2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Monocular Camera Localization in 3D LiDAR Maps

Tim Caselitz Bastian Steder Michael Ruhnke Wolfram Burgard

***Abstract*— Localizing a camera in a given map is essential for vision-based navigation. In contrast to common methods for visual localization that use maps acquired with cameras, we propose a novel approach, which tracks the pose of monocular camera with respect to a given 3D LiDAR map. We employ a visual odometry system based on local bundle adjustment to reconstruct a sparse set of 3D points from image features. These points are continuously matched against the map to track the camera pose in an online fashion. Our approach to visual localization has several advantages. Since it only relies on matching geometry, it is robust to changes in the photometric appearance of the environment. Utilizing panoramic LiDAR maps additionally provides viewpoint invariance. Yet low-cost and lightweight camera sensors are used for tracking. We present real-world experiments demonstrating that our method accurately estimates the 6-DoF camera pose over long trajectories and under varying conditions.**

I. Introduction

Accurate localization is an important prerequisite for many navigation tasks. For example, accurate information about their pose in the environment enables autonomous mobile robots or pedestrians to plan a path to a given goal location. While GPS can provide accurate position estimates at a global scale, it suffers from substantial errors due to multi-path effects in urban canyons and does not provide sufficiently accurate estimates indoors. This is a major drawback, since localization is restricted in populated areas that have a high demand for navigation services, e.g., city centers and shopping malls. A popular approach to mobile robot localization is to match sensor data against a previously acquired map. Many existing methods use the same sensor type for mapping and localization. LiDARs provide accurate range measurements, but are typically expensive and heavy. Cameras are low-cost, lightweight, and widely available, but do not directly provide range information. Our method exploits the advantages of both sensors by using a LiDAR for mapping and a camera for localization. Today, map providers already capture 3D LiDAR data to build large-scale maps. Our method enables people to accurately localize themselves in these maps without being equipped with a LiDAR. Instead, they only require a single monocular camera, which is favorable in many applications. In an autonomous driving context the lower price is highly relevant, for localizing a drone the low weight is a crucial aspect, and in pedestrian localization cameras benefit from their enormous availability since they are integrated in every modern smartphone.

All authors are with the Department of Computer Science, University of Freiburg, Germany. This work has been partially supported by the European Commission under the grant numbers ERC-AG-PE7-267686-LifeNav and FP7-610603-EUROPA2.

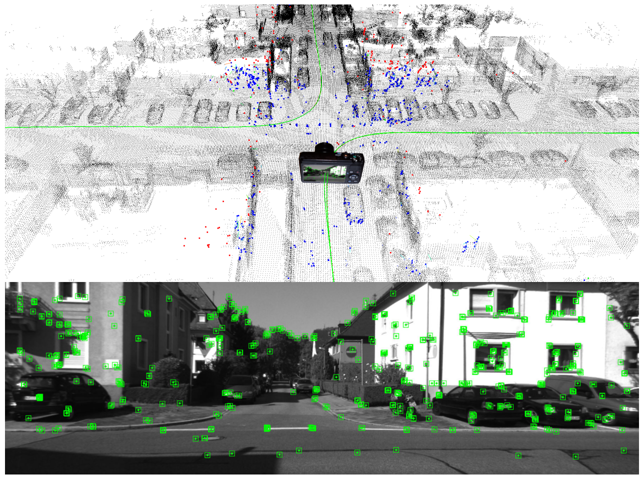
Fig. 1. Our method tracks the 6-DoF camera pose by reconstructing 3D points (blue, red) from image features (bottom, green) and matching them against a given 3D LiDAR map (black). Blue points are estimated to lie inside the map, while red points have no correspondence. The green line represents the camera trajectory.

The key idea of our work is to approach visual localization by matching *geometry*. Since the acquisition of a map can date back considerably compared to the time of localization, it is essential that maps contain time-invariant information. Common methods typically match *photometry*, either in form of feature descriptors or by directly using intensity values. We argue that our approach is advantageous because the geometry of the environment tends to be more stable than its photometric appearance which can change tremendously even over short periods. For example, during the course of one day a place might look substantially different due to varying illumination while its geometric structure has not changed all. In contrast, changes in geometry always influence the photometric appearance of the environment.

