2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

CURL-MAP: Continuous Mapping and Positioning with CURL Representation†

Kaicheng Zhang[1,2], Yining Ding[2], Shida Xu[1,2], Ziyang Hong[2], Xianwen Kong[2] and Sen Wang[1]

***Abstract*— Maps of LiDAR Simultaneous Localisation and Mapping (SLAM) are often represented as point clouds. They usually take up a huge amount of storage space for large-scale environments, otherwise much structural detail may not be kept. In this paper, a novel paradigm of LiDAR mapping and odometry is designed by leveraging the Continuous and Ultra-compact Representation of LiDAR (CURL) proposed in [1]. Termed CURL-MAP (Mapping and Positioning), the proposed approach can not only reconstruct 3D maps with a continuously varying density but also efficiently reduce map storage space by using CURL's spherical harmonics implicit encoding. Different from the popular Iterative Closest Point (ICP) based LiDAR odometry techniques, CURL-MAP formulates LiDAR pose estimation as a unique optimisation problem tailored for CURL. Experiment evaluation shows that CURL-MAP achieves state-of-the-art 3D mapping results and competitive LiDAR odometry accuracy. We will release the CURL-MAP codes for the community.**

I. Introduction

3D mapping is a pivotal component for Simultaneous Localisation and Mapping (SLAM) and autonomous robots. The performance of 3D mapping is significantly affected by the representations of 3D maps. Among the various 3D map representations, point cloud stands out as the most prevalent due to its simplicity and rich algorithms and tools to process it, such as Point Cloud Library (PCL). However, most point cloud-based odometry/SLAM systems, e.g., LOAM [2] and its variants [3], downsample their point clouds for mapping due to computational and storage efficiency, which results in a loss of map resolution and 3D geometric details. Meanwhile, storing all points is not ideal since map size grows exponentially, causing significant processing overhead besides the ample storage space required.

Surfel representation is favoured in several SLAM systems [4], [5]. Nevertheless, its level of map detail largely depends on the number of surfels it has. Mesh maps, widely used for 3D modelling, provide compelling memory benefits, particularly in structured scenarios. However, to attain high accuracy, it is often necessary to incorporate considerable additional vertices. Furthermore, updating a mesh map, especially involving dynamic obstacles, requires re-establishing connections between vertices, which is computationally expensive.

CURL representation [1] was recently proposed to implicitly encode 3D LiDAR points using spherical harmonics functions. It is highly compact and is capable of performing continuous reconstruction. However, CURL has only been used to directly encode and reconstruct LiDAR points/maps. Whether it is suitable for LiDAR dense mapping and odometry has not been explored.

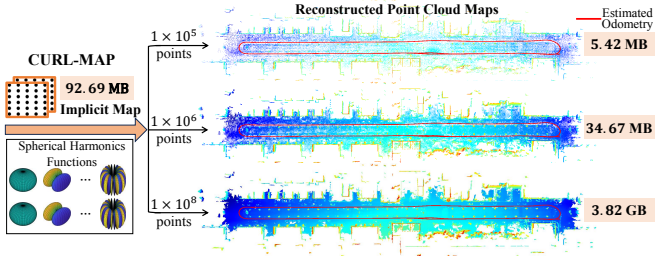


Fig. 1: Continuous map reconstruction of KITTI Sequence 06 using CURL-MAP. Point cloud maps with different resolutions are reconstructed using an identical CURL-MAP's implicit map. Storage sizes of our implicit map and the reconstructed point cloud maps are highlighted in ▭.

In this paper, we introduce the first CURL-based LiDAR mapping and odometry algorithm, named CURL-MAP (Mapping and Positioning). It is designed with a specific emphasis on 3D mapping, although it is also capable of positioning/localisation. Our main contributions include:

- A novel CURL-based LiDAR mapping and positioning system that utilises the CURL representation and achieves unprecedentedly more accurate, denser 3D map reconstruction with varying densities and more efficient storage space usage.
- A new pose estimation algorithm that leverages the implicit CURL map for LiDAR odometry without using the standard ICP methods.
- Experiment evaluation shows state-of-the-art mapping results and comparable odometry performance.

The unique use of CURL representation in CURL-MAP allows for recovering 3D maps with varying densities through the same CURL implicit map, as shown in Fig. 1. Its compact CURL map can be 300 times smaller than its densely reconstructed point cloud maps. Our code has been open-sourced for the community[1].

II. Related Work

In the prevailing LiDAR odometry and mapping frameworks, such as LOAM [2], Lego-LOAM [6], LIO-SAM [3] and FAST-LIO2 [7], the emphasis has been largely on using pose optimisation to enhance the accuracy of their

†CURL: Continuous, Ultra-compact Representation for LiDAR [1]

[1] I-X & Department of Electrical and Electronic Engineering, Imperial College London, UK {k.zhang23, s.xu23, sen.wang}@imperial.ac.uk

[2]School of Engineering and Physical Sciences, Heriot-Watt University, UK {kz13, yd2007, sx2000, zh9, x.kong}@hw.ac.uk

