Online Environment Mapping

Jongwoo Lim
Honda Research Institute USA, inc.
Mountain View, CA, USA
jlim@honda-ri.com

Jan-Michael Frahm
Department of Computer Science
University of North Carolina
Chapel Hill, NC, USA
jmf@cs.unc.edu

Marc Pollefeys
Department of Computer Science
ETH Zurich, Switzerland
marc.pollefeys@inf.ethz.ch

Abstract

The paper proposes a vision based online mapping of large-scale environments. Our novel approach uses a hybrid representation of a fully metric Euclidean environment map and a topological map. This novel hybrid representation facilitates our scalable online hierarchical bundle adjustment approach. The proposed method achieves scalability by solving the local registration through embedding neighboring keyframes and landmarks into a Euclidean space. The global adjustment is performed on a segmentation of the keyframes and posed as the iterative optimization of the arrangement of keyframes in each segment and the arrangement of rigidly moving segments. The iterative global adjustment is performed concurrently with the local registration of the keyframes in a local map. Thus the map is always locally metric around the current location, and likely to be globally consistent. Loop closures are handled very efficiently benefiting from the topological nature of the map and overcoming the loss of the metric map properties as previous approaches. The effectiveness of the proposed method is demonstrated in real-time on various challenging video sequences.

1. Introduction

The progress of robotics and computing hardware in the last decade has increased the demand for online metric map reconstruction from cameras. At the same time the scale of those maps has been increased by two to three orders of magnitude [1, 10, 8]. This poses a significant challenge for current state of the art camera based large scale modeling approaches. One of the most demanding applications of vision based map reconstruction is in robotics. Robots inherently need to model the surround environment to safely navigate in the space while performing the various tasks. Our proposed approach for online simultaneous localization and mapping (SLAM) is tackling this problem which has also been intensively studied in the robotics community for decades.

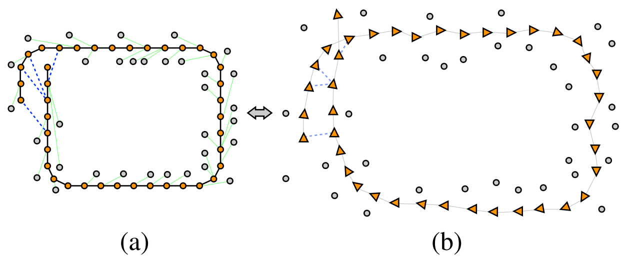


Figure 1. (a) A keyframe pose graph which shows the topological structure of the environment. Orange circles represent the keyframes and gray circles the landmarks. (b) A metric embedding of the keyframes together with the landmarks. Refer the text for detailed description.

Traditionally laser range finders (LIDAR) have been used in this task mainly because they directly measure the distance to the surface with high precision. However there are significant limitations in this type of sensors. The major limitation is that typical LIDAR sensors only scan a 2D slice of the space and the slice needs to be in the same plane for a SLAM system to work. This limits the use of laser-based SLAM systems in an environment with objects with complex height profile (such as tables or shelves) and when the robot moves freely in 3D space, not restricted on a flat floor. Moreover LIDAR sensors require highly accurate tracking on mobile platforms when moving. Another issue with the sensor is its size, weight and power consumption, which is significantly larger than passive sensors like video cameras.

Recently there have been many attempts to address SLAM for larger environment using monocular or stereo cameras [9, 14, 8]. By using cameras, the visual SLAM system can avoid the shortcomings of laser-based SLAM systems, but instead it needs to solve the problem of estimating the 3D environment from 2D image observations. The recent advances in structure-and-motion research provide the theoretical basis and useful machineries [23, 21].

In SLAM systems, the most difficult problem is to maintain the map (the perceived model of the environment) consistent to all observations, especially when loops exist in the robot trajectory. Traditionally in visual SLAM this map adjustment is performed by bundle adjustment which scales cubically with the problem size prohibiting online computation in large scale environments. Topological mapping is devised to avoid this problem. It represents the world as a graph with a set of places (nodes) and the relative location information between the places (edges). In this representation, a loop closure does not require any additional error adjustment. However, in return, it loses the globally metric property. For example, a robot can not perform spatial reasoning for proximity unless the link between the map locations is present in the topological map.

The proposed approach deploys a hybrid mapping method combining the benefits of metric Euclidean maps and topological maps, namely the locally-metric globally-topological mapping. The map is represented as a graph of the keyframes (nodes) and the relative pose between keyframes (edges), like the topological approach. The main distinction to previous approaches is that the system enforces the metric properties as much as possible, i.e. if one follows the links and merges the relative poses in order, it will become the identity transformation. Our method strictly enforces the locally metric property all the time via local adjustment, and global "metricness" is achieved (with some delay) via a decoupled parallel global bundle adjustment module.

