Accurate RGB-D SLAM in Dynamic Environments using Observationally Consistent Conditional Random Fields

Anonymous cvm submission

Paper ID 28

Abstract

In this paper, we present a novel RGB-D SLAM approach for accurate pose estimation in dynamic environments. Previous methods detect dynamic components only across a short time-span of consecutive frames. Instead, we provide a more accurate dynamic 3D landmark detection method, followed by the use of observationally consistent conditional random fields, which leverages long-term observations from multiple frames. We further introduce an efficient initial camera pose estimation method based on the initially distinguishing dynamic from static points using graph-cut RANSAC. These static/dynamic labels are used as priors to guide the definition of the unary potential in the conditional random fields, which further improves the accuracy of dynamic 3D landmark detection. Evaluation using the public TUM RGB-D dynamic dataset shows that our approach significantly outperforms state-of-the-art methods, providing much more accurate camera trajectory estimation in a variety of highly dynamic environments.

1. Introduction

Simultaneous localization and mapping (SLAM) technology underpins various applications and is the subject of intense research interest in computer vision, computer graphics, and robotics communities. Although SLAM technology has made significant progress in the past few decades [6, 9], most works focus on static environments lacking dynamic components such as people and moving objects. However, real applications often work in dynamic environments which include human beings and other moving objects—in such cases, methods designed for static environments typically fail to accurately estimate pose. Accurate dynamic SLAM which works efficiently in dynamic environments is urgently needed as a basis for applications in robotics, virtual reality, autonomous vehicles, *etc*.

The critical challenge for dynamic SLAM lies in the presence of dynamic components, which violate the data relationships assumed in static SLAM, and thus lead to poor pose estimation. An intuitive solution to this problem is to detect and remove any dynamic components, and to perform pose estimation solely on the static components. However, joint pose estimation and dynamic component detection is a chicken and egg problem, since detection of dynamic components relies on pose estimation, while pose estimation requires dynamic component removal. Thus, dynamic SLAM is a more difficult problem than static SLAM.

Detection and tracking of moving objects (DATMO), introduced by the impressive work of Wang et al. [28], is one of the main approaches for tackling dynamic SLAM; it works by detecting moving objects in two or a few consecutive frames. However, previous DATMO based methods suffer from drawbacks that make assumptions on the moving objects: they should occupy a limited range of distances [1], there should only be a few of them [27, 20], or their speed should be restricted [2]. CoSLAM [7] introduced a SLAM system for dynamic environments using multiple cameras. It uses the error between each reprojected feature point and the observed feature point in the 2D image plane to distinguish static and dynamic feature points. Kim et al [14] suggested a method based on segmenting foreground and background regions of each frame, and using the background (hopefully, mostly static) to perform pose estimation. Li et al [18] provided a static weighting method for edge points, using the static weight in an intensity assisted iterative closest point method for camera pose estimation. However, the way all of these previous methods used to determine which regions are static and dynamic relies only on an analysis of a short time-span of consecutive frames. This precludes improving the accuracy of moving object detection over time, with a consequent impact on the accuracy of camera pose estimation.

In this paper, we provide a more accurate dynamic SLAM method with an RGB-D sensor, by analysis of frames over long-term timescales instead of only short-term ones. The key component of our RGB-D SLAM system is a dynamic camera tracking module based on accurate dynamic 3D landmark detection. For dynamic 3D landmark detection, our key observation is that moving objects can

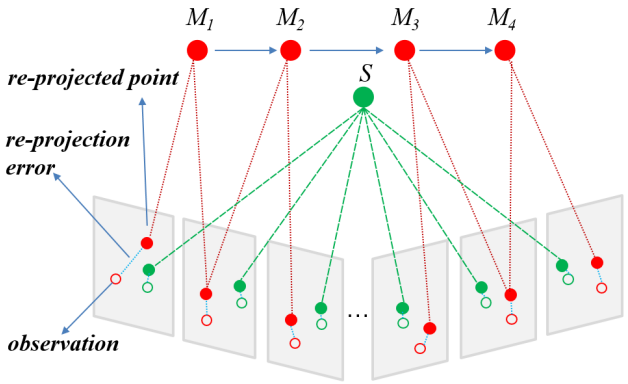


Figure 1. A static landmark has more consistent observations than a dynamic one. M is a moving landmark which moves from M_1 to M_4 quickly; just a few frames observe the same location. Static landmark S stays at the same location and is seen at the same position in more frames. Re-projected points from static landmarks triangulate to a consistent landmark, while re-projected points from dynamic landmarks triangulate to different landmarks.

be determined more reliably by using long-term observations instead of only brief observations. Static objects will be observed to be consistent, i.e. the 2D observations in the views in different frames can all be back-reprojected to one single object. In contrast, the 2D observations of dynamic objects will not be consistent and will be back-reprojected into multiple objects due to the objects' motion, as shown in Fig. 1.

Based on this key observation, we build an observationally consistent conditional random field (OC-CRF) model to assist in 3D dynamic landmark detection, based on analyzing the observations of static and dynamic landmarks over a long-term series of consecutive frames. Solving the labeling problem with the aid of the CRF provides highly accurate dynamic detection results. By eliminating the dynamic 3D landmarks in this way, we can estimate the camera pose with much higher precision using only the static 3D landmarks.

The performance of dynamic 3D landmark detection in our method is influenced by the initial estimate of the camera pose: the 2D observations used in our analysis in the OC-CRF model need an excellent initial camera pose. To obtain a reasonable estimate for each frame, we coarsely label the temporal 3D landmarks as static or dynamic by formulating the labeling problem as the inlier/outlier determination problem for fundamental matrix estimation between the current frame and a reference frame; we solve it efficiently using graph-cut (GC) RANSAC [3]. While this does not provide completely accurate labels, the labeling is good enough to provide an initial camera pose. Furthermore, the initial static and dynamic labels from the initial camera pose estimate provide strong priors for subsequent dynamic 3D landmark detection, which makes the dynamic 3D landmark detection more accurate.

We have evaluated our approach on the public TUM RGB-D dynamic dataset [25]; which contains several dynamic scenes, ranging from 720 frames to 4200 frames, with two persons walking through an office. The results show that our approach frequently outperforms state-of-the-art approaches, such as BaMVO [14] and SPW [18].

In summary, this paper makes the following contributions:

(1) a reliable dynamic 3D landmark detection method based on an observationally consistent conditional random field, which constitutes the main component of the dynamic camera tracking method, and

(2) an efficient method for obtaining an initial estimation of the camera pose for each frame, based on GC-RANSAC filtering, which also provides strong static versus dynamic priors for dynamic 3D landmark detection.