[1]https://github.com/SenseRoboticsLab/CURL-MAP.git

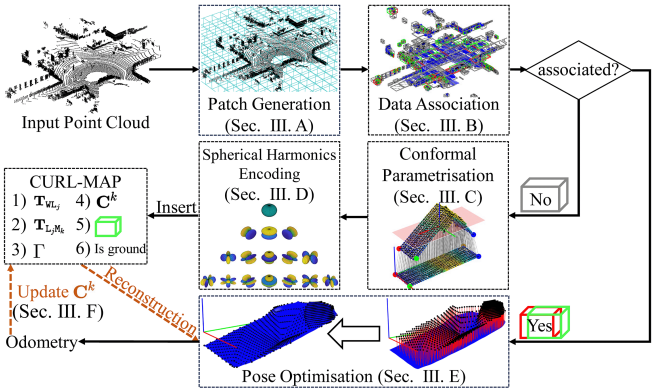


Fig. 2: Pipeline of the proposed CURL-MAP .

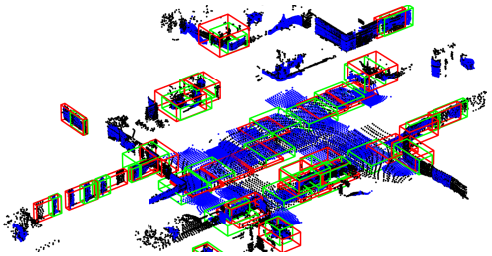


Fig. 3: Data association between patches in a scan and in the map. Black points are from LiDAR scan. Red bounding boxes show patches that are successfully paired with map patches (green bounding boxes). Blue points are reconstructed from CURL-MAP's map for visualisation only.

point cloud maps. While these results are commendable, accumulated odometry errors imposed by inherent sensor noises underscore the need for direct map optimisation, which likely offers more consistent map reconstruction. Furthermore, updating a point cloud map is challenging due to dealing with discrete 3D points.

ElasticFusion [8], to some extent, paves the way for vision-based map-centric methodologies by maintaining a deformation node list on a map instead of a pose graph. The idea is extended to LiDAR [5], [9], adopting both the surfel representation and the deformation graph within a surfel map context. While increasing the number of deformation nodes can reduce local spatial distortion, it also leads to an exponential rise in computational costs. As a result, it can be challenging for this method to map large environments with sufficient details.

Recently, several new map representations are proposed, such as the quadric representation [10] and the VoxelMap [11]. However, their primary focus remains on pose estimation, rather than detailed map reconstruction. The mesh-based approach, PUMA [12], provides a detailed mesh map and incorporates a point-to-mesh pose estimation method. But its real-time performance is limited. The recent SLAMesh [13] has achieved a real-time mesh-based SLAM system. SegMap [14] stands out by employing a CNN-derived descriptor to represent point cloud segments. Although it achieves reduced memory consumption, it is designed mainly as a loop closure technique and still depends on ICP for pose estimation. Thus, we propose a new implicit mapping method that supports map reconstruction with continuously varying resolution and enables LiDAR positioning without using ICP.

III. Proposed CURL-MAP Methodology

Fig. 2 shows the full pipeline of our CURL-MAP method. Instead of using the popular planar and edge features detected from LiDAR point clouds or the ICP methods, like LOAM [2] and its variants, the proposed CURL-MAP encodes point clouds as spherical harmonics coefficients following CURL in [1]. We first describe how patches are generated from a 3D LiDAR scan.

A. Patch Generation

Since ground and non-ground points provide different constraints to pose estimation, the current LiDAR scan j is first segmented into ground and non-ground points using patchwork++ [15]. We then divide the scan into a set of patches $M^k \in \mathbb{M}_j$ through a simple yet efficient volumetric clustering method, i.e., the 3D space is represented by fixed-size voxels and 3D LiDAR points that fall within the same voxel are clustered as a patch.

B. Data Association of Patches

For each patch $M^k \in \mathbb{M}_j$, it is checked whether it can be associated with a patch in the map $\mathbb{M}$. Assume $\mathbf{T}_{\mathtt{WL}_j}$, the transformation from j-th LiDAR frame $\mathtt{L}_j$ to the world frame $\mathtt{W}$, and $\mathbf{T}_{\mathtt{L}_j\mathtt{M}_k}$, the transformation from M^k's patch frame $\mathtt{M}_k$ to the LiDAR frame, are known, M^k can be transformed into $\mathtt{W}$ via $\mathbf{T}_{\mathtt{WM}_k} = \mathbf{T}_{\mathtt{WL}_j}\mathbf{T}_{\mathtt{L}_j\mathtt{M}_k}$. Its axis-aligned bounding box is then computed by finding its minimum and maximum values along the Euclidean axes of $\mathtt{W}$. Intersection over Union (IoU) is calculated between M^k and its nearby patches in $\mathbb{M}$ for data association. An IoU threshold is used to determine whether a pair of patches is an inlier. If a patch has multiple inlier associations, the one with the highest IoU score is chosen. An example of data association is given in Fig. 3. For patches that are associated with the map, they are directly used for pose optimisation (see details in Section III-E).

