$

where $\boldsymbol{R}^*$ signifies the optimized camera orientation, $\boldsymbol{R}$ is the 3×3 rotation matrix, $\boldsymbol{t}$ is the 3×1 position vector provided by the total station, and $\boldsymbol{u}_i$ corresponds to the coordinates of the green points within the red box in Fig. 2. The s_i denotes the scale factor for the projection of the ith world point $\boldsymbol{P}_i$, computed as $\boldsymbol{K}(\boldsymbol{R}\boldsymbol{P}_i + \boldsymbol{t})$. Here, $\boldsymbol{K}$ represents the camera's 3×3 intrinsic matrix, and n denotes the number of matches between pixels in the image and points in the physical world. In this context, the value of n is 9, indicating that there are nine points in the image that correspond to actual locations on the ground. The optimization defined by the above equation can be solvable using any standard optimization tools.

D. Image Processing

The rhombus shape, recognized for its simplicity, serves as a pavement marking in numerous countries, a testament to its practicality as a road marker. To facilitate the identification of such markers in the autonomous system, an incoming image is concurrently processed through two distinct neural networks. Initially, the image is channeled to the Mask-RCNN [48] to segment and isolate the road markers. Simultaneously, the SCNN network [49] operates to demarcate the lane boundaries through segmentation. This dual-network processing ensures a comprehensive analysis of the road terrain, enhancing the system's responsiveness and accuracy in detecting vital road signals.

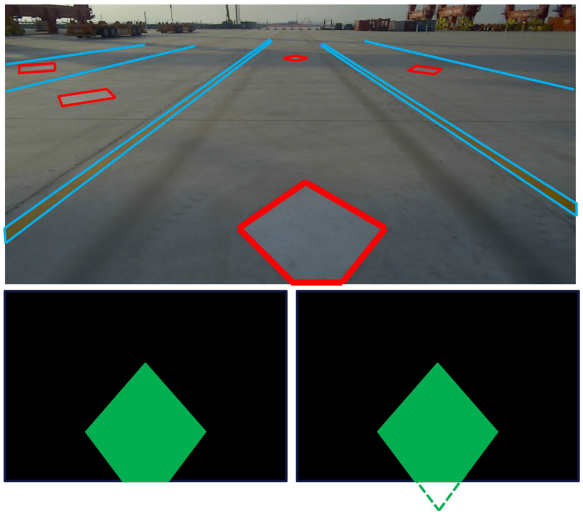


Fig. 3. Up: the real rhombus from image. Left: original rhombus. Right: the complete rhombus. Red: rhombus in the raw image. Blue: lane line in the raw image. Green: mask of the largest rhombus.

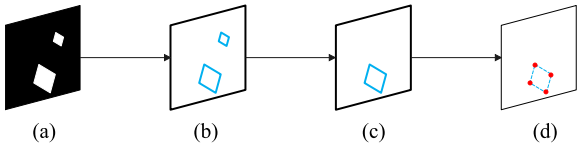


Fig. 4. (a) Image obtained through segmentation, with the white region denoting the segmented rhombus. (b) Following this, we use a flood-fill approach on the image in (a) to delineate the rhombus contours. Subsequently, the largest rhombus is selected from (b) to produce image (c). Ultimately, using the method detailed in Fig. 5, the red region in (d) highlights the four vertices of our target rhombus.

1) Lane Marker: In this article, the sole function of lane lines is to furnish the system with heading angles. It stands to reason to convert all the segmented pixels into 2-D points and then categorize them using the L2 distance clustering algorithm. Following this, we use the previously calibrated IPM matrix, denoted as $\boldsymbol{H}$, to transfer these points to the world frame. To further refine this representation, we implement a RANSAC-based line fitting method to delineate the line in the world frame. Essentially, we extract a line from the image and transpose it to a calibration frame using the pre-established IPM matrix $\boldsymbol{H}$. This process facilitates the depiction of the line as two connected points, $\mathfrak{p}_a^w$ and $\mathfrak{p}_b^w$, within the calibration frame.

2) Road Marker: While the Mask-RCNN yields segmentation results, they cannot be used directly; our primary interest lies in identifying the four corners of the segmentation. The line fitting algorithms presented in [2] require a hyperparameter d and are incapable of handling incomplete rhombus, as illustrated in Fig. 3. This issue can result in the failure to recognize rhombus when the vehicle is moving at high speeds. The line-fitting algorithm using RANSAC tends to be computationally intensive, typically operating on the CPU, necessitating enhancements in the method for recognizing rhombus corners.

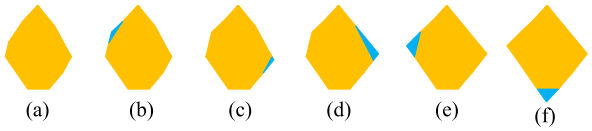


Fig. 5. For a convex polygon, consider any two adjacent angles. By extending their corresponding edges to an intersection, a new vertex is formed, replacing the original two angles with a single angle. This operation reduces the polygon's number of angles by one and increases its area. In the provided figure, the added area is highlighted in blue. Throughout this procedure, we consistently opt for the method that minimally increases the area until the polygon becomes a quadrilateral. The steps from (a) to (f) in the figure illustrate this progression.

Rhombi that appear larger in the image are generally closer to the camera, offering a reduced margin of error in measurement due to a smaller real-world distance represented per pixel. Therefore, we only process the largest rhombus in the image to enhance the precision of our localization efforts. Fig. 4 illustrates the procedure for rhombus selection.

Drawing inspiration from [50], we find it feasible to approximate the rhombus, a convex polygon, using a minimum circumscribed quadrilateral. Corner extraction only needs data on mask contour. So we use the flood-filling algorithm to extract the pixels on the outline. The method first obtains the convex hull of the outline pixels and the convex hull point set. This step can greatly reduce the number of pixels to be processed. Deleting an edge in the convex hull and extending its two adjacent edges of this edge can increase the area of the convex polygon by S_i and reduce the convex polygon by one corner. Repeatedly delete the corner with the smallest area until the polygon has only four corners, which are the corner points we need. The algorithm complexity is $O(n \log n)$, and n is the number of outline pixels of the mask. Fig. 5 provides an example illustrating this process.

E. Data Matching and Filtering

Unlike lane markings, rhombuses in images appear more compact and concentrated, while lane markings cover a broader image area. Fig. 6 illustrates a scenario where scene recognition becomes more complex due to rainfall. As a result, accurately extracting the details of the rhombus shape becomes challenging. Moreover, the results of deep learning network segmentation do not ensure flawless extraction and corner localization of rhombuses. As a result, it becomes crucial to use a series of validation methodologies. Next, this section introduces procedures for data matching and filtering, with a primary focus on eliminating low-quality data, ensuring precise image-to-map associations, and enhancing system robustness. This specific procedure involves examining rhombus edge lengths, aligning them with map rhombuses, using them for localization, and filtering data anomalies resulting from localization.

1) Verification of Rhombus Edge Lengths: According to Section III-D, we denote the last 4 pixels extracted in the image as $\tilde{\boldsymbol{p}}_j^i$, where j denotes the index of the extracted pixel. For any two pixels in the images, $\boldsymbol{p}_a^i$ and $\boldsymbol{p}_b^i$, projected from the ground, we can calculate the distance d_{ab} as follows:

$$d_{ab} = \left\| \boldsymbol{H}\left(\tilde{\boldsymbol{p}}_a^i - \tilde{\boldsymbol{p}}_b^i\right) \right\|_2^2 \tag{6}$$

Fig. 6. Lane lines are marked in magenta, normal rhombi in red, and rhombi that are difficult to detect due to their similar color to the surroundings in blue. False rhombi formed by rainwater are shown in green. Detecting these markers accurately is challenging due to their appearance variations.