2. Related work

Simultaneous localization and mapping has been studied for more than four decades, with sub-topics of lidar SLAM, visual SLAM, and sensor fusion SLAM according to the different sensors used. In this paper, we focus on visual SLAM, which utilizes cameras (which may be monocular, stereo, or RGB-D) as the primary sensor for localization. In this section, we discuss works of particular relevance to ours and refer to [6] for a more detailed overview of visual SLAM progress in the past few decades.

2.1. Static Visual SLAM

There has been much progress in visual SLAM techniques since the pioneering work of MonoSLAM [8] in 2003. Current visual SLAM approaches can be divided into two categories: (i) *feature-based* visual SLAM methods, which use sparse feature points as landmarks for camera tracking, e.g. PTAM [15] and ORB-SLAM2 [21], and (ii) *direct* visual SLAM, which directly uses image intensity for camera tracking without feature points or landmarks, e.g. DTAM [23], SVO [12], LSD-SLAM [11] and DSO [10]. Direct visual SLAM techniques have the advantage of allowing efficient camera tracking without the time-consuming 2D feature detection required by feature-based visual SLAM techniques, but they often suffer from lack of robustness in changing light conditions.

Currently, most visual SLAM techniques assume a static environment and do not work well in dynamic environments including human beings or other moving objects. Since feature-based visual SLAM methods such as ORB-SLAM2 [21] work well for robust camera tracking, our approach is also a feature-based SLAM system containing

three components: camera tracking, local mapping and loop closing as like ORB-SLAM2 does. The novelty component of our SLAM system lies in the camera tracking subsystem, which in our case handles scenes with dynamic objects. We integrate our dynamic 3D landmark detection and elimination method into the camera tracking component, allowing it to work more accurately in dynamic environments.

2.2. Dynamic Visual SLAM

Wang et al. [28] introduced a remarkable method for dynamic SLAM based on the detection and tracking of moving objects (DATMO) in 2006. It was designed for lidar SLAM, and gave an impressive performance for pose estimation. The idea of DATMO inspired many following dynamic visual SLAM approaches. Alcantarilla et al [1] showed how to robustly detect moving objects with the aid of dense scene flow. However, the maximum distance any object may move between two consecutive frames is limited to 5 meters, and static points can often be misjudged to be dynamic points in textureless regions. Bakkay et al [2] presented a segmentation-based method that can successfully detect quickly moving objects, but they assumed that objects have constant motion, which limits its applicability.

Kerl et al [13] give the Dense Visual Odometry (DVO) algorithm, which uses a robust error function to reduce the influence of moving objects on camera pose estimation. However, since the error function is only computed across two consecutive frames, the DVO algorithm can only work well for slowly moving environments instead of rapidly changing ones, which cause incorrect data associations. Recently, Kim et al [14] introduced a background-model-based dense-visual-odometry (BaMVO) algorithm to estimate the background of each frame and to perform camera pose estimation by eliminating the foreground moving objects. Li et al [18] provide a dynamic RGB-D SLAM method which uses foreground edge points to estimate the camera's ego-motion. In this method, every edge point is assigned with a static weight which is used in an intensity-assisted iterative closest point (IAICP) algorithm for ego-motion estimation; this reduces the influence of dynamic components. Most of these methods detect dynamic components by analysis of only a few consecutive frames, two frames in DVO [13] and just the current frame in BaMVO [14] and Li et al [18].

However, short-term analysis is not informative enough for moving object detection, since many dynamic components may remain static for short periods, which may mislead short-term determination of static/dynamic status. If not properly detected and eliminated, such dynamic components may be used as landmarks for later camera tracking, misleading downstream 3D-2D data association, thus lowering the accuracy of camera pose estimation.

In this paper, instead, we provide a dynamic component detection method that uses long-term analysis.Distinguishing static from dynamic components can be done more reliably using long-term observations. Based on this insight, we build an observationally consistent conditional random field by introducing feature vectors based on multiple visual observation errors over a long period of consecutive frames.

Other works [29, 30] use deep networks such as Faster-RCNN [24] to detect moving objects. Although such methods achieve good performance, the computational cost is much higher due to the use of deep networks. We believe that a geometric approach to dynamic component detection is still not well explored and show that accuracy can be significantly improved without the need for a deep network.

3. Method

3.1. System Overview

An overview of our proposed approach is given in Figure 2. Following ORB-SLAM2 [21] (RGB-D version), our system also consists of three components: dynamic camera tracking, local mapping and loop closing. Local mapping and loop closing are performed as in ORB-SLAM2. The dynamic camera tracking component aims to efficiently estimate the ego-motion for the incoming frames by accurately detecting and eliminating dynamic 3D landmarks. Specifically, we first efficiently estimate an initial camera pose for each frame using a GC-RANSAC filter, then accurate dynamic 3D landmark detection is performed based on our observationally consistent conditional random fields (OC-CRF). After detecting dynamic 3D landmarks, we perform ego-motion estimation again based on the static 3D landmarks. Our dynamic camera tracking component thus contains two main subcomponents.

The first subcomponent performs *initial camera pose estimation* using GC-RANSAC filtering (see Section 3.2). The initial camera pose guess is important for our downstream dynamic 3D landmark detection process, since the 2D observations used in the OC-CRF are influenced by the initial camera pose. In this subcomponent, we make an initial identification of static and dynamic points using 2D-2D matching with GC-RANSAC filtering, which is very efficient and accurate. Then the points determined as static are used for initial camera pose estimation. This initial static/dynamic identification is also used in the dynamic 3D landmark detection step later.

The second subcomponent performs *dynamic 3D landmark detection* using an observationally consistent CRF (see Section 3.3). Given the initial camera pose estimate from the previous step, we build an observationally consistent conditional random field (OC-CRF) and use it to *accurately* identify static and dynamic feature points, by solving

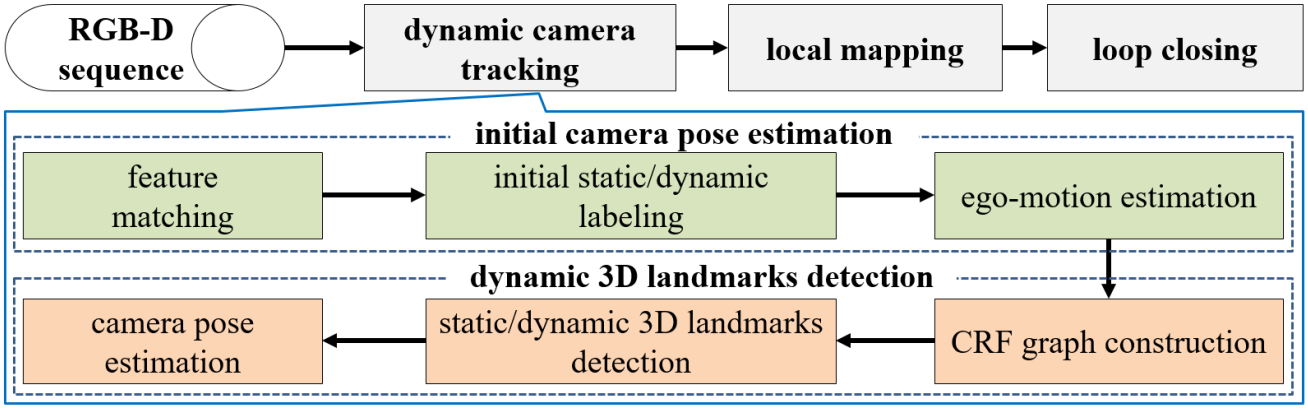


Figure 2. Overview of our proposed approach.