For patches that are not associated with the map and have IoUs less than threshold β, they are inserted into the map by generating their CURL representation which has the following attributes: 1) $\mathbf{T}_{\mathtt{WL}_j}$; 2) $\mathbf{T}_{\mathtt{L}_j\mathtt{M}_k}$; 3) quasi-conformal mapping function Γ of M^k (Section III-C); 4) spherical harmonics coefficients $\mathbf{C}^k$ of M^k (Section III-D); 5) axis-aligned bounding box in $\mathtt{W}$; 6) label of ground or non-ground patch. Now we describe how to compute the second, third, and fourth attributes before formulating the CURL-based pose optimisation.

C. Conformal Parametrisation

Conformal mapping is a popular method for texture mapping and mesh deformation in computer graphics [16]. It is adapted here to map a 3D point patch into a 2D rectangular

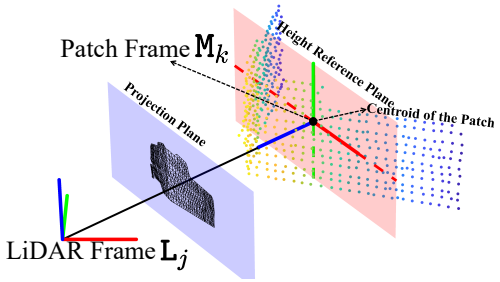


Fig. 4: Coordinates of patch frame $\mathtt{M}_k$, projection plane and height reference plane. Red, green, and blue lines represent the x, y, and z axes, respectively. Later figures maintain this correspondence.

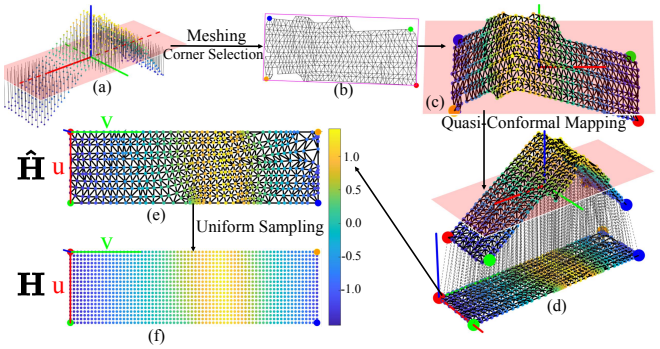


Fig. 5: Conformal parametrisation. Pink plane is the height reference plane in Fig. 4. Colour bar represents heights.

space by a two-step process: 1) surface triangulation of a patch, and 2) quasi-conformal mapping to rectangular space.

1) Surface Triangulation: Triangulation is introduced to model the 3D geometric shape of a patch for quasi-conformal mapping. The centroid of M^k's patch points is defined as the patch frame $\mathtt{M}_k$'s origin.

If M^k is a non-ground patch, its z axis points towards the origin of $\mathtt{L}_j$ and its y axis aligns with the z axis of $\mathtt{L}_j$. For efficiency, we follow the meshing idea proposed in [1] to perform 3D meshing on a 2D plane. Specifically, the patch points are projected to the projection plane (the blue plane in Fig. 4) that is perpendicular to $\mathtt{M}_k$'s z at the unit distance to $\mathtt{L}_j$, and are then used to perform Delaunay triangulation. The 2D mesh is then mapped to 3D.

If M^k is a ground patch, its normal direction is defined as the z axis, with its projection plane intersecting with its centroid.

2) Height-Augmented Quasi-Conformal Mapping: Quasi-conformal mapping [16] is employed to deform the 3D mesh into a 2D rectangular space. Since mesh shapes are normally irregular, mesh's four corner vertices are selected as the corners of the 2D rectangle for quasi-conformal mapping, as illustrated in Fig. 5(b). Now a bijection relationship is established between the patch's 3D points and the conformal map. Therefore, the quasi-conformal mapping function $\Gamma(\cdot)$ can be formulated as

$$\Gamma\left(\mathsf{P}_{xy}([x,y,z]^T)\right) = \begin{bmatrix}\Gamma_x \\ \Gamma_y\end{bmatrix} = \begin{bmatrix}u \\ v\end{bmatrix} \doteq \boldsymbol{\mu} \tag{1}$$

where $[x,y,z]^T$ is the location of a mesh vertex in $\mathtt{M}_k$, $\mathsf{P}_{xy}([x,y,z]^T) = [x,y]^T$, and $[u,v]^T$ is the coordinates in the 2D conformal map. Fig. 5(d) presents a conformal mapping example. Refer to [16] for more details on quasi-conformal mapping.

Moreover, the conformal map is augmented with height information by computing the distances of the mesh's vertices to its height reference plane which is defined as the x-y plane of $\mathtt{M}_k$. An example of the height-augmented conformal map $\hat{\mathbf{H}}$ is shown in Fig. 5(e). However, $\hat{\mathbf{H}}$ tends to be sparse and unevenly distributed with biased height distribution. To increase the accuracy of spherical harmonics encoding (Section III-D), the natural neighbour interpolation [17] method is utilised to generate a uniformly upsampled conformal map $\mathbf{H}$ (Fig. 5(f)).

