DeepPointMap2: Accurate and Robust LiDAR-Visual SLAM with Neural Descriptors

Anonymous Authors

ABSTRACT

Simultaneous Localization and Mapping (SLAM) plays a pivotal role in autonomous driving and robotics. Given the complexity of road environments, there is a growing research emphasis on developing robust and accurate multi-modal SLAM systems. Existing methods often rely on hand-craft feature extraction and cross-modal fusion techniques, resulting in limited feature representation capability and reduced flexibility and robustness. To address this challenge, we introduce ***DeepPointMap2***, a novel learning-based LiDAR-Visual SLAM architecture that leverages neural descriptors to tackle multiple SLAM sub-tasks in a unified manner. Our approach employs neural networks to extract multi-modal feature tokens, which are then adaptively fused by the *Visual-Point Fusion Module* to generate sparse neural 3D descriptors, ensuring precise localization and robust performance. As a pioneering work, our method achieves *state-of-the-art* localization performance among various Visual-based, LiDAR-based, and Visual-LiDAR-based methods in widely used benchmarks, as shown in the experiment results. Furthermore, the approach proves to be robust in scenarios involving camera failure and LiDAR obstruction.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; **Vision for robotics**; **Reconstruction**; *Scene understanding*.

KEYWORDS

Visual-LiDAR SLAM, Multi-modal Fusion, Neural Descriptors

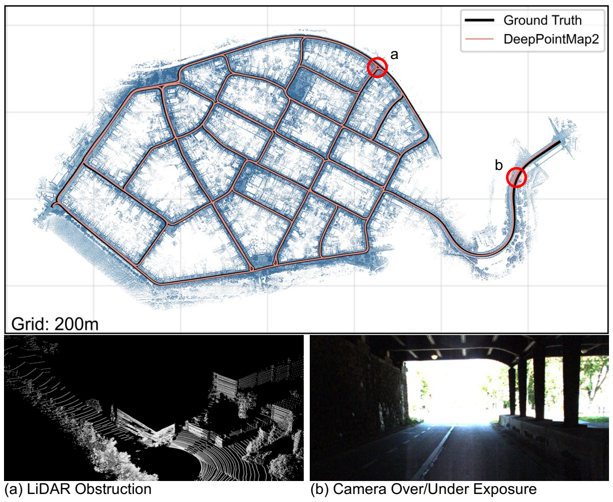


Figure 1: Examples of challenging scenes in road environments. (a) LiDAR obstructed by large vehicles, resulting in incomplete point clouds. (b) Camera with limited dynamic range struggles to properly expose in high-contrast scenes.

1 INTRODUCTION

Simultaneous Localization and Mapping (SLAM), aiming to estimate the agent's location while mapping its environment, is pivotal in autonomous driving and robotics, enabling navigation in unseen environments and understanding the surroundings. The environments in which autonomous vehicles operate are highly complex, making it challenging to achieve accurate SLAM with a single sensor. Multi-modal perception emerges as a crucial strategy, integrating various sensors like monocular cameras and LiDARs to enhance SLAM system capabilities.

As shown in Fig. 1, autonomous driving scenarios sometimes face challenges such as obstructed LiDAR point clouds and under/overexposure images. Such conditions can lead to a degradation in the performance of multi-modal SLAM methods. Some existing approaches [46, 59] typically process point cloud and image data separately in individual branches and employ post-fusion to estimate the trajectory. However, these approaches suffer from a limitation in adequately integrating information from multiple modalities during the feature extraction process, leading to a weak feature representation capability. While some other approaches [4, 31, 44, 55] fuse the feature before position estimation. Although these methods achieved accurate localization performance, they lack robustness against sensor failure or degeneration.

The pioneering work, DeepPointMap [62], utilizes neural descriptors to achieve accurate localization. It is confirmed that the neural descriptors enable efficient and accurate feature representation and benefit the SLAM performance. However, it only involves LiDAR point cloud modality and encounters challenges in scenarios with sparse reference objects and limited geometric information. To address these challenges, we introduce a learning-based LiDAR-Visual SLAM approach, *DeepPointMap2*, aiming to enhance the key processes of feature extraction and localization in SLAM task using neural networks. We utilize two arbitrary backbone networks to extract multi-scale tokens from both image and point cloud. A *Visual-Point Fusion module* is followed to aggregate these multi-modal tokens into 3D neural descriptors. This end-to-end learnable framework allows the model to adaptively learn how to fuse multi-modal features, providing better feature representation and robustness, compared to existing manual-designed extraction and fusion strategies.

The proposed framework offers several advantages over existing methods. **Representation capacity**: Our method aggregates multi-scale features in a learning-based manner, providing descriptors with both local fine-grained geometric features and coarse-grained scene information, thereby enhancing the representation capacity. **Robustness**: Our approach has strong robustness by utilizing learning-based cross-modal feature fusion strategy. When encountering LiDAR obstruction or camera failure, the model can still maintain accurate localization and mapping. **Flexibility**: Our method is a simple and end-to-end pipeline, and can accommodate arbitrary single-modal backbone. Users can choose the appropriate backbone according to their needs. In addition, the network structure is also potentially able to fuse more modal inputs.

Our contributions can be summarized as follows:

- We proposed a pioneering learning-based Visual-LiDAR SLAM framework, *DeepPointMap2*, that leverages multi-modal neural 3D descriptors to represent the environment with high fidelity. These descriptors are designed to capture more refined features and adeptly integrate information from multiple modalities, thereby ensuring enhanced representation and robust performance.
- Our framework includes a specialized deep neural network module for robust cross-modal feature fusion, which integrates features from both images and point clouds. This module incorporates learnable weights and spatial correspondence mechanisms to fuse information effectively.
- Experimental results show that our approach achieves *state-of-the-art* (SOTA) performance in localization accuracy. Notably, *DeepPointMap2* achieves stable performance when facing modalities missing scenarios, such as in the event of camera failure or LiDAR obstruction.

2 RELATED WORK

2.1 LiDAR SLAM

LiDAR point clouds are commonly represented as unordered sets of 3D coordinates. In SLAM methods, the process typically involves first (1) extracting key-points with geometric features from the point cloud, followed by (2) matching corresponding key-points between adjacent frames based on their geometric features, and finally (3) solving the relative pose transformation through SVD [2] or iterative techniques [29]. Extracting geometric features from the point cloud constitutes a critical step in LiDAR-based SLAM approaches.

Knowledge-based methods compute the feature based on predefined geometric metrics, such as curvature and density. LOAM [58], as one of the early works, and its subsequent approaches [19, 42, 52, 53] utilized point-wise curvature to detect edge and planar points. Then, the association is applied within each category. Additionally, MULLS [34] further classify key-points into more specific categories to establish a more accurate association. PUMA [49] introduced a surface mesh representation that better captured the geometric appearance of objects in the scene. Although these methods can extract features efficiently, their representation capability is limited, necessitating more key-points and complex association algorithms.

Learning-based methods utilize deep neural networks to extract point cloud features. PointNetLK [1] employs PointNet [35] to extract scan-level features and applies a modified Lucas-Kanade algorithm for transformation estimation. LO-Net [22] proposed a scan-to-scan odometry network that predicts normals, identifies dynamic regions, and incorporates a spatiotemporal geometrical consistency constraint for improved interactions between sequential scans. To achieve accuracy loop detection, LCDNet [5] and its lightweight variant DeLightLCD [54] utilized a 3D voxel CNN network to extract descriptors and estimate coarse transformation. DeepPointMap [62] pioneers the use of neural networks for unified odometry and loop detection, employing neural descriptors for accurate localization with efficient memory use.

