Robotics: Science and Systems 2018
Pittsburgh, PA, USA, June 26-30, 2018

Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments

Jens Behley and Cyrill Stachniss
Institute of Geodesy and Geoinformation, University of Bonn, Germany
Email: {firstname.lastname}@igg.uni-bonn.de

***Abstract*—Accurate and reliable localization and mapping is a fundamental building block for most autonomous robots. For this purpose, we propose a novel, dense approach to laser-based mapping that operates on three-dimensional point clouds obtained from rotating laser sensors. We construct a surfel-based map and estimate the changes in the robot's pose by exploiting the projective data association between the current scan and a rendered model view from that surfel map. For detection and verification of a loop closure, we leverage the map representation to compose a virtual view of the map before a potential loop closure, which enables a more robust detection even with low overlap between the scan and the already mapped areas. Our approach is efficient and enables real-time capable registration. At the same time, it is able to detect loop closures and to perform map updates in an online fashion. Our experiments show that we are able to estimate globally consistent maps in large scale environments solely based on point cloud data.**

通过scan
和对应的地图
来求去poses，
但是具体不太明白

I. Introduction

Most autonomous robots, including self-driving cars, must be able to reliably localize themselves, ideally by using only their own sensors without relying on external information such as GPS or other additional infrastructure placed in the environment. There has been significant advances in vision-based [6, 7] and RGB-D-based [18, 33, 3] SLAM systems over the past few years. Most of these approaches use (semi-)dense reconstructions of the environment and exploit them for frame-to-model tracking, either by jointly optimizing the map and pose estimates or by alternating pose estimation and map building [21]. Dense approaches have a prospective advantage over feature-based and sparse approaches as they use all available information and thus do not depend on reliable feature extraction or availability of such features.

In contrast to these developments, current 3D laser-based mapping systems mainly accomplish the estimation relying on feature-based solutions [34, 35], reduced map representations [14, 13], voxel grid-based methods [16], or point sub-sampling [30], which all effectively reduce the data used for alignment. Compared to most indoor applications using RGB-D sensors, we have to tackle additional challenges in outdoor applications using 3D laser sensors, *i.e.*, (1) fast sensor movement resulting in large displacements between scans, (2) comparably sparse point clouds, and (3) large-scale environments.

In this paper, we present a dense mapping approach called Surfel-based Mapping (SuMa), which builds globally consistent maps by tracking the pose, so-called odometry, and closing loops using a pose graph as illustrated in Figure 1. To this end, we employ a surfel map to efficiently generate synthetic views for projective data association and additional loop closure detection with a verification step afterwards. Furthermore, the surfel representation enables us to efficiently represent large environments. We are able to update the map on loop closures by exploiting a pose graph optimization integrating odometry and loop closure constraints.

The key contributions of our work are as follows:

- We present a SLAM system for efficient mapping of three-dimensional laser range data resulting in globally consistent maps.
- We propose a novel map-based criterion for loop closure

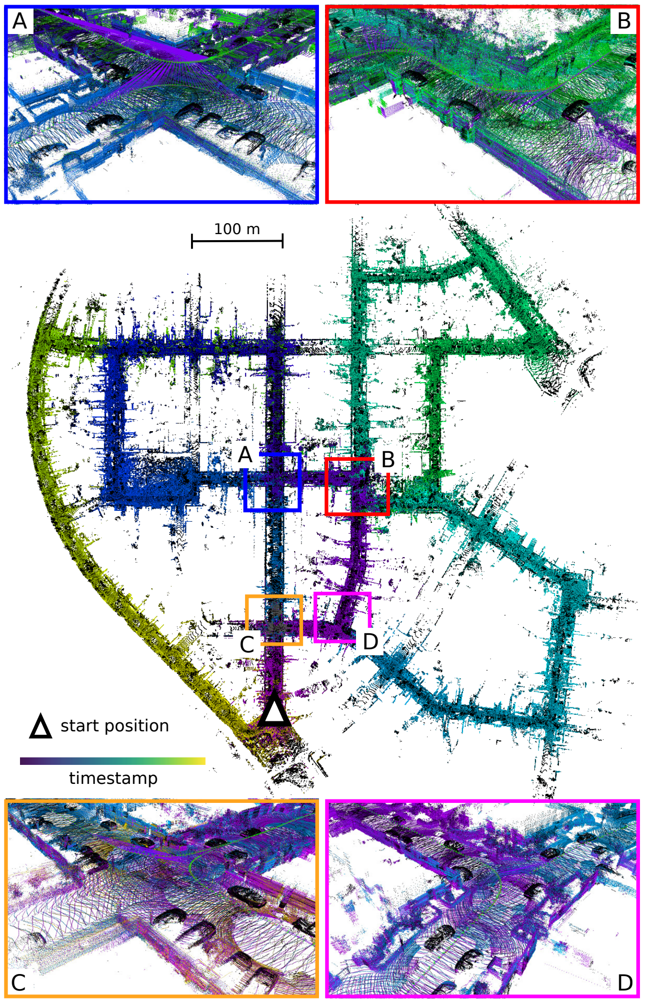

Fig. 1. Mapping result of our approach on sequence 00 of the KITTI Vision Benchmark. Shown is the complete trajectory color encoded with the timestamp of the scan ranging from purple to yellow. Also shown are some loop closures (A)-(D) found by our approach after pose graph optimization.

detection that is able to detect loop closures even in situations with small overlap between scans.

Our approach is inspired by recent work in RGB-D SLAM [12, 33], but is designed to operate in setups with fast moving vehicles in comparably large outdoor environments. Our experimental evaluation on the KITTI Odometry Benchmark shows that our approach is on par with other laser-based approaches [34, 35], while performing 3D point cloud registration, map update, and loop closure detection including loop verification at 20 Hz on average. To the best of our knowledge, this is the first approach for 3D laser-based mapping using surfels for mapping with loop closures that produces globally consistent maps and operates at that speed. 有代码 We furthermore release our implementation of the approach.[1]

II. Related Work

Odometry estimation and Simultaneous Localization and Mapping (SLAM), especially with (stereo) cameras and 2D laser range scanners, is a classical topic in robotics and in the computer vision community. We acknowledge the large body of work in this field, but concentrate here on approaches based on 3D laser range data and closely related work using RGB-D sensors. For a more detailed recent overview, we refer to the article by Cadena *et al.* [1] and the references therein.

Laser-based Mapping. Laser-based odometry and mapping systems often reduce the 3D point cloud data by relying on features [34], subsampled clouds [16, 30], or voxel-based [34] as well as NDT-based map representations [27, 23, 4]. In contrast to that, we operate on all laser points and perform a registration to a surfel map at every step of the algorithm. While the approach by Moosmann and Stiller [16] also uses depth images to accelerate the computation of normals, it uses nearest neighbors in a grid-based map representation to estimate correspondences. Thus, it is inevitable to use subsampling to accelerate the processing and only $1,500$ points are retained. The Lidar Odometry and Mapping (LOAM) [34, 35] by Zhang and Singh extracts distinct features corresponding to surfaces and corners, which are then used to determine point-to-plane and point-to-line distances to a voxel grid-based map representation. To enable real-time operation, LOAM switches between frame-to-frame at 10 Hz and frame-to-model operation at 1 Hz update frequency. Like the approach by Stückler *et al.* [28], we also use surfels, but we exploit the representation to render synthetic views allowing fast data association and furthermore perform loop closure detection.

