2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Efficient Multi-Drive Map Optimization towards Life-long Localization using Surround View

Marc Sons
FZI Research Center for Information Technology
Karlsruhe, Germany
sons@fzi.de

Christoph Stiller
Institute of Measurement and Control Systems
Karlsruhe Institute of Technology(KIT)
Karlsruhe, Germany

***Abstract*—Current vision-based localization approaches enable reliable positioning in areas where global navigation satellite systems (GNSS) fail due to multipath and shadowing effects. These approaches require an up-to-date map. It seems promising to update such maps iteratively after passing the mapped area again. However, bundling more and more passes into the existing map leads to unbounded computation and memory complexity.**
Herein we propose an iterative optimization approach to create highly accurate maps comprising any number of drives with constant computation complexity. The optimization bases on keypoint correspondences matched between the recorded images from multiple drives. First, each new drive is reconstructed separately by a sliding window bundle-adjustment. Thereafter, the estimated trajectory is divided into disjoint clusters. To align the new drive to the current map, we optimize pairs of clusters which are interconnected through loop-closure or inter-drive correspondences. We derive pose differences from all clusters to estimate the final map poses. For global accuracy, we add GNSS measurements from a low cost receiver. We show in our experiments that the approach enables a joint estimate of the trajectories and landmarks from numerous city-scaled passes within several hours on desktop computers.

I. Introduction

A multitude of applications, e.g. object-fusion, scene understanding or trajectory planning heavily rely on precise localization within previously created maps. Commonly, inertial measurement units (IMU) are coupled with GNSS to localize the ego vehicle position. These approaches are unreliable, inaccurate or fail entirely in many (sub-)urban scenarios, e.g. street canyons, forests or tunnels. Recent vision-based approaches [1]–[4] upon previously created maps enable accurate localization even when GNSS data is not available. Despite Vision-based Simultaneous Localization and Mapping (VSLAM) [5] approaches, it has been established to estimate only the egopose in a previously acquired map for localization in city-scaled areas. Through this, the enormous complexity overhead of real-time mapping is avoided. However, environmental changes, e.g. different seasons, construction sites or parking vehicles lead to obsolescence of a once created map. Hence, it is strongly required to keep maps up-to-date to preserve reliable, robust and accurate localization over extended periods of time. Recent approaches towards lifelong vision based localization [6]–[9] focus on iteratively updated maps and the selection of landmarks which is indeed a challenging and important step in the mapping pipeline. By that, the size of the map is kept bounded while preserving reliable localization in different environmental conditions. However, even using landmark selection, the complexity of the adjustment grows unboundedly when more and more drives are added.
In this work, we focus on the joint reconstruction of the trajectories and landmarks of multiple drives within iterative mapping pipelines. We present an approach which guarantees constant computation time independent of the number of

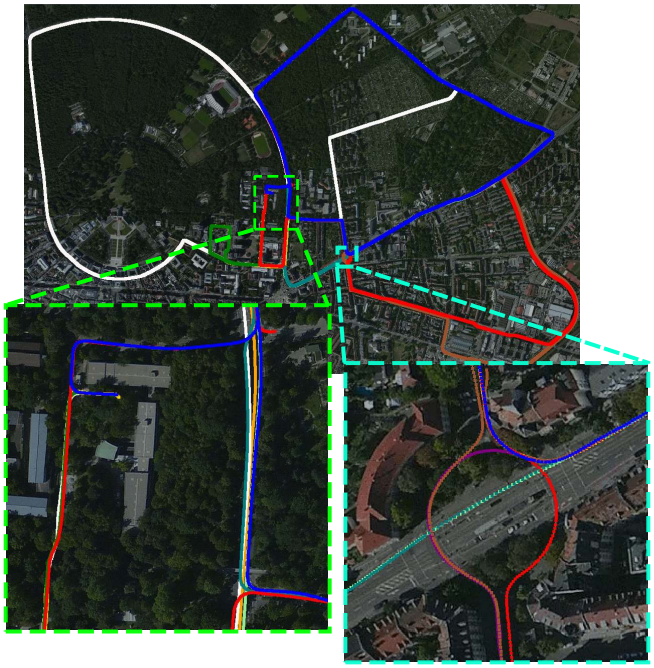

Figure 1: Depiction of jointly estimated trajectories of 14 passes through (sub-)urban area which partly or entirely overlap. Each pass is highlighted in a different color. In total, the map comprises 70k vehicle frames. The overall driven distance is about 69km. The map is generated fully automatically out of $\sim$ 194k images within 15 hours on a desktop computer without any feature selection except outlier removal. To achieve global referencing we involve $\sim$ 18k low quality GNSS measurements.

