



Article

Reducing Redundancy in Maps without Lowering Accuracy: A Geometric Feature Fusion Approach for Simultaneous Localization and Mapping

Feiya Li ¹, Chunyun Fu ^{1,*} , Dongye Sun ¹, Hormoz Marzbani ²  and Minghui Hu ¹

¹ College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing 400044, China; fairlee@cqu.edu.cn (F.L.); dysun@cqu.edu.cn (D.S.); hu_ming@cqu.edu.cn (M.H.)

² School of Engineering, RMIT University, Bundoora, VIC 3083, Australia; hormoz.marzbani@rmit.edu.au

* Correspondence: fuchunyun@cqu.edu.cn

Abstract: Geometric map features, such as line segments and planes, are receiving increasing attention due to their advantages in simultaneous localization and mapping applications. However, large structures in different environments are very likely to appear repeatedly in several consecutive time steps, resulting in redundant features in the final map. These redundant features should be properly fused, in order to avoid ambiguity and reduce the computation load. In this paper, three criteria are proposed to evaluate the closeness between any two features extracted at two different times, in terms of their included angle, feature circle overlapping and relative distance. These criteria determine whether any two features should be fused in the mapping process. Using the three criteria, all features in the global map are categorized into different clusters with distinct labels, and a fused feature is then generated for each cluster by means of least squares fitting. Two competing methods are employed for comparative verification. The comparison results indicate that using the commonly used KITTI dataset and the commercial software PreScan, the proposed feature fusion method outperforms the competing methods in terms of conciseness and accuracy.

Keywords: mapping; geometric features; feature fusion



Citation: Li, F.; Fu, C.; Sun, D.; Marzbani, H.; Hu, M. Reducing Redundancy in Maps without Lowering Accuracy: A Geometric Feature Fusion Approach for Simultaneous Localization and Mapping. *ISPRS Int. J. Geo-Inf.* **2023**, *12*, 235. <https://doi.org/10.3390/ijgi12060235>

Academic Editors: Sisi Zlatanova and Wolfgang Kainz

Received: 28 March 2023

Revised: 26 May 2023

Accepted: 30 May 2023

Published: 7 June 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Simultaneous localization and mapping (SLAM) is one of the major challenges in the field of autonomous driving and robotics [1]. In the process of SLAM, two parallel tasks are simultaneously dealt with, including localization and mapping [2]. The latter one is normally referred to as the process of constructing a certain type of representation of the surrounding environment, in which the autonomous vehicles or robots perform localization and path planning.

In recent years, many works have been proposed to tackle the mapping problem for various map types, such as [3–7]. In the existing literature, the maps used for SLAM can be categorized into the following three major types: occupancy grid maps, topological maps and geometric feature maps (e.g., point feature maps, line feature maps and plane feature maps). Due to the lower memory usage compared to occupancy grid maps, and higher accuracy compared to topological maps, geometric feature maps have received significantly more attention in recent years. Among the various types of map features, point features are normally used to provide geometric constraints for precise localization [8]. However, it should be noted that the use of point features often leads to low-quality maps, because the real environment cannot be faithfully represented by points alone and important information about the environment may be lost in point feature maps. To tackle this issue, more sophisticated map features—line features and plane features—have been proposed in the literature [9,10].

The use of line features/plane features has shown several obvious advantages over the traditional approaches. It has been pointed out in [11] that the employment of line features/plane features yields significant advantages in terms of storage space and computational load. In addition, the adverse effects of sensor noise can be sufficiently suppressed if plane features/line features are used, since they are normally extracted from a large number of points [12,13]. Moreover, for mapping environments with abundant flat structures, plane features can be employed to better constrain the camera motion and effectively suppress drift [14]. This paper contributes to the field by addressing the feature fusion problem of these two important types of geometric features.

1.1. Literature Review

In recent years, a number of methods have been proposed in relevant literature for identifying and merging redundant map features (e.g., line features and plane features), aiming to represent the structure outlines in a more clear and concise manner.

Line features are commonly used to represent environments in which straight or flat structures are dominant. These features are often observed in maps constructed by means of LiDAR sensors [15], cameras [16] and RGB-D sensors [17]. In the existing literature, fusion algorithms for line features can be categorized into the following two major types: offline algorithms [18–20] and online algorithms [21–24]. In offline algorithms, line features obtained in the sensor coordinate system are first transformed to the global coordinate system, and then they are clustered and merged in the global coordinate system. In comparison, online algorithms achieve map compression normally by means of line feature association/matching; in the meantime, the vehicle/robot pose is simultaneously obtained in real-time. Then, new features extracted in the current frame are used to update the global map.

Sarkar et al. [18] proposed an offline approach to fuse line features. In this method, angular clustering and spatial clustering are applied to all line features, and then the features in each cluster are merged to a line of infinite length. The endpoints of all features in one cluster are projected onto this infinite line, and the two farthest points, making the longest length form the resulting (fused) feature. Elseberg et al. [19] put forward an offline fusion method and used similarity measures (i.e., the angular distance and the spatial distance) to cluster line features, and the merged feature direction for a cluster is the averaged direction of all constituent features in that cluster. Amigoni and Li [20] reproduced the A-V approach [21] for merging line features. This method fuses line segments by checking the maximum Euclidean distance from endpoints to the supporting line against a threshold and ensuring non-empty intersections of projections within the set of line features.

Gomez-Ojeda et al. [22] proposed an online method to match current and previous features by comparing their directions, lengths, and endpoint positions, thereby achieving feature fusion for key frames. Liu et al. [23] achieved an initial match between line features in the image plane and those in the global coordinate system using bundle adjustment, and then obtained the endpoints of fused features by means of the least squares method. Gao et al. [24] proposed an online approach in which line features with 3 degrees of freedom are used in an enhanced EKF-SLAM framework to construct sparse feature maps. Wen et al. [25] introduced an online merging method for line features. In this method, three measures (heading deviation, separation distance and overlap) are employed to evaluate the spatial proximity between the previous global feature map and line features of the current frame.

Plane features have been employed in the existing LiDAR SLAM and RGB-D SLAM solutions [26–30]. Along with the development of plane feature-based SLAM methods, plane feature fusion algorithms have also been invented and developed accordingly.

Wang et al. [26] transformed the plane features obtained in consecutive frames to the global coordinate system, and calculated the Mahalanobis distances between plane pairs. Then, a constant threshold was chosen to check whether any pair of planes should

represent the same structure surface. Lastly, the least squares method was employed to fuse the features obtained in the current frame and the global map from the previous step. Hsiao et al. [27] presented a fusion method to match new plane features with existing plane features. For each plane feature to be fused, three criteria should be evaluated: (1) the normal value of the plane feature, (2) the distance between the plane feature and origin of the coordinate system, and (3) the residual of the plane model. Ćwian et al. [28] proposed a series of criteria to determine whether two plane features should be fused, including the angle between two plane features, the mean residual error, the minimum point-to-point distance between two plane features, and the planarity and curvature parameters. Grant et al. [29] proposed a new plane fusion method, taking into account the constraints of having similar normals and distance values between the two planes. When these constraints are met, the two planes are projected onto their mean plane obtained by averaging their normals and offsets, and the intersection area and union area of the two projected planes are calculated. If the ratio of the intersection area over the area of the smaller projected plane is over a threshold (or the ratio of the intersection area over the union area is greater than a threshold), then the two plane features are associated and must be merged in the next step. Pan et al. [30] proposed to merge newly detected plane features with previous planes using surface normal similarity and spatial proximity criteria. By detecting and merging matched small plane segments from multiple RGB-D frames, a large indoor plane can then be obtained.

1.2. Problem Statement

In general, line features and plane features are superior to point features, since they not only reduce the amount of data involved in mapping, but also provide better geometric information about the environment [31]. Although great efforts have been made in fusing line features and plane features, challenges still remain in the accuracy and conciseness of the fused feature maps. Existing methods frequently falter in efficiently merging features detected at different time points, resulting in redundancy and potential clarity issues in the resulting maps. Furthermore, the computational burden of handling these redundant features can slow down the mapping process. These challenges present an obvious research gap that needs to be addressed for improvement in this field.

Given this context, to effectively utilize these two types of features, the ‘feature fusion’ problem must be properly solved to merge the features extracted at different times. The ‘feature fusion’ problem in this context is referred to as the process of merging the features that represent the same structures in the environment but are extracted at different (and mostly consecutive) times. For example, as shown in Figure 1, when an autonomous vehicle navigates through an unknown environment, big structures (e.g., building walls) are very likely to appear repeatedly at several time steps. As a result, redundant features that originate from exactly the same structure may exist at different times. Therefore, appropriate measures must be taken to fuse these redundant features, for the following two important reasons: (1) avoiding ambiguity and enhancing mapping accuracy, and (2) reducing the computation load and accelerating the mapping process.

