2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Freetures: Localization in Signed Distance Function Maps

Alexander Millane[‡], Helen Oleynikova[†], Christian Lanegger[‡], Jeff Delmerico[†]
Juan Nieto[†], Roland Siegwart[‡], Marc Pollefeys[†], César Cadena[‡]
[‡]Autonomous Systems Lab, ETH Zürich, [†]Microsoft Mixed Reality and AI Zurich Lab

***Abstract*— Localization of a robotic system within a previously mapped environment is important for reducing estimation drift and for reusing previously built maps. Existing techniques for geometry-based localization have focused on the description of local surface geometry, usually using pointclouds as the underlying representation. We propose a system for geometry-based localization that extracts features directly from an implicit surface representation: the Signed Distance Function (SDF). The SDF varies continuously through space, which allows the proposed system to extract and utilize features describing both surfaces and free-space. Through evaluations on public datasets, we demonstrate the flexibility of this approach, and show an increase in localization performance over state-of-the-art handcrafted surfaces-only descriptors. We achieve an average improvement of ~12% on an RGB-D dataset and ~18% on a LiDAR-based dataset. Finally, we demonstrate our system for localizing a LiDAR-equipped Micro Aerial Vehicle (MAV) within a previously built map of a search and rescue training ground. Our system, called *freetures*, is available as open source[1].**

I. Introduction

Localization is a key capability for robotics systems. During exploratory motion, localization is critical to reducing accumulated estimation drift. During deployment in previously visited environments, localization allows for reuse of existing maps [1]. Techniques leveraging 3D data are particularly important for the robots automating our homes, factories, and cities, in part because these robots frequently make use of 3D sensing, for example LiDAR or depth-cameras, but also because of the complementary properties of 3D sensing to images, such as illumination invariance and geometric accuracy.

Existing approaches to 3D localization are typically based on extraction and description of features from pointcloud data, with several successful approaches proposed [2]–[7]. Dense representations based on SDFs have also become increasingly popular, primarily for their success in fusing noisy RGB-D data, but also for fusing LiDAR data [8], for generating consistent maps [9], and for robotic path-planning [10]. SDFs have proven themselves a useful alternative to pointcloud-based representations in many contexts, but are relatively unexplored for use in localization. In this work we investigate such an approach.

This research was funded by Microsoft, as well as the National Center of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation.

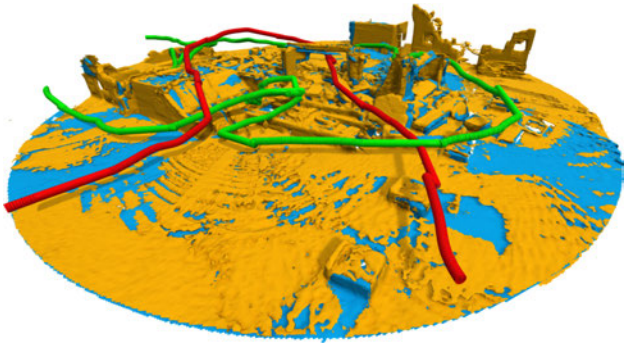

Fig. 1: Two maps of rubble from a collapsed building, produced by a LiDAR equipped MAV during two flights through a search and rescue training ground (estimated MAV trajectories are shown in red and green). The query map (yellow) is registered against the target map (blue) using the proposed localization system. See Sec. VI-D for details.

We introduce a method for geometry-based localization in dense 3D maps. We aim to locate an agent within a previously observed scene, which is represented as a collection of submaps. Surface geometry within each submap is stored as an implicit surface, the SDF. We extract local features *directly* on the SDF, an idea first introduced in 2D in our previous work [11] which is further developed and applied to 3D data, here. The approach allows for characterization of both surface and free-space geometry, since the SDF treats both of these spaces equally. Our hypothesis is that by extending the region of description beyond surface boundaries alone, localization performance can be improved. We evaluate this hypothesis by testing the efficacy of our system for localization on public and self-collected RGB-D and LiDAR datasets.

In summary, the contributions of this paper are:

- An investigation into the use of distance functions explicitly for the purpose of localization in 3D
- A keypoint detection and description approach for characterization of local SDF geometry
- An open-source implementation of the proposed system[1]

II. Related Work

In this section, we give a brief review of SDF-based mapping methods as well as an overview of a number of different localization methods. We will compare these to our proposed SDF-based localization method.

[1]https://github.com/alexmillane/freetures

A. Localization in SDF-based Mapping Systems

Implicit surfaces have become a popular means for representing geometry in mapping systems. Truncated Signed Distance Functions (TSDFs), popularized by KinectFusion [12], have proven particularly successful for fusing noisy, high-rate data from commodity depth-cameras. Several authors have since improved on these seminal works, enabling application of TSDFs to larger environments [13], for maintaining global-consistency [9], for fusing LiDAR data [8], and for use in robotic path-planning [10]. Distance function-based representations are, at present, one of the most widely used for dense mapping.

In order to maintain map consistency, several works have explored localization in the context of SDF-based Simultaneous Localization And Mapping (SLAM). The most common approach is to detect image features in RGB-D frames and leverage successful localization systems developed for visual SLAM [8], [14], [15]. BundleFusion [14], for example, relies on Scale-Invariant Features Transform (SIFT) image feature matching [16]. In a more specialized approach, Glocker *et al*. [17] suggest a feature designed specifically for RGB-D frames.

The use of image features has several disadvantages in the context of this work. Firstly, observation data produced by some common sensors do not contain image data, for example time-of-flight based cameras or LiDAR (see Sec. VI-C). Secondly, large viewpoint or illumination differences during place-revisiting can render image-based localization challenging [18]. 3D data in this context has complimentary properties. Lastly, performing localization on the SDF-based map allows for a natural means of localizing between differing sensor modalities. With these motivating reasons, we turn our attention to place-recognition based on 3D data alone.

