Proceedings of 19th International Conference on Ubiquitous Robots (UR 2022), Jeju, Korea, July 2022. http://2022.ubiquitousrobots.org

A Semantic Enhancement of Unified Geometric Representations for Improving Indoor Visual SLAM

Ioannis Asmanis[1], Panagiotis Mermigkas[1], Georgia Chalvatzaki[2], Jan Peters[2] and Petros Maragos[1]

***Abstract*— Over the last two decades, visual SLAM research has taken a turn towards using geometric structures more complex than points for describing indoor environments. At the same time, semantic information is becoming increasingly available to robotic applications, improving robots' perceptive capabilities. In this work, we introduce a method for uniting these two approaches. Namely, we propose a novel mechanism for propagating semantics directly into the optimization level of an RGB-D SLAM framework. This framework internally uses unified geometric representations to jointly describe points, lines and planes. We also validate our approach with experiments on various datasets, both synthetic and real-world, with comparisons against representative systems from the literature.**

I. Introduction

Simultaneous Localization And Mapping (SLAM) is an overarching problem in robotics. SLAM refers to the problem of obtaining a representation of the robot's environment and simultaneously estimating its trajectory. On-board sensors are used to perceive the environment and then SLAM algorithms process their measurements to infer information about the robot's surroundings, as well as its own path. Both the sensor technology and algorithm complexity are constantly evolving, thus providing robotic systems with increased autonomy in a variety of settings. Over the last two decades, numerous solutions have been proposed for all variants of the SLAM problem [1], [2].

An important sector of SLAM research concentrates on indoor spaces. Enabling safe robot interaction with humans indoors relies on robust autonomy, with perception being one of its cornerstones. Works on perception have considered both localization and semantic understanding [3], therefore it would be justified to combine SLAM with semantics towards this direction. Of course, indoor spaces present similar difficulties to most SLAM use-cases, including low-texture areas (empty hallways), repetitive structures (near-identical rooms) or dynamic entities (humans) [4]. Such open challenges have motivated this work to explore possible solutions and improvements for the indoor setting.

There are two popular trends in indoor visual SLAM (vSLAM) research that have recently gained considerable momentum: (i) employing complex geometric structures [5]–[7] and (ii) incorporating semantic segmentation methods, as in the works of [8]–[11]. Accordingly, this work[1] considers the problem of uniting elements from each of the aforementioned research directions to improve indoor vSLAM.

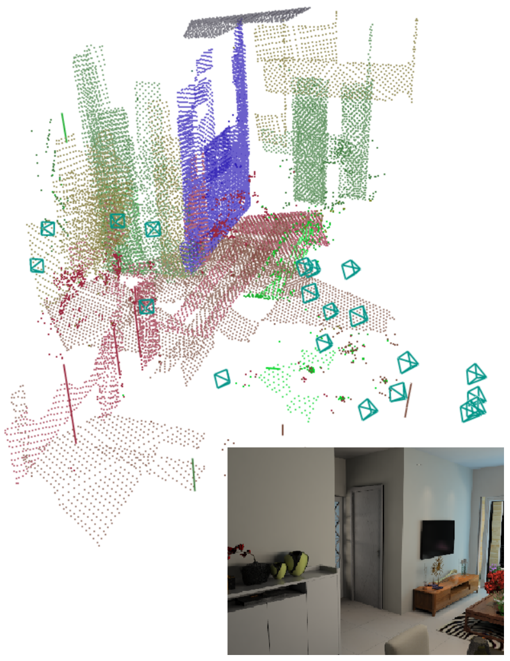


Fig. 1: Point-cloud segmentation, obtained during the execution of our RGB-D vSLAM system. Semantic classes are displayed in different colors. A sample frame from the sequence the detections were made in is shown at the bottom-right.

Our contribution is a novel methodology for enhancing unified geometric representations (entities jointly expressing points, lines, and planes) with semantic information. We propose a voting scheme to achieve this at the feature level, and observe improvements in trajectory estimation, especially when rich semantics are present. We introduce the average amount of semantic labels as the measure of semantic richness in our data sequences. This method differs significantly from other semantic vSLAM approaches, where the semantic and geometric pipelines are either computationally restricting [12], held completely separate [9], or use semantics solely for mapping [13]. Our method directly affects the optimization process and outputs both a sparse map and pose estimates. Furthermore, it makes use of heterogeneous geometric primitives, instead of just points, in order to take advantage of the regular geometric structures in indoor environments. Finally, we evaluate a complete system implementation on three indoor datasets, comparing against

[1]Electrical and Computer Engineering, National Technical University of Athens, 15780 Zografou, Athens, Greece `asmanis.ioannis@gmail.com`, `p.mermigkas@gmail.com`, `maragos@cs.ntua.gr`

[2]Intelligent Autonomous Systems, Technische Universität Darmstadt, Germany (`georgia@robot-learning.de`, `mail@jan-peters.net`)

[1]Code available at: `https://gitlab.com/_JackFrost_/sem-sashago`

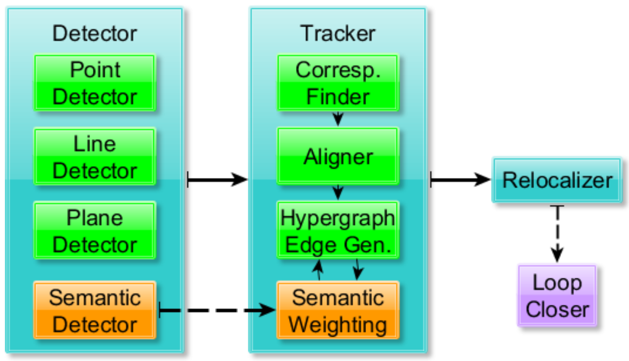


Fig. 2: Sem-SASHAGO architecture overview: The three geometric detector modules detect the necessary geometric structures. The semantic detector finds semantic bounding boxes for objects in the input images. The data is then propagated to the tracker modules that construct the back-end pose graph, while also applying our semantic weighting process, using the voting scheme of Sect. III-D. Finally, the relocalizer module checks for possible loop closures and, if necessary, calls the loop closer to add the constraints.

state-of-the-art methods.