Despite the robustness of the LiDAR sensor, its inability to capture informative texture makes it suffer from structure-less environments (*e.g.*, tunnels). The partial obstruction issues may further hinder its performance.

2.2 Visual SLAM

Similar to LiDAR SLAM, monocular visual SLAM can also be considered a data association task. Visual SLAM can be categorized into indirect and direct methods depending on the association method.

Direct methods directly minimize the pixel-wise photometric error between frames to estimate camera motion. DTAM [33] utilized all pixels of frames and estimated the relative pose of the camera. To reduce computational complexity, LSD-SLAM [11] and DSO [10] selected pixels with large gradients. As one of the most famous methods, SVO [13] further introduced the FAST feature detector to enhance feature extraction ability and achieve precise association.

Indirect methods focus on detecting and matching sparse feature descriptors from images to reduce computational complexity. To achieve this, some methods [20, 21, 41, 45] extract point features from image using pixels' neighbor, while some [50, 61] focus on line features.Meanwhile, some methods utilize neural networks to select and extract descriptors. SemanticFusion [30] utilizes CNNs to perform semantic segmentation to build the semantic map. To suppress the effects of dynamic objects, Cheng et al. [6] also uses a neural network to select static sparse descriptors.

Although visual SLAM methods only require inexpensive cameras, they may encounter challenges such as sensitivity to illumination (*e.g.*, HDR environment) or weather (*e.g.*, rain).

2.3 Visual-LiDAR SLAM

Visual-LiDAR SLAM models can be divided into two categories, depending on the cross-modal feature fusion strategy.

Loosely-Coupled methods consider the estimation of several modalities separately. The cross-modal fusion procedure is applied after each estimation is generated. FAST-LIVO [63], R^{2}LIVE [26], and its subsequent R^{3}LIVE [25] employ a Kalman-Filter to fuse LiDAR, Visual, and IMU measurements, yielding precise odometry results. Similarly, LIV-LAM [39] proposes an unsupervised learning method for object discovery based on a camera detector and a LiDAR odometry, followed by the fusion of detected objects and LiDAR measurements using pose-graph optimization.

Tightly-Coupled methods, unlike loosely-coupled ones, fuse sensor measurements from each modality before the state estimating,

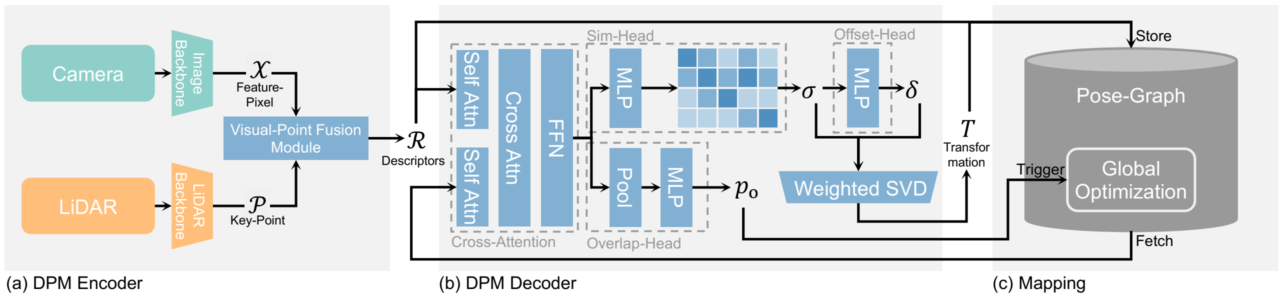


Figure 2: *DeepPointMap2* consists of three components. (a) DPM encoder extract multi-modal features and aggregate them into 3D neural descriptors. (b) DPM decoder utilizes descriptors for solving multiple SLAM subtasks *i.e.*, odometry and loop detection. (3) Mapping Module manage the observation and reconstructed map.

which are often more accurate. DEMO [57] utilized depth information from LiDAR (or RGB-D camera) to enhance the bundle adjustment-based visual odometry. In DEMO, the depth information is integrated into the optimization process to refine the estimated camera poses. Huang et al. [18] introduced a visual-LiDAR odometry method using point and line features extracted from images. After estimating depth based on LiDAR data, the point and line depths are utilized as prior factors in the point-line bundle adjustment process. LAMV-SLAM [55] integrates LiDAR and monocular visual data by employing online photometric calibration and a depth fusion algorithm to provide accurate depth values for visual features, enhancing mapping and localization in outdoor environments.

Loosely-coupled methods provide modal flexibility but risk losing critical information and accuracy due to underutilized inter-modal relationships. Although tightly-coupled methods are more integrated, they rely on all sensors operating correctly and a single sensor failure can diminish performance or cause system failure.

3 ARCHITECTURE

3.1 Model Overview

As illustrated in Fig. 2, our proposed *DeepPointMap2* consists of three main components: (a) **DPM Encoder** aims to extract feature tokens from multi-modal sensor data and aggregate them into comprehensive descriptors for each frame, (b) **DPM Decoder** utilize neural networks to estimate the transformation matrix between two frames based on their descriptors and perform loop detection to assist with constructing a consistent map, and (c) **Mapping Module** store the frame-wise information into a pose-graph structure and executes global-optimization once a loop closure is confirmed.

3.2 DPM Encoder

DeepPointMap2 is a multi-modal SLAM framework that utilizes neural descriptors $\mathcal{R}$ with compressed semantic features to represent 3D scenes. A descriptor $\mathbf{r}_i$ can be denoted as $\mathbf{r}_i := (\mathbf{r}_i^{\text{xyz}}, \mathbf{r}_i^{\text{feat}})$, where $\mathbf{r}_i^{\text{xyz}}$ denotes the 3D coordinate and $\mathbf{r}_i^{\text{feat}}$ is the associated feature.

DPM Encoder takes both point cloud and image as input, as illustrated in Fig. 3 (a). For point clouds, we use PointNeXt [36], one of the most famous neural architectures for point cloud understanding, as our backbone to extract multi-scale key-points $\mathbf{p}_i := (\mathbf{p}_i^{\text{xyz}}, \mathbf{p}_i^{\text{feat}})$, where $\mathbf{p}_i^{\text{xyz}}$ is the coordinate of key-point and $\mathbf{p}_i^{\text{feat}}$ is its feature. For image data, we employ ConvNeXt [28] with FPN [27] to extract multi-scale feature map. We denote each feature-pixel as $\mathbf{x}_i := (\mathbf{x}_i^{\text{uv}}, \mathbf{x}_i^{\text{feat}})$, where $\mathbf{x}_i^{\text{uv}}$ is its UV coordinate.

After extracting multi-scale feature tokens (as $\mathcal{P}$ and $\mathcal{X}$), the *Visual-Point Fusion Module* fuse these features and generate the descriptors $\mathcal{R}$. As shown in Fig. 3, the *Visual-Point Fusion Module* consists of two parts: Reference-Points Generator (RPG) and Multi-Modal Transformer (MMT) Decoders. Given the multi-scale tokens of both modalities $(\mathcal{X}, \mathcal{P})$, the RPG generates a set of 3D reference points (ref-points) based on the input point cloud. Subsequently, the MMT cascadely fuses feature tokens into these ref-points, and finally generates descriptors $\mathcal{R}$.