The paper is organized as follows; in the next section we review the related work. Our proposed novel method is described in detail in Section 3. More information on the design choices of the method are provided in Section 4 and Section 5 presents the experimental results on various challenging sequences. Finally we summarize the proposed method in Section 6.

2. Related Work

Triggs et al. [27] brought bundle adjustment into the focus of the computer vision community about a decade ago. Since then bundle adjustment has become a topic of high interest in computer vision given the ability to perform large scale structure from motion from video [23] or from Internet photo collections [25]. The results of the inherently incremental structure from motion process has to be adjusted to overcome drift [7]. We propose a combination of nested dissection and topological mapping to address this challenge.

Essentially bundle adjustment parameterizes structure from motion as an optimization problem, which characterizes each camera with six degrees of freedom (DOF) for the translation and rotation of the camera and plus parameters for the camera calibration and radial distortion. Additionally, the 3D points are parameterized through their three position parameters. The projection equations are then used to derive a non-linear set of equations which are then linearized through a Taylor series. This delivers a large sparse linear equation system that can be efficiently solved through a sparse Levenberg-Marquardt solver [12]. Large scale reconstructions are challenging since the complexity of the bundle adjustment is at least cubic in the number of cameras plus a linear complexity in the number of points.

The scalability challenge has been recently addressed by several researchers through a variety of methods. A very prominent approach is nested dissection [2, 15], which is a divide-and-conquer method to solve a set of sparse constraints as used in structure from motion. Nested dissection regroups the reconstruction problem into sub-maps, which conceptually can be solved independently except for the cross sub-map interactions through parameters that influence multiple sub-maps. Then the parameters of the global problem are sorted in two groups one group of parameters that only influence one sub-map (intra-sub-map parameters) and the second parameter group effecting more than one sub-map (inter-sub-map parameters). To enable parallelism during the sparse Cholesky factorization in the inner loop of the sparse Levenberg-Marquardt solver, which provides a sizable speed-up. The concept was initially deployed in the field of photogrammetry to improve the efficiency of bundle adjustment [5].

A concept similar in spirit to nested dissection has been introduced by Ni et al. [20]. They proposed a hierarchical bundle adjustment to overcome the computational complexity by partitioning the bundle adjustment into connected sub-maps by modeling the parameters of each sub-map in a local coordinate system and also dividing them into intra-sub-map parameters and boundary variables. In contrast to nested dissection the sub-problems are solved till convergence and only then propagate the residual energy into the global solution achieving significantly higher convergence. In contrast to both of the above systems our proposed approach is modeling the inter-sub-map relationships through topological transformations effectively summarizing the constraints into the transformation. This provides a more compact representation of the boundary variables leading to a computationally more efficient solution of the global problem. Ni and Dellaert recently extended their approach [19] to a multi-level sub-map tree which represents the constraints between the different sub-maps through the graph edges.

Alternatively, Snavely et al. [26] aimed at reducing the redundancy in the data by determining a set of essential cameras ("skeletal cameras") and removing all remaining cameras from the optimization. Our system reduces the number of cameras by selecting key-frames along the lines of the approach of Clipp et al. [8], and in global problem further reduction using segments of keyframes is employed.

In our method we encode the constraints between sub-maps through the topological transformations between the sub-maps. This is similar to the reduction of the parameter space introduced by Konolige and Agrawal in their FrameSLAM method [14]. The reduction in FrameSLAM is achieved through a marginalization of feature constraints into 6DOF constraints between cameras with their corresponding uncertainties. In contrast to FrameSLAM our method does not depend on the linearization of the projection function to create the marginalized constraints, hence we do not suffer under the inaccuracy in the linearization.

Holmes et al. [13] linearize the 3D features in the local coordinate system of the cameras to obtain a locally consistent fast solution of the bundle adjustment losing the globally metric property of the solution. In contrast our method achieves a globally metric solution while maintaining the efficiency during the solution of the sub-maps.

Similarly the HMT-SLAM of Blanco et al. [4] aims at large scale modeling in real-time. Blanco et al. propose to update a topological map of metric sub-maps to a globally consistent map by optimizing the constraints of adjacent sub-maps (sub-maps which have a topological transformation defined in between them) through iterative optimization of the non-linear constraints. The latter process is similar to bundle adjustment but does not encode the information of the local metric sub-map as in our proposed map.

After briefly reviewing the previous work we now will introduce our method in more detail.

3. Online Mapping

3.1. Keyframe Pose Graph

The environment map is represented as a *keyframe pose graph*, whose nodes are the keyframes and edges represent the relative pose between two keyframes. More precisely, an edge $a \rightarrow b : P_{ab}$ represents a link from node a to node b with the associated 3D Euclidean transformation P_{ab} (P_{ab} is a 4×4 camera projection matrix with six 3D pose parameters, which is inverse of the camera motion matrix). As the pose graph is an undirected graph; if $a \rightarrow b : P_{ab}$ is in the graph, $b \rightarrow a : P_{ba} = P_{ab}^{-1}$ is also in the graph. An example keyframe pose graph is shown in Fig. 1 (a). Please note that there intentionally is no global coordinate system in this representation.