II. Related Work

Visual SLAM was preceded by localization methods that also used optical sensors, known as Visual Odometry (VO). VO methods are still applied in some settings with strict performance constraints or demands for algorithmic efficiency. They avoid constructing a full map of the environment, focusing instead on localizing the agent and executing only some local, temporal optimization steps, commonly referred to as Bundle Adjustment (BA) [14], [15]. However, contemporary SLAM solutions have become capable of running in real-time on modern hardware [16], [17]. Some approaches integrate a VO solution running at a low level and only use vSLAM components periodically [18].

Initial research in SLAM was constrained by processing power, which naturally led to using LiDAR sensors instead of cameras, since their data streams are sparser. LiDARs also capture geometry directly and their measurements are immune to lighting changes [19]. As a result, researchers made extensive use of sensor measurement filtering for pose estimation [20]–[23]. As CPUs became more potent, using camera feeds directly proved to be a realistic possibility. Not only are cameras generally cheaper than LiDAR sensors, they can also capture an entire frustum of optical information, as opposed to range information from just one 2D planar slice of the world. Recently, the deep learning revolution has brought GPUs to the forefront of computation [24], also enabling large scale visual data processing; in fact, GPUs have even been used directly for vSLAM [25].

VSLAM systems can be divided in multiple categories according to their algorithmic design. Direct dense or semi-dense vSLAM methods, such as [26], [27], produce maps with high levels of detail, while indirect sparse methods [17] are generally faster to compute, and appropriate when the resulting map can be simpler. The processing is usually split between a front-end and a back-end [28]. The front-end, which involves data association and graph updates, creates and maintains the graph. It also calculates the initial egomotion estimate, for example by obtaining frame-to-frame pose estimations using a solver for the Perspective-n-Point problem (PnP) [29]. Finally, it handles outlier rejection, e.g. using RANSAC [30], [31], to formulate more accurate initial motion estimations. The back-end typically relies on a generic non-linear least squares optimizer [32], [33], and is responsible for optimizing the pose graph.

VSLAM solutions may also be classified based on their intended application. For outdoor applications, the scale of the environment limits the use of close-range sensors, such as depth cameras. Instead, alternative sensors, such as stereo cameras, LiDAR, IMU or GPS, are frequently used to provide sources of perception [34]–[36]. Indoor SLAM systems typically operate on a much smaller scale than their outdoor counterparts and with different objects in their vicinity. Outlier removal methods are commonly used in both settings to filter out dynamic entities (e.g. moving humans) or spurious feature matches, with recent works also employing machine learning [8], [9]. In addition, indoor environments may contain regular geometric structures, such as lines and planes, which can be detected and tracked [6], [37], [38]. This idea led to the development of unified representations of heterogeneous geometric primitives, which are of great interest to our work [7], [39].

Nowadays, semantic information is being used in vSLAM with increasing frequency. Semantic detectors based on neural networks are fast and accurate [40]–[42]. [11] used stereo image pairs to generate 3D object-bounding volumes, along with viewing angle estimates. Other approaches focus on optimizing data associations. Specifically, [12] use Expectation Maximization (EM) to track the most probable semantic class for an object detection. They define a semantic error term, which is added to their GTSAM factor graph [43]. [44] is a variant of this method, which assumes multimodal sum-of-Gaussian variables and uses belief propagation to take all possible class assignments under weighted consideration.

Our approach leverages the advantages of merging semantic detections with regular indoor geometric structures. Contrary to other works in the literature, we aim to inject semantic information directly into the pose graph, by storing both the geometric and semantic history of tracked features in the generated sparse map. In semantically barren environments, our solution will rely on its geometric detections; empty indoor spaces typically provide salient geometric structures, such as clear walls and corners that can be reliably detected, as noted in some of our experimental sequences.

III. Methodology

A. *System overview*

The architecture of our system (Sem-SASHAGO), is presented in Fig. 2. The detection modules are responsible for detecting geometric and semantic features in the input RGB-D frames. We fuse semantic information with geometric primitives (Sect. III-B), using a semantic voting mechanism (Sect. III-D). The tracker performs two operations: it aligns the new frame with the observed world map through feature association and it generates the hypergraph edges that will update the back-end pose graph. Finally, the system executes relocalization; it searches for previous scenes with high similarity and generates loop closure constraints, if necessary. The loop closer is only executed when such constraints are imposed, optimizing the graph accordingly. The system outputs a camera pose estimate for each input frame, along with a global geometric-semantic map.