D. Spherical Harmonics Encoding

The height-augmented conformal map $\mathbf{H}$ are encoded as spherical harmonics which is a series of spherical functions defined on the surface of a sphere. Briefly, the spherical harmonics expansion function with the maximum degree L is defined as

$$f(\theta,\phi) \approx \sum_{l=0}^{L}\sum_{m=-l}^{l} c_{l,m}Y_{l,m}(\theta,\phi), \ (\theta\in[0,\pi], \phi\in[0,2\pi]) \tag{2}$$

where $f(\theta,\phi)$ in spherical coordinate (θ,ϕ) is represented by a linear combination of the spherical harmonics function $Y_{l,m}(\theta,\phi)$ with degree l and order m, defined as

$$Y_{l,m}(\theta,\phi) = (-1)^m\sqrt{\frac{(2l+1)}{4\pi}\frac{(l-m)!}{(l+m)!}}P_{l,m}(cos\theta)e^{im\phi} \tag{3}$$

where $P_{l,m}(x)$ is the associated Legendre polynomials [18]. $\mathbf{C}^k = \{c^k_{0,0}, c^k_{1,-1}, \ldots, c^k_{l,m}, \ldots, c^k_{L,L}\}$ is the spherical harmonics coefficients of patch M^k.

In order to encode $\mathbf{H}$ through spherical harmonics, we need to first establish a bijective map between $[u,v]^T \in \mathbf{H}$ and $[\theta,\phi]^T$. Since the north and south poles of $f(\theta,\phi)$ can only have a single value and do not preserve the bijective property, the following mapping is established

$$\begin{aligned}\theta(u) &= \frac{u}{H}\cdot\pi\cdot\eta + \frac{\pi}{2}\cdot(1-\eta) \\ \phi(v) &= \frac{v}{W}\cdot 2\pi\cdot\eta + \pi\cdot(1-\eta)\end{aligned} \tag{4}$$

where W and H are the width and height of $\mathbf{H}$ respectively, and η is a scaling factor (fixed as 0.8 in our experiments). Iterative Residual Fitting method [19] is used to compute the spherical harmonics coefficients for balanced accuracy and efficiency. With the spherical harmonics coefficients, continuous patch reconstructions can be achieved by reconstructing height-augmented conformal maps with a width resolution ω. Fig. 6 shows an example of continuously reconstructing the patch in Fig. 5 using 10-degree spherical harmonics and $\omega = 100$. Refer to [1] for more details about spherical harmonics encoding and CURL.

(a) Height-augmented conformal map (b) 3D mesh

Fig. 6: Continuous reconstruction of the height-augmented conformal map in Fig. 5(f) and 3D mesh in Fig. 5(c).

After finishing the spherical harmonics encoding of a patch, its CURL representation is inserted into the map $\mathbb{M}$. This enables us to continuously reconstruct and update the map through CURL. Now we discuss how to use the CURL map for pose estimation.

E. Pose Optimisation

The current LiDAR pose relative to the previous scan is estimated by minimising the errors between the projected heights of the LiDAR scan patches and the reconstructed heights of their corresponding map patches. Therefore, we formulate the optimisation problem using the functions of the quasi-conformal mapping and the spherical harmonics.

Suppose there are K patches in $\mathbb{M}_j$ associated with the map and ${}_{\mathtt{L}_j}\mathbf{p}_i \in {}_{\mathtt{L}_j}\mathbf{P}_{M^k}$ denotes the ith point in patch M^k. The objective function is formulated as

$$e(\boldsymbol{\xi}) = \sum_{k=1}^{K} \sum_{i} \left|\left| \mathsf{P}_z \left(\mathbf{T}_{\mathtt{WM}_k}^{-1} \mathbf{T}_{\mathtt{WL}_{j-1}} \mathsf{Exp}(\boldsymbol{\xi}) \, {}_{\mathtt{L}_j}\mathbf{p}_i \right) - \mathbf{I}^k(\boldsymbol{\mu}) \right|\right|^2 \quad (5)$$

where $\boldsymbol{\xi} \in \mathfrak{se}(3)$ is the Lie algebra of the 6-DoF relative pose from $\mathtt{L}_j$ to $\mathtt{L}_{j-1}$, $\mathsf{Exp}(\cdot)$ is the exponential map, $\mathsf{P}_z([x, y, z]^T) = z$, and $\mathbf{I}^k(\boldsymbol{\mu})$ is the height of the corresponding point in the conformal map that is reconstructed using the spherical harmonics:

$$\mathbf{I}^k(\boldsymbol{\mu}) = \sum_{l=0}^{L} \sum_{m=-l}^{l} c_{l,m}^k Y_{l,m} \left(\theta(u), \phi(v)\right) \quad (6)$$

where $\boldsymbol{\mu} = \Gamma \left(\mathsf{P}_{xy} (\mathbf{T}_{\mathtt{WM}_k}^{-1} \mathbf{T}_{\mathtt{WL}_{j-1}} \mathsf{Exp}(\boldsymbol{\xi}) \, {}_{\mathtt{L}}\mathbf{p}_i \right)$. Note $\mathbf{T}_{\mathtt{WM}_k}$ can be derived from the CURL's attributes defined in Section III-B. Hence, (5) computes the sum of errors between the projected heights of the patches in the current LiDAR scan j and the heights of the reconstructed map patches. Intuitively, referring to Fig. 7, if the projected heights were considered as pixel intensities, this was similar to minimising photometric errors in visual odometry and SLAM systems [20], [21], [22]. Since $e(\boldsymbol{\xi})$ is non-linear with respect to $\boldsymbol{\xi}$, its Jacobian matrix is needed. For the sake of simplicity, the Jacobian matrix of the residual $r_{k,i}(\boldsymbol{\xi})$ inside $||\cdot||^2$ of (5) is

