Edge SLAM: Edge Points Based Monocular Visual SLAM

Soumyadip Maity, Arindam Saha, Brojeshwar Bhowmick
Embedded Systems and Robotics, TCS Research & Innovation, Kolkata, India
{soumyadip.maity, ari.saha, b.bhowmick}@tcs.com

Abstract

Visual SLAM shows significant progress in recent years due to high attention from vision community but still, challenges remain for low-textured environments. Feature based visual SLAMs do not produce reliable camera and structure estimates due to insufficient features in a low-textured environment. Moreover, existing visual SLAMs produce partial reconstruction when the number of 3D-2D correspondences is insufficient for incremental camera estimation using bundle adjustment. This paper presents Edge SLAM, a feature based monocular visual SLAM which mitigates the above mentioned problems. Our proposed Edge SLAM pipeline detects edge points from images and tracks those using optical flow for point correspondence. We further refine these point correspondences using geometrical relationship among three views. Owing to our edge-point tracking, we use a robust method for two-view initialization for bundle adjustment. Our proposed SLAM also identifies the potential situations where estimating a new camera into the existing reconstruction is becoming unreliable and we adopt a novel method to estimate the new camera reliably using a local optimization technique. We present an extensive evaluation of our proposed SLAM pipeline with most popular open datasets and compare with the state-of-the-art. Experimental result indicates that our Edge SLAM is robust and works reliably well for both textured and less-textured environment in comparison to existing state-of-the-art SLAMs.

1. Introduction

Autonomous navigation of robots requires robust estimation of robot’s pose (position, orientation) as well as 3D scene structure. To this end, in recent years, researchers have proposed a variety of algorithms and pipelines for Simultaneous Localization and Mapping (SLAM) [31, 4] using a camera. These visual SLAMs require point correspondences between images for camera (robot) position estimation as well as structure estimation. Feature based methods for visual SLAM [16, 25] try to find the point correspondences between images using SIFT [23], SURF [2] or ORB features [30]. Using such features, visual SLAMs obtain the camera and structure estimation by minimizing the reprojection error through incremental bundle adjustment [36]. These SLAMs precisely depend on the extracted features (SIFT, SURF, ORB) and therefore they miserably fail when the number of points extracted is too less or erroneous especially when the amount of texture present in a scene is very less as shown in figure (a) & (b). Therefore, these SLAMs often produce a partial reconstruction and stop camera tracking when 3D-2D correspondences are less due to insufficient feature correspondences or insufficient 3D points from bundle adjustment. In contrast to feature-
based methods, direct methods [8, 7] find such points by minimizing the photometric error [35, 37] and jointly optimizing the poses of the cameras. While these SLAMs are independent of the feature extraction, they are very erroneous in camera pose estimation due to wrong photometric error estimation in lighting change or view change. Moreover, they do not produce good camera estimations in absence of well-textured environment as discussed in figure 1.

In this paper, we use a feature based approach where we detect reliable edges in the image to track the points lying on these edges using bi-directional robust optical flow [3]. Such tracking will yield strong point correspondences which are further refined using three-view geometry [11]. Using these feature correspondences and epipolar geometry between images we select keyframes for two-view initialization required for structure estimation. Among many such keyframes in an image sequence, we select a particular pair of keyframes using a novel two-view selection method for good initialization. We present a novel two-view selection method for good initialization. We present a comparative result of our novel initialization with existing state-of-the-art methods that clearly exhibit the effectiveness of our method. Then we keep on estimating the new keyframes and the 3D structure using incremental bundle adjustment. Similar to other feature based method, if the 3D-2D point correspondences are ill-conditioned during the addition of a new camera then we apply a novel camera tracking recovery method for continuous tracking of the cameras. If the recovery method fails to produce a reliable camera estimation, the scenario is called track-loss and tries for relocalization. Also, incremental pose estimation accumulates errors introduced at every pose estimation over time resulting in a drift in the camera estimations. Our Edge SLAM uses structural properties of edges in images for computing reliable point correspondences which are used in 3D reconstruction using a local bundle adjustment. We refine the reconstruction globally once a certain number of cameras are estimated. This global bundle adjustment rectifies the drift in the reconstruction. Subsequently, loop closure further refines the camera poses and rectifies such drift. Our Edge SLAM uses structural properties of edges in the images for closing a loop. Our SLAM is robust and reliable in both well-textured and less-textured environment as shown in figure 1. The block diagram of our complete SLAM pipeline is shown in figure 2 where our contributed blocks are shown in red.

