2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

BE-SLAM: BEV-Enhanced Dynamic Semantic SLAM with Static Object Reconstruction

Jun Luo, Gang Wang, Hongliang Liu, Lang Wu, Tao Huang, Dengyu Xiao, Huayan Pu, Jun Luo

Abstract— **The quality of a robot's environmental perception determines whether it can achieve more intelligent applications, such as semantic interaction with humans. SLAM, on the other hand, is one of the crucial capabilities for a robot to perceive its environment. However, when only a monocular image is provided, dynamic objects in the environment significantly impact the accuracy of map construction by the robot, leading to erroneous perception results. To address this issue, we propose a Visual SLAM framework based on BEV perception results, named BE-SLAM. With this framework, we can handle dynamic objects, occlusions, and incompletely observed objects. It can construct a stable static map by strengthening trust in static objects. Considering that object-level semantic maps can enhance a robot's perception abilities, we also reconstruct static objects in the map and use them to optimize the pose. Through experiments on existing publicly available dataset, we compare BE-SLAM with several existing methods that have shown good performance. The experimental results demonstrate that BE-SLAM performs exceptionally well on high-dynamic sequences and achieves comparable results on static or low-dynamic sequences.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) is an essential capability for mobile robots to perceive and understand their environment [1]. It employs sensors such as LiDAR or cameras to gather data, which is subsequently utilized to estimate the position and orientation of the robot, facilitating the construction of a map of the unknown environment it navigates [3]. In recent years, Visual SLAM, which emphasizes cameras due to their abundant data and cost effectiveness, has garnered considerable research interest [2]. In certain specific conditions, such as scenes with consistent lighting and no dynamic objects, existing Visual SLAM methods (such as ORB-SLAM2 [4], ORB-SLAM3 [5], DM-VIO [6], etc.) are already capable of solving most problems [7]. Hence, the current challenges involve tackling dynamic scenes and enabling sophisticated semantic interaction between robots and their environment [8].

The future of intelligent mobile robots lies in their ability to interact with the environment. Handling dynamic objects during this interaction is crucial for constructing a stable static map. Dynamic objects can cause robots to perform wrong feature matching and incorrect loop closure detection, which is devastating for robot perception results [8]. The fundamental problem is that the removal of dynamic objects is not perfect, or the wrong judgment of dynamic objects. Challenges such as incomplete observations of dynamic objects make constructing static maps difficult. To address this problem, the Bird's Eye View (BEV) provide a new means to accurately perceive and better understand the three-dimensional shape of the environment. The understanding of semantic information in a scene is a crucial factor in assessing the intelligence of robots [11]. In our approach, we opt to reconstruct the three-dimensional shapes of all static scenes (including static environment and static objects), as this is the most straightforward method to convey semantic information [12] [13].

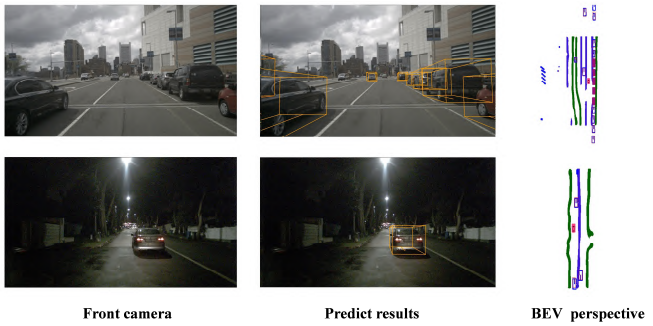

Fig. 1: Considering that the depth uncertainty of a monocular camera increases with distance, we perform checks on nearby vehicles to ascertain whether they are static objects. Within the BEV perspective, we display the outcomes of the BEV module: red boxes signify the results of detection, blue boxes indicate precise locations, and concurrently, stationary objects are denoted with red shading.

In this paper, we propose a new Visual SLAM system based on BEV perception. The system can enhance the perception capability of robots in dynamic scenes, and we name it BE-SLAM. BE-SLAM mainly utilizes the results of BEV perception models to remove dynamic objects and construct high-precision maps. Fig. 1 shows some of static globally consistent BEV perception results. The overall system framework is shown in Fig. 2. The main contributions of this paper are as follows:

- A complete BEV-based Visual SLAM system, named BE-SLAM, which consists of BEV module, semantic module, object reconstruction module, and Visual SLAM module. The robot can use this system to exclude dynamic objects and construct an environmental map with reconstructed static objects.
- A robust method for mask generation based on BEV perception results is proposed. This method can

This work is supported in part by the National Natural Science Foundation of China (Nos. 62203072, 62325302, U2013202 and 62033001) and the China Postdoctoral Science Foundation (No. 2022M710516).
(Corresponding author: Gang Wang. E-mail: wangg@cqu.edu.cn).
J. Luo, G. Wang, H. L. Liu, L. Wu, T. Huang, D. Y. Xiao, H. Y. Pu, and J. Luo are with the State Key Laboratory of Mechanical Transmission for Advanced Equipment, College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing 400044, China. (e-mail: 202207131098t@stu.cqu.edu.cn; wangg@cqu.edu.cn; 20200701049g@cqu.edu.cn; wulangcqu@cqu.edu.cn; huangtao@stu.cqu.edu.cn; xiaodengyu@cqu.edu.cn; phygood_2001@shu.edu.cn; luoj@cqu.edu.cn)

generate 3D bounding boxes for dynamic objects in 2D images using only the BEV perception results, and can also generate more detailed masks by utilizing optional LiDAR data.

- By testing our proposed algorithm on existing dataset, we obtained more accurate semantic map construction results, demonstrating the effectiveness of BE-SLAM.

The structure of this paper is as follows: Section II introduces some related works. Section III provides a detailed description of our proposed method. Section IV describes the experiments and analyzes the experimental results. Finally, in Section V, a summary of the entire paper is presented.