The map is incrementally constructed as the camera moves. Most keyframes are linked to the previous keyframes via commonly observed landmarks (these temporal links are shown as black lines in Fig. 1). When the robot visits previously seen places, the location recognition module [8] finds additional links between keyframes, which creates loops in the pose graph (dashed lines in Fig. 1). The landmarks are attached to the *anchor keyframes* in which they are first observed (green lines in Fig 1 (a). Each landmark's position is represented as a homogeneous 4-vector $\boldsymbol{x}$ in the anchor keyframe's coordinate system.

A metric embedding of the keyframe pose graph is constructed as follows. For a given reference keyframe a_0,

```
1. put the keyframe $a_0$ at the origin ($\hat{P}_{a_0} = I_{4\times4}$), and
   push $(0, a_0)$ into a priority queue $pq$.
2. repeat until $pq$ is empty.
   - pop $(d, a)$ with smallest $d$ from $pq$.
   - for each neighbor keyframe $b$ of $a$, $a \rightarrow b : P_{ab}$,
      - if $b$ is not in the embedding,
         - add the keyframe $b$ with the pose $\hat{P}_b = P_{ab}\hat{P}_a$.
         - put $(d + |P_{ab}|_G, b)$ into $pq$.
3. for each landmark $l$ and its anchor keyframe $c_l$,
   - add the landmark $l$ at the location $\hat{\boldsymbol{x}}_l = \hat{P}_{c_l}^{-1}\boldsymbol{x}_l$.
```

$\hat{P}_a$ denotes the pose of a keyframe a in the embedded space. $|P|_G$ denotes the norm of the translation component in P, thus d in (d, a) is in fact the geodesic distance from a_0 to a on the graph.

Conceptually this procedure performs weighted breadth-first search of the pose graph from the reference keyframe and embeds the keyframes according to the order. The landmarks are then embedded using their anchor keyframe's embedded pose. Fig. 1 (b) shows an example of the embedded keyframe pose graph. Note, that the embedded maps may be very different depending on the choice of the reference keyframes, and also there is no guarantee that a loop in the map remains as a valid loop in the embedded map. If there is metric inconsistency in a loop (when the combined transformation along a loop $\neq I_{4\times4}$), the accumulated error will break the farthest link from the reference keyframe.

Compared to the relative bundle adjustment (RBA) [24], the proposed system is to improve this artifact in topological mapping by enforcing the metric property through local and global adjustment. Note that there is no simple way to enforce the metric constraint in purely topological framework. Our hybrid approach enables us to maintain the benefit of topological maps (instant loop closure) whereas the map is enforced to be metrically correct after local and global adjustment.

3.2. Local Adjustment

Intuitively it is expected that a new keyframe provides the majority of changes in the map in its nearby keyframes with commonly visible landmarks. In our method this change is computed through the local adjustment module, which improves the estimated motion of the current keyframe, and ensures a locally metric map around the current keyframe's location. When a new keyframe is added, the estimated pose of the keyframe may contain small error because it is computed and optimized using the landmarks, which are themselves fixed in position. In the case of a detected loop, the local and global metric constraint may additionally be violated by the newly added link. It is

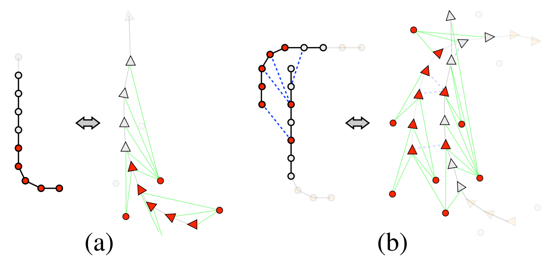


Figure 2. (a) Local bundle adjustment. Active keyframes and landmarks are drawn in red, and fixed keyframes in light gray (window size=5). Green links show the observations of landmarks by keyframes. The fixed keyframes are those outside the window but with observations to the active landmarks. Dimmed keyframes and landmarks are not used in the local optimization. (b) Local bundle adjustment with a detected loop.

very important to resolve these inconsistencies in time, at least locally, to ensure that the pose estimation and location recognition in the next keyframes will work properly.

Our proposed local bundle adjustment updates the links to the *active* keyframes and the positions of the active landmarks. The most recent-w keyframes (w is the window size parameter, typically 5$\sim$10) are initially selected as the active keyframes. If there are recognized links to the initial active keyframes, those keyframes are also added to the active keyframe set. The size of the active keyframe set is bounded since the number of active keyframes is at most twice the window size w due to the fact that the location recognition adds no more than one additional link per keyframe. Next, all landmarks visible from the active keyframes are used in the optimization as the active landmarks. All other keyframes which have the observations of the active landmarks are included as *fixed* keyframes that allows to use all available observations of the landmarks in the local adjustment.