3.2.1 Reference-Points Generator. The Reference-Points Generator (RPG) aims to generate a set of 3D reference points (ref-points) $\mathbf{r}_i$ that serve as *seeds* for aggregating multi-modal tokens in subsequent modules. These ref-points are carefully selected to be both uniform across the point cloud and representative of the underlying scene structure, which is essential for effective aggregation and robust feature representation. To achieve this, we employ farthest-point-sampling (FPS) to select n ref-points coordinates $\mathbf{r}_i^{\text{xyz}}$ from the original point cloud. This strategy ensures that the selected ref-points are well-distributed and cover the spatial extent of the environment. We then initialize the features of each ref-point using an MLP based on their spatial coordinates:

$$\mathbf{r}_i^{\text{feat}} = \text{MLP}\left(\mathbf{r}_i^{\text{xyz}}\right) \tag{1}$$

3.2.2 Multi-Modal Transformer Decoder Layers. A sequence of L Multi-Modal Transformer (MMT) Decoders is applied to further process and refine the aggregated multi-modal tokens, following the Reference-Points Generator (RPG). Each MMT takes multi-scale (1) image feature-pixels $\mathcal{X} = (\mathbf{x}^{\text{uv}}, \mathbf{x}^{\text{feat}})$, (2) LiDAR key-points $\mathcal{P} = (\mathbf{p}^{\text{xyz}}, \mathbf{p}^{\text{feat}})$, and (3) reference points $\mathcal{R} = (\mathbf{r}^{\text{xyz}}, \mathbf{r}^{\text{feat}})$ as inputs. Initially, the ref-points are input into a multi-head self-attention module, which effectively enhances their features by allowing information exchange among ref-points. As shown in Figure Fig. 3 (b), each MMT layer consists of two branches: LiDAR and Image. Each branch is a transformer decoder structure with multiple layers, where `Query` is ref-points $\mathcal{R}$, both `Key` and `Value` are feature-pixels $\mathcal{X}$ in image branch or key-points $\mathcal{P}$ in LiDAR branch.

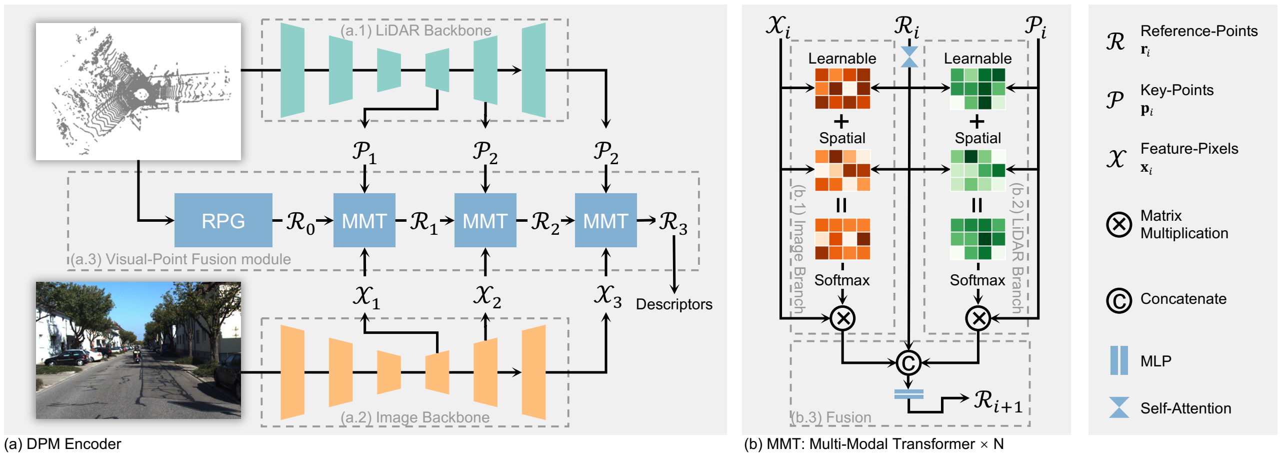
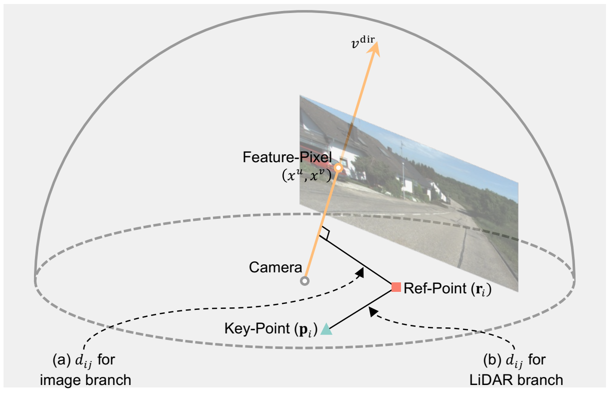


Figure 3: Overview of DPM Encoder. The encoder takes image and point cloud as inputs, and fuses the multi-scale feature tokens into descriptors to represent the environments and solve subsequent SLAM subtasks.



v^{dir}
Feature-Pixel (x^u, x^v)
Camera
Ref-Point ($\mathbf{r}_i$)
Key-Point ($\mathbf{p}_i$)
(a) d_{ij} for image branch
(b) d_{ij} for LiDAR branch

Figure 4: Distance of Feature Tokens to Reference points.

To further improve the performance of decoder layers, we introduce a novel cross-attention transformer module, named Biased Transformer, to replace the original scale-dot transformer block. As illustrated in Fig. 3 (c), the Biased Transformer module adaptively combines relative spatial distances with feature similarities between ref-points and feature-tokens. This integration facilitates a more precise and context-aware feature fusion strategy, thereby enhancing the overall accuracy and robustness of the SLAM framework.

LiDAR branch: Unlike the original scale-dot transformer [47], the Biased Transformer incorporates the distance between query and key in both spatial and feature space. Following scale-dot attention, we calculate the *learnable attention map* $E^{(1)}$ in feature space using pair-wise dot-product without positional embedding:

$$E^{(1)}_{ij} = \frac{1}{\sqrt{d}} \cdot W_q(\mathbf{p}^{\text{feat}}_i) \cdot W_k(\mathbf{r}^{\text{feat}}_j) \tag{2}$$

where W_q and W_k are learnable weights layers and d is the feature dimension. Meanwhile, as illustrated in Fig. 4 (a), since we know the relative position between key-points $\mathbf{p}^{\text{xyz}}_i$ and ref-points $\mathbf{r}^{\text{xyz}}_j$, a *spatial attention map* $E^{(2)}$ is derived based on Euclidean distance:

$$E^{(2)}_{ij} = \exp(-\alpha_1 \cdot d_{ij}) \tag{3}$$

where $d_{ij} = \|\mathbf{p}^{\text{xyz}}_i - \mathbf{r}^{\text{xyz}}_j\|_2$ is the distance between ref-points and key-points and α_1 is a pre-defined hyperparameter.

The final attention map for the Biased Transformer is obtained by merging these two attention maps by:

$$E^{\text{LiDAR}} = \text{Softmax}\Big(E^{(1)} + \xi\Big(E^{(2)}\Big)\Big) \tag{4}$$

where ξ is a Z-score normalization function. This normalization ensures the stability and comparability of the combined attention weights, allowing for a more effective fusion of spatial and feature information in the LiDAR branch.

