SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

October 2022, Vol. 65 202206:1–202206:14
https://doi.org/10.1007/s11432-021-3425-8

Accurate RGB-D SLAM in dynamic environments based on dynamic visual feature removal

Chenxin LIU[1], Jiahu QIN[1,2*], Shuai WANG[1], Lei YU[1] & Yaonan WANG[3,4]

[1]*Department of Automation, University of Science and Technology of China, Hefei* 230027*, China;*
[2]*Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei* 230088*, China;*
[3]*College of Electrical and Information Engineering, Hunan University, Changsha* 410082*, China;*
[4]*National Engineering Research Center of Robot Visual Perception and Control Technology, Hunan University, Changsha* 410082*, China*

Received 27 July 2021/Revised 13 November 2021/Accepted 30 December 2021/Published online 27 September 2022

Abstract Visual localization is considered an essential capability in robotics and has attracted increasing interest for the past few years. However, most proposed visual localization systems assume that the surrounding environment is static, which is difficult to maintain in real-world scenarios due to the presence of moving objects. In this paper, we present DFR-SLAM, a real-time and accurate RGB-D SLAM based on ORB-SLAM2 that achieves satisfactory performance in a variety of challenging dynamic scenarios. At the core of our system lies a motion consensus filtering algorithm estimating the initial camera pose and a graph-cut optimization framework combining long-term observations, prior information, and spatial coherence to jointly distinguish dynamic and static visual features. Other systems for dynamic environments detect dynamic components by using the information from short time-span frames, whereas our system uses observations from a long period of keyframes. We evaluate our system using dynamic sequences from the public TUM dataset, and the evaluation demonstrates that the proposed system outperforms the original ORB-SLAM2 system significantly. In addition, our system provides competitive localization accuracy with satisfactory real-time performance compared to closely related SLAM systems designed to adapt to dynamic environments.

Keywords SLAM, dynamic environments, indoor localization, graph-cut, robot navigation

Citation Liu C X, Qin J H, Wang S, et al. Accurate RGB-D SLAM in dynamic environments based on dynamic visual feature removal. Sci China Inf Sci, 2022, 65(10): 202206, https://doi.org/10.1007/s11432-021-3425-8

1 Introduction

Visual localization has received a considerable amount of attention in recent years due to the requirements for autonomous driving [1, 2] and robot navigation [3, 4]. Many state-of-the-art visual localization systems [5–8] are proposed with high accuracy and robustness. These vision-based localization systems, which include visual simultaneous localization and mapping (vSLAM) and visual odometry (VO), can estimate the ego-motion of the camera using only consecutive images captured from the surrounding environment. To simplify the problem formulation, most existing systems make the premise that the surrounding environment is static. However, because moving objects, such as humans, are always present in real-world scenes, suitable systems for these scenes are strictly limited. Traditional SLAM algorithms would easily intermingle the movement of moving objects and the camera, resulting in poor localization accuracy and even system failure. For the scenarios in that the camera view is occluded by a small number of moving objects, general methods like robust cost function [6] and random sample consensus (RANSAC) [9] can remove the majority of the interference caused by moving objects. In contrast, when moving objects occupy the majority of the camera view, and especially when the majority of visual features are located within these areas, the quality of visual localization can be severely compromised. To address the problem of localization accuracy reduction or system failure in highly dynamic scenes, the

* Corresponding author (email: jhqin@ustc.edu.cn)

SLAM community has adopted some methods such as prior knowledge generation [10, 11] and analysis based on short-term observations [12]. Prior knowledge, such as semantic information, can, however, hardly recognize objects that have not been pre-trained, and such methods work well at the expense of a significantly high computational cost on the learning part. In addition to prior knowledge, observations over a short time span of frames are insufficient for determining dynamic visual features, because some dynamic objects may remain stationary for a short time interval and be misidentified as static components. Therefore, it still is challenging to improve the performance of vSLAM and VO in dynamic scenes.

Different from the above two types of methods, in this paper, we propose an effective RGB-D SLAM method to detect and eliminate dynamic visual features. The key components of our method are inspired by the idea that observations over long-term timescales can provide sufficient information to distinguish visual features, and nearby visual features tend to have the same motion patterns. Specially, we first estimate the initial camera pose with the temporally labeled static and dynamic identification generated by the proposed motion consensus filtering algorithm. In this step, the properties of indoor scenarios and RGB-D camera are considered. Then we build a sparse undirected graph using Delaunay triangulation [13] from all tracked visual features to determine their adjacency relationship and apply a graph-cut optimization framework to assist in dynamic visual feature detection. The step fully utilizes the observations of corresponding visual features extracted from keyframes over a long time span, as well as the prior property and spatial coherence of visual features in the current frame. The graph-cut optimization framework is used in our method to compute the optimal binary segmentation by minimizing the established energy function. Using the graph-cut model to solve the detection problem yields satisfactory results. Finally, we seamlessly integrate the proposed method into the original ORB-SLAM2 [6], and a real-time and accurate RGB-D SLAM system named dynamic visual feature removal SLAM (DFR-SLAM) is developed, which eliminates the influence of dynamic objects without external sensors as well as prior knowledge. The system is evaluated on public benchmarks [14], and the results demonstrate that DFR-SLAM accurately estimates the camera pose and achieves a more comprehensive performance than some closely related SLAM systems in most sequences. The main contributions of our work are summarized as follows.

(1) A motion consensus filtering algorithm is proposed, which can coarsely estimate the initial camera pose. It exploits the geometry constraint and the motion consistency of visual features belonging to the background.

(2) We propose an effective dynamic visual feature removal method based on a graph-cut optimization framework that considers long-term observations, prior information, and spatial correlations of the visual features.

(3) In dynamic environments, we propose a real-time RGB-D SLAM system that is seamlessly integrated with our dynamic visual feature removal method and achieves high accuracy of pose estimation while having a lower computational cost without the use of external sensors or pre-training models.

The remaining of this paper is structured as follows. Section 2 discusses various accomplishments in respect of vSLAM and VO for dynamic environments. In Section 3, the technical details of the proposed method are elaborated. Section 4 presents our experimental results and analysis. Finally, in Section 5, a brief conclusion is summarized and future work is given.

2 Related work

As most existing vSLAM and VO systems can only eliminate part of the influence of dynamic components in low-dynamic scenarios, many algorithms have been proposed to address more specifically the dynamic scenario content related to visual localization. These methods can be roughly categorized into the following categories: methods based on dynamic factor removal, methods incorporating external sensors, and methods based on deep learning technologies.

2.1 Methods based on dynamic factor removal

The motivation of this kind of method is to remove dynamic visual features on the same moving object by motion consistency. Li and Lee [15] used frame-to-keyframe registration to obtain the reprojection errors of edge points and weight the edge points according to the reprojection errors. The likelihood of an edge point being maintained for localization is decided by the weight. Cheng et al. [16] proposed to

detect and track dynamic regions utilizing a Bayesian framework; the probability propagation model is built by combining the temporal information from the consecutive frames and the observations between the reference frame and the current frame. Dai et al. [17] built a sparse graph using the tracked 3D landmarks and then segmented the graph based on point correlations. The size of the connected component distinguishes static and dynamic visual features. This method detects dynamic components by utilizing temporal information from frame-to-keyframe or consecutive frame registration, whereas we propose to use long-term observations and spatial coherence to distinguish static and dynamic visual features, which have not been considered in other related work based on dynamic factor removal.

2.2 Methods incorporating external sensors

Incorporating external sensors like wheel-encoders and inertial measurement units (IMUs), which can compensate for the unreliable pose estimation, is another efficient way to address the problem. Kim et al. [18] utilized inertial data between two consecutive frames, which are relatively accurate, to compensate for the rotation components of estimation and then introduce a motion vector filter to classify the extracted visual features as dynamic or static ones. Yang et al. [19] fused the observations of wheel-encoders and RGB-D camera to estimate the robot movement. The combination of the deep learning method and a geometric method whose reprojection error is obtained from wheel-encoder compensation is used to identify dynamic pixels. Although combining additional sensors improves the accuracy of visual localization systems, the systems' applications are also limited to scenarios that include these additional sensors. Furthermore, the process of multi-sensor calibration and combination can introduce additional implementation challenges. In contrast to this kind of method, our method eliminates the influence of dynamic visual features without external sensors and can be extended to other kinds of cameras if the depth of visual features can be accurately obtained.