II. Related Works

In this section, we have reviewed existing studies relevant to our contribution. We recommend interested readers to refer to the original papers mentioned in the references section, and [8] for a complete recent review.

A. BEV Perception

LSS [14] is a classic framework for bottom-up perception methods. It accurately estimates the depth distribution of feature points obtained through plane subsampling, generates frustums containing image features, fuses the frustums of all cameras using intrinsic and extrinsic parameters, and processes the generated feature map to obtain perception results. The top-down approach predefines the BEV feature map and utilizes the global perception capability of the Transformer to query relevant information from features in multiple perspective images, fusing and updating the information into the BEV feature map [10][15][16]. BEVFormer [17] represents dense feature modeling in BEV and incorporates temporal feature fusion to address target occlusion or instability. We utilize the high accuracy of BEV perception to detect objects in input images, determine if they are in motion based on the perception results, and decide whether to reconstruct these objects.

B. Object-level SLAM

SLAM++ [18] use a pre-established model database, resulting in highly accurate and complete globally consistent object-level maps. MaskFusion [19] performs real-time object reconstruction without relying on prior model information. However, this real-time reconstruction may not fully capture the entire object. CubeSLAM [20] and QuadricSLAM [21] use simple rectangular and elliptical bounding boxes, respectively, to represent objects, enabling complete reconstruction of encountered objects. BE-SLAM builds upon BEV perception to reconstruct static objects in the scene, allowing for the acquisition of more object information and the reconstruction of additional objects, including statically occluded and partially observed objects.

C. 3D Object Reconstruction

Explicit representation involves reconstructing objects using point clouds, voxels, or meshes. Implicit representation relies on neural networks for object reconstruction [23].

DeepSDF [24] learns from partially input and noisy three-dimensional data to achieve high-quality shape representation, interpolation, and completion. NeRF [25] uses a ray-tracing method to render implicit representation into two-dimensional images. This process includes emitting rays, sampling RGB and density information, integrating these values, and synthesizing images. We use a similar rendering method as DeepSDF, which allows BE-SLAM to reconstruct static objects in the map.

III. Method

Our system was developed based on inspiration from the ORB-SLAM2 and DSP-SLAM [22]. ORB-SLAM2 is a widely studied Visual SLAM framework, while DSP-SLAM is a Visual SLAM system that enhances semantic information, increases the reusability of the map, and reconstructs objects. Our proposed framework constructs a sparse map of the environment, rejecting feature points extracted from the dynamic objects to prevent the impact of feature matching imprecision. Nevertheless, the framework still reconstructs some static objects, such as parked cars and use them in Visual SLAM Module.

A. BEV Module

Considering the detection of occluded objects, we have chosen a detection algorithm in the BEV space (which also benefits from the physical characteristic of vehicles, i.e., their larger size) instead of other monocular detection algorithms, such as Fcos3D [26] and YOLO. Considering the computational speed and accuracy, we have used BEVFormer as our BEV module. In the future, we plan to replace the existing BEV module with FastBEV [27] to realize the deployment of BE-SLAM on mobile devices. The BEV perception model was pre-trained on the nuScenes dataset [28], which contains 1000 street scenes that are over 20 seconds long, covering common objects such as cars and pedestrians.

Initially, dynamic objects undergo a global consistency check based on the results of the detection process to determine their locations. We solely consider removing high-dynamic objects, such as pedestrians and automobiles. The images obtained after eliminating dynamic objects will be utilized for camera tracking, loop closure detection, and improving object reconstruction accuracy during mapping. During the process of mapping, there is no specific type of object that is continuously in motion, which means there may be potential moving objects. We treat the parked car on the side of the road as a static object, which can assist with camera tracking, although the car's movement may cause some deviation in subsequent loop closure detection.

The BEV perception results can accurately determine whether the car is stationary or moving. This reflects the advantages of BEV perception over traditional geometric methods and general object detection methods, which can further distinguish objects in the environment.

B. Object Semantic Association and Reconstruction

Aligning the BEV perception results with LiDAR data at the same timestamp can enhance the accuracy of the detection results. LiDAR data is optional as the BEV perception results are already sufficient to generate masks for dynamic objects. However, incorporating LiDAR data can refine the size of the masks, making them closer to the original objects. We provide an example of processing LiDAR data using the KITTI dataset [29]. Initially, the LiDAR data is mapped to the