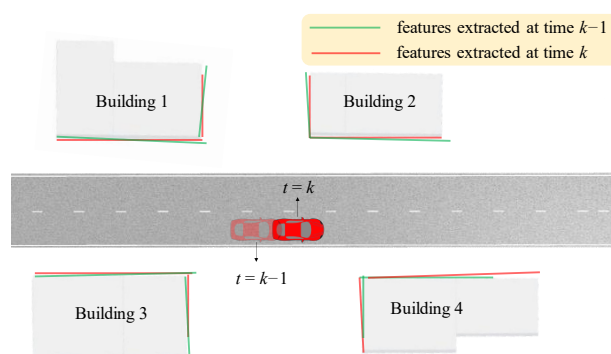


Figure 1. Features extracted at two consecutive time steps.

1.3. Original Contributions

To tackle the above problem, in this paper we propose a novel feature fusion algorithm for geometric feature maps, which effectively merges the repeating features extracted at different time steps. Specifically, the proposed algorithm is mainly applicable to two types of geometric features—line features and plane features. In this study, the plane features are assumed to be perpendicular to the ground and are projected onto the ground plane. Hence, the line features and plane features can be uniformly treated using the proposed algorithm.

In this paper, three important criteria are proposed to evaluate whether the features extracted at two different times represent the same structure, which are as follows: (1) a small included angle, (2) large overlapping area of feature circles, and (3) small relative distance between features. Features that satisfy the above three criteria simultaneously are merged by means of least squares fitting, namely, the repeating features extracted at two different times are fused to form a single feature. By this means, the mapping accuracy and mapping speed are both improved, as only the fused features remain in the constructed feature map.

The effectiveness of the proposed algorithm is verified through comparative studies with two typical feature fusion methods proposed in [18,31]. The principles of the competing methods are revisited in Section 2. The fusion performances of these three methods are compared using the KITTI dataset [32] and the commercial software PreScan. It is shown in the fusion results that the proposed algorithm outperforms the competing methods in terms of conciseness and accuracy.

1.4. Outline of the Paper

The rest of the paper is organized as follows. The principles of the two competing methods are introduced in Section 2. The details of the proposed feature fusion algorithm are elaborated in Section 3. The feature fusion results are compared and discussed in Section 4. Conclusions and future works are given in Section 5.

2. Background

In this paper, two line feature fusion methods [18,31] are employed for comparison purposes. The first method is devised based on the principle of mean shift clustering [31]. The second method is a more recent and advanced feature fusion algorithm designed based on density clustering, in which the typical mean shift clustering is also involved. In Section 4, the proposed feature fusion method is compared with these two competing methods in terms of conciseness and accuracy, in order to evaluate the effectiveness of the proposed feature fusion algorithm.

2.1. Feature Fusion Based on Mean Shift Clustering—Competing Algorithm 1

In this method, 2D line segments are employed as the form of map representation, and the mean shift clustering algorithm [31] is adopted to merge the perceptually similar line segments into a single instance, thereby significantly downsizing the line feature dataset while providing better geometric information of the map. In the resulting global 2D map, each set of the originally redundant linear features is represented (and replaced) by a single line segment without any ambiguity.

Note that in [31], the mean shift clustering algorithm is applied sequentially to cluster the line segments, based on the feature orientations and feature centers. Then, each cluster of features is merged by projecting individual line segments onto a representative segment located in the middle, leading to the final fused feature for that cluster. Interested readers are referred to [31] for more details of this feature fusion method.

2.2. Feature Fusion Based on Density Clustering—Competing Algorithm 2

The second competing algorithm was designed based on density clustering [18]. The core of this approach is two successive density-based clustering steps for identifying the

same clusters of line features. Note that in this approach, the typical mean shift clustering algorithm is also involved in the clustering steps.

In the two successive clustering steps, the first step is based on feature orientations. After computing the orientation of each line feature, the mean shift clustering algorithm based on angular orientations is applied to classify the line features extracted from all individual scans. The second step is based on feature spatial proximity. Two indicators are employed to evaluate the proximity of line features—lateral separation and longitudinal overlap. The line features that are considered to be close in terms of both orientation and spatial proximity are merged to produce a resulting (fused) line feature by taking into account the weight (i.e., length) of each constituent line feature.

3. Proposed Feature Fusion Method

In this section, we propose a novel feature fusion method for merging line features and plane features. Note that the majority of plane features extracted from the environments result from large structures such as building walls. Assuming that the ground is flat, these plane features are at large perpendicular to the ground and can be treated as lines if they are projected onto the ground plane. Hence, the fusion method proposed in this section can be uniformly applied to both line features and plane features.

As mentioned in Section 1.2, when an autonomous vehicle navigates through an unknown environment, it is highly likely that big flat structures, such as building walls, repeatedly appear at several time steps, and as a result, redundant features (which originate from exactly the same structure) are repeatedly extracted at these time steps. This problem has been schematically shown in Figure 1. Ideally, in the final constructed map, only one fused feature should be retained to represent each flat structure (e.g., a building wall) in the environment. To achieve this goal, three important criteria for feature fusion are proposed in our method.

3.1. Criteria for Feature Fusion

The following three criteria are proposed to determine whether the features extracted at two different times should be merged. These criteria are based on an intuitive principle that only the features that are ‘close enough’ should be fused.

3.1.1. Criterion 1—Small Included Angle

The first criterion is to evaluate the closeness of the two extracted features in terms of the included angle between them. Any two features extracted at time k and $k - i$ ($i = 1, 2, \dots, k - 1$) can be expressed in the following form:

$$\Gamma_k = \{(x_k^a, y_k^a), (x_k^b, y_k^b)\} \quad (1)$$

$$\Gamma_{k-i} = \{(x_{k-i}^a, y_{k-i}^a), (x_{k-i}^b, y_{k-i}^b)\} \quad (2)$$

where (x_k^a, y_k^a) and (x_k^b, y_k^b) represent the endpoint coordinates of the line feature at time k , and (x_{k-i}^a, y_{k-i}^a) and (x_{k-i}^b, y_{k-i}^b) denote the endpoint coordinates of the line feature at time $k - i$.

The equations for the above two line features in the x - y plane (i.e., the ground plane) of the global coordinate system are written as:

$$A_k x + B_k y + C_k = 0 \quad (3)$$

$$A_{k-i} x + B_{k-i} y + C_{k-i} = 0 \quad (4)$$

where $A_k, B_k, C_k, A_{k-i}, B_{k-i}$ and C_{k-i} denote the coefficients of these two linear equations. Then, the included angle β between these two line features is given by

$$\beta = \cos^{-1}\left(\frac{|A_k A_{k-i} + B_k B_{k-i}|}{\sqrt{A_k^2 + B_k^2} \cdot \sqrt{A_{k-i}^2 + B_{k-i}^2}}\right) \quad (5)$$

where $\beta \in [0, \frac{\pi}{2})$.

If two line features originate from exactly the same flat structure, then the following necessary condition must be met:

$$0 \leq \beta \leq \delta \quad (6)$$

where δ denotes the threshold for the magnitude of the included angle β . This criterion evaluates the closeness of the features in terms of their orientations in the global coordinate system.

3.1.2. Criterion 2—Large Overlapping Area of Feature Circles

The first criterion can be satisfied as long as the features involved are generally parallel, even if they are located far apart. For example, the features that represent the buildings on the two sides of a straight road can be easily found parallel to each other. Hence, these features satisfy the constraint imposed by Criterion 1, although they are separated by the road and located far from each other. This problem calls for more stringent constraints on the similarities of the features to be fused.

In the second criterion, the concept of feature circles is introduced to solve the above problem by evaluating the level of feature circle overlapping. For line features, the feature circle is defined as the circle that takes the midpoint of the line segment as the center, and the length of the line segment as the diameter. Figure 2 shows an example of the feature circles resulting from one single scan. In this example, 10 line features are present and each of these features is assigned a label, ranging from 1 to 10. In Figure 3, the feature circles resulting from two consecutive scans are drawn, with feature labels ranging from 1 to 13.

It can be observed that some features repeatedly appear in two consecutive scans and they are very closely located, such as feature 7. These features are referred to as ‘survived features’, and they are very likely to have originated from exactly the same structures. To reflect the true locations of the real structures, the survived features should be properly merged in the map. By observing Figures 2 and 3, intuitively, features 2, 3, 4, 5, 6, 7, and 10 are expected to be survived features. The expressions for survived features at different time steps are given by Equations (1) and (2).

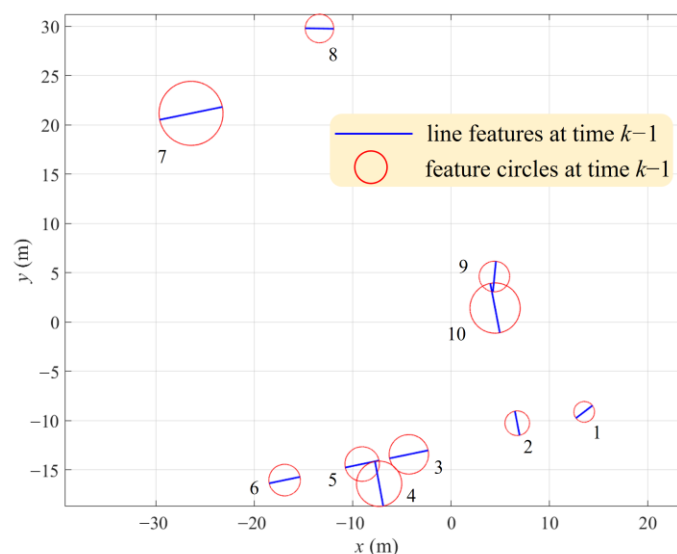


Figure 2. Example of feature circles resulting from one single time step.