2.3 Methods based on deep learning technologies

More than that, with the development of deep neural networks, more and more learning-based methods are introduced into SLAM systems to improve their performance [20, 21]; several SLAM systems utilize semantic information to detect pre-trained dynamic objects and exclude visual features extracted from the moving objects. Yuan et al. [22] presented SaD-SLAM in which semantic and depth information is used to distinguish visual features extracted from static, movable, and moving objects. Bescos et al. [23] employed Mask R-CNN [24] to perform instance segmentation and utilize visual features to track dynamic objects; the trajectories of both dynamic objects and camera are tightly optimized within a novel bundle adjustment proposal. Deep learning based methods can perform well in scenarios with pre-defined dynamic objects; however, in general dynamic environments, it is still necessary to combine geometry models and deep learning technologies because moving objects that are not in the training set are difficult to detect by semantic segmentation. Furthermore, because sparse visual features, rather than all pixels belonging to dynamic objects, are processed in the part of dynamic visual feature detection, our method has better real-time performance.

3 Methodology

3.1 Problem statement

For most feature-based VO and vSLAM systems, bundle adjustment (BA) is performed to optimize camera pose by minimizing the reprojection errors [6] between the 3D landmarks and their corresponding 2D visual features. The optimization function can be formulated as

$$\min_{\boldsymbol{R}_k,\boldsymbol{t}_k} \sum_{(i,k)\in\mathcal{C}} \rho\left(\left\|\boldsymbol{u}_{ik} - \pi\left(\boldsymbol{R}_k\boldsymbol{P}_i + \boldsymbol{t}_k\right)\right\|^2_{\Sigma_{ik}}\right), \tag{1}$$

where $\boldsymbol{R}_k \in \mathrm{SO}(3)$ and $\boldsymbol{t}_k \in \mathbb{R}^3$ are the orientation and position of camera for frame k, respectively. $\mathcal{C}$ is all the association set, $\boldsymbol{u}_{ik} \in \mathbb{R}^2$ denotes the i-th 2D visual feature observed by frame k. $\boldsymbol{P}_i \in \mathbb{R}^3$ is the matched 3D landmark in world coordinates. $\pi : \mathbb{R}^3 \to \mathbb{R}^2$ stands for the projection functions. Σ_{ik} is the covariance matrix and $\rho(\cdot)$ is the Huber function for eliminating the interference of outliers.

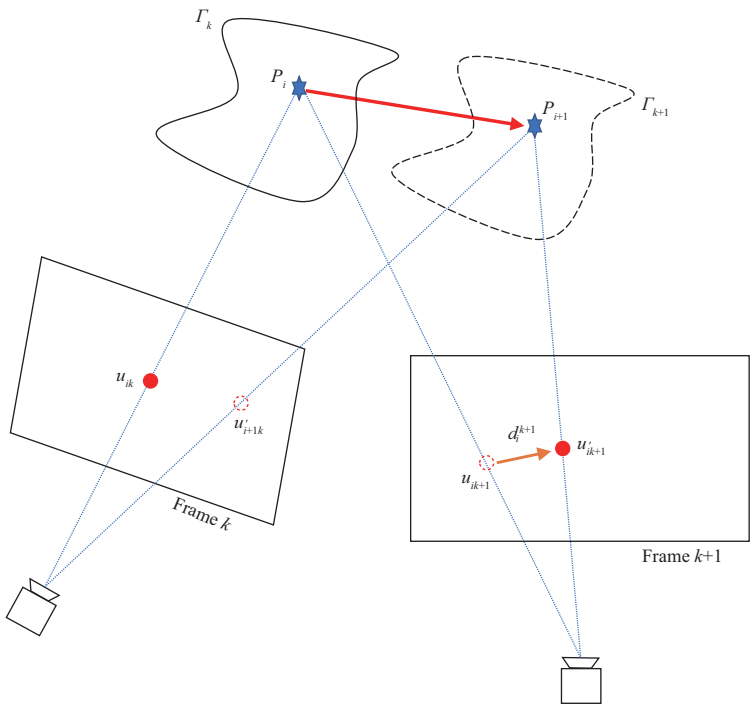


Figure 1 (Color online) BA optimization in a dynamic scenario. Γ_k is a moving object at time k and d_i^{k+1} indicates the displacement from u_{ik+1} to u'_{ik+1} in the frame $k+1$.

BA optimization performs well and the accuracy of pose estimation can be guaranteed when no moving object appears in the camera view; however, in real scenes, moving objects are always present. Figure 1 illustrates the process of BA in dynamic environments. The location of the 3D landmark $\boldsymbol{P}_i$ is changed due to the motion of moving object Γ, making the observation of $\boldsymbol{P}_i$ in frame $k+1$ move from $\boldsymbol{u}_{ik+1}$ to $\boldsymbol{u}'_{ik+1}$. If these kinds of visual features are used in BA optimization and the displacement vector of $\boldsymbol{u}_{ik}$ is represented as $\boldsymbol{d}_{ik}$, Eq. (1) can be reformulated as

$$\min_{\boldsymbol{R}_k,\boldsymbol{t}_k} \sum_{(i,k)\in\mathcal{C}} \rho\left(\|\boldsymbol{u}_{ik}+\boldsymbol{d}_{ik}-\pi\left(\boldsymbol{R}_k\boldsymbol{P}_i+\boldsymbol{t}_k\right)\|^2_{\Sigma_{ik}}\right). \tag{2}$$

In dynamic environments, the camera movement and object movement are difficult to predict, and $\boldsymbol{d}_{ik}$ in (2) is non-negligible, which lead to a poor result of BA optimization. For improving the performance of BA optimization, we should ensure all the visual features used in the optimization process are static.

3.2 Method overview

The overview of our method is illustrated in Figure 2. The inputs consist of the current RGB-D image represented by $(I_{\text{cur}}, D_{\text{cur}})$, the reference RGB-D image represented by $(I_{\text{ref}}, D_{\text{ref}})$, and a number of previous images $(I_{\text{KF}}, D_{\text{KF}})$ called keyframes, where $I_{(\cdot)} \in \mathbb{R}^2$ and $Z_{(\cdot)} \in \mathbb{R}^2$ denote the colour and depth images, respectively. First, visual features are extracted in I_{cur} and matched with the visual features in I_{ref}. Then the matched visual features in I_{cur} are grouped into N clusters by applying the k-means++ [25] to the 3D coordinates of those features having depth values. With the N clusters, we use the proposed motion consensus filtering algorithm to obtain some visual features, which are indeed static, to coarsely estimate the camera pose $[\boldsymbol{R} \mid \boldsymbol{t}]$. Subsequently, the initial camera pose is employed to track local map similar to [6], which aims at obtaining more matched features to improve the performance of system; however, tracking local map can introduce some dynamic visual features. To address the problem, we then create a sparse graph using Delaunay triangulation for all corresponding features and utilize the long-term reprojection errors, prior probability, and spatial information of visual features to distinguish dynamic and static visual features within a graph-cut optimization framework. Finally, the dynamic visual features are eliminated and the remaining static ones are used for pose refinement.

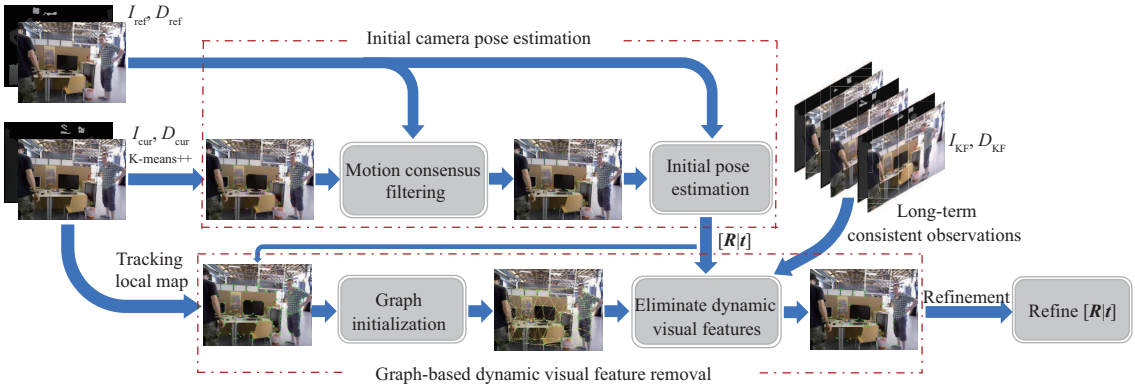


Figure 2 (Color online) Overview of the proposed method. A pair of RGB-D images $(I_{\rm cur}, D_{\rm cur})$ and $(I_{\rm ref}, D_{\rm ref})$ are processed. After the two main stages (initial camera pose estimation and graph-based dynamic visual feature removal) finishing, the static visual feature-tracking local map is determined and applied to refine the camera pose $[\boldsymbol{R} \mid \boldsymbol{t}]$.