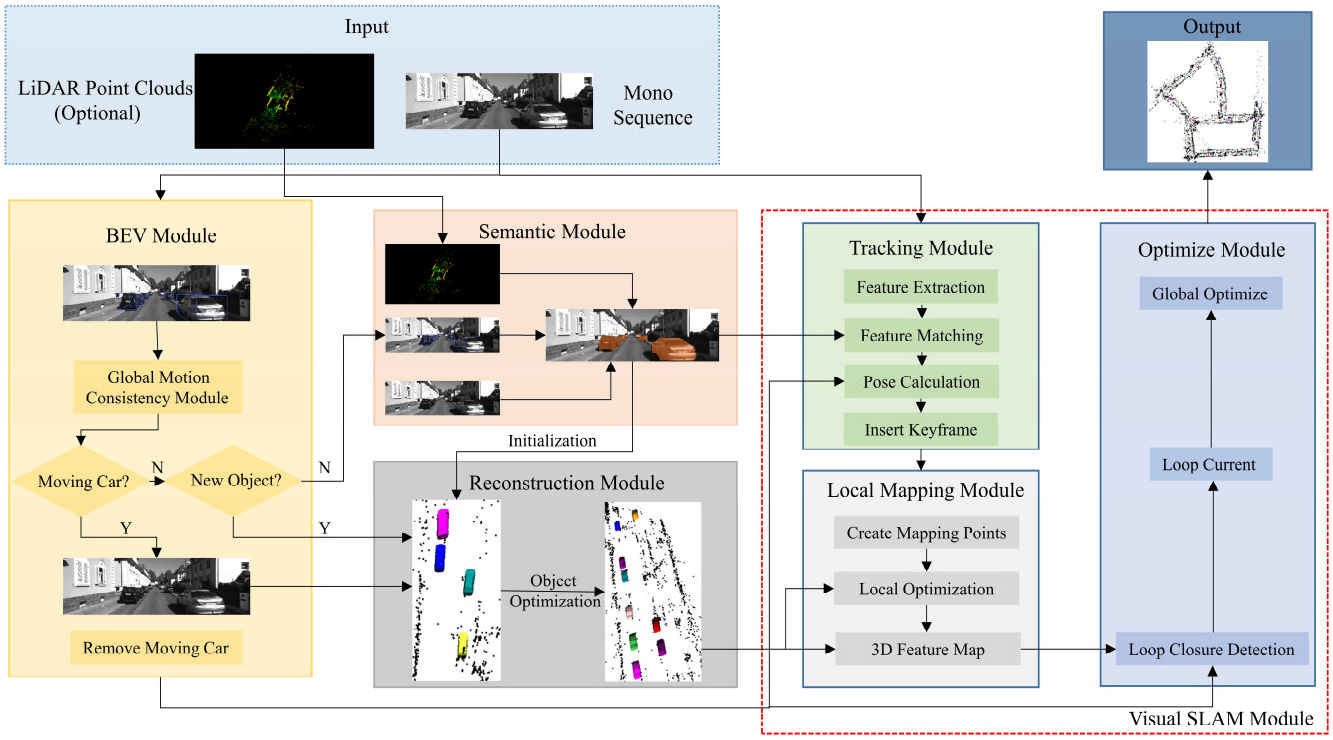

Fig. 2: The system structure of BE-SLAM. BE-SLAM takes monocular image sequences as input, and finally outputs 3D feature maps with semantic information of object reconstruction.

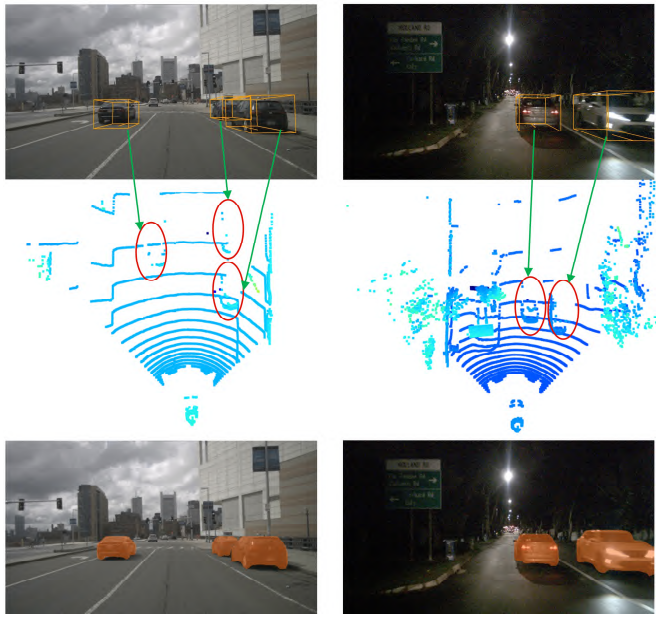

Fig. 3: Considering that static objects can extract more stable features, we used LiDAR data and monocular image data to generate finer masks using Algorithm 1(nuSences dataset).

camera coordinate system. Consider P_3 as the Cartesian coordinates of the laser point cloud, whereas p_3 denotes its homogeneous coordinate form:

$$P_3 = (x_3, y_3, z_3), p_3 = (x_3, y_3, z_3, 1). \quad (1)$$

Where x_3, y_3, z_3 are values in the LiDAR frame. Then the transformed coordinates P_2 can be given by the following:

$$P_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = P_i^{'} R_0 Tr_{lc} p_3 = P_i^{'} R_0 Tr_{lc} \begin{bmatrix} x_3 \\ y_3 \\ z_3 \\ 1 \end{bmatrix}, \quad (2)$$

where Tr_k represents the transformation matrix from the LiDAR coordinate system to the reference camera coordinate system, and R_0 represents the transformation matrix from the camera reference coordinate system to the calibrated camera coordinate system. In general, the transformation of these two matrices can obtain the transformation relationship from the LiDAR coordinate system to the camera coordinate system.

Then, align the point cloud and the image using the camera's field of view and the size of the image. Further, transform the perspective to a top-down view. Following this, the optimization of detection results using BEV perspective LiDAR data is performed. The primary methodology involves employing standard rectangular bounding boxes for matching the point cloud data within this perspective, followed by fine-tuning the rectangular boxes. It is noteworthy that, during the matching process, efforts are made to constrain the point cloud within the confines of the rectangular boxes. Importantly, this process preserves the original values of the detection boxes along the z-axis. Subsequently, the updated 3D information of the objects' bounding boxes is projected back onto the original image.

We generate more accurate masks for objects in the neighboring regions and annotate them in the original image. The algorithm for generating dynamic object masks is shown in Algorithm 1, some results are shown in Fig. 3. The core aspect lies in the processing of BEV detection results. Firstly,