In contrast to the mentioned approaches, our method uses projective data association to estimate dense correspondences for each projected point and therefore is more robust to missing features or missing data. Unlike other approaches, it can perform frame-to-model tracking in every iteration and updates the map representation at the same time. Lastly, none of the mentioned approaches integrates loop closures in an online fashion to obtain globally consistent maps. Very recently, Park *et al.* [20] proposed a surfel-based mapping approach based on ElasticFusion by Whelan *et al.* [33] that allows optimization of continuous-time trajectories for a rotating 2D laser scanner. In contrast to this work, we use projective data association to find correspondences between the current scan and rendered model views, which is computationally more efficient than relying on nearest neighbor search.

RGB-D-based Mapping. In recent years, there has been considerable progress in the field of RGB-D-based SLAM. KinectFusion, the seminal system by Newcombe *et al.* [18], largely impacted the development in the subsequent years. In line with Newcombe *et al.*'s approach, the approach of Keller *et al.* [12] for RGB-D SLAM uses projective data association in a dense model, but relies on a surfel-based map [31] for tracking. The surfel-based representation allows to model comparably large environments without a compromise in terms of reduced reconstruction fidelity due to reduced grid resolution or a more complex implementation [32, 19]. Other recent RGB-D approaches use the surfels to extract planar surfaces [24] or perform online loop closure integration by deforming the surfel representation [33, 31].

While our method borrows some ideas for frame-to-model registration from Keller *et al.* and Whelan *et al.*, we specifically designed our approach to work with rotating laser range scanners in highly dynamic environments, like busy inner-city traffic, and furthermore perform loop closure detection and verification using a map-based criterion, which enables a more reliable detection of loop closures using 3D laser range data.

Laser-based Loop Closure Detection. For loop closure detection using three-dimensional laser-range data, mainly feature-based approaches [25, 26, 22] were investigated. These approaches either use specific interest points [25, 26] to aggregate features or generally generate a global feature representation [15, 22] of a point cloud, which is then used to compute a distance between two point clouds. Often, a simple thresholding is used to identify potential loop closure candidates, which then must be verified. Only recently, the SegMatch approach by Dubé *et al.* [5] investigated an approach that matches segments extracted from a scan to find loop closures via segment-based features. A geometric test via RANSAC is used to verify a potential loop closure identified by the matching procedure.

In contrast to the aforementioned approaches, we directly determine the compatibility of the current laser scan and nearby poses using ICP. For this purpose, we propose to use a criterion based on a rendered view of the map to determine if a loop closure leads to a consistent map given the current scan. We verify this multiple times after the first detection before we actually integrate the scan into the map. Hess *et al.* [10] propose a similar strategy for 2D laser range data, as they also search in nearby submaps for loop closure candidates based on the current pose estimate. However, they use an exhaustive branch-and-bound search on the occupancy grid maps, which is not possible for 3D laser range data due to runtime constraints.

[1]See our project website for additional information, results, and videos: `https://jbehley.github.io/projects/surfel_mapping/`

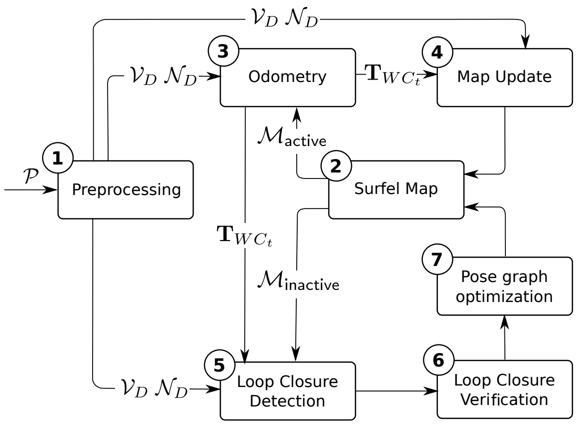
$\mathcal{V}_D\ \mathcal{N}_D$
$\mathcal{V}_D\ \mathcal{N}_D$
3 Odometry
$\mathbf{T}_{WC_t}$
4 Map Update
$\mathcal{M}_{\text{active}}$
$\mathcal{P}$
1 Preprocessing
2 Surfel Map
$\mathbf{T}_{WC_t}$
$\mathcal{M}_{\text{inactive}}$
7 Pose graph optimization
$\mathcal{V}_D\ \mathcal{N}_D$
5 Loop Closure Detection
6 Loop Closure Verification

Fig. 2. Our approach. Starting with a point cloud $\mathcal{P}$, (1) a vertex map $\mathcal{V}_D$ and normal map $\mathcal{N}_D$ are generated, which are used for frame-to-model alignment (3) to our model vertex map $\mathcal{V}_M$ and normal map $\mathcal{N}_M$ rendered from $\mathcal{M}_{\text{active}}$ (2). The transformation $\mathbf{T}_{WC_t}$ is then used to update the surfel map $\mathcal{M}_{\text{active}}$ (4). Using the inactive map $\mathcal{M}_{\text{inactive}}$, we try to find a loop closure candidate (5), which then needs to be verified (6) in subsequent scans. Verified loop closures are then integrated into a pose graph and used for optimization (7) in a separate thread. The optimized poses are then used to modify the surfel map yielding a globally consistent map.

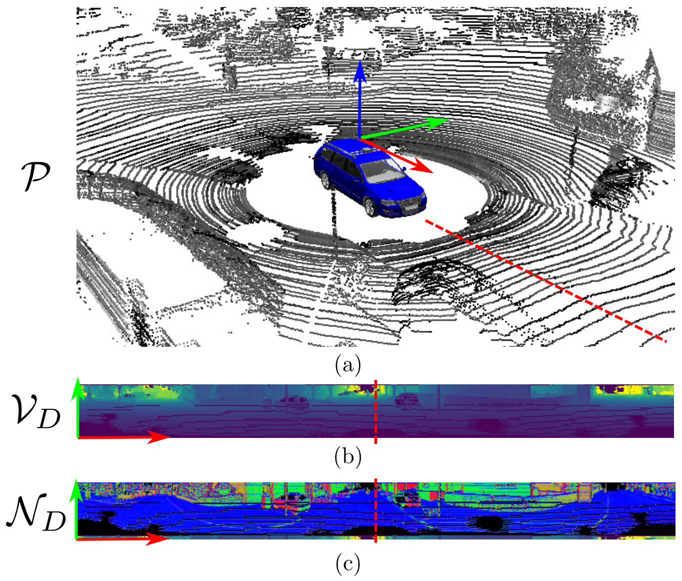

Fig. 3. Preprocessing of a point cloud generated by a rotating laser range scanner. (a) A point cloud from the KITTI Vision Benchmark, (b) the corresponding vertex map $\mathcal{V}_D$, and (c) the corresponding normal map $\mathcal{N}_D$. In the vertex map $\mathcal{V}_D$, colors ranging from purple to yellow, correspond to the distance of the point in the given pixel. Colors in the visualization of the normal map $\mathcal{N}_D$ correspond to the direction of the normal, *i.e.*, blue are normals pointing in the direction of the z-axis (upward). Also shown are the coordinate systems and the correspondence between the point cloud and the vertex/normal map indicated by a dashed line.