Therefore, the main contributions of this paper are:

- We use structural properties of edges for correspondence establishment and loop closing.
- We propose an automatic and robust initialization procedure through validating the reconstructed map quality, which is measured by the geometrical continuation of the map.
- We propose a novel recovery method of camera tracking in a situation where pose estimation becomes unreliable due to insufficient 3D-2D correspondences.

We organize the remaining part of this paper as follows. In Section 2, we review the related work. In Section 3, we describe the entire pipeline of our Edge SLAM and evaluate our contribution. Finally, in section 4, we present the experimental results on popular open sequences.

2. Related Work

In this section, we describe the related work on SLAM which is broadly divided into feature based methods and direct methods.

**Feature based SLAM:** Klein and Murray present the first path-breaking visual SLAM, Parallel Tracking And Mapping (PTAM) [16] which uses FAST corners points [29] as features and provides simple methods for camera estimation and map generation by decoupling localization and mapping modules. PTAM fails to produce reliable camera estimation in a less-textured environment where availability of point feature is minimal. More recently, Mur-Artal et al. present ORB SLAM [25] which uses ORB point feature [30] for point correspondence and yield better accuracy in a well-textured environment. ORB SLAM presents an automatic initialization based on a statistical approach to model selection between planar and non-planar scenes using homography [11] or fundamental matrix [11] respectively. A better initialization always produces a stable 3D structure, but the reconstructed map has never been used to benchmark initialization because the reconstructed map is sparse. ORB SLAM also fails in challenging low-textured environment.

**Direct SLAM:** Direct methods [26] gains popularity for its semi dense map creation. Recently, Engel et al. present LSD SLAM [8] a direct SLAM pipeline that maintains a semi-dense map and minimizes the photometric error of pixels between images. LSD SLAM [8] initializes the depth of pixels with a random value of high uncertainty by using...
Canny are not suitable for 2D edge-point tracking as it is
dient based edge detection approach. Edges detected by
ction method is Canny Edge detection [5] which is a gra-
tect reliable edges in an image first. A popular edge detec-
methods, we choose a set of effective points to track correspon-
correspondence based visual SLAM. Unlike direct meth-
3.1. Correspondence Generation

Point & Line based SLAM: Pumarola et al. [28] present PL SLAM, which is built upon ORB SLAM [25]. They use line feature along with ORB point feature in tracking and mapping. PL SLAM requires very high processing power for dual feature processing. Therefore, none of the feature based SLAM or direct SLAM and visual odometry pipelines work reliably well in both well-textured and less-textured environment. In this paper, we try to address this problem by designing an efficient pipeline of feature based SLAM which works in both well-textured and less-textured environments.

3. Methodology

3.1. Correspondence Generation

Feature Correspondence estimation is a major building block which decides the overall performance of any feature correspondence based visual SLAM. Unlike direct methods, we choose a set of effective points to track correspondences. The key idea of feature extraction is to preserve structural information especially edges and therefore we detect reliable edges in an image first. A popular edge detection method is Canny Edge detection [5] which is a gradient based edge detection approach. Edges detected by Canny are not suitable for 2D edge-point tracking as it is sensitive to illumination change and noise. The precise location of an edge and its repetition in consecutive frames is very important for edge-point tracking. A multi-scale edge detection algorithm is proposed by Lindeberg [21] and phase congruency based edge detectors were proposed by Kovesi [17,18]. We find the DoG based edge detector [24] is reliable due to its robustness in illumination and contrast changes. We thin [19] the edges further to generate edges of a single pixel width. We apply an edge filtering process described by Juan and Sol [24] upon the thinned edges to calculate connectivity of the edge points. This point connectivity information plays an important role in validating edge continuation in different stages of our Edge SLAM pipeline. Edge continuation may not be calculated properly if the input image is blurred or defocused. Those images are high contributing attributes for erroneous feature correspondences as well. We identify and discard those based on an adaptive thresholding method using the variance of gray intensities of edge points of the whole image.