a labeling problem on the OC-CRF. This allows us to eliminate the dynamic points, and just use the static points to refine the camera pose of the current frame.

Using our dynamic camera tracking method, we can accurately estimate the ego-motion between the current frame and the previous frame, and robustly detect and eliminate any dynamic points. Then keyframes are selected as in ORB-SLAM2, and are sent to the local mapping components with a graph-based bundle adjustment to improve the camera pose estimation further.

3.2. Initial Camera Pose Estimation

For every incoming frame, we need to determine a reasonable initial estimate of its camera pose. A general way to do this is to estimate the ego-motion between two consecutive frames by solving a perspective-n-point (PnP) problem [19] with 3D-2D data association, as ORB-SLAM2 does. However, in dynamic scenarios, the 3D-2D data association will contain incorrect matches due to the existence of moving objects, as shown in Fig. 3. To avoid this, feature points on moving objects must be detected and eliminated, leaving static feature points to provide an accurate estimate of the ego-motion.

In this step, we first efficiently and coarsely label landmarks as static or dynamic, and then estimate the ego-motion using only the static landmarks. For efficiency, we formulate the static/dynamic landmark identification problem as inlier/outlier identification during fundamental matrix estimation using the GC-RANSAC algorithm.

Specifically, when estimating the ego-motion between the current frame K_i and its reference frame K_i', as shown in Fig. 4, we first perform 2D-2D matching and obtain a set of 2D feature point matching pairs $P = \{(p_i, p_i') | p_i \in K_i, p_i' \in K_i'\}$. Using the set P, we estimate the fundamental matrix $F(K_i, K_i')$ relating K_i and K_i', with which a 2D feature point $p_i \in K_i$ is projected as an epipolar line $l_i' \in K_i'$ with $l_i' = Fp_i$. For a 2D matching pair

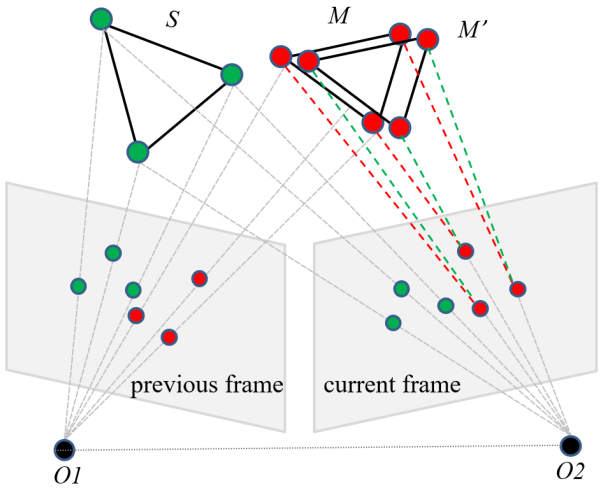


Figure 3. Incorrect associations between landmarks and feature points in the presence of dynamic objects. S is a static object, while M is a moving object. Between the previous and current frames, M moves to a new position M'. When landmarks in the previous frame are projected to the current frame, landmarks from M will be mismatched with the feature points of M'. Correct associations are shown by green dashed lines, while red dashed lines show erroneous associations.

$(p_i, p_i') \in P$, if $p' \in K_i'$ lying on the epipolar line l_i', then (p_i, p_i') is an inlier pair for F, otherwise it is an outlier. Once the fundamental matrix F relating K_i and K_i' has been accurately estimated, we can judge whether the feature point p_i of an inlier matching pair (p_i, p_i') is a static point or a dynamic point. Thereafter we compute the 3D-2D matching using only the static point set of K_i and accurately estimate the ego-motion by solving the perspective-n-point problem. The key issue thus lies in accurately estimating the fundamental matrix F.

Graph-cut RANSAC is a robust and efficient algorithm providing a state-of-the-art, accurate, geometric approach

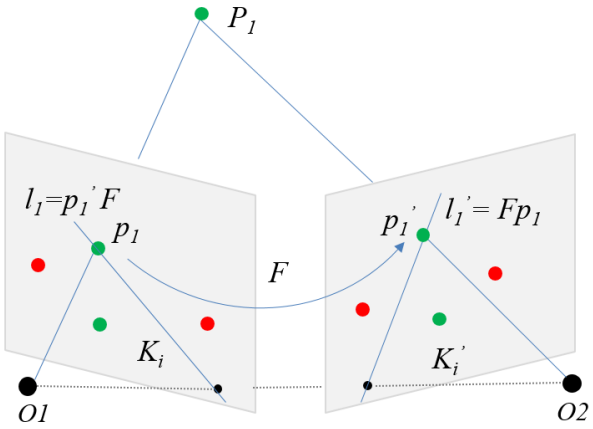


Figure 4. Fundamental matrix and epipolar constraint. For a matched pair (p_i, p_i'), if p_i and p_i' are related to the same 3D point, the epipolar constraint can be expressed as: $p_i' F p_i = 0$, i.e. p_i' lies in the epipolar line $l_1' = Fp_i$ or p_i lies in the epipolar line $l_1 = p_i' F$, where F is the fundamental matrix.

for a range of problems such as fitting lines, homographies, affine transformations, and fundamental and essential matrices. Its main idea is to formulate inlier/outlier selection as a binary ($\{0,1\}$) labeling problem on each iteration of RANSAC, which is efficiently solved via local optimization using the graph-cut algorithm [5]. Specifically, during the fundamental matrix F estimation, for a given 2D-2D matching pair set $M = \{(p_i, p_i')|i = 1, \ldots, n\}$, on each iteration of RANSAC we label each matching pair as an inlier or an outlier for fundamental matrix F estimation. This is performed by optimizing the energy function $E(L) = \sum_i B(L_i) + \lambda \sum_{(i,j)\in G} R(L_i, L_j)$ with $L = \{L_i \in \{0,1\}|i = 1, \ldots, n\}$ being a label assignment for the matching pair set M, and G being a neighbor graph.

