 doi:10.20944/preprints202001.0123.v1

Article

DM-SLAM: Monocular SLAM in Dynamic Environments

Xiaoyun Lu[1,2], Hu Wang[1,2]*, Shuming Tang[2,3], ,Huimin Huang[1,2], and Chuang Li[1,2]

[1] Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China
[2] University of Chinese Academy of Sciences, Beijing 100049, China
[3] Institute of Automation, Chinese Academy of Sciences, University of Chinese Academy of Sciences, Beijing 100190, China
* Correspondence: wanghu@opt.ac.cn;

Abstract: Many classic visual monocular SLAM systems have been developed over the past decades, however, most of them will fail when dynamic scenarios dominate. DM-SLAM is proposed for handling dynamic objects in environments based on ORB-SLAM. The article mainly concentrates on two aspects. Firstly, DLRSAC is proposed to extract static features from the dynamic scene based on awareness of nature difference between motion and static, which is integrated into initialization of DM-SLAM. Secondly, we design candidate map points selection mechanism based on neighborhood mutual exclusion to balance the accuracy of tracking camera pose and system robustness in motion scenes. Finally, we conduct experiments in the public dataset and compare DM-SLAM with ORB-SLAM. The experiments verify the superiority of the DM-SLAM.

Keywords: static features extraction; dynamic environments; 3D reconstruction; monocular SLAM

1. Introduction

Estimating the pose of the camera and reconstructing the three-dimensional structure of the environments in the field of view of continuous frames is the main task of visual SLAM. Visual-based autonomous robot acquires camera poses and environments information to perform more complex tasks such as robot navigation, human-robot interaction and path planning.

Monocular, binocular and RGB-D cameras can be used to implement visual SLAM. Due to acquiring depth of environments directly, binocular-based and RGB-D-based SLAM are more robust than monocular SLAM. Although monocular cameras have their own inherent limitations, such as unobservability of the scale and state initialization, but respect to size, power and cost [1], monocular cameras have broad application scenarios.

However, despite some remarkable results such as ORB-SLAM[2], LSD[3], DSO[4] in visual SLAM , most approaches, whether it is feature-based, direct or semi-dense one [5], [6], assume that static targets or backgrounds take dominate in the scene [1]. For feature-based SLAM, the assumption can be interpreted as the feature points on the static scene take dominate in all feature points in the image. These systems generally adopt RANSAC [8] and robust kernel algorithms, which perform well when the dynamic feature points are relatively few, but as the proportion of static feature points gradually decreases, the computation required for obtaining camera poses with higher confidence increases dramatically. Such assumption greatly limits applications of visual SLAM in actual scenarios.

 doi:10.20944/preprints202001.0123.v1

The existence of dynamic objects will not only affect the tracking accuracy if the matches belonged to dynamic objects are used in tracking algorithms, but also cause system to establish the unprecise map including dynamic objects and reduce the accuracy of loop closure detection, which is used to reduce the cumulative error. For many long-term applications such as virtual reality, augmented reality, medical imaging, precise and stable map from previous runs which can be reused is significant, if dynamic objects can be detected and processed appropriately [7].

The premise of solving the camera pose is to get the static feature set, while camera pose is required to filter the static features from the image features of the noise features, mismatches and dynamic features. When the scene does not satisfy the main assumption that static features occupy, how to get accurate camera pose seems to be problem with "chicken-and-egg" characteristic.

Tan et al. [9] propose prior-based adaptive RANSAC algorithm to get camera motion model, and which will estimate the probability that current model is camera motion model by the degree of distribution of points within the model for each iteration. Sun et al. [10] warp the last frame with the perspective matric and subtract the current frame with the warped frame to distinguish dynamic areas. The perspective matrix is constructed by computing the optimal homography with RANSAC algorithm, where camera ego-motions are represented as 2-D perspective matrices. Above methods do reconstruction before motion segmentation, while the results are greatly affected by the initial reconstruction result.

There are also some methods of reconstruction after motion segmentation. Kitt et al. [11] uses the classifier trained in advance to classify the feature points as dynamic or static, but it cannot be used to explore the unknown environment for the classifier is limited by training data. Berta [1] et al. remove features on the movable objects in the prior-dynamic environments by CNN, then use multi-view geometry to eliminate moving objects that are not set to movable. Chao Yu [12] adds semantic segmentation thread and dense semantic map creation thread based on ORB-SLAM [2]. The SegNet is used to segment the image semantically and the result of SegNet is given to tracking thread to eliminate the features on movable objects. After solving pose of camera with the remaining features, system determines whether the movable objects are really moving according to whether the features on the movable object satisfy the solved camera model.

In addition to the above solution ideas, factorization has an elegant mathematical formulation and can solve the problem of segmentation and reconstruction simultaneously. Ref [13], [14] can separate moving objects while objects should be all rigid and enough frames have been taken into account, they usually require that the same feature set is available throughout the certain chosen frames. Ref [15] utilizes two properties of trajectory data including geometric constraint and locality to segment a wide range of motions including independent, articulated, rigid, non-rigid, degenerate, non-degenerate or any combination of them.