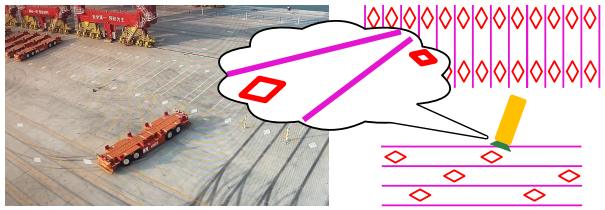


Fig. 7. Left image depicts the actual view of the vehicle within the port, while the right image illustrates the vehicle's position on the map, determined through localization results. Using the extrinsic parameters linking the camera, vehicle, and map data, it becomes possible to infer the expected map elements within the camera's field of view.

where d_{ab} represents the distance between these two ground-projected pixels.

Precise rhombus identification is essential. Unlike lane lines, rhombi occupy a smaller portion of the image and may not always be fully captured. We use the metric $\max \text{abs}(d_{ab} - \bar{d})$ to validate that the distance between adjacent rhombus corners aligns with our expected side length, represented by $\bar{d}$. This approach effectively reduces the number of false positives.

2) Matching Rhombus Data to Map: A map can be constructed through various means, including total station surveys, construction CAD drawings, or the methods introduced in Section III-F. We project map data onto the image using the current vehicle position to identify overlaid map elements. Clearly, not all the elements are represented on the map. Only a small portion of elements in front of the camera are projected onto the image. So, it is not necessary to exhaustively enumerate all the rhombuses on the map. Only a few rhombuses closer to the vehicle have the value of being checked, and ikd-tree [51] can speed up the process. Fig. 7 offers an intuitive demonstration of acquiring virtual image elements. With the aid of virtual image elements, data association can be easily achieved in the image based on the nearest-point matching principle. Fig. 8 illustrates the process of matching map elements with image elements.

Given the previously provided context, the "nearest-point matching principle" mentioned serves as a foundational method, enabling the filtration of erroneous rhombus shapes. Theoretically, the coordinates of map elements projected into the camera should perfectly align with the image elements. According to (5), $\boldsymbol{K}(\boldsymbol{R}\boldsymbol{p} + \boldsymbol{t}) = \boldsymbol{u}$ should hold true, where $\boldsymbol{K}$ represents the camera intrinsic parameters, $\boldsymbol{R}$ and $\boldsymbol{t}$ denote

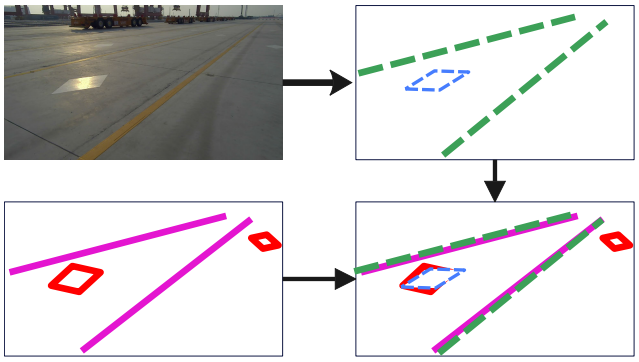


Fig. 8. Upper left image represents the camera's view, while the upper right image displays the result after passing through a segmentation neural network. The lower left image shows the data obtained by projecting the map from Fig. 7 onto the camera. Lane lines are depicted as straight lines, and data matching is achieved by having the green lines locate the nearest pink lines. Rhombuses can be represented by their centers, and data matching is completed by identifying the red rhombus in the image that is closest to the center of the blue rhombus.

the extrinsic parameters of the current camera in the world coordinate system, $\boldsymbol{u}$ signifies the pixels in image elements, and $\boldsymbol{p}$ corresponds to the elements in the map.

Inevitably, discrepancies exist, making the equation potentially inaccurate. These discrepancies primarily arise from two sources: errors attributed to camera parameters and localization errors, with the former generally being negligible due to their minimal impact and the difficulty in quantification.

Angular and translational errors are represented by $\triangle \boldsymbol{R}$ and $\triangle \boldsymbol{t}$, respectively. Ideally, the pixel error, denoted as i, is defined as $||\boldsymbol{u} - \boldsymbol{K}(\boldsymbol{\Delta R p} + \boldsymbol{t} + \boldsymbol{\Delta t})||_2^2$.

Aware of the covariance characterization of the uncertainty in vehicle pose localization, we can use three times the standard deviation in place of $\triangle \boldsymbol{R}$ and $\triangle t$, deriving an error measure i, used as a filtering threshold to exclude noncompliant image data.

In industrial applications, an allowable maximum localization error is generally acknowledged, varying with the specific localization task. The allowable maximum angular error $\triangle \boldsymbol{R}$ and translational error $\triangle \boldsymbol{t}$ can also be used to compute an error measure i, serving as a filtering threshold. Fig. 9 illustrates the process of using a threshold to filter out incorrect matches.

In engineering practice, this parameter can be adjusted flexibly based on different localization tasks, for instance, by proportionally scaling i. The two strategies presented herein are frequently used in our practical work, serving as robust and valuable references for various localization tasks.

3) Data Filtering via Localization: At any given moment, our system possesses a localization result characterized by covariance. The specifics of the localization method will be discussed in Section III-F. This section will outline how localization is leveraged to filter out irrelevant data. The position inferred from the wheel encoder is denoted as $\boldsymbol{t}$, with a corresponding covariance of Σ_t. Meanwhile, the location result derived from rhombus data is represented as $\boldsymbol{t^p}$. We use the Mahalanobis distance [52] to express the divergence between them, expressed as

$$D_M = \sqrt{(\boldsymbol{t} - \boldsymbol{t^p})^\top \Sigma_t^{-1} (\boldsymbol{t} - \boldsymbol{t^p})}. \tag{7}$$

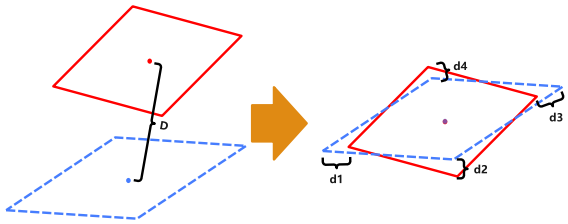


Fig. 9. Red rhombus represents the shape projected from a map element onto the image, while the blue rhombus is obtained using the method described in Section III-D. The positions of these rhombi in the image are their original locations. We can calculate the distance D between their center positions. If this distance D exceeds the threshold i, the match is deemed unsuccessful. Once a successful match is achieved by aligning the centers of the rhombi, we can proceed to associate the data of the four corners of the rhombus using the nearest-point matching principle.

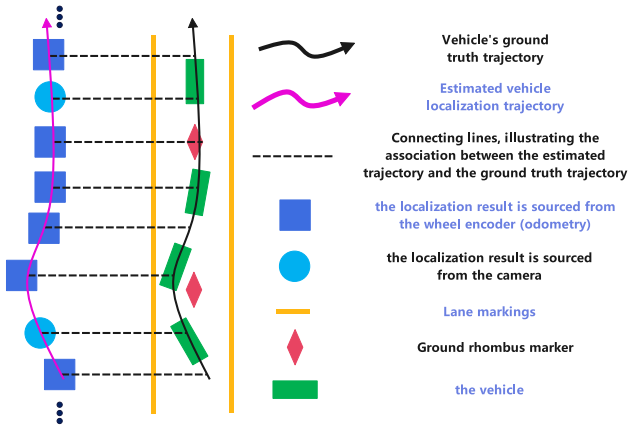


Fig. 10. Adaptive localization results demonstrating the dynamic sourcing of vehicle pose data from wheel-encoder-derived odometry and camera data upon rhombus marker detection.

Based on the three-sigma rule, we select three as the threshold here, given that approximately 99.7% of the data points in a normal distribution are within three standard deviations from the mean.