III. APPROACH

Notation. In the following, we denote the transformation of a point $\mathbf{p}_A$ in coordinate frame A to a point $\mathbf{p}_B$ in coordinate frame B by $\mathbf{T}_{BA} \in \mathbb{R}^{4\times4}$, such that $\mathbf{p}_B = \mathbf{T}_{BA}\mathbf{p}_A$. Let $\mathbf{R}_{BA} \in \mathrm{SO}(3)$ and $\mathbf{t}_{BA} \in \mathbb{R}^3$ denote the corresponding rotational and translational part of transformation $\mathbf{T}_{BA}$.

We distinguish between the observed *data* in the coordinate frame C_t at timestep t and a rendered representation, called *model*, which corresponds to our map representation at a given coordinate frame C_k, $k \in \{0, \ldots, t\}$. Each data coordinate frame C_t is associated to the world frame W by a pose $\mathbf{T}_{WC_t} \in \mathbb{R}^{4\times4}$, transforming the observed point cloud into the world coordinate frame. Our aim is to determine this transformation for each point cloud via pose changes $\mathbf{T}_{C_{k-1}C_k}$ given the rendered model at $\mathbf{T}_{WC_{k-1}}$, *i.e.*,

$$\mathbf{T}_{WC_t} = \mathbf{T}_{WC_0}\mathbf{T}_{C_0C_1}\cdots\mathbf{T}_{C_{t-1}C_t}, \tag{1}$$

assuming $\mathbf{T}_{WC_0}$ to be the identity $\mathbf{Id} \in \mathbb{R}^{4\times4}$ or the extrinsic calibration of the laser in respect to a fixed reference frame.

Overview. Before we explain our approach in detail, we provide a brief overview of the processing steps, shown in Figure 2. For each point cloud $\mathcal{P} = \{\mathbf{p} \in \mathbb{R}^3\}$, we estimate the pose $\mathbf{T}_{WC_t}$ at timestep t by performing the following steps, also indicated by corresponding numbers in Figure 2:

1) First, we use a projection function $\Pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$ to generate a vertex map $\mathcal{V}_D : \mathbb{R}^2 \mapsto \mathbb{R}^3$ mapping a two-dimensional image coordinate $(u, v)^T \in \mathbb{R}^2$ to a point $(x, y, z)^T \in \mathbb{R}^3$. We generate a corresponding normal map $\mathcal{N}_D : \mathbb{R}^2 \mapsto \mathbb{R}^3$ exploiting the vertex map $\mathcal{V}_D$.[2]
2) Next, we render a vertex map $\mathcal{V}_M$ and a normal map $\mathcal{N}_M$ at the last pose estimate $\mathbf{T}_{WC_{t-1}}$ from the current active surfel map $\mathcal{M}_{\text{active}}$, which contains the last Δ_{active} point clouds up to timestep $t-1$.
3) Using both the vertex map and normal map, we estimate the odometry, *i.e.*, the transformation $\mathbf{T}_{C_{t-1}C_t}$. We use ICP to align the points in $\mathcal{V}_D$ with the points of $\mathcal{V}_M$, where we exploit projective data association. This so-called frame-to-model ICP yields $\mathbf{T}_{C_{t-1}C_t}$, which is used to update the global pose $\mathbf{T}_{WC_t}$ via Equation 1.
4) With the current pose $\mathbf{T}_{WC_t}$, we update the surfel map $\mathcal{M}_{\text{active}}$ by initializing surfels for previously unseen areas, but also refining surfels of already covered areas.
5) Next, we search for a potential loop closure in the so-called inactive map $\mathcal{M}_{\text{inactive}}$ and try to align the current measurements with the map.
6) If we have found a loop closure candidate at timestep t, we try to verify it in the subsequent timesteps $t+1, \ldots, t+\Delta_{\text{verification}}$, which ensures that we only add consistent loop closures to the pose graph.
7) In a separate thread, a pose graph is optimized consisting of the relative poses of the odometry and the loop closures. The optimized poses are then used for updating the surfel map.

We will now describe these steps in more detail.

A. Preprocessing

For projective data association, we first project the point cloud $\mathcal{P}$ to the vertex map $\mathcal{V}_D : \mathbb{R}^2 \mapsto \mathbb{R}^3$, where each

[2]Note that we convert points $\mathbf{p} \in \mathbb{R}^3$ and normals $\mathbf{n} \in \mathbb{R}^3$ to corresponding homogeneous coordinates before application of the affine transformation, *i.e.*, $\dot{\mathbf{p}} = (x, y, z, 1)$ and $\dot{\mathbf{n}} = (x, y, z, 0)$, but will not include this operation explicitly in the following derivations.

pixel contains the nearest 3D point. Each point $\mathbf{p}_i = (x, y, z)$ is converted via the function $\Pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$ to spherical coordinates and finally to image coordinates (u, v), *i.e.*,

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\left[1 - \arctan(y, x) \cdot \pi^{-1}\right] \cdot w \\ \left[1 - \left(\arcsin(z \cdot r^{-1}) + \mathrm{f_{up}}\right) \mathrm{f}^{-1}\right] \cdot h \end{pmatrix}, \quad (2)$$

where $r = ||\mathbf{p}||_2$ is the range, $\mathrm{f} = \mathrm{f_{up}} + \mathrm{f_{down}}$ is the vertical field-of-view of the sensor, and w, h are the width and height of the resulting vertex map $\mathcal{V}_D$. This projection function is then also used to find the correspondence between the current scan and the rendered model view.

This projective data association is the key to enable highly efficient, dense mapping with laser range data, since we avoid to explicitly search for nearest neighbors.

Figure 3 shows an example point cloud with corresponding vertex and normal map, where we also included the corresponding coordinate systems.

Given the vertex map $\mathcal{V}_D$, we compute for each coordinate (u, v) a corresponding normal in $\mathcal{N}_D$ using cross products over forward differences, *i.e.*,

$$\begin{aligned}\mathcal{N}_D((u, v)) = &\left(\mathcal{V}_D((u+1, v)) - \mathcal{V}_D((u, v))\right) \\ &\times \left(\mathcal{V}_D((u, v+1)) - \mathcal{V}_D((u, v))\right). \quad (3)\end{aligned}$$

We only store normals if the vertex map contains valid points for all coordinates and wrap image coordinates in the x-direction, *i.e.*, $\mathcal{V}_D((u, v)) = \mathcal{V}_D((0, v))$ if $u \geq w$ and $\mathcal{V}_D((u, v)) = \mathcal{V}_D((w-1, v))$ if $u < 0$. While this normal estimation can be affected by noisy measurements, we found that the accuracy of the normal estimates did not influence the performance of the frame-to-model ICP significantly. We furthermore did not see any benefit from a bilateral filtering of the vertex map before the computation of normal estimates that is usually applied with RGB-D data [18].