978-1-7281-0323-5/18/$31.00 ©2018 IEEE

mapped drives and of any feature selection. For that, we divide all trajectories into disjoint clusters and interconnect those clusters which provide a sufficient number of interconnecting keypoint correspondences. To obtain these correspondences, we use the localization estimate of the new drive within the current map and match keypoints between existing map frames and frames of the drive which show the same scene. By that, combinations of interconnected clusters can be optimized independently of all other clusters which guarantees constant complexity of each subproblem and enables parallel processing. Since the number of possible combinations of clusters may grow with each new pass, we optimize only a subset of all possible combinations.

As a preprocessing step, we apply subsequent visual odometry or, if available, use another odometry source to initialize the trajectory of the new pass which is refined by a sliding window bundle-adjustment. Finally, we derive pose differences from all optimized clusters and solve a posegraph optimization problem using only the derived pose differences. This optimization scales to the number of pose differences instead to the significantly higher number of landmarks of a full-bundle-adjustment and, hence, enables to jointly estimate the trajectories of multiple city-scaled drives. Since the pose difference constraints are derived from the estimates of the previous submap optimizations, the resulting estimate has almost the same accuracy as a full bundle-adjustment.

If available, we further constraint this optimization by measurements of a low cost GNSS sensor to achieve a rough geo-referencing and large scale accuracy. Since the results of all costly optimizations are propagated into pose differences which are stored as preliminary results, our pipeline enables to re-estimate the entire map with minor effort, e.g. if the global reference changes afterwards.

In our experiments, we present an iteratively generated multi-drive map which was created over night on a desktop computer. We neither need support from a high-end GPU nor from a computation cluster to optimize numerous city scaled passes within a few hours (see Fig. 1). We underlay aerial images to our estimated trajectories which shows an accurate global alignment. To show the local accuracy of our approach, we compare our results to the optimal estimate of a full bundle-adjustment comprising all passes.

II. Related Work

Using previously built feature maps for vision-based localization in city-scaled areas gained importance in recent years. Lategahn et. al [3] proposed a feature-based approach to localize the ego vehicle precisely in orientation and position using a single session map. The map was created from a stereo camera while localization was performed using a monocular camera. Upon this, we propose a variant which extends the approach to use multiple cameras for localization and mapping [2]. We presented a novel map scheme which allows to access mapped features efficiently toward surround view. We refer to this work for details to the camera setup, vision frontend, feature matching and the localization used in the remainder of this work.

A major work on vision based multi-session mapping towards lifelong localization was proposed by Mühlfellner et al [6]. The mapping process is similar to our approach. After bundling all sessions they created a compressed summary map for online localization. They further showed detailed long-term evaluations of the localization using multi-session maps. For that, they passed a parking area and a mid-size inner-city area numerous times over approximately a year including all seasons and daytimes. To adjust all passes they use a high-performance computation cluster. A further approach is from Schneider et. al [10] who presented `maplab`, a framework for visual-inertial mapping and localization. The approach combines aspects of SLAM and overnight multisession mapping. Their presented experiments show accurate maps created from handheld devices including a monocular camera and an inertial measurement unit.

Beside iterative mapping, numerous work on smoothing and Mapping (SAM) [11] can be found from which our work is inspired. Ni et. al [12] followed the divide-and-conquer idea towards map adjustment. They showed that the linearization of submaps can be cached and reused while combining them to the overall map. In difference, we explicitly split the optimization and merge them afterwards iteratively in terms of pose differences which renders our pipeline more flexible.

III. Multi-Drive Mapping

This section describes our mapping pipeline including all optimization steps. The general process is similar to [6]. While passing a mapped area again, the current map is used for online localization (see [2]). All images and localization estimates are stored persistently to bundle the drive into the existing map as a post-processing step. A new map estimate is generated when the passed area was not mapped before.

Detailed information about the vision frontend, the camera setup and the surround view localization can be found in our previous work [2]. All cameras are jointly triggered and calibrated [13]. The calibration provides a transformation for each camera k referring to a rig reference frame and a projection map

$$\pi_k(\mathbf{l}) = \mathbf{z}, \tag{1}$$

which maps a point $\mathbf{l} \in \mathbb{R}^3$ from real-world to a point $\mathbf{z} \in \mathbb{R}^2$ in the image plane. Since all mounted cameras take images at equal timestamps, we assume that a set of images recorded at a particular time refers to a single vehicle frame $\mathbf{p} \in SE(3)$. Each landmark $\mathbf{l}_i \in L$ is described by a set of matched features $Z_i = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\}$ which in general may origin from images from different cameras, passes and timestamps. We add a connecting edge $e_{i,j}$ between two vehicle frames $\mathbf{p}_i, \mathbf{p}_j \in P$ to our topology database when the related image sets provide a sufficient number of interconnecting matches. We denote the set of all stored edges as E and the set of all vehicle frames spanning the map as P. The following sections describe each step of the optimization pipeline in more detail.