The unary term of the energy function is formulated as:

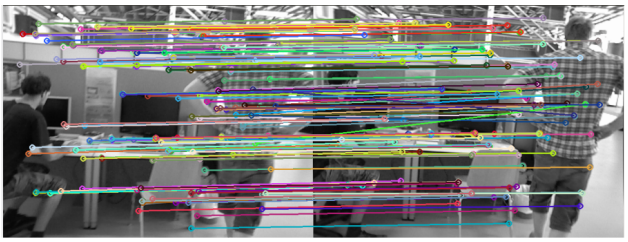
$$B(L_i) = \begin{cases} K\left(\phi\left(p_i, p_i', \theta\right), \epsilon\right) & \text{if } L_i = 0 \\ 1 - K\left(\phi\left(p_i, p_i', \theta\right), \epsilon\right) & \text{if } L_i = 1 \end{cases} \quad (1)$$

where θ is the parameter for fundamental matrix F, $K(\sigma, \varepsilon) = \exp(-\sigma^2/(2\varepsilon^2))$. Label $L_i = 0$ represents an inlier pair and 1 represents an outlier. $\phi(p_i, p_i', \theta)$ measures the distance from matching pair (p_i, p_i') to the fundamental matrix θ, and ε is a threshold for the inlier/outlier determination.

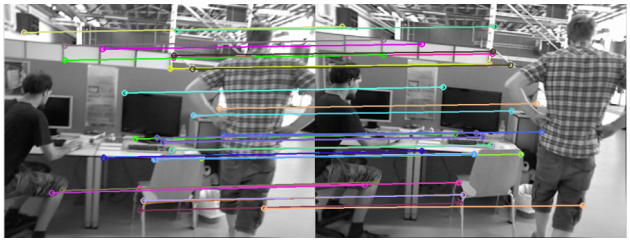
The pairwise energy is defined as follows:

$$R(L_i, L_j) = \begin{cases} 1 & \text{if } L_i \neq L_j \\ \frac{1}{2}(B(L_i) + B(L_j)) & \text{if } L_i = L_j = 0 \\ 1 - \frac{1}{2}(B(L_i) + B(L_j)) & \text{if } L_i = L_j = 1 \end{cases} \quad (2)$$

The total energy can be efficiently optimised by the graph cut algorithm [4].



(a) Feature matching between reference and current frames before GC-RANSAC



(b) Feature point pairs labeled as inliers after GC-RANSAC

Figure 5. Static feature point selection by GC-RANSAC filter. Left:current frame, Right:reference frame. We choose the 10th frame before the current frame as the reference frame. After GC-RANSAC filtering, inliers are almost all static feature points, and are used for initial ego-motion estimation.

In this paper, we wish to aggressively remove dynamic points, even at the expense of discarding some static ones. We achieve this by empirically setting $\varepsilon = 0.1$, and $\lambda = 0.14$.

Through GC-RANSAC, all landmarks are labeled as inlier (static) or outlier (dynamic). We then estimate the ego-motion using just the static landmarks, to obtain a more accurate pose estimate. As an input to the GC-RANSAC algorithm, in addition to the current frame, we also need to select a reference frame. As shown in Fig.3, there are many mismatches between adjacent frames arising due to dynamic objects, and which is difficult to filter out by GC-RANSAC. Thus, instead, we choose two frames that are far apart in time as an input to GC-RANSAC. This helps to ensure that almost all inliers labeled as static are indeed static.

However, there are two main issues: firstly, as shown in Fig.5, after our aggressive GC-RANSAC filtering, only a few feature points are labeled as static landmarks, and used for pose optimization, so the pose is not determined accurately enough. Secondly, only a small number of landmarks are matched, which results in tracking being lost frequently. To overcome these problems, using the ego-motion estimated above, we find more static landmarks for pose estimation by projecting all landmarks seen in the previous frame into the current frame, and estimate the fundamental matrix between previous frame and current frame.

We later use the estimated fundamental matrix to derive static/dynamic priors for accurate dynamic point

Algorithm 1 Initial Camera Pose Estimation

Input:
current frame f_c, reference frame f_r, previous frame f_l
Output:
pose of current frame T_c

```
Match features between f_c and f_r
Suggest static feature points by GC-RANSAC
for each static feature point p_i in f_c do
    Find the corresponding landmark P_i in f_r
end for
Estimate ego-motion on static landmarks
Project landmarks seen by f_l to f_c
Estimate fundamental matrix by GC-RANSAC
for Each landmark P_i in f_c do
    Compute the static/dynamic identification prior
end for
return T_c
```

detection—see Section 3.3. Specifically, as shown in Fig.4. For each 2D matching pair $(p_i, p_i'), p_i \in K_i, p_i' \in K_i'$, where K_i and K_i' are the current frame and the previous frame respectively, assuming P_i is the corresponding 3D landmark, and $l \in K, l' \in K'$ are the corresponding epipolar lines $l = p_i' \cdot F$, $l' = F \cdot p_i$, we compute the distances between the 2D feature point and the epipolar line as $d_i = l \cdot p_i$ and $d_i' = l' \cdot p_i'$. In general, if landmark P_i is a static point, we expect the symmetric epipolar distance error $\gamma_i = (d_i + d_i')/2$ to be small. We thus define a likelihood of static for each landmark P_i as $P_{\gamma_i} = \exp(-(\gamma_i - \mu_\gamma)^2/(2\sigma_\gamma^2))$, where μ_γ is the mean of all the γ_i's. We then use P_{γ_i} as the static/dynamic identification prior for each landmark P_i for detecting dynamic points.

This processing is summed up in Algorithm 1. To effectively distinguish static and dynamic feature points, we choose the 10th frame before the current frame as the reference frame f_r. This reduces the number of mismatches significantly.

3.3. Dynamic Landmark Detection by CRF

After estimating the initial camera pose of the current frame, we now identify the 3D landmarks as static or dynamic. As shown in Fig. 1, the basis of our approach is that dynamic points tend to have more inconsistent observations than static points, especially over a long time of frames. Here, by observation we mean the corresponding 2D feature point in the image plane as seen from a given frame. If a point's observations from multiple frames can be accurately triangulated to a single 3D landmark, we say that point's observations are consistent. Obviously, a dynamic point's observations will be less consistent than those of a static point. Furthermore, dynamic points often have larger photometric re-projection errors between the re-projected point and the corresponding 2D feature point. We also note that points in the neighborhood of a static or dynamic point also tend to be static or dynamic, respectively. This key observation motivates us to use an observationally consistent conditional random field (OC-CRF) to accomplish the dynamic point detection.