In our Edge SLAM, we estimate feature correspondences of thinned edge points using a bi-directional sparse iterative and pyramidal version of the Lucas-Kanade optical flow [3] using intensity images. We use the window based approach of optical flow to avoid the aperture problem [1]. Point correspondences obtained using only optical flow may contain noise and therefore can create drift in the estimation of flow vectors. We remove those noisy correspondences using several filtering methods. We discard the redundant pixels (pixels whose Euclidean distance is very low) first and then remove the points having a higher bi-directional positional error. If a 2D correspondence present in 3 consecutive keyframes (see Section 3.2 for keyframe selection), we calculate both the corresponding epilines [11] on the middle keyframe as shown in figure 3 and discard 2D correspondences which are not lying at the intersection of the corresponding epilines [11]. Such filtering of the noisy point correspondences between images reduces noisy points and improve tracking accuracy.

3.2. Keyframe Selection

Keyframes are a subset of frames, which we choose to avoid low parallax error and redundancy for robotic movement. The overall accuracy of the system varies on keyframe selection and number of keyframes. ORB SLAM [17] presents usefulness of keyframe selection and we adopt a similar technique which is suitable for our pipeline. Let \( I_{t-1} \) and \( I_t \) are two consecutive frames. We process current frame \( I_t \) and last selected keyframe \( K_m \), where \( 0 \leq m < t - 1 \), for next keyframe selection. We select next keyframe \( K_{m+1} \) using following criteria if any one of the following conditions holds:

- We calculate pairwise rotation between \( I_t \) and \( K_m \) using epipolar geometry [27]. If the rotation is more than
Figure 4. Few frames after initialization in the NewCollege sequence [32]. Top: ORB-SLAM, initialize camera poses calculating fundamental matrix but the reconstructed 3D structure (red points) does not convey any meaningful understanding of the scene. Middle LSD-SLAM, initialize the map with erroneous planar depth. Bottom: Edge-SLAM, initialize camera poses with a continuous 3D structure. Continuous edges (e.g., wall on left side in the point cloud, separation line on the right side in the point cloud) are visible on the picture.

Figure 5. (a) A sample image from fr2_desk [33] sequence. (b) Initialization at frame number 11 reconstructs discontinuous 3D structure. Due to low quality-factor, our pipeline discard this initialization. If the pipeline accepts the initialization and continues, at frame number 117 it produces 1.12 cm root mean square of absolute trajectory error [33] against ground-truth. (c) Initialization at frame number 34 generates continuous 3D structure. Due to higher quality-factor our pipeline select it as valid initialization and at frame number 117 it produces 0.48 cm root mean square of absolute trajectory error [33] against ground-truth.

3.3. Two-view Initialization

SLAM, being an incremental camera estimation pipeline, uses incremental bundle adjustment for estimating cameras. To initialize the bundle adjustment many of the SLAM techniques select two-view as seed pair for the initialization of the cameras after computing epipolar geometry [27] between the image pair and triangulate [12] the point correspondences for initial 3D structure followed by refinement using bundle adjustment. Then new cameras are added into the existing reconstruction through re-sectioning utilizing 3D-2D correspondences. However, such camera and structure estimation may not be well constrained under low-parallax and the output may suffer from ambiguity and drift [22]. Therefore, many of the existing SLAMs use geometrical validation of camera poses for reliable initialization of cameras and structure [16, 25].

In our Edge SLAM, we also use incremental bundle adjustment. We choose two keyframes based on any one of the following conditions hold:

- Pairwise rotation between the keyframes is more than a threshold (generally 15°).
- Averaging Euclidean distance between previous and current pixel exceed 20 percent of input image width.
- Time difference of 1 second time between the frames.