3.3 Initial camera pose estimation

In our approach, the camera pose is estimated within a coarse-to-fine scheme. For every current frame, in the coarse stage, we perform feature extraction and match the extracted features with the visual features in the reference frame, then estimate the camera pose by applying EPnP [26] to the corresponding 3D landmarks and matched 2D feature points. Influenced by the moving objects in dynamic environments, parts of 3D to 2D data associations violate the multiple view geometry constraints [27] and the estimated camera pose is inaccurate. In order to obtain a reliable initial camera pose, the features used to estimate the camera pose are supposed to be indeed static.

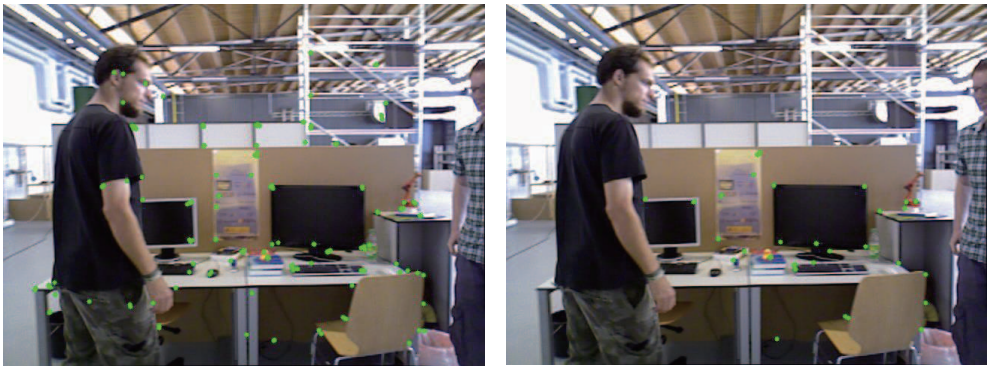
In this stage, we firstly apply k-means++ algorithm to group the matched visual features that have reliable depth values in the current frame regionally into N geometric clusters $\boldsymbol{C} = \{C_i, i = 1, \ldots, N\}$ using their corresponding 3D landmarks. The reliable depth value is associated with the property of RGB-D camera and the property is that depth measurement uncertainty of RGB-D camera is significantly positively correlated with the square of the measured value [28]. In general, a threshold is set to determine whether a depth value is reliable, which is empirically set to be 4.5 m in our experiment. We assume that visual features in the same cluster have motion consistency. This assumption is plausible because we aim at seeking a certain cluster that has the same motion property with static background, not eliminating all the dynamic visual features in this step. After obtaining N clusters, we apply EPnP to estimate the transformation $\boldsymbol{T}_{C_i}$ of cluster C_i from the coordinate of the reference frame to the current frame, where $\boldsymbol{T}_{C_i} = [\boldsymbol{R}_{C_i} | \boldsymbol{t}_{C_i}]$. Moreover, the number of clusters needs to be considered. Too many leads to a small number of visual features in each cluster, which is likely to influence the accuracy of $\boldsymbol{T}_{C_i}$. Conversely, if few clusters are retrieved, the visual features belonging to the same cluster may have different motion properties. In our experiment, we empirically set N as 5.

As only the matched visual features with reliable depth values in the current frame are used for clustering, there must be some matched visual features left having unreliable depth values. We define these left 2D visual features with their corresponding 2D visual features in the reference frame as $\mathcal{X}_{\mathcal{B}}$, $\mathcal{X}_{\mathcal{B}} = \{(\boldsymbol{u}_{ir}, \boldsymbol{u}_{ic}) \mid i = 1, \ldots, n\}$, where $\boldsymbol{u}_{ir}, \boldsymbol{u}_{ic} \in \mathbb{R}^2$ denote the i-th 2D visual feature observed in the reference and current frame, respectively. Given the fact [29] that in indoor scenarios, distant observations are more likely to be extracted from the static components (furniture, ceiling, wall, etc.) and moving objects which are far distant from the camera can be regarded as approximately static ones across a very short time span. Combined with the mentioned property of RGB-D camera, we assume that visual features with unreliable depth values are almost from the fixed elements of the surroundings. Then a voting process is applied to determine which cluster has the most similar motion to the static background. The voting scheme is inspired by epipolar geometry constraints [27], associated with the fundamental matrix $\boldsymbol{F}$, which indicates motion of moving object and is defined as

$$\boldsymbol{F} = \boldsymbol{K}^{-\mathrm{T}} [\boldsymbol{t}]_{\times} \boldsymbol{R} \boldsymbol{K}^{-1},$$

where $\boldsymbol{K}$ represents the intrinsic camera parameter and $[\cdot]_{\times}$ is a skew-symmetric matrix defined as

$$[\boldsymbol{t}]_{\times} = \left[[0, t_3, -t_2]^{\mathrm{T}}, [-t_3, 0, t_1]^{\mathrm{T}}, [t_2, -t_1, 0]^{\mathrm{T}} \right].$$



(a) (b)

Figure 3 (Color online) Results of visual features determination generated by the proposed motion consensus filtering algorithm. The green feature points in (a) are all the features matched with the visual features in the reference frame, and those in (b) are the indeed static visual features.

According to the properties of fundamental matrix, feature points extracted from moving objects have different matrix $\boldsymbol{F}$ and their corresponding feature points do not lie on the corresponding epipolar lines. We adopt the Sampson distance [27] to measure the distance from the feature points to their corresponding epipolar line and decide how to vote. For each cluster, it receives a vote if a 2D to 2D matching pair in $\mathcal{X}_{\mathcal{B}}$ satisfies the following formula:

$$\frac{\left[(\tilde{\boldsymbol{u}}_{ir})^{\mathrm{T}}\boldsymbol{F}_{C_i}(\tilde{\boldsymbol{u}}_{ic})\right]^2}{(\boldsymbol{F}_{C_i}\tilde{\boldsymbol{u}}_{ic})_1^2+(\boldsymbol{F}_{C_g}\tilde{\boldsymbol{u}}_{ic})_2^2+(\boldsymbol{F}_{C_i}\tilde{\boldsymbol{u}}_{ir})_1^2+(\boldsymbol{F}_{C_i}\tilde{\boldsymbol{u}}_{ir})_2^2}<\sigma_d, \tag{3}$$

where $\boldsymbol{F}_{C_i}$ is the fundamental matrix of motion of cluster C_i, $(\tilde{\boldsymbol{u}}_{ir},\tilde{\boldsymbol{u}}_{ic})$ are the homogeneous coordinate form of $(\boldsymbol{u}_{ir},\boldsymbol{u}_{ic})$, $(\cdot)_k$ denotes the k-th entry of vector $(\cdot)$, σ_d is an adaptive threshold associated to the scale of $(\boldsymbol{u}_{ir},\boldsymbol{u}_{ic})$.

After obtaining the vote V_{C_i} of each cluster, we select the transform matrix $\boldsymbol{T}_{C_i}$ whose corresponding cluster has the highest vote to obtain the initial camera pose $\boldsymbol{T}_{\text{init}}$ and then iteratively estimate $\boldsymbol{T}_{\text{init}}$. For each 3D to 2D matching pair $(\boldsymbol{P}_i,\boldsymbol{u}_{ic})$ with reliable depth values, its corresponding reprojection error is $e_i=\|\boldsymbol{u}_{ic}-\pi\left(\boldsymbol{R}_{\text{init}}\boldsymbol{P}_i+\boldsymbol{t}_{\text{init}}\right)\|^2_{\boldsymbol{\Sigma}_i}$, and only the pair whose reprojection error is below a pre-defined threshold σ_r, which we empirically set as 3.944, can be added to the pair set $\mathcal{S}$ for the next EPnP iteration. $\boldsymbol{P}_i$ is 3D landmark in world coordinates generated by the i-th matched 2D visual features in the reference frame using the camera intrinsics. $\boldsymbol{\Sigma}_i=\sigma_i\boldsymbol{I}_{2\times2}$ and σ_i is the variance associated to the scale of $\boldsymbol{u}_{ic}$. $\boldsymbol{T}_{\text{init}}$ is iteratively updated using the pairs in $\mathcal{S}$ and the termination condition only considers the iteration number N_{iter} which we set as 20. An example result is shown in Figure 3, and the effectiveness of this module to our system is verified in Subsection 4.2. The detailed procedures of initial camera pose estimation are given in Algorithm 1.