A. Single-Drive Optimization

In a first step, a rough trajectory estimate of the new drive is achieved from odometry. Our framework allows the use of

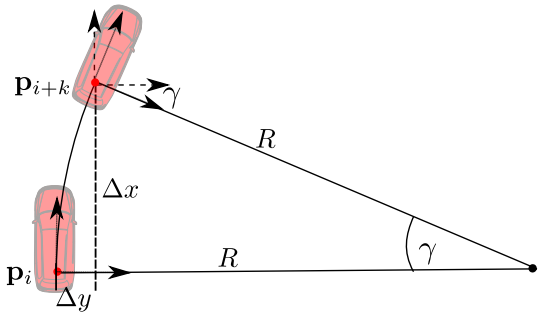

Figure 2: Geometric relations of instantaneous-center-of-rotation constraint which is used as vehicle model.

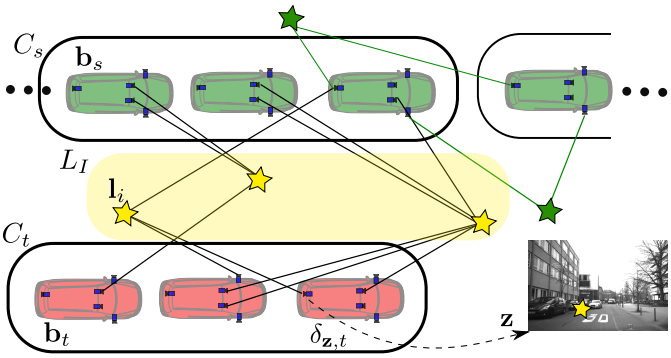

Figure 3: Schematic depiction of two interconnected clusters C_s and C_t of different drives (red, green). The set L_I of yellow stars depict landmarks related to interconnecting correspondences. Optimization of (4) yields $\hat{\mathbf{b}}_s$ and $\hat{L}_I$. In case of less reliable landmarks within L_I, interconnecting landmarks to the previous and next subsequent cluster (green stars) can be added which are fix during optimization to prevent the optimizer converging to wrong minima.

arbitrary odometry sources as long as an extrinsic calibration between the odometry source frame and the camera rig is available. We mainly use stereo visual odometry [14] or surround view visual odometry [15] which doesn't require any overlap of the fields of view between the different cameras. Based on the odometry estimate, we subsample subsequent vehicle poses equally distributed with a defined minimum distance to each other. Thereafter, we estimate a landmark $\mathbf{l}_i \in L$ for each correspondence Z_i through triangulation [13] using the odometry estimate and the camera calibration. This renders an initial solution for the optimization of problem

$$\arg\min_{P,L} \sum_{i=1}^{|L|} \sum_{\mathbf{z}\in Z_i} \left\| \pi_k(\mathbf{p}_\mathbf{z}^{-1}\mathbf{l}_i) - \mathbf{z} \right\|^2_{\Omega_\mathbf{z}} \tag{2}$$

whose optimal set of map frames $\hat{P}$ and landmarks $\hat{L}$ best explain the measurements $\{Z_1,\dots,Z_N\}$. Index k denotes the camera in which feature $\mathbf{z}$ was observed, $\mathbf{p}_\mathbf{z}$ the related vehicle frame and $\Omega_\mathbf{z}$ the covariance matrix of $\mathbf{z}$. To ensure that problem (2) can always be solved in feasible time we divide the problems in spatial overlapping windows. We refer to [2] for details of the single drive adjustment. We slightly modify the overlapping cluster to sliding window adjustment which renders a solution closer to the theoretically optimum. Additionally, we add instantaneous-center-of-rotation costs

$$c_{icr}(\Delta y, \Delta x, \gamma) = \left| \Delta y - \Delta x \frac{1-\cos(\gamma)}{\sin(\gamma)} \right| \tag{3}$$

which penalize subsequent vehicle frames when they deviate from driving along instantaneous circles. The angle γ denotes the difference yaw-angle and Δx and Δy the longitudinal and lateral driven distance respectively (see Fig. 2). This prevents the optimizer to run into wrong minima in cases where many matched keypoints occur on moving objects, e.g. while driving through canyons of trucks or on objects which pass a crossing in front of the ego vehicle.

