Fusing Semantics and Motion State Detection for Robust Visual SLAM

Gaurav Singh Meiqing Wu Siew-Kei Lam
School of Computer Science and Engineering, Nanyang Technological University, Singapore
gaurav012@e.ntu.edu.sg, meiqingwu@ntu.edu.sg and assklam@ntu.edu.sg

Abstract

Achieving robust pose tracking and mapping in highly dynamic environments is a major challenge faced by existing visual SLAM (vSLAM) systems. In this paper, we increase the robustness of existing vSLAM by accurately removing moving objects from the scene so that they will not contribute to pose estimation and mapping. Specifically, semantic information is fused with motion states of the scene via a probability framework to enable accurate and robust moving object extraction in order to retain the useful features for pose estimation and mapping. Our work highlights the importance of distinguishing between motion states of potential moving objects for vSLAM in highly dynamic environments. The proposed method can be integrated into existing vSLAM systems to increase their robustness in dynamic environments without incurring much computation cost. We provide extensive experimental results on three well-known datasets to show that the proposed technique outperforms existing vSLAM methods in indoor and outdoor environments, under various scenarios such as crowded scenes.

1. Introduction

Visual Simultaneous Localization and Mapping (vSLAM) is a key component of modern autonomous systems, augmented reality and visual positioning systems [3, 30]. vSLAM explores the static correspondences in images to simultaneously estimate the pose of the ego-object and map of the environment as a joint problem. Maximum consensus schemes (e.g. RANSAC [22, 9]) are generally adopted to remove the dynamic outliers from the static inliers, under the assumption that the majority of the scene contains static elements. In recent years, vSLAM has achieved good progress. For instance, ORB-SLAM2 [20], a state-of-the-art SLAM algorithm, is able to achieve $\sim 1\%$ (of trajectory length) average translation error on KITTI odometry datasets [11] and about 0.015 meter Root Mean Square Error (RMSE) on TUM-static datasets [25]. However, the accuracy of these vSLAMs [20] reduce significantly in dynamic environments, such as those found in TUM-dynamic [25] and Apollo datasets [14]. This inconsistency in performance is due to the presence of large amount of moving objects (e.g. car, bicycle, pedestrian, etc.) in the scenes, which violates the assumption required by maximum consensus schemes. These dynamic elements, which are prevalent in realistic environments, can cause failures in pose tracking and irreversible corruptions in the map.

To address this problem, existing works [17] attempt to remove features associated with potential moving objects so that they will not contribute to pose estimation and mapping. However, this direct feature removal step can result in losing useful information in scenes where, potential moving objects are motionless, for example parked vehicles or pedestrians waiting at a traffic intersection. Our experiments show that in scenes containing large number of parked vehicles (e.g. sequences 17 and 16 of Apollo datasets [14]), the direct removal approach leads to reduction in features for pose estimation that cause a significant drop in pose accuracy.

In this paper, we aim to increase the robustness of vSLAM in all scenarios, e.g. indoor, outdoor, and highly dynamic scenes, by introducing a preprocessing step to remove moving objects before the vSLAM pipeline. In particular, semantic information is fused with motion states of the scene objects in a probability framework to enable accurate and robust moving region extraction in order to retain the useful features for pose estimation and mapping.

Our work shows the importance of distinguishing between motion states of potential moving objects for vSLAM in highly dynamic environments. The proposed method consists of the following steps. First, a lightweight scene flow estimation and clustering method is proposed to extract the moving regions in the scene. Second, semantic segmentation method is utilized to extract the semantic knowledge in the scene. Finally, to tackle the uncertainty in the extracted knowledge both for the moving regions and semantic maps, a probability framework is proposed to fuse the motion states cue with the semantic cue to detect the moving regions in a robust way. By excluding the features corresponding to the final determined moving regions, remaining features contain less outliers for the following vSLAM pipeline processing. Extensive experiments are undertaken on TUM [25], KITTI [11], and Apollo [14] datasets to evaluate the effectiveness of the proposed technique. Our experiment results show that the proposed technique is able to enhance the robustness of existing vSLAMs in various scenarios from indoor to outdoor scenes, and from low dynamic to crowded scenes.

The rest of the paper is organized as follows. Section 2 discusses the existing work in vSLAM particularly those that tackle the problem of dynamic environment. In Section 3, we describe the proposed framework to extract semantic-aware motion states for accurate moving region extraction in vSLAM. Experiments and comparison with existing state-of-the-art vSLAMs are presented in Section 4. Finally, Section 5 concludes this paper.

2. Related Work

Modern vSLAMs use either stereo [11], monocular [4] or RGB-D [25] images to localize itself and map the environment. vSLAM estimates the pose by exploiting the geometric relationship between the static correspondences observed in the images. The existing vSLAMs and visual odometry (VO) can be classified into three categories: indirect [23, 20, 12], direct [21, 7] and hybrid methods [10]. Indirect (i.e. feature-based) methods use distinct image features, while direct (i.e. appearance-based) approaches utilize image intensity information. Hybrid methods leverage on both feature and appearance-based benefits [24].