Image branch: The *learnable attention map* $E^{(1)}$ can be calculated same as Eq. (2). However, due to the distinct coordinates of feature-pixels $\mathbf{x}_i$ and ref-points $\mathbf{r}_j$, the *spatial attention map* $E^{(2)}$ cannot be directly computed using Equation Eq. (3). Some methods project LiDAR key-points onto image coordinates and then extract fixed-size image patch features. These patch-wise feature are attached to the corresponding key-points. However, it is important to note that due to the potential scale uncertainty introduced by perspective projection, the fixed-size fusion mechanism may be inaccurate. To solve this problem, we introduce a metric based on the point-ray distance. As demonstrated in Fig. 4 (b), each feature-pixel $\mathbf{x}_i$ in the image can be defined as a ray $\mathbf{v}_i$ emanating from the camera's optical center $\mathbf{v}^{\text{ori}}_i$, with its directional vector $\mathbf{v}^{\text{dir}}_i$ calculated from its pixel coordinate $\mathbf{x}^{\text{uv}}_i$.

$$v_i := \begin{cases} \mathbf{v}^{\text{ori}}_i = T \\ \mathbf{v}^{\text{dir}}_i = T \times K^{-1} \times (\mathbf{x}^{\text{u}}_i, \mathbf{x}^{\text{u}}_i, 1, 1)^\top \end{cases} \tag{5}$$

where T is the transformation matrix from the camera to LiDAR and K is the intrinsic matrix of the camera. Given the disparate sensing ranges of cameras and LIDAR, it is necessary to initially determine if each ref-point falls within the camera's field of view and subsequently create a corresponding mask M by:

$$M_{ij} = \begin{cases} 1 \text{ ,if } \mathbf{v}_i^{\text{ori}} \cdot \left(\mathbf{r}_j^{\text{xyz}} - \mathbf{v}_i^{\text{ori}}\right) \geq 0 \\ 0 \text{ ,otherwise.} \end{cases} \tag{6}$$

The attention map can be calculated based on the distance d_{ij} between ref-points to the rays, given by:

$$d_{ij} = \frac{\mathbf{v}_i^{\text{dir}}}{\left\|\mathbf{v}_i^{\text{dir}}\right\|_2} \cdot \left(\mathbf{r}_j^{\text{xyz}} - \mathbf{v}_i^{\text{ori}}\right) \tag{7}$$

The *spatial attention map* is then calculated similar to Eq. (3) by:

$$E_{ij}^{(2)} = M_{ij} \cdot \exp\left(-\alpha_2 \cdot d_{ij}\right) \tag{8}$$

To this end, the *spatial attention map* $E_{ij}^{(2)}$ of the image branch is calculated, but now indicating the spatial relationship between the 3D ref-points and the 2D feature-pixels. It is worth noting that since the computation is done in 3D space, the weights $E_{ij}^{(2)}$ are able to precisely account for scale variations in the image that arise from perspective projection. Finally, the combined attention map E^{Image} for the image branch is computed using Eq. (4), adaptively integrating both the learnable and spatial attention mechanisms.

Feature fusion: After obtaining the corresponding modalities features, the feature of l-th layer ref-points $\mathcal{R}_l^{\text{feat}}$ are updated as:

$$\mathcal{R}_l^{\text{feat}} = \text{MLP}\left(\mathcal{R}_{l-1}^{\text{feat}} \oplus \mathcal{R}_{l-1}^{\text{feat}} \times E^{\text{LiDAR}} \oplus \mathcal{R}_{l-1}^{\text{feat}} \times E^{\text{Image}}\right) \tag{9}$$

The ref-points $\mathcal{R}_L$ outputted from the MMT block are then treated as the final descriptors $\mathcal{R}$, providing multi-modal condensed environment information.

3.3 DPM Decoder

Following DeepPointMap [62], we use the DPM Decoder to solve the relative transformation matrix T for odometry and overlap probability p_{overlap} for loop detection between frames t_1 and t_2 based on their descriptors $\mathcal{R}_{t_1}$ and $\mathcal{R}_{t_2}$. In detail, the DPM Decoder contains a transformer-based block and three individual heads, as shown in Fig. 2 (b). The block exchanges information between two set of descriptors and output *correlated descriptors*, denoted as $\bar{\mathcal{R}}_{t_1}$ and $\bar{\mathcal{R}}_{t_2}$.

For odometry, the **Similarity Head** estimates the correspondence σ between two sets of descriptors based on pairwise descriptor feature similarity. To tackle the problem led by the spatial sparsity of descriptors, **Offset Head** predict the relative offsets δ between descriptor pairs. Finally, the precise relative transformation T can be estimated using weighted-SVD [2].

For loop detection, we use **Overlap Head** to predict the loop-probability p_{o} that the distance between two frames is less than a predefined threshold $\varepsilon_{\text{loop}}$. We obtain frame-wise features for both frames by average pooling $\bar{\mathcal{R}}_{t_1}$ and $\bar{\mathcal{R}}_{t_2}$, then concatenating them and predicted the desired probability p_{o} via an MLP.

3.4 Mapping

We utilize *Pose-Graph* to store our reconstructed map, following Zhang et al. [62]. For each frame, we extract its descriptors (Sec. 3.2) and retrieve its nearest keyframe from the pose-graph to estimate its pose (Sec. 3.3). A key-frame selection process then determine the frame should be assigned as a keyframe. If a frame is assigned as a keyframe, a scan-to-map refinement will be applied to improve the pose estimation accuracy. We also conduct loop detection once a keyframe is established. A standard pose-graph optimization will be applied once the loop closure is conformed, to ensure the global consistency of the reconstructed map.

4 TRAINING

We jointly train the DPM Encoder and DPM Decoder end-to-end, with the following multiple losses.

Pairing Loss. Representing the geometry and texture features of ref-points is the key to descriptors. The ideal descriptors should share similar features if they are close in global coordinates and vice versa. Thus, we adopt InfoNCE [17] loss as pairing loss $\mathcal{L}_{\text{p}}$ on descriptors $\mathcal{R}$. For each descriptor $\mathbf{r}_{(i,t_1)} \in \mathcal{R}_{t_1}$ from frame t_1, a descriptor $\mathbf{r}_{(j,t_2)} \in \mathcal{R}_{t_2}$ from frame t_2 with the distance of $d_{ij} = \left\|\mathbf{r}_{(i,t_1)}^{\text{xyz}} - \mathbf{r}_{(j,t_2)}^{\text{xyz}}\right\|_2$ are assigned as (1) *positive pair* $\mathcal{R}_{t_2}^+$ iff $j = \arg\min_j d_{ij}$ and $d_{ij} \leq \varepsilon_{\text{pair}}$ where $\varepsilon_{\text{pair}}$ is a pre-defined threshold, or (2) *nature pair* $\mathcal{R}_{t_2}^\circ$ if $d_{ij} \leq \varepsilon_{\text{pair}}$, otherwise (3) *negative pair* $\mathcal{R}_{t_2}^-$. The pairing loss is calculated as:

$$\mathcal{L}_{\text{p}} = \mathbb{E}_{\mathbf{r}_{(i,t_1)}}\left[-\log\left(\frac{\sum_{\mathbf{r}\in\mathcal{R}_{t_2}^+} \exp\left(\frac{\mathbf{r}_{(i,t_1)}^{\text{feat}} \odot \mathbf{r}^{\text{feat}}}{\tau}\right)}{\sum_{\mathbf{r}\in\mathcal{R}_{t_2}^+\cup\mathcal{R}_{t_2}^-} \exp\left(\frac{\mathbf{r}_{(i,t_1)}^{\text{feat}} \odot \mathbf{r}^{\text{feat}}}{\tau}\right)}\right)\right] \tag{10}$$

where τ is a pre-defined constant. Note that the *nature pairs* are not contributed to this loss.

To accelerate convergence, we apply the same loss to the *correlated descriptors* $\bar{\mathcal{R}}$ as well as key-points $\mathcal{P}$, denoted as *Coarse Pairing Loss* $\mathcal{L}_{\text{c}}$ and *Backbone Auxiliary Loss* $\mathcal{L}_{\text{b}}$.