Local adjustment performs the following steps for the new keyframes.

1. find the active keyframes $\{a_j\}$, then determine active landmarks $\{l_i\}$ and fixed keyframes $\{a'_k\}$.
2. embed $\{a_j\}$ $\{l_i\}$ $\{a'_k\}$ into $\{\hat{P}_{a_j}\}$ $\{\hat{\boldsymbol{x}}_{l_j}\}$ $\{\hat{P}_{a'_k}\}$ in a local metric space centered at the most recent keyframe $a_0 \in \{a_j\}$.
3. run the sparse bundle adjustment algorithm [16].
4. update the map using adjusted keyframe poses $\{\tilde{P}_{a_j}\}$ and landmark positions $\{\tilde{\boldsymbol{x}}_{l_j}\}$:
   - any existing $a \to b$, $a \in \{a_j\}$, set $a \to b : \tilde{P}_b \tilde{P}'^{-1}_a$.
   - any existing $b \to a$, $a \in \{a_j\}$, set $b \to a : \tilde{P}_a \tilde{P}'^{-1}_b$.
   - any $l \in \{l_j\}$ with its anchor keyframe c_l, $\boldsymbol{x}_l = \tilde{P}_{c_l} \tilde{\boldsymbol{x}}_l$.

In the embedded metric space the standard sparse bundle adjustment (SBA) algorithm [16] is used for optimization. After applying the Schur complement, the largest linear system to be solved has at most $12\times w$ variables, which can be solved very quickly. The number of landmarks and fixed keyframes affects the performance through the increased number of observations, but in usual setup the local adjustment runs efficiently. Once the keyframes and landmarks are embedded in a metric space, the explicit topological structure is not used anymore, but it still remains in the observations that associates keyframes and landmarks. Thus in SBA, the same topological structure is being used implicitly through the Jacobian matrices for keyframes and landmarks. After it finishes, the optimized keyframes and landmarks are imported back into the graphical environment map. Note that all anchor keyframes for the active landmarks are always part of the embedding from the construction.

While this process on first sight seems similar to RBA [24] it shows advantages over RBA through the details. The most important improvement is that our method guarantees the local metric constraint since all entities are embedded in a metric space, whereas that is not guaranteed in RBA. Moreover, our approach has no need to propagate Jacobian matrices over the edges of the graph making it more computationally efficient. Additionally, our technique is conceptually simpler than RBA since it fully operates in the familiar metric space and can use all known bundle adjustment methods.

3.3. Global Adjustment

With the above described local adjustment, it is guaranteed that the map is locally metric, but still the entire map may not be metric due to the errors along the loops. Achieving global metric consistency is in general not simple in topological maps. The approach we take is to embed the entire map into the metric space, optimize the embedded structure, and update the result back into the topological map. This is fundamentally identical to the local adjustment step, but when a large number of keyframes and landmarks exist, this may take significant computation time and may have difficulty in converging to the right map.

In this section we propose a novel divide-and-conquer strategy to efficiently solve the global adjustment problem. First the keyframes are clustered into multiple disjoint sets (called *segments*) using geodesic distance on the graph, then the global adjustment module iterates local segment-wise optimization and global segment optimization:

1. group the keyframes into κ segments $s_k = \{a_j^{(k)}\}$.
2. for each segment s_i, run local adjustment if necessary.
3. run global segment optimization:
   - embed all segments $\{s_k\}$ into a metric space, $\{\hat{Q}_{s_k}\}$.
   - embed all landmarks $\{l_i\}$ into the same space, $\{\hat{\boldsymbol{x}}_{l_i}\}$.
   - optimize $\{\hat{Q}_{s_k}\}$ and $\{\hat{\boldsymbol{x}}_{l_j}\}$, into $\{\tilde{Q}_{s_k}\}$ and $\{\tilde{\boldsymbol{x}}_{l_i}\}$.
   - update the map using $\{\tilde{Q}_{s_k}\}$ and $\{\tilde{\boldsymbol{x}}_{l_i}\}$.