At this point, we propose distribution and local-based RANSAC algorithm (DLRSAC) to address this problem. We integrate DLRSAC into the ORB-SLAM monocular system, where DLRSAC is used to initialize map through extracting static features. In tracking process, we design map updating mechanism to deal with scenarios including non-rigid dynamic objects or objects whose movement is difficult to identify between frames, which will decay subsequent map expansion and camera .pose calculation.

2. Materials and Methods.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1

2.1. DLRSAC for reconstructing static map

For SLAM in dynamic scene, the key is to reduce the interference of moving objects on localization and reconstruction, the core issues are as follows:

1. The problem carries the nature of chicken-and-egg characteristic, and the nature difference between motion and static is the key to solve the problem.
2. Due to the presence of inherent sensor noise, only when the motion exceeds noise interference, motion of different moving objects and camera motion are distinguished.

The second issue can be resolved by checking in more frames, or in two frames with larger time intervals. In the DS-SLAM system to be introduced later, we designed a targeted mechanism to avoid noise interference, construct a map with high confidence and more accurate camera pose.

We believe that understanding the nature of static and motion is the key to break chicken-and-egg characteristic. In the relative motion of the camera and the environment, all motion models solved by image association in the scene can be characterized as Equation 1.

$$T=T_c \square <T_S, T_1, T_2, \ldots T_n> \tag{1}$$

□ represents the synthesis of two motions; T represents the motion of the object (including rotation, translation), T_c represents the motion of the camera, and T_s represents the motion of the static object, which is the only constant in the equation (rotation as identity matrix, translation into three-dimensional 0 vector), T_n represents the motion of the nth motion model. Due to the coupling with T_c, the value of $T_c \square T_s$ is constantly changing, T_c is indistinguishable when there is no prior information of T_c or Tn.

However, considering first issue from the agents in environment, we find a way to define static. For the agent exploring in the environment, it is necessary to provide a reference coordinate system with wide distribution and constant motion state, so that the agent can locate itself and reconstruct environment conveniently. And this is the nature difference between static and motion in our living environment. So we designed the DLRSAC based on three basic observations as follow:

- The definition of static on the distribution of inliers within model, not the numerical score of the motion model and the number of features points which motion model includes.
- The closer features are more likely to belong to the same motion model.
- Models between different moving objects shares a few inliers.

Based on above observations, our algorithm aims to obtain motion model which is solved by set of feature points geometrically as close as possible, while the final inliers within the model distribute as evenly as possible. Equation (2) as below defines our objective function:

$$\arg \min_{F\sim(x_{s1},x_{s2},\ldots x_{s8})} \frac{D(x_{s1},x_{s2}\cdots x_{s8})}{D(x_{t1},x_{t2}\cdots x_{tn})} \tag{2}$$

In (2), $(x_{s1},x_{s2}\cdots x_{s8})$ means the 8 points sampled from current frame, F means the fundamental matrix solved through eight-points algorithm by $(x_{s1},x_{s2}\cdots x_{s8})$, $(x_{t1},x_{t2}\cdots x_{tn})$ are all inliers of F. $D(\cdot)$ is the function of calculating the distribution of input variables. The specific form of $D(\cdot)$ is as Equation (3).

In (3), X is the features set, which is also the input variables of $D(\cdot)$. p_i is the feature belonging to X, $\bar{p}$ is the centroid of X and n is the number of features in X. To reduce the complexity of solving the problem, we designed the DRLRSAC algorithm to extract static features.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1

$$D(X)=\frac{1}{n}\sqrt{\sum_{p\in X}\left(p_i-\overline{p}\right)^2} \tag{3}$$

$$\overline{p}=\frac{1}{n}\sum_{p\in X}p_i \tag{4}$$

Before we describe the algorithm flow, some definitions need to be clarified in advance.

Gird: We will divide the image into $l*m$ girds of the same size.

Grid model: Fundamental matrix represents current grid solved by RANSAC [8] with the matched feature in current grid.

Couple: Refers to test how many inliers in tested grid satisfy the testing *Grid model*.

Couple Degree: The result of Couple divided by the total number of inliers in tested grid.

Coupling Grid: Testing grid or testing grid model in the process of Coupling.

Coupled Grid: Tested grid or tested grid model in the process of Coupling.

Couple Matrix: Consists of Couple Degrees. A row of elements in the *Coupling Matrix* share a common *Coupling Grid*, and the *Coupled Grid* of same column of elements is the same.

Coupled Grid set: Consists of the grids whose *Couple Degree* with *Coupling Grid* is greater than a certain threshold and *Coupling Grid* itself.

Grid model distribution: Refers to the geometric distribution of Coupled Grid set of one Grid model.