We apply Cauchy-loss functions to robustify the optimization and, additionally, remove correspondences with to high reprojection errors before and after the optimization of each window.

Additionally, we use vision-based place recognition [16] or, if available, GNSS measurements to find proposals for loop-closuring frames. To ensure that these frames are actually loop-closures, we create local temporary localization maps and localize subsequent frames around the proposal within the local map. We only assume a loop-closure when several subsequent frames were localized successfully. Based on the loop-closure localization, we match keypoints between those images which depicts the same scene from a similar point of view.

After performing the aforementioned single track optimization steps, we only store the inlier correspondences and, additionally, subsequent and loop-closure pose differences $\Delta_{i\to j} = \mathbf{p}_i^{-1} \otimes \mathbf{p}_j$ where the $\otimes : SE(3) \times SE(3) \to SE(3)$-operator concatenates two affine transformations. Thereby, we derive pose differences $\Delta_{i\to j}$ between those frames $\mathbf{p}_i$, $\mathbf{p}_j$ which are connected by an edge $e_{i,j}$.

B. Inter-Drive Optimization

To align landmarks and vehicle frames of a new drive with the current map, we match features which are already stored in the map with features of inlier correspondences of the new drive. Based on the localization, we pair camera frames that show the same scene from a similar point of view and match keypoints between the related images. These matches enable an accurate alignment of the new drive with the existing map trajectories using (2). However, to divide the complexity of this problem into feasible portions, we insert subsequent vehicle frames into disjoint cluster C and interconnect two clusters C_s, C_t when the number of frame connections $\left|\{e_{i,j} | \mathbf{p}_i \in C_s, \mathbf{p}_j \in C_t\}\right|$ between them is larger than a certain threshold. In order to jointly estimate the frames and landmarks within and between interconnected clusters, we define the first frame of each cluster C_v as baseframe $\mathbf{b}_v$ and optimize the problem

$$\arg\min_{L_I, \mathbf{b}_{\{v\}}} \sum_{i=1}^{|L_I|} \sum_{\mathbf{z}\in Z_i} \left\| \pi_k((\mathbf{b}_v \otimes \delta_{\mathbf{z},v})^{-1} \cdot \mathbf{l}_i) - \mathbf{z} \right\|^2_{\Omega_\mathbf{z}} \tag{4}$$

where the keypoint $\mathbf{z}$ origins from one of the images recorded at vehicle frame $\mathbf{p}_{\mathbf{z},v}$ and L_I denotes the set of interconnecting

landmarks. Thereby, v refers to cluster C_v which contains vehicle frame $\mathbf{p}_{\mathbf{z},v} = \mathbf{b}_v \otimes \delta_{\mathbf{z},v}$, where $\delta_{\mathbf{z},v}$ represents $\mathbf{p}_{\mathbf{z},v}$ relative to frame $\mathbf{b}_v$. Furthermore, the baseframe of the first involved cluster is kept fix during optimization (see Fig. 3). In case of a small number $|L_I|$ or many outlier correspondences it is advantageous to add interconnecting landmarks and correspondences of the subsequent previous and next cluster of C_v to problem (4) and keep them fix so that $\mathbf{b}_v$ cannot converge to wrong minima (see Fig. 3). Since we generate only connections e between frames which have sufficient matches and remove rough outliers, we found that adjusting clusters pairwise yields almost equal results as combining more clusters with superior problem complexity. As in the single drive optimization, we use Cauchy-loss functions and pre- and post-select outlier landmarks. Finally, we extract and store pose differences $\Delta_{i\to j}$ along all inter-cluster edges $e_{i,j}$. In general, whenever an already mapped area is passed again and added to the map, the number of interconnections between the clusters in this area increases and, hence, also the number of possible cluster combinations. Therefore, we bound the number of interconnections of each cluster C_s to N and pair C_s only with the $\{C_t,\dots,C_{t+n}\}$, $n \leq N$ most connected clusters in terms of the number of interconnecting landmarks $|L_I|$.