B. Geometry-based Localization

Geometry-based localization has received considerable research focus. One common approach is to match current observations to an existing map using the same techniques used to generate constraints between successive poses [19], [20]. The primary challenge with this approach is that registration techniques converge to local-minima if not initialized sufficiently close to the globally-optimal solution. Registration is therefore typically attempted with several initializations, with the expectation that one will converge to the correct solution [19]. Google's Cartographer System [20] improves this approach by employing a branch and bound technique to efficiently prune candidates early in the matching process, effectively increasing the number of proposals that can be tested. In general, however, this class of techniques requires an accurate pose prior for localization to function. This limits their efficacy to small environments or to trajectories without long periods of exploratory motion. In contrast, the method presented in this paper is intended for *global*-localization.

Local geometric features and robust registration algorithms such as Random Sample Consensus (RANSAC) enable pose estimation independent of priors. Several features for this purpose have been suggested. Bosse and Zlot [2] extract 3D Gestalt descriptors from pointcloud data, and use a voting system to determine localization candidates. Steder *et al.* [3] suggest to extract Normal-Aligned Radial Features (NARF) features from pointclouds, and combine this with a Bag of Words (BoW)-based approach for global correspondence search. Segmatch [4] uses segments, contiguous elements in the scene, as the basis for description and matching. The authors of [6] generate a collection of scene fragments by accumulating consecutive RGB-D frames, and perform registration using a popular pointcloud-based descriptor, Fast Point Feature Histograms (FPFH) [5]. Gawel *et al.* [21] present an evaluation of several of these features for localization.

All of these approaches to 3D localization utilize pointclouds as the basis for place description, and are thus limited to capturing the geometry of surfaces only. Our previous work [11] introduced *freespace features* for geometry-based localization in the context of 2D LiDAR-based mapping. We proposed extracting features *directly* on a distance function representation of the scene. The results showed that this approach gave significant performance increases over a baseline 2D pointcloud-based descriptor, which we showed was primarily due to the utilization of features extracted in freespace. In the present proposal we continue this investigation. First we extend our approach to 3D and address the additional challenges that this presents, such as determination of Local Reference Frames (LRFs). Secondly, we introduce further utilization of the SDF for localization, by integrating an SDF-based fitness measure into the robust registration pipeline, which we found to further increase performance.

III. Preliminaries

In this section we describe the notation used throughout this paper. Coordinate frames are upper-case letters. Vectors are denoted as bold lower case letters, e.g. $\mathbf{p}_A \in \mathbb{R}^3$, a 3-dimensional vector expressed in reference frame A. Matrices are bold upper-case letters, e.g. $\mathbf{A}$. We denote the rigid transformation between coordinate frames A and B as $\mathbf{T}_{AB} \in \mathrm{SE}(3)$, such that $\mathbf{p}_A = \mathbf{T}_{AB}\mathbf{p}_B$. Central to our approach is the SDF for representing geometry. As is common in modern reconstruction systems, this function is stored as a spatially sparse set of samples over a voxel grid, where the sparsity comes from the fact that only observed voxels are stored. We denote the space of observed voxel indices as $\Psi \subset \mathbb{Z}^3$. The SDF is then described by two functions, $\Phi : \Psi \to \mathbb{R}$ and $\omega : \Psi \to \mathbb{R}$, which respectively map points in observed space to the signed distance to the nearest observed surface, and to a weighting/confidence measure.

IV. Problem Statement

Given a set of SDF submaps $\{\mathbb{S}_k\}_{k=1}^N$ and associated coordinate frames $\{S_k\}_{k=1}^N$, we consider the problem of finding pairs of submaps $(\mathbb{S}_i, \mathbb{S}_j)$ that correspond to the same location in the environment, that is, contain significant overlap in the region of the environment they describe. In

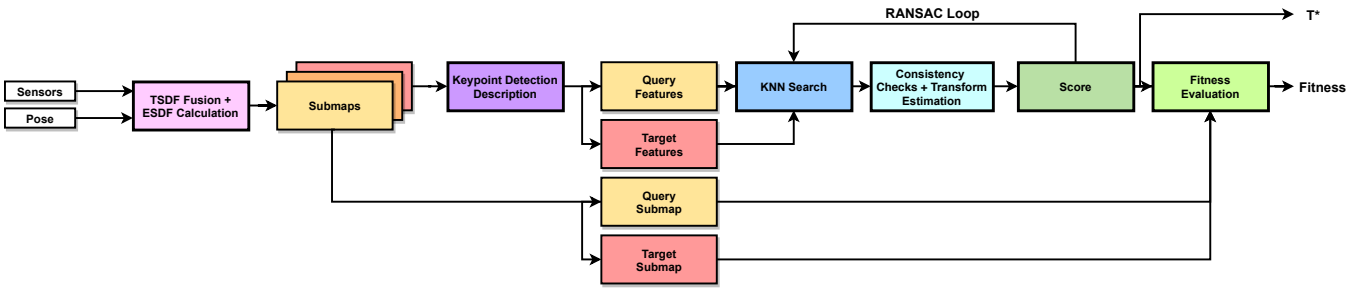

Fig. 2: A diagram detailing data flow in our system. The front-end that produces submaps is a product of our previous works [8], [15]. The proposed method identifies submaps within a collection that describe overlapping regions in the environment.

addition, we aim to determine $\mathbf{T}_{S_i S_j} \in \mathrm{SE}(3)$, the rigid transformation describing the submaps' relative pose.

V. Method

Our approach is described in detail in the following sections, with an overview of the steps shown in Fig. 2.

A. Submap Construction

The input to the proposed system is a set of submaps $\{\mathbb{S}_k\}_{k=1}^{N}$. For submap construction we make use of several of our previous works. Sensor data is integrated into a submap TSDF by ray casting into a spatially-hashed sparse voxel grid (see [10]). Periodically, we create a new submap by defining a voxel grid, with a coordinate frame co-located and aligned with the sensor pose at the time of submap creation (see [15]). After completion of a submap, we follow [8], [10] and extract the Euclidean Signed Distance Function (ESDF), as well as a set of points on the zero-level set. For the remainder of this paper, we will refer to the ESDF when discussing the SDF.