The state-of-the-art ORB-SLAM2 [20] is one of the most versatile feature-based vSLAM algorithms that performs well for both indoor and outdoor scenarios. It achieves high pose estimation accuracy in static or slightly dynamic environments such as those found in KITTI odometry datasets [11] and static TUM RGB-D datasets [25]. However, the pose estimation accuracy of ORB-SLAM2 degrades when a significant part of the scene is occupied by moving objects (e.g. vehicles) [31]. Similar behaviour is observed in other vSLAM [23] and VO [12] systems as well. This is contributed by the fact that the maximum consensus outlier removal schemes (like RANSAC [12, 20] can remove outlier features corresponding to moving objects only when the proportion of its features in the scene is small compared to those associated with the static objects. When significant number of features corresponding to moving object/s are present, they constitute to a motion field that impairs the effectiveness of the maximum consensus approach [17, 31, 33]

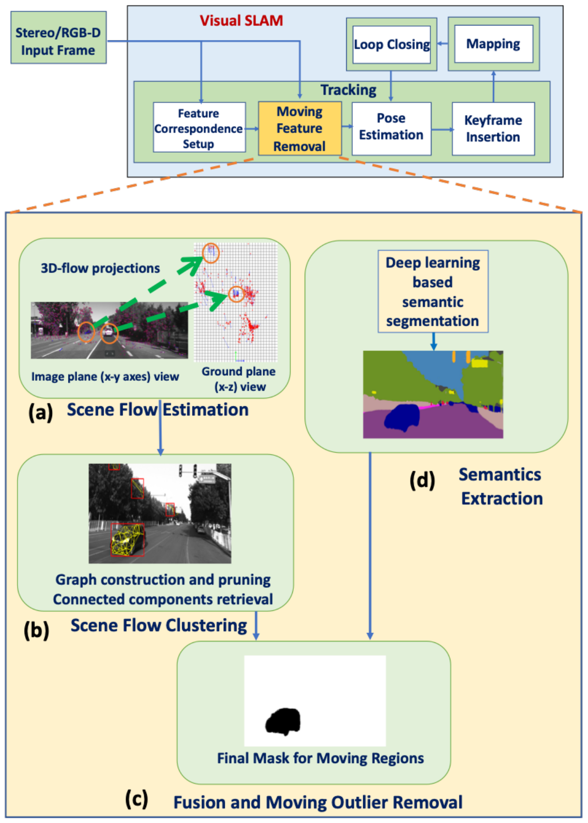
To address this problem, a number of works detect the moving regions in the scene and remove them before the vSLAM processing pipeline. Wang et al. [28] and MVO [16] proposed to detect moving regions by performing clustering of the point trajectories over time. In [28], image segmentation is done by performing clustering of the optical flow. It then refines the over-segmented and under-segmented regions by splitting-merging of groups. This produces dense moving segmentation, and incurs high computational complexity. MVO uses feature tracked over several frames to perform multi-motion clustering. But it significantly suffers from lack of track-lets and feature dropouts, and has been tested only in controlled scenarios. Fang et al. [8] uses dense optical flow to estimate dynamic objects by applying uniform sampling method, but has lower accuracy and the computation complexity is still high when using dense sampling. On the other hand, some works, e.g. MaskSLAM [17] and Detect-SLAM [33], try to remove all features belonging to semantically labelled dynamic class of objects such as car, bicycle, pedestrian etc., irrespective of their actual motion state (moving or not). This results in the loss of many stationary features (as some of these objects are not in a moving state) that could have contributed to effective pose estimation, for example, the scenes with many parked vehicles (motionless state) occupy large parts of the scene (sequence 00 and 08 [11]).

Instead of exploiting either the geometric motion cue or semantic cue alone as mentioned above, some work tries to fuse the two cues together. DS-SLAM [31] first identifies outliers using moving consistency check, which is based on RANSAC, and is thus subjected to the limitations of maximum consensus. It then searches if any of these outliers falls in regions which are semantically labelled as dynamic objects. Due to the presence of random outlier distribution, consistency check do not guarantee true motion. Thus, it results in over-removing large number of stationary features as well.

The proposed method in this paper also exploits both the geometric motion cue and semantic cue to complement to each other. However, unlike [31], we do not rely on RANSAC based moving consistency check for geometric moving object detection, which will not work in scenarios where the majority of the features belong to non-static elements. Instead, a graph based method is proposed to cluster the scene flow to detect the geometric moving region. In addition, taking into consideration that there exist uncertainties both in the extracted geometric motion cue and semantic cue, a probability framework is proposed to fuse them in a robust way. This enables us to distinguish two nearby objects with different motion states as separate components, hence increasing the robustness of the method.

3. Proposed Method

The motion of the static scene with respect to the camera is the inverse of the motion of the camera, which is installed rigidly on a moving vehicle. Hence, the camera pose can be estimated by exploiting the motion pattern of the static scene correspondence in the captured images. The real world, however, is dynamic in nature and contains a lot



Stereo/RGB-D Input Frame
Visual SLAM
Loop Closing
Mapping
Tracking
Feature Correspondence Setup
Moving Feature Removal
Pose Estimation
Keyframe Insertion
3D-flow projections
Image plane (x-y axes) view
Ground plane (x-z) view
(a) Scene Flow Estimation
Deep learning based semantic segmentation
(d) Semantics Extraction
Graph construction and pruning
Connected components retrieval
(b) Scene Flow Clustering
Final Mask for Moving Regions
(c) Fusion and Moving Outlier Removal

Figure 1: Proposed moving feature removal technique can be integrated into any existing vSLAM to make it robust for dynamic environments.

of moving objects. Outlier removal schemes like RANSAC [22] are generally adopted to separate the outliers from the inliers before estimating the pose. However, when the scene is highly dynamic, a large proportion of the image is often occupied by moving objects, as exemplified in the dynamic TUM [25] and Apollo [14] datasets. Thus, the assumption for RANSAC is violated and the accuracy of the estimated pose degrades significantly. Our work focuses on improving the vSLAM capability in dynamic environments by removing large proportions of outlier features related to moving objects. This increases the robustness of maximum consensus based outlier removal in dynamic scenes as the necessary condition for outlier removal is now satisfied, i.e. majority of features satisfy one dominant motion (the camera motion).