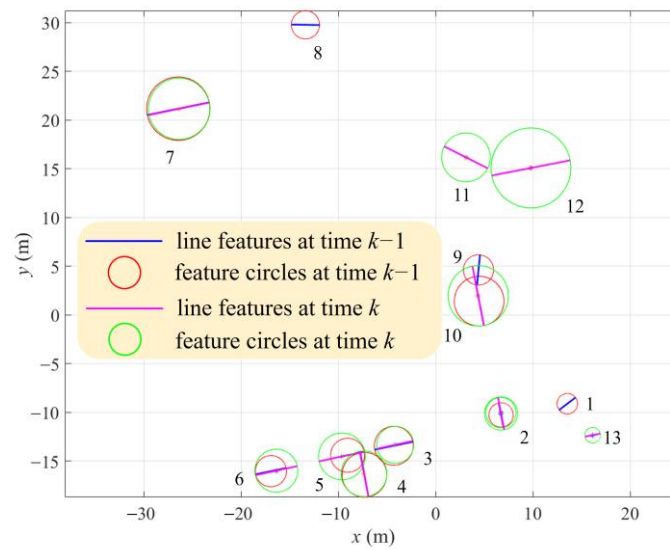


Figure 3. Example of feature circles resulting from two consecutive time steps.

The radii of the feature circles at time k and time $k-i$ are as follows:

$$R_k = \frac{1}{2} \sqrt{(x_k^a - x_k^b)^2 + (y_k^a - y_k^b)^2} \quad (7)$$

$$R_{k-i} = \frac{1}{2} \sqrt{(x_{k-i}^a - x_{k-i}^b)^2 + (y_{k-i}^a - y_{k-i}^b)^2} \quad (8)$$

The level of feature circle overlapping is employed in this study as an indicator to evaluate the extent of feature closeness. Specifically, the following overlapping indicator ε is defined as follows:

$$\varepsilon = \frac{S_{\text{overlap}}}{S_{\text{min}}} \quad (9)$$

where S_{overlap} denotes the overlapping area of the feature circles at time k and time $k-i$, and S_{min} represents the area of the smaller feature circle.

Let us denote the distance between these two circle centers by d ; apparently, if $R_k + R_{k-i} \leq d$, then we have $\varepsilon = 0$, and if $|R_k - R_{k-i}| \geq d$, then $\varepsilon = 1$ holds. When $|R_k - R_{k-i}| < d < R_k + R_{k-i}$, the indicator ε takes a value between 0 and 1. Note that in this case, the two feature circles intersect and an overlapping area is formed, as schematically shown in Figure 4. It can be readily proven that the overlapping area, S_{overlap} , is given by the following equation:

$$S_{\text{overlap}} = \alpha_k R_k^2 + \alpha_{k-i} R_{k-i}^2 - d R_k \sin \alpha_k \quad (10)$$

where α_k represents the magnitude of angle $\angle MO_k O_{k-i}$ in radians, and α_{k-i} denotes the magnitude of angle $\angle MO_{k-i} O_k$ in radians. Note that in Equation (10), the first two terms reflect the areas of sectors $MO_k N$ and $MO_{k-i} N$ respectively, and the last term stands for the area of quadrangle $MO_k NO_{k-i}$.

With the above definition, if two features are considered survived features and to be merged, the following inequality is enforced as a necessary condition for feature fusion:

$$\gamma \leq \varepsilon \leq 1 \quad (11)$$

where γ is the lower boundary of the indicator ε , which is dependent on the environments to be explored.

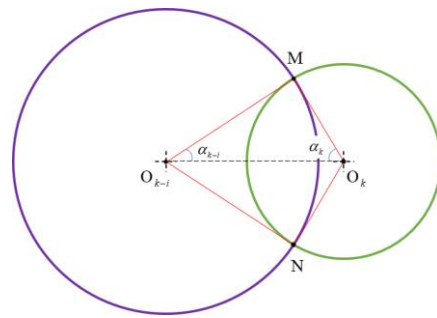


Figure 4. An example of two intersecting feature circles. The purple and green circles respectively represent the feature circles at time $k - i$ and time k .

3.1.3. Criterion 3—Small Relative Distance between Features

This criterion is proposed to eliminate the adverse influence of trivial features extracted from the environment. This purpose cannot be achieved by applying only Criterion 1 and Criterion 2. The trivial features may originate from small structures in the environment rather than large buildings, such as fences on the road side. These trivial features should not be fused in the map, as they could lead to significant deviations of the fused features from the true locations of large structures (e.g., buildings) in the environment.

For example, in Figure 5, it is assumed that feature FG originates from a large building and feature HJ results from a small structure on the road. Apparently, features FG and HJ satisfy both Criteria 1 and 2, and they would be fused if Criterion 3 was not enforced. However, the fusion of FG and HJ could result in a fused feature $F'G'$ that is located far from the major structure FG, which in turn leads to serious problems for localization or path planning. Therefore, measures should be taken to eliminate the adverse effect caused by trivial features such as HJ.

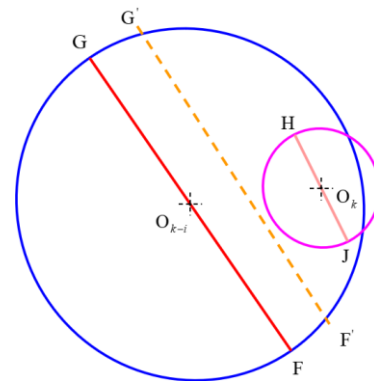


Figure 5. An illustration of the effect of a trivial feature on the fusion result. The blue and purple circles respectively represent the feature circles at time $k - i$ and time k .

To tackle the above problem, in this criterion, the distance from the center of one feature (e.g., point O_k) to the other feature (e.g., line segment FG) is employed as an indicator. In the case shown in Figure 5, denoting the distance from O_k to FG by $d_{k \rightarrow k-i}$ and that from O_{k-i} to HJ by $d_{k-i \rightarrow k}$, the following indicator can be defined:

$$d_{\max} = \max(d_{k \rightarrow k-i}, d_{k-i \rightarrow k}) \quad (12)$$

Then, the inequality below is enforced as a necessary condition for feature fusion:

$$0 \leq d_{\max} \leq \eta \quad (13)$$

where η is the upper boundary for the indicator d_{\max} , which is dependent on the environments to be explored.

To sum up, if any two features extracted at time k and time $k - i$ satisfy the above three criteria at the same time, then these two features are considered to have originated from the same structure. Such features will later be fused using the approach proposed in the following section.

3.2. Feature Fusion Strategy

By means of the three criteria proposed in Section 3.1, we are able to determine whether the features extracted at two different times should be merged. Note that for each time step, it is quite common to obtain more than one feature from the feature extraction algorithm. Before the feature fusion process takes place, each feature extracted at time k is compared with those extracted at time $k - i$, using the above proposed criteria. If any two features extracted at two different times satisfy all three criteria simultaneously, then they are classified in the same cluster and a unique label (i.e., an integer) is assigned to that cluster. Repeating this process sequentially, all features in the global map can be categorized into different clusters with distinct labels. Note that a new label is assigned to a cluster only when the first two features (which satisfy the three criteria) for that cluster are found, and in the following process, the newly added features will be attached the same label as the others in the same cluster.

For major structures in the environment (e.g., buildings), they repeatedly appear in multiple scans and the number of repetitions can be quite large. In comparison, trivial features such as fences appear in significantly fewer scans due to their considerably smaller dimensions. For this reason, the number of features in each cluster is evaluated before feature fusion takes place, in order to retain only the significant (i.e., sufficiently large) features in the map. In other words, if the number of features in a cluster is too low, then this cluster is neglected, as the trivial features only appear in a limited number of scans.

The feature fusion process can be completed using the straightforward least squares fitting method. Specifically, the endpoints of all features in a cluster are employed as the data points to be fitted. Applying least squares fitting, the mathematical expression (i.e., the linear equation) for the fused feature in the x - y plane (i.e., the ground plane) of the global coordinate system can be obtained. In order to determine the new endpoints for the fused feature, the endpoints of the original features in this cluster are all projected onto the fused line. Then, the two projected points that maximize the length of the fused line segment are chosen as the new endpoints. This process is schematically shown in Figure 6.

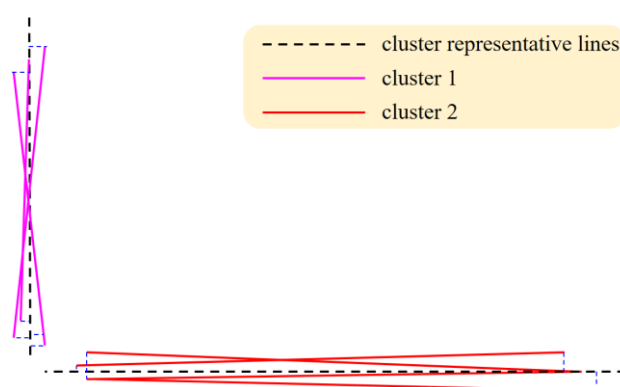


Figure 6. Feature fusion process.