Offset Loss. Following the definition of three pair types above but with a different threshold $\varepsilon_{\text{offset}}$, we use both *positive* and *nurture* pairs of *correlated descriptors* $\bar{\mathcal{R}}$ to train the Offset Head to predict the offsets.

$$\mathcal{L}_{\text{o}} = \mathbb{E}_{\mathbf{r}_i}\left[\frac{1}{\left|\bar{\mathcal{R}}_{t_2}^+ \cup \bar{\mathcal{R}}_{t_2}^\circ\right|} \sum_{\bar{\mathcal{R}}_{t_2}^+ \cup \bar{\mathcal{R}}_{t_2}^\circ} \left\|\delta_{i,j} - \delta_{i,j}^*\right\|_\Sigma\right] \tag{11}$$

where $\delta_{i,j}$ represents the predicted offset from $\bar{\mathbf{r}}_{(i,t_1)}$ to $\bar{\mathbf{r}}_{(j,t_2)}$ in t_1 coordinate system, and δ_i^* is its ground-truth. $\|\cdot\|_\Sigma$ represents the Mahalanobis distance. We utilize both *positive* and *neutral* pairs with a different distance threshold ε_{o} to accelerate the convergence and improve the robustness.

Overlap Loss. We use Binary Cross Entropy (BCE) loss $\mathcal{L}_{\text{d}}$ to train the Overlap Head.

Training Procedure. We utilize two-phase training procedure discussed in Zhang et al. [62]. Phase one aims to train the registration ability of our method. We randomly sample frame pairs within 20 m from dataset, and use the loss $\mathcal{L} = \lambda_{\text{p}}\mathcal{L}_{\text{p}} + \lambda_{\text{c}}\mathcal{L}_{\text{c}} + \lambda_{\text{b}}\mathcal{L}_{\text{b}} + \lambda_{\text{o}}\mathcal{L}_{\text{o}}$ to train *DeepPointMap2*. Phase two aims to train the loop-detection ability. Thus we randomly sample frame pairs with a distance less/greater than $\varepsilon_{\text{loop}}$ with equal probability. In this phase, only the Overlap Head is trained with the loss of $\mathcal{L}_d$ whereas other modules are frozen.

Table 1: Localization Accuracy on KITTI Odometry Benchmark (Trans↓ and Rot↓).

Modality	Method	06		07		08		09		10	
		Trans	Rot	Trans	Rot	Trans	Rot	Trans	Rot	Trans	Rot
LiDAR	LOAM [60]	0.65	-	0.63	-	1.12	-	0.77	-	0.79	-
	LO-Net [22]	-	-	0.56	0.45	1.08	0.43	0.77	0.38	0.92	0.41
	ISC-LOAM [53]	0.76	0.41	0.56	0.43	1.20	0.50	1.40	0.59	1.87	0.62
	SC-LeGO-LOAM [19]	2.54	1.15	2.48	1.78	2.30	1.24	5.37	2.78	10.50	3.79
	F-LOAM [52]	0.84	0.33	0.88	0.62	0.87	0.33	1.03	0.32	1.20	0.29
	LiODOM [14]	0.83	0.29	0.88	0.61	0.86	0.33	1.03	0.32	1.20	0.29
	LiLO [48]	0.54	0.32	0.60	0.61	1.07	0.41	0.63	0.32	0.99	0.33
Camera	VISO2 [16]	0.79	0.51	1.46	1.13	1.62	0.66	0.84	0.64	1.29	0.64
	ORB-SLAM2 [32]	0.89	0.27	0.89	0.50	1.03	0.31	0.86	0.25	0.62	0.29
	VINS-Fusion [37, 38]	1.35	0.71	1.21	0.90	1.83	0.72	1.82	0.53	2.64	1.01
	OV2-SLAM [12]	1.13	0.28	1.03	0.57	1.11	0.31	0.96	0.20	0.52	0.18
	SOFT2 [7]	0.60	0.23	0.45	0.29	0.91	0.26	0.75	**0.22**	0.74	**0.24**
LiDAR+Camera	DEMO [57]	0.96	-	1.16	-	1.24	-	1.17	-	1.14	-
	DVL-SLAM [43]	0.92	-	1.26	-	1.32	-	0.66	-	0.70	-
	Huang et al. [18]	0.61	-	0.56	-	1.27	-	1.06	-	0.83	-
	LAMV-SLAM [55]	0.49	-	0.84	-	1.19	-	0.80	-	**0.55**	-
	DeepPointMap2	**0.47**	**0.20**	**0.39**	**0.25**	**0.77**	**0.22**	**0.62**	0.23	0.75	0.40

5 EXPERIMENTAL ANALYSIS

5.1 Settings

Datasets. Our experiments utilize three multi-modal autonomous driving-oriented datasets: (1) The KITTI Odometry Dataset [15], a widely used benchmark containing 11 LiDAR-Camera sequences (`00-10`), encompassing diverse scenarios from urban to highway environments. (2) KITTI-360 [24], a large dataset with 9 LiDAR-Camera sequences that introduce challenges with longer distances and more complex environment. (3) KITTI-Carla [8], a simulated dataset with 6 noise-free LiDAR-Camera sequences generated by Carla [9] simulator, which are used to assist training.

Settings. The model is trained on 6× RTX 3090 GPUs, with AdamW optimizer [40], initial $lr = 1 \times 10^{-3}$, $wd = 1 \times 10^{-4}$, and cosine scheduler. The training set contains the first 6 sequences of KITTI Odometry dataset (`00-05`), the first 6 sequences of KITTI-360 dataset (`00`, `02-06`), and the entire KITTI-Carla dataset (`Town01-06`). Since the ground-truth label in the original KITTI Odometry is not ideal, we utilize a more precise label provided in SemanticKITTI [3] to train our model. However, we still use the original KITTI ground-truth for evaluation to make a fair comparison. The evaluation model is trained for 21 epochs for Phase One and another 10 for Phase Two. During Phase Two the lr and wd are decayed with a rate of 0.1. We set the loss weight λ_p,λ_c,λ_b,λ_o = 1,0.1,0.1,1, the threshold ϵ_{pair} = 1m, ϵ_{offset} = 2m and ϵ_{loop} = 20m.

Metrics. We adopt the official metrics of each benchmark for quantitative evaluation: We use *Relative Translation Error* (Trans↓) (%) and *Average Rotation Error* (Rot↓) (°/100m) to measure relative localization accuracy in KITTI Odometry benchmark, and use the *Mean Absolute Pose Error* (APE↓) (m) to evaluate the global trajectory accuracy for KITTI-360.

5.2 Localization Accuracy

This experiment aims to demostrate the localization accuracy of our proposed method, *DeepPointMap2*, in various road scenarios. The experiment is conducted on five KITTI Odometry sequences (`06-10`) with comparison methods divided into LiDAR-based, Visual-Based and LiDAR-Visual-based groups, where each group contains multiple widely-used and advanced SLAM approaches.

As presented in Tab. 1, *DeepPointMap2* achieves the lowest translation error in four sequences and the lowest rotation error in three sequences. Some methods do not report rotation error metrics, which are marked with “-”. Sequence `08` is the longest sequence among them, covering a larger urban area and containing multiple loops. In such complex scenario, *DeepPointMap2* significantly outperforms all LiDAR-Visual-based methods, reducing translation error by 35% compared to the existing SOTA method LAMV-SLAM [55].

To illustrate the superiority of our proposed method in large-scale scenes, we select two representative sequences (`07` and `09`) from the KITTI-360 benchmark, and compare *DeepPointMap2* with the recent SOTA LiDAR-based method DeepPointMap. Sequence `07` was collected in a highway/urban roadway with a length of 4.9 km. The absence of loop closures in this sequence presents a challenge for odometry. Sequence `09`, on the other hand, was collected in a complex, large-scale urban environment, with a trajectory length of over 10.5 km. The environmental complexity and numerous loops challenge the model’s loop closure capability.