In this paper we present a method to track the 6-DoF pose of a monocular camera in a 3D LiDAR map. Since our approach is not intended for global localization, a coarse estimate of the initial pose in this map is required. We employ a visual odometry system based on local bundle adjustment to reconstruct the camera poses relative to a sparse set of 3D points. The main contribution of this work is to continuously align this point set with the given map by matching their geometry and applying an estimated similarity transformation to indirectly localize the camera. We formulate the estimation problem as non-linear least squares error minimization and solve it within an iterative closest point procedure. Fig. 1 shows a camera image and the corresponding pose in the 3D LiDAR map.

II. Related Work

The method proposed in this paper is related to approaches developed in the computer vision and robotics community in the last decades. While vision-based approaches rely on passive cameras, LiDAR sensors that actively measure range information are frequently used in robotics. Both modalities have successfully been used for solving the Simultaneous Localization and Mapping (SLAM) problem [15], [18]. However, the integration of both has mostly been done in the so-called back-end of the SLAM process and not by matching their data directly.

Approaches using LiDARs typically apply scan matching techniques based on variants of the Iterative Closest Point (ICP) algorithm [3], [19] for estimating incremental movements and use feature-based approaches to detect loop closures [20]. Modern methods often rely on graph-based optimization techniques [11], [9] to estimate consistent trajectories and maps. Since LiDARs provide accurate range information and are mostly invariant to lighting conditions, such methods have the potential to create highly accurate maps of indoor and outdoor environments. Localizing a LiDAR within these maps can be achieved with pure scan matching or using filtering approaches [23].

Visual SLAM refers to the problem of using images as only source of information [7]. In contrast to approaches treated under the name Structure from Motion (SfM), e.g. [1], methods signed with the keyword visual SLAM typically target real-time applications. MonoSLAM [5] was the first system achieving real-time performance. It is based on an extended Kalman filter which estimates the cameras pose, velocity and feature positions. Bundle adjustment [24], a global optimization technique also used in SfM, was considered too expensive for real-time operation. This changed with PTAM presented by Klein *et al.* [10]: their key idea was to parallelize camera tracking and map building in order to utilize expensive optimization techniques without forfeiting real-time performance. Strasdat *et al.* [21] later showed that keyframe-based optimization approaches outperform filtering methods in terms of accuracy per computational cost.

Compared to visual SLAM, visual odometry [17] is only concerned with the incremental camera motion. However, *local* bundle adjustment is often performed to reduce drift. In contrast to approaches based on stereo cameras, monocular visual odometry suffers from the unobservability of scale. Strasdat *et al.* [22] point out that scale "is liable to drift over time". This is a crucial insight for our method. One of the key contributions of Strasdat *et al.* [22] is the description of the Lie group and its relation to the Lie algebra of similarity transformations. We rely on this formulation for estimating similarity transformations.

Our method employs visual odometry to track the camera motion and to reconstruct a sparse set of 3D points via local bundle adjustment. For this purpose we rely on components of ORB-SLAM presented by Mur-Artal *et al.* [13]. It is a state-of-the-art solution for monocular SLAM that stands in a line of research with PTAM and is available open-source.

The majority of research concerning visual localization has focused on matching photometric characteristics of the environment. This is either done by comparing image feature descriptors like SIFT [12] or SURF [2] or directly operating on image intensity values. Yet, one of the main issues in visual localization is that the environment's photometric appearance changes substantially over time, especially across seasons. Churchill *et al.* [4] approach this problem by storing multiple image sequences for the same place from different times. Naseer *et al.* [14] tackle the problem by matching trajectories using a similarity matrix.

In contrast to methods based on matching photometry, approaches for camera localization in geometric maps built from LiDAR data are less present in the literature. Wolcott *et al.* [25] proposed a method to localize an autonomous vehicle in urban environments. Using LiDAR intensity values, they render a synthetic view of the mapped ground plane and match it against the camera image by maximizing normalized mutual information. While this approach only provides the 3-DoF pose, the method presented by Pascoe *et al.* [16] estimates the full 6-DoF camera pose. Their appearance prior (map) combines geometric and photometric data and is used to render a view that is then matched against the live image by minimizing the normalized information distance.

Both approaches perform matching in 2D space and therefore require expensive image rendering supported by GPU hardware. Furthermore, their prior comprises LiDAR intensities or visual texture respectively. In contrast, our method relies on geometric information only. By directly matching 3D geometry, we also avoid the need for computations on a GPU. However, we achieve comparable results in terms of accuracy and frame rate.