The last step in the feature fusion process is to optimize the fused features that form the building edges. It can be observed in Figure 7a,b that the endpoints of some fused features are closely located (but are not coincident). Intuitively, these features reflect the building edges (corners) and their endpoints should indeed coincide. In this study, if the distance between two endpoints is within a certain threshold, then these two endpoints are replaced by a new endpoint located exactly in the middle of the two original endpoints. By this means, the original intersecting or disconnected lines are now replaced by connected

line segments, leading to smooth and continuous building profiles without any protrusions or gaps, as shown in Figure 7c,d.

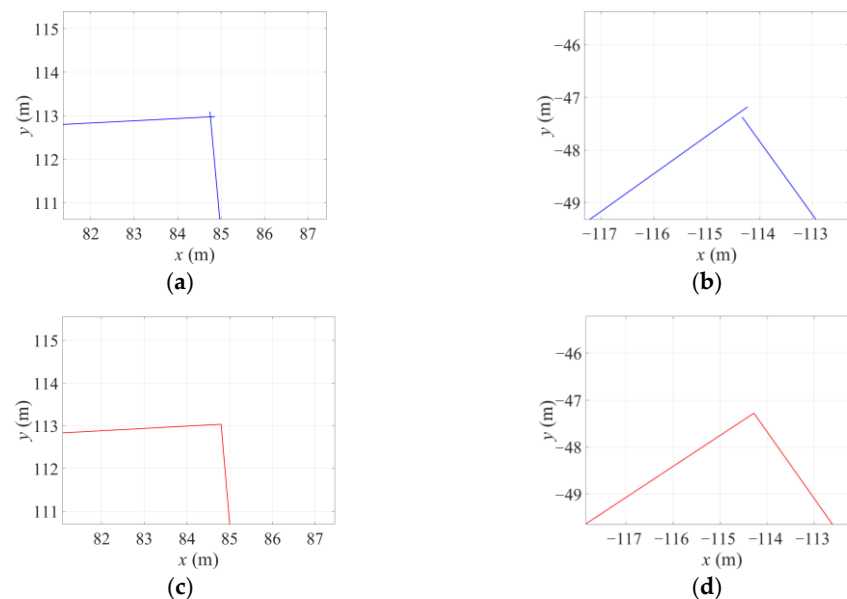


Figure 7. Fused features before and after optimization. (a) Fused features before optimization; (b) fused features before optimization; (c) fused features after optimization; (d) fused features after optimization.

4. Results and Discussion

In this section, the effectiveness of the proposed feature fusion method is verified through comparative studies with two typical feature fusion methods introduced in Section 2.2. Firstly, the commonly used KITTI dataset [32] is employed for this verification. Specifically, the dataset numbered '2011_09_30_drive_0018_sync' that reflects a residential environment is used. In addition, to provide quantitative comparison results, the commercial software PreScan is also employed in this study for further verification. All the comparative experiments were run on a desktop equipped with a 3.4 GHz Intel Xeon CPU and a 64 GB RAM.

Two types of data in the KITTI dataset are employed in this section for comparative verification: point clouds, as well as coordinates (latitude and longitude) and the heading angle of the vehicle. The point clouds in the dataset were collected by a 64-beam HDL-64E Velodyne LiDAR, which features a 0.09 degree angular resolution, 2 cm distance resolution and 10 Hz frequency [32]. The field of view of this LiDAR is 360° horizontal and 26.8° vertical, while the range of this sensor is 120 m [32]. The coordinates and heading angle of the vehicle were obtained using an OXTS RT3003 inertial and GPS navigation system. This system features a 100 Hz data output rate, 0.02 m positioning resolution and 0.1° heading resolution [32].

The proposed method is compared with the competing methods in terms of two major aspects: conciseness and accuracy. For the former one, we evaluate the number of features needed to represent the flat structures in the map. A lower number of features indicates a smaller storage space and less processing time. For the latter one, we check the closeness between the fused features and the 'ground truth', i.e., the structure locations obtained from Google maps or commercial software PreScan. The accuracy of fused features is first demonstrated graphically based on Google maps, and then the accuracy is quantified by the sum of the distances between the endpoints obtained by the fusion algorithm and the corresponding vertex, using an artificial environment constructed in PreScan.

4.1. Comparison in Terms of Conciseness

Figure 8a demonstrates a global map without feature fusion, in which all features extracted at different times are directly plotted. It can be easily observed that redundant

features are locally concentrated, as they originate from the same flat structures. Figure 8b shows a global map consisting of fused plane features, resulting from the proposed feature fusion method. In this figure, almost all redundant planes have been fused, which yields a rather neat feature map compared to Figure 8a.

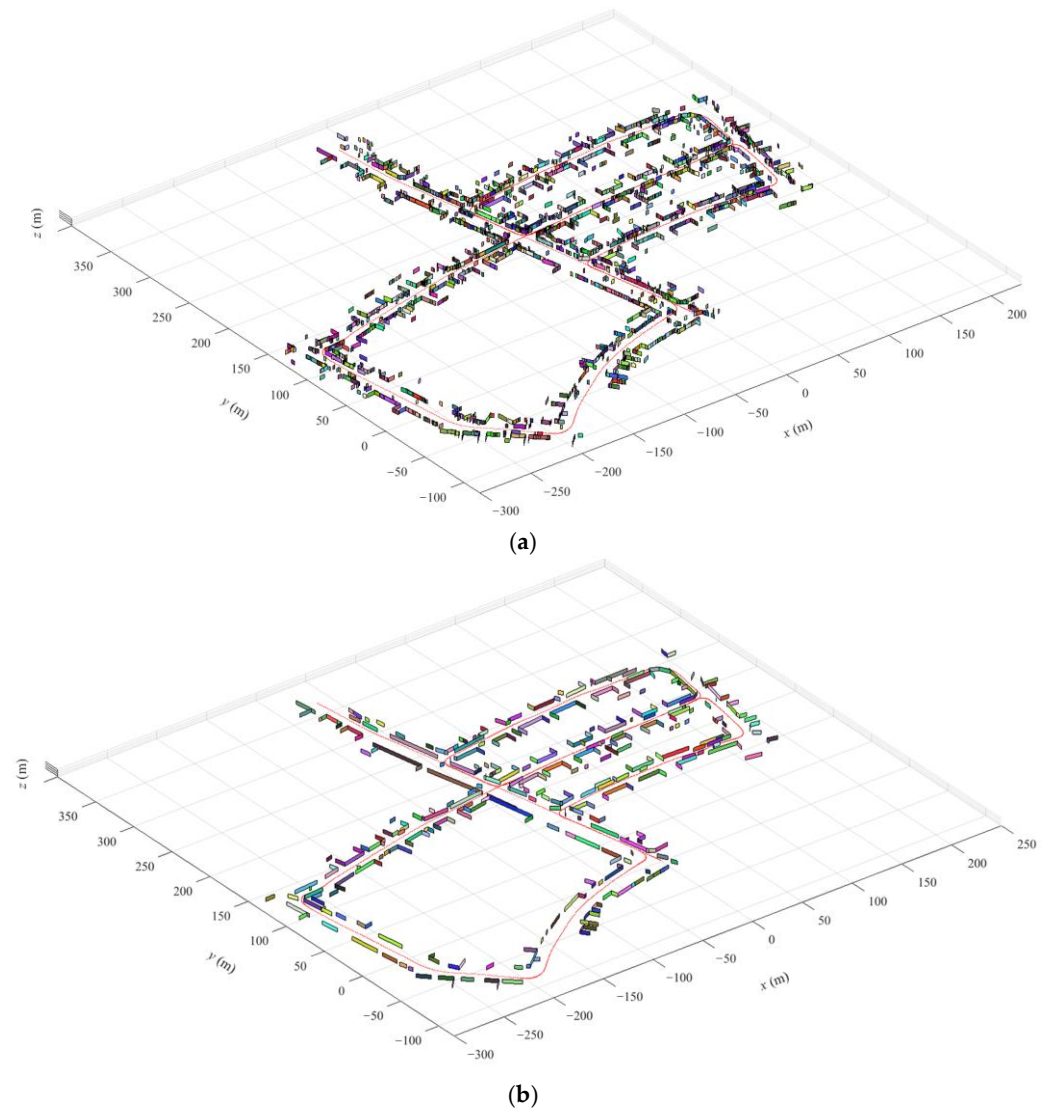


Figure 8. Global maps before and after feature fusion. (a) Global map before feature fusion; (b) global map after feature fusion.