Algorithm flow is described as below：

1. Extract features in the frames used for initialization and match them.
2. Divide the image into l*m grids, assign features into grids according to whether features belong to. For each *Grid*, if the number of features falling doesn't satisfied with solution condition, we will discard it. Then, we will solve the centroid with all the feature points within these grids which pass above check.
3. For each grid, normalize all the features in the grid, and then use RANSAC to solve the *Grid model* with the normalized features.
4. Constructing *Couple Matrix* by calculating *Coupled Degree* of each grid with other grids.
5. Get *Coupled Grid set* of each *Grid model* by setting certain threshold, calculate distribution of the set using the centroids of the grids in the set.
6. Select the *Grid model* with the largest *Grid model* distribution as the camera pose, the features in grids contained in the *Coupled Grid set* of selected grid are considered to be candidate static feature points.
7. Triangulate static feature points in step 6, initialize map with successful triangulated features.
8. Use bundle adjustment to adjust 3D map points and camera pose.

2.2. Neighborhood mutual exclusion for selecitng candidate map points.

Since the newly added map points are more likely to be used for subsequent system tracking in SLAM, which will affect subsequent estimation of the camera pose and map expansion. Therefore, it is important to ensure that the newly selected map points belong to stable map, otherwise the system will fail quickly. We need to verify that it conforms to geometric constraints in more key frames, but this will slow down map expansion and reduce robustness of the system, especially in low-texture area.

The new matches introduced by the new key frame are the main source for generating new map points. When key frame K_i inserts, match is searched between features in K_i and other unmatched features in connected key frames K_c. If the match passes the following test, it will be taken as a candidate map point.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1

1. The new match satisfies the fundamental matrix of and corresponding key frame.
2. Triangular ORB pairs after checking parallax, projection error, uniform scale, and forward depth.
3. There are no alternate map points in the neighborhood of current tested feature match.

Details can be seen Algorithm 1.

Algorithm 1 Candidate map points selection mechanism

```
For new match (x-x') between optimal co-view key frames with new key frame;
    Triangulate match(x-x') to 3D point X;
    If triangulate fails, then
        Continue;
    Else check parallax, projection error, uniform scale, and forward depth;
        If fail, then
            Continue;
        Else check whether there has been new candidate map points in the
neighborhood[50*50 in our system];
            If has exist candidate map points, then
                Continue;
            Else check number of candidate map points Cn>α* Ln;
            If inequality above is false, then
                Add X as candidate map point;
            Else
                Continue;
            End if
        End if
    End if
End if
End for
```

The distribution and number of added map points are constrained by the neighborhood. The advantage of this is that the map points on the objects that become moving can be removed, and the method of neighborhood suppression is used in the process of selecting the map points. There are two advantages to this:

Due to the concentrated distribution of moving objects, the number and probability of features sampled onto moving objects are reduced by the neighborhood suppression selection.

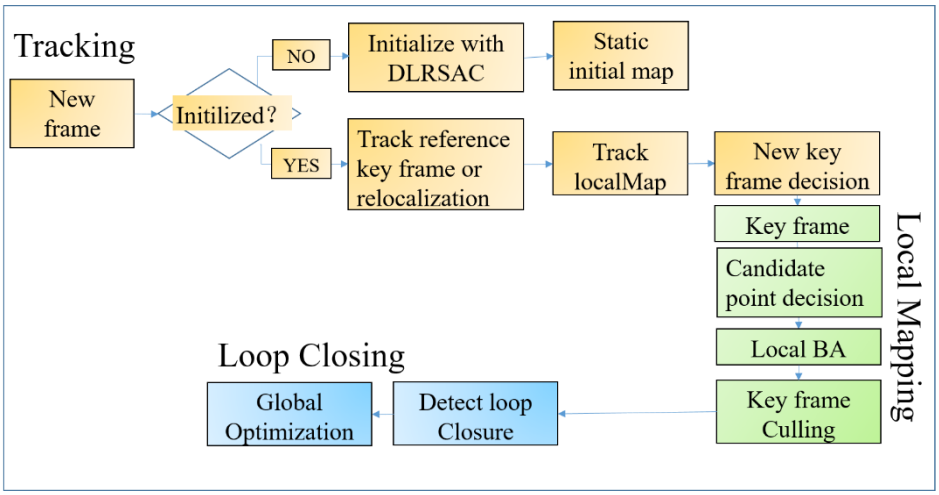
Limit the number of selected objects that become dynamic. Even if features on moving objects are selected, the impact on reconstruction is weakened.

2.3. System review

DM-SLAM will be introduced in detail in this section. ORB-SLAM has excellent and robust performance in most practical situations. So we adapted ORB-SLAM to provide basic framework for creating static map and estimating camera pose robustly. Among them, the closed loop process in the system is identical to ORB-SLAM. Figure 1 shows the framework of DM-SLAM. DLRSAC is integrated into initialization in the tracking thread to construct original map. And the expansion rules

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1

of the map are designed to weaken the effects of dynamic objects in mapping thread. Below we describe the mechanism that have been taken to improve the robustness of system motion versus ORB-SLAM.



(**a**)

Figure 1. The framework of DM-SLAM. DM-SLAM share same loop closing thread with ORB-SLAM. We adjust the process of Initialization when system begins tracking and redesign the rule of map point insertion and culling in local mapping

3.3.1. DLRSAC implementation in DM-SLAM

DLRSAC performs static semantic extraction to get static initial map with respect to ORB-SLAM. This algorithm doesn't require any prior information, while introducing the distribution degree evaluation to break the problem of "chicken-egg" circulation in dynamic scenes. The camera pose estimation can be completed more confidentially and more accurately even if static feature points in the scene don't take dominate.