III. Proposed Method

The objective of our method is to localize a monocular camera in a 3D LiDAR map. The inputs are an image stream and a map that is represented as a point cloud, the output is a 6-DoF camera pose estimate at frame rate. Our approach builds on a visual odometry system that uses local bundle adjustment to reconstruct camera poses and a sparse set of 3D points from image features. Given the camera poses relative to these points, we indirectly localize the camera by aligning the reconstructed points with the map.

The alignment is performed once the visual odometry, which is based on components of ORB-SLAM [13], provides the first local reconstruction consisting of two keyframe poses and a minimal amount of 3D points. Since our approach is not intended for global localization, a coarse estimate for the transformation between this initial reconstruction and the map is required. Subsequently, our method aligns the local reconstruction whenever it is updated, i.e., a new keyframe is selected by the camera tracking and added to the local mapping. Since our method directly operates on the map maintained by ORB-SLAM, the local mapping needs to be paused during the alignment. Therefore, the alignment must run sufficiently fast in order to avoid long interruptions that can cause the camera tracking to fail since newly explored areas are not covered by the map.

By continuously aligning the local reconstruction with the map, we eliminate the drift accumulated by visual odometry. Since we use a *monocular* camera, drift occurs not only in translation and rotation, but also in scale. We therefore realize the alignment with a 7-DoF similarity transformation. In the remainder of this section we first discuss the local reconstruction, afterwards address the data association between local reconstruction and map, and finally detail the optimization problem we solve to estimate the transformation that is applied to perform the alignment. We use homogeneous coordinates to represent rigid-body and similarity transformations:

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix} \in SE(3), \quad \mathbf{S} = \begin{pmatrix} s\mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix} \in Sim(3) \quad (1)$$

with $s \in \mathbb{R}^+$, $\mathbf{R} \in SO(3)$, $\mathbf{t} \in \mathbb{R}^3$. A point $\mathbf{p} \in \mathbb{R}^3$ is denoted $\tilde{\mathbf{p}} = (\mathbf{p}\ 1)^\top$ in homogeneous coordinates. To simplify notation we define $\xi(\tilde{\mathbf{p}}) \coloneqq \mathbf{p}$.

A. Local Reconstruction

The visual odometry system reconstructs keyframe poses and 3D points by solving a local bundle adjustment problem. What is considered *local* is determined by a covisibility graph that describes how many image features are covisible between keyframes. All keyframes directly connected to the current keyframe via this graph and all points that are observed by these keyframes are the variables of the visual odometry's local estimation problem.

We include the same keyframes and points in our local reconstruction which is defined by the two sets

$$\begin{aligned} \mathcal{D} &= \left\{\mathbf{d}_i \mid \mathbf{d}_i \in \mathbb{R}^3, i = 1, \dots, D\right\}, && (2)\\ \mathcal{T} &= \left\{\mathbf{T}_i \mid \mathbf{T}_i \in SE(3), i = 1, \dots, T\right\}. && (3) \end{aligned}$$

Adhering to this choice has two reasons. First, it ensures that we use a consistent local reconstruction because the bundle adjustment optimization is performed right before our alignment. Second, the alignment does not influence and potentially corrupt the visual odometry, since we *uniformly* transform its local bundle adjustment problem.

B. Data Association

To determine the alignment of local reconstructions with the map, we search correspondences between reconstructed points in $\mathcal{D}$ and points of the map

$$\mathcal{M} = \left\{\mathbf{m}_i \mid \mathbf{m}_i \in \mathbb{R}^3, i = 1, \dots, M\right\}. \quad (4)$$

We perform this within an ICP scheme, i.e., update the data associations based on the current similarity transformation estimate over $k = 1, \dots, K$ iterations. For each reconstructed point, we find its nearest neighbor in the map, which is stored in a kd-tree to allow for fast look-up. If the nearest neighbor is close enough, the pair is added to the correspondence set

$$\mathcal{C}_k = \left\{(\mathbf{d}_i, \mathbf{m}_j)\ \forall \mathbf{d}_i \in \mathcal{D} \mid \exists \operatorname*{argmin}_{\mathbf{m}_j \in \mathcal{M}} \|\mathbf{d}_i - \mathbf{m}_j\|_2 < \tau_k\right\}. \quad (5)$$

We reduce the distance threshold τ_k over the iterations k. τ_{max} and τ_{min} are parameters of our method and define the linear threshold function

$$\tau_k = -\frac{\tau_{max} - \tau_{min}}{K} k + \tau_{max}. \quad (6)$$

As the algorithm converges, point-to-point distances are decreasing, and we can choose the threshold more strictly in order to retain only high-quality associations.