B. Map Representation

We employ a surfel-based map [31, 12, 24, 33], since it allows us to represent even large-scale environments and maintain dense, detailed geometric information of the point clouds at the same time (see Figure 4 for an example). Surfels are furthermore relatively fast to render and therefore well suited for our application. A surfel map $\mathcal{M}$ is an unordered set of surfels s, where each surfel is defined by a position $\mathbf{v}_s \in \mathbb{R}^3$, a normal $\mathbf{n}_s \in \mathbb{R}^3$, a radius $r_s \in \mathbb{R}$. Each surfel additionally carries two timestamps: the creation timestamp t_c and the timestamp t_u of its last update by a measurement.

Furthermore, a stability log odds ratio l_s is maintained using a binary Bayes Filter [29] to determine if a surfel is considered stable or unstable. For rendering of the model view using the map, only stable surfels are considered and rendered. We update l_s as follows:

$$\begin{aligned}l_s^{(t)} = l_s^{(t-1)} + \mathrm{odds}\left(p_{\text{stable}} \cdot \exp\left(-\frac{\alpha^2}{\sigma_\alpha^2}\right) \exp\left(-\frac{d^2}{\sigma_d^2}\right)\right) \\ - \mathrm{odds}(p_{\text{prior}}), \quad (4)\end{aligned}$$

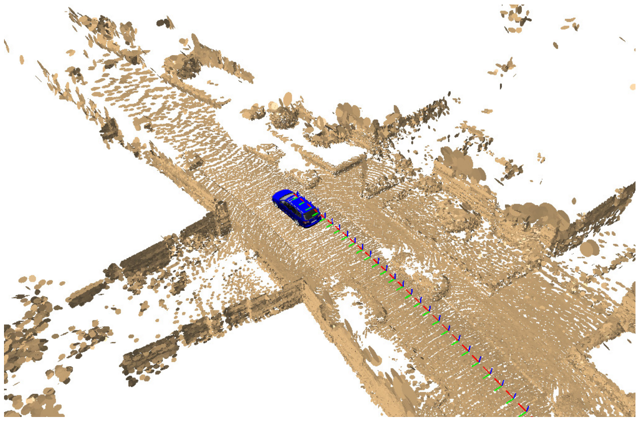

Fig. 4. Surfel map aggregated over multiple scans.

where $\mathrm{odds}(p) = \log(p \cdot (1-p)^{-1})$ and p_{stable} and p_{prior} are probabilities for a stable surfel given a compatible measurement and the prior probability, respectively. The terms $\exp(-x^2\sigma^{-2})$ are used to account for noisy measurements, where α is the angle between the surfel's normal $\mathbf{n}_s$ and the normal of the measurement to be integrated and d is the distance of the measurement in respect to the associated surfel. The measurement normal is taken from $\mathcal{N}_D$ and the correspondences from the frame-to-model ICP, see Section III-C.

In contrast to approaches using a deformation graph [31, 33], we let the poses directly manipulate the map by linking surfels to a pose via the creation timestamp t_c. Thus, the surfel's position $\mathbf{v}_s$ and normal $\mathbf{n}_s$ are local coordinates in the coordinate frame C_{t_c}, which allows us to modify the surfel map by a simple modification of the corresponding pose $\mathbf{T}_{WC_{t_c}}$ at time t_c. Thus, we are able to modify the map after a loop closure without the need to rebuild the map by reintegrating the past laser scans.

In line with Whelan *et al.* [33], we distinguish the active parts $\mathcal{M}_{\text{active}}$ and inactive map parts $\mathcal{M}_{\text{inactive}}$ at timestep t, comprised of all recently updated surfels, *i.e.*, $t_u \geq t - \Delta_{\text{active}}$, and not recently created surfels, *i.e.*, $t_c < t - \Delta_{\text{active}}$, respectively. The odometry is only estimated using $\mathcal{M}_{\text{active}}$ and the loop closures are only searched in $\mathcal{M}_{\text{inactive}}$.

C. Odometry Estimation

The observerd point cloud is aligned to a rendered representation $\mathcal{V}_M$ and $\mathcal{N}_M$ of the model in coordinate frame C_{t-1}, *i.e.*, we apply $\mathbf{T}_{WC_{t-1}}^{-1}$ before rendering all surfels, resulting in a local view of the already mapped world at timestmap $t-1$. We incrementally minimize the point-to-plane error given by

$$E(\mathcal{V}_D, \mathcal{V}_M, \mathcal{N}_M) = \sum_{\mathbf{u} \in \mathcal{V}_D} \underbrace{\mathbf{n}_u^\top \cdot \left(\mathbf{T}_{C_{t-1}C_t}^{(k)} \mathbf{u} - \mathbf{v}_u\right)^2}_{r_\mathbf{u}}, \quad (5)$$

where each vertex $\mathbf{u} \in \mathcal{V}_D$ is projectively associated to a reference vertex $\mathbf{v}_u \in \mathcal{V}_M$ and its normal $\mathbf{n}_u \in \mathcal{N}_M$ via

$$\mathbf{v}_u = \mathcal{V}_M\left(\Pi\left(\mathbf{T}_{C_{t-1}C_t}^{(k)} \mathbf{u}\right)\right), \quad (6)$$

$$\mathbf{n}_u = \mathcal{N}_M\left(\Pi\left(\mathbf{T}_{C_{t-1}C_t}^{(k)} \mathbf{u}\right)\right). \quad (7)$$

Here, $\mathbf{T}^{(k)}_{C_{t-1}C_t}$ corresponds to the current pose estimate of the so-called *frame-to-model ICP* at iteration k. If $\Pi(\mathbf{u})$ maps to outside of the vertex map or the corresponding vertex or normal are undefined, we ignore the error term. For outlier rejection, we filter out correspondences exceeding a distance of δ_{ICP} or having an angle difference larger than θ_{ICP} between $\mathbf{n}_u$ and the corresponding normal of $\mathbf{u}$ in $\mathcal{N}_D$. We initialize the ICP pose $\mathbf{T}^{(0)}_{C_{t-1}C_t} = \mathbf{T}_{C_{t-2}C_{t-1}}$ with the last pose increment to warm start the optimization.

For solving the non-linear least squares problem, we represent the linearized pose increments by Lie-algebra elements $\delta \in \mathfrak{se}(3)$, which we simply write as vectors $\delta \in \mathbb{R}^6$ and use the Rodrigues' formula to compute the exponential map $\exp(\cdot) : \mathfrak{se}(3) \mapsto SE(3)$ to update the current pose estimate $\mathbf{T}^{(k)}_{C_{t-1}C_t}$, *i.e.*, $\mathbf{T}^{(k)}_{C_{t-1}C_t} = \exp(\delta) \cdot \mathbf{T}^{(k-1)}_{C_{t-1}C_t}$.