F. Localization

The primary concept of localization is to use lane markings to provide orientation for the vehicle and use rhombus-shaped data to offer positional constraints. When effective visual constraints are unavailable, odometry from wheel encoders is used to furnish information on the vehicle's pose, as illustrated in Fig. 1. The status of a vehicle at time k is represented as $\boldsymbol{x}_k = [t_{xk}, t_{yk}, \theta_k]^\top$, where t_{xk}, t_{yk}, and θ_k denote the coordinates and orientation of the vehicle, respectively. This section focuses on how different types of sensor data, specifically from the wheel encoder and camera, are used as input for the localization system. Fig. 10 demonstrates how our localization results switch between camera data and wheel encoder data.

1) Wheel Encoder: The wheel encoder data are naturally converted into wheel odometry [53], forming a queue $\mathfrak{Q} = \{\boldsymbol{d}_i\}$, where each element $\boldsymbol{d}_i = [dx_i, dy_i, \theta_i]^\top$ represents the displacement between time i and time $i+1$. This queue is crucial for tracking the vehicle's movement over time. If the vehicle's pose at time k is known, its pose at the last time can be easily determined.

Given the known covariance of the vehicle pose at time k and the covariance of $\boldsymbol{d}$, the pose covariance after movement

can be deduced through covariance propagation, as described in [54].

2) Camera: Based on the methods described in Section III-D, four corners of road markers and lane lines are extracted from the image data provided by the camera.

a) Lane line: When valid lane information is present in the image, two straight lines in the world frame at time k can be obtained, which are then converted into a heading angle for the vehicle as per [2]. The average heading angle $\overline{\theta}_k$ and its covariance n^l are calculated from the orientations obtained for every lane.

By linear interpolation from $\mathfrak{Q}$, we can get $\boldsymbol{x} = [\check{x_k}, \check{y_k}, \check{\theta_k}]^\top$ and can update the orientation at time k by

$$\begin{pmatrix} \check{x_k} \\ \check{y_k} \\ \check{\theta_k} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \check{x_k} \\ \check{y_k} \\ \check{\theta_k} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \overline{\theta}_k \end{pmatrix}. \tag{8}$$

Then the covariance of the pose vehicle can be updated by $\hat{\Sigma}_k^l = \boldsymbol{A}\check{\Sigma}_k\boldsymbol{A}^\top + \boldsymbol{B}$, where $\boldsymbol{A}$ is a three-by-three matrix in (8), and $\boldsymbol{B}$ is the covariance matrix converted by $\boldsymbol{n}^l = [0, 0, n^l]^\top$. Assuming that the vehicle is moving at a constant speed, at time k, by linear interpolation in $\mathfrak{Q}$, we can get $\boldsymbol{x}_k$, and its covariance is $\check{\Sigma}_k$. To minimize the estimated covariance, we merge the results of interpolation and lane estimation method by the basic Gaussian distribution fusion [55].

b) Rhombus road marker: When the input data comprise the four corners of a rhombus, using the PnP algorithm is a rational choice due to its efficient solution of such geometric problems. However, this method faces challenges; it requires a higher degree of camera calibration accuracy and involves significant CPU computations.

Based on (4), using the IPM matrix, we can obtain

$$\boldsymbol{t}_j = p_j^d - \boldsymbol{R}\boldsymbol{H}\hat{p}_j^i \tag{9}$$

where $\boldsymbol{t}_j = [t_{xkj}, t_{ykj}]^\top$ is the vehicle position estimated by jth pixels at time k. And $\boldsymbol{R}$ is the rotation matrix that represents the current position of the vehicle. Then we can use $\tilde{\boldsymbol{t}}_k = (1/4)\sum_{j=1}^4 \boldsymbol{p}_{b_j}^i$ to represent the final estimate by the marker input.

Similar to (8), we can get

$$\begin{pmatrix} \hat{x_k} \\ \hat{y_k} \\ \hat{\theta_k} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \check{x_k} \\ \check{y_k} \\ \check{\theta_k} \end{pmatrix} + \begin{pmatrix} t_{xkj} \\ t_{ykj} \\ 0 \end{pmatrix}. \tag{10}$$

Assuming the coordinate covariance of the corner points, along with covariance propagation and the fusion of Gaussian distributions, the vehicle's pose estimation at time k can be determined.

Note that when calculating priorities, we consider wheel odometry, lane data, and rhombus data. If there are no input data for lanes or rhombi, the corresponding processes can be omitted.

G. Map Layout and Generation

1) Map Layout: The rhombus, being a simple element for road markers, is a reasonable choice for localization markers. Having too many markers can compromise esthetics and increase workers' workload. Multiple markers in a single image can also complicate data association. To address these challenges, a thoughtful marker layout is essential.

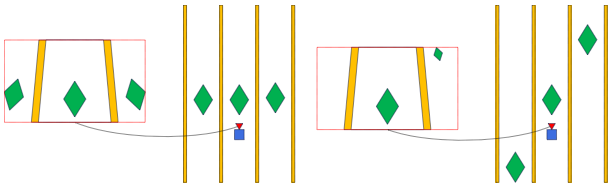


Fig. 11. Green rhombus represents the rhombus marker. Yellow strip: the lane line. Blue square: the vehicle. Red triangle: the camera in the front of the vehicle. On the left, three markers are aligned, while on the right they are staggered.

A straightforward approach is to stagger the markers, as shown in Fig. 11. If markers are aligned, multiple rhombuses of similar size might appear in the image simultaneously. By staggering the markers, the rhombus area in the image will differ significantly from the second-largest rhombus, simplifying data association.

2) Map Generation: In localization tasks, the map encompasses the coordinates of the rhombus corners and the linear equations of each lane line. Most vehicles operate in open-air environments, making it challenging to use surveying tools for marking or lane coordinate measurements. Even though we have CAD drawings of construction drawings, workers cannot guarantee that they will fully implement them according to the specifications of the construction drawings. Thus, we aim to automate map generation.

A robust GNSS signal, when combined with real-time kinematic (RTK), can yield stable and accurate localization results. Using a well-calibrated vehicle, we can automatically generate road marker coordinates in areas with strong GNSS signals. Based on the accurate localization result, we can provide a method to auto generate the coordinate of rhombus corners.

Lane Line: Lane masks can be extracted using [49]. Notably, most lane detection algorithms are compatible, enhancing the algorithm's flexibility. Each pixel in the mask can be transformed into the world frame using the IPM matrix

$$\boldsymbol{p}_{jk}^{wl} = \boldsymbol{R_k}\boldsymbol{H}\boldsymbol{p}_{jk}^{il} + \boldsymbol{t_k} \tag{11}$$

where $\boldsymbol{R_k}$ and $\boldsymbol{t_k}$ represent the orientation and translation of the vehicle, respectively, $\boldsymbol{p}jk^{il}$ denotes the jth lane pixel in the image, and $\boldsymbol{p}^{wl}jk$ is the coordinate of the jth lane pixel converted to the ground in the world frame. In every symbol, k indicates the specific time instance. After the vehicle traverses the entire path, numerous 2-D points can be obtained in the world frame. The L2 distance clustering algorithm can classify each lane, and subsequently, SVD can be used to fit the line, thereby providing orientation for localization.

In rhombus detection, a significant challenge lies in the fact that false detections can severely degrade the accuracy of the map. Although (11) allows us to project the rhombus from the camera onto the map, this may introduce erroneous projections.