We generate the initial 3D structure based on initial pairwise pose estimation by Five-point algorithm from Nister [27] followed by triangulation [12]. We further optimize the initial structure using Bundle Adjustment. For correctness of this initialization we introduce a novel map validation method. We find the spatial smoothness among 2D points which should also conform to the reconstructed 3D points. An edge can be considered as connected straight as the new keyframe ($K_{m+1}$).

- If number of 3D-2D correspondences reduces below 250, we consider ($I_{t-1}$) as a new keyframe ($K_{m+1}$).
- We compute the average positional change of feature correspondences between $I_t$ and $K_m$ by averaging Euclidean distance between previous and current pixel positions of all correspondences. If average positional change is more than twenty percent of the image width, we consider current frame ($I_t$) as a new keyframe ($K_{m+1}$).
- If none of the previous conditions occur, we consider new keyframe ($K_{m+1}$) in a fixed interval of 1 second.
lines of small length and a straight line in 3D should have same continuation. The ordering of points in straight lines should remain same in both 2D and 3D. In particular, any two neighbouring edge points in 2D should also be neighbour in 3D. We identify those straight lines based on local neighbourhood gradient. The number of such corresponding small straight lines in both 2D images and in reconstructed point cloud using two-views signify the quality of the map. We assign a quality-factor to the map based on that number and keep on finding the two views until the quality-factor is above a threshold. If the quality-factor is greater than a threshold, we fix those seed pair as a valid initialization also produces a better 3D structure for SLAM. Apart from the improvement in camera pose, our initialization also produces a better 3D structure for SLAM. Figure 3 shows an example of our initialization on dataset fR2_desk which is a TUM RGB-D benchmark dataset. Initialization until frame number 11 produces discontinuous structure see figure 3(b) for a sample two-view initialization using keyframes within first 11 frames. The root mean square (RMS) of Absolute Trajectory Error (ATE) against ground-truth until frame number 117 with such initialization is 1.12 cm. Our pipeline rejects this initialization for discontinuity in reconstructed structure. Instead, we initialize from frame number 34 (suggested by our quality metric and the corresponding initial structure is shown in figure 3(c)) using which the RMS of ATE till frame number 117 is 0.48 cm. This clearly shows that our two-view selection strategy using the quality of the initial structure is significantly better than initialization using any two views which produce very sparse or noisy 3D structure. Apart from the improvement in camera pose, our initialization also produces a better 3D structure for SLAM. Figure 4 shows an example where due to our good two-view initialization the 3D structure obtained after a certain period of time by our pipeline is significantly better than LSD-SLAM [8] or ORB SLAM [25]. We attribute the failures of LSD-SLAM and ORB-SLAM for producing good structure is due to the initialization using a planner depth and initialization without structure validation respectively.

### 3.4. Tracking and Mapping

#### 3.4.1 Incremental Pose Estimation & Mapping

Using the initial 3D structure from two-view bundle adjustment, we keep on adding new cameras through re-sectioning [20]. We only add new keyframe into the existing reconstruction instead of all frames. Our re-section based pose estimation method for keyframe only is more general than constant velocity motion model used by ORB SLAM [25] using every frame. The accuracy of initial estimation by constant velocity motion model may drastically fall for any non-linear motion.

After two-view initialization we find the next keyframes using keyframe selection method described in Section 3.2.

---

**Figure 6. Track-loss recovery**: Images $K_{t-2}$, $K_{t-1}$ and $K_t$ are estimated separately in $L$ coordinate and these estimations are transformed to SLAM coordinate by coordinate merging after scale correction.

We add these keyframes into the existing reconstruction by re-sectioning using 3D-2D correspondences followed by addition of new structure points through triangulation [12].