B. Keypoint Detection

We aim to detect keypoints that can be reliably re-detected during place-revisiting. The SDF is smooth by definition and has a gradient magnitude of 1 *almost everywhere*. As a result, typical keypoint detectors, which have high response in areas of large gradient, at image corners for example, do not function well when applied to the SDF. Therefore, we detect keypoints in areas of high curvature, as local-extrema of the Determinant of Hessian (DoH) volume, $\delta : \Psi \to \mathbb{R}$. In particular, for a voxel with center indices $\mathbf{k} \in \Psi$,

$$\delta(\mathbf{k}) = \det(\mathbf{H}(\mathbf{k})), \tag{1}$$

with $\mathbf{H}(\mathbf{k})$ the Hessian,

$$\mathbf{H}(\mathbf{k}) = \begin{bmatrix} h_{xx}(\mathbf{k}) & h_{xy}(\mathbf{k}) & h_{xz}(\mathbf{k}) \\ h_{yx}(\mathbf{k}) & h_{yy}(\mathbf{k}) & h_{yz}(\mathbf{k}) \\ h_{zx}(\mathbf{k}) & h_{zy}(\mathbf{k}) & h_{zz}(\mathbf{k}) \end{bmatrix}, \tag{2}$$

where the components are computed through application of Sobel derivative kernels, as well as additional Gaussian blur with tunable variance, set to $\sigma_{\text{grad}} = 2$ voxels for the remainder of this work. The first Hessian element $h_{xx}(\mathbf{k})$, for example, is given by:

$$h_{xx}(\mathbf{k}) = (\mathbf{D}_x * \mathbf{D}_x * \mathbf{G} * \Phi)(\mathbf{k}) \tag{3}$$

$$= (\mathbf{K} * \Phi)(\mathbf{k}). \tag{4}$$

Here, $\mathbf{D}_x$ is the first-order Sobel derivative kernel in the x direction, $\mathbf{G}$ a Gaussian kernel, Φ is the SDF, and $*$ the convolution operator. Note that because our data is distributed sparsely over the voxel grid, valid computation of the components of $\mathbf{H}$ at $\mathbf{k}$ requires that Φ be defined over the support of $\mathbf{K}$ at $\mathbf{k}$. A voxel is selected as a keypoint if it is an extremum point relative to its immediate neighbours. For submap $\mathbb{S}_i$, the set of keypoint locations $\{\mathbf{k}_{S_i}^j\}_{j=1}^{M}$ is passed to the next stages for description.

C. Local Reference Frame Assignment

To achieve invariance of the proposed descriptor to submap poses, we assign a LRF to each keypoint and perform description in this local frame. Several methods for this task have been suggested (see [22] for a review), however many of the proposed methods rely on the presence of a strong surface normal. Our keypoints are, in general, not located on surfaces, and as a result these methods are not applicable in this context. We found that an approach based on an Eigenvalue decomposition of the gradient field within the descriptor support to generate the most repeatable frames.

We compute the gradient field of the submap, $\mathbf{g}_{\text{raw}} : \Psi \to \mathbb{R}^3$, through convolution with the Sobel kernel by reusing intermediate results generated in keypoint detection. For each keypoint we weight the gradient vectors in the descriptor support, a sphere with radius r_f, with a Gaussian weighting function of tunable variance, set to $\sigma_{\text{desc}} = r_f$ for the remainder of this work. This gives more importance to gradients close to the descriptor center, a common approach. We construct the structure tensor for a keypoint located at $\mathbf{k}$ based on voxels within the descriptor support Ω,

$$\mathbf{S}_\Omega(\mathbf{k}) = \sum_{\mathbf{k} \in \Omega} \mathbf{S}(\mathbf{k}), \tag{5}$$

where

$$\mathbf{S}(\mathbf{k}) = \begin{bmatrix} g_x(\mathbf{k})^2 & g_x(\mathbf{k})g_y(\mathbf{k}) & g_x(\mathbf{k})g_z(\mathbf{k}) \\ g_y(\mathbf{k})g_x(\mathbf{k}) & g_y(\mathbf{k})^2 & g_y(\mathbf{k})g_z(\mathbf{k}) \\ g_z(\mathbf{k})g_x(\mathbf{k}) & g_z(\mathbf{k})g_y(\mathbf{k}) & g_z(\mathbf{k})^2 \end{bmatrix}, \tag{6}$$

where $g_x(\mathbf{k})$ is the x component of the weighted gradient vector, $\mathbf{g}$ at $\mathbf{k}$. The (unsigned) axes of our LRF are the Eigenvectors of $\mathbf{S}_\Omega$, $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$, sorted in descending order according to their corresponding Eigenvalues. To assign directionality to the axes, we project gradient vectors within

the descriptor support onto the Eigenvectors. We set the first axis of the LRF, $\mathbf{a}_1$ as

$$\mathbf{a}_1 = \begin{cases} \mathbf{v}_1 & \text{if } s \geq k_{\text{axis}} \\ (\mathbf{v}_1, -\mathbf{v}_1) & \text{if } -k_{\text{axis}} \leq s \leq k_{\text{axis}} \\ -\mathbf{v}_1 & \text{if } s \leq -k_{\text{axis}} \end{cases} \tag{7}$$

where

$$s = \frac{\sum_{\mathbf{k}\in\Omega} \mathbf{g}(\mathbf{k}) \cdot \mathbf{v}_1}{\sum_{\mathbf{k}\in\Omega} |\mathbf{g}(\mathbf{k}) \cdot \mathbf{v}_1|}, \tag{8}$$

$\mathbf{v}_1$ is the first Eigenvector. We perform this procedure for the 1st and 3rd Eigenvectors, and choose the remaining axis direction to complete a right handed coordinate system. Note that, in the case that $-k_{\text{axis}} \leq s \leq k_{\text{axis}}$ we create multiple LRFs, one with each direction of the relevant Eigenvector, and perform the description detailed in the Sec. V-D for each of the LRFs. We set $k_{\text{axis}} = 0.5$ for the remainder of this work. The output of this stage is rotation matrix $\mathbf{R}_{F_i S} \in \text{SO}(3)$ describing the rotation from the submap frame S to the i^{th} feature frame F_i.