B. Unified geometric representations

We begin by presenting the definition of the unified geometric primitives used throughout this work. These are points, lines and planes, expressed using degenerate quadrics as:

$$(\boldsymbol{x}-\boldsymbol{p})^T\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{p})=\mathbf{0} \tag{1}$$

where $\boldsymbol{x}$ is the coordinates of the points in the primitive, $\boldsymbol{p}$ is the centroid and $\boldsymbol{A}$ is a symmetric, positive-definite matrix. A **matchable** [7] is mathematically defined as:

$$\boldsymbol{M} := \langle \boldsymbol{p_M},\ \boldsymbol{R_M},\ \boldsymbol{\Lambda_M}\rangle \tag{2}$$

where $\boldsymbol{p_M}$ is the centroid of the matchable, and matrices $\boldsymbol{R_M} \in SO(3)$ and $\boldsymbol{\Lambda_M}$ correspond to the orientation and shape of the matchable respectively, derived from matrix $\boldsymbol{A}$ in (1) after refactoring in the form $\boldsymbol{A}=\boldsymbol{R\Lambda R}^T$, where $\boldsymbol{R} := [\boldsymbol{r}_x,\ \boldsymbol{r}_y,\ \boldsymbol{r}_z]$. The direction of the matchable $\boldsymbol{M}$ is defined as $\boldsymbol{d_M} := \boldsymbol{r}_x$. The shape matrix $\boldsymbol{\Lambda_M}$ is 3×3-diagonal and takes one of the following forms, depending on $\boldsymbol{M}$'s geometry:

$$\boldsymbol{\Lambda_M} = diag(\boldsymbol{\lambda}), \qquad \boldsymbol{\lambda} = \begin{cases}(1,\ 1,\ 1) & \text{for points}\\ (0,\ 1,\ 1) & \text{for lines}\\ (1,\ 0,\ 0) & \text{for planes}\end{cases} \tag{3}$$

The geometric primitives considered here - points, lines and planes - are represented as zero-radius spheres, zero-radius cylinders and coincident planes, respectively. Points are detected using FAST detectors [45]. Lines are detected using the Line Segment Detection (LSD) algorithm [46]. Finally, planes are detected from depth images by clustering points of the point cloud [39]. Only lines and planes have a direction (for planes it is their normal). Note that the detectors need to be parameterized for good results.

C. Graph setup

In our implementation, the pose graph is implemented with the g2o library [32], appropriately extended to represent matchables in the back-end. The optimization can be formulated as a non-linear least squares problem, which resides in a manifold, i.e. a space that is only locally linear [28]. Since our system uses matchables as its fundamental geometric units, some adjustments to the standard SLAM error framework are necessary. Therefore, matchable errors contain not only a spatial distance component $\boldsymbol{e}_p$, but also a directional misalignment component $\boldsymbol{e}_d$ and an orthogonality component e_o:

$$\boldsymbol{e}(\boldsymbol{M}_a,\boldsymbol{M}_b) = \begin{bmatrix}\boldsymbol{e}_p\\ \boldsymbol{e}_d\\ e_o\end{bmatrix} = \begin{bmatrix}\boldsymbol{R}_b^T(\boldsymbol{p}_a-\boldsymbol{p}_b)\\ \boldsymbol{d}_a-\boldsymbol{d}_b\\ \boldsymbol{d}_a^T\boldsymbol{d}_b\end{bmatrix} \tag{4}$$

Since the error terms defined above are not relevant for all possible pairs of matchables, an activation matrix $\boldsymbol{C}$ is used to discriminate between different cases:

$$\boldsymbol{C} = \begin{bmatrix}\boldsymbol{C}_p & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1}\\ \mathbf{0}_{3\times3} & \boldsymbol{C}_d & \mathbf{0}_{3\times1}\\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \boldsymbol{C}_o\end{bmatrix} \tag{5}$$

The block-diagonal elements $\boldsymbol{C}_p$, $\boldsymbol{C}_d$ and $\boldsymbol{C}_o$ in (5) serve this purpose, by selectively (de)activating error components, depending on the type of observed matchable pair. Consider one pose-graph edge linking pose $\boldsymbol{x}_i$ to matchable $\boldsymbol{M}_j$ via measurement $\boldsymbol{z}_{ij}$ with raw input sensor information matrix $\bar{\boldsymbol{\Omega}}_{ij}$. Then, the modified information matrix is $\boldsymbol{\Omega}_{ij} = \boldsymbol{C}\cdot\bar{\boldsymbol{\Omega}}_{ij}\cdot\boldsymbol{C}^T$ and the corresponding error contribution is $e_{ij}(\boldsymbol{x}_i,\boldsymbol{M}_j) = \left\|\boldsymbol{e}(\boldsymbol{x}_i\boldsymbol{M}_j,\boldsymbol{z}_{ij})\right\|_{\boldsymbol{\Omega}_{ij}}$. The overall optimization problem is formulated as:

$$\boldsymbol{X}^* = \arg\min_{\boldsymbol{X}}\sum_{ij}\boldsymbol{e}_{ij}^T(\boldsymbol{X})\boldsymbol{\Omega}_{ij}\boldsymbol{e}_{ij}(\boldsymbol{X}) \tag{6}$$

where $\boldsymbol{X}$ represents the pose vector. The system is optimized iteratively in the back-end using g2o [32]. The process takes small perturbations around the current vector of state estimates, selecting the direction that steps towards minimizing the error using a Jacobian of the error metric, until convergence [28]. The Gauss-Newton and Levenberg-Marquardt algorithms are commonly used to achieve this, with the latter allowing for dampening the iteration step, for more stability around the optimum.