To show how the proposed method outperforms the competing methods, four local maps are selected from the global map for comparison purposes, as observed in Figures 9 and 10. These figures demonstrate the feature fusion results for three methods in 3D and 2D, respectively. It is observed that the proposed method yields a fewer number of planes, and no redundant (i.e., repeating) features are retained. In contrast, the competing methods maintain more features in the map, and redundant features are still present in these four local maps. In other words, with the competing fusion methods, ambiguity in the map is not completely eliminated; in addition, more storage space and higher computation power are also required.

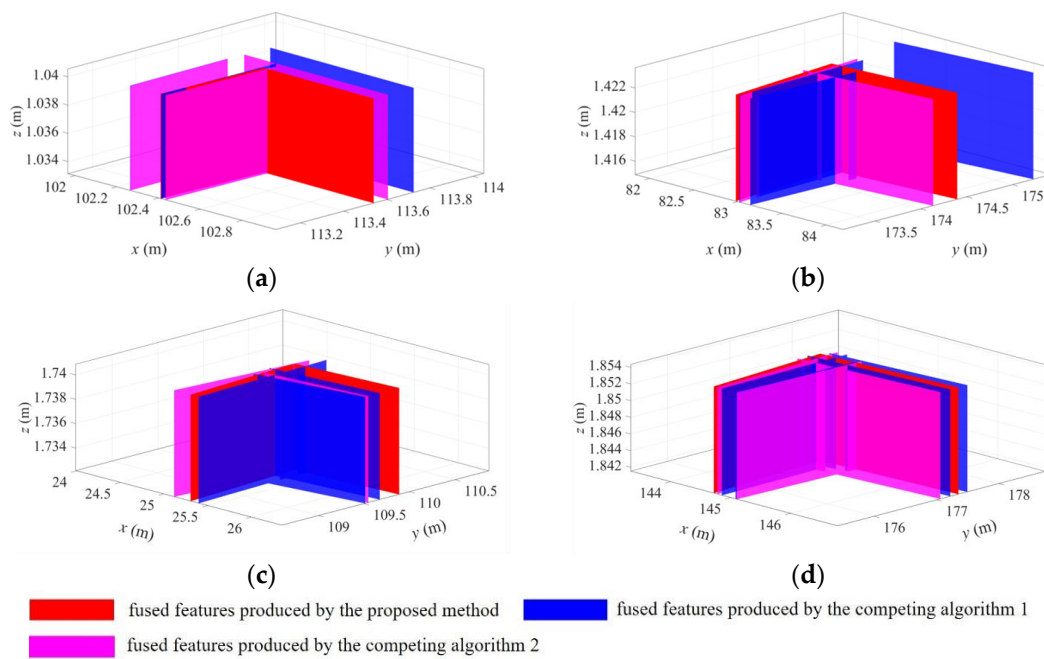


Figure 9. Fused features produced by the three competing methods in 3D. Subfigures (a–d) illustrate the feature fusion results of the three competing methods in 3D at four selected positions.

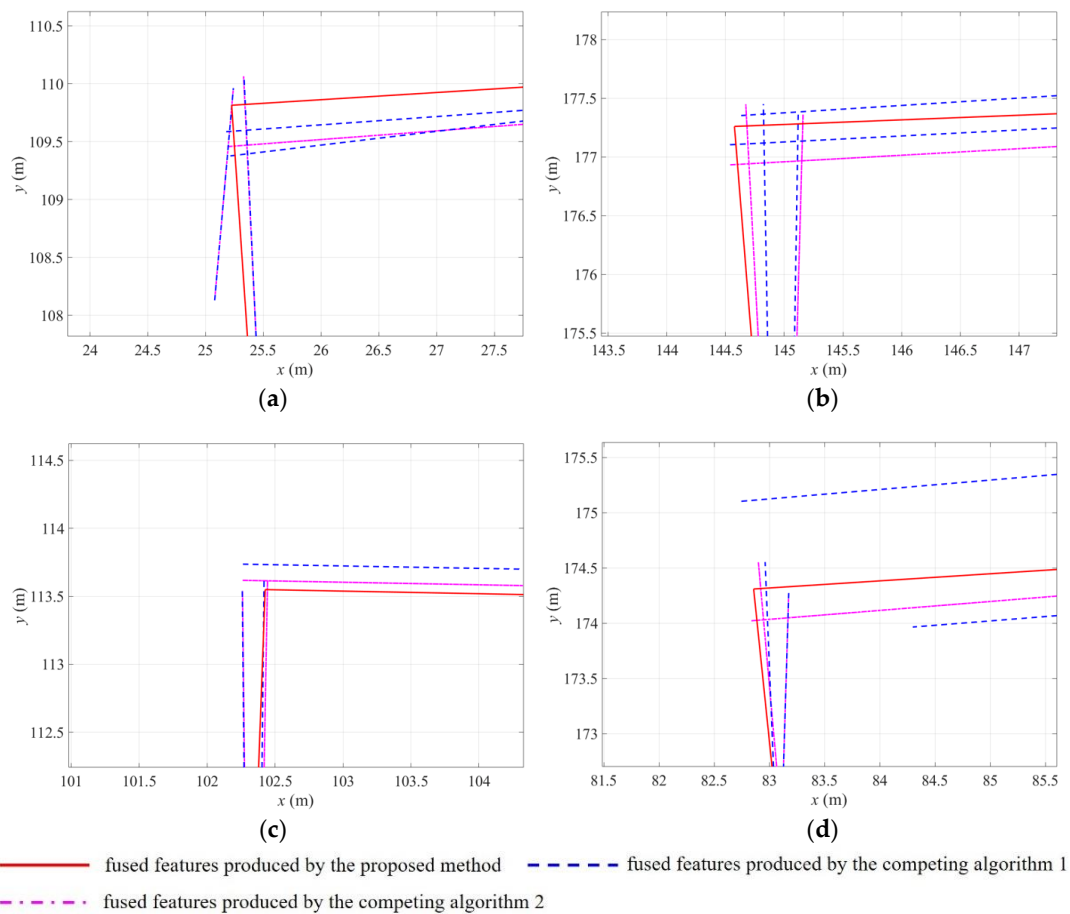


Figure 10. Fused features produced by the three competing methods in 2D. Subfigures (a–d) illustrate the feature fusion results of the three competing methods in 2D at four selected positions.

Apart from the graphical results, the number of resulting features is also counted using these three methods. Table 1 compares the number of fused features produced by the three methods, for the four local maps shown in Figures 9 and 10, as well as the global map shown in Figure 8b. It can be observed that the proposed method provides a consistently lower number of fused features for all cases, thereby leading to improved mapping efficiency.

Table 1. Comparison of fused feature numbers.

Map	Number of Fused Features		
	Competing Algorithm 1	Competing Algorithm 2	Proposed Algorithm
Figure 11a	4	3	2
Figure 11b	4	3	2
Figure 11c	2	3	2
Figure 11d	4	3	2
Global Map	1333	513	360

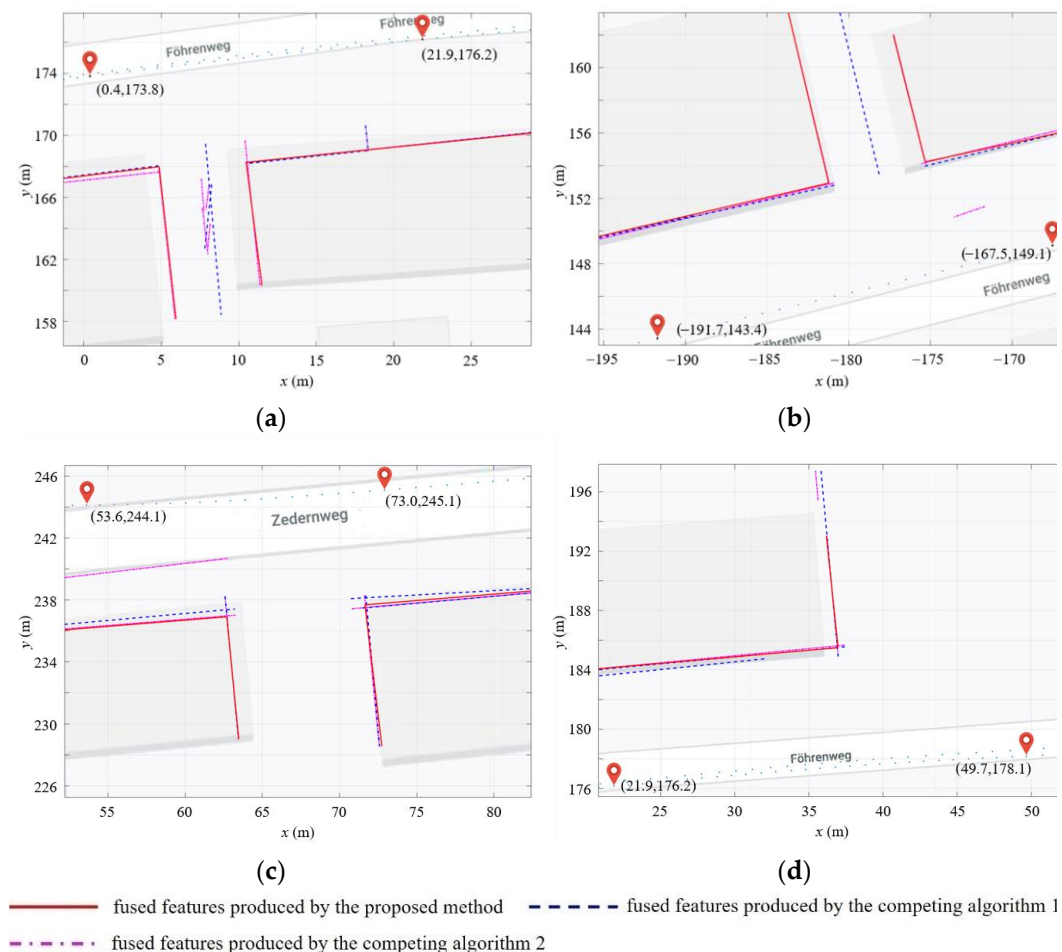


Figure 11. Line features maps on top of 2D Google maps for the three methods. Subfigures (a–d) present the comparison results of the three feature fusion methods, overlaid on Google maps (ground truth). Each subfigure corresponds to a distinct local map.