We define two keyframes as co-visible keyframes, only if they have more than 100 common visible 3D points. We choose $C \subseteq K$, where $K$ is the set of all keyframes. $C$ contains the new keyframe $K_m$ and its co-visible keyframes. 3D point set $B$ contains all 3D points visible by all the keyframes of $C$. Now we proceed local bundle adjustment on $C$ and $B$.

$$\min_{C_i, B_i} \sum_{i=1}^{n} \sum_{j=1}^{m} V_{ij} D(P(C_j, B_i), x_{ij} \Psi(x_{ij}))$$

where, $V_{ij} \in \{0, 1\}$ is the visibility of the $i^{th}$ 3D point in the $j^{th}$ camera, $P$ is the function which projects a 3D point $B_i$ onto camera $C_j$ which is modelled using 7 parameters (1 for focal length, 3 for rotation, 3 for position), $x_{ij}$ is the actual projection of the $i^{th}$ point onto the $j^{th}$ camera, $\Psi(x_{ij}) = 1 + r \|x_{ij}\|^2$ is the radial distortion function with a single parameter ($r$) and $D$ is the Euclidean distance. We minimize (equation[1] using [50]. We fix the focal length of the camera using the known internal calibration and pose of the cameras as well as map points are optimized.

We refine the pose and reconstruction globally using global bundle adjustment in every 25 second or 25 locally optimized keyframe accumulated. This refinement using global bundle adjustment correct the drift, if any, in the estimated camera trajectory.

#### 3.4.2 Track-loss Handling

Track-loss is one of the major issues present in the literature of visual SLAM where estimated camera track and reconstructed map breaks due to tracking failure during re-section. Some of the dominant reasons for such failure are occlusion, abrupt motion, corresponding structure point re-
duction in bundle adjustment, reduction in the number of 2D point correspondences due to low-textured environment etc. Every existing SLAM tries to relocate after track-loss but relocation is not guaranteed in robotic movements unless the camera returns back very near to its previously visited location. To mitigate the track-loss problem without relocation, we take a sequence of three consecutive keyframes $K_{t-2}, K_{t-1}$ and $K_{t}$ where first two keyframes are already estimated in SLAM coordinate and we are unable to track current keyframe $K_{t}$ using re-section. There are two possible reasons for such failures in re-section.

**Case 1:** Here 2D correspondences exist in all three keyframes $K_{t-2}, K_{t-1}$ and $K_{t}$ but our pipeline rejects the corresponding 3D-2D correspondences due to high reprojection error on keyframes $K_{t-2}$ and $K_{t-1}$ . So here we try to use these 2D correspondences for estimating $K_{t}$.

**Case 2:** Here 2D correspondence exists till keyframe $K_{t-1}$ and absent in $K_{t}$ due to unstable optical flow. Therefore 3D-2D correspondences are in ill-conditioned for keyframe $K_{t}$. We keep this case out of the scope as optical flow fails.

We try to estimate the keyframe $K_{t}$ separately from SLAM coordinate using 2D-2D point correspondences. Epipolar geometry [27] provides a pairwise rotation and an unit direction from frame $K_{t-1}$ to frame $K_{t}$ in a coordinate system where frame $K_{t-1}$ is at origin. We name this coordinate system as $L \in \mathbb{R}^3$ in rest of the paper, which has an unknown scale difference with SLAM coordinate system. We create new 3D points with 2D correspondences and estimated poses of frames $K_{t-1}$ and $K_{t}$ in $L$ through triangulation [12] and re-estimate the frame $K_{t-2}$ in $L$ by re-sectioning utilizing 3D-2D correspondences. Bundle adjustment further refines the poses for frames $K_{t-2}$ and $K_{t}$ without changing the coordinate system i.e. pose for frame $K_{t-1}$ remain unchanged after bundle adjustment. Bundle adjustment produces stable pairwise rotation and translation pair $(R_{t-1} \rightarrow t, t_{t-1} \rightarrow t), (R_{t-2} \rightarrow t_{t-2}, t_{t-1} \rightarrow t_{t-2})$ for frames $K_{t}$ and $K_{t-2}$ respectively. The pipeline continues only when bundle adjustment produces enough inliers 3D points (more than 100) otherwise try for relocalization. The main idea is to merge the coordinate $L$ with the SLAM coordinate after correcting the scale difference. Figure 6 represents the method graphically where three connected frames $K_{t-2}, K_{t-1}, K_{t}$ are estimated in $L$ and then merge with SLAM coordinate.