D. Semantic voting

We wish to tightly couple semantics with the geometric optimization process. To achieve this, we augment geometric detections with semantic labels, and use the semantic data to better guide the optimizer. Since each frame contains hundreds of potential detections, with dozens of possible associated classes, the approach of [12] is rendered computationally infeasible, due to the exponential complexity of its EM (note that [12] considers object-level detections of only two semantic classes). Instead, our method approximates a Maximum Likelihood (ML) logic, in a way that adds negligible computational overhead.

Our goal is to augment each matchable with the most appropriate semantic label. For this reason, we detect semantic entities in semantic bounding boxes (SBBs), which are computed either using Faster R-CNN [41], or from ground truth (GT) files where available. The kth semantic measurement in frame i, $\boldsymbol{s}^{i,k}$, following the notation of [12], is defined as:

$$\boldsymbol{s}^{i,k} = \langle s_c^{i,k},\ s_s^{i,k},\ \boldsymbol{s}_b^{i,k}\rangle \tag{7}$$

where $s_c^{i,k} \in \mathcal{C}$ is an integer representing the semantic class of the observation, $s_s^{i,k} \in [0.0, 1.0]$ is the confidence score of detection, computed by the detector, and $\boldsymbol{s}_b^{i,k} = [(x_{tl}, y_{tl}),\ (x_{br}, y_{br})] \in \big([0, W-1] \times [0, H-1]\big)^2$ is the top-left-bottom-right-corner expression of the pixel region of an SBB in a $W \times H$-sized image.

1) Mapping matchables to unique labels: Now, let $(x^{i,j}, y^{i,j}) = \boldsymbol{r}^{i,j} := \pi_{\boldsymbol{x}^i}(\boldsymbol{p}^{i,j})$ be the reprojection of the jth 3D matchable position at frame i, $\boldsymbol{p}^{i,j}$, onto the image plane for camera pose $\boldsymbol{x}^i$. We denote as $S_b^{i,j}$ the set of all SBBs detected in frame i, such that the reprojection of the centroid of the jth matchable falls within the SBB, that is:

$$S_b^{i,j} := \bigcup_k \left\{ \boldsymbol{s}_b^{i,k} \,\middle|\, x^{i,j} \in [x_{tl}^{i,k}, x_{br}^{i,k}] \wedge y^{i,j} \in [y_{tl}^{i,k}, y_{br}^{i,k}] \right\} \tag{8}$$

Out of all semantic measurements that contain the j-th matchable in frame i, we select the one with minimal pixel area. This choice was based on the fact that it most likely corresponds to a nested foreground object, thus resolving detection ambiguities. Next, we formulate our semantic label association objective for measurement $\boldsymbol{s}^{i,j}$, $\hat{\boldsymbol{s}}^{i,j}$:

$$\hat{\boldsymbol{s}}_b^{i,j} = \operatorname*{arg\,min}_{\boldsymbol{s}_b \in S_b^{i,j}} \text{Area}(\boldsymbol{s}_b) \tag{9}$$

At this point, each matchable $\boldsymbol{M}$ is associated with an index, indicating the semantic detection it has been assigned to. Therefore, we can augment (2) to accommodate this index:

$$\boldsymbol{M}^s := \langle \boldsymbol{p}_{\boldsymbol{M}},\ \boldsymbol{R}_{\boldsymbol{M}},\ \boldsymbol{\Lambda}_{\boldsymbol{M}},\ \boldsymbol{s}_{\boldsymbol{M}} \rangle \tag{10}$$

where $\boldsymbol{M}^s$ is the augmented matchable and $\boldsymbol{s}_{\boldsymbol{M}} = \hat{\boldsymbol{s}}^{i,j}$ corresponds to the semantic measurement found through (9), and that best describes the jth matchable $\boldsymbol{M}$.

2) Dominant landmark classes: A landmark is a matchable that has been observed several times during a trajectory. Each re-observation comes with a potentially different class assignment to the matchable. Since geometric landmarks are abundant in general (typically hundreds per frame), we require a direct way of computing the most probable class of a landmark, given our observations thus far. Therefore, we estimate class $\hat{l}_c^{F,j}$ of the j-th landmark at frame F to be the one that maximizes the sum of its observation scores $\hat{s}_s^{i,j}$ over all frames thus far:

$$\hat{l}_c^{F,j} = \operatorname*{arg\,max}_{v \in \mathcal{C}_+} \sum_{i=0}^{F} \mathbb{1}[\hat{s}_c^{i,j} = v] \cdot \hat{s}_s^{i,j} \tag{11}$$

where $\mathcal{C}_+$ is the set of semantic classes along with a special no-detection class (for frames where the landmark was not observed, with a default confidence score of 1.0). Essentially, (11) uses a weighted voting scheme to determine which is the most probable class for each landmark, called the *dominant class* of the landmark, based on previous semantic detector output. This approach clearly adds a negligible computational overhead, even when dozens of semantic classes are used, and hundreds of landmarks present in each frame.

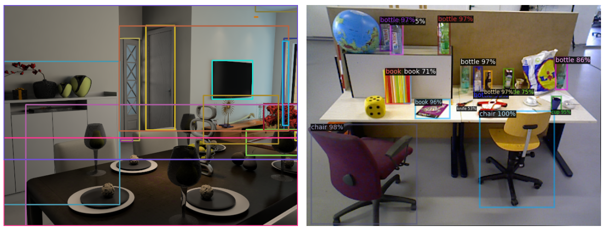

(a) Ground truth (b) Faster R-CNN

Fig. 3: Semantic detection in different datasets (InteriorNet and TUM RGB-D). Note the difference in semantic detection density.