Algorithm 1: Object Mask's Generation Algorithm

input : Original frame, F_1; BEV frame, F_2; BEV frame's box, B_2; LiDAR frame, F_3; LiDAR frame's points, P_3;
output: The mask of dynamic object, M;

if LiDAR frame F_3 is *available* **then**
  Convert the points to camera $P_2 = LiDARtoCam(P_3)$;
  Keep points in front of camera $P_2 = OutlierPoints(P_2)$;
  Adjust to top-down view $P_2 = TopDown(P_2)$;
  Project onto BEV $P_2 = CamtoBEV(P_2)$;
  `// BEV viewpoint alignment of point cloud and image;`
  for each frame F_2 and Box B_2 from *BEV Module* **do**
    $Box_b, BoxNum_b \leftarrow GetBox(F_2, B_2)$;
    $Box_l, BoxNum_l \leftarrow GetBox(F_3, P_3)$;
    `// Matching data preparation;`
    for $i \leftarrow 1$ **to** $BoxNum_b$ **do**
      for $j \leftarrow 1$ **to** $BoxNum_l$ **do**
        $B_{temp} = FineTuningBox(Box_{bi}, Box_{lj})$;
        $B_{ori} = BoxtoOri(B_{temp}, F_1)$;
        $M = BoxtoMask(B_{ori}, F_1)$;
else
  $Box_b, BoxNum_b \leftarrow GetBox(F_2, B_2)$;
  for $i \leftarrow 1$ **to** $BoxNum_b$ **do**
    $B_{temp} = MatchBox(Box_{bi}, Box_{bi})$;
    $B_{ori} = BoxtoOri(B_{temp}, F_1)$;
    $M = BoxtoMask(B_{ori}, F_1)$;
return M;

we map the 3D detection boxes onto the image to obtain a polygonal region. Subsequently, edge feature points are extracted within this region with canny algorithm. Next, the region is segmented using six identically sized rectangles arranged in a clockwise manner, denoted as R_i (i=1, 2, 3, 4, 5, 6). For R_i, a third-degree polynomial is employed to fit the extracted edge points. The curves of every two adjacent regions are connected at their ends to ensure smoothness at the curve intersections. To enforce smoothness at these connections, a smoothness constraint is introduced. The constraint function can be obtained by the (3).

$$\xi = \sum_{j=1}^{M} \left| d_e - d_j \right|, \tag{3}$$

where M is the number of points that share a boundary between R_i and R_{i+1}, d_e represents the direction of the gradient at the edge points on the region, d_j represents the gradient direction of the fitted curve at that point. The goal of optimization is to minimize (3). Finally, the pixel gradients of the extracted valid edge points are utilized to constrain the ultimate contour, resulting in a more accurate mask.

The static object is beneficial for the calculation of the camera pose, because we can extract more stable feature points on the static object. Benefiting from the accurate results of BEV detection, we reconstruct the static object and set the reconstructed object as a landmark to participate in the subsequent pose optimization. In situations where only monocular sequence input is available, we employ the BEV perception module to extract depth information. Alternatively, if LiDAR data is present, we retrieve depth information from the LiDAR sensor. We use the results of motion consistency detection of BEV Module as the input to the algorithm to optimize the pose and reconstruction quality of stationary objects. The algorithm for object reconstruction is shown in Algorithm 2.

C. Visual SLAM Module

BE-SLAM made several changes based on ORB-SLAM2, including accurately removing dynamic objects from the

Algorithm 2: Object Reconstruction Algorithm

input : BEV detetion result, B_{det}; The removed object's picture, P_{re};
output: The full dense shape $\mathbf{z}$; The object pose, $\mathbf{T_{co}}$;

for each object B_{det} from *BEV Module* **do**
  if BEV detetion result B_{det} is *a new object* **then**
    $z, T_{co}, P_2 \leftarrow B_{det}, P_{re}$;
    `// z :the initial shape;`T_{co}`:the initial object pose;`
  else $M \leftarrow$ **Algorithm 1**, $z, T_{co} \leftarrow M$, $P_2 \leftarrow B_{det}, P_{re}$;
  The set of P_2 : Ω;
  Surface consistency term:$E_{surf} = \frac{1}{|P_2|}\sum_{u \in P_2} G^2(T_{co}\pi^{-1}(u, \Omega), z)$;
  Differentiable sdf renderer term:$E_{rend} = \frac{1}{|P_2|}\sum_{u \in P_2}(d_u - \hat{d_u})^2$;
  `//` d_u `is rendered depth;`
  Construct optimization function: $E = \lambda_s E_{surf} + \lambda_r E_{rend} + \lambda_c ||z||$;
  if $E = E_{min}$ **then**
    Reture the optimized shape $\mathbf{z}$;the object pose, $\mathbf{T_{co}}$;
  else Iterate z and T_{co} using the Gauss-Newton method;

environment through the BEV module and object semantic association module to improve the accuracy of pose calculation.

During the feature matching, we increase the trustworthiness of the mask generated by the semantic association module. Specifically, it is to lower the threshold which obtained from multiple experiments of feature point selection in the mask region. This enables us to acquire more feature points in the mask region. Then, we use these feature points to calculate the pose transformation matrix between adjacent frames, resulting in more accurate and robust pose transformations. Finally, we determine whether to insert keyframes based on the strategy of ORB-SLAM2.

We utilize the selected keyframes from the tracking module for providing data to the local map. Then, the local map undergoes optimization, and the reconstructed 3D objects are visualized in the local map to 3D feature map. At the same time, considering the static property of the reconstructed object, we set it as a static landmark and then participates in the local optimization. In the local optimization process, moving map points from different perspectives are removed. Through this method, we solely maintain a keyframe database and a local map containing static 3D objects and map points.

We have enhanced the loop closure detection method of ORB-SLAM2, improving the accuracy by incorporating the results of BEV perception. When determining the occurrence of a loop, we utilize the BEV perception results to improve the accuracy of judgment. In this process, BEV information is used to mark each reconstructed result, and then the reconstructed object is updated and corrected. Specifically, we employ images with dynamically removed objects for the secondary evaluation of co-visible keyframes, thereby increasing the rigor of loop detection. Simultaneously, we more rely on matched feature point pairs derived from the 3D reconstructed objects for accurate pose calculation and correction of the pose transformation matrix in the loop closure keyframes. This means that we give priority to the feature points on the reconstructed object in the loop closure detection.

IV. Experiments and Results

In this section, we utilize experiments to evaluate our proposed method, BE-SLAM. We mainly compared BE-SLAM with some existing methods, and then we analyzed the runtime required for the system's module.