Due to the movement of the camera, the projected region in the image corresponding to the static world also encodes motion. In order to differentiate the two types of motion and segment the moving regions from static world, two main approaches are adopted in the literature: 1) motion clustering based on optical flow (i.e. 2d velocity) [13] or scene flow (i.e. 3d velocity) [27, 18]. These works are based on the foundation that motions from different objects exhibit discontinuities in the flow. 2) Movable object identification via object detection [33] or semantic segmentation [17]. Once regions are associated with a dynamic object class, they are regarded to be in motion. However, both of these two approaches have their own limitation. The former suffers from the noise incurred from the computation of the optical flow or scene flow. For the latter method, object detection / semantic segmentation only provides the category information about the object, but cannot guarantee their motion state at a particular time instant. The motion states are important as stationary features from dynamic object classes e.g. parked cars must be retained as they contribute to effective pose estimation.

In this work, we proposed a new approach to detect the moving object regions in the scene by exploring both the geometric motion pattern and semantics in the scene. First, a lightweight scene flow estimation and clustering method is proposed to compute the geometric moving region (Section 3.1). Second, semantic segmentation maps are generated using state-of-art semantic segmentation methods (Section 3.2). Finally, a probability framework is designed to fuse the geometric motion cue and semantic maps (Section 3.3). The proposed probability framework takes into consideration the uncertainties within the geometric motion cue and semantic map to detect the moving objects more robustly. Once the moving object regions are detected, features corresponding to these regions are discarded and the remaining features are fed into the existing vSLAM pipeline to estimate the camera pose and map points (shown in Fig. 1).

3.1. Sparse Scene Flow based Segmentation

3.1.1 Sparse Scene Flow Estimation

It has been shown in LibViso2 [12] that the viso2 features are lightweight and distinctive enough for feature matching. As such, we have adopted viso2 features [12] to compute the correspondences between previous and current frame for the 2d optical flow and also the correspondences between left and right images for the disparity in the case of stereo camera setup. In the case of RGB-D input, the depth values (i.e. disparity d) can be directly obtained [25].

Given the i^{th} feature point $p^i_{t-1}(u^i_{t-1}, v^i_{t-1}, d^i_{t-1})$ at time step $t-1$ and its correspondence $p^i_t(u^i_t, v^i_t, d^i_t)$ at time step t, where u, v represents the horizontal and vertical coordinates in the image coordinate and d refers to the disparity value, the corresponding 3D coordinates, $P^i_\tau = (X^i_\tau, Y^i_\tau, Z^i_\tau)^T$ can be reconstructed via camera triangulation process:

$$X^i_\tau = \frac{(u^i_\tau - c_u)\cdot b}{d^i_\tau}, \quad Y^i_\tau = \frac{(v^i_\tau - c_v)\cdot b}{d^i_\tau}, \quad Z^i_\tau = \frac{b\cdot f}{d^i_\tau}, \quad for \;\; \tau = t-1, t \tag{1}$$

where b is stereo baseline and (c_u, c_v) is principal point. The reconstructed world coordinates P^i_{t-1} and P^i_t using Eq. (1), takes the camera pose at time $t-1$ and time t as their coordinate origin, respectively. Assume, $\tilde{T}_{t-1,t} = [\tilde{R}|\tilde{t}]$ is the approximate relative camera pose from $t-1$ to t. And, the relative pose $\tilde{T}_{t-k,t}$ is estimated with respect to last keyframe (at time $t-k$), which we call as fast-tracking[1]. Then,

$$\tilde{T}_{t-1,t} = \hat{T}_{t-1,t-k}\tilde{T}_{t-k,t} \tag{2}$$

where $\hat{T}_{t-1,t-k}$ is known from previous frame tracking [1].

After compensating for the motion due to camera, the coordinates $\tilde{P}^i_t = (\tilde{X}^i_t, \tilde{Y}^i_t, \tilde{Z}^i_t)^T$ are given by (Eq. (3)),

$$\tilde{P}^i_t = \tilde{R}P^i_{t-1} + \tilde{t} \tag{3}$$

The position difference $V^i_t = (\Delta X^i, \Delta Y^i, \Delta Z^i)^T$, then represents the scene flow:

$$\Delta X^i = \tilde{X}^i_t - X^i_t, \quad \Delta Y^i = \tilde{Y}^i_t - Y^i_t, \quad \Delta Z^i = \tilde{Z}^i_t - Z^i_t \tag{4}$$

Ideally, if $\{\tilde{R}, \tilde{t}\}$ = $\{R, t\}$ and feature correspondences are free of matching errors, then scene flow vector for all static points would be zero-magnitude or close to zero-magnitude. This enables a simple thresholding operation to separate static features from moving features. However, in practice, this is not always possible. Due to incorrect feature matching or the use of motion approximation in Eq. 3, even static points may be associated with large magnitude scene flow. In the following section, a graph based clustering technique is proposed to distinguish between the static points and moving points.

3.1.2 Scene Flow Clustering

Scene flow clustering generally relies on the basis that scene flow corresponding to the same moving rigid body follows a uniform unique motion pattern, while scene flow associated with the static world is incurred from computation noise and exhibits a random pattern [16]. Many clustering methods have been proposed in the literature. However, clustering methods such as K-means or K-Nearest Neighbors (KNN) require the number of clusters as the prior knowledge which is difficult to obtain in practice [15]. Other clustering methods such as subspace clustering is computationally expensive [6]. In our work, we adopted the Delaunay triangulation [2] based graph clustering method for scene flow clustering due to its non-parametric property, low computational complexity, and capability in processing data that contains spatial geometric information [18]. Using this clustering approach, only few edges are connected compared to the number of vertices and efficiently clustered using connected component search [26]. In addition, we prune the graph using both Mahalanobis and Euclidean distance, and final moving regions are obtained using convex hull.