C. Pose Optimization

Finally, to estimate the vehicle frames $\mathbf{p} \in P$ of all mapped drives jointly, we optimize a posegraph based on the derived pose differences from the previous optimizations. The pose differences propagate the high local accuracy into the final optimization which scales to the number of pose differences instead to the significantly higher number of landmarks. This enables a joint estimate of multiple city-scaled drives which achieves almost the accuracy of a full bundle-adjustment. Details of the pose difference optimization can be found in [17]. Since the number of edges $e_{i,j}$ connecting the vehicle frame $\mathbf{p}_i$ to other frames depends on the number of matches between the related image sets, $\mathbf{p}_i$ is usually connected to numerous other frames in the local area. Hence, we only add pose difference measurements $\Delta_{i,j}$ of a subset of all edges $e_{i,j} \in E$ to the optimization. For each $\mathbf{p}_i \in P$, we only add the subsequent differences $\Delta_{i-1,i}$, $\Delta_{i,i+1}$, the loop-closure difference $\Delta_{i,l}$ to the spatial nearest pose $\mathbf{p}_l \in P$ and one inter-drive difference $\Delta_{i,j}$ to the nearest $\mathbf{p}_j \in P$ of a different drive of each connected drive (see Fig. 4). For initialization, we integrate all involved frames along the pose differences using breath-first-search traversal. To robustify the optimization, we use posegraph relaxation methods as proposed in [18].

If available, we add GNSS measurements from a low cost receiver. For that, we transfer the measured GNSS-coordinates into metric UTM-coordinates $\mathbf{t}_{UTM} \in \mathbb{R}^3$ and add further constraints

$$\mathbf{c}_{gnss} = \mathbf{t}_i - \mathbf{t}_{UTM} \tag{5}$$

to the optimization problem. We add one constraint for each vehicle frame $\mathbf{p}_i$ that is approximately synchronize in time to a GNSS measurement, where $\mathbf{t}_i \in \mathbb{R}^3$ denotes the positional component of $\mathbf{p}_i$. We also relax all added reference constraints to deal with highly inaccurate GNSS measurements.

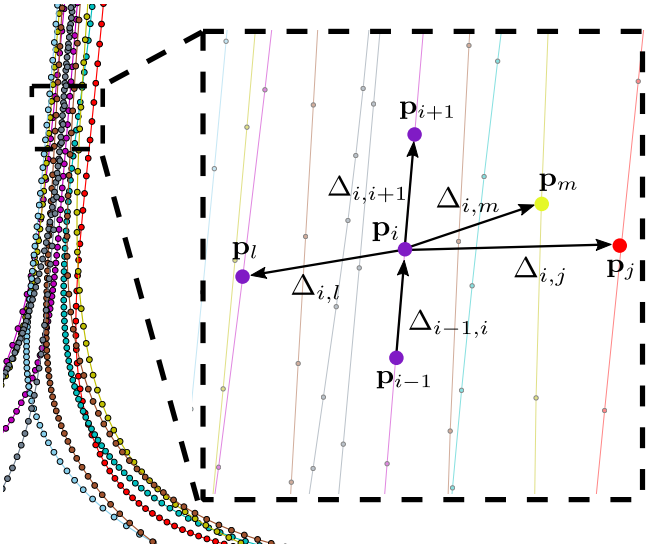

Figure 4: Detailed view of a mapped t-crossing that was passed 7 times (colored trajectories) from all directions. Due to the surround view camera setup, almost all poses (points) in the zoomed area are pairwise interconnected. To reduce the complexity of the final posegraph optimization, we add only pose differences $\Delta_{i,j}$ of a subset (highlighted as arrows) of all edges connected to $\mathbf{p}_i$.

We adjust the map frames in an iterative manner. Hence, we keep all already existing vehicle frames in the map fix and optimize only the frames of the new drive. However, since all costly optimization step results are stored in terms of pose differences, a re-optimization of all mapped frames can be achieved within a few minutes due to the benign scalability compared to a full bundle-adjustment.

Finally, the landmarks of all stored inlier correspondences are triangulated using the final vehicle frames. After the triangulation, we again check the back-projection error and remove outlier landmarks. However, we notice in our experiments that the number of removed landmarks is very low which emphasises the accuracy of the final estimate.

IV. Experiments

We demonstrate the capabilities of our mapping approach by analyzing the resulting map generated from 14 different drives through partly urban and suburban area. The explicit intention of these experiments is to demonstrate the efficiency of our pipeline in terms of reconstruction accuracy and runtime. Hence, we do not apply any landmark selection except outlier removal during the optimization steps.

All drives were recorded during 6 months from November to April with varying environmental conditions and daytimes. All drives overlap partly or complete (see Fig. 1). The map was created from recorded images of three cameras mounted on our experimental vehicle (two front-sided, one rear-sided) and position measurements from a low cost GNSS

receiver. We triggered all cameras jointly with a rate of 10Hz. The image resolution was approximately 0.5 megapixel after rectification. The GNSS measurements were recorded with a UBLOX-M8P[1] receiver.