TABLE. I
RMSE COMPARISON OF ABSOLUTE POSE ERROR (APE) AND RELATIVE POSE ERROR (RPE) ON THE KITTI ODOMETRY DATASET. RESULTS IN **BOLD** ARE RELATIVE BEST RESULTS.

Seq.	ORB-SLAM2		ORB-SLAM3		DynaSLAM		DSP-SLAM		BE-SLAM(ours)	
	APE(m)	RPE(m)	APE(m)	RPE(m)	APE(m)	RPE(m)	APE(m)	RPE(m)	APE(m)	RPE(m)
00	0.8812	**0.0273**	0.8496	0.0286	0.9610	0.0275	0.9732	0.0314	**0.8394**	0.0296
01	8.0001	0.0518	10.8700	0.0749	7.7696	0.0495	8.2334	0.0498	**6.9412**	**0.0484**
02	5.7159	0.0284	**2.8285**	0.0345	5.7025	**0.0281**	5.7767	0.0330	4.7370	0.0312
03	0.3098	0.0174	0.3096	0.0172	**0.2743**	**0.0167**	0.3901	0.0218	0.3305	0.0202
04	0.1741	**0.0184**	0.1672	0.0225	0.1487	0.0189	0.1844	0.0210	**0.1248**	0.0186
05	0.4118	0.0159	0.6958	0.0185	0.3657	**0.0154**	0.5194	0.0227	**0.3198**	0.0207
06	0.6862	0.0183	1.2095	0.0326	0.8255	**0.0180**	0.4449	0.0194	**0.2560**	0.0190
07	0.4500	**0.0165**	0.5051	0.0183	0.5066	**0.0165**	0.3286	0.0221	**0.2845**	0.0243
08	3.3544	0.0391	3.8081	0.0395	3.4125	0.0391	3.7111	0.0412	**3.0832**	**0.0378**
09	3.1875	**0.0222**	**1.3537**	0.0439	1.5990	0.0224	3.1451	0.0284	1.4892	0.0298
10	0.9688	0.0205	1.0971	**0.0203**	0.9441	0.0208	1.0123	0.0229	**0.9011**	0.0224
Mea.	2.1945	0.0251	2.1540	0.0319	2.0483	**0.0248**	2.2472	0.0285	**1.7551**	0.0275
Std.	2.4826	0.0107	2.9497	0.0162	2.4123	0.0103	2.5589	0.0092	**2.1393**	**0.0088**

A. Overview of The Experiment

Configuration. Our experiments were conducted on PC equipped with the Ubuntu 20.04 operating system. The specifications of this computer are as follows: the central processor is an Intel Core i9-13900K, and it is equipped with 2 x NVIDIA GeForce RTX 4090 GPU.

Datasets. We primarily evaluated our method on the KITTI dataset because it contains relatively comprehensive sensor information. Additionally, we per-trained the BEV Module of BE-SLAM in the nuScenes dataset.

Metrics. We use absolute pose error (APE) and relative pose error (RPE) to evaluate the accuracy of the camera pose (translation + rotation) obtained by BE-SLAM with scale optimize, compared with ground truth. By using the evo tool, we can also obtain the RMSE for APE and RPE individually.

B. Model Comparison

We compare our proposed method BE-SLAM with some state-of-the-art SLAM methods through experiments. Including ORB-SLAM2[4], ORB-SLAM3 [5], DynaSLAM [9], DSP-SLAM [22]. Table I presents our results from evaluating the KITTI dataset. In the assessment of the entire dataset, we achieved relatively optimal performance in the APE across seven sequences. Compared to other visual SLAM systems, our comprehensive performance has improved by 20.02% (ORB-SLAM2), 18.52% (ORB-SLAM3), 14.23% (DynaSLAM), and 21.90% (DSP-SLAM). This demonstrates that by incorporating the detection results of BEV, BE-SLAM achieves higher tracking precision and benefits from strategies that eliminate dynamic objects and place greater trust in feature points extracted from static objects. Although we did not achieve optimal results in RPE, the difference between our values and the best results is minimal. Overall, BE-SLAM exhibits strong robustness in outdoor scenes and delivers exceptional outcomes in large-scale environments.

Specifically, in certain sequences (e.g., the 06 sequence), BE-SLAM's performance improved by over 50% compared to other methods. A more in-depth analysis reveals that the camera's broad field of view in this sequence is prone to erroneous feature matching. BE-SLAM enhances the reliability of the static objects around it through BEV detection outcomes, obtaining more stable feature matching associations; concurrently, it reduces error impacts by optimizing pose using reconstructed objects. In the 07 sequence, the presence of numerous dynamic objects during the camera's movement, alongside many static cars on the road, posed substantial demands on the object recognition capabilities of the Visual SLAM systems. BE-SLAM's APE accuracy in this sequence improved by 43.84% relative to DynaSLAM, indicating its successful utilization of static objects (including the accurate identification of occluding objects, extraction of feature points, and pose optimization). This also attests to the effectiveness of BE-SLAM's BEV module in distinguishing between dynamic and static objects adeptly. Fig. 4 demonstrates the feature point extraction

Fig. 4: Processing results for dynamic objects. From top to bottom, they are BE-SLAM, ORB-SLAM2, DynaSLAM.