Conditional random fields (CRF) are undirected graph models used for multi-class data segmentation and labeling [17], with unary potentials on individual nodes and pairwise potentials on adjacent nodes. In this paper, we construct a fully connected graph [16] linking each pair of 3D landmarks. For each node P_i, we assign a label $x_i = L_i \in \{0, 1\}$ where 0 represents a static point and 1 a dynamic point. By minimizing the Gibbs energy E, we obtain the optimum label for every 3D landmark:

$$E(X) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j) \tag{3}$$

We design the unary potential $\psi_u(x_i)$ and pairwise potential $\psi_p(x_i, x_j)$ to incorporate static/dynamic information from the long-term observations, which is why we call our CRF 'observationally consistent' (OC-CRF).

The unary potential is determined as follows. During SLAM processing, each landmark can be seen in several keyframes. We record the corresponding 2D observations $o_j^i \in R^2$, i.e. the 2D position, from keyframe j for each 3D landmark P_i. Specifically, the photometric re-projection error e_j^i between P_i and o_j^i is calculated. By averaging the re-projection errors we obtain $\alpha_i = (\sum_j e_j^i)/\beta_i$ where β_i is the total number of observations of P_i. Similar to the static likelihood prior P_{γ_i} for the landmark P_i, we define a second static likelihood from all the observations as $P_{\beta_i} = \exp(-(\beta_i - \mu_\beta)^2/(2\sigma_\beta^2))$ and a third one from the average re-projection error as $P_{\alpha_i} = \exp(-(\alpha_i - \mu_\alpha)^2/(2\sigma_\alpha^2))$, where $\mu_\cdot$ and $\sigma_\cdot$ and represent means and standard deviations of respective quantities.

For each landmark, we thus have three different estimations of the likelihood that the landmark P_i is static: P_{α_i}, P_{β_i} and P_{γ_i}. We simply use a simple use the average of these estimations to measure the likelihood that P_i is static: $P_i^s = (P_{\alpha_i} + P_{\beta_i} + P_{\gamma_i})/3$. Given a threshold th, if the average $P_i^s \geq th$, P_i is initially labeled as static, otherwise, labeled as dynamic.

Following [22], the unary potential is then defined as

$$\psi_u(x_i) = \begin{cases} -\log(P_i^s) & \text{if } x_i = 0 \\ -\log(1 - P_i^s) & \text{if } x_i = 1 \end{cases} \tag{4}$$

where $x_i = 0$(static) if $P_i^s \geq th$, otherwise, $x_i = 1$(dynamic). In this paper, we set $th = 0.3$.

The pairwise potential aims to encourage similar kinds of landmarks to have similar labels. The pairwise potential is the sum of two Gaussian kernels, as follows:

$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \sum_m \omega^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) \quad (5)$$

where $\mu(x_i, x_j) = 1_{[x_i \neq x_j]}$ is a simple Potts model, $\mathbf{f}_i$ and $\mathbf{f}_j$ are feature vectors for nodes i and j, and each $k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)$ is a Gaussian kernel.

We use 2 Gaussian kernels here: the *observation kernel* and the *location kernel* which are defined in terms of average re-projection error α_i, total number of observations β_i, 3D location of the landmark P_i, and 2D location of p_i. For landmarks P_i and P_j with different labels, we expect there to be significant differences in the attributes mentioned above, so the pairwise potential of $\psi_p(x_i, x_j)$ should be assigned a lower value, leaving the labels x_i and x_j more likely to be unchanged. Otherwise, if these values for P_i and P_j are similar, $\psi_p(x_i, x_j)$ will be assigned a higher value, so P_i and P_j tend to belong to the same class.

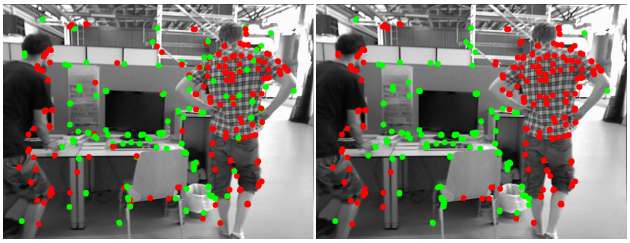
The *observation kernel* is based on the idea that landmarks with a similar number of observations and average re-projection errors are likely to be in the same class. A dynamic landmark can be seen in the same position only for a few keyframes, while a static landmark can be seen in many more keyframes over a longer-term. Similarly, static landmarks have lower average re-projection errors than dynamic landmarks. Thus, landmarks with different labels should have apparent differences in the number of observations and average re-projection error. So, the observation kernel is defined as:

$$k^{(1)}(\mathbf{f}_i, \mathbf{f}_j) = \exp(-\frac{|\alpha_i - \alpha_j|^2}{2\sigma_\alpha^2} - \frac{|\beta_i - \beta_j|^2}{2\sigma_\beta^2}) \quad (6)$$

The *location kernel* is based on the idea that nearby landmarks are likely to be in the same class, belonging as a compact group to an object which is either static (e.g. a table) or dynamic (e.g. a person). Thus the location kernel penalizes pairs of landmarks with different labels but close to each other. This particularly helps to remove isolated landmarks surrounded by landmarks with the opposite label. As shown in Fig.6, in (a) and (b), the static feature points in the person are surrounded by dynamic ones (left image), and these are re-labeled as dynamic by OC-CRF inference (right image). The location kernel function is defined as:

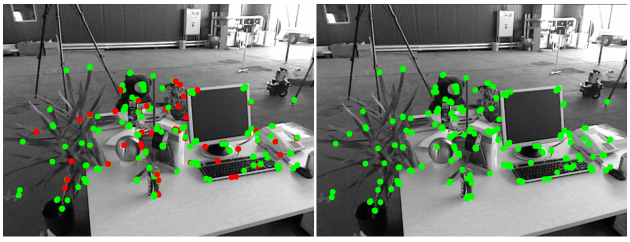
$$k^{(2)}(\mathbf{f}_i, \mathbf{f}_j) = \exp(-\frac{|P_i - P_j|^2}{2\sigma_P^2} - \frac{|p_i - p_j|^2}{2\sigma_p^2}) \quad (7)$$

The static/dynamic labeling problem represented by our OC-CRF can be solved efficiently using a mean field approximation method [16]. We show several examples illustrating our landmark labeling results for sequences from the TUM RGB-D benchmark in Fig. 6. As can be seen, our method significantly improves the results for static/dynamic point labeling. Dynamic landmarks are segmented accurately even for highly dynamic scenes, refer to the attachment video for more results.