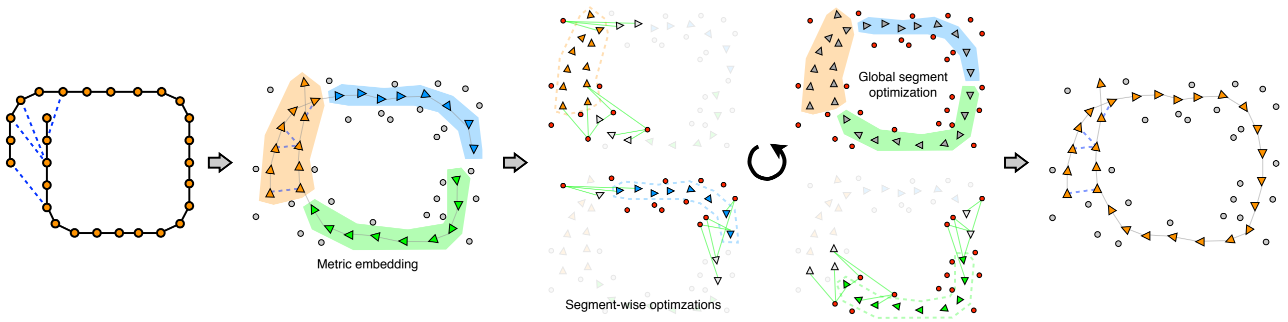


Figure 3. Global adjustment procedure. The keyframe pose graph is partitioned into multiple segments and the segments are embedded into a metric space. Each segment is optimized by the local adjustment algorithm if necessary, then the global segment optimization adjusts the segments' poses and landmarks' position assuming that the segments are moving rigidly. After iteration the result keyframe poses and landmark positions are updated back into the keyframe pose graph. The green lines in the segment-wise optimization illustration are shown to visualize how the fixed keyframes are selected.

In global segment optimization, segments are treated as rigid bodies in embedding and optimization in Step 3. $\hat{Q}_s$ denotes a segment-wise six degree of freedom 3D rigid motion, and the projected coordinate of landmark l to keyframe j in segment k is

$$p(\hat{\boldsymbol{x}}_l, \hat{P}_j, \hat{Q}_k) = K\hat{P}_j\hat{Q}_k\hat{\boldsymbol{x}}_l \tag{1}$$

where K is the 3×4 camera projection matrix. Note that $\hat{P}_j$ is only updated in Step 2 and kept constant in Step 3. Figure 3 illustrates the proposed algorithm.

Since each segment moves as a rigid body, the number of variables in the linear system after Schur complement is $6{\times}\kappa$. The main idea is to make the global segment adjustment faster by reducing the number of variables, and more stable by grouping nearby keyframes together. As discussed earlier, in the embedded space the inconsistency along a loop is concentrated at the farthest link from the reference keyframe, thus there may be a large opening or overlap at the link. If individual keyframes are used with SBA for global adjustment, it is likely that the keyframes around this link may not converge to the right pose (see Fig. 5 (b)). If a group of keyframes is restricted to move rigidly, the contribution of each observation is accumulated to the segment instead of the individual keyframe, and it is more likely to find the correct pose, although it may not be as accurate as optimizing individual keyframes. The small errors that may be caused by rigid segment treatment will be reduced by the segment-wise optimization in the next iteration.

As briefly discussed in Section 2, the proposed method has several advantages over existing methods. Using nested dissection with boundary variables [20, 19] has a serious problem of most of variables being boundary when the graph is not very sparse and segmentation is fine. Long tracks of features induces dependencies among all keyframes that observe common landmarks, and the sparsity is significantly reduced. The proposed method does not have this issue since it treats each segment as a virtual camera, so the size of global optimization does not depend on the sparsity of the map.

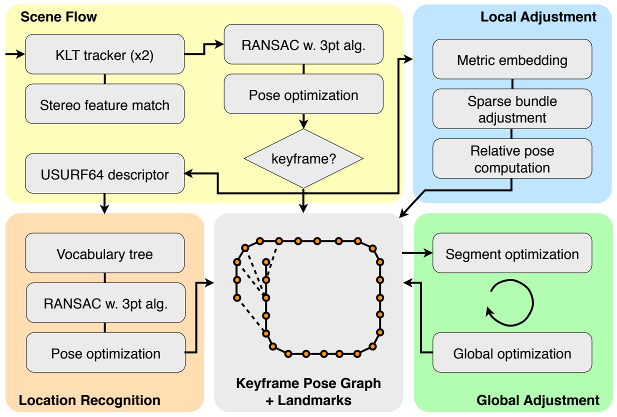


Figure 4. System overview. The system consists of four independently running tasks. Scene Flow processes input stereo frames, tracks features and generates keyframes if there are enough changes. The keyframes are added to the map and passed to Local Adjustment and Location Recognition modules. Location Recognition detects possible loop closure, and Local Adjustment enhances local geometry around the recent keyframes in the map. Global Adjustment periodically checks and optimizes the global map.

4. Algorithm

In this section we detail our algorithm and its realization on a robot platform. The method takes a calibrated stereo video stream as input and generates an environment map of sparse 3D point landmarks. The proposed online environment mapping system consists of four major components: Scene Flow Computation, Location Recognition, Local Adjustment and Global Adjustment (see Fig. 4). All components are executed in parallel to minimize latency and to maximize throughput. Necessary informations are propagated between modules using message passing.

Scene Flow The Scene Flow module is responsible for detecting and tracking salient features in the input video stream, finding inlier features among the tracked features and computing the initial six degree of freedom motion estimates of the camera system. This is done for all input frames, so the robot has the pose estimate at all times.