A major problem of finding correspondences exclusively based on nearest neighbors is that the set of reconstructed points typically overlaps *only partially* with the map. Therefore points $\mathbf{d}_i$ might have no meaningful correspondence in $\mathcal{M}$ even though their position is perfectly estimated. The partial overlap is often caused by an incomplete mapping due to the LiDAR's limited vertical field of view. Reconstructed points originating from unmapped geometric structure would be associated with boundary points of the map and thus lead to biased transformation estimates. Since these can have severe consequences in terms of accuracy but also convergence, we refine the correspondence set using the map's local point distribution described bellow.

1) Local Point Distribution: In order to analyze its *local* point distribution, the map is voxelized with a resolution Δ depending on its density. We perform a principal component analysis on the covariance matrix $\mathbf{\Sigma}(\mathcal{V}) \in \mathbb{R}^{3\times 3}$ of the points with the mean $\boldsymbol{\mu}(\mathcal{V}) \in \mathbb{R}^3$ inside each voxel

$$\mathcal{V}_{x,y,z} = \left\{\mathbf{m}_i \mid \mathbf{m}_i \in \mathcal{M},\ \Delta \begin{pmatrix} x \\ y \\ z \end{pmatrix} \preceq \mathbf{m}_i \prec \Delta \begin{pmatrix} x+1 \\ y+1 \\ z+1 \end{pmatrix}\right\} \quad (7)$$

to determine the main directions along they are distributed. The operators $\preceq$ and $\prec$ are meant component-wise. The eigenvectors $\mathbf{V}_1, \mathbf{V}_2 \in \mathbb{R}^3$ with the two largest eigenvalues and the orthogonal vector $\mathbf{V}_3 = \mathbf{V}_1 \times \mathbf{V}_2$ allow us to describe the point distribution in a voxel $\varepsilon_{x,y,z} = (\mathbf{T}, \boldsymbol{\sigma}, N)$ with

$$\begin{aligned} \mathbf{T} = &\begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix}^{-1} \text{ with } \mathbf{R} = (\mathbf{V}_1 \mathbf{V}_2 \mathbf{V}_3),\ \mathbf{t} = \boldsymbol{\mu}(\mathcal{V}),\\ \boldsymbol{\sigma} = &\left(\sqrt{a_{11}}\ \sqrt{a_{22}}\ \sqrt{a_{33}}\right)^\top \text{ with } \mathbf{A} = \mathbf{T}\mathbf{\Sigma}(\mathcal{V})\mathbf{T}^{-1},\\ N = &|\mathcal{V}|. \end{aligned} \quad (8)$$

The transformation $\mathbf{T}$ defines the principal components aligned basis of a voxel, the term $\boldsymbol{\sigma}$ represents the standard deviation of the points along these axes, and N denotes the amount of points inside a voxel. Since the local point distribution

$$\begin{aligned} \mathcal{E} = \Big\{&\varepsilon_{x,y,z} = (\mathbf{T}, \boldsymbol{\sigma}, N) \mid \mathbf{T} \in SE(3), \boldsymbol{\sigma} \in \mathbb{R}^3, N \in \mathbb{N},\\ &\begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{Z}^3, \min_{\mathbf{m}_i \in \mathcal{M}} \mathbf{m}_i \preceq \begin{pmatrix} x \\ y \\ z \end{pmatrix} / \Delta \prec \max_{\mathbf{m}_i \in \mathcal{M}} \mathbf{m}_i \Big\} \end{aligned} \quad (9)$$

is constructed offline, its computation does not affect the online performance of our method.

2) Correspondence Refinement: We determine a refined subset of correspondences $\mathcal{C}_k' \subset \mathcal{C}_k$ based on the map's local point distribution. The objective is to filter out correspondences associating points $\mathbf{d}_i$ that are located in areas not covered by or on the boundaries of the map. To ensure proper