(a) Dynamic scene 1

(b) Dynamic scene 2



(c) Static scene

Figure 6. Dynamic landmark detection in dynamic scenes (a,b) and a static scene (c). Left: initial static/dynamic labeling, with static points marked in green ($p_i^s \geq th$) and dynamic points red ($p_i^s < th$. Right: final dynamic 3D landmark detection results after OC-CRF optimization.

After dynamic landmark detection, we discard dynamic landmarks and use the remaining static ones to redetermine the camera pose of the current frame more accurately. These steps are summarized in Algorithm 2.

4. Experiments

4.1. Preliminaries

To evaluate the accuracy of estimated camera pose, we tested our method on the public TUM RGB-D dynamic dataset [26], we selected 6 different indoor dynamic sequences with moving people and violent camera shaking for evaluation.

We compared our method with the original ORB-SLAM2 method, which does not have dynamic point detec-

Sequence	ORB-SLAM ATE (m)				OC-CRF SLAM ATE (m)			
	RMSE	Std	Mean	Median	RMSE	Std	Mean	Median
fr3/walking-xyz	0.366287	0.255601	0.262363	0.162223	**0.018335**	**0.008711**	**0.015775**	**0.013733**
fr3/walking-halfsphere	0.382438	0.187725	0.333194	0.318089	**0.029800**	**0.014495**	**0.022781**	**0.022489**
fr3/walking-static	0.214124	0.083655	0.197106	0.175119	**0.010446**	**0.006624**	**0.015890**	**0.010950**
fr3/walking-rpy	0.744576	0.401184	0.627252	0.603298	**0.114289**	**0.084301**	**0.077172**	**0.051715**
fr3/sitting-xyz	0.010889	0.005032	0.009656	0.008829	**0.009333**	**0.004939**	**0.007922**	**0.007235**
fr2/desk-with-person	0.074397	**0.016205**	0.072610	0.073081	**0.071795**	0.016229	**0.069937**	**0.070072**

Table 1. Absolute trajectory error for dynamic datasets for the ORB-SLAM method and our OC-CRF SLAM method, measured in metres.

Algorithm 2 Dynamic Point Detection and Accurate Pose Estimation

Input:
landmarks seen by current frame f_c
Output:
accurate pose of current frame $T*_c$
Initialize CRF
for Each landmark **do**
  Compute unary potentials from Eq. (4)
end for
for Each pair of landmarks **do**
  Compute pairwise potentials from Eqs. (6,7)
end for
Determine dynamic landmarks by CRF inference
Estimate pose $T*_c$ from static landmarks
Return $T*_c$

tion, to evaluate the improvement that our dynamic point detection module makes in a dynamic environment. We also compared our OC-CRF approach with other related dynamic SLAM methods: DVO [13], BaMVO [14] and static point weighting (SPW) [18].

Various system parameters were set as follows for these tests: in kernels, we set $\mu_\gamma = 0.3$,$\sigma_\gamma = 0.2$, $\mu_\alpha = 1.51$, $\sigma_\alpha = 0.6$, $\mu_\beta = 4.81$, $\sigma_\beta = 1.86$. In pairwise potentials, we set $\omega^1 = 10$, $\omega^2 = 30$. For the location kernel, we set $\sigma_P = 2.75$ and $\sigma_p = 20.0$. All experiments were performed on a computer with a 3.6 GHz Intel Core i9-9900K CPU with 16GB RAM, and without use of a GPU.

In the following, we first compared our OC-CRF SLAM with the original ORB-SLAM system. The effectiveness of GC-RANSAC is also tested. Finally, to evaluate the accuracy, we compared our OC-CRF SLAM with other state-of-the-art dynamic SLAM systems.

4.2. Comparison with unmodified ORB-SLAM

We first evaluated the performance for our dynamic camera tracking compared with the original ORB-SLAM. We tested our method and ORB-SLAM on the six dynamic sequences from the TMU RGB-D dataset, and then calculated the absolute trajectory error (ATE) and relative pose error (RPE), as defined in [26], between the camera poses estimated by our method and the ground truth.

The comparison of ATE is shown in Table 1, where 'RMSE' means the root mean squared error of ATE and 'Std' means the standard deviation of ATE. For highly dynamic sequences (names beginning with 'walking', i.e. fast moving persons or camera), our proposed method achieves significantly lower RMSE, Std, Mean and Median than unmodified ORB-SLAM. In the last two scenarios 'siting-xyz' and 'desk-with-person' with less dynamic environments, our algorithm also achieves slightly better results.

The ATE between estimated trajectories and ground-truth is visualized in Fig 7. As can be seen clearly, the trajectories estimated by our OC-CRF SLAM (in the middle row) are much closer to the real trajectories than those of the unmodified ORB-SLAM (in the top row).

RPE is compared in Table 2, where 't' denotes translational and 'r' denotes rotational error. For all highly dynamic RGB-D datasets, our proposed method has significantly lower RMSE and Std. However, for less dynamic environments, some static landmarks may be detected as dynamic landmarks, affecting the subsequent pose estimation. Nevertheless, our method is still better than or very close to ORB-SLAM in such cases.

4.3. Effectiveness of GC-RANSAC Filter

We also evaluated the performance of the initial camera pose estimation using the GC-RANSAC filter from Section 3.2. We built a SLAM system without the initial camera pose estimation component by just assigning an initial camera pose using velocity prediction as ORB-SLAM does. Consequently, the unary and pairwise potentials also do not contain the initial static/dynamic priors for the OC-CRF for the dynamic landmark detection. We compared such a system (without the GC-RANSAC filter) with our full OC-CRF SLAM system by evaluating the ATE and RPE of the six dynamic sequences of the TUM RGB-D dataset.

Table 3 shows ATE results on our OC-CRF SLAM with and without the GC-RANSAC filter. Without use of the GC-RANSAC filter for initial pose estimation, the ATEs are significantly greater for highly dynamic sequences such as *fr3/walking-xyz* and *fr3/walking-halfsphere*. For less