Our feature detection uses the Harris corner detector, limited to detection on edges [28]. This ensures the feature placement to be on the true corners in the scene. Detected features are then tracked by two 2D KLT trackers [18] on the left and right camera's video streams separately. Stereo correspondences of the features are established using Normalized SSD [6] when they are initially detected, and they are constantly checked if they are on the epipolar line with valid disparity during tracking. The initial 3D position of a landmark is computed using the disparity from the stereo feature match, and as the camera moves the Local and Global Adjustment modules update the position using all available observations from different view points.

Some of tracked features may be drifted or may be from independently moving objects. We employ a 3-point algorithm embedded in a RANSAC [11] for robust motion estimation and outlier rejection. Once RANSAC finds the initial 3D camera pose and inlier features, the 3D pose is enhanced with a non-linear optimization using all inlier features, and a new set of inliers is found with the enhanced pose estimate.

If there exists enough camera motion or change in features, a keyframe is created and added to the map. The new keyframe is then passed to the Location Recognition and the Local Adjustment for further processing. Newly established features are added as new landmarks, and the landmarks with too few observations are later removed from the map when they are lost in tracking.

To boost performance all low-level image processing routines are parallelized including image rectification, KLT feature tracking and stereo feature matching, and are executed on commodity graphics processors.

Location Recognition The Location Recognition module tries to find possible loop closures, i.e. if the system is revisiting a previously-captured location. It uses USURF-64 [3] descriptors computed on the tracked feature points. This is possible because the scale of each feature can be computed from the inverse of the depth of the feature. The advantages are increased performance by saving the interest point detection and better stability in determining the feature's scale. The descriptors are computed using integral image technique on the CPU as proposed in [3], and it is attached to the landmark observation.

Candidate keyframes are selected using the vocabulary tree [22] on the USURF-64 descriptor (width=40, depth=3), which is trained off-line from 1.76 million descriptors from various indoor and outdoor training videos. For each candidate we perform the relative pose estimation using RANSAC with the 3-point algorithm similarly to the Scene Flow module, and the candidate with most inliers (above a given threshold) is chosen as the location recognition result. The obtained pose estimate and the inlier set are improved via a non-linear optimization. If a match is successfully found, a new link connecting the current keyframe to the detected keyframe is added into the keyframe pose graph. This link will then be optimized by both the Local and the Global Adjustment modules.

Local Adjustment The Local Adjustment module performs a windowed bundle adjustment of the recently added keyframes as described in Section 3.2. The standard sparse bundle adjustment algorithm [17] with pseudo Huber norm is implemented internally using LAPACK.

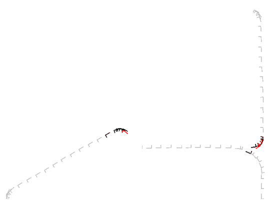
Global Adjustment The Global Adjustment module performs the optimization of the entire map as described in Section 3.3. The keyframes, which are currently considered in the Local Adjustment's windowed bundle adjustment are excluded from the Global Adjustment to avoid inconsistencies by updating the same link in different modules. Segment-wise optimization is performed in the same way as the Local Adjustment with all keyframes in the segment as active keyframes. For global segment optimization, we need to compute a Jacobian matrix for segment-wise motion (Eqn. 1), and the rest is similar to the local adjustment.

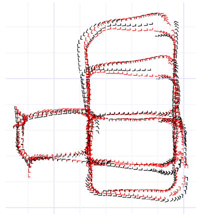
To make Global Adjustment use as many keyframes as possible, the global optimization iterates only once and new segmentation is found using all available keyframes including newly added keyframes after the previous global optimization.

5. Experimental Result

We use two stereo camera rigs for our experiments, one with 7.5cm baseline and about 110° horizontal field of view mounted in the head of a humanoid robot. The second stereo rig has a baseline of 16 cm and a 95° horizontal field of view and is mounted at the front of an electric cart. The effective image resolution after rectification is 640×360 pixel. The test videos are recorded at 12∼15 fps.

First we demonstrate that our proposed local adjustment can handle the topological changes successfully. The Two images shown in Fig. 5 (a) are the local embeddings before and after the loop closure (in the robot sequence in Fig. 6 (a)). Note that the topological changes are reflected in the optimization, where the loop closure creates additional constraints among keyframes. With only local adjustment and location recognition, the resulting map is only locally metric. Globally the embedding may have large openings like the black trajectory in Fig. 5 (b). Severe misalignments may even prevent traditional bundle adjustment


(a)


(b)

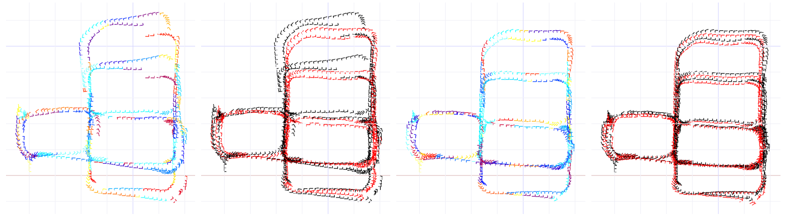

(c)