$$\mathbf{J} = \frac{\partial r_{k,i}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \nabla \mathsf{P}_z \nabla \mathsf{Q} - \nabla \mathbf{I}^k \nabla \Gamma \nabla \mathsf{P}_{xy} \nabla \mathsf{Q} \quad (7)$$

where $\mathsf{Q} = \mathbf{T}_{\mathtt{WM}_k}^{-1} \mathbf{T}_{\mathtt{WL}_{j-1}} \mathsf{Exp}(\boldsymbol{\xi}) {}_{\mathtt{L}}\mathbf{p}_i$. More details of the terms are given in the Appendix. The Levenberg-Marquardt method is used to solve the optimisation problem.

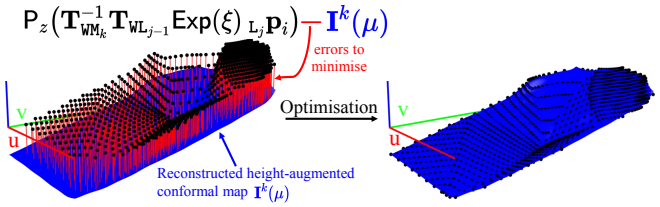


Fig. 7: Pose optimisation on errors between projected heights and reconstructed height-augmented conformal map.

F. Correction of Spherical Harmonics Coefficients

After the pose optimisation, we update the spherical harmonics coefficients of the map patches based on the optimal pose $\boldsymbol{\xi}^*$. According to [1], (2) can be re-written in a vector fashion $\mathbf{F} = \mathbf{YC}$. Hence, we can recover the projected heights of a patch using its existing spherical harmonics coefficients $\mathbf{C}$. Meanwhile, by using $\boldsymbol{\xi}^*$ we can obtain the projected heights $\mathbf{P}_z^*$ by vectorising $\mathsf{P}_z(\cdot)$ of all the LiDAR scan points in the patch. Then, the residuals between them

$$\mathbf{r} = \mathbf{P}_z^* - \mathbf{YC} \quad (8)$$

Therefore, the updated spherical harmonics coefficients $\mathbf{C}^*$ can be computed using

$$\mathbf{C}^* = \mathbf{C} + (\mathbf{Y}^T\mathbf{Y})^{-1}\mathbf{Y}^T\mathbf{r} \quad (9)$$

IV. Experiments

We evaluate the proposed CURL-MAP on both mapping and odometry performance. Since the main focus of CURL-MAP is continuous mapping rather than odometry, the evaluation lays special emphasis on the mapping results. For the experiments, voxel edge length for ground and non-ground patches generation (Section III-A) are $5m$ and $3m$ respectively, the default degree of the spherical harmonics for ground patches is set 2, while for non-ground patches it is 10. This is because ground patches usually have less geometric variation to capture. The IoU threshold β to insert a new patch is 0.1. The spherical harmonics coefficients of a patch are updated by using accumulated residuals from every twenty successful data associations. All experiments are performed on a laptop with an Intel i7-10875H CPU.

A. Evaluation on Map Reconstruction

The Newer College dataset [23] is used for map evaluation because of its survey-level ground truth maps from a Leica BLK360. We report the mapping results on the 3.5 quad loops from the *long experiment* sequence considering its salient building structures to evaluate map accuracy. For continuous reconstruction of a map in CURL-MAP, its density can be adaptive by simply adjusting ω while keeping the identical CURL map.

Fig. 8 shows the errors of our maps reconstructed in three resolutions and the map built by A-LOAM [24] which is an advanced implementation of the popular LiDAR odometry and mapping algorithm LOAM [2]. The error colour represents the distance between the map point to the nearest point in the ground truth map. We can see the maps built

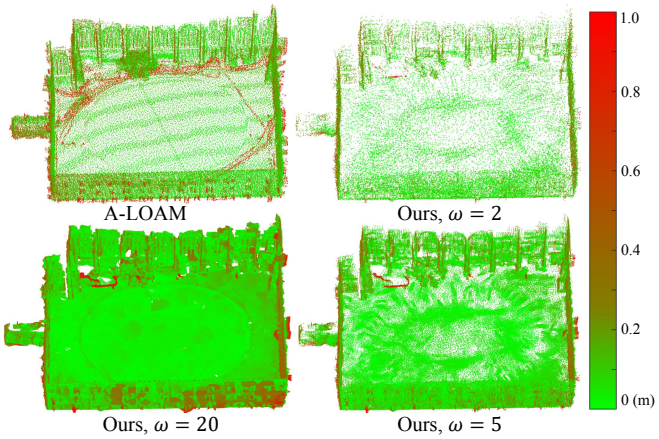
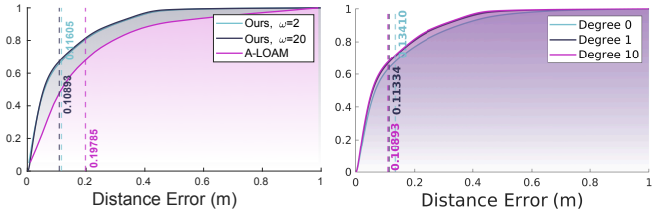


Fig. 8: Errors of reconstructed maps compared with the ground truth map. Colour bar indicates point errors in meter.