Static initial maps are constantly used in camera tracking, which then affect subsequent map expansion, therefore, the accuracy of DLRSAC is much more important than the recall rate. So we improve the accuracy of the algorithm by analyzing the reason why the algorithm fails, introducing detection methods and resetting system when failure appears. Analysis of the reasons for failure is reviewed below:

1. The parallax between the two frames is so small that solving F matrix at this time is illness problem.
2. The motion in the scene is too complicated to affect the model selection in each grid, so that the motion model is finally selected.
3. The scene appears motion blur, the low-textured area, leading to few matches, causing few sufficient Grid model.

Detection methods are reviewed below:

1. The parallax between the two frames is so small that solving F matrix at this time is illness problem.
2. Verify the parallax of map points, if the mean value is too small, the algorithm is considered to fail. In order to call a large parallax, we use two frames separated by several frames instead of two consecutive frames at the time of initialization, another benefit of doing this is to make the motion and noise separable.
3. Verify that the distribution of selected Grid model, if the value is too small, the algorithm is considered to fail.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1

4. Check the two frames used for initialization, reset system if there are too few features matching.

We delay the initialization until this method pass the failure test. Figure 2 shows the static feature extraction effect of the algorithm in different motion scenarios.

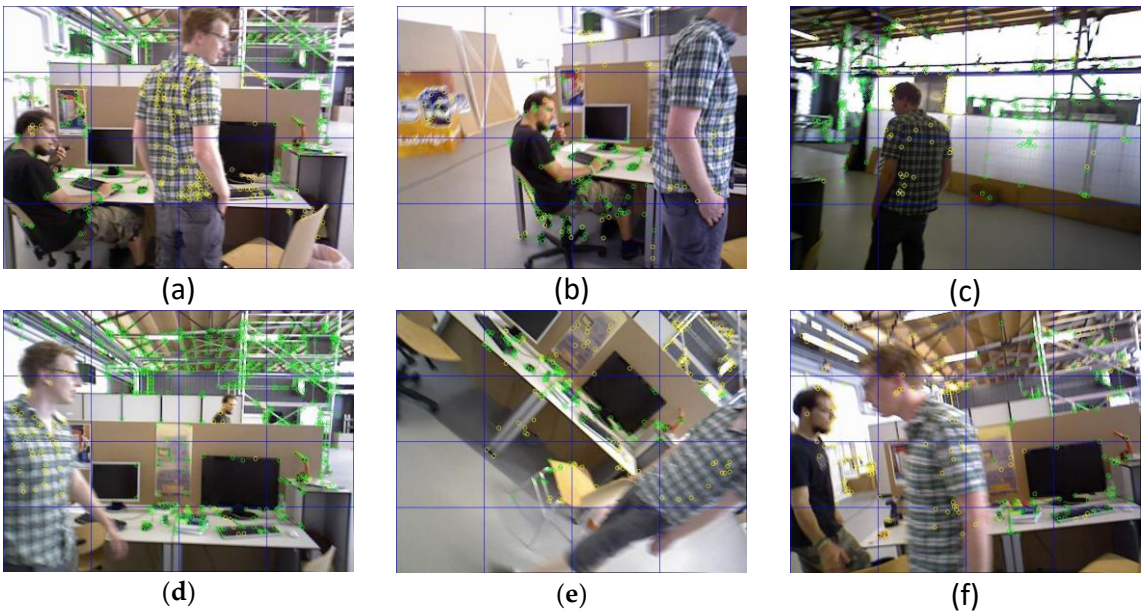


Figure 2. Static features extraction results of different frames in sequence fr3_walking_rpy. Inliers are marked with green circle, while yellow circles are considered to be outliers.

3.3.2. Tracking thread

In tracking thread of other visual SLAM systems, the motion model [16] usually is used to initial camera pose, and then outliers are eliminated in subsequent optimizations. Due to that the final pose results are more inclined to initial camera pose, rough estimate of camera pose tends to amplify the error of pose estimation especially in scenes existing non-rigid motion. First we track the matching features between current frame and current reference key frame, estimate the camera's initial pose then track the local map to get more feature matches, at last, perform camera pose optimization. This approach essentially is way of reconstruction after segmentation. The reason we can deal with dynamic objects is because the definition of the static map has been implemented in the initialization. See ORB-SLAM [16] for details on local map.

3. Results

In order to show the effectiveness of the DRLRSAC algorithm in more detail, we not only run DM-SLAM on a public data set, but also perform DRLACS experiments. Experiments are tested in dynamic environments within public TUM RGB-D dataset [17] while depth images are ignored. High dynamic fr3_walking sequences are very challenging because people in the scene sometimes walking increase the difficult of monocular SLAM. Firstly we show the static features extraction results of initialization using DLRSAC in different scenarios of fr3_walking_rpy sequence after tacking fails, and intermediate results in the tracking. Then we tested the time required by the DLRSAC algorithm and compared it with RANSAC and ARSAC. Finally, we evaluated the performance of DM-SLAM in some sequences of TUM-RGBD dataset. All experiments are performed on a computer with intel i7 CPU, and 8GB memory.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1

3.1.evalutions of DLRSAC

Unlike the data used in comparison of algorithms [9] like RANSAC, LoSAC [18], Multi-GS [19] and ARSAC [9], where the dynamic objects change significantly in geometric, the data used in the actual operation of the system is quite different. Since the camera frame rate is high enough, the change of the scene including the moving objects isn't obvious, which is very challenging for algorithms to distinguish moving objects. In our experiments we use successive frames in video sequences including our own data and TUM sequence fr3_walking_rpy.