To address this issue, we consider using the strategy detailed in Section II-E, "Verification of Rhombus Edge Lengths," to effectively eliminate these false detections. With an accurate

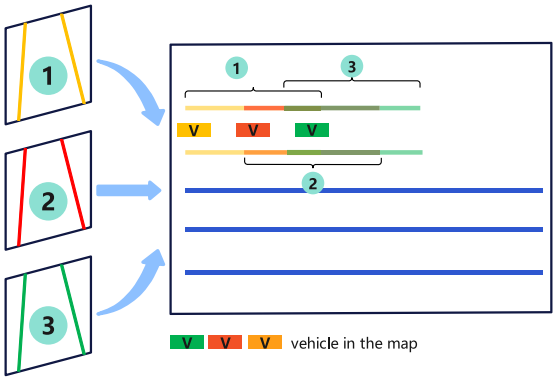


Fig. 12. In the left image, lane markings observed from three distinct vehicle positions are depicted. On the right image's map, these markings correspond to positions 1–3. Rectangles in the image indicate the vehicle's positions. The yellow, red, and green colors represent lane markings from each position. It iss evident that some lane markings overlap when mapped. These overlaps create a continuous lane marking, enabling us to use linear fitting for a complete lane representation. The blue section in the right image depicts lanes parallel to the primary one.

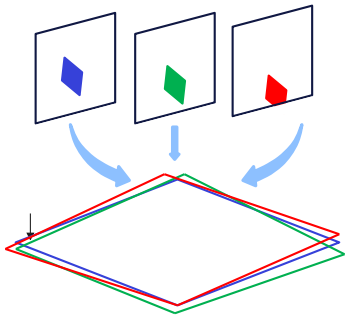


Fig. 13. Rhombus may be detected multiple times. As shown in the figure, the repeated detections of the rhombus often result in overlaps on the map. By clustering the corner coordinates from these detections and calculating their average, the precise location of the rhombus can be determined.

construction map at hand, we can further refine our detections using "matching rhombus data to map" and "data filtering via localization" methods. However, in the absence of this map, it is suggested to map the rhombus onto a provisional map upon its initial detection and then continually validate using the aforementioned strategies.

Referring to Figs. 12 and 13, multiple data descriptors might be acquired for a genuine rhombus. By averaging the coordinates of the rhombus that have undergone the IPM transformation and are projected in the world coordinate system, we can precisely determine its position on the map. The workflow is illustrated in the accompanying figure.

III. Experiments

We first perform the simulated and real-world experiments for static IPM-based localization and PnP localization. Next, real-world experiments for our system are performed. Finally, we compare the computation time for each frame.

A. Experimental Environment and Setup

Our experiments were conducted at a container port, consistent with our previous work [2]. The experimental site is depicted in Fig. 6. We used a pinhole camera with a resolution of 1280 × 720. Its intrinsic parameters were calibrated following the method presented in [56].

The camera extrinsic parameters and IPM matrix are obtained by the methods provided in Section III-C. To make the experiment reflect the real situation, the values of parameters are also used in the simulation experiment. The side length of rhombus we used is 1 m, and we use 0.2 m as the limit of maximum value of d_{ab} in (6).

Unless specified otherwise, all the experiments were conducted using an Intel Core i7-8700K processor with 24 GB of RAM and an NVIDIA RTX 3090 GPU equipped with 24 GB of memory. This setup will henceforth be referred to as the "x86-64 platform."

B. Performance of Static Localization

We keep all the equipment static to get localization accuracy by different methods. By simulating the UGV's pose and the ground marker's coordinates, we can project the pixels onto the image. Fig. 2 shows the markers on the ground and the corresponding pixels in the images. We add noise to the pixels in the images and then use two methods for positioning.

1) *PnP-Based Localization:* We use PnP [57] to get the camera pose and the vehicle's position. Note that we are only interested in the three-axis pose, so the result will be projected to a 2-D plane to compare with the IPM localization method.
2) *IPM Localization:* Although we use lane information in our system to get the vehicle orientation, the IPM localization can get the orientation directly. In this experiment, we simultaneously use IPM to estimate the translation and orientation.

1) Estimate Orientation and Translation: We use 1.4 pixels as the standard noise variance in this experiment. Then, the translation error result is shown in Figs. 14 and 15. It should be noted that the orientation variance is large. When few matched points are available, we can observe that the IPM localization method has a stronger anti-noise ability than the PnP method. In addition, a small increase in the number of point matches can only provide limited help to improve accuracy. Because $\sin(0.5°) \approx 0.01$, which means if our vehicle with a heading error of 0.5°, the vehicle will deviate 0.1 m as long as go forward 10 m.

2) Estimate Translation Only: The 1.4 pixels as the noise standard variance and heading angle with 0.1° noise will be fed into the two systems. We use only 4 matched pixels to estimate the vehicle position. The result is shown in Fig. 16. We can see that by a good orientation provided to the system, the stability and reliability of the system can be upgraded. In addition, the PnP method has a smaller variance in the horizontal direction. However, it costs 5.23 ms per frame, and the IPM method costs 0.01 ms for one frame. The reason is that matrix multiplication is much faster than matrix factorization. The IPM method requires only matrix multiplication.

C. Performance of Moving Localization

On an 1800-m route, we tested the positioning accuracy of the following methods, respectively.

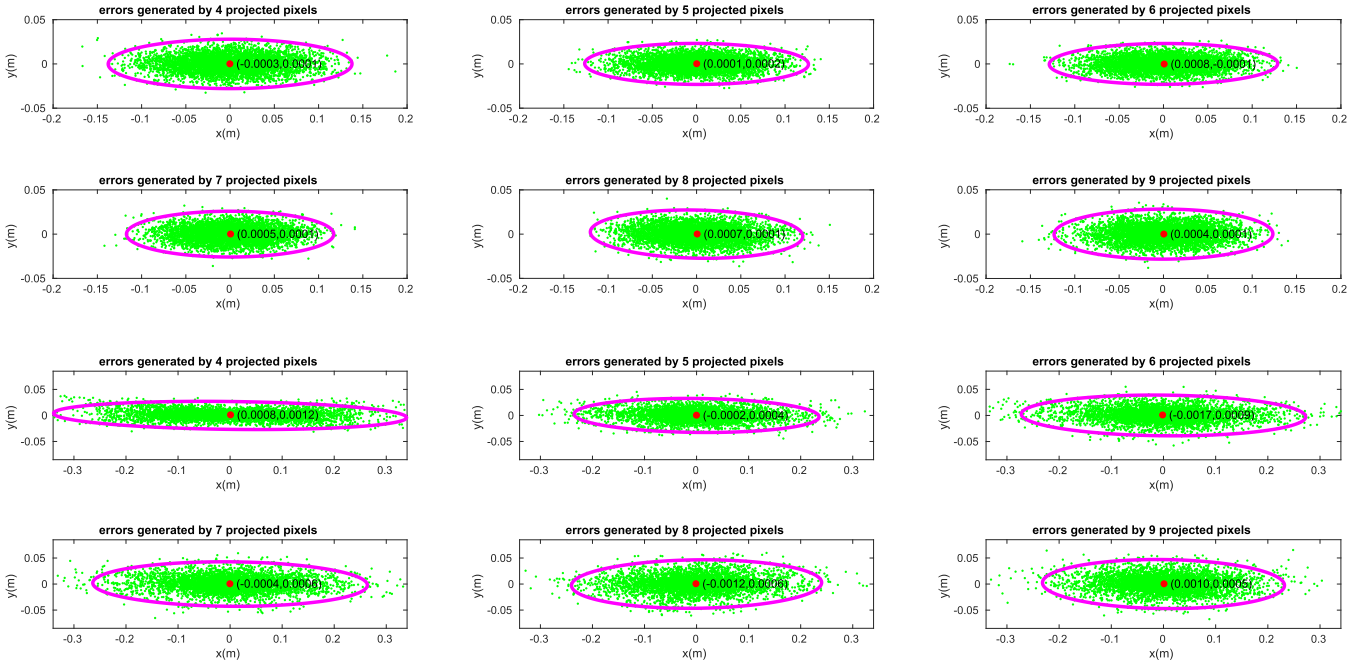


Fig. 14. Top six: IPM localization translation errors; bottom six: PnP translation errors. Each figure corresponds to 4–9 pixels in positioning. Green points show x–y-direction errors, magenta illustrates triple covariance ellipse, and red denotes mean error.