3.4 Graph-based dynamic visual feature removal

For improving the robustness and accuracy of our system, local map points generated by the visual features in previous keyframes are projected to the current frame to obtain more feature matches. However, in dynamic environments, tracking local map brings some unreliable feature matches. The motion consensus filtering algorithm identifies visual features that are indeed static and provide a coarse camera pose. Thus, in this stage, we are committed to further detecting and eliminating dynamic visual features from the tracked features. The basis of our method is that visual features extracted from the static environment have consistent motion over a long time, while those extracted from moving objects have different motion patterns and can be easily determined by comparing information from multiple keyframes. Furthermore, we note that the visual features distribute regionally and the motion patterns of nearby visual features tend to be the same, which mean a visual feature near a dynamic visual feature (respectively static visual feature) is more likely to be dynamic (respectively static visual feature). Motivated by the above insights, we formulate the problem of distinguishing dynamic and static visual features as a binary classification problem, which is developed based on a graph-cut optimization framework [30].

In the first step, to represent visual feature correlations, a sparse undirected graph $\boldsymbol{G}$ is constructed based on Delaunay triangulation, in which for a given set of discrete visual features, no visual feature is

Algorithm 1 Motion consensus filtering

Input: The current RGB-D image $(I_{\text{cur}}, D_{\text{cur}})$ and the reference RGB-D image $(I_{\text{ref}}, D_{\text{ref}})$;
Output: The initial camera pose of current frame $\boldsymbol{T}_{\text{init}}$;
Extract visual features from I_{cur} and match features between I_{cur} and I_{ref};
Apply k-means++ to group the matched visual features in I_{cur} into N clusters;
Estimate the motion $\boldsymbol{T}_{C_i}$ of each cluster by EPnP;
for $i = 1$ to N **do**
  for $(\boldsymbol{u}_{ir}, \boldsymbol{u}_{ic}) \in \mathcal{X}_{\mathcal{B}}$ **do**
    if $(\boldsymbol{u}_{ir}, \boldsymbol{u}_{ic}, \boldsymbol{T}_{C_i})$ satisfies (3) **then**
      $V_{C_i} \leftarrow V_{C_i} + 1$;
    end if
  end for
end for
Find the highest V_{C_i} and obtain $\boldsymbol{T}_{\text{init}}$ via its corresponding motion $\boldsymbol{T}_{C_i}$;
iterations $\leftarrow 1$;
repeat
  $\mathcal{S} \leftarrow \emptyset$;
  for each matched pair $(\boldsymbol{P}_i, \boldsymbol{u}_{ic})$ **do**
    if $\|\boldsymbol{u}_{ic} - \pi(\boldsymbol{R}_{\text{init}}\boldsymbol{P}_i + \boldsymbol{t}_{\text{init}})\|^2_{\boldsymbol{\Sigma}_i} < \sigma_r$ **then**
      Add $(\boldsymbol{P}_i, \boldsymbol{u}_{ic})$ to $\mathcal{S}$;
    end if
  end for
  Estimate $\boldsymbol{T}_{\text{new}}$ using the 3D to 2D matching pair in $\mathcal{S}$ by EPnP;
  $\boldsymbol{T}_{\text{init}} \leftarrow \boldsymbol{T}_{\text{new}}$;
  iterations $\leftarrow$ iterations $+ 1$;
until iterations $> N_{\text{iter}}$;
return $\boldsymbol{T}_{\text{init}}$.

inside the circumcircle of any triangle, as shown in Figure 4(a). Each vertex in $\boldsymbol{G}$ denotes a matched visual feature in the current frame. The adjacent visual features are connected in the constructed graph and this kind of structure is the basis of binary label assignment.

The next step is label assignment. Let $\mathcal{L} = \left\{l_1, l_2, \ldots, l_{|\mathcal{L}|}\right\}$ denote the label set whose entry specifies assignment to each vertex in $\boldsymbol{G}$, $|\mathcal{L}|$ is the number of the matched visual features and $l_i \in \{0, 1\}$. For the sake of simplicity, we set static labels as 0 and dynamic labels as 1. The problem of distinguishing dynamic and static visual features is reformulated as a binary classification problem optimized by minimizing the following energy function:

$$E(\mathcal{L}) = \sum_i R_{l_i}(\boldsymbol{u}_{ic}) + \lambda \sum_{(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc}) \in \mathcal{K}} B(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc})\, \psi(l_i, l_j). \tag{4}$$

Eq. (4) combines unary energy term $R_{l_i}(\boldsymbol{u}_{ic})$ and pairwise energy term $B(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc})\, \psi(l_i, l_j)$, using λ as the proportionality parameter. $\psi(l_i, l_j) = 1$, if $l_i \neq l_j$, otherwise $\psi(l_i, l_j) = 0$. $E(\mathcal{L})$ represents the sum of all the energies and $\mathcal{K}$ is the set of pairs of connected visual features in $\boldsymbol{G}$.

Unary energy term. The unary energy term decides the individual penalties for assigning visual features to dynamic or static ones. This term for visual feature $\boldsymbol{u}_{ic}$ is defined as

$$R_{l_i}(\boldsymbol{u}_{ic}) = -\log^{l_i}\left(p_i^d\right) \log^{1-l_i}\left(p_i^s\right), \tag{5}$$

where p_i^d and p_i^s denote dynamic and static probability of visual feature $\boldsymbol{u}_{ic}$, respectively. In our experiment, $p_i^d = 1 - p_i^s$. p_i^s contains two sources of information and is defined as $p_i^s \propto p_i^l \cdot p_i^p$.

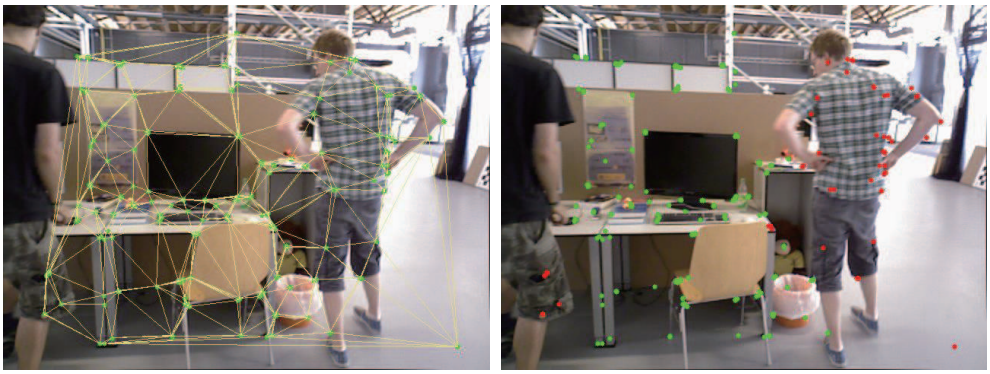
The first information p_i^l emphasizes the long-term observations by assigning a high static probability to the i-th visual feature with low reprojection error, which is described as

$$p_i^l \propto \frac{1}{\sqrt{|\boldsymbol{\Sigma}_i|}} \exp\left(-\boldsymbol{e}_{\text{proj}}(\boldsymbol{u}_{ic})\right),$$

where $\boldsymbol{e}_{\text{proj}}(\boldsymbol{u}_{ic})$ is the long-term reprojection error by averaging all the reprojection errors of the i-th visual feature, and is expressed as

$$\boldsymbol{e}_{\text{proj}}(\boldsymbol{u}_{ic}) = \frac{1}{N_i} \sum_{k=1}^{N_i} \left\| \boldsymbol{u}_i^k - \pi\left(\boldsymbol{T}_w^k {\boldsymbol{T}_{\text{init}}}^{-1} \pi^{-1}(\boldsymbol{u}_{ic}, z_i)\right) \right\|^2_{\boldsymbol{\Sigma}_i},$$

where $\boldsymbol{u}_i^k$ and $\boldsymbol{T}_w^k$ represent the corresponding 2D observation of the i-th visual feature in the k-th keyframe and the pose of the k-th keyframe, respectively. N_i denotes the total number of the keyframes,



(a) (b)

Figure 4 (Color online) Results of graph construction and visual features determination. A sparse graph is shown in (a), the green points and yellow lines represent the tracked visual features and edges between adjacent visual features, respectively. In (b), visual features extracted from moving objects are colored in red, and the green points represent the visual features extracted from the static objects.

where the i-th visual feature can be seen. The visual feature $\boldsymbol{u}_{ic}$ is back-projected into the 3D space in world coordinates with its depth value z_i and the initial camera pose $\boldsymbol{T}_{\text{init}}$, using the back-projection function $\pi^{-1}(\cdot)$. The inspiration of this term is that short-term analysis leads to inaccurate detection, while long-term information can be more reliable which is determined by projecting a visual feature to the keyframes where the visual feature corresponding 3D landmark can be seen for comparison.

