2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Semantically Guided Feature Matching for Visual SLAM

Oguzhan Ilter[1], Iro Armeni[1], Marc Pollefeys[1,2], Daniel Barath[1]

***Abstract*— We introduce a new algorithm that utilizes semantic information to enhance feature matching in visual SLAM pipelines. The proposed method constructs a high-dimensional semantic descriptor for each detected ORB feature. When integrated with traditional visual ones, these descriptors aid in establishing accurate tentative point correspondences between consecutive frames. Additionally, our semantic descriptors enrich 3D map points, enhancing loop closure detection by providing deeper insights into the underlying map regions. Experiments on public large-scale datasets demonstrate that our technique surpasses the accuracy of established methods. Importantly, given its detector-agnostic nature, our algorithm also amplifies the efficacy of modern keypoint detectors, such as SuperPoint. The implementation of our algorithm can be found on Github[3].**

I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is a cornerstone in both robotics and computer vision, enabling an autonomous agent to construct a 3D map of an unknown environment while concurrently determining its position within that space. Since its introduction in 1986 [1], the problem has gained significant attention due to its broad applicability in autonomous navigation, mobile robotics, and augmented reality, with a number of improvements proposed throughout the years, e.g., [2]–[5]. Recently, a shift in focus has been observed towards interpreting the environment beyond mere 3D geometry, placing emphasis on extracting and leveraging the semantics surrounding the agent [6].

SLAM systems incorporating semantics provide insights into object attributes, including size, class, and mobility (static or dynamic), while crafting semantically augmented maps. Studies focusing on object-based estimation [7]–[10] generate maps by estimating the size and shape of the objects along with their 3D position. Nonetheless, such methods often lag in localization accuracy compared to feature-centric strategies, predominantly because of the inherent noise and uncertainties in object discernment and position approximation. Hybrid methods, which combine feature and object reasoning [11], [12], aim to accurately localize the agent via standard feature matching while simultaneously updating and maintaining the object representations in the map. This comes at the cost of increased computational complexity.

Apart from building maps that contain rich information about both semantics and 3D geometry, such high-level knowledge proves invaluable in refining individual segments of a SLAM system. Components benefiting from this include those focusing on camera pose estimation and bundle adjustment [13], [14], loop closure detection [15], map scale recovery [16], and coping with dynamic environments [17]–[19]. At its core, semantic comprehension offers a robust and complementary signal to 3D geometry, aiding in addressing its inherent limitations. Even though feature matching is a critical component in state-of-the-art SLAM systems and despite the rich literature in this field, there is a limited number of papers [20] that improve matching using semantics.

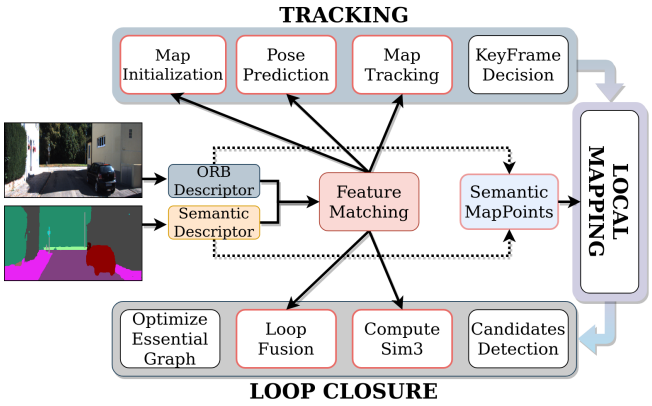

Fig. 1. The ORB-SLAM2 [3] pipeline and the components with which our proposed semantic feature matching interacts. Semantic feature descriptors are extracted together with the standard ORB ones. They are then jointly used both in the matching procedure and for generating 3D map points.

In this paper, we focus on exploiting semantics in feature matching by leveraging both the object class and the geometric attributes of the features. The main contribution is a new matching algorithm that uses semantics to find better features in a SLAM pipeline. The method constructs a semantic descriptor from the neighborhood of each feature and combines it with the visual descriptor traditionally used for feature matching. We demonstrate on publicly available datasets that our method leads to significant improvements in terms of accuracy while running in real-time.

II. RELATED WORK

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in robotics and computer vision that aims to estimate the position and orientation of a moving camera in 3D space while simultaneously constructing a map of the surrounding environment. To achieve this, SLAM algorithms typically rely on feature-based or direct methods for data association between frames. Direct methods directly optimize a photo-metric error between consecutive frames via, e.g., optical flow. While such methods often are faster than their feature-based alternatives, they tend to be sensitive to lighting changes and fast movements, where the data association

[1]Computer Vision and Geometry Group, ETH Zurich, 8092 Zurich, Switzerland, [2]Microsoft Mixed Reality and AI Zurich lab
[3]https://github.com/oguzhanilter/Semantically-Guided-Feature-Matching-for-Visual-SLAM

fails [6]. Feature-based methods relate consecutive frames by detecting and matching image feature points. The established tentative correspondences are used in a robust estimation procedure to estimate the camera motion. Although such approaches are marginally slower than direct ones, they are more robust in challenging environments.