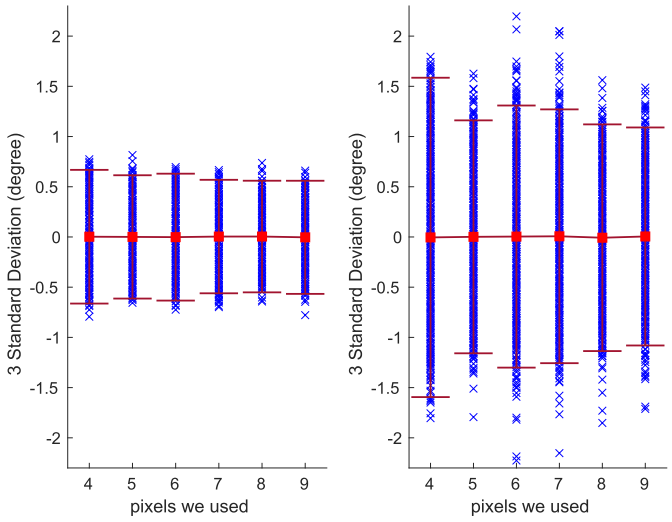


Fig. 15. Orientation errors: IPM method (left) versus PnP method (right). Blue X indicates orientation (yaw) error, and red horizontal line represents three times the standard variance of yaw error.

1) $\boldsymbol{I}_1$: IPM with orientation provided by lane.
2) $\boldsymbol{I}_2$: IPM with orientation provided by ground truth.
3) $\boldsymbol{I}_3$: IPM without any orientation provided.
4) $\boldsymbol{P}_1$: PnP with orientation provided by lane.
5) $\boldsymbol{P}_2$: PnP with orientation provided by ground truth.
6) $\boldsymbol{P}_3$: PnP without any orientation provided.

We are interested in positioning when using a marker, and this positioning accuracy directly determines whether the whole system can work well. First, we use the *ground-truth map*; moreover, the result can be found in Table I. We can see that the IPM method can get the best performance, even if the orientation is provided by ground truth (RTK-GNSS). We also test our system in *generated map*, in Table II, and

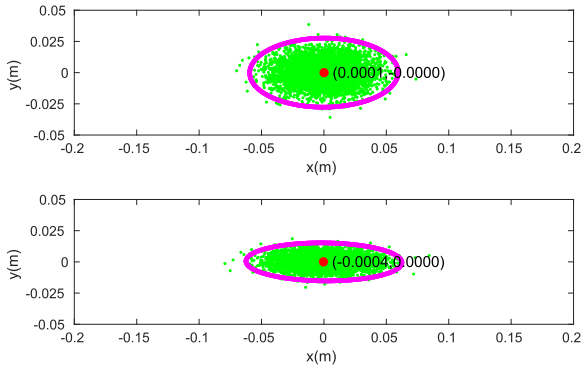


Fig. 16. Error distribution: IPM method (top) versus PnP method (bottom). Color descriptions match Fig. 14.

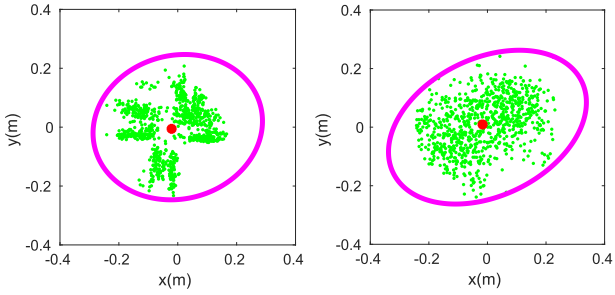


Fig. 17. Error distribution: $\boldsymbol{P}_1$ (left) versus $\boldsymbol{I}_1$ (right).

we can see that the IPM method is more robust than the PnP method. We also plot the error distribution of P_1 and I_1; the result is shown in Fig. 17. We can find that $\boldsymbol{P_1}$ distribution does not look like a normal distribution. Moreover, compared with $\boldsymbol{P_1}$, $\boldsymbol{I_1}$ is closer to a normal distribution.

D. Bad Calibration for Camera

We exchanged the front and rear cameras' internal parameters and distortion coefficients, and the pixel positions

TABLE I
LOCALIZATION RESULT WITH GROUND-TRUTH MAP

Case	I_1	I_2	I_3	P_1	P_2	P_3
max T(m)	**0.246**	0.246	0.520	0.255	0.251	0.399
max R(deg)	**0.220**	NaN	2.292	0.220	NaN	1.990
mean T(m)	**0.116**	0.120	0.147	0.125	0.125	0.141
mean R(deg)	**0.095**	NaN	0.329	0.095	NaN	0.420

TABLE II
LOCALIZATION RESULT WITH GENERATED MAP

Case	I_1	I_2	I_3	P_1	P_2	P_3
max T(m)	**0.220**	0.222	0.339	0.669	0.672	0.712
max R(deg)	**0.220**	NaN	1.737	0.220	NaN	2.724
mean T(m)	**0.099**	0.098	0.139	0.424	0.424	0.434
mean R(deg)	**0.093**	NaN	0.496	0.093	NaN	1.126

TABLE III
LOCALIZATION RESULT WITH SWAPPED PARAMETERS IN GT MAP

Case	I_1	I_2	I_3	P_1	P_2	P_3
max T(m)	**0.249**	0.249	0.515	0.280	0.274	0.416
max R(deg)	**0.220**	NaN	2.275	0.220	NaN	2.089
mean T(m)	**0.115**	0.120	0.147	0.128	0.127	0.147
mean R(deg)	**0.095**	NaN	0.326	0.095	NaN	0.473

of the image will be different from that before due to the changed distortion coefficient.

Therefore, it is necessary to calibrate various components, e.g., the IPM matrix and the relative pose between the camera and the vehicle body center.

Subsequently, we replicate the experimental procedures outlined in Section II. Table III shows the results, and we can see that the IPM method is robust for camera parameters.

E. Comparison to Other Algorithms

A critical comparison to prior work, such as that in [2], is essential to benchmark the performance of our system. We conducted a comprehensive evaluation against SOTA VI odometry (VIO) systems, including VINS-Mono [5] and ORB-SLAM3 [23]. To improve the accuracy of the two VIO systems, we integrated an IMU into the vehicle's setup. This integration was carefully calibrated with the onboard camera using the Kalibr toolkit [58]. It is noteworthy that the DSL technique necessitates a point cloud map, as detailed in [43]. For this requirement, we used G-LOAM [7] to generate the needed data. Furthermore, comparisons were made with LiDAR-based approaches, including Fast-LIO2 and NDT-LOAM [59], to demonstrate the potential for synergy between visual and LiDAR systems.

In addition to traditional methods, and considering the limitations of depth information in outdoor environments, we selected the most recent 2023 SOTA learning-based monocular camera SLAM solutions for our experimentation, which include GO-SLAM [36], Orbeez-SLAM [34], and panoptic visual odometry [39]. Ultimately, DROID-SLAM was also included in the evaluation, as it is commonly used as a baseline in recent learning-based research. We initialized the SLAM systems with the ground-truth coordinates. And

TABLE IV
ACCURACY COMPARISON OF DIFFERENT ALGORITHMS (m, deg)

	max T	max R	mean T	mean R
I_1	**0.249**	**0.220**	**0.115**	**0.095**
VINS-Mono [5]	1.582	6.491	0.573	1.477
ORB-SLAM3 [23]	0.792	2.782	0.247	0.758
DSL [43]	0.572	0.754	0.247	0.223
Ref. [2]	0.315	0.234	0.139	0.128
FastLIO2 [60]	X	X	X	X
NDT-LOAM [59]	X	X	X	X
GO-SLAM [36]	2.317	2.301	0.505	0.684
Orbeez-SLAM [34]	2.395	2.955	0.577	0.391
PVO [39]	1.915	0.937	0.589	0.347
DROID-SLAM [37]	3.392	2.411	1.207	0.802

TABLE V
CORNER EXTRACTION TIME COMPARISON

Case(ms)	I_1	Ref. [2]
Cortex-A78AE	**5**	37
Cortex-R5	**8**	88
Intel i7	**2**	19

due to the intrinsic limitations of monocular visual SLAM in providing essential scale information for accurate localization, we optimized the scale factor to minimize errors, thereby enhancing system performance [34], [36], [37], [39]. Due to the increase in memory consumption with distance when running Orbeez-SLAM, we restricted our comparison to trajectory data during periods when memory usage was below 16 GB.