(a) (b)

Fig. 9: (a) Cumulative distribution of errors of reconstructed maps. (b) Influence of spherical harmonics degree on reconstructed map accuracy with $\omega = 20$.

by the proposed CURL-MAP are all accurate and have slightly better accuracy than A-LOAM. This matches with the results in Fig. 9a showing that our method achieves a higher percentage of accurately mapped points across different map densities. Importantly, CURL-MAP is capable of continuously increasing the map resolution and density, which provides a considerable amount of extra points to capture geometric details. For instance, the garden edge (ellipse shadow in the quad centre) is visible in our continuous reconstruction with $\omega = 20$, but not in other maps. Note the identical CURL map is used to reconstruct these three maps with different resolutions, which means its size is fixed.

Meanwhile, since the patches in our method store the number of times of being successfully paired during data association, outlier patches, for example, from dynamic objects can be naturally removed by rejecting patches with a low number of successful associations. As shown in our reconstructed maps in Fig. 8, the dynamic ghost shadows are largely removed compared with A-LOAM.

Fig. 9b illustrates the errors of the reconstructed map in relation to the degree of the spherical harmonics. There is a marked improvement in precision when moving from degree 0 to degree 1. However, the enhancement in precision from degree 1 to degree 10 is less pronounced. This could be attributed to the structural nature of the environment, where a few degrees of spherical harmonics can result in a reasonably accurate reconstruction.

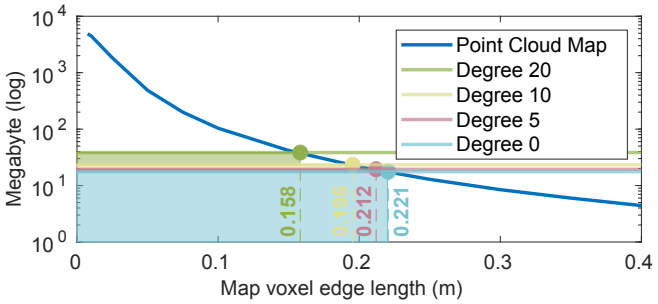


Fig. 10: Map storage against different voxel sizes. Since CURL-MAP is capable of doing continuous reconstruction, its storage is fixed for a defined degree.

B. Evaluation on Map Size

Given that our CURL-MAP method only requires the CURL map stored with the attributes (defined in Section III-B) for continuous reconstruction, its storage size is fixed for different resolutions. Fig. 10 shows map storage space required against different voxel sizes, evaluated using the Newer College dataset. For the point cloud map of A-LOAM, its storage space is obtained by setting various voxel sizes. Both the point cloud map and the CURL map are saved in ASCII format with single-precision floating-point format. The figure shows that the storage size of the point cloud map surges at an exponential rate as the voxel edge length decreases. In contrast, the size of our map remains constant for a given degree. The size of the CURL map, unfortunately, exhibits a quadratic increase corresponding to the degree of the spherical harmonics coefficients. However, we find in most cases degree 10 to 20 is able to achieve very good map accuracy and density.

C. Evaluation on Odometry Estimation and Runtime

The KITTI dataset [25] is selected for odometry evaluation given its established odometry benchmarking.

Table I presents the odometry evaluation results compared with A-LOAM. We can see that our frame-to-frame odometry estimation outperforms A-LOAM's frame-to-frame result. However, while our frame-to-map results show improvement over its frame-to-frame version, they are slightly less accurate than the frame-to-map results of A-LOAM. It may be because our odometry optimisation is only one stage, while A-LOAM has two-stage optimisation and some well-studied techniques to boost accuracy. Our approach predominantly utilises a frame-to-map technique.

Fig. 11 shows the estimated trajectories using our method and A-LOAM against the ground truth. Due to the lack of non-ground patches and noisy data association in KITTI 01 highway, our default set of parameters fails to finish it. But with a different $\beta = 0.15$, our method can successfully run it for a reasonable accuracy.

In pursuit of real-time speed for *frame-to-map fast* method, we reduced the average paired patches of a scan to 100 (from

TABLE I: Odometry evaluation on KITTI dataset (F2F: Frame-to-Frame; F2M: Frame-to-Map)