Figure 3 displays results tested in our data, where the middle book in the source images is moving. Our data is used to compare algorithms where there are rigid-body motions in the scene. Figure 4 shows the results tested in TUM fr3_walking_rpy, which verify the performance of algorithms when there are non-rigid objects in the scene. We compared our algorithm with others including standard RANSAC [8], ARSAC [9]. We choose ARSAC instead of PARSAC because prior information of inliers ratio in the images isn't practical to obtain. As can be seen in Figure 4(e-f), most static features can be extracted with DLRSAC except for static features in below conditions:

1. The feature is so independent from other features that the number of feature points in the local grid doesn't satisfy solution condition.
2. Some static features belong to the grid in which moving features take dominate, causing that the grid isn't classified into the Coupled Grid set so all features in the grid are considered as moving features.

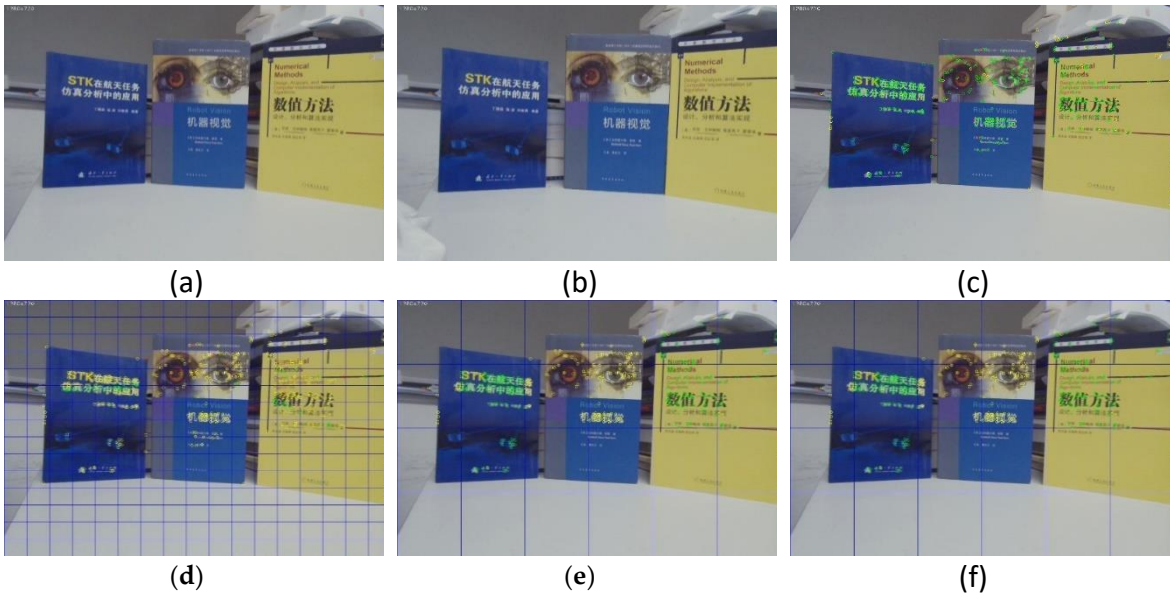
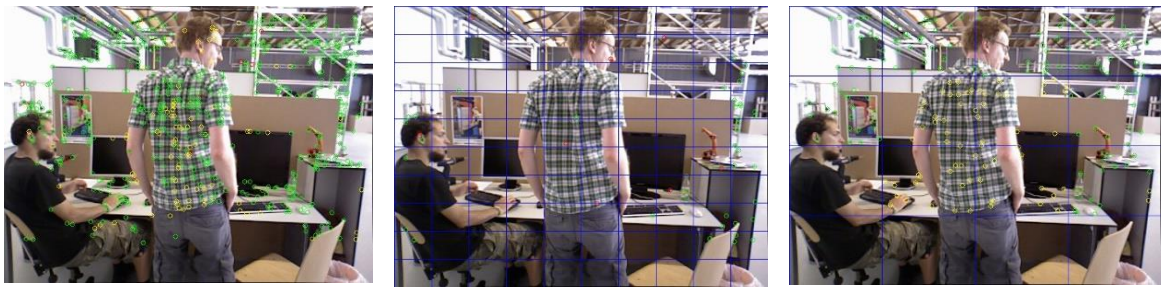


Figure 3. Real example for static extraction. (a) and (b) are source images, witnessing that the book in the middle is moving.(c) The static points recognized by RANSAC, marked with green circle. The outliers are marked with yellow circles. (d) The static points recognized by ARSAC [16]. (e) The static points recognized by our DLRSAC with the size of grid is 180*180. (f) The static points recognized by our DLRSAC with the size of grid is 240*240.

 doi:10.20944/preprints202001.0123.v1



(a) (b) (b)

Figure 4. Samples in Tum sequence fr3_walking_rpy. As Figure 3, static points are marked with green circles while others are marked with yellow circles. (a) Recognition results by RANSAC. (b) Recognition results by ARSAC. (c) Recognition results by DLRSAC.