As shown in Fig. 5, our *DeepPointMap2* demonstrates an advantage in global trajectory estimation error by achieving the ATE of 26.00 in sequence `07`, which is better than the APE of 93.77 achieved by DeepPointMap. The challenge in the highway scenario lies in the monotonous geometric patterns, which make accurate odometry difficult when relying solely on LiDAR point clouds. However, the inclusion of the visual modality, with its rich textural information,

Table 2: Robustness of *DeepPointMap2* when Camera Unavailable and LiDAR Obstruction (Trans↓) and Rot↓

		06		07		08		09		10		**Mean**	
Scenario	**Frame%**	Trans	Rot	Trans	Rot	Trans	Rot	Trans	Rot	Trans	Rot	Trans	Rot
Camera Failure	5%	0.50	0.24	0.38	0.22	0.76	0.21	0.70	0.27	0.76	0.42	0.62 +3%	0.27 +5%
	30%	0.46	0.20	0.36	0.21	0.86	0.26	0.66	0.27	0.71	0.28	0.61 +2%	0.24 -6%
	50%	0.53	0.25	0.47	0.30	0.88	0.28	0.85	0.35	0.72	0.38	0.69 +15%	0.31 +20%
	100%	0.64	0.30	0.50	0.33	1.46	0.44	1.76	0.51	0.83	0.49	1.04 +73%	0.41 +59%
LiDAR Obstruction	5%	0.44	0.19	0.37	0.22	0.81	0.23	0.70	0.26	0.80	0.35	0.62 +4%	0.25 -4%
	30%	0.43	0.17	0.47	0.30	0.96	0.24	0.81	0.27	0.64	0.32	0.66 +10%	0.26 +0%
	50%	0.40	0.17	0.55	0.35	0.92	0.26	0.85	0.29	0.85	0.39	0.71 +19%	0.29 +12%
	100%	0.37	0.16	0.53	0.40	1.03	0.34	0.86	0.29	0.89	0.37	0.74 +23%	0.31 +20%
Normal Input		0.47	0.20	0.39	0.25	0.77	0.22	0.62	0.23	0.75	0.40	0.60	0.26

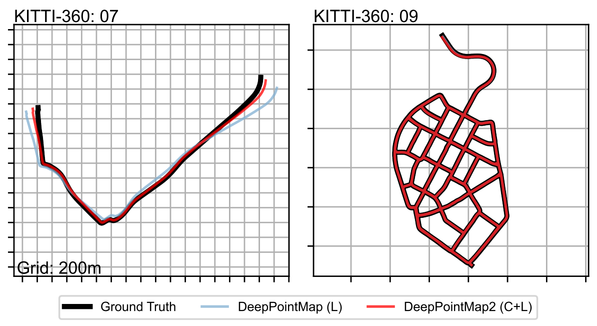


Figure 5: Estimated Trajectories on KITTI-360 Benchmark.

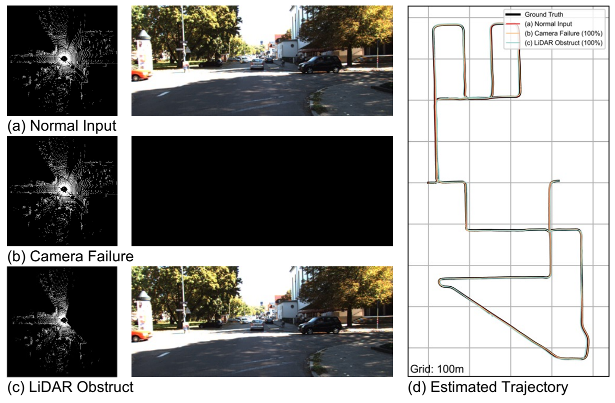


Figure 6: Robustness of *DeepPointMap2*.

enhances the representation ability and, consequently, improves odometry accuracy in such scenarios. As for sequence 09, our method successfully achieves accurate localization and constructs a precise map in large-scale complex urban environments, showcasing its robustness and adaptability.

5.3 Strong Robustness

In certain real-world scenarios, one of the sensors (*i.e.*, LiDAR or camera) may be temporarily unavailable or experience a degradation in performance, posing significant challenges for multi-modal SLAM models. To investigate the robustness of our approach, we design two additional experiments to evaluate our multi-modal *DeepPointMap2* under these conditions, without any finetuning.

In the first experiment, we simulate camera failure scenarios by randomly selecting x% (x=5,30,50,100) of frames and blacking out their image input, as shown in Fig. 6 (b). Quantitative results indicate that *DeepPointMap2* consistently produces remarkable results as image modalities are missing, as shown in Tab. 2. Even when 50% of the images are absent, our approach incurs only an average performance decrement of approximately 15%. Furthermore, when all image modalities are unavailable, the model still functions normally in most scenes (*e.g.*, sequences 06, 07 and 10) and demonstrates competitive performance with other methods on sequence 08 and 09. Meanwhile, we also evaluate the robustness of our method in scenarios where the LiDAR point cloud is obstructed by large vehicles (*e.g.*, trucks or buses), as demonstrated in Fig. 6 (c). To simulate partial obstruction, we randomly select x% of frames and perform the following steps: (1) generate a random-sized and -oriented 3D box at a random location, and (2) remove all LiDAR points that pass through this box. Despite this challenge, our *DeepPointMap2* successfully maintains robust and consistent localization performance. This is attributed to the method's ability to extract discernible features and the adaptability of the cross-modal fusion process, which is further enhanced by the implementation of the *RandomOcclusion* data augmentation technique Even in situations where all frames are affected by occlusion, our method's performance exhibits only a modest average decline of approximately 23%, and it remains competitive when compared to other SOTA Visual-LiDAR-based methods, as evidenced by the quantitative analysis presented in Tab. 2. The visualization of the localization and mapping result on KITTI sequence 08 with the LiDAR obstruction rate and camera failure rate of 100% are shown in Fig. 6 (right). It can be observed that under such harsh scenarios, *DeepPointMap2* still successfully reconstructed the challenging scenes.

5.4 Ablation Study

Reference-Point Generation. As mentioned in Sec. 3.2.1, we use *farthest-point-sample* strategy to sample $n = 256$ reference-points $\mathbf{r}^{\text{xyz}}$ from the input point cloud. In addition, we adapt *uniform-sampling* and *normal-sampling* methods to generate ref-points with

Table 3: Reference Point Generate Strategies (Trans↓).

Dist.	Num.	06	07	08	09	10
FPS	128	0.49	**0.38**	0.94	0.85	0.78
	256	**0.47**	0.39	**0.77**	**0.62**	**0.75**
	512	0.57	0.41	0.88	0.74	0.85
Uniform	256	25.97	31.91	13.48	13.50	13.98
	4096	6.92	6.25	10.79	9.86	16.58
Normal	256	25.19	5.21	11.06	26.71	20.85
	4096	5.83	39.70	18.04	14.88	71.25

different numbers. The *uniform-sampling* variant of the RPG module produces n points drawn from the uniform distribution $\mathbf{r}^{xyz} \sim \mathcal{U}_{[0,1]}$ and scales them linearly to fit the 3D world space. The *normal-sampling* variant follows a similar procedure but generates points from a normal distribution $\mathbf{r}^{xyz} \sim \mathcal{N}(0, 0.25)$.

As indicated in Tab. 3, the localization performance peaks with $n = 256$ ref-points when employing the FPS strategy. Reducing the ref-points number can result in a greater distance between matched descriptors, which may reduce the model's ability to accurately predict the offset values δ. Conversely, increasing the number of ref-points to 512 does not yield a significant enhancement in performance, as the overly close ref-points fail to capture distinct features from the key-points, given that the LiDAR backbone only extracts 256 key-points.