	Sequence											
Approach	00	01	02	03	04	05	06	07	08	09	10	Average
A-LOAM [24] F2F	4.1/1.7	3.9/1.0	7.5/2.5	4.5/2.2	1.7/1.1	4.1/1.7	1.0/0.5	3.1/2.0	4.8/2.0	5.7/1.9	3.6/1,8	4.0/1.7
A-LOAM [24] F2M	0.7/0.3	2.8/0.6	4.9/1.6	1.2/0.7	1.2/0.4	0.6/0.3	0.7/0.4	0.6/0.4	1.2/0.4	1.1/0.4	1.4/0.5	1.5/0.5
CURL-MAP F2F	2.6/1.0	-	3.0/1.1	1.6/1.4	3.0/1.4	1.8/0.8	1.3/0.6	1.5/1.0	2.3/1.2	2.9/1.0	5.7/2.9	2.6/1.2
CURL-MAP F2M Fast	1.7/0.7	-	4.4/1.5	1.4/1.0	2.4/1.0	1.4/0.7	0.9/0.5	0.8/0.5	1.7/0.7	2.6/0.9	3.1/1.2	2.0/0.9
CURL-MAP F2M Slow	1.8/0.7	5.2/1.2*	2.5/0.9	1.4/1.0	2.3/1.0	1.4/0.7	0.9/0.5	0.9/0.6	1.9/0.8	2.2/0.9	3.1/1.1	1.8/0.8

Relative errors averaged over trajectories of 100 to 800 m length: relative translational error in % / relative rotational error in degrees per 100 m. Compared to CURL-MAP F2M Slow ($\beta = 0.1$), F2M Fast ($\beta = 0.05$) uses fewer associations and utilises degree 5 instead of 10 for spherical harmonics. *01 uses a different parameter ($\beta = 0.15$).

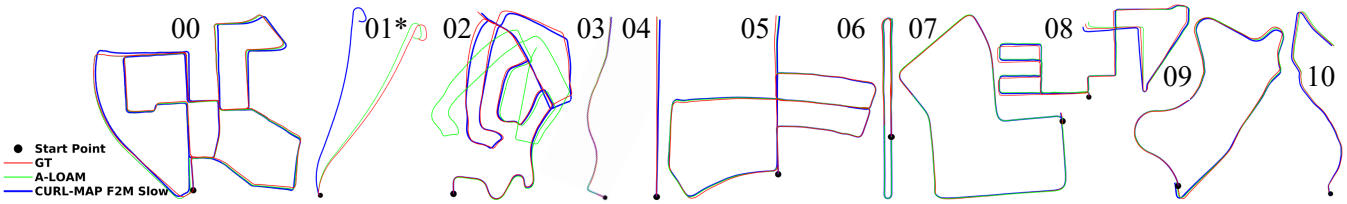


Fig. 11: Estimated trajectories on KITTI dataset.

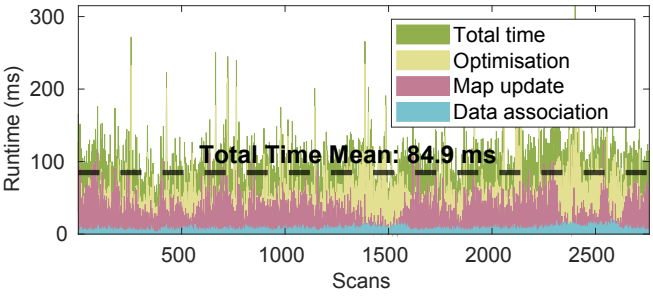


Fig. 12: Runtime evaluation of Frame-to-Map Fast on KITTI 05. Map update includes the total time of the new patch insertion and the spherical harmonics coefficients update.

130), inserted map patches to an average of 6.5 (down from 13), and made the spherical harmonics degree from 10 to 5, though at the expense of accuracy to a certain extent. Detailed runtime analysis is shown in Fig. 12. In this mode, we can still generate a dense map by re-using additional patch points once the odometry is complete.

V. Conclusions

This paper has presented CURL-MAP, a LiDAR mapping and positioning algorithm based on the CURL representation. It is capable of continuously reconstructing 3D maps with varying densities and estimating poses without using ICP. Experiments on two public LiDAR datasets show that CURL-MAP achieves more accurate and denser 3D mapping while using less storage space. Its odometry performance is also comparable with the state-of-the-art LiDAR system.

Our future work will focus on improving the efficiency and robustness of CURL-MAP's odometry and extending it to a full SLAM system. We also see potential in adaptively selecting the degree of spherical harmonics coefficients for enhancing reconstruction accuracy and storage efficiency.

Appendix

This appendix provides the terms in (7) that were not given in the text.

$$\nabla \mathsf{P}_z = [0 \quad 0 \quad 1] \tag{10}$$

$$\nabla \mathsf{Q} = \begin{bmatrix} \mathbf{\Lambda} & -\mathbf{\Lambda}[{}_{\mathtt{L}}\mathbf{p}_i]_\times \end{bmatrix} \tag{11}$$

where $\mathbf{\Lambda} = \mathbf{R}_{\mathtt{M}_k\mathtt{W}}\mathbf{R}_{\mathtt{WL}_{j-1}}\mathsf{Exp}(\boldsymbol{\theta})$, $\boldsymbol{\xi} = [\boldsymbol{\rho}, \boldsymbol{\theta}]^T \in \mathbb{R}^6$ and $[\cdot]_\times$ is the skew-symmetric matrix of the vector.

$$\nabla \mathbf{I}^k = \begin{bmatrix} \nabla_x \mathbf{I}^k & \nabla_y \mathbf{I}^k \end{bmatrix} \tag{12}$$

where

$$\nabla_x \mathbf{I}^k \approx \frac{\mathbf{I}^k([u+\delta, v]^T) - \mathbf{I}^k([u-\delta, v]^T)}{2\delta}$$

$$\nabla_y \mathbf{I}^k \approx \frac{\mathbf{I}^k([u, v+\delta]^T) - \mathbf{I}^k([u, v-\delta]^T)}{2\delta}$$

and δ is a small constant.