D. Description

For each keypoint and each associated LRF, we compute a description based on a gradient orientation histogram, popularized by the Histogram of Oriented Gradients (HOG) [23] and SIFT [16] image features. For the i^{th} feature we rotate the weighted gradients from the submap frame, S, to the feature frame, F_i,

$$\mathbf{g}_{F_i}(\mathbf{k}) = \mathbf{R}_{F_i S} \mathbf{g}_S(\mathbf{k}), \quad \forall \mathbf{k} \in \Omega \cap \Psi, \tag{9}$$

where $\Omega \cap \Psi$ denotes valid voxels in the descriptor support, and $\mathbf{R}_{F_i S}$ is described by the feature LRF computed as described in Sec. V-C.

We then convert the weighted gradients to spherical coordinates and compute a weighted histogram by dividing azimuth $\phi \in [-\pi, \pi]$ and elevation $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ angles into $n_{div}/180°$ divisions. We employ soft binning such that each gradient vector contributes its magnitude to the surrounding bins through bilinear interpolation, as suggested by [23]. We normalize the histogram counts by the number of voxels in the descriptor support, and each bin by its solid angle (see [24] for details).

We augment the gradient orientation histogram with two further pieces of information. Firstly, we compute the weighted sum of the SDF under the descriptor support

$$b_{\text{dist}} = \frac{1}{N_{b_{\text{dist}}}} \sum_{\mathbf{k}\in\Omega\cap\Psi} k_{\text{gauss}}(\mathbf{k}) \Phi(\mathbf{k}). \tag{10}$$

Where k_{gauss} is the same central weighting function used to weight the gradient vectors, and we normalize by

$$N_{b_{\text{dist}}} = \sum_{\mathbf{k}\in\Omega\cap\Psi} k_{\text{gauss}}(\mathbf{k}). \tag{11}$$

The addition of b_{dist} biases matching of features in free-space to other features at a similar distance to surfaces.

Secondly, we augment the descriptor with the type of curvature at each keypoint. In particular, we compute

$$b_{\text{class}} = \sum_{i=1}^{3} [e_i > 0], \tag{12}$$

where e_i is the Eigenvalue associated with the Eigenvector $\mathbf{v}_i$ (calculated in Sec. V-C), and $[\cdot]$ returns 1 if the condition is true and 0 otherwise. The addition of b_{class} to the descriptor biases matching towards features of the same stationary point type. For example, keypoints at maxima of the distance function are biased towards matching other maxima.

We concatenate the $\alpha_{\text{dist}} b_{\text{dist}}$ and $\alpha_{\text{class}} b_{\text{class}}$ to a flattened (one dimensional) view of the gradient histogram, where α_{dist} and α_{class} are weighting constants and take the values $1e^{-7}$ and $1e^{-5}$ for the remainder of this work. The final descriptor vector has size $2 \times n_{div}^2 + 2$.

E. Registration

We treat the place recognition problem as one of pairwise matching. Given two submaps, $(\mathbb{S}_i, \mathbb{S}_j)$, representing a putative match, we aim to determine if they are indeed a match, as well as the transform relating their poses $T_{S_i S_j}$. We implement a RANSAC-based registration pipeline for this purpose. First, we generate correspondences by performing a K-Nearest-Neighbour search for all keypoints in the query submap, within the target submap (we choose $K = 5$ neighbours for the remainder of this work). We accumulate all correspondences in the set $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$, where c_i is a pair of corresponding keypoints i.e. $c_i = \{\mathbf{p}^k_{S_i}, \mathbf{p}^l_{S_j}\}$, with $\mathbf{p}^k_{S_i} \in \mathbb{R}^3$ is the location of the k^{th} keypoint in submap i.

We select $M = 3$ correspondences at random, resulting in a correspondence sub-set, $\mathcal{C}_{\text{sub}} = \{c_i, c_j, c_k\} \subset \mathcal{C}$. We test this set for geometric consistency by checking that the distances between correspondence endpoints are similar in query and target submaps, a technique first suggested in [25] and implemented in Open3D [26]. In particular, we reject correspondence sub-sets not meeting

$$k_{consist} d_t(c_i, c_j) < d_q(c_i, c_j) < \frac{1}{k_{consist}} d_t(c_i, c_j) \tag{13}$$

for all combinations of correspondences pairs $\{c_i, c_j\} \in \mathcal{C}_{\text{sub}}$, where $d_t(\cdot,\cdot)$ measures the distances between the correspondence endpoints in the target submap, and d_q is similarly defined. We set $k_{consist} = 0.9$ for the remainder of this work. Correspondence sets passing the consistency check are used to estimate transformation candidates $\mathbf{T}_{S_i S_j}$. We rank these candidates based on inlier count,

$$score(\mathbf{T}_{S_i S_j}) = \sum_{c_i \in \mathcal{C}} [d(c_i, \mathbf{T}_{S_i S_j}) < k_{\text{dist}}] \tag{14}$$

where

$$d(c_i, \mathbf{T}_{S_i S_j}) = \left\| \mathbf{p}^k_{S_i} - \mathbf{T}_{S_i S_j} \mathbf{p}^l_{S_j} \right\|_2 \tag{15}$$

where $\mathcal{C}$ denotes a sum over the correspondence set, and $\mathbf{p}^i_{S_k}$ and $\mathbf{p}^j_{S_l}$ are the positions of keypoints i and j respectively, and $[\cdot]$ returns 1 if the condition is true and 0 otherwise. Note

that k_{dist} is dependent on the scale of the environment and is set to various values in our evaluations.