After scene flow is computed for all the feature points, a threshold is applied to remove the feature points whose Mahalanobis magnitude of the scene flow is very small and are confidently regarded as static features. We denote the remaining feature points as potential moving points (PMP). Building the Delaunay triangulation graph on the set of PMP will lead to significant savings in computation cost.

Given the set of PMP, a weighted graph is constructed using the scene flow and the positions (Fig. 1(b)). Each vertex represents a PMP (feature point) by its flow vector V^i_t and its position P^i_t. Vertices in the graph are connected using Delaunay triangulation based on the adjacency of their positions. Then, the edges of the graph are weighted based on the Mahalanobis distance $\Delta(\mathbf{V}^i_t, \mathbf{V}^j_t)$ of the associated scene flow vectors V^i_t, V^j_t as,

$$\Delta(\mathbf{V}^i_t, \mathbf{V}^j_t) = (\mathbf{V}^i_t - \mathbf{V}^j_t)^T \Sigma^{-1}_{ij} (\mathbf{V}^i_t - \mathbf{V}^j_t) \tag{5}$$

which takes into account the uncertainty incurred in the reconstructed position P^i_t, P^j_t due to measurement noise. The uncertainty is modelled as $\Sigma_{ij} = \Sigma_i + \Sigma_j$. Each Σ_i represents the covariance noise defined by $\Sigma_i = \mathbf{J_i S J_i}^T$. $\mathbf{S}$ is the measurement noise matrix taken as **S=diag**(0.5) pixel and $\mathbf{J}_i$ is the Jacobian of scene flow [5].

Once the weighted graph is constructed, the anomalous edges are removed if their weights are large (Eq. 5). We set the threshold for Mahalanobis distance based on the probable speed of moving objects in the scene. For outdoor dataset this threshold is set as $0.15\times$camera fps, and for indoor dataset as $0.01\times$camera fps. A lower threshold value is set for indoors as most of the moving objects are people, who move relatively slow ($\sim$ 0.1-1.5 m/s). This enables us to achieve separate clusters of objects with different speed. On the other hand, for outdoor scenes, the moving objects (mostly vehicles) have faster speeds ($\sim$ 2.7-30 m/s) and are easily distinguishable by using a higher threshold. We also removed any long edges based on the inter-node 3D Euclidean distance ($d_{i,j} = \|P^i_t - P^j_t\|$) , because most of these long edges connect different objects in the real world. Such long edges may occur when different objects move with similar motion, leading to a cluster spanning large image space and potentially covering even the unoccupied static region. We set a threshold of 3 metres for outdoors and 1 metres for indoors, considering the size of moving objects.

This pruning process rapidly removes edges between far features in 3D coordinates and the edges with significantly different scene flow vectors between nodes. After the graph refinement, only the highly similar nodes with close proximity stay connected and forms clusters. We represent each cluster by its convex hull, which is extracted by finding the

[1]Keyframe contains relatively stable features, and estimation can be done in less time than full pose estimation (see supplementary for details).

[2]No other vertex lies inside the circumcircle of a triangle.

connected vertices in the pruned graph, as shown in Fig. 1(b). This scene flow clustering approach efficiently identifies the moving cluster of points and all the features within the hull region can be considered as moving and are omitted from pose estimation and mapping.

The proposed scene flow clustering approach is able to identify the moving objects in the scene correctly in most cases. However, it still suffers from noise incurred during the computation of scene flow. A single false point (graph vertex) can cause incorrect removal of some static (good) features. To further improve the robustness, we explore the use of semantic cues as discussed in the following section.

3.2. Semantic Segmentation

We utilize the semantic information as an additional cue to assist final moving object decision making. Although any state-of-the-art semantic segmentation like [32] and [29] can be utilized, but for fair comparison with the baselines (e.g. [31]) we use SegNet [1]. In particular, the moving cluster regions identified in the previous section will be considered valid only when they are also classified as movable class (vehicle, person, etc.). However, due to the significant uncertainties in both the obtained geometry cue and semantic cue, a simple "AND" or "OR" operation for fusing both information will either lead to over-removal of large number of static regions or failure in removing several moving outliers. Hence, a probability framework is proposed to fuse the two cues in a robust way as described in the next section.

3.3. Fusion of Geometric and Semantic methods

Scene flow based geometric clustering (Section 3.1) indicates which part of the images corresponds to moving regions as shown in Fig. 2(a). A corresponding binary image G is generated to denote the observed moving region map obtained from geometric clustering. Each pixel x_i in G is labelled as 1 if it is estimated as moving point (if it lies inside cluster boundary), otherwise 0 if it is a static point (if it lies inside cluster boundary), as shown in Fig. 2(b). Semantic segmentation from Section 3.2 provides another observation on which part of the images corresponds to dynamic classes(potential moving objects). Similarly, a corresponding binary image S is generated to denote the observed semantic map. All pixels in S will have a value 1 if corresponding pixels are labelled as one of the dynamic classes, otherwise value 0 (Fig. 2(c)).