The second information p_i^p represents the prior static probability, taking into account that distant observations are likely to be static and dynamic components of the scenarios tend to remain moving through time, which mean a visual feature has a lower prior static probability if it is close to a dynamic feature in the last frame and is not a distant observation. In our experiment, we define p_i^p as

$$p_i^p = \begin{cases} 0.9, & \text{if } z_i \geqslant z_{\max}, \\ 0.25, & \text{if } z_i < z_{\max} \text{ and } \exists m, d_{2\text{D}}(\boldsymbol{u}_{ic}, \boldsymbol{u}_{ml}) \leqslant d_{\text{th}}, \\ 0.7, & \text{otherwise}, \end{cases}$$

where $z_{\max}$ is an adaptive threshold, which is set as the mean depth of all matched visual features multiplied by two. $\boldsymbol{u}_{ml}$ corresponds to the m-th visual feature in the last frame, which is determined as a dynamic feature. $d_{2\text{D}}(\boldsymbol{u}_{ic}, \boldsymbol{u}_{ml})$ denotes the distance between $\boldsymbol{u}_{ic}$ and $\boldsymbol{u}_{ml}$ in the image plane. d_{th} is an empirical threshold that is set as 25. We think the number of visual features extracted from static objects is more than that extracted from moving objects, which means a visual feature is more likely to be static; thus the general prior static probability is set to 0.7.

Pairwise energy term. The pairwise energy term represents the spatial regularization, which aims to penalize the connected visual features in $\boldsymbol{G}$ that does not share the same label. This term is based on the idea that the nearby visual features with similar $\boldsymbol{e}_{\text{proj}}(\cdot)$ are more likely to have the same labels. The $B(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc})$ term is defined as

$$B(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc}) \propto \frac{\exp(-\omega \cdot (\boldsymbol{e}_{\text{proj}}(\boldsymbol{u}_{ic}) - \boldsymbol{e}_{\text{proj}}(\boldsymbol{u}_{jc}))^2)}{d_{3\text{D}}(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc})}, \tag{6}$$

where $d_{3\text{D}}(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc}) = \|\pi^{-1}(\boldsymbol{u}_{ic}, z_i) - \pi^{-1}(\boldsymbol{u}_{jc}, z_j)\|_2$, corresponding to the distance of 3D landmarks associated to $\boldsymbol{u}_{ic}$ and $\boldsymbol{u}_{jc}$. ω is a pre-defined constant. $B(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc})$ is large when the absolute value of difference of the average reprojection error between visual features $\boldsymbol{u}_{ic}$ and $\boldsymbol{u}_{jc}$ is small, while $B(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc})$ is close to zero when the absolute value of difference between them is very large. Furthermore, the corresponding penalty decreases as the distance $d_{3\text{D}}(\boldsymbol{u}_{ic}, \boldsymbol{u}_{jc})$ increases.

For reducing the runtimes, we use an efficient graph-cut algorithm [31], which builds two reusable breadth-first search trees, starting from source terminal and sink terminal to detect augmenting paths, solve the binary labeling problem and determine the optimal label vector $\mathcal{L}^*$. An example result is shown in Figure 4. After dynamic visual feature removal, the remaining static visual features are employed to refine pose estimation by performing BA.

Table 1 Information of the selected sequences used in our experiments

Sequence	Duration (s)	Trajectory length (m)	Person movement	Camera movement
fr3/w/xyz	28.83	5.791	Walk around an office	Move along xyz
fr3/w/rpy	30.61	2.698	Walk around an office	Rotate along rpy
fr3/w/half	35.81	7.686	Walk around an office	Move on a small halfsphere
fr3/w/static	24.83	0.282	Walk around an office	keep static manually
fr3/w/xyz/v	–	–	Walk around an office	Move along xyz
fr3/w/rpy/v	–	–	Walk around an office	Rotate along rpy
fr3/w/half/v	–	–	Walk around an office	Move on a small halfsphere
fr3/w/static/v	–	–	Walk around an office	Keep static manually
fr3/s/xyz*	42.50	5.496	Sit, talk, and gesticulate	Move along xyz
fr3/s/half*	37.15	6.503	Sit, talk, and gesticulate	Move on a small halfsphere
fr2/d/person*	142.08	17.044	Sit and interacte with the objects	Move around an office

"*" indicates that the sequences are low-dynamic sequences.
"–" indicates that the sequence has no corresponding public information.

4 Experiments

In this section, we provide an experimental evaluation to verify the effectiveness of our method and show the comprehensive performance of the proposed DFR-SLAM on the public TUM RGB-D dataset [14]. Each sequence in the dataset contains both the RGB-D images with 640×480 resolution and ground-truth trajectories. The sequences we use are recorded with a handheld Kinect camera in an office scene and belong to the dynamic objects category, in which the camera moves with different patterns (xyz, rpy, halfsphere and static), while people in the scene also walk or sit. The sequences are classified into two types based on the size of the camera view covered by dynamic parts of the environment: low-dynamic sequences and high-dynamic sequences. People sit with parts of their bodies moving in low-dynamic sequences, while people walk through an office scene in high-dynamic sequences. The localization system faces difficulties because different sequences have different characteristics. The name of each sequence, which includes the robot number, the motion of the people, and how the camera moves, describes the characteristics of the sequence. For instance, the name freiburg3_walking_rpy indicates the sequence is recorded by a robot named freiburg3 when the persons are walking around the office, and the camera is rotated along the principal axes (roll-pitch-yaw). The difficulties with freiburg3_walking_rpy are that high-dynamics cause the major portion of the camera view to be occupied by moving objects, and rotating along rpy leads to many mismatches. More detailed information of each sequence is shown in Table 1. In the implementation, for demonstrating the efficiency of our system, all the experiments are performed on a low computing power desktop PC with Intel i3-6100 CPU and 16 GB RAM.

4.1 Performance of dynamic visual feature detection

In this subsection, we validate the effectiveness of dynamic visual feature removal. Figure 5 shows eight results of dynamic visual feature detection from different dynamic sequences. The detected dynamic visual features are represented by red points, while the static visual features are represented by green ones. The results show that our method provides satisfactory performance in high-dynamic sequences. For instance, in fr3/w/static, the dynamic visual features extracted from two moving persons are all detected accurately. Despite performing well in high-dynamic scenarios, our method suffers from misclassifications caused by the presence of low-dynamic scenarios and the unavoidable mismatch of visual features. In low-dynamic scenarios, like fr3/s/xyz, some visual features extracted from the person in the plaid shirt are identified as static features since some parts of the person keep still for a long time. This type of misclassification has no significant impact on localization accuracy, because these visual features are extracted from movable but static objects that do not violate the geometry constraints. Inevitable mismatches of visual features, which can cause large reprojection errors and result in a wrong unary energy term in (6), are caused by visual descriptor performance limitations. As we can see in Figure 5(e), several visual features belonging to the static scene are identified as dynamic ones, but these visual features only occupy an extremely small proportion of all the tracked visual features. Therefore, the performance of pose estimation is guaranteed.