For alternative sampling strategies such as random distribution, their main shortcoming stems from the extensive nature of the point cloud. Many ref-points are generated far from any points in the point cloud (and key points), which poses a challenge for the network to effectively aggregate features for these remote points. Furthermore, the vast scale of the scene leads to a sparse arrangement of ref-points, potentially surpassing the maximum range (*i.e.*, ϵ_{offset}) within which Overlap Head can operate optimally. This scenario compromises the effectiveness of the offset compensation mechanism, thus diminishing overall performance. Due to the aforementioned factors, neither uniform nor normal distributed sampling can construct a reasonable map at $n = 256$. It is only when n is increased to 4096 that the model can achieve minimal localization accuracy and build recognizable maps in benchmark sequences.

Fusion Module Design. In this additional experiment, we explore the importance and advantages of our proposed Biased Transformer. Some existing methods such as PointPainting [51] focus on pixel-level early-fusion, where each LiDAR point is associated with a pixel and the RGB values are attached to the point before processing by the LiDAR backbone. In contrast, other multi-modal models opt for feature-level fusion, integrating features from both image and LiDAR modalities. As the most straightforward approach, RoI-Pooling aggregates the feature within a window and attaches these pooled features to the corresponding points. However, the fixed window size results in a lack of scale invariance, where the pooling region *should but not* appears smaller at a distance and larger when close to the point.

Some attention-based methods can also be used for cross-modal fusion. DeepFusion [23], as one of the SOTA 3D detectors, employs a learnable scale-dot cross-attention module to fuse LiDAR and camera features. Our Biased Transformer introduces two branches that leverage both feature similarity and spatial distance. In this experiment, we disable each branch to investigate its importance. To save the computation, we follow Yin et al. [56] and fine-tune all the ablation models (based on the modal evaluated in Sec. 5.2) on the KITTI dataset for 10 epochs.

Table 4: Attention Module Design (Trans↓).

Fusion Strategy	06	07	08	09	10
Point Painting	0.56	0.56	1.15	0.94	1.55
RoI Pooling (3×3)	2.34	1.56	4.12	4.24	17.83
Scale-Dot Attention	2.73	1.00	9.63	6.48	6.48
w/o Learnable-Attn.	0.72	0.46	18.00	1.16	1.21
w/o Spatial-Attn.	41.54	21.54	29.91	26.28	20.87
Biased Attention (ours)	**0.47**	**0.39**	**0.77**	**0.62**	**0.75**

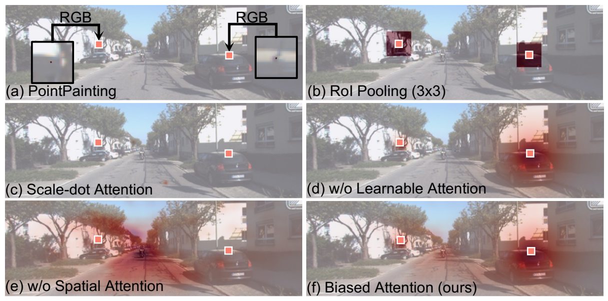

RGB
RGB
(a) PointPainting
(b) RoI Pooling (3x3)
(c) Scale-dot Attention
(d) w/o Learnable Attention
(e) w/o Spatial Attention
(f) Biased Attention (ours)

Figure 7: Attention Map Visualization.

As observed in Tab. 4 and Fig. 7, the performance decreased when replacing the fusion strategy. Although attention-free approaches (*e.g.*, Painting and RoI) have advantages in inference speed, both approaches exhibit weaknesses in localization performance. By removing the *learnable-attention map*, the model relies on the prior Camera-LiDAR calibration and still maintains a relatively high localization accuracy, except for one loop detection error in `seq08`. Removing the *Spatial-Attention Map* brings difficulty of learning the correspondence between images and point clouds, resulting in failure to localize and map in most of the sequences. Finally, the classic scale-dot attention module successfully reconstructs most of the sequences. However, since the spatial information only exists in the positional embeddings, the model is required to learn an appropriate spatial attention map by itself, resulting in slower convergence and reduced performance, compared to Biased Attention.

6 CONCLUSIONS

We present *DeepPointMap2*, a novel learning-based LiDAR-Visual SLAM architecture that leverages a flexible *Visual-Point Fusion Module*. This module adeptly aggregate multi-modal tokens, ensuring precise and resilient performance even in adverse conditions such as LiDAR obstructions and camera failures.

Limitation. The RPG module utilizes a parameter-free strategy, FPS, to obtain initial ref-point. We believe that a learn-based strategy can be used to actively sample these ref-points and avoid sampling points from dynamic objects (*e.g.*, cars).

REFERENCES

[1] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. 2019. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 7163–7172.

[2] K Somani Arun, Thomas S Huang, and Steven D Blostein. 1987. Least-squares fitting of two 3-D point sets. *IEEE Transactions on pattern analysis and machine intelligence* 5 (1987), 698–700.

[3] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. 2019. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*. 9297–9307.

[4] Ke Cao, Ruiping Liu, Ze Wang, Kunyu Peng, Jiaming Zhang, Junwei Zheng, Zhifeng Teng, Kailun Yang, and Rainer Stiefelhagen. 2023. Tightly-Coupled LiDAR-Visual SLAM Based on Geometric Features for Mobile Agents. In *2023 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 1–8.

[5] Daniele Cattaneo, Matteo Vaghi, and Abhinav Valada. 2022. Lcdnet: Deep loop closure detection and point cloud registration for lidar slam. *IEEE Transactions on Robotics* 38, 4 (2022), 2074–2093.

[6] Jiyu Cheng, Chaoqun Wang, and Max Q-H Meng. 2019. Robust visual localization in dynamic environments based on sparse motion removal. *IEEE Transactions on Automation Science and Engineering* 17, 2 (2019), 658–669.

[7] Igor Cvišić, Ivan Marković, and Ivan Petrović. 2022. Enhanced calibration of camera setups for high-performance visual odometry. *Robotics and autonomous systems* 155 (2022), 104189.

[8] Jean-Emmanuel Deschaud. 2021. KITTI-CARLA: a KITTI-like dataset generated by CARLA Simulator. *arXiv preprint arXiv:2109.00892* (2021).

[9] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*. 1–16.

[10] Jakob Engel, Vladlen Koltun, and Daniel Cremers. 2017. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence* 40, 3 (2017), 611–625.

[11] Jakob Engel, Thomas Schöps, and Daniel Cremers. 2014. LSD-SLAM: Large-scale direct monocular SLAM. In *European conference on computer vision*. Springer, 834–849.

[12] Maxime Ferrera, Alexandre Eudes, Julien Moras, Martial Sanfourche, and Guy Le Besnerais. 2021. OV2 SLAM: A fully online and versatile visual SLAM for real-time applications. *IEEE robotics and automation letters* 6, 2 (2021), 1399–1406.

[13] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. 2014. SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 15–22.

[14] Emilio Garcia-Fidalgo, Joan P Company-Corcoles, Francisco Bonnin-Pascual, and Alberto Ortiz. 2022. LiODOM: Adaptive local mapping for robust LiDAR-only odometry. *Robotics and Autonomous Systems* 156 (2022), 104226.

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 3354–3361.

[16] Andreas Geiger, Julius Ziegler, and Christoph Stiller. 2011. Stereoscan: Dense 3d reconstruction in real-time. In *2011 IEEE intelligent vehicles symposium (IV)*. Ieee, 963–968.

[17] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9729–9738.