F. SDF-based Fitness Evaluation

Following RANSAC, we calculate a fitness measure for the transformation candidate, $\mathbf{T}^*_{S_iS_j}$, achieving the highest inlier score, eq. (14). Global localization decisions are made on the basis of this fitness measure, which we found to lead to improvements in performance, while introducing negligible additional run-time cost.

To calculate the fitness of transformation $\mathbf{T}^*_{S_iS_j}$, we transform isosurface points from submap j, $\mathcal{P}_{S_j} = \{\mathbf{p}^i_{\text{iso},S_j}\}_{i=1}^N$, into submap i and calculate a weighted average of the SDF values at these points. This operation is performed bidirectionally, i.e. we also transform iso-surface points in submap i into submap j. Given a complete, perfectly reconstructed distance function, and the true relative transformation, this score is zero, as all iso-surface points of one submap lie on the SDF zero-level set of the other. The expectation in a more realistic setting is that the closer this measure is to zero, the more accurate the transform $\mathbf{T}^*_{S_iS_j}$. In particular, the fitness cost for $\mathbf{T}^*_{S_iS_j}$ is given by,

$$f(\mathbf{T}^*_{S_iS_j}) = -\frac{1}{N}\left(d_{\text{iso},S_i}(\mathcal{P}_{S_j}, \mathbf{T}^*_{S_iS_j}) + d_{\text{iso},S_j}(\mathcal{P}_{S_i}, \mathbf{T}^{*-1}_{S_iS_j})\right), \tag{16}$$

where d_{iso,S_1} is the weighted sum of the SDF values,

$$d_{\text{iso},S_i}(\mathcal{P}_{S_j}, \mathbf{T}^*_{S_iS_j}) = \sum_{\mathbf{p}_{\text{iso}} \in \mathcal{P}_{S_j}} \omega_{S_i}(\mathbf{T}^*_{S_iS_j}\mathbf{p}_{\text{iso}})\Phi_{S_i}(\mathbf{T}^*_{S_iS_j}\mathbf{p}_{\text{iso}}), \tag{17}$$

and d_{iso,S_j} is defined similarly. We normalize by the sum of all weights such that,

$$N = \sum_{\mathbf{p}_{\text{iso}} \in \mathcal{P}_{S_j}} \omega_{S_i}(\mathbf{T}^*_{S_iS_j}\mathbf{p}_{\text{iso}}) + \sum_{\mathbf{p}_{\text{iso}} \in \mathcal{P}_{S_i}} \omega_{S_i}(\mathbf{T}^*_{S_jS_i}\mathbf{p}_{\text{iso}}). \tag{18}$$

The SDF function Φ and ω are defined sparsely and thus evaluation is not generally possible at all $\mathbf{p}_{\text{iso}}$. We require a minimum fraction of points in $\mathcal{P}_{S_i}$ and $\mathcal{P}_{S_j}$ to return valid distances, otherwise the match is not considered. We set this overlap as $k_{\text{overlap}} = 15\%$ for all evaluations in this work. Note that the fitness score (16) has units of of metric length.

VI. Results

In this section, we aim to validate the hypotheses of this paper, that a) the proposed system enables global place recognition in a collection of SDF submaps by extracting features *directly* on the SDF; that b) the use of these features can increase localization performance when compared to existing features computed based on surface information only; and that c) the use of free-space information is integral to the improvements in performance we show.

A. Parameters

The performance of both our proposal and the baseline methods are affected by parameters. We found our proposal to be robust across the various environments in this section and use fixed parameter values as described in the method

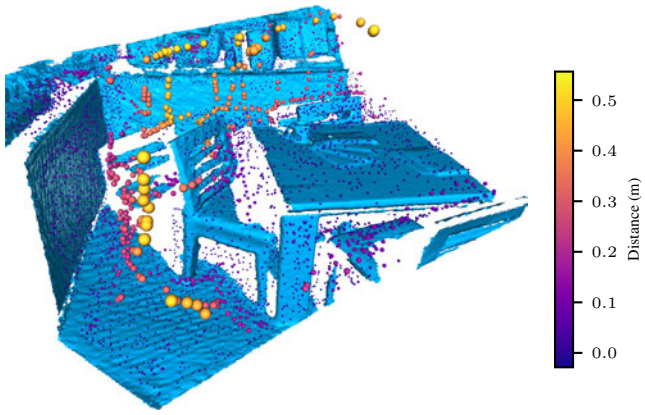

Fig. 3: A scene fragment from the 3DMatch dataset [27], analyzed in Sec. VI-B. Colored spheres show the keypoint locations described by our method. The size and color of the keypoints are modulated by the distance from the keypoint to the closest surface. The plot shows features located in free-space; these are particularly visible in the foreground, between the chair and the cabinet.

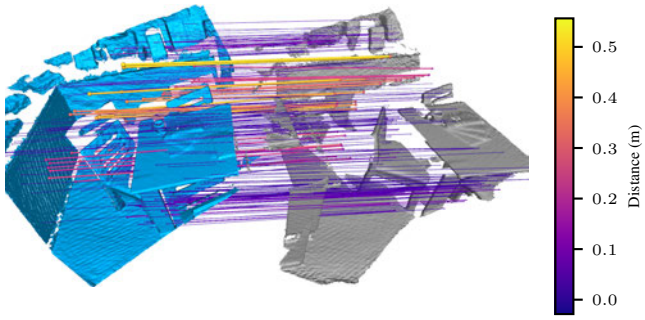

Fig. 4: An example match between two fragments in the 3DMatch dataset [27]. We show a random subset of inlier features and correspondences that generated the match. The size and color of the keypoints and correspondence lines are modulated by the distance from the keypoint(s) to the closest surface. This alignment resulted in ~1200 inliers.

in all evaluations. We set the feature radius $r_f = 15$ voxels, and number of angular divisions $n_{\text{div}} = 10\,\text{divs}/180°$. For FPFH and Signature of Histograms of OrienTations (SHOT) in the 3DMatch [27] dataset, we use feature radius and normal calculation radius values provided by the authors in [28], where a similar analysis is performed. For the Deutsches Museum dataset we grid searched for optimal parameters, resulting in feature radii of 1.5 m and 3.0 m for FPFH and SHOT respectively. We selected the scale-dependant RANSAC inlier threshold, k_{dist}, for all methods through a grid search for values leading to optimal results.