We can see from Figure 3 (c) and Figure 4 (a) that RANSAC has almost no distinguishing ability for moving objects in consecutive frames. Although ARSAC can extract a part of static features, like Figure 3(d) and Figure 4(b) but the extracted features are much less than that of DLRSAC. We think the reason is that ARSAC only takes the degree of distribution as the objective function optimization variable.

In addition, the computational overheads of DLRSAC is a little bigger than RANSAC and ARSAC while within acceptable circumstances, as listed in Table 1 (the iteration number is set to 2000).

Table 1. The average running time of different algorithms.

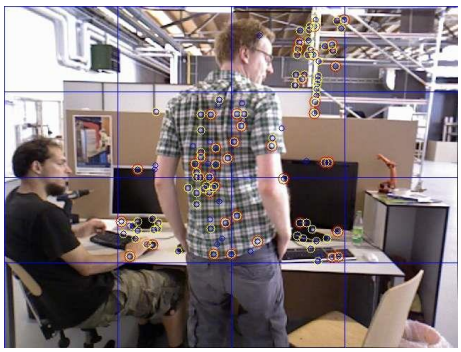
Algorithm	RANSAC	ARSAC	DLRSAC
Running time	34.6ms	36.8ms	40.1ms
Iterations	2000	2000	2000

The purpose of dividing image into grids is to reduce the complexity of motion within grid, So that the points that are sampled to solve Grid model are more likely to belong to the same motion model. As we know, the closer the features are, Under RANSAC taken to solve Grid model, the number N of iterations required to guarantee a correct solution is:

$$N = \frac{\log(1-p)}{\log(1-(1-\varepsilon)s)} \tag{5}$$

Where s is the number of data points, ε is the percentage of outliers, and p is the probability of success (confidence level) [20]. In fact, RANSAC tends to choose model that are solved by features from multiple models. Fig. 3 shows this nature. These models have stronger Coupling ability in local grid while less in global image, which help DLRSAC filter out real static features. Other than this, we find that the selected model has stabilized after several hundred iterations due to this nature. The grid selected in Figure 5 contains the moving person and the surrounding static environments, where red circles are the selected eight initial points used to solve the solution model. The eight points are both from moving and static objects, but almost all the points in the grid are their inliers.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1



(**a**)

Figure 5. Solution result in grids containing mixed model. Red circles: eight features solving Grid model; Blue circles: all matching feature points; Yellow circles: inliers of the Grid model.

3.2. Initialization and tracking performance

In sequence fr3_walking, there are two men walking around the desk, which are only movable objects causing highly dynamic scene. Sequence fr3_walking includes rpy, xyz and half sphere, which stands for four types of camera ego-motions. In order to verify the performance of DLRSAC in different motion scenarios, we randomly test our algorithm in different frames in the fr3_walking sequence, and then selected the several typical motion scenes to show performance of static features extraction of our algorithm, Figure 6 shows the results. In Figure 6(e-f), although there is certain motion blur in the environment, the algorithm still extracts most of the static features.

The results of DM-SLAM in the tracking process are shown in Figure 6. It can be seen that DM-SLAM removes most motion features during the tracking process. In the map expansion, although motion features are added to the map points occasionally, the limitation on the distribution and number of newly added map points weakened the influence of new added points on the camera pose calculation. In turn the correct camera pose assures the geometric constraint relationship, which allows the system to eliminate motion features based on the projection error of the newly added map points in subsequent frames.