G and S introduce uncertainty especially on the boundary of the regions corresponding to moving regions or dynamic class. As such, decision of the final moving regions cannot be made directly on G and S. Instead, we introduce a probability model, to compute the probability $p(g_i = mov|x_i)$ that pixel x_i in original image corresponds to moving region based on G and the probability $p(s_i = dyn|x_i)$ that pixel x_i correspond to dynamic class based on S. A technique called distance transform [19] is utilized to convert G and S into a distance map and Gaussian modelling is then utilized to estimate the probability. The intuition is that the likelihood should decrease with the distance to the nearest region observed as moving region (value 1) in G or dynamic class (value 1) in S. The probability model takes into consideration the uncertainty, and G and S can be fused in a more robust way.

Given the binary geometric observation map G, a distance map DT_g is created using the distance transform technique, where Euclidean distance is adopted as the distance metric in this work, as illustrated in Fig. 2(d) and defined in equations Eq 6 and Eq 7.

$$D[i][j] = min\{Distance[(i,j),(x,y)] : B[x][y] = 1\} \quad (6)$$

$$Distance[(i,j),(x,y)] = \sqrt{(i-x)^2 + (j-y)^2} \quad (7)$$

Here, B refers to the binary map. Based on the distance map DT_g, we define the probability $p(g_i = mov|x_i)$ that pixel x_i correspond to moving region (mov) as

$$p(g_i = mov|x_i) = \frac{1}{\sqrt{2\pi}\sigma_g} e^{-\frac{1}{2\sigma_g^2} DT_g(x_i)^2} \quad (8)$$

where σ_g models the uncertainty in the scene flow based geometric segmentation.

Similarly, given the binary semantic observation map S, another distance map DT_s is created using Eq 6 and Eq 7. Based on DT_s (Fig. 2(e)), the probability $p(s_i = dyn|x_i)$ that pixel x_i correspond to dynamic class (dyn) is defined as

$$p(s_i = dyn|x_i) = \frac{1}{\sqrt{2\pi}\sigma_s} e^{-\frac{1}{2\sigma_s^2} DT_s(x_i)^2} \quad (9)$$

where σ_s models the uncertainty in the semantic image classification.

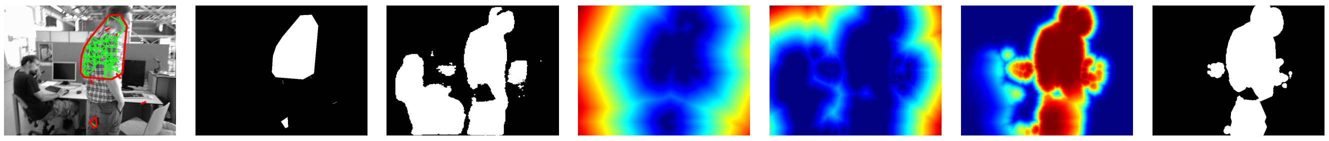
Since semantic segmentation S and scene flow based geometric segmentation G are independently estimated, we can reduce the individual uncertainty of detecting actual moving regions by fusing both the estimates as follows to get true moving object likelihood $p(f_i = trueM|x_i)$.

$$p(f_i = trueM|x_i) = p(s_i = dyn|x_i) \cdot p(g_i = mov|x_i) \quad (10)$$

$$= \frac{1}{2\pi\sigma_s\sigma_g} e^{-\left(\frac{DT_s(x_i)^2}{2\sigma_s^2} + \frac{DT_g(x_i)^2}{2\sigma_g^2}\right)} \quad (11)$$

The fused region can then be computed as shown in Fig. 2(f), based on $\sigma_s = 40$ pixels and $\sigma_g = 80$ pixels. These values depend on the individual quality of segmentation. The final mask for moving region is evaluated by thresholding this estimated fusion probability as shown in Fig. 2(g). The threshold is taken as 0.85 in the current implementation, which is selected based on the qualitative experiments.

As illustrated in the first row of Fig. 2, the cluster (red bounding region Fig. 2(a)) is not able to extract the complete moving person, while, the semantic segmentation is


(a) clusters (red) and edges (green) (b) geometric map (G) of (a) (c) semantic map (S) (d) distance transform DT_g of (b) (e) distance transform DT_s of (c) (f) probability of moving regions (g) final mask obtained from (f)

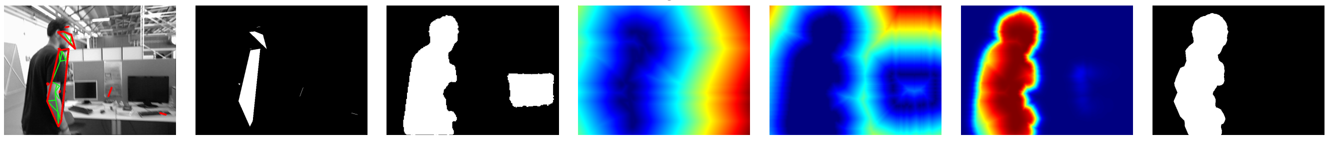

(h) clusters (red) and edges (green) (i) geometric map (G) (j) semantic map (S) (k) distance transform DT_g of (i) (l) distance transform DT_s of (j) (m) probability after fusion (n) final mask obtained from (m)

Figure 2: The process of fusion between geometric clustering and semantic segmentation to get the moving region segmentation mask (g and n) is depicted for sample input frames (a and h).

able to extract the most of the region of the person (Fig. 2(c)). However, the semantic segmentation detects both the static person (on left side in Fig. 2(a)) and moving person (on right side in Fig. 2(a)). Hence, after applying the proposed fusion approach to get probability distribution Fig. 2(f), we can finally extract more accurate mask for the moving person as shown in Fig. 2(g). The second advantage of this fusion approach is that it removes unwanted regions which are falsely segmented as dynamic by semantic segmentation. For example, as shown in Fig. 2(j), the monitor-screen has been segmented as human, but the binary mask obtained from geometric clustering shows that only the person is moving. Hence, after probabilistic fusion, the monitor is removed.