We calculate the scale difference between two coordinate systems using the estimations of frames $K_{t-2}$ and $K_{t-1}$ in both the SLAM and $L$ coordinate systems using the equation 2 where $C_{Ki}$ and $C_{Si}$ denote the camera center of $i^{th}$ frame in $L$ and SLAM coordinate system respectively. Scale corrected camera center $C_{scaled,t}$ for frame $K_{t}$ in $L$ follow the relation as given in equation 3.

$$\lambda_{init} = \frac{C_{Si(t-1)} - C_{Si(t-2)}}{C_{Li(t-1)} - C_{Li(t-2)}}$$

(2)

$$C_{scaled,t} = \lambda_{init}(R_{L(t-1)\rightarrow t}^{-1}t_{t-1} \rightarrow t)$$

(3)

We require to align the axes of both the coordinate system in order to merge them. Therefore, we rotate the SLAM coordinate axes to aligned with frame $K_{t-1}$ and calculate the camera center for frame $K_{t}$ in the rotated SLAM coordinate through equation 4 and calculate the camera center for frame $K_{t}$ in the SLAM coordinate through a reverse rotation as given in equation 5.

$$C_{T,S,L,rot} = C_{T,S(t-1)}R_{t-1}^{-1} + C_{T,scaled,t}$$

(4)

$$C_{T,S,t} = C_{T,S,L,rot} \ast R_{t-1}$$

(5)

Pairwise rotation is always independent of any coordinate system and thus we use the pairwise rotation of frame $K_{t}$ to get absolute rotation $R_{t}$ in SLAM coordinate using equation 6 [11] where $R_{t-1}$ is the absolute rotation of the frame $K_{t-1}$ in SLAM coordinate.

<table>
<thead>
<tr>
<th>Sequences</th>
<th>ORB-SLAM Track lost</th>
<th>LSD-SLAM Track lost</th>
<th>No. of times track-loss recovery is used</th>
<th>RMS of ATE (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>frb_str_not_far</td>
<td>Unable to initialize</td>
<td>frame 357</td>
<td>7</td>
<td>6.71</td>
</tr>
<tr>
<td>frb_str_not_far</td>
<td>Unable to initialize</td>
<td>frame 357</td>
<td>7</td>
<td>6.71</td>
</tr>
<tr>
<td>frb/office</td>
<td>frame 354</td>
<td>frame 193</td>
<td>5</td>
<td>19.5</td>
</tr>
</tbody>
</table>

Table 1. Comparison among ORB SLAM, LSD SLAM and our Edge SLAM in some low-textured sequences along with the details of our track-loss recovery method.
Loop closing tries to rectify such drift pose estimations accumulate errors and create a drift in estimated trajectory. Loop closure tries to rectify such drift. Loop closing corrects any existing drift in the estimations of $K_t$ and its neighbouring keyframes. We update the pose by matching structural properties of images between non-neighbouring keyframes (does not share any point correspondence).

### 3.5.1 Loop Detection

We consider loop if two non-neighbouring estimated keyframes share a similar scene from a different view point. Our loop detection method tries to find loop only on keyframes. A reliable loop detection method should be invariant to scale changes, translation, rotation of scene structure. Image moment invariants are invariant of those specific transformations and are very good features to use when dealing with a large set of images and a large displacement. This is a well-accepted technique to classify objects as well as for pattern recognition where it identifies objects through edge matching by the third order moments invariants of polygon.

So we exploit image moment invariants and match edges of keyframes based on third order moments. Subsequently, we adopt a multilevel matching mechanism where every keyframe is divided into 16 quadrants and matching score is calculated based on the matched quadrants between a pair of keyframes. An adaptive weighted average of the number of edges, average edge pixel density and average intensity in each quadrant are used to find the matched quadrant. We derive a final matching score between two keyframes, averaging the matching score of moments invariants and voting.

Initially, our method calculates matching score of last keyframe $K_t$ with maximum 5 previous keyframes, which have less than 30$\degree$ degree viewing direction change (immediate neighbour with high cohesion) and retain the lowest matching score $M_{min}$ as the threshold to choose matching with non-neighbouring keyframes. Subsequently, it calculates the matching score of all non-neighbouring keyframes with $K_t$ and retains only the keyframes that are having a matching score more than $M_{min}$. We consider a keyframe as a matched keyframe with $K_t$ only if three consecutive keyframes are having matching score more than $M_{min}$ to avoid wrong matches. Our method select the keyframe as loop candidate ($K_{loop}$) having maximum matching score among the three consecutive keyframes. Subsequently, we calculate point correspondences between keyframe $K_t$ and $K_{loop}$ as described in sec. We use the 3D-3D correspondences as given in Table 1 and consider $K_{loop}$ as the loop keyframe of $K_t$ if we find $T_{sim}$ with enough inliers (more than 100).