ORB-SLAM [2] is one of the most successful feature-based SLAM pipelines. Its first version was proposed by Mur-Artal et al. in 2015, and it largely improves upon its predecessor by introducing a loop closure mechanism using the visual bag-of-words algorithm [21] to reduce the cumulative drift by optimizing over the entire trajectory. ORB-SLAM2 [3] and ORB-SLAM3 [4] further improved the performance and added the capability of leveraging additional input sensor modalities, like depth. The ORB-SLAM algorithms use ORB [22] features to efficiently extract distinctive keypoints and descriptors from image frames. The found features are matched and tracked across frames to estimate the camera motion and construct the 3D map. While ORB-SLAM achieves remarkable performance in many applications, its strictly geometric map lacks the granular details about the surrounding environment required for many complex robotic and augmented reality tasks.

In recent years, researchers have explored several approaches to improve the robustness and accuracy of SLAM pipelines by incorporating certain levels of semantic understanding about the observed environment. Object-based SLAM exploits semantics to describe a map only with the objects within it and their functional attributes, like size or color. CubeSLAM [7] approximates objects with 3D cuboids and estimates the camera position with the estimated 3D object positions. QuadricSLAM [8] and SO-SLAM [9] use quadrics to estimate the 3D object position and shape accurately. However, in practice, the camera localization performance of object-based methods cannot compete with feature-based methods due to the noise and uncertainties in object detection and position estimation. To address this limitation, other studies have focused on combining feature and object-based approaches to create a unified map of 3D points and objects with semantic attributes. Hybrid models like EAO-SLAM [11] and DSP-SLAM [12] leverage both feature and object-based methods to increase the accuracy of camera localization and object representation.

Another branch of work focuses on using semantics in SLAM to improve the accuracy of specific SLAM modules. VSO [13] introduced a novel semantic cost based on semantically segmented images, incorporated into pose optimization to enhance camera localization and reduce drift. Wang et al. [14] leverage semantic segmentation to detect static points in the map, such as buildings, and match these static points with a prior map of the environment to increase localization accuracy. Other studies [17], [18] use semantic labels to reason about an object being static or dynamic. Feature points stemming from dynamic objects are removed from the estimation as they are inconsistent with the assumption of having static surroundings, i.e., one of the most common assumptions that unlocks the ability to estimate the camera motion. This signifies that semantics can complement geometry, mitigating its inherent limitations and bolstering the robustness of SLAM systems.

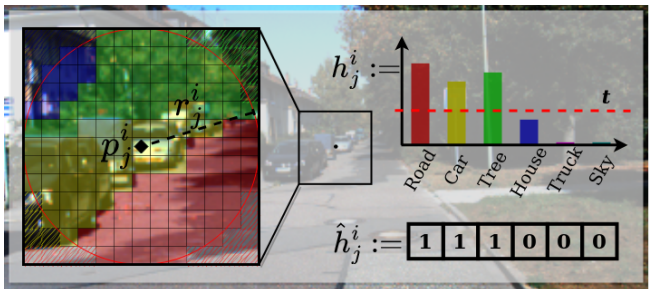

Fig. 2. The semantic descriptor h_j^i for point p_j^i in the ith frame is constructed as the histogram of semantic labels falling inside a circle centered on p_j^i with radius r_j^i. The radius comes from the feature size. The final descriptor $\widehat{h}_j^i$ is calculated by binarizing h_j^i using threshold t.

In this paper, we focus on leveraging semantic information in the feature-matching procedure of feature-based SLAM algorithms. Despite being a crucial component, there are very few studies that use semantics to improve feature matching. Arandjelović et al. [23] filters out the matches of feature points that have different semantic class coverage. Kobyshev et al. [20] use semantics to reduce the search space for matching by keeping only those pairs of features that are assigned the same semantic label in both frames. However, relying on potentially erroneous semantic detectors can lead to irrecoverable matching failures. [24] uses confidence scores from a semantic segmentation network to select feature points on the image by eliminating points. In this paper, we propose a novel technique, crafting high-dimensional semantic descriptors for feature points, which, in tandem with visual descriptors, aids in feature matching.

III. SEMANTIC FEATURE MATCHING

In this section, we focus on incorporating pixel-wise semantic segmentation into the feature matching procedure. In order to be directly usable in state-of-the-art SLAM pipelines, we consider ORB features [22] in the rest of the paper, if not stated otherwise, and will speak in the context of using ORB-SLAM2. However, our proposed method is straightforwardly applicable to any kind of feature (e.g., SIFT [25] or SuperPoint [26]) used in other applications.