We minimize the objective of Equation 5 using Gauss-Newton and determine increments δ by iteratively solving

$$\delta = \left(\mathbf{J}^\top \mathbf{W} \mathbf{J}\right)^{-1} \mathbf{J}^\top \mathbf{W} \mathbf{r}, \tag{8}$$

where $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a diagonal matrix containing weights w_u for each residual r_u, $\mathbf{r} \in \mathbb{R}^n$ is the stacked residual vector, and $\mathbf{J} \in \mathbb{R}^{n\times 6}$ the Jacobian of $\mathbf{r}$ with respect to the increment δ. For a single residual $r_{\mathbf{u}}$ of vertex $\mathbf{u}$, it follows (see *e.g.* [18, 9]) that a row of the Jacobian $\mathbf{J}$ is given by

$$\mathbf{J}_u = \left[\begin{array}{cc} \mathbf{n}_u^\top & (\mathbf{n}_\mathbf{u} \times \mathbf{v}_\mathbf{u})^\top \end{array}\right]. \tag{9}$$

We use a Huber weighting [11] to weaken the influence of outliers that could not be removed by our outlier rejection.

D. Map Update

Given the current pose $\mathbf{T}_{WC_t}$ of the frame-to-model ICP, we integrate the points inside $\mathcal{V}_D$ into the surfel map. For this purpose, we have to decide which already existing surfels must be updated and which measurements must be used to initialize a new surfel in the surfel map.

For this decision, we start by computing the radius r_s of potential new surfel s for each $\mathbf{v}_s \in \mathcal{V}_D$ and corresponding normal $\mathbf{n}_s \in \mathcal{N}_D$ with the aim to cover roughly the corresponding pixel of the vertex map:

$$r_s = \frac{\sqrt{2}\,\|\mathbf{v}_s\|_2 \cdot p}{\text{clamp}(-\mathbf{v}_s^\top \mathbf{n}_s \cdot \|\mathbf{v}_s\|_2^{-1}, 0.5, 1.0)}, \tag{10}$$

where $p = \max(w \cdot f_{\text{horiz}}^{-1}, h \cdot f_{\text{vert}}^{-1})$ corresponds to the pixel size and $\text{clamp}(x, u, l) = \min(u, \max(x, l))$ to the clamping operation. In contrast to prior approaches [31, 12, 33], we found this restriction needed in outdoor environments to avoid overly large surfels at long ranges. The radius is proportional to the accuracy of the measurement, since a small radius corresponds to a close distance measurement on surfaces with a normal pointing in the direction of the laser beam.

Then, we render $\mathcal{V}_M$, $\mathcal{N}_M$ and an index map $\mathcal{I}_M$ containing the indices of the nearest surfels in respect to the sensor origin with the final estimated pose $\mathbf{T}_{WC_t}$ to determine visible model surfels and their indices for updating. Each measurement point $\mathbf{v}_s$ is projected to $\mathcal{I}_M$ using Equation 2 to find the corresponding model surfel s'.

For updating the surfel map, we first determine if the data surfel s is compatible with the associated surfel s', *i.e.*, it holds $|\mathbf{n}_{s'}^\top(\mathbf{v}_s - \mathbf{v}_{s'})| < \delta_M$ and $||\mathbf{n}_s \times \mathbf{n}_{s'}|| < \sin(\theta_M)$. Depending on the compatibility, we now distinguish the following cases for a data surfel s and its associated model surfel s':

1) If the data surfel is compatible, we increase the stability of the associated model surfel. If the measurement is more precise, *i.e.*, $r_s < r_{s'}$, we also update the model surfel using an exponential moving average:

$$\mathbf{v}_{s'}^{(t)} = (1-\gamma) \cdot \mathbf{v}_s + \gamma \cdot \mathbf{v}_{s'}^{(t-1)}, \tag{11}$$

$$\mathbf{n}_{s'}^{(t)} = (1-\gamma) \cdot \mathbf{n}_s + \gamma \cdot \mathbf{n}_{s'}^{(t-1)} \tag{12}$$

$$r_{s'}^{(t)} = r_s \tag{13}$$

Thus, we only refine surfels to avoid losing information we gathered from closer ranges [12].

2) Otherwise, we decrease the stability of the associated model surfel s' and initialize a new surfel. If a measurement cannot be assigned to any existing surfel, we initialize a new surfel.

After updating the map, we remove all unstable surfels which have a too low stability value, $l_s < \tau_D$, and are at the same time already too old, $t - t_c > \tau_O$. By this means, we are able to remove unstable surfels mainly caused by dynamic objects, but also irrelevant surfels that are caused by clutter.

E. Loop Closures

For loop closure detection, we exploit the surfel map to render views from $\mathcal{M}_{\text{inactive}}$. Analogous to the odometry, we now use this rendered view to register the current point cloud. A loop closure is only considered if a composed virtual map view built from this alignment fits to the current measurements. We try to verify a potential loop closure by tracking the pose in the active and the inactive map at the same time. Only after successful verification, we include a loop closure constraint in the pose graph.

Detection. For loop closure detection, we first try to find a possible loop closure candidate in the inactive map $\mathcal{M}_{\text{inactive}}$. From all poses, $\mathbf{T}_{WC_0}, \ldots, \mathbf{T}_{WC_{t-\Delta_{\text{active}}}}$, we consider only one candidate in a radius δ_{loop} of the current pose estimate, *i.e.*, $j^* = \arg\min_{j\in 0,\ldots,t-\Delta_{\text{active}}} ||\mathbf{t}_{WC_t} - \mathbf{t}_{WC_j}||$. Note that the orientation of the candidate does not matter and therefore even candidates with small overlap between scans and different viewpoints are considered.

For candidate j^*, we try to align the current point cloud to the rendered view at the corresponding pose $\mathbf{T}_{WC_{j^*}}$ using the frame-to-model ICP. Since ICP heavily depends on the initialization, we check multiple initializations $\mathbf{T}^{(0)}_{C_{j^*}C_t}$ with respect to the old pose to compensate for rotational and translational drift. We compose the initial pose from the rotational difference $\mathbf{R}_{C_{j^*}C_t}$ between the current pose estimate and the candidate's rotation and the translational difference $\mathbf{t}_{C_{j^*}C_t}$, *i.e.*,

$$\mathbf{R}_{C_{j^*}C_t} = \mathbf{R}_{WC_{j^*}}^{-1} \mathbf{R}_{WC_t} \tag{14}$$

$$\mathbf{t}_{C_{j^*}C_t} = \mathbf{R}_{WC_{j^*}}^{-1} (\mathbf{t}_{WC_t} - \mathbf{t}_{WC_{j^*}}) \tag{15}$$

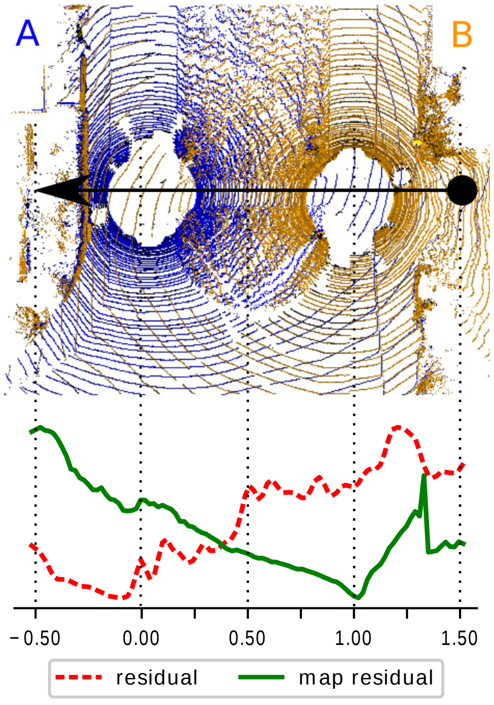

Fig. 5. Shown are two scans A (blue) and B (orange) from different sides of a street from KITTI sequence 06 from a birds eye view. From both sides, the other side's street is not visible and therefore hard to distinguish the correct from the incorrect loop closure just by means of residual of an alignment. Shown below is the residual along the arrow, *i.e.*, at every position on the arrow we compute the residual from A to B if B would exactly lie at that position. Shown is the residual (red) of B to A and the map residual (green) of scan B to the composed virtual view .