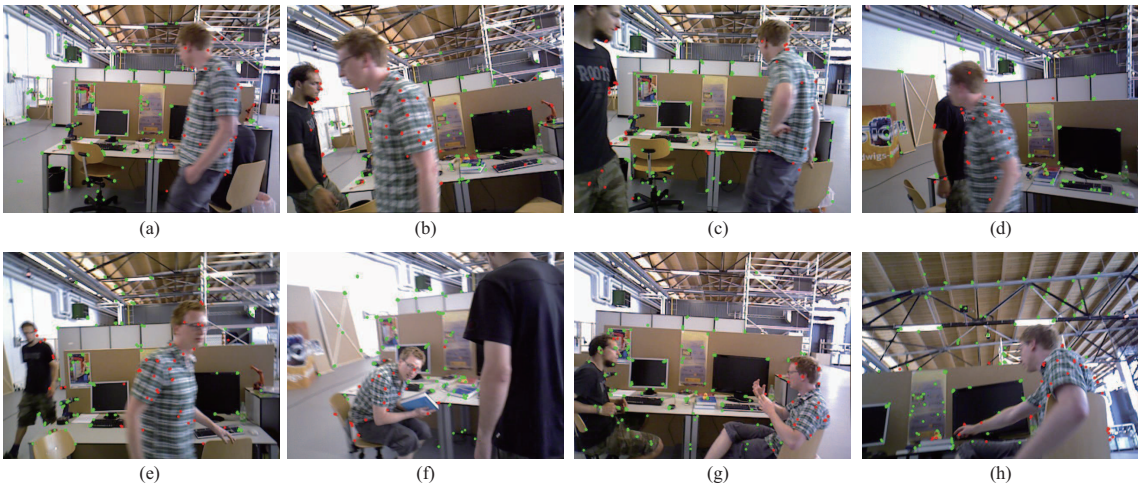


Figure 5 (Color online) Dynamic visual feature detection in multiple sequences. (a) fr3/w/xyz; (b) fr3/w/rpy; (c) fr3/w/static; (d) fr3/w/halfsphere; (e) fr3/w/rpy/v; (f) fr3/w/halfsphere/v; (g) fr3/s/xyz; (h) fr3/s/halfsphere. The colored points represent the visual features matched with the local map. The red points are determined as belonging to moving objects and the green ones belong to the static scenes. The sequences (g) and (h) are low-dynamic sequences, in which the persons are sitting and part of their bodies are moving.

Table 2 Comparison of the absolute trajectory error (ATE) for the translation part between original ORB-SLAM2 and DFR-SLAM

Sequence	ORB-SLAM2 (m)				DFR-SLAM (ours) (m)				Improvements (%)			
	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.
fr3/w/xyz	0.7616	0.6186	0.5166	0.4442	**0.0140**	0.0120	0.0104	**0.0071**	**98.16**	98.06	97.99	**98.40**
fr3/w/rpy	0.7495	0.6504	0.6340	0.3723	**0.0398**	0.0301	0.0214	**0.0261**	**94.69**	95.37	96.62	**92.99**
fr3/w/half	0.6798	0.5806	0.3935	0.3536	**0.0367**	0.0311	0.0250	**0.0193**	**94.60**	94.64	93.64	**94.54**
fr3/w/static	0.4474	0.4299	0.4030	0.1238	**0.0076**	0.0069	0.0064	**0.0033**	**98.30**	98.40	98.41	**97.33**
fr3/w/xyz/v	1.5339	1.3919	1.2256	0.6446	**0.0117**	0.0105	0.0098	**0.0052**	**99.24**	99.25	99.20	**99.19**
fr3/w/rpy/v	0.6556	0.5141	0.3671	0.4068	**0.0310**	0.0249	0.0187	**0.0185**	**95.27**	95.16	94.91	**95.45**
fr3/w/half/v	0.6885	0.5745	0.4781	0.3795	**0.0304**	0.0251	0.0198	**0.0171**	**95.58**	95.63	95.86	**95.49**
fr3/w/static/v	0.6190	0.4622	0.3414	0.4117	**0.0070**	0.0062	0.0057	**0.0032**	**98.87**	98.66	98.33	**99.22**
fr3/s/xyz*	**0.0095**	0.0084	0.0077	**0.0045**	0.0101	0.0085	0.0076	0.0055	−6.32	−1.19	1.30	−22.22
fr3/s/half*	0.0194	0.0147	0.0118	0.0127	**0.0177**	0.0156	0.0143	**0.0084**	**8.76**	−6.12	−21.18	**33.86**
fr2/d/person*	0.0715	0.0692	0.0683	0.0179	**0.0708**	0.0697	0.0699	**0.0121**	**0.97**	−0.72	−2.34	**32.40**

"*" indicates that the sequences are low-dynamic sequences.

4.2 Performance of visual localization accuracy

In this subsection, we show the quantitative and qualitative results of our DFR-SLAM system. The control variable method is used to ensure that the number of extracted visual features, keyframe selection strategy, and optimization method is all the same. On the TUM dynamic sequences, a detailed comparison of original ORB-SLAM2 and DFR-SLAM is also provided. For quantitative evaluations, we employ absolute trajectory error (ATE) [14] as the evaluation metric. Table 2 presents the evaluation results. In our experiment, the root mean square error (RMSE) and standard deviation (S.D.) values are more concerned and highlighted because they are better suited for indicating the system's robustness and stability. Furthermore, the quantitative results of our system for the ORB-SLAM2 improvements are reported to demonstrate the efficacy of our method. The improvement values brought by DFR-SLAM are calculated as

$$\eta = \frac{\alpha - \beta}{\alpha} \times 100\%,$$

where η indicates the RMSE improvement, and α and β denote the RMSE value obtained from ORB-SLAM2 and DFR-SLAM. The qualitative evaluation results are shown in Figure 6, which display ATE plots of ORB-SLAM2, our system without the module of initial camera pose estimation, and DFR-SLAM to show the improvements of localization accuracy.

As shown in Table 2, DFR-SLAM achieves tremendous improvements in the high-dynamic sequences

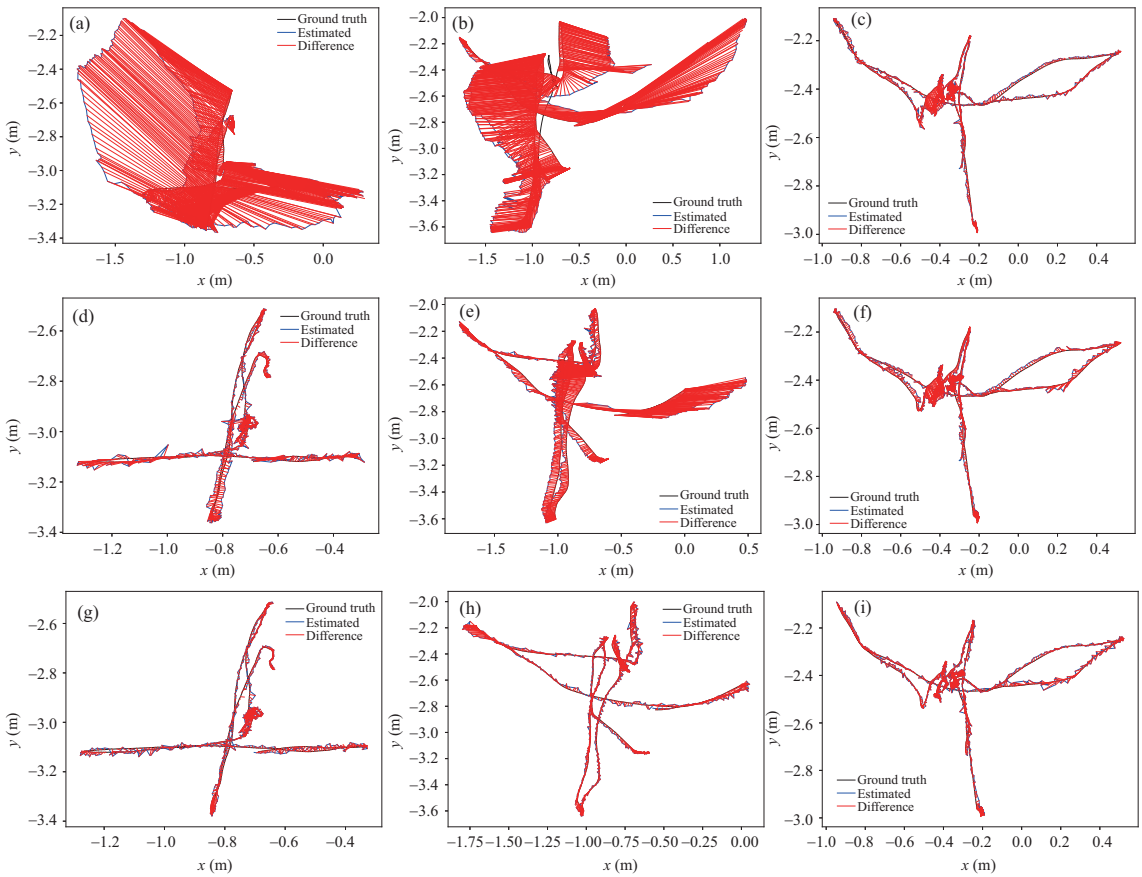


Figure 6 (Color online) Estimated trajectories of sequences fr3/w/xyz ((a), (d), (g)), fr3/w/halfsphere/v ((b), (e), (h)), and fr3/s/halfsphere ((c), (f), (i)). Blue trajectories in (a)–(c) are generated by ORB-SLAM2, those in (d)–(f) correspond to the results without the motion consensus filtering algorithm, and those in (g)–(i) are the results achieved by our DFR-SLAM. The differences in red for each subfigure represent ATE.