3) Optimization: We update the optimization framework of [7] by using a weight penalization for mismatches between expected landmark class $\hat{l}_c^{i,j}$ and observed matchable detection class $\hat{s}_c^{i,j}$. This weight $w_s^{i,j}$ is computed to diminish the importance of observations between classes that are not highly correlated in the confusion matrix $\mathcal{D}_C$:

$$w_s^{i,j} = f\left(\mathcal{D}_C(\hat{s}_c^{i,j}, \hat{l}_c^{i,j})\right) \simeq \begin{cases} 1 & ,\ \hat{s}_c^{i,j} = \hat{l}_c^{i,j} \\ W_p & ,\ \hat{s}_c^{i,j} \neq \hat{l}_c^{i,j} \end{cases} \tag{12}$$

where $W_p < 1$ is a tunable penalization weight constant. We edit the pose graph edge by scaling the error terms proportionally to the weights in (12), as in $\boldsymbol{\Omega}_{ij}^{'} = w_s^{i,j} \cdot \boldsymbol{\Omega}_{ij}$, eventually solving a semantically-aware instance of the optimization problem (6).

IV. Experimental Evaluation

A. Datasets & systems

For evaluating the performance of the proposed method, we experiment with three datasets: (i) ICL-NUIM [47], a photorealistic synthetic indoor dataset, (ii) InteriorNet [48] (recently used in ICRA's 2019 workshop on SLAM benchmarking), a visually and semantically richer photorealistic synthetic indoor dataset, which comes equipped with COCO-format [49] semantic GT labels and (iii) TUM RGB-D [50], a standard real-world indoor dataset, commonly used in the vSLAM literature.

The first system we evaluate is the base implementation of SASHAGO [7]. Our system, Sem-SASHAGO, is run both with and without semantics (Ours and Ours (w/o sem.) respectively), in order to demonstrate the effect of using only geometry. In this regard, the system has been improved by overhauling parts of the KD-Tree backend, as well as by adding dampening logic to the final iteration steps of the iterative point-cloud alignment process (in the *Aligner* block of Fig. 2). These enhancements, along with other minor adjustments, comprise the difference between Sem-SASHAGO without semantics and the base SASHAGO system, but since they are technical in nature they are not analyzed in detail here. We also evaluate MaskFusion [8], which is a state-of-the-art semantic dense vSLAM system. For completeness, we reference the results obtained with ORB-SLAM2 [17], a graph-based, sparse vSLAM method, even though it only uses point-features for geometry and does not support semantics.

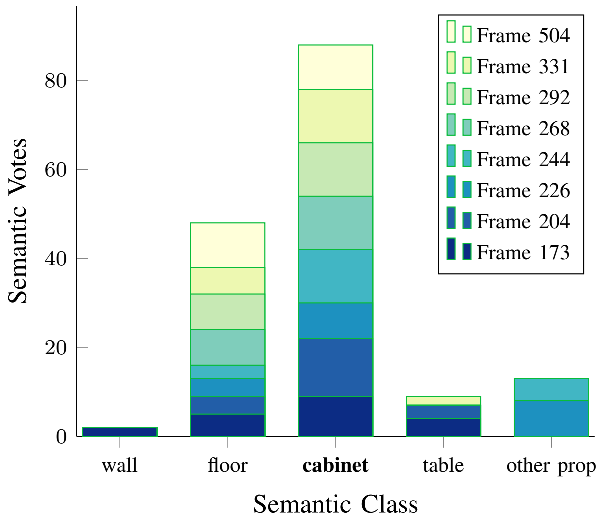


Fig. 4: Semantic votes over time for a landmark in the InteriorNet dataset. Note that several semantic mismatches persist, despite the ideal ground-truth detections. New observations belonging to non-dominant classes are penalized by our framework.

B. Preprocessing

Geometry: For [48], we found that depth images needed to be recomputed, as the images initially contained the distance to the focal point rather than the distance to the image plane. For [50]'s Freiburg 1 sequences, we denoise depth data using a median filter with a kernel size of $k = 5$.
Semantics: The method for computing weights based on the confusion matrix (12) depends on the semantic detector. In datasets such as [48], the GT semantic information is available, so (12) can be simplified to use a fixed penalization weight. When using Faster R-CNN to extract semantic labels for [50], we found that confusion matrix values were not reported for the detector, so it would be impossible to dynamically adjust our weights for misclassified classes, as suggested by (12). To address this issue, we considered a constant W_p, aiming to equally penalize all mismatches.

C. Semantic Information Analysis

To aid in our analysis of results, we introduce a direct metric of semantic richness, *SemInfo*, expressed as the median and average of semantic detections per frame (where applicable). As shown in the two Tables of the next subsection (Tables II and III), the InteriorNet sequences are significantly semantically richer as compared to TUM RGB-D sequences. Fig. 3 shows a characteristic example of this difference in semantic detections between these two datasets.

To emphasize the advantages of our semantic approach, Fig. 4 shows the histogram of semantic votes that were assigned to a landmark, according to the methodology described in Subsection III-D, from InteriorNet's `5o11_full` sequence (also shown in Fig. 3a), comprising 1000 frames. This specific landmark was chosen amongst the most observed ones, in order to highlight the progression of votes over a longer segment. The observations/votes are grouped in different color batches of 20 (the first 20 observations occur up to Frame 173, the next 20 up to Frame 204, etc.).