A. Runtime analysis

In total, we mapped about 69 driven kilometers using $\sim$ 194k images and $\sim$ 18k GNSS recordings. After bundling all passes iteratively, the map comprised $\sim$ 70k vehicle frames after 0.1m distance resampling, and 16 million landmarks with 90 million related keypoints after removing all identified outliers[2]. The overall processing time to iteratively bundle all 14 drives was 15 hours using 16 Intel-i7 cores, 64 Gb RAM and a Solid-State-Drive (SSD) hard-disk. Our pipeline doesn't require any GPU, however, some processing steps could be improved in runtime using a GPU. We used `ROS` [19] for online localization and sensor data recording. To store metadata, and the output of intermediate processing steps, we use `SQLite`[3] with which serialized data can be persistently stored within an embedded SQL-database. All optimization problems were solved using `Ceres` [20]; a C++ library to efficiently solve large non-linear least-squares problems. Multiple optimizations were processed in parallel using `Intel Thread Building Blocks` [21] which significantly decreased the overall runtime. Furthermore, we parallelized the keypoint detection, description and matching. We observed that storing and loading intermediate processing results constituted approximately 30% of the overall processing time although using up-to-date SSD harddisks. This bottleneck, however, can be resolved by using more efficient caching algorithms and reducing the amount of data which must be processed.

The processing time of the optimization steps depends on the number of landmarks detected in the environment. The following statistics concern areas of the map where we drove along street canyons. Such environment provides rich structure which led to a superior amount of landmarks in the surrounding during our experiments. The size of the sliding window of the single-drive optimization was set to 100 frames ($\sim$ 10m) and was shifted 20 frames between two optimization steps. On average, each window comprised $\sim$ 48k landmarks and $\sim$ 340k landmark observations after removing rough outliers beforehand. We removed landmarks with high reprojection error before ($\geq$ 7 pixel) and after ($\geq$ 3 pixel) optimization. The mean processing time of a single window was 1.5 minutes on average while processing multiple optimizations in parallel (see Fig. 5).

We set the maximal number of interconnections of a cluster to $N = 3$ and only combined two clusters, if they had more than 4 interconnecting edges. Thereby, we created an edge $e_{i,j}$ between two vehicle frames if the number of matches between the two related image sets was $\geq$ 300. Since a larger part of the mapped area was passed more than 3 times, almost every generated cluster in this area reached the bound of $N = 3$ interconnections to other clusters. However, the number of possible cluster interconnections was significantly higher. We set the number of vehicle frames per cluster to 40. The mean computation time of optimizing one cluster pair comprising two baseframes and on average $\sim$ 14k interconnecting landmarks was 1.3 minutes. As before, the processing time was measured while running multiple optimizations in parallel fully utilizing the available computation capacity.

The runtime of the optimization of the final posegraph based on the derived pose differences from the preceding optimizations and the GNSS measurements tooks less than 5 minutes in total and, hence, is of minor importance in terms of runtime.

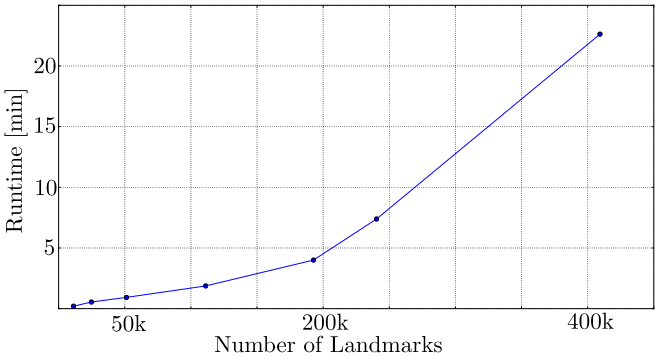

Figure 5: Analysis of the optimization time of problem (2) for different numbers of landmarks. The number of landmarks mainly affects the problem complexity. The chosen window length of 100 vehicle frames correspond to approximately 30k-50k landmarks within the window.

B. Local accuracy analysis

The aforementioned window and cluster sizes were determined empirically[4]. The overall runtime of adding another drive to the map is mainly affected by the selected window and cluster sizes. Hence, we analyzed the achieved local accuracy for different window and cluster sizes. We compared the outcome of our mapping pipeline against the result of a full bundle-adjustment of limited areas of the map since we did not have the required ressources for a joint optimization of the entire map. For that, we estimated all vehicle frames and landmarks from 7 drives within several disjoint limited areas of about 150m length jointly through a full bundle-adjustment. The computation of each full bundle adjustment took several hours and reached the memory limits of our server. We involved no GNSS measurements for these experiments. In comparison, we evaluated our iterative approach within the same areas for window sizes of 40 to 300 vehicle frames and cluster sizes of 10 to 50 vehicle frames. The number of maximal cluster interconnections was set to $N = 3$.