A detailed comparison is presented in Table IV, where it can be observed that most systems exhibit underperformance in expansive environments, primarily due to drift. However, our algorithm shows a marked improvement in accuracy. Our algorithm was custom-developed to be highly tailored for specific scenarios, and thus, it is expected to yield optimal results. It also indicate that purely monocular systems [34], [36], [37], [39], in the absence of auxiliary data, tend to incur substantial localization errors. LiDAR systems, equipped with precise depth information, inherently have superior positioning accuracy compared with visual systems. As discussed in [60], they do not perform well in degraded scenarios. The LiDAR algorithms were unable to operate stably in the experiments conducted.

F. Resource Utilization in Algorithms

To comprehensively assess the computational efficiency of different algorithms, we tabulate the average processing time per frame for each algorithm when processing a new image. We particularly contrast the corner extraction time of our approach with the method presented in [2], it results are provided in Table V.

To assess the efficiency of computational resources, the performance of the algorithms was compared across three distinct platforms: NVIDIA Jetson AGX Xavier 4 GB (with an integrated Cortex-R5 CPU), Jetson Orin NX 8 GB (featuring a Cortex-A78AE CPU), and an Intel Core i7-8700K coupled with an NVIDIA RTX 3090 GPU. The intensive resource demands of certain algorithms, as delineated in [34], [36], [39],

TABLE VI
PROCESSING TIME PER FRAME FOR ALGORITHMS (ms)

	Cortex-A78AE	Cortex-R5	Intel i7
$\boldsymbol{I}_1$	**6**	**10**	**3**
$\boldsymbol{P}_1$	12	24	7
$\boldsymbol{P}_3$	16	29	8
VINS-MONO [5]	55	83	32
ORB-SLAM3 [23]	31	42	15
DSL [43]	344	397	174
Ref. [2]	112	188	58
GO-SLAM [36]	X	X	117
Orbeez-SLAM [34]	X	X	57
PVO [39]	X	X	176
DROID-SLAM [37]	X	X	45

TABLE VII
GPU MEMORY USAGE FOR DIFFERENT ALGORITHMS

Method	GPU Memory Usage
$\boldsymbol{I}_1$	3.2 GB
GO-SLAM [36]	17.9 GB
Orbeez-SLAM [34]	8 GB*
PVO [39]	11.9 GB
DROID-SLAM [37]	7.7 GB

and [37], particularly regarding GPU memory constraints, precluded their execution on ARM-based systems. Consequently, their evaluation was confined to computational latency metrics on the x86-64 architecture. Furthermore, Fast-LIO2 and NDT-LOAM were unable to function in the prescribed scenario, which, incidentally, corresponds to the degenerate cases reported in [60]. As a result, these algorithms were omitted from our testing protocol. Table VI presents a detailed comparison of the benchmarking results.

We also compared the GPU memory usage of algorithms that use a GPU, as excessive memory requirements can significantly impact the deployment of these algorithms, affecting the cost of industrial implementation. The comparative results can be found in Table VII. It is important to highlight that within the open-source Orbeez-SLAM [34] algorithm, the GPU memory consumption continuously increases as the vehicle moves.

Drawing from the experiments conducted, it becomes evident that our system outshines other SOTA localization solutions when tested against data from port environments. Our system demonstrates superiority not only in terms of accuracy but also in the efficiency of computational resource utilization. Our methodology stands out as a viable solution that is ready for industrial application and deployment.

G. Localization Performance With Varied Marker Geometries

As discussed earlier in Section II-B, a key reason for selecting rhombus-shaped markers is their widespread presence in standard traffic signage, which implies that the corresponding molds are readily available. This not only facilitates the production of markers but also enhances compatibility and recognition efficiency within practical applications of our system. However, geometrical shapes such as equilateral triangles, squares, and regular pentagons are not commonly adopted in our system's intended application scenarios, as they are not prevalent in the existing traffic sign system. Consequently, we are unable to obtain these shapes' markers in the real world.

TABLE VIII
COMPARISON OF LOCALIZATION PERFORMANCE FOR DIFFERENT MARKER GEOMETRIES

Shape	runtime(ms)	max T(m)	mean T(m)
triangles△	6	0.07	0.01
rhombus ◇	6	0.08	0.01
squares □	6	0.08	0.02
pentagon ⬠	6	0.05	0.01

In light of this, we resorted to simulation experiments to investigate the potential impact of these less common geometric shapes on system performance. This approach allowed us to precisely compare the effects of different marker shapes on localization accuracy and computational efficiency within a controlled environment. The simulation experiments were focused on analyzing the influence of various marker shapes on localization precision and their computational time requirements under static conditions.

The data presented in Table VIII summarize the results of simulation experiments designed to evaluate the impact of marker geometry on the localization performance of our system. From the results, it can be observed that the run-time for all the shapes is consistent at 6 ms, indicating that the computational efficiency of the system is shape-agnostic within the tested range. In terms of localization precision, the maximum and mean translation errors remain relatively low, with only slight variations among the different shapes. The rhombus and squares exhibit a marginally higher maximum translation error at 0.08 compared with the triangles and pentagons, which may be attributed to their specific geometrical properties. However, these differences are minimal, suggesting that the choice of marker geometry does not significantly affect the overall performance of the system.

These findings imply that the system's robustness to the variations in marker shape is high, and such robustness can be advantageous in practical scenarios where marker diversity might be required. This also hints at the possibility of incorporating a wider range of geometrical shapes into the system without compromising localization accuracy or computational efficiency.

IV. DISCUSSION

The proposed method presents a visual-based localization system tailored for autonomous driving applications. Our system, underscored by its robustness and reliability, has already seen deployment in commercial settings.

The experimental results indicate that our approach demands minimal precision in camera internal parameters, thereby substantially reducing camera calibration efforts. Notably, our system is computationally efficient, allowing ample processing time for the CPU. Furthermore, it can be adapted for various scenarios and is readily modifiable to accommodate different geometric road markers, such as other regular polygons, not just rhombuses.

Moreover, comparative experiments reveal the IPM method's superiority over the PnP method in both speed and precision. IPM is especially proficient in quantifying its uncertainty via Gaussian distribution, facilitating subsequent sensor fusion tasks. In situations with sparse matching data, provisioning ample heading information markedly enhances the accuracy of the localization results. While a static heading angle might yield larger residual errors, it invariably results in more precise positioning.

Limitations: While the system is advanced, there is room for improvement. Calibration requires specialized surveying tools, such as a total station. However, testing over two years has revealed that these parameters remain relatively stable. Although calibration is a one-time necessity per vehicle, it does curtail the system's full automation. Each vehicle mandates GPU utilization for segmentation. Even though NVIDIA's Orin offers substantial computational prowess, it comes at a premium cost. Adverse snowy conditions incapacitate the visual system, narrowing the algorithm's geographic applicability. Finally, while the system can localize using minimal corner pixels, it heavily relies on heading data from lane lines. Our method currently does not accommodate curved lane lines, making accurate positioning challenging in their absence. This article primarily underscores our system's practical application, but the underlying mathematical rationale merits further exploration.