B. 3DMatch Dataset

We quantify the performance of the proposed method for pairwise alignment of scene fragments using the 3DMatch dataset [27]. This dataset consists of reconstructions of 8 indoor scenes scanned with an RGB-D sensor. We use the provided toolbox to fuse sequences of consecutive RGB-D images to form partially overlapping surface pointclouds. Additionally, we generate SDFs using cblox [15], which form the input to our system. We follow the approach in [27] and sample 5000 random surface points as keypoints for the surface-based descriptors. To control for feature number,

Method	Recall at Precision 0.8								
	3DMatch Validation Dataset Index[2]								
	#1	#2	#3	#4	#5	#6	#7	#8	Mean
fpfh (de.)	0.46	0.21	0.10	0.12	0.14	0.20	0.31	0.12	0.21
shot (de.)	0.54	0.23	0.15	0.13	0.14	0.22	0.34	0.11	0.23
fpfh (sp.)	0.46	0.34	0.13	0.20	0.19	0.35	0.33	0.11	0.26
shot (sp.)	*0.57*	*0.39*	*0.16*	*0.23*	*0.25*	**0.44**	*0.35*	*0.14*	*0.32*
freespace	**0.70**	**0.44**	**0.20**	**0.28**	**0.25**	*0.43*	**0.36**	**0.16**	**0.35**
Rel. (%)	**21.8**	**12.1**	**20.9**	**20.4**	**2.8**	-1.8	**2.3**	**13.9**	**11.5**

TABLE I: Results of pairwise matching on the 3DMatch dataset [27]. Two baseline surface-based features, FPFH and SHOT, with two different RANSAC validation methods, correspondence-based (sp) and pointcloud-based (de). The final line shows percentage performance difference between the proposed method and the best surface descriptor. Best performance is in bold text, the second best in italics.

Method	Recall at Precision 0.8						
	Deutsches Museum Dataset Index[3]						
	#1	#2	#3	#4	#5	#6	Mean
fpfh (vox)	0.31	0.57	0.39	0.42	0.36	0.41	0.41
fpfh (iso)	0.34	0.61	0.44	*0.48*	0.44	0.46	0.46
shot (vox)	0.21	0.55	0.40	0.44	0.39	0.41	0.40
shot (iso)	*0.39*	*0.61*	*0.48*	0.46	*0.47*	*0.48*	*0.48*
freespace	**0.58**	**0.68**	**0.51**	**0.50**	**0.56**	**0.56**	**0.56**
Rel. (%)	**49.5**	**10.9**	**7.4**	**4.3**	**19.3**	**15.5**	**17.8**

TABLE II: Results of the Deutsches museum experiment described in Sec. VI-C. The table shows recall values generated by each method at 0.8 precision. The final line shows percentage performance difference between the proposed method and the best surface descriptor. Best performance is in bold text, the second best in italics.

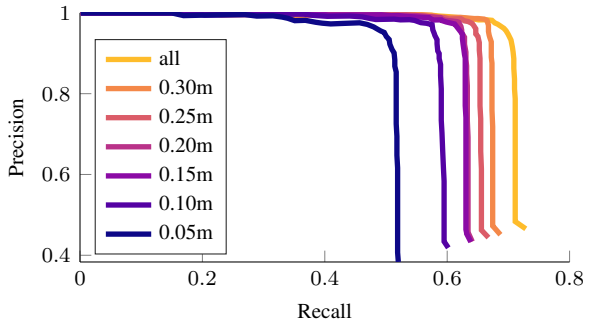

Fig. 5: Precision-recall curves generated by eliminating features in free-space to varying degrees (Sec. VI-B). The curve for 0.05 m for example only permits feature extraction within 5 cm of a surface. The figure shows that features in free-space contribute significantly to the proposed method's performance.

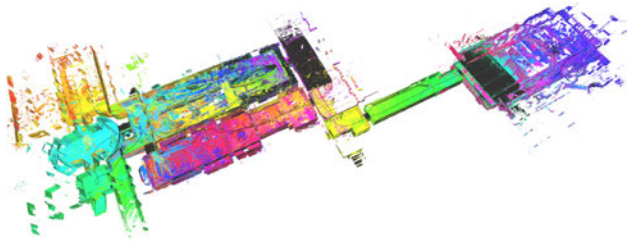

Fig. 6: A mesh extracted from an SDF map of part of the Deutsches museum (Sec. VI-C). The reconstruction is performed using Cartographer [20] and cblox [15]. The color of the mesh at each location indicates the submap contributing to the reconstruction at that location.

we also limit our proposed method to extracting the 5000 keypoints, chosen to have the strongest DoH response, eq. (1). An example scene fragment and extracted features are shown in Fig. 3. To determine ground truth match/non-match labels, we calculate the overlapping volume between each pair of voxel grids. Pairs sharing more than $1\,\mathrm{m}^3$ overlap are considered ground-truth matches. Estimated labels are computed by thresholding fitness values from registration. We generate precision-recall curves by varying this fitness threshold. For pairs testing positive for a match, we additionally require the relative fragment pose to be within 0.2 m of the true value.

We compare the proposed approach to FPFH [5] and SHOT [7] features, which have shown state-of-the-art performance in comparisons of handcrafted features [21], and have open-source implementations released with PCL [29]. For both features, we test two RANSAC-based registration pipelines. The first was proposed with FPFH in [5]. This approach uses a pointcloud-based fitness measure, and is implemented as part of Open3D [26]. The second approach uses a fitness measure based on keypoint distance, analogous to our $score$ (14). We denote these methods *dense* and *sparse* respectively in Table I. Note that we perform the geometric consistency checks described in Sec. V-E for all methods. We set the number of RANSAC iterations to $4e^6$ for all experiments.