4. Experiments

In our experiments, we have employed the widely-used ORB-SLAM2 as our base vSLAM system. It is worth mentioning that the methods proposed in this work can be applied to other vSLAM systems to increase the robustness. We integrate the proposed technique with ORB-SLAM2 and denote it as proposed-SLAM. In this section, an extensive experimental study will be presented to demonstrate the effectiveness of the proposed technique in various scenarios, i.e. indoor with both static and crowded scenes (TUM [25]), and outdoor with static (KITTI [11]) and crowded scenes (Apollo [14]). In addition to ORB-SLAM2, the recent dynamic SLAM systems DS-SLAM [31], Detect-SLAM [33] and DynaSLAM [2] are chosen as baselines. DynaSLAM is available in both RGB-D and Stereo configurations.

TUM dataset suggests the error metric ATE (absolute trajectory error) to be used for SLAM evaluation. The results in this paper are generated through their online evaluation kit [25]. For the KITTI and Apollo datasets, the poses have been evaluated using the RMSE of relative pose errors (RPE) as defined by KITTI [11]. It measures the average deviation (in translation and rotation) of the estimated poses with respect to ground truth averaged over 100m to 800m intervals i.e. translation errors in $\%$ and rotation errors in $deg/100m$.

4.1. Ablation Studies

We compare the SLAM performance of our three approaches: 1) Using only semantic labels to remove the moving features (**B+S**), 2) using only graph-clustering based moving features removal (**B+G**), and 3) fusion of both geometric and semantic information (**B+G+S**). The experimental results of these three configurations and ORB-SLAM2 (without using our proposed approach) i.e. **B** are shown in Table 1, 2 and 3 for three different datasets.

On the TUM dataset (Table 1), it can be observed that the geometric method **B+G** and semantic method **B+S** have lower average ATE errors than the baseline **B** in most sequences. This is because these sequences contain large part of moving objects in the scene and removing the features in these regions improves the pose estimation accuracy. However, when both information are fused together, i.e. in **B+G+S**, there is a reduction of about 90% in pose errors compared to the baseline **B**. This is due to the fact that semantic method **B+S** removes all features of semantic class of movable objects (e.g. people), irrespective of their actual motion state. But, features on stationary person are useful in cases where, the rest of features (wall, desk etc.) are insufficient in number. On the other hand, geometric method **B+G** sometimes could not segment the entire region of the moving object due to noisy sparse flow, and only provides an idea about the likelihood of moving objects. Hence, the fusion **B+G+S** overcomes the limitation of both **B+S** and **B+G**, by extracting true moving regions as shown in Fig. 2(g). A similar behaviour is observed on KITTI (Table 2) and Apollo datasets (Table 3), where **B+G+S** has the lowest errors on most sequences, but **B+S** over-removes some

Table 1: Results on TUM RGB-D dataset, showing RMSE of absolute trajectory errors (ATE) in meters. The proposed graph-clustering only SLAM (B+G), proposed semantic only (B+S) and proposed fusion (B+G+S) are compared with the Baseline ORB-SLAM2 (B) and recent dynamic vSLAMs DS-SLAM, Detect-SLAM and DynaSLAM.

Sequences	**B (ORB-SLAM2)**	**DS-SLAM**	**Detect-SLAM**	**DynaSLAM**	**B+G**	**B+S**	**B+G+S**
fr3_wlk_xyz	0.7521	0.0247	0.0241	0.017	0.1458	0.0253	**0.015**
fr3_wlk_stat	0.3900	0.0081	-	0.007	0.0806	0.0168	**0.007**
fr3_wlk_rpy	0.8705	0.4442	0.2959	-	0.3376	0.3000	**0.029**
fr3_wlk_half	0.4863	0.0303	0.0514	0.026	0.2627	0.0345	**0.025**
fr3_sit_stat	0.0087	0.0065	-	0.007	0.0078	0.0063	**0.006**
fr3_sit_xyz	**0.0091**	0.0183	0.020	0.015	**0.0091**	0.0145	0.011
fr3_sit_half	0.0264	0.0260	0.0231	0.028	0.0259	0.0209	**0.019**

of the useful features (parked cars) e.g. in sequence 17_23 (Table 3). Occasionally, **B+G** also removes few stationary features, e.g. in sequence $16\_3A\_A2\_19$(Table 3). This happens when the geometric clusters extend beyond the moving object due to noisy sparse flow. However, the fusion approach **B+G+S** overcomes this problem. In sequences where moving vehicles are continuously visible in front of camera, e.g. KITTI sequence 04 (Table 2), the **B+G+S** error reduction is more noticeable.

4.2. Evaluation on TUM Dataset (RGB-D)

The TUM RGB-D dataset [25] consists of several low-dynamic (sitting people) to highly dynamic sequences (walking people). In extreme cases, more than half of the images are occupied with moving objects e.g. TUM (sequence *fr3_wlk_half*) [25]. In walking sequences, people are walking in front of the camera, and static, half, rpy and xyz denotes the four types of camera motion. The quantitative comparison results of ORB-SLAM2 (**B**), DS-SLAM, Detect-SLAM, DynaSLAM and the proposed-SLAM (**B+G+S**) are shown in Table 1. The RMSE ATE errors are very high for ORB-SLAM2. This is due to the fact that their outlier removal method is impaired in the case where the scene is majorly occupied by moving objects. In addition, DS-SLAM and Detect-SLAM have larger errors in all sequences than the proposed-SLAM. This is because many semantically dynamic pixels are removed even though they are in static motion state. These features could have contributed to better pose estimation. DynaSLAM performs better than DS-SLAM and Detect-SLAM in some sequences, because it use motion detection with semantics. In very low dynamic sequence *fr3_sit_xyz*, a few noisy clusters slightly increase the errors for **B+G+S**, but the proposed **B+G+S** consistently achieves very low errors in all sequences.