V. Conclusion

In this article, we presented a reliable and precise localization system leveraging road markers and lanes. Using the IPM matrix, our method determines the vehicle's pose within the global frame. The experimental results highlight its potential not only in container ports but also in various autonomous settings. Our solution operates continuously in real-world ports, achieving positioning accuracies of up to 10 cm in most areas, satisfying the precision needs for the alignment between rail cranes and UGVs. This approach can be adapted to recognize different road marker shapes. Future directions encompass online updates of the visual database and 3-D structures for enhanced relocalization capabilities, especially in dynamic settings. Further plans include detecting damaged ground markers, integrating data from other sensors, and providing mathematical explanations for observed experimental behaviors.

Acknowledgment

The authors would like to thank equipment support from ZPMC, Shanghai, China.

References

[1] C. Mi, Y. Huang, C. Fu, Z. Zhang, and O. Postolache, "Vision-based measurement: Actualities and developing trends in automated container terminals," *IEEE Instru. Meas. Mag.*, vol. 24, no. 4, pp. 65–76, 2021.

[2] Y. Yu, P. Yun, B. Xue, J. Jiao, R. Fan, and M. Liu, "Accurate and robust visual localization system in large-scale appearance-changing environments," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 6, pp. 5222–5232, Dec. 2022.

[3] L. Carballo Piñeiro, M. Q. Mejia, and F. Ballini, "Beyond COVID-19: The future of maritime transport," *WMU J. Maritime Affairs*, vol. 20, pp. 127–133, May 2021.

[4] V. Corcoba, X. G. Pañeda, D. Melendi, R. García, L. Pozueco, and S. Paiva, "COVID-19 and its effects on the driving style of Spanish drivers," *IEEE Access*, vol. 9, pp. 146680–146690, 2021.

[5] T. Qin, P. Li, and S. Shen, "VINS-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[6] R. Elvira, J. D. Tardós, and J. Montiel, "ORBSLAM-Atlas: A robust and accurate multi-map system," in *Proc. IEEE IROS*, Nov. 2019, pp. 6253–6259.

[7] L. Zheng, Y. Zhu, B. Xue, M. Liu, and R. Fan, "Low-cost GPS-aided LiDAR state estimation and map building," in *Proc. IEEE Int. Conf. Imag. Syst. Techn. (IST)*, Dec. 2019, pp. 1–6.

[8] G. He, X. Yuan, Y. Zhuang, and H. Hu, "An integrated GNSS/LiDAR-SLAM pose estimation framework for large-scale map building in partially GNSS-denied environments," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–9, 2021.

[9] T. Li, L. Pei, Y. Xiang, X. Zuo, W. Yu, and T.-K. Truong, "P^3-LINS: Tightly coupled PPP-GNSS/INS/LiDAR navigation system with effective initialization," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023.

[10] B. Siciliano and O. Khatib, "Robotics and the handbook," in *Springer Handbook of Robotics*. Berlin, Germany: Springer, 2016, pp. 1–6.

[11] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, Mar. 2018.

[12] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Computer Vision—ECCV*. Berlin, Germany: Springer, 2014, pp. 834–849.

[13] R. Fan, U. Ozgunalp, B. Hosking, M. Liu, and I. Pitas, "Pothole detection based on disparity transformation and road surface modeling," *IEEE Trans. Image Process.*, vol. 29, pp. 897–908, 2020.

[14] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[15] I. Cvišic, I. Markovic, and I. Petrovic, "SOFT2: Stereo visual odometry for road vehicles based on a point-to-epipolar-Line metric," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 273–288, Feb. 2023.

[16] R. Fan, U. Ozgunalp, Y. Wang, M. Liu, and I. Pitas, "Rethinking road surface 3-D reconstruction and pothole detection: From perspective transformation to disparity map segmentation," *IEEE Trans. Cybern.*, vol. 52, no. 7, pp. 5799–5808, Jul. 2022.

[17] R. Fan, X. Ai, and N. Dahnoun, "Road surface 3D reconstruction based on dense subpixel disparity map estimation," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 3025–3035, Jun. 2018.

[18] J. Yuan, S. Zhu, K. Tang, and Q. Sun, "ORB-TEDM: An RGB-D SLAM approach fusing ORB triangulation estimates and depth measurements," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–15, 2022.

[19] Y. Liu, Y. Wu, and W. Pan, "Dynamic RGB-D SLAM based on static probability and observation number," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021.

[20] K. Sun et al., "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 965–972, Apr. 2018.

[21] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 298–304.

[22] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *Int. J. Robot. Res.*, vol. 36, no. 10, pp. 1053–1072, 2017.

[23] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[24] J. Zhang and S. Singh, "Visual-LiDAR odometry and mapping: Low-drift, robust, and fast," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 2174–2181.

[25] J. Yin, D. Luo, F. Yan, and Y. Zhuang, "A novel LiDAR-assisted monocular visual SLAM framework for mobile robots in outdoor environments," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–11, 2022.

[26] X. Zuo et al., "LIC-fusion 2.0: LiDAR-inertial-camera odometry with sliding-window plane-feature tracking," in *Proc. IEEE IROS*, Oct. 2020, pp. 5112–5119.

[27] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled LiDAR-visual-inertial odometry via smoothing and mapping," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 5692–5698.

[28] J. Lin and F. Zhang, "R^{3}LIVE++: A robust, real-time, radiance reconstruction package with a tightly-coupled LiDAR-inertial-visual state estimator," 2022, *arXiv:2209.03666*.
[29] S. Cao, X. Lu, and S. Shen, "GVINS: Tightly coupled GNSS-visual-inertial fusion for smooth and consistent state estimation," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2004–2021, Aug. 2022.
[30] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "iMAP: Implicit mapping and positioning in real-time," in *Proc. IEEE ICCV*, Oct. 2021, pp. 6229–6238.
[31] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proc. ECCV*, 2020, pp. 99–106.
[32] Z. Zhu et al., "NICE-SLAM: Neural implicit scalable encoding for SLAM," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12786–12796.
[33] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, "Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation," in *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Oct. 2022, pp. 499–507.
[34] C.-M. Chung et al., "Orbeez-SLAM: A real-time monocular visual SLAM with ORB features and NeRF-realized mapping," in *Proc. IEEE ICRA*, May 2023, pp. 9400–9406.
[35] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–15, Jul. 2022.
[36] Y. Zhang, F. Tosi, S. Mattoccia, and M. Poggi, "GO-SLAM: Global optimization for consistent 3D instant reconstruction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 3727–3737.
[37] Z. Teed and J. Deng, "DROID-SLAM: Deep visual SLAM for monocular, stereo, and RGB-D cameras," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 16558–16569.
[38] A. Rosinol, J. J. Leonard, and L. Carlone, "Probabilistic volumetric fusion for dense monocular SLAM," in *Proc. IEEE/CVF WACV*, Jan. 2023, pp. 3097–3105.
[39] W. Ye et al., "PVO: Panoptic visual odometry," in *Proc. IEEE/CVF CVPR*, Jun. 2023, pp. 9579–9589.
[40] H. Kloeden, D. Schwarz, E. M. Biebl, and R. H. Rasshofer, "Vehicle localization using cooperative RF-based landmarks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2011, pp. 387–392.
[41] Y. Zhu, B. Xue, L. Zheng, H. Huang, M. Liu, and R. Fan, "Real-time, environmentally-robust 3D LiDAR localization," in *Proc. IEEE IST*, Dec. 2019, pp. 1–6.
[42] H. Huang, H. Ye, Y. Sun, and M. Liu, "GMMLoc: Structure consistent visual localization with Gaussian mixture models," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5043–5050, Oct. 2020.
[43] H. Ye, H. Huang, and M. Liu, "Monocular direct sparse localization in a prior 3D surfel map," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 8892–8898.
[44] H. Huang, Y. Sun, H. Ye, and M. Liu, "Metric monocular localization using signed distance fields," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1195–1201.
[45] T. Qin, Y. Zheng, T. Chen, Y. Chen, and Q. Su, "A light-weight semantic map for visual localization towards autonomous driving," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11248–11254.
[46] J. Wu et al., "Quadratic pose estimation problems: Globally optimal solutions, solvability/observability analysis, and uncertainty description," *IEEE Trans. Robot.*, vol. 38, no. 5, pp. 3314–3335, Oct. 2022.
[47] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.
[48] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
[49] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial CNN for traffic scene understanding," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2018, vol. 32, no. 1, pp. 1–8.
[50] K.-L. Low and A. Ilie, "Computing a view frustum to maximize an object's image area," *J. Graph. tools*, vol. 8, no. 1, pp. 3–15, 2003.
[51] Y. Cai, W. Xu, and F. Zhang, "IKD-Tree: An incremental K-D tree for robotic applications," 2021, *arXiv:2102.10808*.
[52] C. Leys, O. Klein, Y. Dominicy, and C. Ley, "Detecting multivariate outliers: Use a robust variant of the Mahalanobis distance," *J. Exp. Social Psychol.*, vol. 74, pp. 150–156, Jan. 2018.
[53] K. S. Chong and L. Kleeman, "Accurate odometry and error modelling for a mobile robot," in *Proc. Int. Conf. Robot. Autom.*, vol. 4, 1997, pp. 2783–2788.
[54] T. D. Barfoot, *State Estimation for Robot*. Cambridge, U.K.: Cambridge Univ. Press, 2017.
[55] A. Stroupe, M. Martin, and T. Balch, "Distributed sensor fusion for object position estimation by multi-robot systems," in *Proc. IEEE ICRA*, vol. 2, May 2001, pp. 1092–1098.
[56] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.
[57] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 8, pp. 930–943, Aug. 2003.
[58] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes," in *Proc. IEEE ICRA*, May 2016, pp. 4304–4311.
[59] S. Chen et al., "NDT-LOAM: A real-time LiDAR odometry and mapping with weighted NDT and LFA," *IEEE Sensors J.*, vol. 22, no. 4, pp. 3660–3671, Feb. 2022.
[60] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.