in terms of localization accuracy. Taking RMSE and S.D. as the standard, the average improvement values are all over 92% for the high-dynamic sequences. Note that, our system brings more improvements in the xyz and static sequences than those in the rpy and halfsphere sequences. The reason for this case is that in the xyz and static sequences, the rotation components of camera movements are less, the estimated initial camera pose could be more accurate, which is important to decide the individual penalties in our graph-cut optimization framework. In the low-dynamic sequences, our approach only provides slightly better results, and in fr3/s/xyz the performance is even degraded. This is because the visual features extracted from moving objects in low-dynamic sequences are easily identified and discarded using RANSAC and the robust cost functions used in ORB-SLAM2. Furthermore, our method considers the prior probability and spatial continuity, which may incorrectly classify some static visual features as dynamic in low-dynamic sequences, causing the system's accuracy to slightly degrade.

The qualitative results are presented by the selected ATE plots in Figure 6, which intuitively show the localization accuracy through a comparison between the estimated and ground-truth trajectories. In each subfigure, the estimated and ground-truth trajectories are shown; the red segments indicate estimation errors between the ground-truth and the estimated trajectories. The trajectory results achieved by ORB-SLAM2 are shown in the first row of Figure 6, while the bottom ones correspond to the proposed DFR-SLAM trajectory result. Notably, a comparison of original ORB-SLAM and DFR-SLAM reveals that DFR-SLAM achieves a more comprehensive performance than ORB-SLAM2 in all shown sequences, and especially in high-dynamic sequences, where our method outperforms ORB-SLAM2 significantly.

Although the module of graph-based dynamic visual feature removal is the core of our method, the initial camera pose estimation is indispensable to our system. In Figure 6, the middle column shows the ATE plots generated by our system without the module of the initial camera pose estimation, whose initial camera pose is estimated by velocity prediction. The system without the module of initial camera pose estimation has larger errors when compared with the ATE plots in Figure 6(g)–(i) generated by

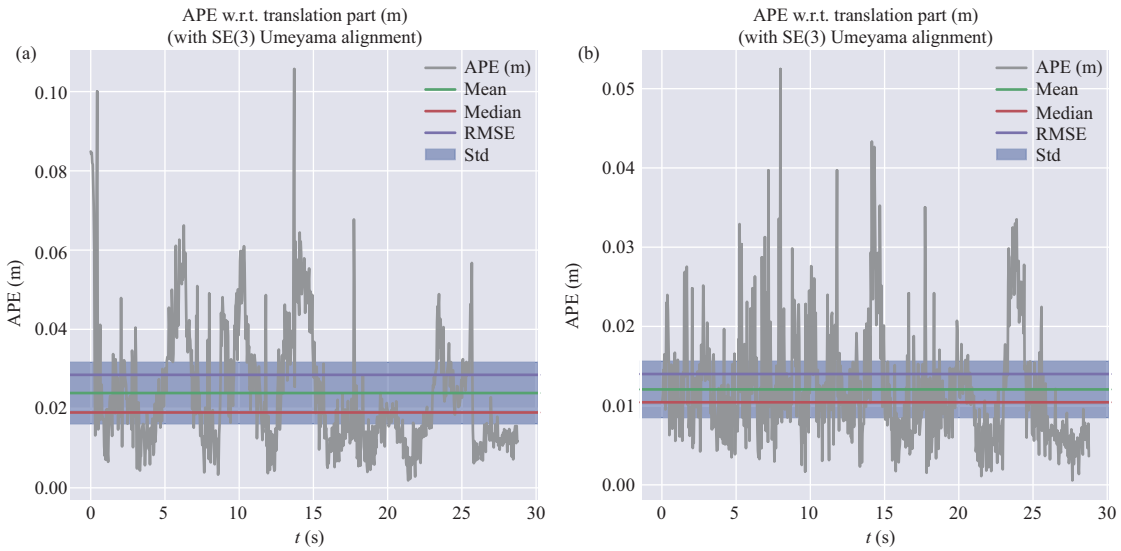


Figure 7 (Color online) Quantified results of localization accuracy in sequence fr3/w/xyz. (a) The result without the proposed motion consensus filtering algorithm; (b) the result of the proposed DFR-SLAM.

Table 3 Comparison of the RMSE of ATE metric of the proposed DFR-SLAM and the other three systems for dynamic environments (unit: m)

Sequence	Translational RMSE of trajectory alignment			
	SPW-SLAM	DSLAM	SaD-SLAM	Ours
fr3/w/xyz	0.0601	0.0874	0.0167	**0.0140**
fr3/w/rpy	0.1791	0.1608	**0.0318**	0.0398
fr3/w/half	0.0489	0.0354	**0.0257**	0.0367
fr3/w/static	0.0261	0.0108	0.0166	**0.0076**
fr3/s/xyz	0.0397	**0.0091**	0.0124	0.0101
fr3/s/rpy	–	0.0225	0.0288	**0.0215**
fr3/s/half	0.0432	0.0235	**0.0151**	0.0177
fr3/s/static	–	0.0096	**0.0060**	**0.0060**

"–" indicates that the result is not reported in the original paper.

the DFR-SLAM system, demonstrating the effectiveness of the module of initial camera pose estimation. This occurs because, in the absence of a reliable initial camera pose, the long-term reprojection error in our graph-cut optimization framework can be significantly inaccurate, and the dynamic and static probabilities of visual features can be incorrect. Besides, as shown in Figure 7, without the module of initial camera pose estimation, the APEs are significantly greater in the selected sequence fr3/w/xyz, and the error fluctuation shows that the module of initial camera pose estimation increases the stability of DFR-SLAM a lot.

4.3 Comparison of localization accuracy with the state-of-the-art

In this subsection, we compare our DFR-SLAM with three other closely related systems that all consider the influence of moving objects: SPW-SLAM [15], DSLAM [17], and SaD-SLAM [22]. Recent systems based on frame-to-keyframe registration and point correlations, respectively, are SPW-SLAM and DSLAM. To address dynamic scene content, SaD-SLAM employs deep learning techniques that combine multiple view geometry cues. The results of the three systems for evaluation are those reported in their papers, and the performance comparison is shown in Table 3.

The results show that DFR-SLAM achieves significantly lower RMSE values of ATE than those provided by SPW-SLAM in the tested dynamic sequences. This is due to the fact that our method makes extensive use of long-term observations, prior probability, and spatial coherence. Furthermore, our DFR-SLAM outperforms DSLAM slightly in low-dynamic sequences and significantly in high-dynamic sequences that ignore spatial coherence. Furthermore, our system performs competitively compared with SaD-SLAM because, in scenarios with people sitting, the majority of their body remains stationary, with only their heads and hands moving. Deep learning-based methods could identify static but movable image

Table 4 Analysis of runtime for each step of the dynamic visual feature removal

	Module	Medium (ms)	Mean (ms)	Std (ms)
Dynamic visual feature removal	Initial camera pose estimation	4.58	6.329	3.858
	Building the graph	6.031	6.592	5.289
	Segmenting the graph	5.526	7.249	6.046
	Total	18.387	20.17	13.361

Table 5 Comparison of average runtime of different systems in dynamic environments

	System	Mean (ms)
Static world assumption	ORB-SLAM2	45.75
Deep learning technologies	DynaSLAM*	421.13
	DS-SLAM*	59.4
Dense motion removal	SFVO	80
	BAMVO	143.85
Sparse visual features removal	DSLAM	30.65
	Ours	48.21

"*" indicates that the system needs GPU acceleration.

regions as dynamic objects and ignore them, resulting in a big loss of useful information and decreasing accuracy.