A. Semantic Feature Descriptor

Let us assume that we are given a set of feature points $\mathcal{P}_i = \{(p_j^i, s_j^i, \alpha_j^i) \mid p_j^i \in \mathbb{R}^2, s_j^i \in \mathbb{R}, \alpha_j^i \in [0, 2\pi)\}_{j=1}^{n_i}$ found in image $\mathcal{I}_i$, where $i \in [1, k]$, $k \in \mathbb{N}^+$ is the number of frames in the sequence, $n_i \in \mathbb{N}$ is the number of features, p_j is the 2D point coordinates, s_j is the size of the feature, α_j is its orientation obtained by the detector. We are given function $G : \mathcal{P} \times \mathcal{I} \to \mathbb{R}^{d_g}$, returning the $d_g \in \mathbb{N}^+$ dimensional visual descriptor of a particular point in the image. In the context of ORB features, $G_{\text{ORB}} : \mathcal{P} \times \mathcal{I} \to \{0, 1\}^{256}$ returns a binary vector. Moreover, we are given a function $F : \mathcal{P} \to \mathcal{S}$ that takes a point as input and returns its semantic class.

Let us assume that function F maps to a finite set of semantic labels and, thus, $0 \leq |\mathcal{S}| < +\infty$. In order to be able to measure how much a particular semantic label $l \in \mathcal{S}$ describes the vicinity of a point $p = [x, y]^{\mathrm{T}}$, let us define function $L : \mathcal{P} \times \mathcal{S} \rightarrow \mathbb{R}$ as follows:

$$L(p, l, r) = \frac{1}{r^2\pi} \sum_{i=-w/2}^{w/2} \sum_{j=-h/2}^{h/2} ([\![F(I(x+i, y+j)) = l]\!] [\![\sqrt{i^2 + j^2} \leq r]\!]), \quad (1)$$

where $w \in \mathbb{R}$ and $h \in \mathbb{R}$ are the size of the window centered on point p, r is the radius of the circle, and $[\![.]\!]$ is the Iverson-bracket which is 1 if the condition inside holds, and 0 otherwise (see Figure 2). In brief, function L calculates the ratio of pixels inside a circle centered on p that are assigned semantic label l. In order to be adaptive to the feature size, r is returned by the detector as the feature size s. We set $w = 2r$ and $h = 2r$ so the window contains the circle. In such a way, this semantic importance measure inherits the invariance properties of the feature detector and becomes invariant to scale changes in the images.

Building on the assumption that we are given a finite set of semantic labels, we can define a semantic descriptor for the jth point p_j^i in the ith image as follows:

$$h_j^i = [L(p_j^i, l_1),\ L(p_j^i, l_2),\ \cdots,\ L(p_j^i, l_{|\mathcal{S}|})] \in [0, 1]^{|\mathcal{S}|}. \quad (2)$$

Note that the bins in the histogram are sorted according to their corresponding semantic class ID which provides a global ordering inside all such semantic descriptors extracted from any image at any point. This allows for matching the descriptors across images. Since the number of pixels inside a local window changes according to the scale s_j^i of the feature point p_j^i, the normalization of the histogram with the circle radius is essential in Eq. 1 to enable comparing descriptors that stem from differently sized local areas.

Additionally, in order to increase the robustness of the semantic feature vectors against noise in the semantic segmentation, we define threshold $t \in [0, 1]$ to convert the normalized semantic vector to a binary vector as follows:

$$\widehat{h}_j^i = [\widehat{L}(p_j^i, l_1),\ \widehat{L}(p_j^i, l_2),\ \cdots,\ \widehat{L}(p_j^i, l_{|\mathcal{S}|})] \in \{0, 1\}^{|\mathcal{S}|}, \quad (3)$$

where

$$\widehat{L}(p, l) = \begin{cases} 1 & \text{if} \quad L(p, l) \geq t, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

Such binarization allows for minimizing the effect of small semantic regions appearing inside the local window by chance due to the noise in the semantic detector.

It is important to emphasize that the method used to construct semantic descriptors is not viewpoint invariant. Similarly, traditional visual descriptors also lack explicit viewpoint invariance. Therefore, this does not impose additional limitations on the matching procedure.

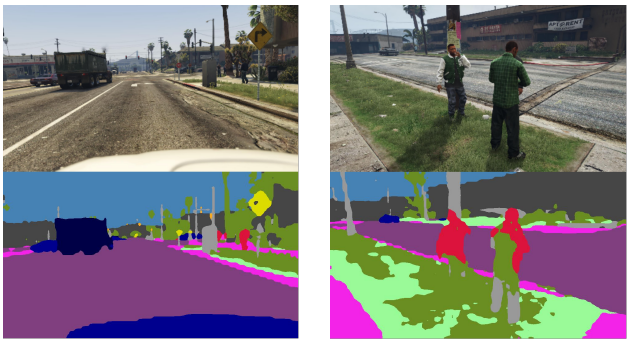

Fig. 3. Images from the P4B dataset [27] with semantic segmentations [28].