4.2. Comparison in Terms of Accuracy

The advantage of providing a lower number of fused features should not be achieved at the cost of jeopardizing the accuracy of the feature map. In this section, the fused features resulting from the three methods are plotted on top of Google maps for exactly the same areas, in order to evaluate the closeness between the fused features and the actual structures.

Figure 11 demonstrates the comparison results for four different local maps, with fused features plotted on top of Google maps.

The vehicle locations at two different times are employed to align the feature maps with Google maps. In Figure 11, the two red location icons in each subfigure represent the vehicle locations at two different times. Note that the vehicle location on the Google map can be obtained by its longitude and latitude, while its location on the feature map can be acquired by converting its longitude and latitude to x and y coordinates on the feature map. Hence, for a specific area, map alignment can be achieved by matching the two vehicle locations on the feature map and Google map. It can be observed in Figure 11 that the fused features produced by the proposed method generally match well with the profiles of actual structures, faithfully reflecting their geometric information. In contrast, the competing methods exhibit inferior performance due to their high sensitivity to the bandwidth choice, causing incorrect feature classification and fused features that deviate from the ground truth. For example, some fused features generated by competing Algorithms 1 and 2 (see Figure 11a,b) are located in between two buildings, resulting in significant feature location errors. The proposed fusion algorithm effectively addresses these limitations, providing a more accurate and reliable feature fusion solution.

Apart from the graphical results, in this study, quantitative results are also obtained to compare the feature accuracies of the three methods. Generating quantitative results requires the knowledge of the actual location and dimension of the buildings (ground truth), which is not available in Google maps. To tackle this issue, an artificial urban environment, as shown in Figure 12, is constructed using the commercial software PreScan. In this simulated environment, the exact building locations are available, thus the discrepancies (i.e., errors) between the extracted features and the actual structures can be readily computed. Note that the simulated vehicle used in PreScan is equipped with a 32-beam LiDAR sensor. This sensor has a range of 150 m and the received 3D points contain Gaussian measurement noise. Plane features are first extracted from the received point clouds, and then fused by means of the proposed method and the two competing algorithms.

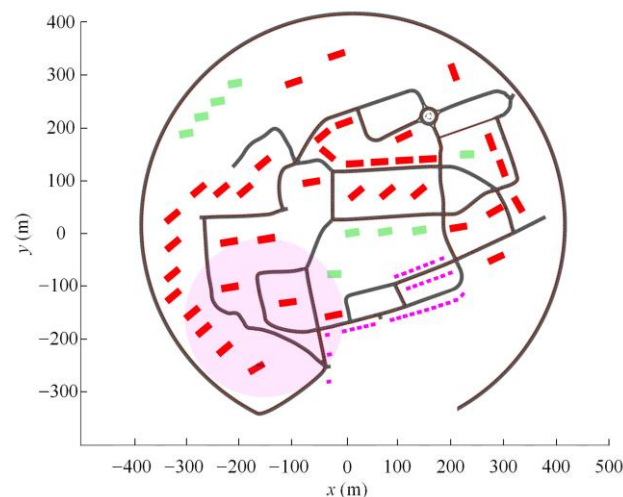


Figure 12. Artificial urban environment constructed in PreScan. Three different types of buildings are represented by red, green and magenta rectangles, while the roads are depicted by black lines. The pink circle denotes the LiDAR's horizontal field of view at its current position.

Figure 13a is a bird's-eye view of the actual outline of a building (ground truth) in the constructed environment. The four vertices of the outline are denoted by lowercase letters a , b , c and d , and the areas enclosed by the green rectangles are zoomed in and denoted by numbers 1, 2 and 3, as shown in Figure 13b–d.

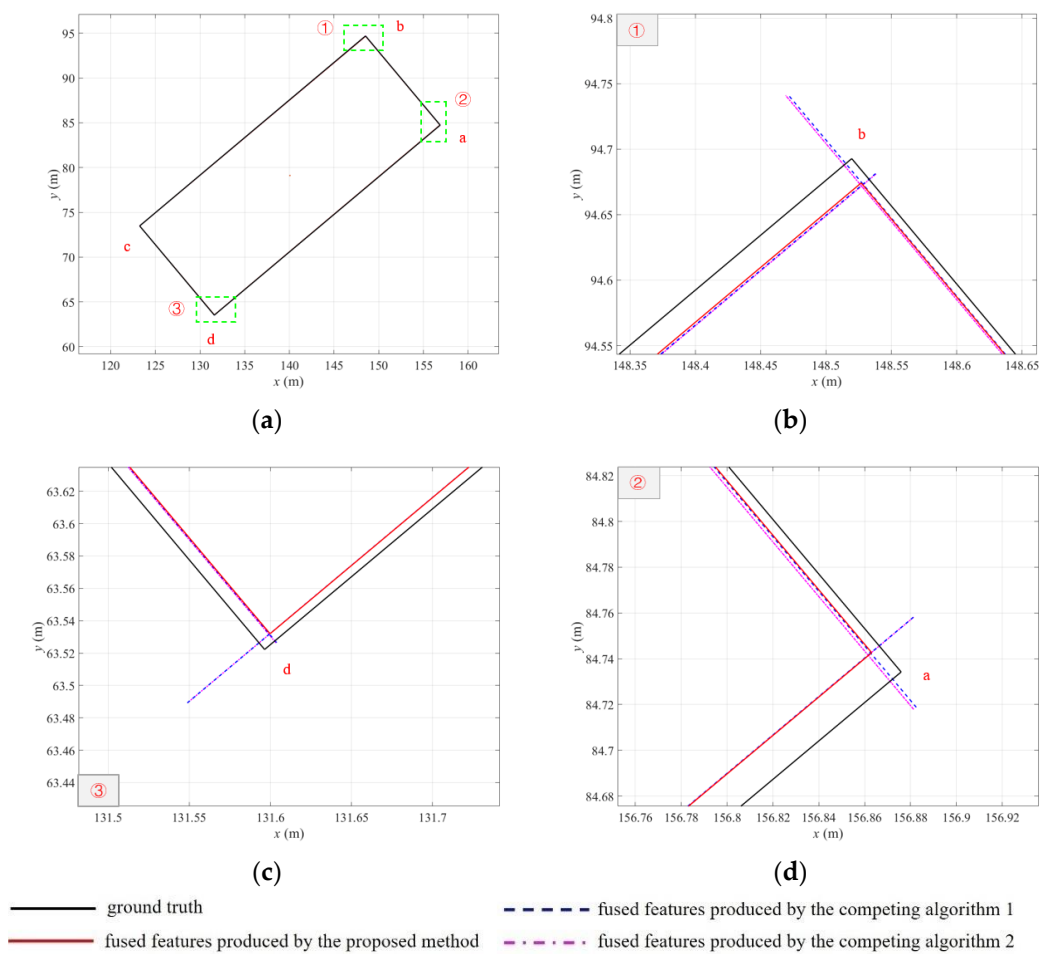


Figure 13. Ground truth and fused features produced by the three methods—Case 1. Subfigure (a) is a bird's-eye view of the actual outline of a building (ground truth) in the constructed environment. Subfigures (b–d) offer magnified views of areas encompassed within the green rectangles in the primary figure. The lowercase letters a, b, c, and d denote the positions of the rectangle's four vertices. Additionally, the zones within the green rectangles are distinctly labeled with the numbers 1, 2, and 3 for further reference.

To quantify the accuracy of fused features, the error (Euclidean distance) between the actual vertex and the endpoint of a fused feature is employed as an indicator. Specifically, for a certain vertex, the sum of all such errors is used to evaluate the accuracy of the fused features. For example, in Figure 13d, the sum of errors between vertex a and the two blue endpoints reflects the accuracy of the fused features (blue dashed segments) resulting from competing Algorithm 1.

Table 2 shows a comparison of the fused feature accuracies, quantified by the sum of errors, for all four vertices a, b, c and d. It can be observed that the proposed method in this paper outperforms the two competing algorithms, producing fused features with consistently higher accuracies for all four cases. This advantage is of great significance for SLAM applications, since more accurate map construction undoubtedly facilitates vehicle/robot positioning. Note that to ensure fair comparison, the same post processing has been employed for all three algorithms in Table 2.