Bohuan Xue (Graduate Student Member, IEEE) received the B.Eng. degree in computer science and technology from the College of Mobile Telecommunications, Chongqing University of Posts and Telecommunications, Chongqing, China, in 2018. He is currently pursuing the Ph.D. degree in electrical engineering with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, SAR, China.

His research interests include simultaneous localization and mapping (SLAM), computer vision, and 3-D reconstruction.

Xiaoyang Yan was born in Zhengzhou, Henan, China, in 1996. He received the B.Sc. degree from Zhengzhou University, Zhengzhou, China, in 2018, and the M.S. degree in telecommunications from the Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2019, where he is currently pursuing the Ph.D. degree.

His research mainly focuses on computer vision for autonomous driving.

Jin Wu (Member, IEEE) received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China. He is currently pursuing the Ph.D. degree with the Hong Kong University of Science and Technology (HKUST), Hong Kong.

In 2013, he was a Visiting Student at Groep T, KU Leuven, Leuven, Belgium. From 2019 to 2020, he was with Tencent Robotics X. He has published more than 140 papers in representative journals and conferences.

Mr. Wu was named in Stanford University List of Top 2% Scientists Worldwide in 2020, 2021, and 2022, according to contributions in aerospace engineering and robotics.

Jintao Cheng received the bachelor's degree from the School of Physics and Telecommunications Engineering, South China Normal University, Foshan, China, in 2021.

His research interests include computer vision, simultaneous localization and mapping (SLAM), and deep learning.

Jianhao Jiao (Member, IEEE) received the B.Eng. degree in instrument science from Zhejiang University, Hangzhou, China, in 2017, and the Ph.D. degree from the Department of Electronic and Computer Engineering, the Hong Kong University of Science and Technology, Hong Kong, China, in 2021, under the supervision of Prof. Ming Liu.

He is currently a Senior Research Fellow with the Department of Computer Science, University College London, London, U.K.

Haoxuan Jiang received the B.Eng. degree in measurement control technology and instruments from Shenzhen University, Shenzhen, China, in 2017, and the M.S. degree in mechanical and automation engineering from The Chinese University of Hong Kong, Hong Kong, in 2018. He is currently pursuing the Ph.D. degree in robotics and autonomous systems with the Hong Kong University of Science and Technology (HKUST) (Guangzhou), Guangzhou, China.

His research interests include reinforcement learning for robotics, simultaneous localization and mapping (SLAM), and computer vision.

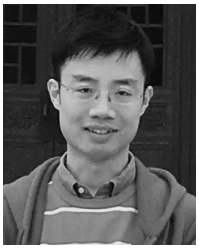
Rui Fan (Senior Member, IEEE) received the B.Eng. degree in automation from Harbin Institute of Technology, Harbin, China, in 2015, and the Ph.D. degree in electrical and electronic engineering from the University of Bristol, Bristol, U.K., in 2018, under the supervision of Prof. John G. Rarity and Prof. Naim Dahnoun.

He worked as a Research Associate (supervisor: Prof. Ming Liu) at the Hong Kong University of Science and Technology, Hong Kong, from 2018 to 2020, and a Post-Doctoral Scholar-Employee (supervisors: Prof. Linda M. Zangwill and Prof. David J. Kriegman) at the University of California San Diego, San Diego, CA, USA, from 2020 to 2021. He began his faculty career as a Full Research Professor with the College of Electronics and Information Engineering, Tongji University, Shanghai, China, in 2021, and was then promoted to a Full Professor at the same college, as well as at the Shanghai Research Institute for Intelligent Autonomous Systems, in 2022. His research interests include computer vision, deep learning, and robotics.

Prof. Fan served as a Senior Program Committee Member for AAAI'23/24. He was honored by being included in the Stanford University List of Top 2% Scientists Worldwide in both 2022 and 2023, recognized on the Forbes China List of 100 Outstanding Overseas Returnees in 2023, and acknowledged as one of the Xiaomi Young Talents in 2023. He is the General Chair of the AVVision Community and organized several impactful workshops and special sessions in conjunction with WACV'21, ICIP'21/22/23, ICCV'21, and ECCV'22. He served as an Associate Editor for ICRA'23, IROS'23, and IROS'24.

Ming Liu (Senior Member, IEEE) received the B.A. degree in automation from Tongji University, Shanghai, China, in 2005, and the Ph.D. degree from the Department of Mechanical and Process Engineering, ETH Zurich, Zurich, Switzerland, in 2013, under the supervision of Prof. Roland Siegwart.

During his master's study with Tongji University, he stayed one year with Erlangen-Nunberg University, Erlangen, Germany, and Fraunhofer Institute IISB, Erlangen, as a Master Visiting Scholar. He is currently an Associate Professor with the Thrust of Robotics and Autonomous Systems, Hong Kong University of Science and Technology (Guangzhou), Nansha, Guangzhou, China. His research interests include dynamic environment modeling, deep learning for robotics, 3-D mapping, machine learning, and visual control.



Chengxi Zhang (Member, IEEE) was born in 1990, Shandong, China. He received the B.S. and M.S. degrees in microelectronics and solid-state electronics from the Harbin Institute of Technology, Harbin, China, in 2012 and 2015, respectively, and the Ph.D. degree in control science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2019.

He is currently an Associate Professor with the School of Internet of Things, Jiangnan University, Wuxi, China. His research interests include embedded system software and hardware design, information fusion, and control theory.

Dr. Zhang is a Board Member of the Shanghai Jiao Tong University Ph.D. Alumni Association of the School of Aeronautics and Astronautics, from 2019 to 2024.