B. Updating Semantic Feature Descriptors

In ORB-SLAM2, the feature descriptors are also used for finding correspondences between the reconstructed 3D map and the features in the current input image. In the monocular setting of ORB-SLAM2, a map point is created when the same feature point is observed in two different keyframes. In addition, map points are updated every time when they are detected in another keyframe, or when they are merged with other map points after a loop closure. This update procedure includes updating their associated descriptors. Therefore, both the visual and semantic descriptors have to be updated.

In this paper, we adopt the update mechanism from the original ORB-SLAM method. Suppose that we are given a point $m \in \mathbb{R}^3$ in the 3D map observed from n keyframes I_1, I_2, $\cdots$, I_n. Thus, we are given a set of semantic descriptors $\mathcal{V}_m = \{\widehat{h}_m^1, \widehat{h}_m^2, \cdots, \widehat{h}_m^n\}$ for point m. We choose the descriptor to represent point m in the map to be the one with the minimal L_1 distance from all other descriptors in the set, i.e., the median. Therefore,

$$\widehat{h}_m^* = \arg_{h \in \mathcal{V}} \min \sum_{h' \in \mathcal{V}} |h - h'| = \text{median}(V_m). \quad (5)$$

Updating the descriptor of map points ensures that a point can be matched to subsequent frames despite the viewpoint change. This allows methods to track 3D map points over a long period. Thus, it decreases the drift in the tracking.

C. Combining Semantic and Visual Descriptors

As we use semantic descriptor $\widehat{h}_j^i$ to support the visual ones g_j^i in feature matching, we need a procedure for using them jointly. Descriptors $\widehat{h}_j^i$ and g_j^i measure completely different properties of the neighborhood of feature p_j. Vector $\widehat{h}_j^i$ is calculated from the semantic labels as a binary vector. Vector g_j^i is calculated from pixel intensities of a local window centered on point p_j. Simply concatenating $\widehat{h}_j^i$ and g_j^i leads to matching in a highly anisotropic space which would require a complicated distance metric that accounts for the differently scaled axes in the high-dimensional descriptor space. Thus, it is sensitive to the choice of the distance function and might be time consuming in practice.

To overcome this issue, we independently calculate the descriptor distances for each descriptor type and take their

TABLE I

ROOT MEAN SQUARE TRANSLATION (IN METERS) AND ROTATION ERRORS (IN DEGREES) OF ORB-SLAM2 [3], VSO [13], AND OUR METHOD WITH AND WITHOUT VSO ON THE KITTI DATASET [29]. THE LAST TWO ROWS SHOW THE ERRORS AVERAGED OVER ALL SEQUENCES WITH AND WITHOUT WEIGHTING BY THE FRAME NUMBER. THE BEST RESULTS ARE IN BOLD, AND THE SECOND BESTS ARE UNDERLINED.

	Translation RMSE (meters)					Rotation RMSE (degrees)				
Path	ORB-SLAM2	+ [20]	+ Ours	+ VSO	+ VSO + Ours	ORB-SLAM2	+ [20]	+ Ours	+ VSO	+ VSO + Ours
00	6.92	9.60	<u>6.20</u>	6.74	**5.98**	0.23	0.33	<u>0.20</u>	0.21	**0.16**
02	25.47	24.66	<u>22.13</u>	22.14	**20.23**	0.13	<u>0.11</u>	0.13	**0.09**	0.12
04	1.56	1.44	**1.50**	**1.50**	1.84	**0.04**	**0.04**	**0.04**	**0.04**	**0.04**
05	<u>4.65</u>	6.02	5.52	**4.63**	4.98	<u>0.08</u>	0.12	0.09	**0.06**	0.09
06	<u>15.44</u>	16.13	16.50	16.08	**15.35**	**0.09**	1.24	1.31	1.32	<u>1.27</u>
08	50.38	49.58	46.03	<u>41.02</u>	**37.69**	0.06	**0.05**	**0.05**	**0.05**	**0.05**
09	33.78	26.03	<u>11.91</u>	22.15	**9.80**	0.08	<u>0.06</u>	<u>0.06</u>	<u>0.06</u>	**0.05**
10	6.68	7.32	**6.24**	<u>6.41</u>	6.48	0.07	**0.06**	**0.06**	**0.06**	**0.06**
w. AVG	22.11	22.05	18.76	<u>18.55</u>	**16.31**	**0.12**	0.20	0.18	<u>0.17</u>	<u>0.17</u>
AVG	18.09	17.60	<u>14.50</u>	15.08	**12.79**	**0.10**	0.25	<u>0.24</u>	<u>0.24</u>	0.28

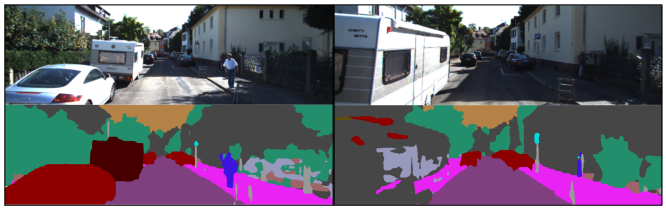

Fig. 4. Images of the KITTI dataset [29] with predicted semantic segmentations from the fine-tuned PIDNet [28].

linear combination. Suppose that we are given a keypoint in the first and one in the second frames with descriptors $(\widehat{h}^1, g^1)$ and $(\widehat{h}^2, g^2)$, respectively. Given that both descriptors are binary vectors, the Hamming distance is an applicable and particularly fast distance metric. Therefore, the distance between each pair of descriptors is calculated as $d_{\text{visual}} = d_{\text{Hamming}}(g^1, g^2)$ and $d_{\text{semantic}} = d_{\text{Hamming}}(\widehat{h}^1, \widehat{h}^2)$.