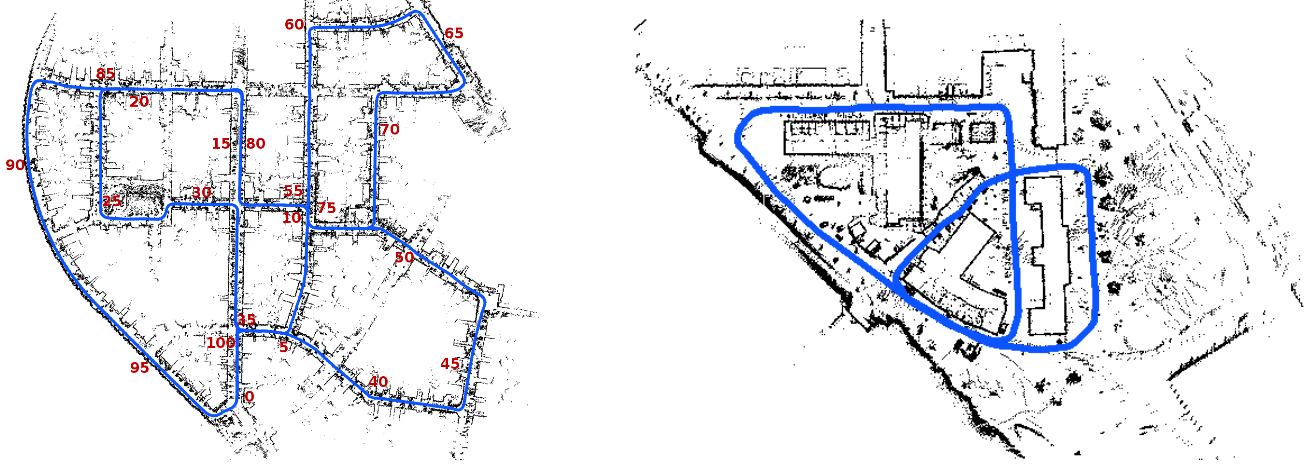

Fig. 2. Top view of the 3D LiDAR maps used in the evaluation (ground removed for visualization). Map points are shown in black, trajectories in blue. Left: KITTI odometry dataset, sequence 00. Red numbers indicate the percentage of the trajectory. Right: Freiburg campus dataset.

coverage, we check if the amount of points in a voxel is sufficient. This criterion is not suited, if points are distributed extremely non-uniformly along a principal component axis, e.g., at map boundaries. Therefore we additionally verify that the point lies inside a multiple standard deviation along the voxel's principle component axes. To smooth discretization effects, we also allow neighboring voxels to fulfill these criteria. This yields the refined correspondence set

$$\mathcal{C}'_k = \Big\{(\mathbf{d}_i, \mathbf{m}_j) \in \mathcal{C}_k \mid \ \Delta\begin{pmatrix} x \\ y \\ z\end{pmatrix} \preceq \mathbf{d}_i \prec \Delta\begin{pmatrix} x+1 \\ y+1 \\ z+1\end{pmatrix}, \bigvee_{-1 \preceq (x,y,z)^\top \preceq +1} \varepsilon_{x,y,z}, N \geq N_{min}, \xi(\mathbf{T}\tilde{\mathbf{d}}_i) \prec N_\sigma \boldsymbol{\sigma} \Big\} \tag{10}$$

where $\bigvee$ is meant as *or* operator used to describe the neighborhood of a voxel.

C. Alignment

Given a set of correspondences, we estimate a similarity transformation that aligns the local reconstruction with the map. The set of refined correspondences $\mathcal{C}'_k$ is updated iteratively. In each iteration k we estimate

$$\mathbf{S}^*_k = \underset{\mathbf{S} \in Sim(3)}{\operatorname{argmin}} \ F_{Data}(\mathbf{S}, \mathcal{C}'_k) \tag{11}$$

by solving a non-linear least squares minimization problem with g2o [11] using the Levenberg-Marquardt algorithm.

We estimate the transformation $\mathbf{S}^*_k$ in the reference frame of the current keyframe $\mathbf{T}_c$. This is advantageous compared to a parametrization in the reference frame of the map because the optimization variables better correspond to the dimensions of drift which we want to compensate with $\mathbf{S}^*_k$. Therefore we transform the refined correspondence set and get

$$\mathcal{C}''_k = \Big\{\big(\xi(\mathbf{T}_c\tilde{\mathbf{d}}_i), \xi(\mathbf{T}_c\tilde{\mathbf{m}}_j)\big) \mid (\mathbf{d}_i, \mathbf{m}_j) \in \mathcal{C}'_k\Big\}. \tag{12}$$