Table I shows the results of this experiment, summarized as recall rates at precision 0.8. Some generalizations are possible. SHOT outperformed FPFH, which agrees with similar analyses [21], [28], and reflects SHOTs greater dimensionality: 352 vs. 33. Registration using the keypoint-based fitness measure, rather than the method proposed in [5], led to better results for both feature types. Note that the approach of selecting (random) keypoints and computing correspondences for these keypoints-only, also avoids dense correspondence computation. The proposed SDF-based feature outperformed the competing methods, leading to an average improvement in recall rates of 11.5% with respect to the runner-up method, SHOT (sparse). The proposed descriptor has dimension 202, lying between the dimensionality of the baseline methods.

Finally, we analyze the contribution of features in free-space to the performance of the proposed method. We generate precision-recall curves, as before, however we limit the proposed method to extracting features near surface boundaries, to varying degrees. In particular, we restrict feature extraction such that each keypoint lies at a distance from surfaces less than a threshold, d_{lim} for $d_{\text{lim}} \in \{0.30, 0.25, 0.20, 0.15, 0.10, 0.05\}$ m. Fig. 5 results of this study for 3DMatch dataset #1. The curves show that the use of features in free-space generates significant performance gains, validating one of the hypotheses of this paper.

[2]Datasets referenced: 1: kitchen, 2: hotel 3, 3: study, 4: hotel 2, 5: home 1, 6: hotel 2, 7: hotel 1, 8: mit lab

[3]Datasets referenced: 1: b3-2016-01-19-13-50-11, 2: b3-2016-03-01-13-39-41, 3: b3-2016-02-09-13-17-39, 4: b3-2016-04-05-13-54-42, 5: b3-2016-02-09-13-31-50, 6: b3-2016-02-02-13-33-30

Additionally, this experiment provides an indication that in circumstances where the SDF-geometry in free-space is disturbed, by dynamic objects for example, smooth decays in performance towards surface-only features, are to be expected.

C. Deutsches Museum

In order to evaluate our system for use with 3D LiDAR data we perform evaluations on the Deutsches Museum dataset, released as part of Google's Cartographer SLAM system [20]. This dataset provides sensor data from backpack mounted 3D LiDARs and an Inertial Measurement Unit (IMU), and was collected during walking traversals through a museum. We select 6 trajectories of varying sizes with which to perform evaluation. The trajectories range in length between ~5 minutes/35 submaps and ~20 minutes/136 submaps.

To produce a collection of SDF submaps with which to evaluate our system, we compute a globally optimized trajectory using Cartographer [20] and accumulate LiDAR data in submap TSDFs using cblox [15]. Fig. 6 shows an example map, where varying surface color indicates the change in the submap contributing to the reconstruction at each location. To produce pointcloud submaps for surface descriptors we follow two approaches. In the first approach, we fuse sensor pointclouds using voxel filtering. In the second, we extract iso-surface pointclouds from the SDF submaps described above. The result of these processes are collections of co-located submaps, computed from the same sensor data, represented as both SDFs and pointclouds. We generate groundtruth and estimated match/non-match labels using submap overlap and fitness thresholds as described in Sec. VI-B.

We compare our approach against FPFH and SHOT as in Sec. VI-B. For FPFH we follow the approach proposed in [5]; we extract features densely and use a pointcloud-based registration pipeline to avoid computing dense correspondences. For SHOT we follow the approach which led to the best results in Sec. VI-B: we randomly select 5000 surface points as keypoints for feature extraction and use a keypoint-based fitness measure. Again, we also limit our proposed method to extracting at most 5000 keypoints.

Table II summarizes the results of this experiment as recall values at precision 0.8. Some generalizations can be made. For the baseline features, iso-surface pointcloud extraction produced better results than voxel filtering, leading to mean improvements of 12.6% for FPFH, and 26.7% for SHOT. This is a somewhat reassuring result. To generate an SDF iso-surface, pointcloud data is first fused into a voxelized TSDF, which one might assume to introduce some discretization error when compared with the voxel filtering approach. These results show that, in this case at least, TSDF fusion has a positive effect on performance. Our proposed feature outperforms the baseline methods. With respect to the runner-up method, which was SHOT (iso-surface) in all but one experiment (in which it was FPFH (iso-surface)), our proposal generates a mean improvement of 17.8%. The average freeture computation time in this experiment was ~7 s per submap, on a single CPU thread of the on-board processor of the MAV described in Sec. VI-D.

D. MAV Dataset

To demonstrate the utility and generalization of our system for robotic applications, we demonstrate its use for localizing an Micro Aerial Vehicle (MAV) against an existing map. We recorded LiDAR, camera and inertial measurements from sensors attached to an MAV (Fig. 7) flying through a search and rescue training area. The target trajectory was ~500 m in length (map area: ~120 m×120 m), and the query trajectory ~200 m (map area: ~70 m×70 m). We performed visual-inertial odometry using ROVIO [30] for local motion tracking and built a globally consistent submap-based map using our previous work *voxgraph* [8]. Fig. 7 shows the two maps positioned relative to one another using Real-Time Kinematic (RTK)-GPS (with the query map raised in height for clarity). The section of overlap occurs at the ruins of a badly damaged building, see Fig. 7 (top left) for a photo. We use the RTK-GPS system to determine ground-truth submap poses and generate ground-truth match/non-match labels.

We perform localization by exhaustively matching each query submap against all submaps in the target collection and calculate estimated labels. We achieve a recall rate of 0.54 at precision 0.8 indicating the applicability of the proposed system to a new environment, with only a single parameter change k_{dist}. Fig. 8 shows the results of one successful match linked by the inlier features that produced it, and Fig. 1 the final registration.