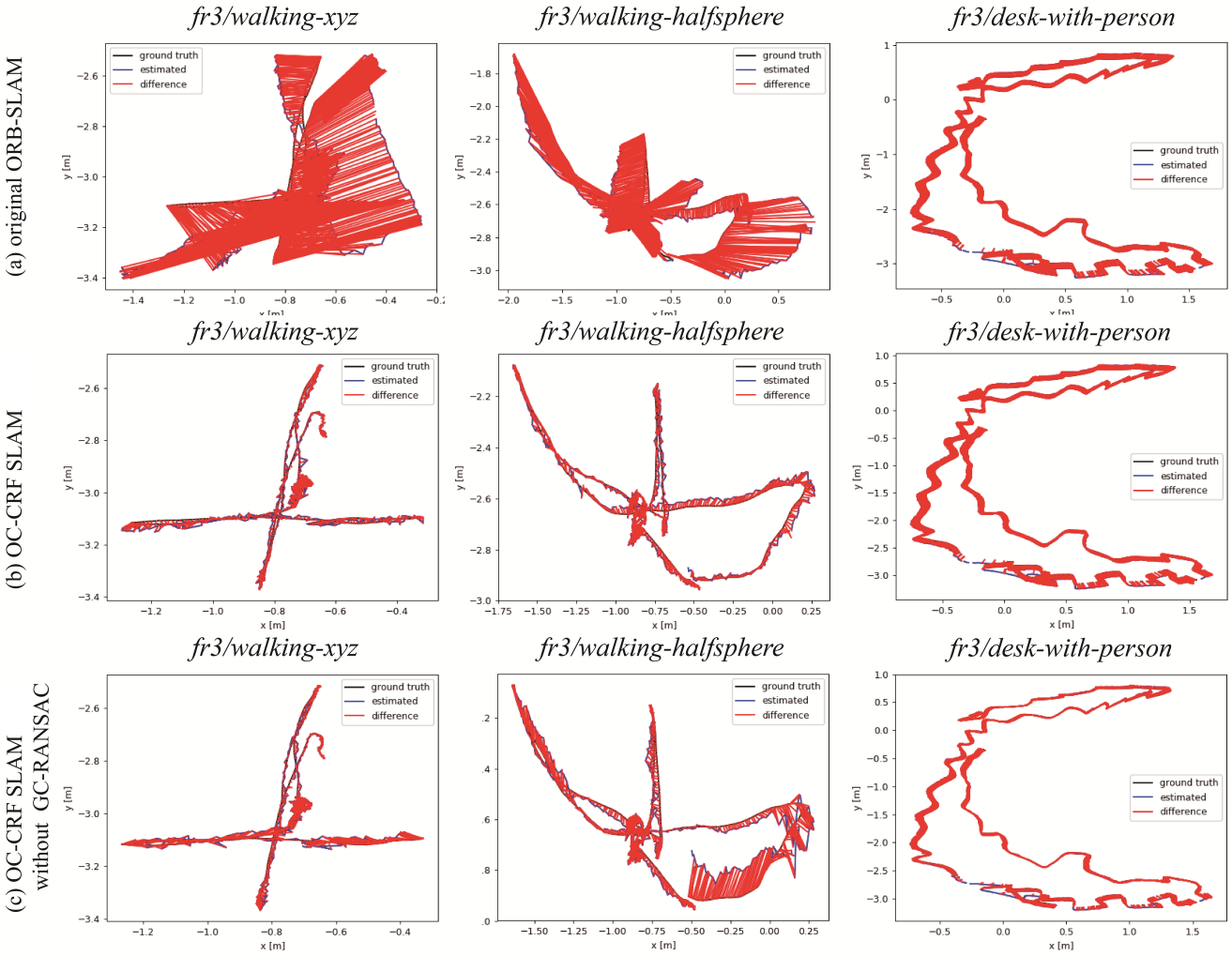


Figure 7. Visualization of ATE. Blue: estimated trajectories. Black: ground-truth trajectories. Red lines connect corresponding points in these two trajectories: their length indicates the estimation error. Top: trajectories from unmodified ORB-SLAM. Center: trajectories from OC-CRF SLAM. Bottom: trajectories from OC-CRF SLAM without GC-RANSAC. Clearly, OC-CRF SLAM generates more accurate camera trajectories than unmodified ORB-SLAM. OC-CRF using CRF alone has somewhat lower accuracy than OC-CRF SLAM with GC-RANSAC.

Sequence	**ORB-SLAM RPE**				**OC-CRF SLAM RPE**			
	t.RMSE	**t.Std**	**r.RMSE**	**r.Std**	**t.RMSE**	**t.Std**	**r.RMSE**	**r.Std**
fr3/walking-xyz	0.517820	0.387784	8.958071	7.274051	**0.025704**	**0.011565**	**0.645109**	**0.375732**
fr3/walking-halfsphere	0.570142	0.316124	9.254973	7.292348	**0.039497**	**0.020618**	**0.815216**	**0.352377**
fr3/walking-static	0.311129	0.211853	5.509479	3.687236	**0.016719**	**0.009602**	**0.402316**	**0.158952**
fr3/walking-rpy	1.093185	0.632268	9.554797	9.790038	**0.164724**	**0.119515**	**3.050945**	**2.183991**
fr3/sitting-xyz	0.015945	0.007168	0.598171	**0.300411**	**0.013663**	**0.006734**	**0.579074**	0.309012
fr2/desk-with-person	0.104156	0.052091	0.722745	0.328341	**0.100364**	**0.049486**	**0.714301**	**0.315173**

Table 2. Relative pose error for dynamic datasets for the original ORB-SLAM and our OC-CRF SLAM, measured in m/s or °/s as appropriate.

dynamic sequences, the ATEs are slightly increased. In Fig. 7, the bottom row shows the ATE generated by OC-CRF SLAM without GC-RANSAC, which has larger errors than the middle row with GC-RANSAC, verifying the usefulness of GC-RANSAC.

Sequence	with GC-RANSAC	without GC-RANSAC
fr3/walking-xyz	**0.018335**	0.027677
fr3/walking-halfsphere	**0.009333**	0.024205
fr3/walking-static	**0.029800**	0.057692
fr3/walking-rpy	**0.010446**	0.016060
fr3/sitting-xyz	**0.114289**	0.100444
fr2/desk-with-person	0.071795	**0.065325**

Table 3. Absolute trajectory error (in meters) of OC-CRF SLAM with and without GC-RANSAC.

Sequence	DVO	SPW	OC-CRF
fr3/walking-xyz	0.0932	0.0601	**0.0183**
fr3/walking-halfsphere	0.0470	0.0432	**0.0298**
fr3/walking-static	0.0656	0.0261	**0.0104**
fr3/walking-rpy	0.1333	0.1791	**0.1142**
fr3/sitting-xyz	0.0482	0.0397	**0.0093**
fr2/desk-with-person	0.0596	**0.0484**	0.0717

Table 4. Absoloute trajectory error (in meters) for dynamic datasets, for DVO, SPW and our OC-CRF SLAM methods.

4.4. Comparison with existing methods

We finally compare our proposed OC-CRF SLAM method with other state-of-the-art RGB-D SLAM systems: BaMVO, dense visual odometry (DVO), and static point weighting (SPW). Table 4 shows the corresponding ATE results (BaMVO's results are missing since not provided in their original paper). We can see that our system outperforms the others for all of the highly dynamic datasets, often by a considerable margin. The only case our method performs poorly is the sequence *fr2/desk-with-person*. This is an almost static scene, and a few static landmarks are labeled as dynamic by GC-RANSAC filter with its standard parameter settings, which degrades the accuracy of the initial pose estimation.