Our error function is the squared Euclidean distance between corresponding points:

$$F_{Data}(\mathbf{S}, \mathcal{C}''_k) = \sum_{\mathcal{C}''_k} \rho\left(\mathbf{e}^\top_{Data}\mathbf{e}_{Data}\right), \tag{13}$$

$$\mathbf{e}_{Data}(\mathbf{S}, \mathbf{d}_i, \mathbf{m}_j) = \xi(\mathbf{S}\tilde{\mathbf{d}}_i) - \mathbf{m}_j. \tag{14}$$

$$\rho(r) = \begin{cases} r^2 & r < r_\Delta \\ 2r_\Delta|r| - r^2_\Delta & \text{otherwise} \end{cases} \tag{15}$$

We use a Huber cost function to be more robust against outliers in the data association. As shown by Fitzgibbon [6] this leads to better convergence properties of the objective function. We choose the kernel size $r_\Delta = \tau_{min}$ to retain the quadratic error range over all iterations k.

Once we estimated $\mathbf{S}^*_k$ for all iterations, we compute a joint similarity transformation

$$\mathbf{S}^* = \prod_{k=0}^{K-1} \mathbf{S}^*_{K-k}. \tag{16}$$

To align the local reconstruction with the map, we transform all point positions $\mathbf{d}_i$ and keyframe poses $\mathbf{T}_i$ (as defined in III-A) with $\mathbf{S}^*$. Since it is in the reference frame of the current keyframe it has to be transformed back into the reference frame of the map before it is applied:

$$\mathcal{D}' = \{\mathbf{d}'_i = \xi(\mathbf{T}_c^{-1}\mathbf{S}^*\mathbf{T}_c\tilde{\mathbf{d}}_i) \ \forall \ \mathbf{d}_i \in \mathcal{D}\}, \tag{17}$$

$$\mathcal{T}' = \{\mathbf{T}'_i = \mathbf{T}_c^{-1}\mathbf{S}^*\mathbf{T}_c\mathbf{T}_i \ \forall \ \mathbf{T}_i \in \mathcal{T}\}. \tag{18}$$

IV. Experimental Evaluation

We evaluated our method in two different real-world experiments. First, we investigated the accuracy of our method on the publicly available KITTI odometry dataset [8]. Second, we performed experiments under varying conditions to support the claimed advantages of our approach.

Our method requires a reasonably good initialization. If a ground truth trajectory is available, we use it to interpolate

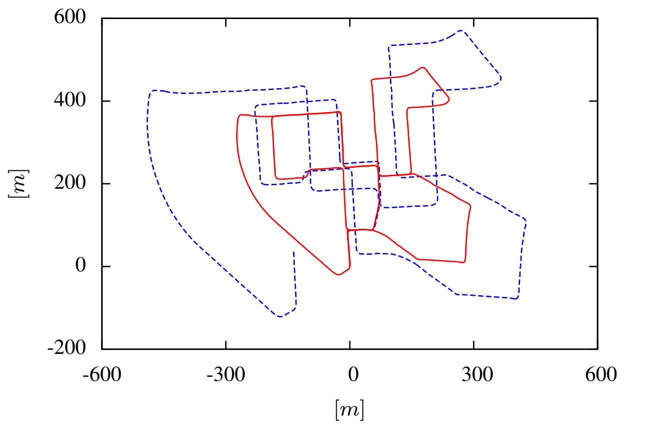

Fig. 3. Trajectories on sequence 00 of the KITTI odometry dataset. The visual odometry (dashed blue line) accumulates drift over time. Our method (solid red line) corrects this drift and provides a consistent trajectory.

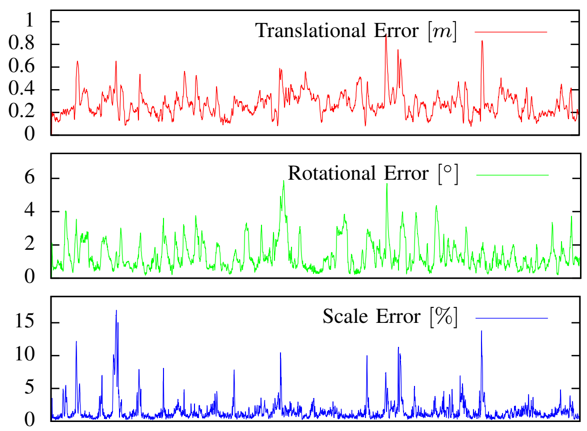

Fig. 4. Mean errors of our method averaged over 10 runs on sequence 00 of the KITTI odometry dataset. The x-axis represents the percentage of the trajectory as shown in Fig. 2 (left).

the poses of the first two keyframes. *Two* poses are necessary to infer the initial scale. For datasets without ground truth, a coarse initial pose and scale estimate must be determined in another way. We do this by manually aligning a reconstructed set of points with the 3D LiDAR map.