VII. Conclusion

In this paper we have presented a novel approach for localization in maps represented as collections of SDF submaps. Our system extracts features directly on the SDF, allowing description of both surface and free-space geometry. We suggest an interest point detector based on extrema of the DoH volume, which selects for regions of high curvature in the distance field. Keypoints are described using a spherical gradient histogram, augmented with two pieces of SDF specific information: feature distance, and curvature class. We test our approach on a variety of environments and sensors. In the RGB-D camera-based 3DMatch dataset [27], we show a mean improvement of ~12% over the best baseline method. In the LiDAR-based Cartographer dataset [20] we show a mean improvement of ~18% over the closest baseline method. Finally, we demonstrate our method for localizing an MAV against an earlier flight through a search and rescue training ground. Our proposal requires only changing a single scale-dependant parameter when applied to these three different environments and sensing setups, an indication of its generality. Future work may investigate if the proposed approach, using free-space and the SDF *directly* for localization, could be applied to recently popular deep-learning-based approaches.

Fig. 7: Two maps used for the localization experiment described in Sec. VI-D. The maps are reconstructed using using our previous work voxgraph [8] and data collected during two flights of an MAV equipped with a monocular camera, inertial sensing, a 3D LiDAR (visible in the lower-right of the figure). The maps partially overlap at the ruins in the top left of the figure. A successful localization is shown in Figs. 8 and 1.

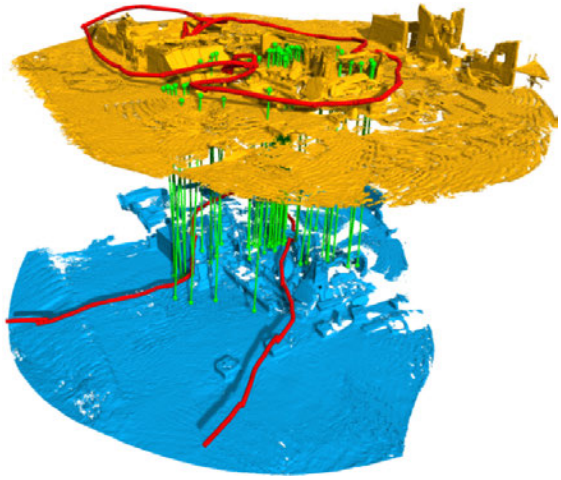

Fig. 8: A successful submap-submap match between the query map (yellow) and the target map (blue). The query map is shown elevated with respect to target map. Inlier features leading to the match are shown in green with vertical bars connecting corresponding features. The final alignment, following Iterative Closest Point (ICP), is shown in Fig. 1

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: Toward the robust-perception age," *TRO*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] M. Bosse and R. Zlot, "Place recognition using keypoint voting in large 3d lidar datasets," in *ICRA*, pp. 2677–2684, 2013.

[3] 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 *IROS*, pp. 1249–1255, 2011.

[4] R. Dubé, A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart, and C. Cadena, "Segmap: Segment-based mapping and localization using data-driven descriptors," *IJRR*, vol. 39, no. 2-3, pp. 339–355, 2020.

[5] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *ICRA*, pp. 3212–3217, 2009.

[6] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *CVPR*, pp. 5556–5565, 2015.

[7] S. Salti, F. Tombari, and L. Di Stefano, "Shot: Unique signatures of histograms for surface and texture description," *CVIU*, vol. 125, pp. 251–264, 2014.

[8] V. Reijgwart, A. Millane, H. Oleynikova, R. Siegwart, C. Cadena, and J. Nieto, "Voxgraph: Globally consistent, volumetric mapping using signed distance function submaps," *RA-L*, vol. 5, no. 1, pp. 227–234, 2019.

[9] N. Fioraio, J. Taylor, A. Fitzgibbon, L. Di Stefano, and S. Izadi, "Large-scale and drift-free surface reconstruction using online subvolume registration," in *CVPR*, pp. 4475–4483, 2015.

[10] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," in *IROS*, pp. 1366–1373, 2017.

[11] A. Millane, H. Oleynikova, J. Nieto, R. Siegwart, and C. Cadena, "Free-space features: Global localization in 2d laser slam using distance function maps," in *IROS*, pp. 1271–1277, 2019.

[12] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *ACM symposium UIST*, pp. 559–568, 2011.

[13] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3d reconstruction at scale using voxel hashing," *ACM ToG*, vol. 32, no. 6, pp. 1–11, 2013.

[14] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM ToG*, vol. 36, no. 4, p. 1, 2017.

[15] A. Millane, Z. Taylor, H. Oleynikova, J. Nieto, R. Siegwart, and C. Cadena, "C-blox: A scalable and consistent tsdf-based dense mapping approach," in *IROS*, pp. 995–1002, 2018.

[16] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, vol. 2, pp. 1150–1157, Ieee, 1999.

[17] B. Glocker, J. Shotton, A. Criminisi, and S. Izadi, "Real-time rgb-d camera relocalization via randomized ferns for keyframe encoding," *TVCG*, vol. 21, no. 5, pp. 571–583, 2014.

[18] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *TRO*, vol. 32, no. 1, pp. 1–19, 2015.

[19] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments," in *RSS*, 2018.

[20] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *ICRA*, pp. 1271–1278, 2016.

[21] A. Gawel, R. Dubé, H. Surmann, J. Nieto, R. Siegwart, and C. Cadena, "3d registration of aerial and ground robots for disaster response: An evaluation of features, descriptors, and transformation estimation," in *SSRR*, pp. 27–34, 2017.

[22] A. Petrelli and L. Di Stefano, "On the repeatability of the local reference frame for partial shape matching," in *ICCV*, pp. 2244–2251, 2011.

[23] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, vol. 1, pp. 886–893, 2005.

[24] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *ACM MM*, pp. 357–360, 2007.
[25] H. Chen and B. Bhanu, "3d free-form object recognition in range images using local surface patches," *Pattern Recognition Letters*, vol. 28, no. 10, pp. 1252–1262, 2007.
[26] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.
[27] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *CVPR*, 2017.
[28] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *CVPR*, pp. 5545–5554, 2019.
[29] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *ICRA*, pp. 1–4, 2011.
[30] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback," *IJRR*, vol. 36, no. 10, pp. 1053–1072, 2017.