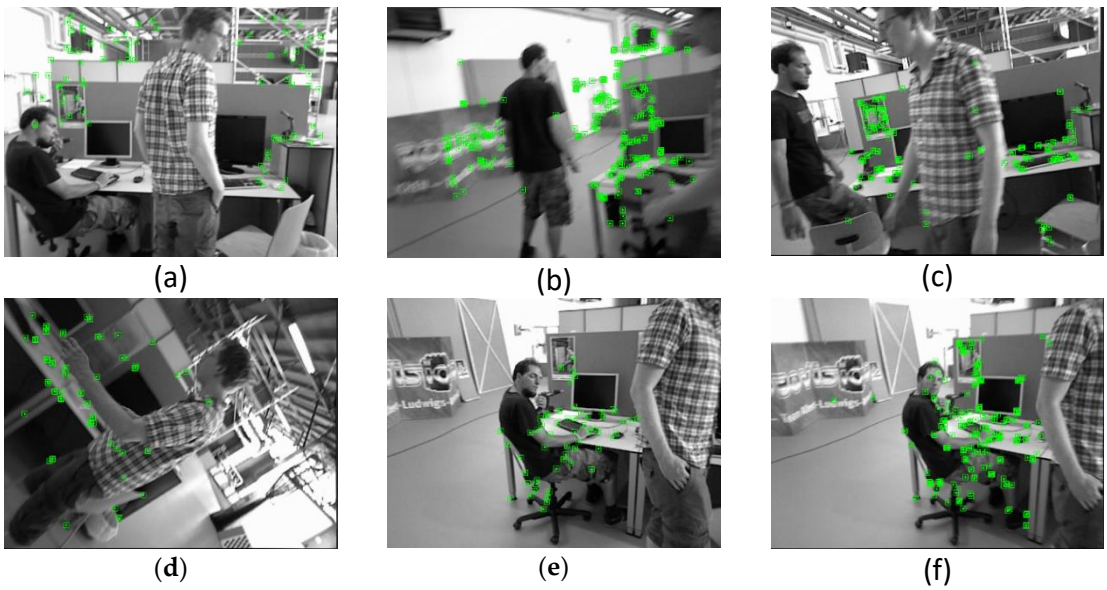


Figure 6. The features used for solving camera pose in the tracking thread, which are marked with green circle.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 January 2020 doi:10.20944/preprints202001.0123.v1

3.3. Overall evaluation using TUM RGB-D dataset

Table II-Ⅳ have shown quantitative comparison results of ORB-SLAM2 and DM-SLAM in sequences fr3_walking and fr1_xyz. RMSE, Mean Error, Median Error and Standard Deviation (S.D.) in this paper, while RMSE and S.D. are more concerned because they can better indicate the robustness and stability of the system. The calculation method of improvements refers to literature [17].

As we can see from Table II-Ⅳ, DM-SLAM make performance in most high-dynamic sequences while make less promotion in the fr1_xyz sequence. In sequence fr3_walking_half, the RMSE and S.D. improvement values of ATE can reach up to 96.42%, 98.45%. In fr1_xyz, the improvements of performance is not obvious. We think when the scene is mainly static, the distinction of DM-SLAM for dynamic features is not necessary for ORB-SLAM system.

Table 1. Results of metric rotational drift(RPE).

Sequences	ORB-SLAM2				DM-SLAM				Improvements			
	RMSE	MEAN	MEDIAN	S.D.	RMSE	MEAN	MEDIAN	S.D.	RMSE	MEAN	S.D.	RMSE
fr3_walking_xyz	2.4385	1.5697	0.9472	1.8661	0.6012	0.5143	0.4302	0.3109	75.30%	67.20%	54.50%	83.30%
fr3_walking_rpy	3.3087	2.8813	2.9891	1.6265	0.9222	0.8085	0.7763	0.4436	75.12%	77.20%	74.02%	72.72%
fr3_walking_half	3.6495	3.8041	3.1594	1.8542	1.2591	1.1187	1.0663	0.5775	65.49%	64.47%	66.24%	68.85%
fr1_xyz	1.0088	0.8911	0.8717	0.4728	0.8372	0.7371	0.6657	0.3969	17.01%	17.28%	17.63%	16.05%

Table 2. Results of metric translational drift(RTE).

Sequences	ORB-SLAM2				DM-SLAM				Improvements			
	RMSE	MEAN	MEDIAN	S.D.	RMSE	MEAN	MEDIAN	S.D.	RMSE	MEAN	S.D.	RMSE
fr3_walking_xyz	0.4888	0.4352	0.4687	0.2224	0.2803	0.2643	0.2643	0.0931	42.65%	42.46%	43.46%	58.13%
fr3_walking_rpy	0.4221	0.3862	0.3955	0.2266	0.1671	0.1405	0.1683	0.0901	60.41%	63.62%	57.44%	60.24%
fr3_walking_half	0.7241	0.6307	0.6437	0.3557	0.3852	0.3251	0.3511	0.1341	46.94%	48.45%	45.45%	57.53%
fr1_xyz	0.0705	0.0606	0.0643	0.0343	0.0438	0.0401	0.0427	0.0186	37.59%	33.82%	21.61%	45.77%

Table 3. Results of metrics absolute trajectory error(ATE).

Sequences	ORB-SLAM2				DM-SLAM				Improvements			
	RMSE	MEAN	MEDIAN	S.D.	RMSE	MEAN	MEDIAN	S.D.	RMSE	MEAN	S.D.	RMSE
fr3_walking_xyz	0.3292	0.3078	0.2942	0.1161	0.1429	0.1298	0.1352	0.0537	56.59%	57.83%	54.04%	53.74%
fr3_walking_rpy	2.9432	2.7549	2.8641	1.8461	0.1055	0.1016	0.0864	0.0285	96.42%	96.31%	96.98%	98.45%
fr3_walking_half	0.5121	0.4534	0.3867	0.2379	0.1548	0.1254	0.1481	0.0716	69.77%	72.34%	61.71%	69.91%
fr1_xyz	0.0492	0.0467	0.0439	0.0151	0.0285	0.0271	0.0277	0.0086	42.01%	41.97%	36.91%	43.05%

4. Discussion

In this paper, we propose DM-SLAM system based on ORB-SLAM to adapt to dynamic environments. DLRSAC is proposed to extract most static matches for initialization even the dynamic objects take dominate whether it is non-rigid or rigid. In tracking thread, we designed neighborhood mutual exclusion algorithm to limit the number of candidate maps. This algorithm reduces the interference of dynamic features on camera tracking and ensures the rapid expansion of the map. Based on above techniques, DM-SLAM system can successfully operate in a dynamic environment,

even if moving objects take dominate in the scene. the static objects represent the reference coordinate system essentially, and the motion of the reference coordinate system cannot be detected in this algorithm.