The same set of parameters was used throughout the evaluation. The visual odometry extracts 2500 features per image and keeps the ORB-SLAM default values for other parameters. The number of iterations used for estimating the alignment is $K = 10$, the parameters for the correspondence distance threshold are $\tau_{max} = 2m$ and $\tau_{min} = 1m$. The local point distribution is voxelized at a resolution of $\Delta = 1m$, thresholds for the refinement are $N_{min} = 10$ and $N_\sigma = 3$. All experiments were carried out on a i7-4790K CPU without support of GPU hardware. Given this configuration, our method runs at approximately 10 fps.

A. Evaluation of Accuracy

In order to evaluate the accuracy of our method and to allow for easy comparison, we tested it on sequence 00 of the KITTI odometry dataset. Since the provided poses lack consistency in loop closure areas, the ground truth trajectory was obtained by a LiDAR-based SLAM system. We used the resulting trajectory and the provided 3D LiDAR data to build a map at a resolution of 20 cm (see Fig. 1 and Fig. 2 left). We employed this map to track the pose of camera 0 on the vehicle. Since the relative transformation between camera and LiDAR is known by calibration, the ground truth camera trajectory is given as an output of the SLAM system. We can therefore compute 6-DoF localization errors for all camera poses. The camera images have a resolution of 1241×376 and are captured at 10 Hz.

Fig. 3 shows a localization result of our method compared to the estimates of the visual odometry. It is clearly visible that the accumulated drift is corrected. The camera was tracked successfully along the whole trajectory in all 10 localization runs. Fig. 4 shows how the averaged error is distributed along the trajectory. Peaks in the error are often caused by turns of the car which introduce substantial drift that is subsequently corrected through our alignment. Averaging over all runs and the entire trajectory yielded a translational error of 0.30 ± 0.11 m and a rotational error of $1.65 \pm 0.91\,°$. Since our method runs at 10 fps, we achieved online tracking in this experiment.

B. Evaluation under Varying Conditions

To evaluate our approach under varying conditions, we captured a dataset with one of our robots equipped with a Velodyne HDL-32E LiDAR. We built a 3D LiDAR map of the Freiburg campus at a resolution of 10 cm (see Fig. 2 right). In addition, we collected images with two different hand-held cameras at two days with varying weather conditions (see Fig. 5 top). Both, a Canon S100 compact camera and an Apple iPhone 5S front camera, were calibrated using standard OpenCV methods. We down-sampled the images to a resolution of 1024×576 and 900×506 and recorded them at 25 Hz and 30 Hz respectively. Since we tracked the camera pose for every frame, online processing was not feasible.

For both camera trajectories we performed 10 successful localization runs. The standard deviation was 0.06 m / 0.46 ° for the Canon camera trajectory and 0.12 m / 0.35 ° for the iPhone trajectory. Since we do not have ground truth trajectories for these camera poses, we cannot provide errors. Yet, given the camera pose with respect to the map, we can post-colorize the map points to visualize the quality of the estimates. Fig. 5 (bottom) and Fig. 6 show the post-colorized map of the Freiburg campus.

The varying conditions under which these images were captured support the hypothesis that our approach to match geometry provides robustness to changes in the photometric appearance of the environment, e.g., caused by different illumination due to time of day or weather. To highlight the viewpoint invariance of our method, the localization runs were performed in the opposite direction compared to the map acquisition. For visual localization approaches relying on non-omnidirectional cameras, this can be extremely difficult as the photometric appearance of the environment can tremendously depend on the viewing direction.

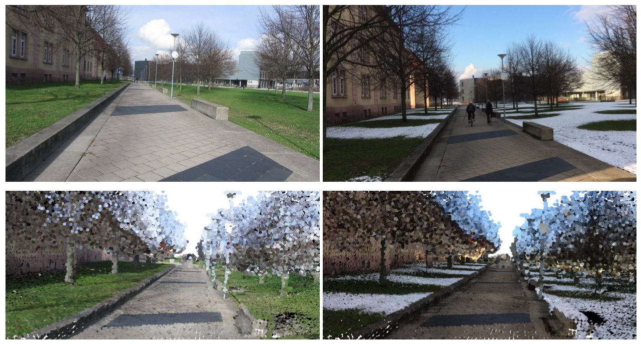

Fig. 5. Top: images from the Canon camera (left) and iPhone (right) datasets captured on the Freiburg campus. Notice the different weather conditions. Bottom: Post-colorized 3D LiDAR map from a similar perspective.