Before combining the descriptors, we have to account for their different scaling factors that stem from calculating Hamming distance in spaces with different dimensionality. Vectors g^1 and g^2 are in a 256-dimensional space. Vectors $\widehat{h}^1$ and $\widehat{h}^2$ are $|\mathcal{S}|$-dimensional. Thus, $d_{\text{visual}} \in [0, 256]$ and $d_{\text{semantic}} \in [0, |\mathcal{S}|]$. To normalize the distances, we simply multiply d_{semantic} by $256/|\mathcal{S}|$. The distance combining both the semantic and visual descriptors is as follows:

$$d_{\text{combined}} = (1 - w) d_{\text{visual}} + w \frac{256}{|\mathcal{S}|} d_{\text{semantic}}, \qquad (6)$$

where $w \in [0, 1]$ is the parameter of linear interpolation. It is a hyper-parameter balancing the weights of the terms.

In summary, the feature matching process calculates d_{combined} of potential correspondences and returns the matches minimizing both the visual and semantic distances.

D. Affected Modules in ORB-SLAM2

ORB-SLAM2 consists of three main processes: Tracking, Local Mapping, and Loop Closure [3]. Our semantic descriptors are created in the pre-processing part of the Tracking step and used in any feature matching process along with the conventional ORB descriptor. Starting from the map initialization, it is a necessary step to calculate the geometric relationship between frame-frame, frame-map, and map-map. As map points are a part of the matching process, during a new map point creation, the semantic descriptor is also defined as it is described in Section III-B. In the Fig. 1, the ORB-SLAM2 pipeline is visualized with the modules directly enhanced by our method.

IV. EXPERIMENTS

We compare our proposed semantically guided feature matching with the original ORB-SLAM2 [3], [20], and with VSO [13], i.e., a method that uses the semantic segmentation in the bundle adjustment of ORB-SLAM2. We also combine our method with VSO to demonstrate that it provides a complementary module to other methods leveraging semantic understanding. On each tested image sequence, all algorithms were run 10 times to account for the random nature of relative pose estimation inside ORB-SLAM2.

SLAM algorithms are expected to run in real-time, so we apply a real-time semantic segmentation network, PIDNet [28], instead of focusing on selecting the best-performing one. To minimize the domain gap between the training set on which PIDNet was trained, we fine-tuned it on the KITTI semantic segmentation benchmark training images [29].

To compare the trajectories to the ground truth, we use the EVO Python package [30] that provides efficient tools for map alignment and camera pose error calculations. Since the compared SLAM algorithms have a common scale ambiguity (i.e., we cannot recover the global metric scale from images), the final reconstructions are to be scaled, translated, and rotated to best overlap with the ground truth. We use this tool to determine the global alignment before calculating the errors. In almost all experiments, we calculate the root mean square absolute translation error (in meters) and root mean square relative rotation error (in degrees) as it gives a better understanding of the drift. Relative rotation error is calculated between two consecutive keyframes.

The experiments were performed on the ETH Zürich Euler cluster. On average, the proposed semantic descriptor creation on 2000 ORB features runs for 0.0253 secs on a single core of an AMD EPYC 7742 CPU.

A. Experiments on KITTI

We compare ORB-SLAM2, VSO, ORB-SLAM2 combined with our semantic feature matching or with [20], and VSO with our algorithm on 10 sequences of the KITTI dataset [29][1]. Instead of fine-tuning the parameters on each sequence independently (as done in [13]), we selected scenes 03 and 07 for tuning the hyper-parameters, which we omit from the evaluation. We chose these sequences since they are among the shortest ones with (03) and without (07) loops. We performed a grid search for all methods to find hyper-parameters minimizing pose error. The used parameters for the proposed method are $w = 0.1$ and $t = 0.1$. Example frames from the dataset and semantic segmentations predicted by the fine-tuned PIDNet are in Fig. 4.

Results are reported in Table I. The last two rows show the mean errors averaged over all paths without weighting and weighted by the number of frames in a sequence. The left part shows the translation errors in meters. Compared to ORB-SLAM2, our method significantly improves the translation error by 2-3 meters on average. The method proposed in [20] for using semantics in feature matching only marginally improves. Combining our semantic feature matching with VSO results in further error reduction, achieving the lowest translation errors. Our method with VSO reduces the average ORB-SLAM2 translation error to its approx. 70%.