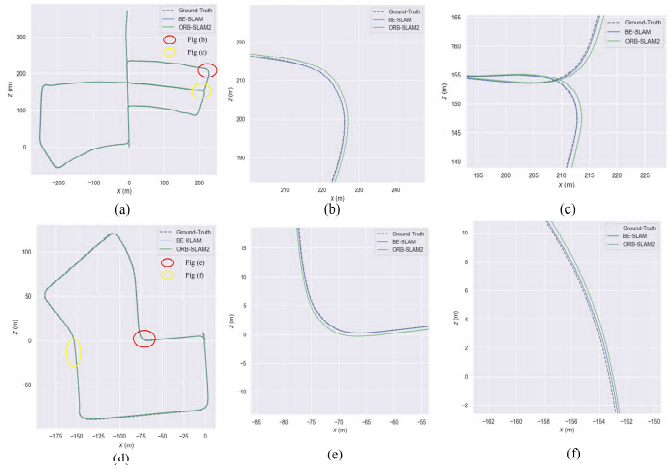

Fig. 5: BE-SLAM and ORB-SLAM2 compared to Ground Truth on KITTI dataset. Where (a) for static scenes at moderate distances, i.e. 05 sequence, (b) and (c) are local enlargements of (a). (d) is the dynamic scene sequence, i.e. 07 sequence, (e) and (f) are local enlargements of (d). These results show the superiority of BE-SLAM in camera trajectory tracking.

Fig. 6: Diagram illustrating the count of cars during a turn (with the top example from 00 sequence, and the bottom from 07 sequence).

performance on dynamic objects by three different SLAM systems within the sequence. DynaSLAM is capable of detecting dynamic objects, albeit with potential blurriness at the edges and imprecise detection for small-sized objects. In contrast, BE-SLAM effectively processes objects within the environment, efficiently removing dynamic objects within the field of view. Moreover, it can stably recognize static objects (such as the car parked on the left side of the road in Fig. 4) and successfully extract feature points from them.

We compared the global trajectory performance of BE-SLAM and ORB-SLAM2 in urban road scenarios (05 sequence). The presence of numerous complex dynamic objects in city roads presents a challenge to the stability of pose estimation. Fig. 5 illustrates the differences between the maps constructed by BE-SLAM and ORB-SLAM2 and the ground truth trajectory. Observation of the magnified trajectories in Fig. 5. (f) reveals that BE-SLAM has superior tracking accuracy compared to ORB-SLAM2. At corners (as shown in Fig. 5. (b) and Fig. 5. (e)), BE-SLAM effectively reduces drift by considering the use of static objects for pose optimization during large-scale movements. Furthermore, BE-SLAM demonstrates greater accuracy in the loop closure

TABLE. II
COMPARISON OF THE ACCURACY OF RECONSTRUCTION RESULT IN THE KITTI DATASET.

Methods		DSP-SLAM			BE-SLAM(Ours)		
Seq.	Tot.	Tru.	Fal.	Per.	Tru.	Fal.	Per.
00	948	494	2	52%	653	0	69%
01	-	-	-	-	-	-	-
02	482	253	3	52%	327	1	68%
03	12	10	0	83%	12	0	100%
04	-	-	-	-	-	-	-
05	261	168	4	64%	231	1	88%
06	118	78	0	66%	110	0	93%
07	186	131	4	70%	150	2	81%
08	821	394	3	48%	579	4	71%
09	213	134	1	63%	182	0	86%
10	48	34	3	70%	41	1	85%
Mea.	310	171	3	63%	229	1	82%
Std.	-	-	-	0.10	-	-	0.10

TABLE. III
AVERAGE TIME PER MODULE

Methods	Semantic(ms)	Tracking(ms)
ORB-SLAM2	/	**28.96**
ORB-SLAM3	/	41.19
DynaSLAM	162.41	41.78
DSP-SLAM	85.84	54.79
BE-SLAM(Ours)	**64.74**	34.37

detection (Fig. 5. (c)), which is closer to ground truth, further validating the effectiveness of static objects.

C. Reconstruction Results Evaluation

To verify the impact of the BEV module on the system, this section mainly compares the accuracy of object reconstruction between BE-SLAM and DSP-SLAM. In BE-SLAM, accurate object reconstruction can provide stable semantic information for pose optimization. We will analyze the precision of the detection and reconstruction results for static objects (i.e., static cars) in different sequences of the KITTI dataset.

We compared the precision of object reconstruction between BE-SLAM and DSP-SLAM by tallying the number of correctly identified cars by each system. The accuracy is calculated using (4):

$$Per. = \frac{Tru.}{Tot.} \times 100\%, \tag{4}$$

where *Per.* represents accuracy, *Tru.* refers to the number of static objects correctly identified by the SLAM system; *Tot.* represents the total number of static cars in the sequence. It is important to note that we only accounted for the neatest car within the field of view, for example, at intersections preparing to turn left, only the nearest static cars are counted, as shown in Fig. 6, we consider the one static car in front. In table II, *Fal.* refers to the number of objects incorrectly identified, such as mistaking moving cars for static ones or misidentifying other objects as cars. This value is given to compare the low number of false identifications of our method.