[1] https://www.u-blox.com/en/product/neo-m8p-series

[2] A video of a different map generated with our approach can be found at https://www.mrt.kit.edu/3203.php

[3] https://www.sqlite.org/index.html

[4] The selection of the numerous other parameters (which are not all mentioned here), e.g. the back-projection error thresholds or the window shift length are based on empirical foundings while implementing and testing the components of the framework.

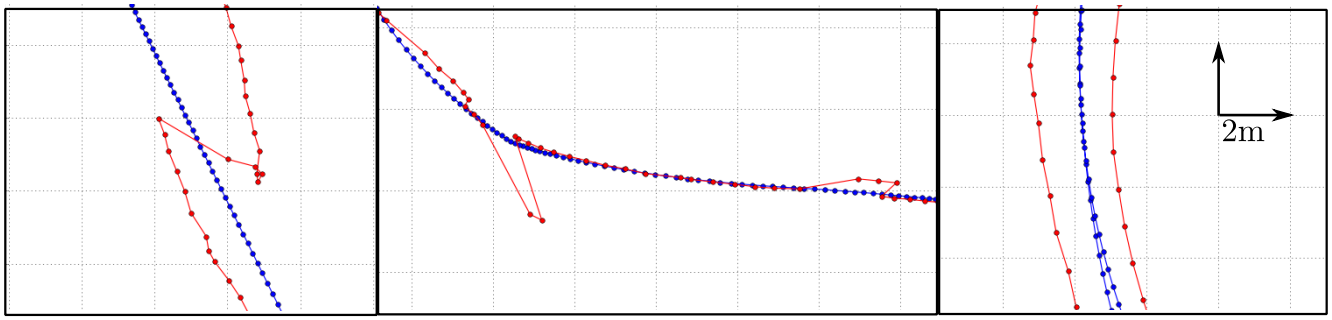

Figure 6: Different types of error of the recordend GNSS data. The blue trajectory depicts our estimated trajectory of one of the mapped drives. The red trajectory depicts corresponding GNSS measurements. Left: Abrupt lateral jump of $\sim$ 3m. Middle: Short sequence of outliers. Right: Gobal drift of $\sim$ 2.5m during a loop-closure caused by atmospherical distortions. Due to the mainly urban environment, the majority of the GNSS recordings suffer frum such errors.

Figure 7: Parts of the mapped trajectories overlayed on aerial images from OpenStreetMap. Since some of the mapped drives passed larger streets on different lanes in different directions, we got no inter-drive matches and, hence, no inter-drive pose differences for the final optimization. Even though the GNSS measurements are highly distorted, averaging them from multiple drives during optimization achieved sufficient global accuracy in such cases.

The achieved time saving was 72.4% on average. To compare both estimates, we evaluated the absolute angle and position differences between corresponding vehicle frames. For all evaluated window and cluster sizes, we found that the differences between the frames of the optimal estimate from the full bundle adjustment and the estimate of our pipeline were less than 0.02m and 0.08° in position and orientation respectively. These negligible differences can be explained through the fact that the majority of all keypoint correspondences covered only a local area which was smaller as the analyzed window sizes and, hence, affected both estimates equally. Less than 1% of all involved correspondences were fragmented by the clustering.

C. Global optimization analysis

Since major parts of the mapped area passed through urban area, the majority of the recorded GNSS measurements showed significant outliers (see Fig. 6). To deal with such measurements, we applied posegraph relaxation to the GNSS cost terms and lowered their weight for the final posegraph optimization by the factor of 100.

Since even our reference DGPS system was not reliable in the mapped area, we evaluated the global-referencing quantitatively by comparing our estimated trajectories with aerial images (see Fig. 1 and Fig. 7). As shown in Fig. 7, the mean global validity of the GNSS measurements enables global accuracy in cases were no inter-drive or loop-closure pose differences could be obtained, e.g. due to large view point differences while driving on the next lane on larger streets in different directions. We observed that stable results could be achieved from multiple passes through the same area even though GNSS accuracy is low.