Figure 14 shows a more challenging case of the simulated environment constructed in PreScan, where the density of buildings is higher than that in Figure 13. Figure 14a displays a bird's-eye view of the actual contours (ground truth) of two closely located buildings in the constructed environment. The areas enclosed by the green rectangles are magnified and labeled with numbers 1 and 3, as shown in Figure 14b,d. In Figure 14b, the

area enclosed by the green rectangle is further magnified and labeled with number 2, as shown in Figure 14c. Within these enlarged areas, we observe that the proposed method exhibits a clear advantage in handling details over its competitors. Specifically, due to the inherent shortcomings of Algorithm 1, the results of Algorithm 1 in Figure 14b significantly deviate from the ground truth, and similar situations are also observed in Figure 11a,b. As for Algorithm 2, we can observe that its fused features cannot accurately align with the contours of the ground truth, in comparison with the proposed algorithm. The above illustrations indicate that our proposed method provides higher accuracy and robustness compared to the other two methods in both real and simulated environments.

Table 2. Comparison of fused feature accuracies.

Vertex	Competing Algorithm 1	Competing Algorithm 2	Proposed Algorithm
a (m)	0.0414	0.0416	0.0310
b (m)	0.0916	0.0917	0.0397
c (m)	0.0196	0.0200	0.0010
d (m)	0.0665	0.0664	0.0212

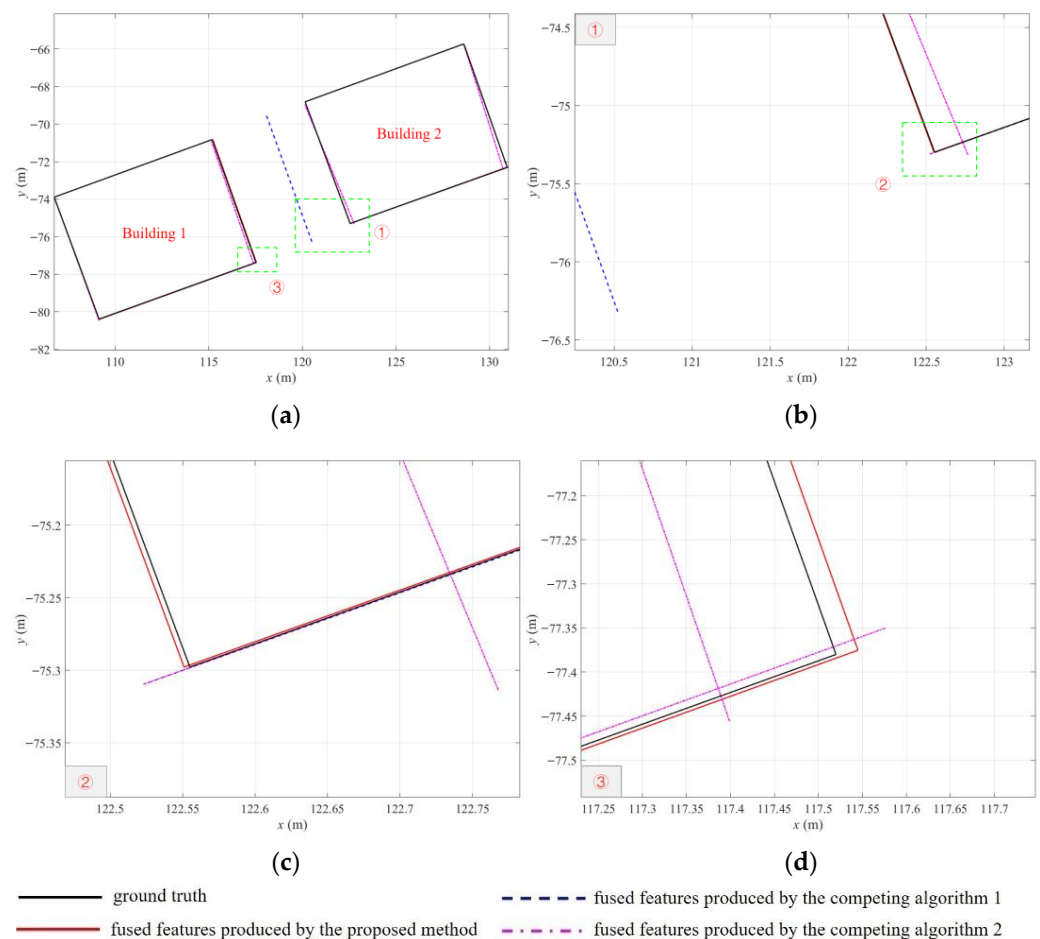


Figure 14. Ground truth and fused features produced by the three methods—Case 2. Subfigure (a) delivers a bird's-eye view of the true contours (ground truth) of two closely positioned buildings within the constructed environment, featuring areas enclosed by green rectangles that are magnified and marked as 1 and 3. Subfigure (b) delves into the area enclosed by green rectangle 1, offering a magnified view and further highlighting the region marked as number 2. Subfigure (c) provides an even more detailed perspective of the area within green rectangle 2, originally highlighted in subfigure (b). Subfigure (d) affords a magnified depiction of the region within green rectangle 3, initially identified in subfigure (a).

Apart from the comparisons in terms of feature fusion performance, the computation time required by the three methods is also investigated in this study. The time consumed is compared and shown in Table 3. The proposed method takes a slightly longer time to complete the feature fusion task, but provides the best feature fusion results among the three. In comparison, competing Algorithm 1 consumes the shortest time but produces the worst fusion performance.

Table 3. Time evaluation of three competing algorithms.

	Competing Algorithm 1	Competing Algorithm 2	Proposed Algorithm
time (s)	3.24	4.68	8.45

Generally, the proposed feature fusion method is superior to the competing methods, in terms of conciseness (i.e., number of fused features) as well as accuracy (i.e., closeness between the fused features and actual structures). The underlying reasons are as follows. Firstly, using the mean shift principle in the stage of orientation clustering, any features that share the same (or similar) orientations are classified in the same cluster regardless of their spatial distances. Spatial clustering is conducted only after orientation clustering is completed. Consequently, in the first stage, features that are located far from each other may be categorized in the same cluster, as long as their orientations are close enough. Secondly, the performance of the two competing algorithms is highly sensitive to the choice of bandwidth. Specifically, if the bandwidth is too large, in the spatial clustering stage, it is possible that line features that originate from different building surfaces are wrongly classified in the same cluster. As a result, the resulting fused features may greatly deviate from the ground truth. To tackle the above limitations, the proposed algorithm deals with the feature fusion problem in a different fashion and does not rely on mean shift clustering any more. It is not a simply modified version of the typical mean shift algorithm, but a novel framework for solving the above feature fusion problem. The proposed algorithm uses three criteria to simultaneously evaluate the orientation, overlapping, and relative distance of any two features extracted at two different times. These criteria determine the closeness of any two features and whether they should be fused in the mapping process. Using the three criteria, all features in the global map are categorized into different clusters with distinct labels, and a fused feature is then generated for each cluster by means of least squares fitting. By this means, the above defects existing in the competing algorithms can be effectively overcome by the proposed method.

It should also be pointed out that two limitations are observed for the proposed method. Firstly, the effectiveness of the proposed approach is dependent on the accuracy of vehicle positioning. Namely, vehicle location errors can lead to deterioration of feature fusion performance. This limitation is not only present in this approach but also in all feature fusion methods that rely on vehicle position information (such as the two competing algorithms). Thus, this is indeed a common research problem that requires further investigation in the future. Secondly, the time consumed by the proposed method is slightly longer than the two competing methods. This is because the proposed method employs three criteria to evaluate whether any two features ‘should or should not be fused’, and also counts the number of feature appearances in LiDAR scans. This pre-processing significantly enhances the feature fusion performance (which is lacking in the two competing algorithms), at the cost of the slightly increased computation time. The second limitation should not become a major problem as it is an offline approach, especially considering the performance enhancement it achieves.

5. Conclusions and Future Works

This paper proposes a feature fusion method for merging geometric features (i.e., line and plane features) for the mapping process in SLAM applications. Large structures in the environment appear repeatedly at multiple time steps, which results in redundant

geometric features that originate from the same structures. These redundant features not only give rise to ambiguity, but also slow down the mapping process. To tackle this issue, three important criteria are proposed in this paper to evaluate the closeness between the features and determine whether the original features should be fused. Based on these criteria, all features in the map are classified into different clusters, and for each cluster, a single fused feature is generated as a representative by means of least squares fitting. For comparative verification, the proposed method is compared with two typical feature fusion methods in the literature, using the commonly used KITTI dataset and the commercial software PreScan. The comparison results show that the proposed method outperforms the competing methods in terms of conciseness (i.e., number of fused features) as well as accuracy (i.e., closeness between the fused features and actual structures).