Figure 5. (a) Local adjustment around a loop closure event. Left image is before a loop closure, and only 5 recent keyframes are active. Right is after the loop closure where additional fixed frames are included in the embedding. (b) Direct application of SBA on an embedding of Fig. 6 (c) without global adjustment. Note that the result is not accurate due to large misalignments in the embedded map. (c) First two iterations of the proposed global adjustment on the same input map as (b) are shown. The first and third images with various colors show the segmentation after segment-wise optimization, and the second and forth show the global segment optimization result. Note that it converges reasonably good even after the first iteration. For all result images, gray is fixed keyframes, black and red are active keyframes before and after the optimization respectively. Only keyframes are shown for presentational clarity, but all associated landmarks are used in optimizations.

from converging to the right map, as the red trajectory in the same figure. Fig. 5 (c) shows how our proposed global adjustment works. For each iteration, it segments the map into several pieces (the 1st and 3rd image), individual segments are optimized locally, afterwards it aligns the segments jointly with all the landmarks. The results of the global segment optimization are also presented in Fig. 5 (c).

Fig. 6 shows the final mapping results of the three different test sequences. For full details on the system's performance, please see the videos in the supplementary material.

The first sequence is from a humanoid robot walking in a large building. Due to the robot's motion characteristics the camera experiences shaking and vibrations, but the feature tracker was able to robustly track the features. There is a corridor with very few features in which case the motion estimation becomes inaccurate, despite this locally decreasing accuracy is global adjustment able to deliver the global geometry correctly as shown in Fig. 6 (a).

Additionally, we present results of a second indoor scene and an outdoor sequence taken from a moving electric cart. The cart-outdoor sequence contains a very long travel around a building, and the accumulated motion estimation error is corrected when loops are detected (Fig. 6 (b)). The cart-indoor sequence is also very interesting since the depth range of the tracked features ranges from very close to far, as well as the sequence contains a significant number of loops. Globally metric mapping methods have difficulties when many loop closures occur within a short time, since there is not enough time to update the map before next loop closes. Our proposed method keeps the local geometry in the map to be metric and correct all the time, whereas the global metric property is improved as global adjustment progresses.

The timing of each of the modules is shown in the bottom row of Fig. 6. The colored pixels represent when the module was running and how long. Each pixel in the image represents 20ms and one column corresponds to one second. The system is running on a laptop computer with a 2.66GHz Core i7 CPU and a GeForce 330M GPU. The input frames are provided to the system at 20 fps, which is 30∼60% faster than the original frame rate. The Scene Flow module is able to process them in real-time, and as can be seen in Fig. 6, Location Recognition and Local Adjustment also perform in real-time. The Global Adjustment module slows down as more keyframes and landmarks are added to the system, but still each iteration runs within 10 seconds even at the end of the sequences.

6. Discussion and Future Work

In this paper we present a novel method that bridges the topological map representation and the metric property in Euclidean maps. While keeping the benefit of an efficient representation of the topological maps allowing instant loop closing, the method tries to enforce the metric property locally and over entire map. As a result, the system is able to maintain the nearby keyframes and landmarks around the current location always accurate and metrically correct. The global map is also optimized to model the environment correctly while maintaining the metric property, and the proposed segment-based iterative optimization is used for efficient processing.

One interesting problem is to keep the number of keyframes and landmarks within a manageable range by merging or trimming redundant map entries. This would allow the map size to be proportional to the size of the space and not to the time spent in the space.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009. 3489

[2] A.George. Nested dissection of a regular finite element mesh. In *SIAM Journal on Numerical Analysis*, volume 10, pages 345–363, 1973. 3490