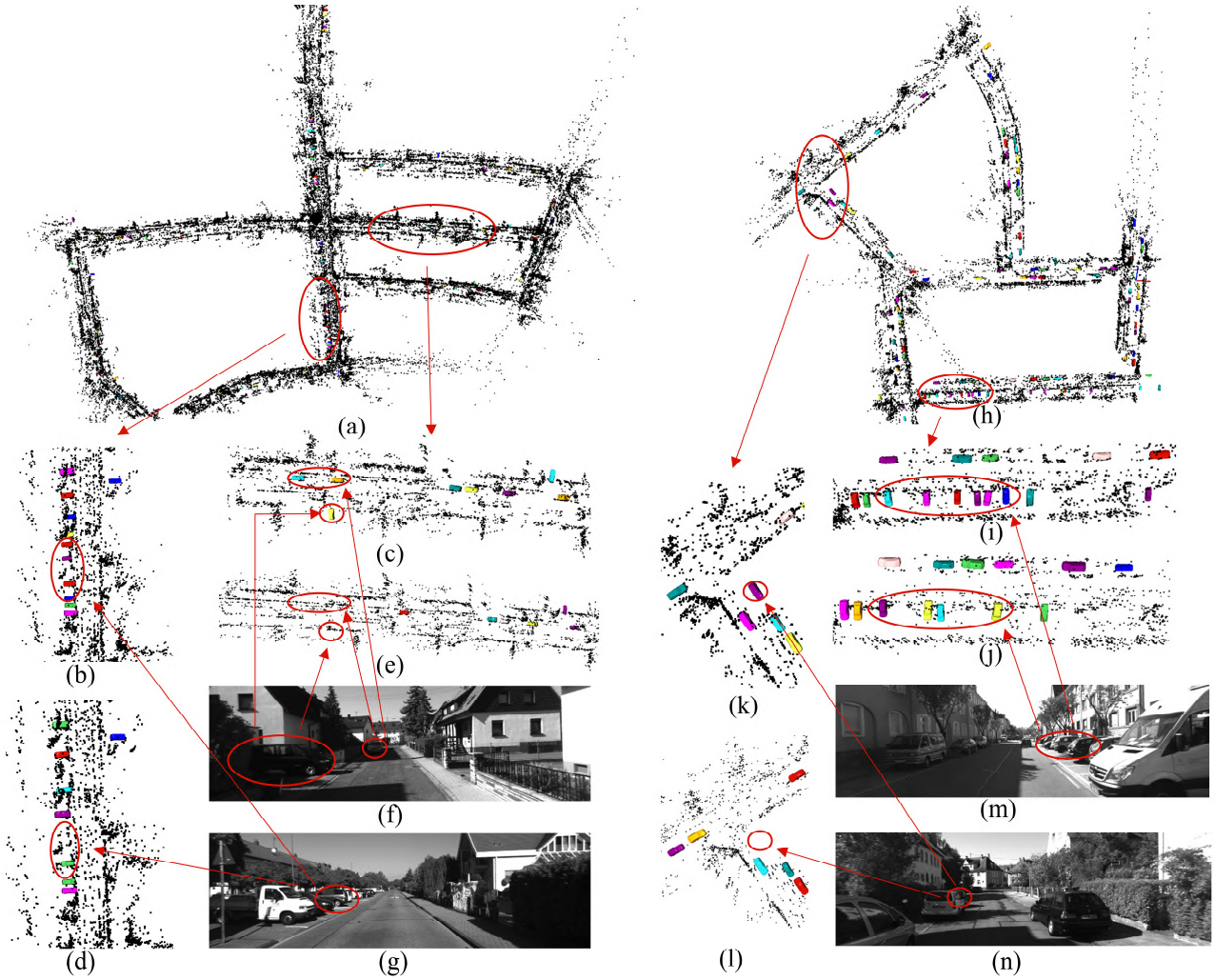

Fig. 7: (a) and (h) are the mapping results of BE-SLAM in the KITTI dataset 05 sequence and 07 sequence respectively. (d), (e), (j), and (l) are the local mapping results of DSP-SLAM in the corresponding data set sequence. It can be seen that BE-SLAM is more sensitive to the occlusion object, and even if only 1/5 of the observation results are obtained, the three-dimensional position of the occlusion object can be obtained and then reconstructed.

Table II presents a comparative assessment between BE-SLAM and DSP-SLAM, illustrating that BE-SLAM surpasses DSP-SLAM in overall accuracy. Specifically, in 00 sequence, BE-SLAM correctly identified 653 cars with no false positives, achieving an accuracy of 69%, compared to DSP-SLAM, which identified 494 cars with 2 false positives, resulting in an accuracy of 51%. In 01 sequence, neither algorithm produced results as the scene corresponded to a highway. In 03 sequence, DSP-SLAM reached its highest detection rate at 83%, while BE-SLAM achieved a perfect detection rate of 100%. Both methods delivered a comparable accuracy of 67% in 04 sequence, yet DSP-SLAM miscategorized some moving cars as stationary, an error not present in BE-SLAM. For sequences 05 to 10, BE-SLAM maintained accuracy rates between 80% and 93%, indicating a higher consistency. On average, BE-SLAM achieved an accuracy rate of 80%, versus 63% for DSP-SLAM, reflecting superior overall performance of BE-SLAM. The close standard deviation values for the two algorithms suggest Fig. 7 shows the differences between BE-SLAM and similar stability. In summary, BE-SLAM demonstrates greater precision in reconstructing stationary objects, offering DSP-SLAM in object reconstruction. BE-SLAM can well deal with the situations that occur during the robot's movement, including occlusion and imperfect observation.

D. Time Consumption Analysis

We evaluated the system time of BE-SLAM and others, the specific data can be found in Table III. All the data in Table III were obtained on the KITTI dataset 07 sequence using the same experimental platform. From the analysis of the data in the table, BE-SLAM demonstrates significant advantages in processing times for both semantic and tracking modules. Specifically, its semantic module averages a processing time of just 64.74 ms, significantly more efficient than DynaSLAM at 162.41 ms and DSP-SLAM at 85.84 ms. Although ORB-SLAM2 shows superior performance in tracking (28.96 ms), among the algorithms (ORB-SLAM3, DynaSLAM, DSP-SLAM) developed upon ORB-SLAM2, BE-SLAM boasts the shortest tracking time (34.37 ms).

V. Conclusion

In this paper, we propose a dynamic Visual SLAM system based on BEV perception results: BE-SLAM. It uses the BEV perception results to remove dynamic objects and strengthen trust in static objects and then reconstructs static objects in the final constructed map, using reconstructed object to optimize pose. In comparison with other algorithms, BE-SLAM achieves the best absolute pose estimation and shows its robustness in both high and low-dynamic scenarios. In the future, we intend to replace the perception method of the BEV module and add IMU data to achieve higher tracking accuracy and enable BE-SLAM to run in real-time on the robot.

References

[1] C. Cesar, C. Luca, C. Henry, L. Yasir, S. Davide, N. Jose, R. Ian, and J. L. John, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," *IEEE Trans. Rob*, vol. 32, no. 6, pp. 1309-1332, 2016.