4.3. Evaluation on KITTI Dataset (Stereo)

The KITTI visual odometry dataset is another popular benchmark. It provides stereo sequences captured from a moving vehicle in mostly static environment i.e. very few moving objects are present in the scenes. Table 2 shows the comparison of the average translation and rotation errors between proposed-SLAM (**B+G+S**), ORB-SLAM2 (**B**) and DynaSLAM. As KITTI dataset contains very few moving objects in the sequences, and ORB-SLAM2 is able to remove most these few dynamic features through its outlier removal, the improvements are minor. However, it is noticeable that the improvements in proposed (**B+G+S**) are more pronounced in sequences which contain more moving vehicles. For example in sequence 04, a moving van is perpetually in front of the camera. Hence, a slight error reduction is observed in this sequence. Apart from this, in sequence 07, a truck is present for few frames, and sequence 08 also contains a few moving vehicles. Overall, error reduction of **B+G+S** over ORB-SLAM2 is 4.5% in translation errors and 0.3% in rotation.

4.4. Evaluation on Apollo Dataset (Stereo)

We chose the newly released dataset called Apollo [14] for our experiments since it has more comprehensive scenarios than KITTI and contain significant number of moving vehicles and pedestrians. Apollo provides outdoor self-localization stereo dataset [14]. The stereo dataset is however not rectified and contains noisy data and missing frames. Hence, we rectified and re-sized the images to fit to the KITTI type stereo frames. Apollo also provides the groundtruth poses for evaluation. To evaluate the poses, we use popular outdoor average $\%$ translation and average rotation error matrices defined by KITTI [11]. The average errors are higher ($\sim$15-20 times) on this dataset as compared to KITTI dataset. This is because camera goes through highly varying speeds and the camera-motion is not as smooth as KITTI. Hence, low number of features can be tracked in successive frames. Also this dataset contains more prominent rotations than KITTI and thus is more challenging for localization.

The quantitative comparison results are shown in Table 3. The translation errors of ORB-SLAM2 increase to large value in some of the sequences (up to 30%). These sequences contain large number of moving objects including

Table 2: KITTI dataset: the translation errors t (%) and rotation errors r ($deg/100m$) for **B**, DynaSLAM, **B+G**, **B+S** and **B+G+S** are shown.

Sequences	B (ORB-SLAM2)		DynaSLAM		B+G		B+S		B+G+S	
	t	r	t	r	t	r	t	r	t	r
00	0.70	0.25	0.74	0.26	**0.68**	**0.24**	0.69	0.26	**0.68**	0.25
01	1.39	**0.21**	1.57	0.22	1.40	0.22	**1.36**	0.22	**1.36**	0.22
02	**0.76**	0.23	0.80	0.24	0.78	0.24	**0.76**	0.23	**0.76**	**0.21**
03	0.71	0.18	**0.69**	0.18	0.71	0.18	**0.69**	**0.17**	0.73	0.20
04	0.48	0.13	0.45	**0.09**	0.45	0.12	0.47	0.14	**0.39**	0.12
05	0.40	**0.16**	0.40	**0.16**	0.39	0.18	0.40	0.17	**0.38**	**0.16**
06	0.51	**0.15**	0.50	0.17	0.49	**0.15**	0.55	0.19	**0.48**	**0.15**
07	0.50	**0.28**	0.52	0.29	0.51	0.31	0.53	0.30	**0.47**	0.29
08	1.05	0.32	1.05	0.32	**1.03**	0.34	1.04	0.34	**1.03**	**0.31**
09	0.87	**0.27**	0.93	0.29	0.95	0.32	0.85	0.32	**0.84**	**0.27**
10	0.60	0.27	0.67	0.32	0.60	0.27	0.59	0.29	**0.56**	**0.26**

Table 3: Results on Apollo datasets, translation errors t (%) and rotation errors r (deg/100m). **B**, **B+S**, **B+G** and **B+G+S** are compared below.

Sequences	B (ORB-SLAM2)		B+G		B+S		B+G+S	
	t	r	t	r	t	r	t	r
17_26	33.03	14.36	22.48	4.18	23.85	4.16	**22.44**	**3.46**
17_23	22.69	3.98	24.02	5.30	25.92	7.45	**20.49**	**0.77**
17_05	18.75	4.53	**16.63**	**3.48**	18.41	6.00	17.77	4.77
16_3A _A2_15	36.50	14.06	30.79	6.35	**25.91**	6.84	26.13	**6.16**
16_3A_A2_19	**20.07**	1.98	21.75	2.28	21.45	2.18	20.09	**1.02**
16_2E_D2_06	32.19	8.64	25.27	6.86	26.66	8.86	**24.85**	**6.16**
16_2E_D2_10	27.94	4.06	**19.21**	**2.12**	27.18	4.34	19.71	2.61
16_2E_D2_23	19.46	1.04	19.00	1.78	20.03	1.8	**18.87**	1.10
16_2E_D2_26	17.95	1.38	17.96	2.26	18.55	1.54	**17.89**	**1.21**
15_03	**22.58**	6.43	24.94	**6.18**	24.61	17.36	23.15	6.12

vehicles and pedestrians in the scene [14]. The comparison shows that proposed (**B+G+S**) overcomes the limitation of ORB-SLAM2 (**B**) in highly outdoor dynamic environments as well, by removing the features related to the moving objects in the scene. For instance, the translation error reduces from 33% to 22% on 17_26 and from 36% to 26% on $16\_3A\_A2\_15$. However, in sequence 15_03, the rotation errors of the proposed-SLAM are slightly higher. This is due to highly incorrect semantic labels in certain frames, where some vehicles are classified as static background and hence even the fusion approach could not remove corresponding moving outliers. Overall, the proposed (**B+G+S**) errors decreased by 12.8% in translation and 33.1% in rotation errors compared to ORB-SLAM2.