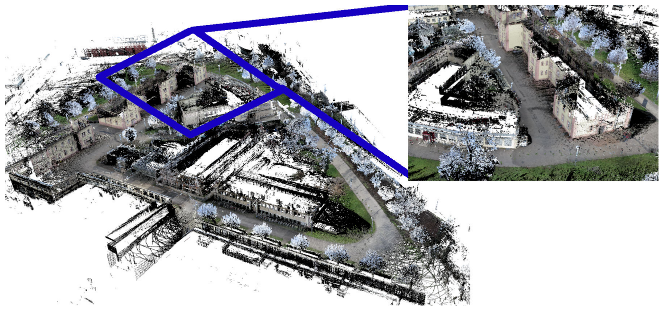

Fig. 6. 3D LiDAR map of the Freiburg campus post-colorized based on the pose estimates of the Canon camera trajectory. Very fine structures like thin tree branches tend to be colorized incorrectly (with sky texture). This is related to the localization accuracy but also caused by map inaccuracies.

V. Conclusions

In this paper, we presented a novel approach to localize a monocular camera with respect to a given 3D LiDAR map. Our method employs visual odometry to track the 6-DoF camera pose and to reconstruct a sparse set of 3D points via bundle adjustment. We align the reconstructed points with the map by continuously applying an estimated similarity transformation to indirectly localize the camera. Using a LiDAR for mapping and cameras for localization combines the advantages of both sensors. Experiments carried out on a publicly available and large-scale dataset demonstrate that the accuracy and frame rate of our method are comparable to state-of-the-art approaches even though it does not rely on any additional information or GPU hardware support. Further experiments carried out under varying conditions indicate that approaching visual localization by matching geometry yields the benefit of being robust to changes in the photometric appearance of the environment. They suggest that localizing a camera in panoramic 3D LiDAR maps additionally provides view-point invariance.

Acknowledgment

The authors would like to thank Mur-Artal *et al.* for providing the ORB-SLAM source code and Geiger *et al.* for providing the KITTI dataset.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*, 2009.

[2] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Proc. of the European Conf. on Computer Vision (ECCV)*, 2006.

[3] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[4] W. Churchill and P. Newman, "Experience-based navigation for long-term localisation," *International Journal of Robotics Research*, vol. 32, no. 14, pp. 1645–1661, 2013.

[5] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[6] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vision Computing*, vol. 21, no. 13, pp. 1145–1153, 2003.

[7] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015.

[8] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[9] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *International Journal of Robotics Research*, p. 0278364911430419, 2011.

[10] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Proc. of the IEEE/ACM Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[11] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[13] R. Mur-Artal, J. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[14] T. Naseer, M. Ruhnke, C. Stachniss, L. Spinello, and W. Burgard, "Robust visual slam across seasons," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[15] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6d slam3d mapping outdoor environments," *Journal of Field Robotics*, vol. 24, no. 8-9, pp. 699–722, 2007.

[16] G. Pascoe, W. Maddern, and P. Newman, "Direct visual localisation and calibration for road vehicles in changing city environments," in *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV) Workshops*, 2015.

[17] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[18] S. Se, D. G. Lowe, and J. J. Little, "Vision-based global localization and mapping for mobile robots," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 364–375, 2005.

[19] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp," in *Proc. of Robotics: Science and Systems (RSS)*, 2009.

[20] B. Steder, M. Ruhnke, S. Grzonka, and W. Burgard, "Place recognition in 3d scans using a combination of bag of words and point feature based relative pose estimation," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[21] H. Strasdat, J. Montiel, and A. J. Davison, "Real-time monocular slam: Why filter?" in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2010.

[22] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular slam," in *Proc. of Robotics: Science and Systems (RSS)*, 2010.

[23] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial intelligence*, vol. 128, no. 1, pp. 99–141, 2001.

[24] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment — a modern synthesis," in *Vision Algorithms: Theory and Practice*. Springer, 1999, pp. 298–372.

[25] R. W. Wolcott and R. M. Eustice, "Visual localization within lidar maps for automated urban driving," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2014.