The limitation of this method is that when a large number of stationary objects appear in the scene and then move in the tracking process, the algorithm will fail. Because in this case, the static objects represent the reference coordinate system essentially, and the motion of the reference coordinate system cannot be detected in this algorithm.

In our observation, static means the most suitable reference coordinate system in the whole world. In under-textured area, non-features pixels can be used to find the global optimal reference coordinate system, which is beyond of feature-based methods. In the future, we consider introducing direct method to extend the robustness of DM-SLAM to adapt to richer dynamic scenarios.

Author Contributions: Conceptualisation, L.X.Y. and T.S.M.; Data curation, L.X.Y. and H.H.M; Formal analysis, , L.X.Y.; Funding acquisition,W.H.; Methodology, L.X.Y.; Project administration, T.S.M.; Resources, L.X.Y. and T.S.M.; Software, L.X.Y.; Supervision, T.S.M. and L.C.; Validation, L.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key R&D Program of China (No.2017YFB1002800)

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Berta, B; Facil, J. M. and Javier, .C. DynaSLAM: Tracking, Mapping and Inpainting in Dynamic Scenes. *IEEE Robotics and Automation Letters* **2018**, vol. 3, no. 4, pp. 4076-4083.
2. Mur-Artal, R.; Tardós, J. D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics* **2017**, vol. 33, no. 5, pp. 1255-1262.
3. Engel, J.; Schöps, T; Cremers, D. LSD-SLAM: Large-scale direct monocular SLAM. *European Conference on Computer Vision* **2014**, pp. 834-849.
4. Engel, J.; Koltun, V.; Cremers, D. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2017**, vol. 40, no. 3, pp. 611-625.
5. Engel, J.; Sturm, J.; Cremers, D. Semi-dense visual odometry for a monocular camera. *The IEEE international conference on computer vision* **2013**. pp. 1449-1456.
6. Forster, C.; Pizzoli, M.; Scaramuzza. D. SVO: Fast semi-direct monocular visual odometry. *IEEE international conference on robotics and automation (ICRA)* **2014**, pp. 15-22.
7. Saputra, M. R. U.; Markham, A.; Trigoni, N. Visual SLAM and structure from motion in dynamic environments: A survey. *ACM Computing Surveys* **2018**, vol. 51, no. 2, pp. 1-37.
8. Fischler, M. A.; Bolles, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* **1981**, vol. 24, no. 6, pp. 381-395.
9. Tan, W.; Liu, H.; Dong. Z. Robust monocular SLAM in dynamic environments. *ISMAR*. **2013**, pp. 209-218.
10. Sun, Y.; Liu, M.; Meng, M. Q. H. Improving rgb-d slam in dynamic environments: A motion removal approach. *Robotics and Autonomous Systems* **2017**, vol. 89, pp. 110–122.
11. Kitt, B.; Moosmann, F.; Stiller, C. Moving on to dynamic environments: Visual odometry using feature classification. *IROS* **2010**, pp. 5551–5556.
12. Chao, Y.; Zuxin, L.; Xinjun, L. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. *Proc. IROS*, **2018**.
13. Schindler, K.;James, U.; Wang, H. Perspective n-view multibody structure-and-motion through model selection. *European Conference on Computer Vision*, Berlin, Heidelberg **2006**, pp. 606-619.
14. Vidal, R.;Ma, Y.; Sastry, S. Generalized principal component analysis. *IEEE transactions on pattern analysis and machine intelligence* **2005**, vol. 27, no. 12, pp. 1945-1959.

15. Yan, J.; Pollefeys, M. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. *European conference on computer vision*. Berlin, Heidelberg **2006**, pp. 94-106.
16. Mur-Artal, R.; Montiel, J. M.M.; Tardos, J. D. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE transactions on robotics* **2015**, vol. 31, no .5, pp. 1147-1163.
17. Sturm, J.;Engelhard, N. A benchmark for the evaluation of RGB-D SLAM systems. *IROS*, 2012.
18. Chum, O.; Matas, J.; Kittler, J. Locally optimized RANSAC. *Joint Pattern Recognition Symposium. Springer, Berlin, Heidelberg* **2003**. pp.236-243.
19. Chin, T.J.; Yu, J.; Suter, D. Accelerated hypothesis generation for multistructure data via preference analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2011**, vol. 34, no .4, pp. 625-638.
20. Fuentes-Pacheco, J.; Ruiz-Ascencio, J.; Rendón-Mancha, J.M. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review* **2015**, vol. 43, no. 1, pp. 55-81.