We then compose the initialization $\mathbf{T}^{(0)}_{C_{j^*}C_t}$ using $\mathbf{R}_{C_{j^*}C_t}$ as rotation and $\lambda\mathbf{t}_{C_{j^*}C_t}$, with $\lambda = \{0.0, 0.5, 1.0\}$, as translation.

Due to these multiple initializations, it is crucial to decide which of the ICP's results is the best and most consistent with the current measurements. As we also have to consider situations with low overlap between individual scans due to occlusions, we cannot simply evaluate the residual (see Equation 5), outlier count, or inlier count, of the current measurement with respect to the rendered inactive map.

Figure 5 shows such a situation for an example of the KITTI Vision Benchmark. Here, two point clouds (blue and orange) are shown which partially overlap and where we would like to find a loop closure. Below the correctly aligned scans, we show the resulting residuals for different points on a line between the ground truth poses, here $x = \{0.0, 0.5, 1.0\}$ correspond to the three initializations used for loop closure detection. The lowest residual (depicted as red dashed line) is reached between -0.2 and 0.0, while the residual at the correct location, $x = 1.0$, is by far larger than the wrong alignments. Thus, we would reject the correct alignment since its residual is simply larger than an incorrect alignment.

For solving such cases with small overlap between scans, we render a virtual view of the map after a potential loop closure using the transformations from the odometry estimation and aforementioned candidates. The virtual view is a composition

TABLE I
PARAMETERS OF OUR APPROACH

Parameter	value	description
$w \times h$	900×64	dimensions of vertex/normal map
θ_{ICP}	$30.0°$	outlier threshold (angle)
δ_{ICP}	$2\,m$	outlier threshold (distance)
δ_{M}	$0.2\,m$	map outlier threshold (distance)
θ_{M}	$30°$	map outlier threshold (angle)
k	0.5	Huber factor
$\text{f}_{\text{up}}, \text{f}_{\text{down}}$	$3.0°$, $25°$	vertical field-of-view
p_{stable}	0.6	stability
p_{prior}	0.5	prior stability
t_{old}	100	surfel age threshold
σ_α, σ_d	1.0	stability scaling
τ_D	$\text{odds}(1 - p_{\text{stable}})$	stability threshold
τ_O	3	max. unstable age
γ	0.9	exponential weight
G	9	number of submap grid cells
e	$10\,\text{m}$	size of a submap
Δ_{active}	100	time horizon for active map
$\Delta_{\text{verification}}$	5	number of verifications needed
δ_{loop}	50	max. distance of loop closure
$\epsilon_{\text{residual}}$	1.15	acceptance threshold

of the rendered model view of $\mathcal{M}_{\text{inactive}}$ and the current point cloud and this composition should only fit to the current scan if the composition agrees with the current point cloud data.

Given the transformation between the current point cloud and the potential loop closure, we can now transform both into a common coordinate frame and generate a composed vertex map $\mathcal{V}_C$ and normal map $\mathcal{N}_C$. First, the vertex and normal maps are filled by rendering the inactive map at the candidate position. From the current vertex $\mathcal{V}_D$ and normal map $\mathcal{V}_D$, we add entries to the composed maps if the point in $\mathcal{V}_D$ is closer than the existing point in $\mathcal{V}_C$.

We compute now $E(\mathcal{V}_D, \mathcal{V}_C, \mathcal{N}_C)$, which we call *map residual* E_{map}, in respect to the current scan, but now with the composed vertex map $\mathcal{V}_C$ and $\mathcal{N}_C$. Only if this composed view is consistent with our current measurements, we consider the candidate as a valid loop closure candidate. A possible alignment is consistent if the relative error between the residuals of that composed map E_{map} and the residual in respect to the active map E_{odom} is small, *i.e.*, $E_{\text{map}} < \epsilon_{\text{residual}} \cdot E_{\text{odom}}$, and if there are enough inliers and valid points in the rendered composed view. As can be seen from Figure 5, the map residual is minimal for the correct pose at $x = 1.0$ and therefore makes it possible to distinguish valid and invalid loop closure candidates.

Verification. Even though we try to rule out wrong loop closures by the rendering of a virtual map, this criterion can still lead to invalid loop closures. Thus, we aim to verify the loop closure by tracking the position in the active and inactive map. At each timestep $t+1, \ldots, t+\Delta_{\text{verification}}$ after a detected loop closure at timestep t, we estimate the odometry increment and apply this to the pose in the active and inactive part of the map. We render a virtual view of the map with these poses and check again for consistency with the current measurements, as described for the detection. Only if the consistency check succeeds for at least $\Delta_{\text{verification}}$ timesteps, we consider the loop closure valid and verified.

TABLE II
RESULTS ON KITTI ODOMETRY (TRAINING)

Approach	Sequence 00*	01	02*	03	04	05*	06*	07*	08*	09*	10	Average
Frame-to-Frame	0.9/2.1	1.2/4.0	0.8/2.3	0.7/1.4	1.1/11.9	0.8/1.5	0.6/1.0	1.2/1.8	1.0/2.5	0.8/1.9	1.0/1.8	0.9/2.9
Frame-to-Model	0.3/**0.7**	0.5/1.7	0.4/1.1	0.5/**0.7**	0.3/**0.4**	0.2/0.5	0.2/**0.4**	0.3/**0.4**	0.4/**1.0**	0.3/**0.5**	0.3/**0.7**	0.3/**0.7**
Frame-to-Model with loop closure	0.2/**0.7**	0.5/1.7	0.4/1.2	0.5/**0.7**	0.3/**0.4**	0.2/**0.4**	0.3/0.5	0.6/0.7	0.4/1.2	0.2/0.6	0.3/**0.7**	0.3/0.8
LOAM [35]	-/0.8	-/**1.4**	-/**0.9**	-/0.9	-/0.7	-/0.6	-/0.7	-/0.6	-/1.1	-/0.8	-/0.8	-/0.8
S-LSD [6]	0.3/0.6	0.3/2.4	0.2/0.8	0.3/1.0	0.3/0.4	0.2/0.7	0.2/0.7	0.3/0.6	0.3/1.1	0.3/1.1	0.3/0.7	0.3/0.9
SOFT-SLAM [2]	0.2/0.7	0.2/1.0	0.2/1.4	0.2/0.7	0.2/0.5	0.2/0.4	0.1/0.4	0.2/0.4	0.2/0.8	0.2/0.6	0.3/0.7	0.2/0.7

Relative errors averaged over trajectories of 100 to 800 m length: relative rotational error in degrees per 100 m / relative translational error in %,
Sequences marked with an asterisk contain loop closures. Bold numbers indicate top performance for laser-based approaches.