4.4.1 Timing Evaluation

The timing results are shown in Table 4 on Intel-Xeon CPU@3.50GHz 6 cores, the baseline B takes 39.4 ms, whereas the proposed $B+G+S$ takes an additional 16.74 ms. The average computation time for semantic segmentation is 29.15 ms, which run in a parallel thread on Nvidia GeForce GTX1080 Ti GPU. This semantic output is used by fusion module, as shown in Fig. 1.

Table 4: Timing evaluation.

Module	Feature extraction	Scene Flow Computation	Geometric Clustering	Fusion	Pose Estimation
Time (ms)	17.9	7.11	4.45	5 .18	21.5

5. Conclusion

We proposed a technique to enhance the robustness of vSLAM in highly dynamic environment by accurately removing features on moving objects. This increases the robustness of maximum consensus based outlier removal in dynamic scenes as the necessary condition for outlier removal is now satisfied. The proposed technique is based on fusion of semantic and geometric information to detect moving regions. Our ablation study highlights the importance of distinguishing between motion states of potential moving objects for vSLAM in highly dynamic environments. To demonstrate the effectiveness of the proposed technique, experiments were conducted on challenging datasets, i.e. TUM, KITTI and Apollo. The results show that the proposed vSLAM outperforms DS-SLAM, Detect-SLAM, DynaSLAM and ORB-SLAM2 in various scenarios.

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[2] B. Bescos, J. M. Fcil, J. Civera, and J. Neira. Dynaslam: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robotics and Automation Letters*, 3(4):4076–4083, Oct 2018.

[3] D. Chekhlov, A. P. Gee, A. Calway, and W. Mayol-Cuevas. Ninja on a plane: Automatic discovery of physical planes for augmented reality using visual slam. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–4. IEEE Computer Society, 2007.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):1052–1067, 2007.

[5] L. Devroye, L. Györfi, and G. Lugosi. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media, 2013.

[6] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2765–2781, 2013.

[7] J. Engel, J. Stckler, and D. Cremers. Large-scale direct slam with stereo cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942, Sept 2015.

[8] Y. Fang and B. Dai. An improved moving target detecting and tracking based on optical flow technique and kalman filter. In *2009 4th International Conference on Computer Science & Education*, pages 1197–1202. IEEE, 2009.

[9] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[10] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, May 2014.

[11] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[12] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968, June 2011.

[13] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[14] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 954–960, 2018.

[15] M. Jaimez, M. Souiai, J. Stückler, J. Gonzalez-Jimenez, and D. Cremers. Motion cooperation: Smooth piece-wise rigid scene flow from rgb-d images. In *2015 International Conference on 3D Vision*, pages 64–72. IEEE, 2015.

[16] K. M. Judd, J. D. Gammell, and P. Newman. Multimotion visual odometry (mvo): Simultaneous estimation of camera and third-party motions. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3949–3956. IEEE, 2018.

[17] M. Kaneko, K. Iwami, T. Ogawa, T. Yamasaki, and K. Aizawa. Mask-slam: Robust feature-based monocular slam by masking using semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 258–266, 2018.

[18] P. Lenz, J. Ziegler, A. Geiger, and M. Roser. Sparse scene flow segmentation for moving object detection in urban environments. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 926–932. IEEE, 2011.

[19] K.-N. Lianos, J. L. Schonberger, M. Pollefeys, and T. Sattler. Vso: Visual semantic odometry. In *Proceedings of the European conference on computer vision (ECCV)*, pages 234–250, 2018.

[20] R. Mur-Artal and J. D. Tards. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, Oct 2017.

[21] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*, pages 2320–2327, Nov 2011.

[22] D. Nistér. Preemptive ransac for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329, Dec 2005.

[23] T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera, and J. J. Berlles. S-ptam: Stereo parallel tracking and mapping. *Robotics and Autonomous Systems*, 93:27–42, 2017.

[24] D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. *IEEE Robotics Automation Magazine*, 18(4):80–92, Dec 2011.

[25] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580. IEEE, 2012.

[26] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, 1972.

[27] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 722–729. IEEE, 1999.

[28] Y. Wang and S. Huang. Towards dense moving object segmentation based robust dense rgb-d slam in dynamic scenarios. In *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 1841–1846. IEEE, 2014.

[29] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 418–434, 2018.

[30] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad. An overview to visual odometry and visual slam: Applications to mobile robotics. *Intelligent Industrial Systems*, 1(4):289–311, 2015.

[31] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei. Ds-slam: A semantic visual slam towards dynamic environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1168–1174. IEEE, 2018.

[32] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. Icnet for real-time semantic segmentation on high-resolution images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 405–420, 2018.

[33] F. Zhong, S. Wang, Z. Zhang, C. Chen, and Y. Wang. Detect-slam: Making object detection and slam mutually beneficial. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1001–1010, March 2018.