Table 5 shows relative pose error results for these methods. For RMSE of rotational drift, our method performs better than all other methods. For RMSE of translational drift, our method also achieves more accurate results for almost all datasets. Specifically, in the best cases, our method has RPE errors which are less than 1/3 of those of the state-of-the-art method, while for the same reason mentioned above, our method performs worse on the *fr2/desk-with-person* sequence in terms of translation drift. This verifies that our OC-CRF SLAM reduces the influence of dynamic objects effectively, especially for highly dynamic scenes.

5. Conclusion

In this paper, we have presented the OC-CRF SLAM system for accurate pose estimation and effective dynamic point detection. To reduce the impact of dynamic points on pose estimation,we firstly compute an initial pose by use of GC-RANSAC and assign each landmark a static/dynamic prior. Then, we use a CRF with appropriate unary and pairwise potentials to label each landmark as static or dynamic. We show that our proposed OC-CRF SLAM is significantly more accurate than existing methods for the highly dynamic examples in the public TUM RGB-D dataset.

Our approach has three main drawbacks. The first is that for almost static scenes, performance may be slightly degraded, due to labeling some static objects as dynamic in an effort not to let dynamic objects corrupt pose estimation. This leaves less static data for pose estimation, reducing its accuracy slightly. Secondly, if a moving object remains stationary for a long time, it will be considered as a static object and cannot be detected. Thirdly, the initial ego-motion estimate depends on GC-RANSAC, which is a randomized algorithm. Thus the final result of dynamic landmark detection inherently presents some randomness. In the future, a deep learning method could be combined to accurately identify dynamic objects, reduce the randomness of the algorithm, so that it can perform robustly in various environments.

References

[1] P. F. Alcantarilla, J. J. Yebes, J. Almazan, and L. M. Bergasa. On combining visual slam and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In *IEEE International Conference on Robotics & Automation*, 2012. 1, 3

[2] M. C. Bakkay, M. Arafa, and E. Zagrouba. Dense 3d slam in dynamic scenes using kinect. In *Iberian Conference on Pattern Recognition & Image Analysis*, 2015. 1, 3

[3] D. Barath and J. Matas. Graph-cut ransac. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6733–6741, 2017. 2

[4] B. Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *IEEE International Conference on Computer Vision*, 2002. 5

[5] Y. Boykov and G. Funkalea. Graph cuts and efficient n-d image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006. 5

[6] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, Dec 2016. 1, 2

[7] Z. Danping and T. Ping. Coslam: collaborative visual slam in dynamic environments. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 35(2):354–366, 2013. 1

[8] A. J. Davison, I. D. Reid, N. Molton, and O. Stasse. Monoslam: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1052–1067, 2007. 2

Sequence	RMSE of translational drift (m/s)				RMSE of rotational drift (°/s)			
	DVO	BaMVO	SPW	our OC-CRF	DVO	BaMVO	SPW	our OC-CRF
fr3/walking-xyz	0.4360	0.2326	0.0651	**0.0257**	7.6669	4.3911	1.6442	**0.6451**
fr3/walking-halfsphere	0.2628	0.1738	0.0527	**0.0394**	5.2179	4.2863	2.4048	**0.8152**
fr3/walking-static	0.3818	0.1339	0.0327	**0.0167**	6.3502	2.0833	0.8085	**0.4023**
fr3/walking-rpy	0.4038	0.3584	0.2252	**0.1647**	7.0662	6.3398	5.6902	**3.0509**
fr3/sitting-xyz	0.0453	0.0482	0.0219	**0.0136**	1.4980	1.3885	0.8446	**0.5790**
fr2/desk-with-person	0.0296	0.0299	**0.0173**	0.1003	1.3920	1.1167	0.7266	**0.3151**

Table 5. Relative pose error for DVO, BaMVO, SPW and our OC-CRF SLAM methods.

[9] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics Automation Magazine*, 13(2):99–110, June 2006. 1

[10] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(3):611–625, 2018. 2

[11] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: large-scale direct monocular SLAM. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II*, pages 834–849, 2014. 2

[12] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 15–22, 2014. 2

[13] C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for RGB-D cameras. In *2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, May 6-10, 2013*, pages 3748–3754, 2013. 3, 8

[14] D.-H. Kim and J.-H. Kim. Effective background model-based rgb-d dense visual odometry in a dynamic environment. *Trans. Rob.*, 32(6):1565–1573, Dec. 2016. 1, 2, 3, 8

[15] G. Klein and D. W. Murray. Parallel tracking and mapping for small AR workspaces. In *Sixth IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR 2007, 13-16 November 2007, Nara, Japan*, pages 225–234, 2007. 2

[16] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. 2012. 6, 7

[17] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. 6

[18] S. Li and D. Lee. Rgb-d slam in dynamic environments using static point weighting. *IEEE Robotics & Automation Letters*, 2(4):2263–2270, 2017. 1, 2, 3, 8

[19] Long Quan and Zhongdan Lan. Linear n-point camera pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):774–780, Aug 1999. 4

[20] D. Moratuwage, B. N. Vo, and D. Wang. Collaborative multi-vehicle slam with moving object tracking. In *IEEE International Conference on Robotics & Automation*, 2013. 1

[21] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Trans. Robotics*, 33(5):1255–1262, 2017. 2, 3

[22] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji. Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. 2019. 6

[23] R. A. Newcombe, S. Lovegrove, and A. J. Davison. DTAM: dense tracking and mapping in real-time. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pages 2320–2327, 2011. 2

[24] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017. 3

[25] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 2

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IEEE/RSJ International Conference on Intelligent Robots & Systems*, 2012. 7, 8

[27] Y. Sun, L. Ming, and Q. H. Meng. Improving rgb-d slam in dynamic environments: A motion removal approach. *Robotics & Autonomous Systems*, 89(Complete):110–122, 2017. 1

[28] C. C. Wang, C. Thorpe, M. Hebert, S. Thrun, and H. Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *International Journal of Robotics Research*, 26(9):889–916, 2007. 1, 3

[29] S. Yang, J. Wang, G. Wang, X. Hu, and Q. Liao. Robust rgb-d slam in dynamic environment using faster r-cnn. In *IEEE International Conference on Computer & Communications*, 2018. 3

[30] F. Zhong, W. Sheng, Z. Zhang, C. Chen, and Y. Wang. Detect-slam: Making object detection and slam mutually beneficial. In *IEEE Winter Conference on Applications of Computer Vision*, 2018. 3