In the next step of investigation, the proposed method will be incorporated in a complete geometric-feature-based SLAM framework for further verification. In addition, experimentation will be followed by the use of small-scale autonomous robots equipped with LiDAR sensors.

Author Contributions: Conceptualization, Feiya Li; methodology, Feiya Li and Chunyun Fu; validation, Feiya Li and Hormoz Marzbani; formal analysis, Feiya Li; resources, Dongye Sun and Minghui Hu; writing—original draft preparation, Feiya Li; writing—review and editing, Chunyun Fu and Hormoz Marzbani; supervision, Dongye Sun and Chunyun Fu; project administration, Chunyun Fu and Minghui Hu; funding acquisition, Chunyun Fu. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Chongqing Technology Innovation and Application Development Project under Grant CSTB2022TIAD-DEX0013.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: <https://www.cvlibs.net/datasets/kitti/>.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, Y.; Yu, H.; Zhang, K.; Zheng, Y.; Zhang, Y.; Zheng, D.; Han, J. FPP-SLAM: Indoor simultaneous localization and mapping based on fringe projection profilometry. *Opt. Express* **2023**, *31*, 5853. [CrossRef] [PubMed]
2. Gostar, A.K.; Fu, C.; Chuah, W.; Hossain, M.I.; Tennakoon, R.; Bab-Hadiashar, A.; Hoseinnezhad, R. State Transition for Statistical SLAM Using Planar Features in 3D Point Clouds. *Sensors* **2019**, *19*, 1614. [CrossRef] [PubMed]
3. Chen, D.; Weng, J.; Huang, F.; Zhou, J.; Mao, Y.; Liu, X. Heuristic Monte Carlo Algorithm for Unmanned Ground Vehicles Realtime Localization and Mapping. *IEEE Trans. Veh. Technol.* **2020**, *69*, 10642–10655. [CrossRef]
4. Sun, J.; Song, J.; Chen, H.; Huang, X.; Liu, Y. Autonomous State Estimation and Mapping in Unknown Environments with Onboard Stereo Camera for Micro Aerial Vehicles. *IEEE Trans. Ind. Inform.* **2019**, *16*, 5746–5756. [CrossRef]
5. Wen, S.; Zhao, Y.; Liu, X.; Sun, F.; Lu, H.; Wang, Z. Hybrid Semi-Dense 3D Semantic-Topological Mapping From Stereo Visual-Inertial Odometry SLAM with Loop Closure Detection. *IEEE Trans. Veh. Technol.* **2020**, *69*, 16057–16066. [CrossRef]
6. Zubizarreta, J.; Aguinaga, I.; Montiel, J.M.M. Direct Sparse Mapping. *IEEE Trans. Robot.* **2020**, *36*, 1363–1370. [CrossRef]
7. Yu, Z.; Min, H. Visual SLAM Algorithm Based on ORB Features and Line Features. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 3003–3008. [CrossRef]
8. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
9. Yuan, C.; Xu, Y.; Zhou, Q. PLDS-SLAM: Point and Line Features SLAM in Dynamic Environment. *Remote Sens.* **2023**, *15*, 1893. [CrossRef]
10. Zi, B.; Wang, H.; Santos, J.; Zheng, H. An Enhanced Visual SLAM Supported by the Integration of Plane Features for the Indoor Environment. In Proceedings of the 2022 IEEE 12th International Conference on Indoor Positioning and Indoor Navigation (IPIN), Beijing, China, 5–8 September 2022; pp. 1–8. [CrossRef]
11. Li, F.; Fu, C.; Gostar, A.K.; Yu, S.; Hu, M.; Hoseinnezhad, R. Advanced Mapping Using Planar Features Segmented from 3D Point Clouds. In Proceedings of the 2019 International Conference on Control, Automation and Information Sciences (ICCAIS), Chengdu, China, 23–26 October 2019; pp. 1–6.
12. Yang, H.; Yuan, J.; Gao, Y.; Sun, X.; Zhang, X. UPLP-SLAM: Unified point-line-plane feature fusion for RGB-D visual SLAM. *Inf. Fusion* **2023**, *96*, 51–65. [CrossRef]
13. Yu, S.; Fu, C.; Gostar, A.K.; Hu, M. A Review on Map-Merging Methods for Typical Map Types in Multiple-Ground-Robot SLAM Solutions. *Sensors* **2020**, *20*, 6988. [CrossRef]

14. Sun, Q.; Yuan, J.; Zhang, X.; Duan, F. Plane-Edge-SLAM: Seamless Fusion of Planes and Edges for SLAM in Indoor Environments. *IEEE Trans. Autom. Sci. Eng.* **2021**, *18*, 2061–2075. [\[CrossRef\]](#)
15. Dai, K.; Sun, B.; Wu, G.; Zhao, S.; Ma, F.; Zhang, Y.; Wu, J. LiDAR-Based Sensor Fusion SLAM and Localization for Autonomous Driving Vehicles in Complex Scenarios. *J. Imaging* **2023**, *9*, 52. [\[CrossRef\]](#)
16. Xie, H.; Zhang, D.; Wang, J.; Zhou, M.; Cao, Z.; Hu, X.; Abusorrah, A. Semi-Direct Multimap SLAM System for Real-Time Sparse 3-D Map Reconstruction. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 1–13. [\[CrossRef\]](#)
17. Xu, Y.; Zhou, L.; Tang, H.; Wu, Q.; Xie, Q.; Chen, H.; Wang, J. Robust and Accurate RGB-D Reconstruction with Line Feature Constraints. *IEEE Robot. Autom. Lett.* **2021**, *6*, 6561–6568. [\[CrossRef\]](#)
18. Sarkar, B.; Pal, P.K.; Sarkar, D. Building maps of indoor environments by merging line segments extracted from registered laser range scans. *Robot. Auton. Syst.* **2014**, *62*, 603–615. [\[CrossRef\]](#)
19. Elseberg, J.; Creed, R.T.; Lakaemper, R. A line segment based system for 2D global mapping. In Proceedings of the IEEE International Conference on Robotics and Automation, Anchorage, Alaska, 3–8 May 2010; pp. 3924–3931.
20. Amigoni, F.; Li, A.Q. Comparing methods for merging redundant line segments in maps. *Robot. Auton. Syst.* **2018**, *99*, 135–147. [\[CrossRef\]](#)
21. Amigoni, F.; Vailati, M. A method for reducing redundant line segments in maps. In Proceedings of the European Conference on Mobile Robots (ECMR), Mlini/Dubrovnik, Croatia, 23–25 September 2009; pp. 61–66.
22. Gomez-Ojeda, R.; Moreno, F.; Scaramuzza, D.; Gonzalez-Jimenez, J. PL-SLAM: A stereo SLAM system through the combination of points and line segments. *arXiv* **2017**, arXiv:1705.09479. [\[CrossRef\]](#)
23. Liu, J.; Meng, Z. Visual SLAM with Drift-Free Rotation Estimation in Manhattan World. *IEEE Robot. Autom. Lett.* **2020**, *5*, 6512–6519. [\[CrossRef\]](#)
24. Gao, H.; Zhang, X.; Li, C.; Chen, X.; Fang, Y.; Chen, X. Directional Endpoint-based Enhanced EKF-SLAM for Indoor Mobile Robots. In Proceedings of the 2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Hong Kong, China, 8–12 July 2019; pp. 978–983.
25. Wen, J.; Zhang, X.; Gao, H.; Yuan, J.; Fang, Y. CAE-RLSM: Consistent and Efficient Redundant Line Segment Merging for Online Feature Map Building. *IEEE Trans. Instrum. Meas.* **2019**, *69*, 4222–4237. [\[CrossRef\]](#)
26. Wang, J.; Song, J.; Zhao, L.; Huang, S.; Xiong, R. A submap joining algorithm for 3D reconstruction using an RGB-D camera based on point and plane features. *Robot. Auton. Syst.* **2019**, *118*, 93–111. [\[CrossRef\]](#)
27. Hsiao, M.; Westman, E.; Zhang, G.; Kaess, M. Keyframe-based dense planar SLAM. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 5110–5117.
28. Ćwian, K.; Nowicki, M.R.; Nowak, T.; Skrzypczyński, P. Planar Features for Accurate Laser-Based 3-D SLAM in Urban Environments. In *Advances in Intelligent Systems and Computing*; Springer: Cham, Switzerland, 2020; pp. 941–953. [\[CrossRef\]](#)
29. Grant, W.S.; Voorhies, R.C.; Itti, L. Efficient Velodyne SLAM with point and plane features. *Auton. Robot.* **2018**, *43*, 1207–1224. [\[CrossRef\]](#)
30. Pan, L.; Wang, P.F.; Cao, J.W.; Chew, C.M. Dense RGB-D SLAM with Planes Detection and Mapping. In Proceedings of the IECON 2019—45th Annual Conference of the IEEE Industrial Electronics Society, Lisbon, Portugal, 14–17 October 2019; pp. 5192–5197.
31. Lakaemper, R. Simultaneous multi-line-segment merging for robot mapping using Mean shift clustering. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE /RSJ IROS), St. Louis, MO, USA, 10–15 October 2009; pp. 1654–1660.
32. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.