The right part of the table reports the rotation errors. While all methods are marginally less accurate on this metric than ORB-SLAM2, it is essential to note that the error is in degrees. Thus, all methods perform particularly accurately with *negligible* rotation error.

Trajectories reconstructed by the compared methods and the ground truth (GT) are in Fig. 5. We show sequences 02, 08, and 09, i.e., the ones with the largest errors in Table I. On 02, our method with VSO follows the GT trajectory the best. On 09, it is the only one that almost perfectly coincides with the GT. Scene 08 shows the scale drift problem of the monocular setting. The left and right-hand sides of the trajectories have different scales. Thus, the global alignment to GT does not work as well as on other scenes.

B. Experiments on P4B

This section compares the algorithms on the Play for Benchmark dataset [27]. We conducted tests on 11 day sequences. The number of frames per sequence ranges from 330 to 2412 without any loops in the GT trajectories. We use the parameters tuned on KITTI without further tuning.

The RMSE errors of the compared methods averaged over the sequences of the P4B dataset are reported in Table III. We do not show the results of [20] as it makes ORB-SLAM2 fail completely on 5 of the 11 sequences. The proposed method significantly improves upon standard ORB-SLAM2. Interestingly, VSO [13] does not work on this dataset as it almost doubles the translation errors. While the VSO paper [13] reported improvements on the P4B dataset, we attribute that to VSO being tuned independently on each sequence. We use a single set of parameters tuned on KITTI for all tested methods. The bottom part shows that the proposed method halves the rotation errors of ORB-SLAM2. While the lowest errors are achieved by VSO + Ours, its translation errors are particularly high. Consequently, the best-performing method on the P4B dataset is ORB-SLAM2 combined with the proposed semantic descriptors.

Example trajectories are in Fig. 5. An interesting behavior can be observed in sequences 003 and 067 which are the same path but 067 contains a large number of moving vehicles, creating a rather challenging problem. Our method is more robust to this problem than the other ones.

[1]Please note that, as per prior work [3], [13], sequence 01 from KITTI is skipped since it a single high-way that ORB-SLAM2 fails to track.

TABLE II

THE AVERAGE RMSE TRANSLATION (IN METERS) AND ROTATION (IN DEGREES) ERRORS OF ORB-SLAM2 ON THE KITTI DATASET, WHEN ENHANCED WITH OUR PROPOSED SEMANTIC DESCRIPTORS. FOR EXAMPLE, 'ORB-SLAM2 + LOOP FUSION' MEANS THAT THE LOOP CLOSURE DETECTION USES THE SEMANTIC DESCRIPTORS. FOR W. AVG, SEQUENCE ERRORS ARE WEIGHTED BY THE FRAME COUNT. THE BEST RESULTS ARE IN BOLD, AND THE SECOND BESTS ARE UNDERLINED.

	Translation RMSE (m)		Rotation RMSE (°)	
	AVG	w. AVG	AVG	w. AVG
ORB-SLAM2	18.09	22.11	**0.10**	**0.12**
+ Map Init.	16.42	20.28	0.19	0.16
+ Pose Pred.	15.29	19.45	0.29	0.23
+ Map Track	15.11	19.11	0.29	0.19
+ Comp. Sim3	15.36	19.34	0.24	0.17
+ Loop Fusion	15.23	19.01	0.18	0.14
+ All (ours)	**14.50**	**18.76**	0.24	0.18

TABLE III

THE RMSE TRANSLATION (IN METERS) AND ROTATION ERRORS (IN DEGREES) OF ORB-SLAM2 [3], VSO [13], AND OUR PROPOSED METHOD WITH AND WITHOUT VSO AVERAGED OVER THE SEQUENCES OF THE P4B DATASET [27]. FOR W. AVG, SEQUENCE ERRORS ARE WEIGHTED BY THE FRAME COUNT. THE BEST RESULTS ARE IN BOLD, AND THE SECOND BESTS ARE UNDERLINED.

	Translation RMSE (meters)			
	ORB-SLAM2	+ Ours	+ VSO	+ VSO + Ours
w. AVG	<u>33.91</u>	**22.15**	54.12	48.97
AVG	<u>24.94</u>	**15.43**	38.87	40.01
	Rotation RMSE (degrees)			
	ORB-SLAM2	+ Ours	+ VSO	+ VSO + Ours
w. AVG	8.90	<u>4.33</u>	4.71	**3.66**
AVG	7.62	<u>3.65</u>	3.89	**3.29**

C. Ablation Study

We conducted an ablation study to discern the impact of the proposed semantic descriptors on specific components of ORB-SLAM2. By integrating these descriptors into each module (as illustrated in Fig.1) one at a time, we gauged the accuracy on the KITTI dataset. The corresponding RMSE translation and rotation errors are reported in Table II. Notably, incorporating the semantic descriptors into *any* component invariably reduces the error. The most pronounced improvement is observed when all components are augmented