Despite the fact that InteriorNet's semantics given as ground-truth, it is observed that some votes are nonetheless misplaced in non-dominant classes. This discrepancy can be attributed to the inherent limitations of vSLAM feature association, that may attach a feature to an erroneous landmark. Additionally, the fact that our semantic label associator of Eq. (9) could assign the wrong class (e.g. in the case of overlapping bounding boxes where the bigger bounding box is the correct one, while the vote is won by the smaller), can also contribute to this phenomenon.

However, the negative effect of these two factors is alleviated by the semantic voting scheme, followed by the modification of pose graph edge weights. The edges that correspond to non-dominant classes (e.g. wall, table, other prop) receive significantly lower weights, and, because they are expected to often correspond to erroneous associations, they are correctly neglected.

D. Evaluation method

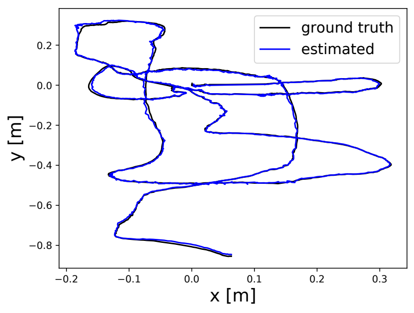
In order to systematically quantify vSLAM precision, we focus on localization accuracy. Our evaluation process uses [50]'s toolkit, which provides numerous error metrics; we use the *Absolute Trajectory Error (ATE)* metric, specifically the *Root Mean Square (RMS)*. For ATE, the GT and estimated pose sequences are first optimally aligned to eliminate reference frame ambiguity and subsequently the error metrics are uniquely determined [51].

E. Results

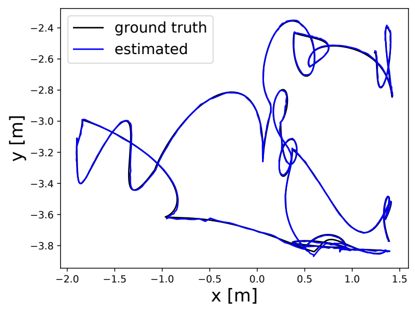
In this section, we present the ATE for all of our experimental sequences and compare with representative baseline methods. For each dataset-system combination, we report the results for the best parameter configuration we could find (see Fig. 5 for sample visualizations of camera trajectories projected on a 2D plane for readability).

1) ICL-NUIM: First, we evaluate our system on a purely geometric dataset, without the use of semantics, which are not available as GT for this dataset and could not be reliably detected due to the dataset's artificiality. Therefore, its main purpose is to highlight the algorithmic improvements made in SASHAGO's [7] base implementation. It also provides positive indications for the applicability of unified geometric representations for indoor data, since our method achieves low errors in sequences that include extended views of large planar surfaces (ceilings, walls etc.). Table I reports the results of our experiments.

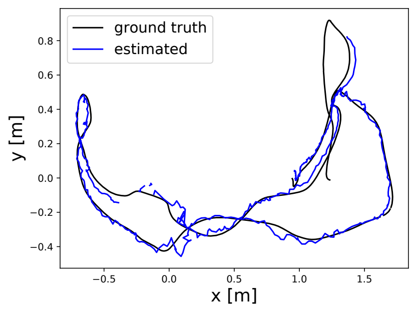
2) InteriorNet: We have selected some representative and challenging sequences from [48] for our experiments (Table II). In sequences `3o11_open` and `4o11_200-600`, sparse semantics set a threshold for the improvement that could be offered by our semantic processing. However, the results remain within millimeters of the best ATE performance thanks to its geometric fallback; it still outperforms SASHAGO and MaskFusion. Note also how ORB-SLAM2's performance slightly deteriorates in these two feature-deprived



(a) ICL-NUIM trajectory.



(b) InteriorNet trajectory.



(c) TUM RGB-D trajectory.

Fig. 5: Estimated trajectories and GT for different datasets. In (c), note the effect of noise on the system's performance in a real-world dataset, as opposed to the synthetic photorealistic datasets (a) and (b).

TABLE I: ICL-NUIM results (RMS ATE [m])

Sequence	Ours (w/o sem.)	SASHAGO	ORB-SLAM2
tr0	**0.0131**	0.0156	0.0212
lr0	**0.0238**	0.0519	0.0036
lrkt1	**0.0046**	0.0091	0.0392
lrkt2	**0.0067**	0.0148	0.0281

TABLE II: InteriorNet results (RMS ATE [m])

Sequence	SemInfo (Med/Avg)	Ours	Ours (w/o sem.)	SASHAGO	MaskFusion	ORB-SLAM2
133_open	8/8.2	**0.0075**	0.0091	0.0203	0.0190	0.0057
2r11_250-600	17/15.8	**0.0161**	0.0267	0.0772	0.0205	0.0147
3o11_open	9/8.9	0.0065	**0.0060**	0.0179	0.0179	0.0189
4o11_200-600	6/5.1	0.0343	**0.0319**	0.0565	0.4595	0.0447
5o11_full	12/12.7	**0.0099**	0.0111	0.0409	0.0110	[track lost]
5o33_200-675	16/15.7	**0.0292**	0.0539	0.0502	0.0575	0.0268
6o11_800-1000	12/11.4	**0.0128**	0.0135	0.0195	0.0164	0.0045