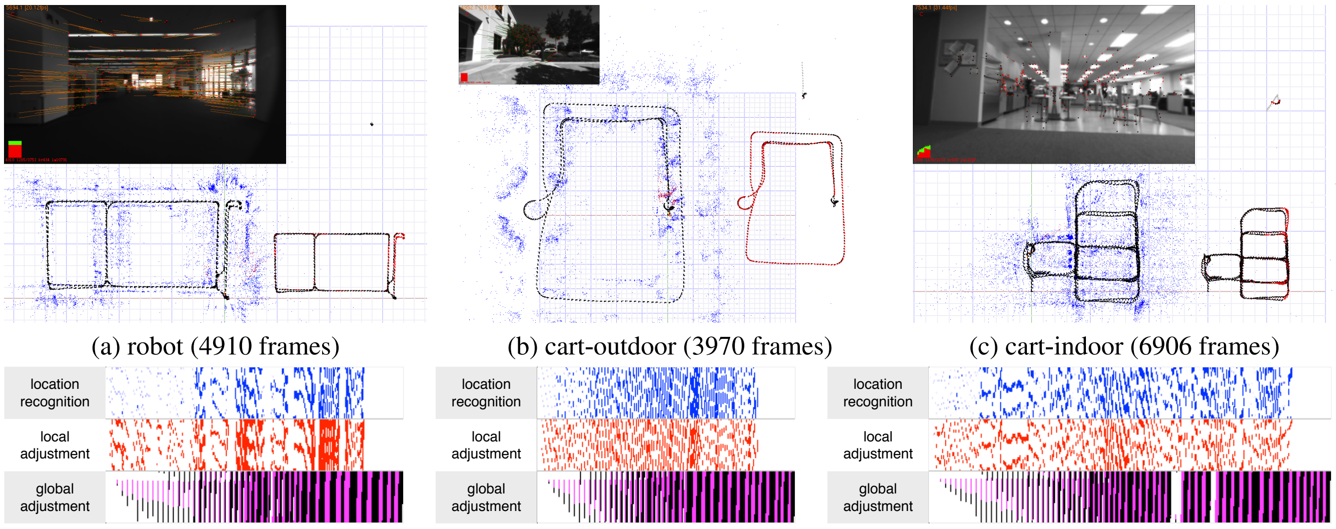


Figure 6. Reconstructed environment maps and timing charts. (Top) The large map directly under the video frame represents the embedding with respect to the current location (size of the finest grid = 2×2m). Beside the large map, the optimization result from Global Adjustment is shown (black is initial keyframes and red is adjusted keyframes). The result from Local Adjustment is shown on the right side of the frame. (Bottom) Timing charts for the sequences are shown. Three rows visualize the time consumed by each module. One pixel in the chart represents 20ms and one column corresponds to 1 second. In global adjustment row, magenta represents segment-wise optimization and black is for global segment optimization. We let the Global Adjustment run 5 iterations after the input sequence ends. It can be seen that Location Recognition and Local Adjustment runs real-time throughout the sequences, and Global Adjustment gets delayed as more keyframes and landmarks are added, but still runs reasonably fast.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 110(3):346–359, 2008. 3494

[4] J.-L. Blanco, J.-A. Fernández-Madrigal, and J. González. Towards a unified bayesian approach to hybrid metric-topological slam. *IEEE Transactions on Robotics*, 24(2):259–270, 2008. 3491

[5] D. C. Brown. The bundle adjsutment - progress and prospects. In *Archives of Photogrammetry*, volume 21, 1976. 3490

[6] M. Z. Brown, D. Burschka, and G. D. Hager. Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 25(8):993–1008, AUGUST 2003. 3494

[7] D. N. Chris Engels, Henrik Stewenius. Bundle adjustment rules. In *PCV*, 2006. 3490

[8] B. Clipp, J. Lim, J.-M. Frahm, and M. Pollefeys. Parallel, real-time visual slam. In *IROS*, 2010. 3489, 3490, 3491

[9] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *TPAMI*, 29(6):1052–1067, 2007. 3489

[10] J.-M. F. et al. Building Rome in a Cloudless Day. In *ECCV*, 2010. 3489

[11] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *IJCV*, 22(2):125–140, 1997. 3494

[12] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004. 3490

[13] S. Holmes, G. Sibley, G. Klein, and D. W. Murray. A relative frame representation for fixed-time bundle adjustment in sfm. In *IEEE International Conference on Robotics and Automation*, pages 2631–2636, 2009. 3491

[14] K. Konolige and M. Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, 2008. 3489, 3491

[15] R. Lipton and R. Tarjan. Generalized nested dissection. *SIAM Journal on Applied Mathematics*, 16(2):346–358, 1979. 3490

[16] M. Lourakis and A. Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the Levenberg-Marquardt algorithm. Technical Report 340, Institute of Computer Science - FORTH, 2004. 3492

[17] M. I. A. Lourakis and A. A. Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Trans. Math. Software*, 36(1):1–30, 2009. 3494

[18] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, pages 1151–1156, 1981. 3494

[19] K. Ni and F. Dellaert. Multi-level submap based slam using nested dissection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010. 3490, 3493

[20] K. Ni, D. Steedly, and F. Dellaert. Out-of-core bundle adjustment for large-scale 3d reconstruction. In *ICCV*, 2007. 3490, 3493

[21] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1), 2006. 3489

[22] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006. 3494

[23] M. Pollefeys, L. V. Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *IJCV*, 2004. 3489, 3490

[24] G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009. 3491, 3492

[25] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH*, pages 835–846, 2006. 3490

[26] N. Snavely, S. M. Seitz, and R. Szeliski. Skeletal sets for efficient structure from motion. In *CVPR*, 2008. 3490

[27] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle Adjustment A Modern Synthesis. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, volume 1883 of *LNCS*, pages 298–372. Proc. of the Intl. Workshop on Vision Algorithms: Theory and Practice, Springer-Verlag, 2000. 3490

[28] J. Xiao and M. Shah. Two-frame wide baseline matching. In *In Proceedings of the International Conference on Computer Vision*, pages 603–609, 2003. 3494