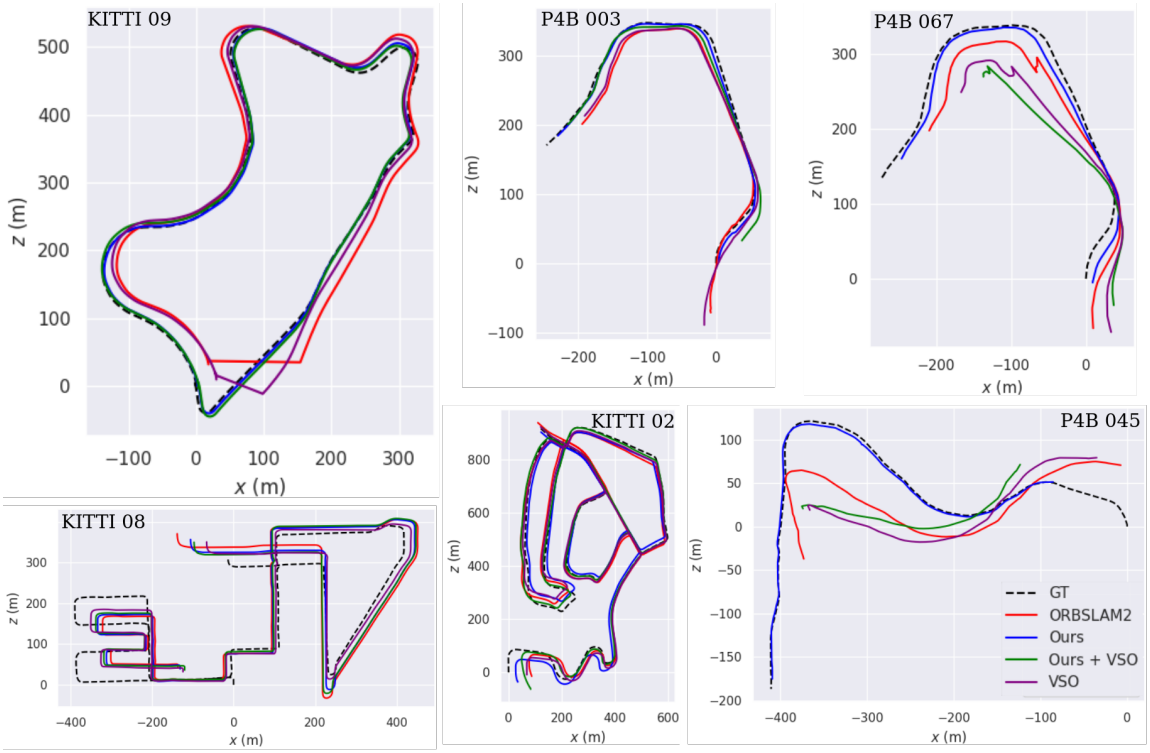

Fig. 5. Reconstructed trajectories of ORB-SLAM2 [3] (red curve), VSO [13] (purple), our method with (VSO + Ours; blue) and without VSO (Ours; green) and the ground truth ones on the KITTI [29] and P4B [27] datasets. For KITTI, we visualize the sequences with the largest errors in Table I. For P4B, we chose sequences 003 and 067 since they are the same path but 067 contains heavy traffic with many moving vehicles.

with the semantics. Again, while the rotation errors increase marginally, all methods have similarly accurate rotations.

D. Experiment with SuperPoint Features

While our earlier discussions concentrated on the efficacy of the proposed method combined with ORB features, it is crucial to note that our approach is detector-agnostic. To underscore the proposed performance enhancement by incorporating semantics, we paired it with SuperPoint features [26]. For merging descriptor distances as in Eq. 6, the maximum visual descriptor distance was adjusted to 1.414 (representing the maximum distance for SuperPoint descriptors) instead of 256. In this analysis, our focus was on gauging the accuracy of the relative pose estimated by RANSAC from the found point correspondences. Consequently, the translation error is also reported in degrees, given that the translation scale cannot be deduced from relative poses.

Table IV demonstrates that SuperPoint and the proposed semantic descriptors significantly improve the translation accuracy while slightly improving the estimated rotations. The success rate has also improved substantially.

TABLE IV

ROOT MEAN SQUARE TRANSLATION (DEGREE) AND ROTATION ERRORS (DEGREE) OF RELATIVE POSE ESTIMATION FROM SUPERPOINT FEATURES [26] WITH AND WITHOUT PROPOSED METHOD. 750 IMAGE PAIRS WITH INTERVAL OF 5, 10 AND 15 BETWEEN FRAMES.