TABLE III: TUM RGB-D results (RMS ATE [m])

Sequence	SemInfo (Med/Avg)	Ours	Ours (w/o sem.)	SASHAGO	MaskFusion	ORB-SLAM2
fr1_desk	4/4.0	**0.0406**	0.0825	0.1188	0.9116	0.0203
fr2_desk_500	6/5.9	0.0183	**0.0178**	0.1350	0.0581	0.0053
fr2_desk_1000	6/5.9	0.0193	**0.0181**	0.1634	0.0581	0.0067
fr2_desk_full	4/4.6	0.0714	**0.0653**	0.2065	0.4559	0.0085
fr3_loh_200	11/10.2	**0.0161**	0.0267	0.0772	0.0337	0.0178
fr3_loh_1000	6/5.9	0.0521	0.0508	0.0867	**0.0272**	0.0087
fr3_loh_full	7/6.6	**0.1527**	0.1576	0.4145	0.2092	0.0097
fr3_sitting_static	10/10.4	0.0087	**0.0081**	0.0092	0.0117	0.0086
fr3_sitting_xyz	9/8.5	0.0441	**0.0436**	0.0506	0.0530	0.0092

sequences, which makes a strong case for the robustness of compound geometric-semantic solutions in indoor environments. By contrast, rich semantics lead to considerable improvements for our method; sequences `2r11_250-600` and `5o33_200-675` have much higher semantic content, and therefore the most noticeable improvements in their performance.

3) TUM RGB-D: This real-world dataset is challenging for all vSLAM systems. The noisy measurements negatively impact environment perception, which renders the estimation of unified geometries especially difficult. Our method outperforms SASHAGO and MaskFusion in most cases (Table III). The three subsequences in `fr2_desk` and `fr3_loh` demonstrate how error accumulates for our system in noisy environments as more frames are processed (also seen in MaskFusion and SASHAGO). We also observe that MaskFusion is impaired by the noisy measurements, as seen by its errors in several sequences. ORB-SLAM2 performs very well in these settings overall, primarily due to its resilience to measurement noise. Compared to the previous dataset [48], scenes generally include fewer objects, and semantic detections might be flawed because they are computed using Faster R-CNN and not GT. Therefore, semantic content is scarcer and the expected accuracy gain from semantics for Sem-SASHAGO is limited; it needs to rely more on geometry. Finally, we considered `fr3_sitting_`* sequences, which feature slightly dynamic entities (moving, seated humans). We found that our system still improved over its baseline SASHAGO, and managed to perform very well in the static shot of the scene.

F. Result summary

Our approach is initially validated in plainer environments (Table I), where we achieve the lowest errors in sequences without many optical features or semantics, but with prominent geometric regularities. Furthermore, we demonstrated that higher average semantic content correlates with a significant reduction in errors (Table II), which is supported by our system's ability to mitigate the problems illustrated in subsection IV-C. Finally, we considered real-world data (Table III), where we showed stable performance, consistently improving over our base system and achieving higher accuracy than MaskFusion in most cases. Overall, our experiments show that the combination of semantics with unified geometric representations yields improvements, compared to the purely geometric approach.

V. Conclusions

In this work, we have developed a methodology for incorporating semantic information in an indoor RGB-D vSLAM framework. Motivated by recent advances in unified geometric representations, we merged semantics with matchables to assemble a complete system. We showcased our solution's capabilities and its advantages, by evaluating on a variety of datasets and comparing to other relevant vSLAM methods. From the results we concluded both that the system accuracy was improved in the presence of richer semantic content and that, in its absence, the geometric component was still effective. In the future, we shall attempt to expand this work by handling dynamic entities in the robot's field of view, as well as tackling the persistent impact of noise coming from real-world measurements.

References

[1] J. Aulinas, Y. Petillot, J. Salvi, and X. Llado, "The SLAM problem: a survey," *Frontiers in Artificial Intelligence and Applications*, vol. 184, Jan. 2008.

[2] C. Cadena, *et al.*, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," *IEEE Trans. Robotics*, vol. 32, no. 6, Dec. 2016.

[3] N. Sünderhauf, *et al.*, "Meaningful maps with object-oriented semantic mapping," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2017.

[4] Y. Li, *et al.*, "Structure-SLAM: Low-Drift Monocular SLAM in Indoor Environments," *IEEE Robotics and Automation Letters*, vol. 5, 2020.

[5] L. Nicholson, M. Milford, and N. Sünderhauf, "QuadricSLAM: Dual Quadrics From Object Detections as Landmarks in Object-Oriented SLAM," *IEEE Robotics and Automation Letters*, vol. 4, 2019.

[6] M. Hosseinzadeh, *et al.*, "Structure Aware SLAM using Quadrics and Planes," *ACCV 14*, 2018.

[7] I. Aloise, B. D. Corte, F. Nardi, and G. Grisetti, "Systematic Handling of Heterogeneous Geometric Primitives in Graph-SLAM Optimization," in *Proc. Robotics: Science and Systems (RSS XV)*, Freiburg im Breisgau, June 2019, pp. 84–92.

[8] M. Rünz and L. Agapito, "MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects," *IEEE International Symposium on Mixed and Augmented Reality*, 2018.

[9] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2017.

[10] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping," *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2020.

[11] P. Li, T. Qin, and S. Shen, "Stereo Vision-based Semantic 3D Object and Ego-motion Tracking for Autonomous Driving," in *Proc. European Conf. Computer Vision (ECCV)*, Sept. 2018.