4.4 Analysis of system efficiency

In this subsection, we evaluate the efficiency of our proposal, and the statistics of the runtimes for different stages of the dynamic visual feature removal module are presented in Table 4. The sequence fr3/s/xyz is chosen to evaluate the efficiency of our proposal because the pose of the camera changes over time, and there are some moving objects in the sequence. The results in Table 4 show the main cost of the runtime is the step of segmenting the graph, which is influenced by the number of visual features. The more visual features there are, the more time it takes to segment the graph. In order to maximize efficiency and localization accuracy, we limit the number of visual features in our tracking step to 80–120. The whole procedure of dynamic visual feature removal needs 20.17 ms on average without GPU acceleration, which implies the proposed DFR-SLAM satisfies the needs of real-time applications.

For further evaluating the real-time performance of DFR-SLAM, a comparison with other systems on the fr3/w/static sequence is conducted and Table 5 presents the results. Compared with ORB-SLAM2, the average runtime of DFR-SLAM just increases a little. The reason for this is that dynamic visual features have been removed from DFR-SLAM, resulting in a reduction in the number of visual features used in BA. For DynaSLAM [32] and DS-SLAM [10], which are based on deep learning technologies and perform well in high-dynamic sequences, such methods are highly computationally expensive on the learning part and need the help of edge inference computers to achieve real-time localization. With regard to the time consumption of methods based on motion consistency, SFVO [29] and BAMVO [33] utilize dense operation leading to higher runtime even though the image they use is already downsampled at QVGA resolution. Among these systems, DSLAM and DFR-SLAM perform better because of sparse visual features, rather than all pixels that belong to dynamic objects, are processed in the part of dynamic visual features detection.

5 Conclusion

We have provided a real-time and accurate RGB-D SLAM system named DFR-SLAM that adds an efficient dynamic visual feature removal method, which makes the proposed system adapt to dynamic environments. To exclude dynamic visual features, the method is used in a coarse-to-fine scheme that employs the proposed motion consensus filtering algorithm and a graph-cut framework that combines long-term observations, prior probability, and spatial coherence. The experiments on the public dataset demonstrate our system's effectiveness and accuracy in both high-dynamic and low-dynamic scenes.

At the coarse stage, the initial camera pose estimation limits the ability of our system to accurately estimate the camera pose in a slightly dynamic environment. In the future, we plan to incorporate IMU factors into our framework in order to obtain a more accurate initial camera pose, which is critical in

our method. We could also tightly couple IMU measurements and feature observations to improve the accuracy and robustness of pose estimation in various dynamic scenarios.

Acknowledgements This work was supported in part by National Natural Science Foundation of China (Grant Nos. 61922076, 61873252).

References

1 Bresson G, Alsayed Z, Yu L, et al. Simultaneous localization and mapping: a survey of current trends in autonomous driving. IEEE Trans Intell Veh, 2017, 2: 194–220
2 Qin T, Chen T, Chen Y, et al. AVP-SLAM: semantic visual mapping and localization for autonomous vehicles in the parking lot. In: Proceedings of International Conference on Intelligent Robots and Systems, Las Vegas, 2020. 5939–5945
3 Fang B, Mei G, Yuan X, et al. Visual SLAM for robot navigation in healthcare facility. Pattern Recogn, 2021, 113: 107822
4 Zhao C H, Fan B, Hu J W, et al. Homography-based camera pose estimation with known gravity direction for UAV navigation. Sci China Inf Sci, 2021, 64: 112204
5 Yang D F, Sun F C, Wang S C, et al. Simultaneous estimation of ego-motion and vehicle distance by using a monocular camera. Sci China Inf Sci, 2014, 57: 052205
6 Mur-Artal R, Tardos J D. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans Robot, 2017, 33: 1255–1262
7 Lv W J, Kang Y, Qin J H. FVO: floor vision aided odometry. Sci China Inf Sci, 2019, 62: 012202
8 Yunus R, Li Y, Tombari F. ManhattanSLAM: robust planar tracking and mapping leveraging mixture of manhattan frames. 2021. ArXiv:2103.15068
9 Fischler M A, Bolles R C. Random sample consensus. Commun ACM, 1981, 24: 381–395
10 Yu C, Liu Z, Liu X J, et al. DS-SLAM: a semantic visual SLAM towards dynamic environments. In: Proceedings of International Conference on Intelligent Robots and Systems, Madrid, 2018. 1168–1174
11 Li A, Wang J, Xu M, et al. DP-SLAM: a visual SLAM with moving probability towards dynamic environments. Inf Sci, 2021, 556: 128–142
12 Cheng J, Zhang H, Meng M Q H. Improving visual localization accuracy in dynamic environments based on dynamic region removal. IEEE Trans Automat Sci Eng, 2020, 17: 1585–1596
13 Barber C B, Dobkin D P, Huhdanpaa H. The quickhull algorithm for convex hulls. ACM Trans Math Softw, 1996, 22: 469–483
14 Sturm J, Engelhard N, Endres F, et al. A benchmark for the evaluation of RGB-D SLAM systems. In: Proceedings of International Conference on Intelligent Robots and Systems, Vilamoura, 2012. 573–580
15 Li S, Lee D. RGB-D SLAM in dynamic environments using static point weighting. IEEE Robot Autom Lett, 2017, 2: 2263–2270
16 Cheng J, Wang C, Meng M Q H. Robust visual localization in dynamic environments based on sparse motion removal. IEEE Trans Automat Sci Eng, 2019, 17: 658–669
17 Dai W, Zhang Y, Li P, et al. RGB-D SLAM in dynamic environments using point correlations. IEEE Trans Pattern Anal Mach Intell, 2022, 44: 373–389
18 Kim D H, Han S B, Kim J H. Visual odometry algorithm using an RGB-D sensor and IMU in a highly dynamic environment. In: Robot Intelligence Technology and Applications 3. Cham: Springer, 2015. 11–26
19 Yang D, Bi S, Wang W, et al. DRE-SLAM: dynamic RGB-D encoder SLAM for a differential-drive robot. Remote Sens, 2019, 11: 380
20 Hyun D, Park C, Yang M C, et al. Target-aware convolutional neural network for target-level sentiment analysis. Inf Sci, 2019, 491: 166–178
21 Bruno H M S, Colombini E L. LIFT-SLAM: a deep-learning feature-based monocular visual SLAM method. Neurocomputing, 2021, 455: 97–110
22 Yuan X, Chen S. SaD-SLAM: a visual SLAM based on semantic and depth information. In: Proceedings of International Conference on Intelligent Robots and Systems, Las Vegas, 2020. 4930–4935
23 Bescos B, Campos C, Tardos J D, et al. DynaSLAM II: tightly-coupled multi-object tracking and SLAM. IEEE Robot Autom Lett, 2021, 6: 5191–5198
24 He K, Gkioxari G, Dollár P, et al. Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, Venice, 2017. 2961–2969
25 Arthur D, Vassilvitskii S. k-means++: the advantages of careful seeding. In: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, 2007
26 Lepetit V, Moreno-Noguer F, Fua P. EPnP: an accurate $O(n)$ solution to the PnP problem. Int J Comput Vis, 2009, 81: 155–166
27 Moulon P, Monasse P, Perrot R, et al. OpenMVG: open multiple view geometry. In: Proceedings of International Workshop on Reproducible Research in Pattern Recognition, 2016. 60–74
28 Khoshelham K, Elberink S O. Accuracy and resolution of Kinect depth data for indoor mapping applications. Sensors, 2012, 12: 1437–1454
29 Jaimez M, Kerl C, Gonzalez-Jimenez J, et al. Fast odometry and scene flow from RGB-D cameras based on geometric clustering. In: Proceedings of the IEEE International Conference on Robotics and Automation, Singapore, 2017. 3992–3999
30 Boykov Y Y, Jolly M P. Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In: Proceedings of the IEEE international conference on computer vision, Vancouver, 2001. 105–112
31 Boykov Y, Kolmogorov V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans Pattern Anal Machine Intell, 2004, 26: 1124–1137
32 Bescos B, Facil J M, Civera J, et al. DynaSLAM: tracking, mapping, and inpainting in dynamic scenes. IEEE Robot Autom Lett, 2018, 3: 4076–4083
33 Kim D H, Kim J H. Effective background model-based RGB-D dense visual odometry in a dynamic environment. IEEE Trans Robot, 2016, 32: 1565–1573