	Trans. RMSE (°)		Rot. RMSE (°)		Success #	
Path	SP	+ Ours	SP	+ Ours	SP	+ Ours
00	38.9	**34.4**	15.3	**15.2**	437	**508**
02	41.0	**37.8**	17.6	**14.0**	284	**351**
03	45.6	**37.9**	13.3	**11.5**	606	**652**
04	43.1	**39.9**	**16.4**	17.4	505	**565**
05	35.9	**32.7**	**14.3**	15.6	502	**563**
06	51.0	**46.1**	25.6	**25.5**	446	**521**
07	42.6	**42.2**	**12.9**	16.6	494	**561**
08	37.1	**31.2**	**12.1**	14.7	396	**464**
09	47.6	**39.9**	24.1	**18.6**	313	**392**
10	41.4	**36.8**	17.3	**15.5**	396	**464**
AVG	42.3	**37.9**	16.9	**16.5**	438	**504**

V. CONCLUSIONS

In this paper, we propose a novel algorithm that leverages semantic information to improve feature-matching in visual SLAM pipelines. By constructing high-dimensional semantic descriptors for detected ORB features and using them jointly with the visual descriptors in establishing tentative point correspondences between consecutive frames, our approach achieves significantly improved accuracy compared to standard approaches. On both the KITTI and P4B datasets, the average translation error is decreased by at least 6 meters compared to the baseline. The rotation accuracy is also significantly better on P4B than that of state-of-the-art methods.

VI. ACKNOWLEDGEMENT

This project was supported by the ETH RobotX and the Hasler Stiftung research grants via the ETH Zurich Foundation.

REFERENCES

[1] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *International Journal of Robotics Research*, vol. 5, no. 4, pp. 56–68, 1986.

[2] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, oct 2015.

[3] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, oct 2017.

[4] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, dec 2021.

[5] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *ICRA*, 2020.

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

[7] S. Yang and S. Scherer, "Cubeslam: Monocular 3-d object slam," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 925–938, 2019.

[8] L. Nicholson, M. Milford, and N. Sunderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 08 2018.

[9] Z. Liao, Y. Hu, J. Zhang, X. Qi, X. Zhang, and W. Wang, "So-slam: Semantic object slam with scale proportional and symmetrical texture constraints," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4008–4015, 2022.

[10] F. Wang, C. Zhang, W. Zhang, C. Fang, Y. Xia, Y. Liu, and H. Dong, "Object-based reliable visual navigation for mobile robot," *Sensors*, vol. 22, no. 6, 2022.

[11] Y. Wu, Y. Zhang, D. Zhu, Y. Feng, S. Coleman, and D. Kerr, "EAO-SLAM: Monocular semi-dense object SLAM based on ensemble data association," in *IROS*. IEEE, oct 2020.

[12] J. Wang, M. Rünz, and L. Agapito, "Dsp-slam: Object oriented slam with deep shape priors," in *2021 International Conference on 3D Vision (3DV)*, 2021, pp. 1362–1371.

[13] N. Lianos, J. L. Schönberger, M. Pollefeys, and T. Sattler, "VSO: Visual Semantic Odometry," in *ECCV*, 2018.

[14] C. Ye, Y. Wang, Z. Lu, I. Gilitschenski, M. Parsley, and S. J. Julier, "Exploiting semantic and public prior information in monoslam," in *IROS*, 2020, pp. 4936–4941.

[15] J. Oh and G. Eoh, "Variational bayesian approach to condition-invariant feature extraction for visual place recognition," *Applied Sciences*, vol. 11, no. 19, 2021.

[16] J. Lee, M. Back, S. S. Hwang, and I. Y. Chun, "Improved real-time monocular slam using semantic segmentation on selective frames," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 3, pp. 2800–2813, 2023.

[17] N. Brasch, A. Bozic, J. Lallemand, and F. Tombari, "Semantic monocular slam for highly dynamic environments," in *IROS*, 2018, pp. 393–400.

[18] L. Cui and C. Ma, "Sof-slam: A semantic visual slam for dynamic environments," *IEEE Access*, vol. 7, pp. 166 528–166 539, 2019.

[19] P. Li, G. Zhang, J. Zhou, R. Yao, X. Zhang, and J. Zhou, "Study on slam algorithm based on object detection in dynamic scene," in *ICAMechS*, 2019, pp. 363–367.

[20] N. Kobyshev, H. Riemenschneider, and L. V. Gool, "Matching features correctly through semantic understanding," in *2014 2nd International Conference on 3D Vision*, vol. 1, 2014, pp. 472–479.

[21] D. Galvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[22] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to sift or surf," in *ICCV*. Ieee, 2011, pp. 2564–2571.

[23] R. Arandjelović and A. Zisserman, "Visual vocabulary with a semantic twist," 11 2014, pp. 178–195.

[24] P. Ganti and S. L. Waslander, "Visual SLAM with network uncertainty informed feature selection," *CoRR*, vol. abs/1811.11946, 2018.

[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, pp. 91–110, 2004.

[26] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *CVPRW*, 2018, pp. 224–236.

[27] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," in *ICCV*, 2017, pp. 2232–2241.

[28] J. Xu, Z. Xiong, and S. P. Bhattacharyya, "Pidnet: A real-time semantic segmentation network inspired by pid controllers," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2023, pp. 19 529–19 539.

[29] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, 2012.

[30] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.