[12] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic Data Association for Semantic SLAM," *IEEE Trans. Robot. Automat.*, May 2017.

[13] R. Dubé, *et al.*, "SegMap: Segment-based mapping and localization using data-driven descriptors," *The International Journal of Robotics Research*, vol. 39, June 2020.

[14] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," *IEEE Robot. Automat. Mag.*, vol. 18, Dec. 2011.

[15] F. Fraundorfer and D. Scaramuzza, "Visual Odometry: Part II - Matching, Robustness, and Applications," *IEEE Robot. Automat. Mag.*, vol. 19, June 2012.

[16] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," in *IEEE Int. Symp. Safety, Security, and Rescue Robotics*, 2011.

[17] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Trans. Robotics*, vol. 33, no. 5, Oct. 2017.

[18] D. Geromichalos, L. Petrou, M. Azkarate, and E. Tsardoulias, "Globally Adaptive SLAM for Autonomous Planetary Rovers using Map Matching," M. Eng. thesis, Aristotle University of Thessaloniki, Thessaloniki, Greece, Mar. 2018.

[19] K. Krinkin, *et al.*, "Evaluation of Modern Laser Based Indoor SLAM Algorithms," *Proc. XXth Conference of Open Innovations Association FRUCT*, vol. 426, May 2018.

[20] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[21] M. Ribeiro and I. Ribeiro, "Kalman and Extended Kalman Filters: Concept, Derivation and Properties," Apr. 2004.

[22] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proc. IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000.

[23] P. M. Djuric, *et al.*, "Particle filtering," *IEEE Signal Processing Magazine*, vol. 20, no. 5, 2003.

[24] D. Steinkraus, I. Buck, and P. Y. Simard, "Using GPUs for machine learning algorithms," in *Proc. 8th Int. Conf. Document Analysis and Recognition (ICDAR'05)*, vol. 2, 2005.

[25] T. Whelan, *et al.*, "ElasticFusion: Real-Time Dense SLAM and Light Source Estimation," *The International Journal of Robotics Research (IJRR)*, vol. 35, 2016.

[26] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," in *Proc. European Conf. Computer Vision (ECCV)*, Sept. 2014.

[27] J. Engel, J. Stückler, and D. Cremers, "Large-scale direct SLAM with stereo cameras," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2015.

[28] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A Tutorial on Graph-based SLAM," *IEEE Trans. Intelligent Transportation Systems*, vol. 2, 2010.

[29] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An Accurate O(n) Solution to the PnP Problem," *International Journal of Computer Vision*, vol. 81, 2008.

[30] R. Raguram, J.-M. Frahm, and M. Pollefeys, "A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus," *Lect. Note. Comput. Sci.*, vol. 5303, Oct. 2008.

[31] S. Choi, T. Kim, and W. Yu, "Performance evaluation of RANSAC family," *Proc. British Machine Vision Conf.*, vol. 24, Jan. 2009.

[32] R. Kümmerle, *et al.*, "g2o: A general framework for graph optimization," *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2011.

[33] S. Agarwal and K. Mierle, "Ceres solver," http://ceres-solver.org.

[34] S. Saftescu, *et al.*, "Kidnapped Radar: Topological Radar Localisation using Rotationally-Invariant Metric Learning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Paris, 2020.

[35] T. Y. Tang, D. De Martini, D. Barnes, and P. Newman, "RSL-Net: Localising in Satellite Images From a Radar on the Ground," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, 2020.

[36] D. Schleicher, *et al.*, "Real-Time Hierarchical Outdoor SLAM Based on Stereovision and GPS Fusion," *IEEE Trans. Intelligent Transportation Systems*, vol. 10, Sept. 2009.

[37] M. Dzitsiuk, *et al.*, "De-noising, stabilizing and completing 3D reconstructions on-the-go using plane priors," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2017.

[38] P. Kim, B. Coltin, and H. Jin Kim, "Linear RGB-D SLAM for planar environments," in *Proc. European Conf. Computer Vision (ECCV)*, Sept. 2018.

[39] F. Nardi, B. D. Corte, and G. Grisetti, "Unified Representation and Registration of Heterogeneous Sets of Geometric Primitives," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, 2019.

[40] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2015.

[41] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, *et al.*, Eds. Curran Associates, Inc., 2015.

[42] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, 2017.

[43] L. Carlone, *et al.*, "Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2014.

[44] K. Doherty, D. Fourie, and J. Leonard, "Multimodal Semantic SLAM with Probabilistic Data Association," *Proc. Int. Conf. Robotics and Automation (ICRA)*, 2019.

[45] M. Trajkovic and M. Hedley, "Fast corner detection," *Image and Vision Computing*, vol. 16, 1998.

[46] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: a Line Segment Detector," *Image Processing On Line*, vol. 2, 2012.

[47] A. Handa, T. Whelan, J. McDonald, and A. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," in *Proc. IEEE Intl. Conf. Robotics and Automation, ICRA*, China, May 2014.

[48] W. Li, *et al.*, "Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset," in *Proc. British Machine Vision Conference (BMVC)*, 2018.

[49] T.-Y. Lin, *et al.*, "Microsoft COCO: Common Objects in Context," in *Proc. European Conf. Computer Vision (ECCV)*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., 2014.

[50] J. Sturm, *et al.*, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. Int. Conf. Intelligent Robot Systems (IROS)*, 2012.

[51] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2018.