[18] Shi-Sheng Huang, Ze-Yu Ma, Tai-Jiang Mu, Hongbo Fu, and Shi-Min Hu. 2020. Lidar-monocular visual odometry using point and line features. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1091–1097.

[19] Giseop Kim, Sunwook Choi, and Ayoung Kim. 2021. Scan context++: Structural place recognition robust to rotation and lateral variations in urban environments. *IEEE Transactions on Robotics* 38, 3 (2021), 1856–1874.

[20] Bernd Kitt, Andreas Geiger, and Henning Lategahn. 2010. Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. In *2010 ieee intelligent vehicles symposium*. IEEE, 486–492.

[21] Georg Klein and David Murray. 2007. Parallel tracking and mapping for small AR workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 225–234.

[22] Qing Li, Shaoyang Chen, Cheng Wang, Xin Li, Chenglu Wen, Ming Cheng, and Jonathan Li. 2019. Lo-net: Deep real-time lidar odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8473–8482.

[23] Yingwei Li, Adams Wei Yu, Tianjian Meng, Ben Caine, Jiquan Ngiam, Daiyi Peng, Junyang Shen, Yifeng Lu, Denny Zhou, Quoc V Le, et al. 2022. Deepfusion: Lidar-camera deep fusion for multi-modal 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17182–17191.

[24] Yiyi Liao, Jun Xie, and Andreas Geiger. 2022. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[25] Jiarong Lin and Fu Zhang. 2022. R3LIVE: A Robust, Real-time, RGB-colored, LiDAR-Inertial-Visual tightly-coupled state Estimation and mapping package. In *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 10672–10678.

[26] Jiarong Lin, Chunran Zheng, Wei Xu, and Fu Zhang. 2021. R2 LIVE: A Robust, Real-Time, LiDAR-Inertial-Visual Tightly-Coupled State Estimator and Mapping. *IEEE Robotics and Automation Letters* 6, 4 (2021), 7469–7476.

[27] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2117–2125.

[28] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11976–11986.

[29] Bruce D Lucas and Takeo Kanade. 1981. An iterative image registration technique with an application to stereo vision. In *IJCAI'81: 7th international joint conference on Artificial intelligence*, Vol. 2. 674–679.

[30] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. 2017. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and automation (ICRA)*. IEEE, 4628–4635.

[31] Lingbo Meng, Chao Ye, and Weiyang Lin. 2022. A tightly coupled monocular visual lidar odometry with loop closure. *Intelligent Service Robotics* 15, 1 (2022), 129–141.

[32] Raul Mur-Artal and Juan D Tardós. 2017. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics* 33, 5 (2017), 1255–1262.

[33] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. 2011. DTAM: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*. IEEE, 2320–2327.

[34] Yue Pan, Pengchuan Xiao, Yujie He, Zhenlei Shao, and Zesong Li. 2021. MULLS: Versatile LiDAR SLAM via multi-metric linear least square. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 11633–11640.

[35] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

[36] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. 2022. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems* 35 (2022), 23192–23204.

[37] Tong Qin, Peiliang Li, and Shaojie Shen. 2018. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics* 34, 4 (2018), 1004–1020.

[38] Tong Qin and Shaojie Shen. 2018. Online temporal calibration for monocular visual-inertial systems. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 3662–3669.

[39] Reza Radmanesh, Ziyin Wang, Vishnu S Chipade, Gavriil Tsechpenakis, and Dimitra Panagou. 2020. LIV-LAM: LiDAR and visual localization and mapping. In *2020 American Control Conference (ACC)*. IEEE, 659–664.

[40] S. J. Reddi, S. Kale, and S. Kumar. 2019. On the Convergence of Adam and Beyond.

[41] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision*. Ieee, 2564–2571.

[42] Tixiao Shan and Brendan Englot. 2018. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4758–4765.

[43] Young-Sik Shin, Yeong Sang Park, and Ayoung Kim. 2020. DVL-SLAM: Sparse depth enhanced direct visual-LiDAR SLAM. *Autonomous Robots* 44, 2 (2020), 115–130.

[44] Chengfu Shu and Yutao Luo. 2022. Multi-modal feature constraint based tightly coupled monocular visual-lidar odometry and mapping. *IEEE Transactions on Intelligent Vehicles* (2022).

[45] Frank Steinbrücker, Jürgen Sturm, and Daniel Cremers. 2011. Real-time visual odometry from dense RGB-D images. In *2011 IEEE international conference on computer vision workshops (ICCV Workshops)*. IEEE, 719–722.

[46] Xiang-Shi Tang and Teng-Hu Cheng. 2023. F-LVINS: Flexible Lidar-Visual-Inertial Odometry Systems. *IEEE Access* (2023).

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[48] Edison P Velasco-Sánchez, Miguel Ángel Muñoz-Bañón, Francisco A Candelas, Santiago T Puente, and Fernando Torres. 2023. LiLO: Lightweight and low-bias LiDAR Odometry method based on spherical range image filtering. *arXiv preprint arXiv:2311.07291* (2023).

[49] Ignacio Vizzo, Xieyuanli Chen, Nived Chebrolu, Jens Behley, and Cyrill Stachniss. 2021. Poisson surface reconstruction for LiDAR odometry and mapping. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5624–5630.

[50] Rafael Grompone Von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall. 2008. LSD: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence* 32, 4 (2008), 722–732.

[51] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. 2020. Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF*

conference on computer vision and pattern recognition. 4604–4612.

[52] Han Wang, Chen Wang, Chun-Lin Chen, and Lihua Xie. 2021. F-loam: Fast lidar odometry and mapping. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4390–4396.

[53] Han Wang, Chen Wang, and Lihua Xie. 2020. Intensity scan context: Coding intensity and geometry relations for loop closure detection. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2095–2101.

[54] Haodong Xiang, Xiaosheng Zhu, Wenzhong Shi, Wenzheng Fan, Pengxin Chen, and Sheng Bao. 2022. DeLightLCD: A Deep and Lightweight Network for Loop Closure Detection in LiDAR SLAM. *IEEE Sensors Journal* 22, 21 (2022), 20761–20772.

[55] Jun Yin, Dongting Luo, Fei Yan, and Yan Zhuang. 2022. A novel lidar-assisted monocular visual SLAM framework for mobile robots in outdoor environments. *IEEE Transactions on Instrumentation and Measurement* 71 (2022), 1–11.

[56] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. 2021. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11784–11793.

[57] Ji Zhang, Michael Kaess, and Sanjiv Singh. 2017. A real-time method for depth enhanced visual odometry. *Autonomous Robots* 41 (2017), 31–43.

[58] Ji Zhang and Sanjiv Singh. 2014. LOAM: Lidar odometry and mapping in real-time.. In *Robotics: Science and Systems*, Vol. 2. Berkeley, CA, 1–9.

[59] Ji Zhang and Sanjiv Singh. 2015. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2174–2181.

[60] Ji Zhang and Sanjiv Singh. 2017. Low-drift and real-time lidar odometry and mapping. *Autonomous Robots* 41 (2017), 401–416.

[61] Lilian Zhang and Reinhard Koch. 2013. An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency. *Journal of visual communication and image representation* 24, 7 (2013), 794–805.

[62] Xiaze Zhang, Ziheng Ding, Qi Jing, Yuejie Zhang, Wenchao Ding, and Rui Feng. 2024. DeepPointMap: Advancing LiDAR SLAM with Unified Neural Descriptors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 10413–10421.

[63] Chunran Zheng, Qingyan Zhu, Wei Xu, Xiyuan Liu, Qizhi Guo, and Fu Zhang. 2022. FAST-LIVO: Fast and tightly-coupled sparse-direct LiDAR-inertial-visual odometry. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4003–4009.