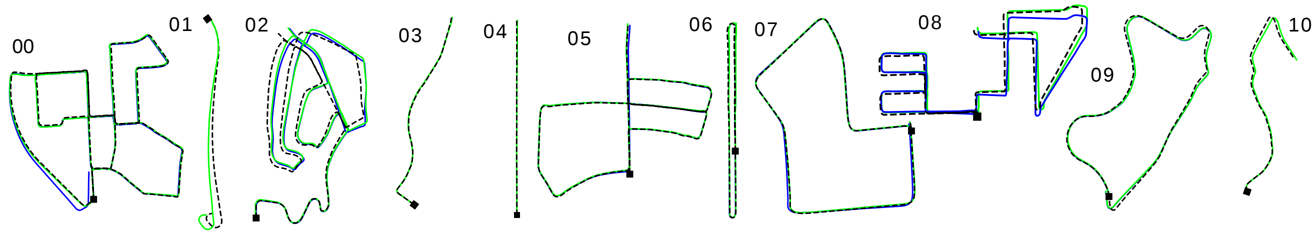

Fig. 6. Trajectories of the training set. The dashed black trajectory correspond to the GPS-based ground truth, blue to our approach without loop closure and green to our approach with loop closure. Sequence 01, 03, 10 have been rotated to save space. (Best viewed in color)

Pose graph optimization. As soon as a verified loop closure is found, we add loop closure constraints to a pose graph, which is then used for optimization. The poses are always initialized with the currently estimated poses.

In line with Hess *et al.* [10], we also perform the actual pose graph optimization in a separate thread. When the pose graph optimization is finished, we integrate the optimized poses in the map and update the current pose estimates accordingly.

F. Implementation Details

We implemented our approach using OpenGL 4.0 and we represented most of the data structures via textures or transform feedbacks and manipulate the values via vertex, geometry and fragment shaders. We furthermore leverage the rendering pipeline to generate vertex and normal maps by rendering small disks for each front-facing surfel.

To allow real-time capable processing even in ever growing maps, we employ a submapping approach to offload parts of the map from the GPU memory to the main memory. To this end, we implemented a two-dimensional rolling grid data structure of size G with extent e for each submap, which is used to determine which parts are obsolete and can be downloaded from the GPU. However, if parts of the environment are revisited, the corresponding surfels can also be uploaded again. The position of the rolling grid is updated according to the currently estimate sensor pose.

To avoid large peaks in the runtime, we implemented a delayed downloading procedure which only downloads a single grid cell in each update iteration. This effectively amortizes the time for downloading larger parts and spreads it over multiple map updates.

For pose graph optimization, we employ gtsam[3] and optimize the pose graph using Levenberg Marquardt with a maximum of 100 iterations. As the optimized poses are directly integrated into the map, we initialize the pose estimates with the last optimized poses and the current odometry estimates. Usually, the pose graph optimization terminates in less than 10 iterations.

IV. EXPERIMENTAL EVALUATION

We evaluate our approach on the odometry datasets of the KITTI Vision Benchmark [8], where we use the provided point clouds from the Velodyne HDL-64E S2 recorded at a rate of 10 Hz. The dataset contains data from different street environments ranging from inner city to highway traffic. Especially, the highway datasets are a major challenge, since the sensor moves at high speed, other fast moving dynamic objects are present, and the environment has only very few distinct structures which can be used. Here, most of the laser returns correspond to the flat street and only few correspond to traffic signs or some sparse trees or bushed along the highway.

The parameters of our approach, which we experimentally selected for the training data, are summarized in Table I. We use an Intel i7-6700@3.4 GHz with 16 GB RAM, and an Nvidia GeForce GTX 960 with 4 GB RAM.

Ablation study. Using the training data of the KITTI Vision Benchmark, we show the influence of the different design decisions of our approach.

Table II shows our approach without using the map (frame-to-frame), using the map (frame-to-model), and the map-based

[3]Version 4.0, available at https://bitbucket.org/gtborg/gtsam.

approach with loop closures for all sequences in detail.

Starting from a ‘frame-to-frame ICP’, which simply projectively associates points based on the last point cloud, we now discuss the influence of different design decisions.

Adding then the surfel map to aggregate past information helps to reduce drift, since it also adds information in areas which are currently sparsely covered. The stability estimation of a surfel ensures that only static objects remain in the surfel map. Therefore dynamic objects cannot cause false correspondences and consequently lead to inferior pose estimation performance. However, since we are not integrating loop closures, we cannot correct accumulated drift if we revisit already mapped areas.

Finally, adding loop closures and the pose graph optimization does lead to globally consistent maps. Adding the ability to close loops does only marginally improve the performance with respect to the KITTI metrics as the performance is average over short parts (up to 800 m) of the trajectory. While the numbers do not properly convey the improvement, the plots of the trajectories (see Figure 6) shows a more globally consistent trajectory. Please see also Figure 1 for a qualitative assessment of the global consistency of the trajectory for sequence 00.

Comparison with state-of-the-art. We included here for comparison the reported results of LOAM [35], a laser-based odometry approach, and for indirect comparison with state-of-the-art vision approaches, Stereo LSD-SLAM [6], a stereo vision-based complete SLAM system with pose graph optimization, and the currently best performing approach SOFT SLAM [2], also a stereo vision-based approach.

We perform generally on par with the state-of-the-art in laser-based odometry and often achieve better results in terms of translational error. However, the actual improvement of the ability to close loops is hard to evaluate using the KITTI Vision Benchmark, since it only provides ground truth trajectories generated with an GPS-based inertial navigation system. In the training data, we observed some inconsistencies in the height of the recorded GPS-based trajectory. For indoor datasets, it is possible to record highly precise motion capture data, which also enables to evaluate the absolute trajectory error in respect to the ‘perfect’ drift free motion capture trajectory.

Finally, we estimated the poses of the testset using the aforementioned parameters. Overall, we achieved an average rotational error of 0.0032 deg/m and an average translational error of 1.4% compared to 0.0017 deg/m and 0.7% translational error of LOAM on KITTI.

From visual inspection, we found that our method cannot correctly estimate the motion of the vehicle in sequences with very few structured objects, like highways. Due to the lack of structure to distinguish static and moving objects, the method sometimes fails to distinguish which objects move, especially if multiple objects are moving consistently, like cars in a traffic jam. In particular, dynamic objects can corrupt the map by spurious wrongly integrated surfels. Often two different cars can lead to consistent surfels in consecutive laser scans and therefore get wrongly integrated. In these ambiguous cases, integration of inertial measurements could help to get more consistent and accurate estimation results.

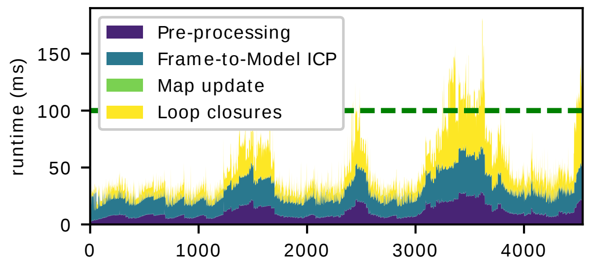

Fig. 7. Processing time needed to map sequence 00 from the KITTI Vision Benchmark. See Figure 1 for the finally registered complete point cloud.