V. Conclusions

We present an iterative approach to accurately estimate maps from multiple drives for precise life-long localization in (sub-)urban areas using a surround view camera system. We showed how to decompose the increasing map into small portions for optimization which enables to estimate maps of an unbounded area with constant complexity and negligible loss of local accuracy. Since we transfer the achieved accuracy of all costly optimization steps to pose differences, it is possible to recompute the map estimate including all mapped drives within minutes. Additionally, we use GNSS measurements to achieve global accuracy and referencing. The modular structure of our framework allows

us to easily combine our proposed mapping approach with more sophisticated landmark selection and map management strategies.

Our experiments show the capabilities of our framework to iteratively adjust multiple passes of city-scaled areas with millions of landmarks in reasonable time on desktop computers even without removing any landmarks except outliers. We further show that we are capable of achieving sufficient global accuracy even with many erroneous GNSS measurements. We have been running the online localization counterpart which uses the maps generated with our framework for several years and numerous events to demonstrate fully-automated driving within urban areas.

References

[1] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, *et al.*, "Making bertha drive—an autonomous journey on a historic route", *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[2] M. Sons, M. Lauer, C. G. Keller, and C. Stiller, "Mapping and localization using surround view", in *Intelligent Vehicles Symposium (IV), 2017 IEEE*, IEEE, 2017, pp. 1158–1163.

[3] H. Lategahn and C. Stiller, "Vision-only localization", *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1246–1257, 2014.

[4] P. Muehlfellner, P. Furgale, W. Derendarz, and R. Philippsen, "Evaluation of fisheye-camera based visual multi-session localization in a real-world scenario", in *Intelligent Vehicles Symposium (IV), 2013 IEEE*, IEEE, 2013, pp. 57–62.

[5] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part ii", *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[6] P. Mühlfellner, M. Bürki, M. Bosse, W. Derendarz, R. Philippsen, and P. Furgale, "Summary maps for lifelong visual localization", *Journal of Field Robotics*, vol. 33, no. 5, pp. 561–590, 2016.

[7] D. M. Rosen, J. Mason, and J. J. Leonard, "Towards lifelong feature-based mapping in semi-static environments", in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, IEEE, 2016, pp. 1063–1070.

[8] M. Bürki, I. Gilitschenski, E. Stumm, R. Siegwart, and J. Nieto, "Appearance-based landmark selection for efficient long-term visual localization", in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, IEEE, 2016, pp. 4137–4143.

[9] M. Dymczyk, S. Lynen, M. Bosse, and R. Siegwart, "Keep it brief: Scalable creation of compressed localization maps", in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, IEEE, 2015, pp. 2536–2542.

[10] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, *et al.*, "Maplab: An open framework for research in visual-inertial mapping and localization", *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1418–1425, 2018.

[11] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "Isam2: Incremental smoothing and mapping using the bayes tree", *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[12] K. Ni, D. Steedly, and F. Dellaert, "Tectonic sam: Exact, out-of-core, submap-based slam", in *Robotics and Automation, 2007 IEEE International Conference on*, IEEE, 2007, pp. 1678–1685.

[13] T. Strauß, J. Ziegler, and J. Beck, "Calibrating multiple cameras with non-overlapping views using coded checkerboard targets", in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, IEEE, 2014, pp. 2623–2628.

[14] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time", in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, Ieee, 2011, pp. 963–968.

[15] J. Graeter, T. Strauss, and M. Lauer, "Momo: Monocular motion estimation on manifolds", in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2017, pp. 1–6.

[16] H. Lategahn, J. Beck, B. Kitt, and C. Stiller, "How to learn an illumination robust image feature for place recognition", in *Intelligent Vehicles Symposium (IV), 2013 IEEE*, IEEE, 2013, pp. 285–291.

[17] M. Sons, H. Lategahn, C. G. Keller, and C. Stiller, "Multi trajectory pose adjustment for life-long mapping", in *Intelligent Vehicles Symposium (IV), 2015 IEEE*, IEEE, 2015, pp. 901–906.

[18] N. Sünderhauf and P. Protzel, "Switchable constraints for robust pose graph slam", in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, 2012, pp. 1879–1884.

[19] A.-M. Hellmund, S. Wirges, Ö. Ş. Taş, C. Bandera, and N. O. Salscheider, "Robot operating system: A modular software framework for automated driving", in *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, IEEE, 2016, pp. 1564–1570.

[20] S. Agarwal, K. Mierle, *et al.*, *Ceres solver*, http://ceres-solver.org.

[21] J. Reinders, *Intel Threading Building Blocks*, First. Sebastopol, CA, USA: O'Reilly & Associates, Inc., 2007, ISBN: 9780596514808.