### 3.5.2 Loop Merging

Loop merging corrects any existing drift in the estimations of $K_t$ and its neighbouring keyframes. We update the pose...
for $K_t$ and its neighbouring keyframes using $T_{sim}$. There is a set of 3D points $M_{loop}$ that are visible to $K_{loop}$ and its neighbours whereas the set of 3D points $M_t$ that are visible to $K_t$ and its neighbours. We project the 3D points belong to $M_{loop}$ to the current keyframe $K_t$ and its neighbours and check for the 3D-2D correspondences between these projected points and the 3D points belong to $M_t$. We merge all those map points where the 3D-2D correspondences are found and those that were inliers in $T_{sim}$. Global bundle adjustment further refines the poses of all keyframes in the entire loop and corresponding map points.

4. Experimental Results

We have used an Intel Core i7-7700 (4 cores @ 3.60GHz) with 8Gb RAM, for implementation of our SLAM pipeline. We extensively experiment with TUM RGB-D benchmark datasets [33] and ICL-NUIM [10] dataset. We have used TUM RGBD Benchmarking tool [33] to compare the camera estimations by our Edge SLAM against ground-truth. Table 2 also shows the comparison of Absolute Trajectory Errors by our Edge SLAM, LSD-SLAM, ORB-SLAM, Edge VO and PL SLAM [28] where it is evident that unlike the existing pipelines, our Edge SLAM works on all kind of datasets reliably. Moreover, we like to emphasize that our pipeline also produces accurate structure and camera estimations even for a low-textured environment where most of the existing SLAMs fail in tracking (six rows from bottom of Table 2). Most recent visual odometry pipeline Edge VO is limited to produce only camera localization without any map and produces the most erroneous result as shown on Table 2. An example on loop closing is presented on $fr_{2\_desk}$ sequence in Figure 8 where the RMS of ATE is reduced from 9.5 cm to 1.75 cm after loop closing.

In addition to better camera localization, our pipeline produces significantly improved structure compared with existing SLAMs. Figure 9 shows an example of structure comparison of our Edge SLAM with LSD & ORB SLAM where, it is evident that our pipeline produce superior quality structure against existing SLAM pipelines. We left out the structure comparison with Edge VO as the pipeline intended to produce only camera motion. We also present the module wise time analysis in Table 3 calculated on $fr_{2\_desk}$ sequence. Our camera tracking method runs on 17 fps, where as the mapping method runs on 4 fps using an unoptimized implementation.

Table 3. Module wise mean execution time on $fr_{2\_desk}$ [33].

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge extraction</td>
<td>19</td>
</tr>
<tr>
<td>Correspondence generation</td>
<td>7</td>
</tr>
<tr>
<td>Keyframe selection</td>
<td>14</td>
</tr>
<tr>
<td>Pose estimation</td>
<td>11</td>
</tr>
<tr>
<td>Map generation</td>
<td>9</td>
</tr>
<tr>
<td>Local bundle adjustment (on 5 keyframes)</td>
<td>115</td>
</tr>
<tr>
<td>Track-loss recovery</td>
<td>14</td>
</tr>
</tbody>
</table>

5. Conclusion

We present a visual SLAM pipeline with the focus on track in both textured as well as very low-textured environments and building recognizable maps. We start with initializing SLAM through a validation process that produces better initialization compared with state-of-the-art visual SLAMs. We present a novel local optimization method for stable camera estimation in the situations where camera tracking becomes unreliable in a very low-textured challenging environment. Our pipeline is capable of an efficient and reliable loop closing using structural properties of edges in images. The pipeline shows a significant improvement in map generation in terms of semantic understanding.
References