$$\nabla \Gamma = \begin{bmatrix} \nabla_x \Gamma_x & \nabla_y \Gamma_x \\ \nabla_x \Gamma_y & \nabla_y \Gamma_y \end{bmatrix} \tag{13}$$

we use numerical derivatives where

$$\nabla_x \Gamma_x \approx \frac{\Gamma_x([x+\delta, y]^T) - \Gamma_x([x-\delta, y]^T)}{2\delta}$$

$$\nabla_y \Gamma_x \approx \frac{\Gamma_x([x, y+\delta]^T) - \Gamma_x([x, y-\delta]^T)}{2\delta}$$

$$\nabla_x \Gamma_y \approx \frac{\Gamma_y([x+\delta, y]^T) - \Gamma_y([x-\delta, y]^T)}{2\delta}$$

$$\nabla_y \Gamma_y \approx \frac{\Gamma_y([x, y+\delta]^T) - \Gamma_y([x, y-\delta]^T)}{2\delta}$$

$$\nabla \mathsf{P}_{xy} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{14}$$

REFERENCES

[1] K. Zhang, Z. Hong, S. Xu, and S. Wang, "CURL: Continuous, Ultra-compact Representation for LiDAR," in *Robotics: Science and Systems*, 2022.

[2] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Robotics: Science and Systems*, 2014.

[3] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping," in *IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2020.

[4] J. Behley and C. Stachniss, "Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments," in *Robotics: Science and Systems*, 2018.

[5] C. Park, P. Moghadam, J. L. Williams, S. Kim, S. Sridharan, and C. Fookes, "Elasticity Meets Continuous-Time: Map-Centric Dense 3D LiDAR SLAM," *IEEE Transactions on Robotics*, 2021.

[6] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2018.

[7] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast Direct LiDAR-Inertial Odometry," *IEEE Transactions on Robotics*, 2022.

[8] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, "ElasticFusion: Dense SLAM Without A Pose Graph," in *Robotics: Science and Systems*, 2015.

[9] C. Park, P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan, "Elastic LiDAR Fusion: Dense Map-Centric Continuous-Time SLAM," in *IEEE International Conference on Robotics and Automation*. IEEE, 2018.

[10] C. Xia, C. Xu, P. Rim, M. Ding, N. Zheng, K. Keutzer, M. Tomizuka, and W. Zhan, "Quadric Representations for LiDAR Odometry, Mapping and Localization," *arXiv preprint arXiv:2304.14190*, 2023.

[11] C. Yuan, W. Xu, X. Liu, X. Hong, and F. Zhang, "Efficient and Probabilistic Adaptive Voxel Mapping for Accurate Online LiDAR Odometry," *IEEE Robotics and Automation Letters*, 2022.

[12] I. Vizzo, X. Chen, N. Chebrolu, J. Behley, and C. Stachniss, "Poisson Surface Reconstruction for LiDAR Odometry and Mapping," in *IEEE International Conference on Robotics and Automation*. IEEE, 2021.

[13] J. Ruan, B. Li, Y. Wang, and Y. Sun, "SLAMesh: Real-time LiDAR Simultaneous Localization and Meshing," *arXiv preprint arXiv:2303.05252*, 2023.

[14] R. Dube, 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," *The International Journal of Robotics Research*, 2020.

[15] S. Lee, H. Lim, and H. Myung, "Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3D point cloud," in *IEEE Robotics and Automation Letters*, 2022.

[16] T. W. Meng, G. P.-T. Choi, and L. M. Lui, "TEMPO: Feature-Endowed Teichm¨uller Extremal Mappings of Point Clouds," *SIAM Journal on Imaging Sciences*, 2016.

[17] The CGAL Project, *CGAL User and Reference Manual*, 5.6 ed. CGAL Editorial Board, 2023. [Online]. Available: https://doc.cgal.org/5.6/Manual/packages.html

[18] C. Müller, *Spherical harmonics*. Springer, 2006.

[19] L. Shen and M. K. Chung, "Large-Scale Modeling of Parametric Surfaces Using Spherical Harmonics," in *Third International Symposium on 3D Data Processing, Visualization, and Transmission*. IEEE, 2006.

[20] C. Kerl, J. Sturm, and D. Cremers, "Robust Odometry Estimation for RGB-D Cameras," in *IEEE International Conference on Robotics and Automation*. IEEE, 2013.

[21] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast Semi-Direct Monocular Visual Odometry," in *IEEE International Conference on Robotics and Automation*. IEEE, 2014.

[22] D. Luo, Y. Zhuang, and S. Wang, "Hybrid sparse monocular visual odometry with online photometric calibration," *The International Journal of Robotics Research*, 2022.

[23] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2020.

[24] T. Qin and S. Cao. Advanced implementation of LOAM. [Online]. Available: https://github.com/HKUST-Aerial-Robotics/A-LOAM

[25] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *The International Journal of Robotics Research*, 2013.