Runtime. Figure 7 shows the processing time needed to process sequence 00 from the KITTI Vision Benchmark (see also Figure 1 for a visual impression of the complexity). We selected this sequence where parts of the environment are revisited and therefore often parts have to be down-, but also uploaded to the GPU, but also loop closure must be detected and subsequently verified. The odometry with the update of the map can be done with 31 ms on average and needs at most 71 ms. Together with the loop closure detection and verification, we need at most 189 ms. After few detection steps, where we have to check multiple candidates (peaks in the runtime plot), we then only have to verify a detected loop closure. Overall, our approach runs at 48 ms on average and therefore is able to process a scan at 20 Hz on average.

V. CONCLUSION

We presented a dense approach for laser-based odometry and mapping that is capable of performing all computations online and produces globally consistent maps. Our approach leverages a surfel-based map representation and performs projective data association of the current scan to the surfel maps at each iteration. As our experimental evaluation illustrates, this yields accurate registration results and globally consistent maps that are on par with the state-of-the-art.

A natural next step would be the integration of color information to improve the selection of correspondences while the projective data association [33, 20]. We furthermore plan to integrate a global loop closure search, which would also enable to find loops closures if the odometry estimate drifts to much. Also, a detection and tracking of dynamic objects such as the one by Moosmann and Stiller [17] has the potential to increase the robustness and quality of the registration results, since it would enable to reliably remove dynamic objects.

VI. ACKNOWLEDGEMENTS

This work has been supported by the DFG (BE 5996/1-1). We thank Johannes Schneider, Lorenzo Nardi, Mathias Hans, Igor Bogoslavskyi, and Philipp Lottes for fruitful discussions. We thank Thomas Whelan for making his ElasticFusion code available, which inspired our implementation.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J.J. Leonard. Past, Present, and Future of Simultaneous Localization And Mapping: Towards the Robust-Perception Age. *IEEE Trans. on Robotics (TRO)*, 32:1309–1332, 2016.

[2] I. Cvisic, J. Cesic, I. Markovic, and I. Petrovic. SOFT-SLAM: Computationally Efficient Stereo Visual SLAM for Autonomous UAVs. *Journal of Field Robotics (JFR)*, 2017.

[3] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. BundleFusion: Real-time Globally Consistent 3D Reconstruction using Online Surface Reintegration. *ACM Trans. on Graphics (TOG)*, 2016.

[4] A. Das, J. Servos, and S.L. Waslander. 3D Scan Registration Using the Normal Distributions Transform with Ground Segmentation and Point Cloud Clustering. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2013.

[5] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C.C. Lerma. SegMatch: Segment Based Place Recognition in 3D Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2017.

[6] J. Engel, J. Stückler, and D. Cremers. Large-scale direct slam with stereo cameras. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[7] J. Engel, V. Koltun, and D. Cremers. Direct Sparse Odometry. *arXiv:1607.02565*, 2016.

[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)*, pages 3354–3361, 2012.

[9] A. Handa. Simplified Jacobians in 6-DoF Camera Tracking. Technical report, University of Cambridge, 2014.

[10] W. Hess, D. Kohler, H. Rapp, and D. Andor. Real-Time Loop Closure in 2D LIDAR SLAM. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2016.

[11] P. J. Huber. *Robust Statistics*. Wiley, 1981.

[12] M. Keller, D. Lefloch, M. Lambers, and S. Izadi. Real-time 3D Reconstruction in Dynamic Scenes using Point-based Fusion. In *Proc. of the Intl. Conf. on 3D Vision (3DV)*, pages 1–8, 2013.

[13] J. Levinson and S. Thrun. Robust Vehicle Localization in Urban Environments Using Probabilistic Maps. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 4372–4378, 2010.

[14] J. Levinson, M. Montemerlo, and S. Thrun. Map-Based Precision Vehicle Localization in Urban Environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2007.

[15] M. Magnusson, H. Andreasson, A. Nuechter, Achim, and J. Lilienthal. Appearance-Based Loop Detection from 3D Laser Data Using the Normal Distributions Transform. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2009.

[16] F. Moosmann and C. Stiller. Velodyne SLAM. In *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*, pages 393–398, 2011.

[17] F. Moosmann and C. Stiller. Joint Self-Localization and Tracking of Generic Objects in 3D Range Data. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 1138–1144, 2013.

[18] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *Proc. of the Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011.

[19] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3D Reconstruction at Scale using Voxel Hashing. *Proc. of the SIGGRAPH Asia*, 32(6), 2013.

[20] C. Park, P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan. Elastic LiDAR Fusion: Dense Map-Centric Continuous-Time SLAM. *arXiv preprint*, abs/1711.01691, 2017.

[21] L. Platinsky, A.J. Davison, and S. Leutenegger. Monocular Visual Odometry: Sparse Joint Optimisation or Dense Alternation? In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2017.

[22] T. Röhling, J. Mack, and D. Schulz. A Fast Histogram-Based Similarity Measure for Detecting Loop Closures in 3-D LIDAR Data. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 736–741, 2015.

[23] J.P. Saarinen, T. Stoyanov, H. Andreasson, and A.J. Lilienthal. Fast 3D Mapping in Highly Dynamic Environments Using Normal Distributions Transform Occupancy Maps. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[24] R. F. Salas-Moreno, B. Glocker, P. H. J. Kelly, and A. J. Davison. Dense Planar SLAM. In *Proc. of the Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, pages 157–164, 2014.

[25] B. Steder, G. Grisetti, and W. Burgard. Robust Place Recognition for 3D Range Data Based on Point Features. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2010.

[26] 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 Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[27] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal. Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. *Intl. Journal of Robotics Research (IJRR)*, 31(12):1377–1393, 2012.

[28] J. Stückler and S. Behnke. Multi-Resolution Surfel Maps for Efficient Dense 3D Modeling and Tracking. *Journal of Visual Communication and Image Representation (JVCIR)*, 25(1):137–147, 2014.

[29] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[30] M. Velas, M. Spanel, and A. Herout. Collar Line Segments for Fast Odometry Estimation from Velodyne Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2016.

[31] T. Weise, T. Wismer, B. Leibe, and L. Van Gool. Online loop closure for real-time interactive 3D scanning. *Computer Vision and Image Understanding*, 115:635–648, 2011.

[32] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald. Real-time large scale dense RGB-D SLAM with volumetric fusion. *Intl. Journal of Robotics Research (IJRR)*, 34(4-5):598–626, 2014.

[33] T. Whelan, S. Leutenegger, R. S. Moreno, B. Glocker, and A. Davison. ElasticFusion: Dense SLAM Without A Pose Graph. In *Proc. of Robotics: Science and Systems (RSS)*, 2015.

[34] J. Zhang and S. Singh. LOAM: Lidar Odometry and Mapping in Real-time. In *Proc. of Robotics: Science and Systems (RSS)*, 2014.

[35] J. Zhang and S. Singh. Low-drift and real-time lidar odometry and mapping. *Autonomous Robots*, 41:401–416, 2017.