[2] H. M. Liu, G. F. Zhen, and H. J. Bi, "A survey of monocular simultaneous localization and mapping," *Journal of Computer-Aided Design & Computer Graphics*, vol. 28, no. 6, pp. 855-868, 2016.

[3] Khan, M. U. Zaidi, S. A. A. Ishtiaq, A. Bukhari, S. U. R, S. Samer, and A. Farman, "A comparative survey of lidar-slam and lidar based sensor technologies," in *2021 Mohammad Ali Jinnah University International Conference on Computing*. IEEE, 2021, pp. 1-8.

[4] A. R. Mur, and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Trans. Rob*, vol. 33, no. 5, pp. 1255-1262, 2017.

[5] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Trans. Rob*, vol. 37, no. 6, pp. 1874-1890, 2021.

[6] L. von Stumberg, and D. Cremers, "DM-VIO: Delayed Marginalization Visual-Inertial Odometry," *IEEE Rob. Autom. Lett*, vol. 7, no. 2, pp. 1408-1415.

[7] B. Macario, M. Michel, Y. Moline, G. Corre and F. Carrel, "A Comprehensive Survey of Visual SLAM Algorithms," *Robotics*, vol. 11, no. 24, 2022.

[8] H. Pu, J. Luo, G. Wang, T. Huang, H. Liu, and J. Luo, "Visual SLAM Integration with Semantic Segmentation and Deep Learning: A Review," *in IEEE Sensors Journal*, doi: 10.1109/JSEN.2023.3306371.

[9] B. Bescos, C. Campos, J. D. Tardós, and J. Neira, "DynaSLAM II: Tightly-coupled multi-object tracking and SLAM," *IEEE Rob. Autom. Lett*, vol. 6, no. 3, pp. 5191-5198, 2021.

[10] Li H, Sima C, et al "Delving into the Devils of Bird's-eye-view Perception: A Review, Evaluation and Recipe," *arXiv preprint arXiv:2209.05324,* 2022.

[11] X. Q. Li, W He, S. Q. Zhu, H. Y. Li Yu, and T. Xie, "Survey of simultaneous localization and mapping based on environmental semantic information," *Chinese J. Eng. Des*, vol. 43, no. 6, pp. 754-767, 2021.

[12] W. Chen, G. Shang, A. Ji, C. Zhou, X. Wang, C. Xu, and K. Hu, "An overview on visual slam: From tradition to semantic," *Remote Sensing*, vol. 14, no. 13, pp. 3010, 2022.

[13] K. Chen, J. Zhang, J. Liu, Q. Tong, R. Liu, and S. Chen, S. "Semantic Visual Simultaneous Localization and Mapping: A Survey," *arXiv preprint arXiv:2209.06428*. 2022.

[14] J. Philion and S. Fidler, "Lift splat shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3D," in *Proc. Eur. Conf. Comput. Vis.(ECCV)*, pp. 194-210, 2020.

[15] Y. Wang, V. C. Guizilini, T. Zhang, Y. Wang, H. Zhao and J. Solomon, "DETR3D: 3D object detection from multi-view images via 3D-to-2D queries", in *Proc. Conf. Robot Learn.*, pp. 180-191, 2022.

[16] Y. Liu, T. Wang, X. Zhang and J. Sun, "PETR: Position Embedding Transformation for Multi-View 3D Object Detection," in *Proc. Eur. Conf. Comput. Vis.(ECCV)*, pp. 531-548, 2022.

[17] Z. Li., W. Wang, H. Li, E. Xie, C. Sima, T. Lu, ... & J. Dai, "Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *Proc. Eur. Conf. Comput. Vis.(ECCV)*, pp. 1-18, 2022.

[18] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly and A. J. Davison, "SLAM++: Simultaneous localisation and mapping at the level of objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, IEEE, 2013, pp. 1352-1359.

[19] M. Rünz, M. Buffier, and L. Agapito, "MaskFusion: Real-time recognition tracking and reconstruction of multiple moving objects," *in Proc. IEEE Int. Symp. Mixed Augmented Reality. (ISMAR)*. IEEE, 2018, pp. 10-20.

[20] S. Yang, and S. Scherer, "CubeSLAM: Monocular 3-D object SLAM," *IEEE Trans. Robot*, vol. 35, no. 4, pp. 925-938, 2019.

[21] L. Nicholson, M. Milford, and N. Sünderhauf, "QuadricSLAM: Dual quadrics from object detections as landmarks in object-oriented SLAM," *IEEE Robot. Autom. Lett.*, vol. 4, no. 1, pp. 1-8, 2019.

[22] J. Wang, M. Rünz, and L. Agapito, "DSP-SLAM: Object oriented SLAM with deep shape priors," in *Intl. Conf. on 3D Vision (3DV)*. IEEE, 2021, pp. 1362-1371.

[23] M. G. Kantarci, B. Gökberk and L. Akarun, "A Survey of 3D Object Reconstruction Methods," in *Sig. Proce. Comm. Appl. Conf. (SIU),* 2022, pp. 1-4.

[24] J. J. Park, P. Florence, J. Straub, R. Newcombe and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR),* IEEE, 2019, pp. 165-174.

[25] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Commun. ACM*, vol. 65, no. 1, pp. 99-106, 2021.

[26] T. Wang, X. Zhu, J. Pang and D. Lin, "Fcos3D: Fully convolutional one-stage monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. on Computer Vision. (ICCV)*. pp. 913-922, 2021.

[27] Y. Li, B. Huang, Z. Chen, Y. Cui, F. Liang, M. Shen, ... & J. Shao, "Fast-BEV: A Fast and Strong Bird's-Eye View Perception Baseline," *arXiv preprint arXiv:2301.12511,* 2023.

[28] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, ... & O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*. pp. 11621-11631, 2020.

[29] A. Geiger, P. Lenz, C. Stiller and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231-1237, 2013.