Fast Sparse LiDAR Odometry Using Self-Supervised Feature Selection on Intensity Images

Tiziano Guadagnino Xieyuanli Chen Matteo Sodano Jens Behley Giorgio Grisetti Cyrill Stachniss

***Abstract*—Ego-motion estimation is a fundamental building block of any autonomous system that needs to navigate in an environment. In large-scale outdoor scenes, 3D LiDARs are often used for this task, as they provide a large number of range measurements at high precision. In this paper, we propose a novel approach that exploits the intensity channel of 3D LiDAR scans to compute an accurate odometry estimate at a high frequency. In contrast to existing methods that operate on full point clouds, our approach extracts a sparse set of salient points from intensity images using data-driven feature extraction architectures originally designed for RGB images. These salient points are then used to compute the relative pose between successive scans. Furthermore, we propose a novel self-supervised procedure to fine-tune the feature extraction network online during navigation, which exploits the estimated relative motion but does not require ground truth data. The experimental evaluation suggests that the proposed approach provides a solid ego-motion estimation at a much higher frequency than the sensor frame rate while improving its estimation accuracy online.**

***Index Terms*—SLAM, Vision-Based Navigation**

I. Introduction

THE ability to accurately estimate the ego-motion of an autonomous vehicle is essential in many areas of mobile robotics. In the context of autonomous driving and outdoor robotics, 3D LiDARs are often employed for this task, as they provide accurate range measurements of the surrounding environment without being affected by illumination. This property allows estimating the poses even in low-light conditions, such as night, where cameras often do not provide sufficient information. In the robotics community, many algorithms have been proposed to compute odometry using 3D LiDAR point clouds [2][7][35], most of which rely on the Iterative Closest Point (ICP) algorithm [3] and variants of it [4][24]. To provide robust and accurate results, these algorithms typically exploit the local geometry around each point of the cloud. These features come in the form of normals [4][25], smoothness of the local surface [35], or a probabilistic distribution of neighboring points [20][24].

Manuscript received: February, 23, 2022; Revised May, 16, 2022; Accepted June, 6, 2022.

This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments.

This work has partially been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070 – 390732324 – PhenoRob.

T. Guadagnino and G. Grisetti are with La Sapienza University of Rome, Italy. X. Chen, M. Sodano, J. Behley, and C. Stachniss are with the University of Bonn, Germany. C. Stachniss is additionally with the Department of Engineering Science at the University of Oxford, UK.

Digital Object Identifier (DOI): see top of this page.

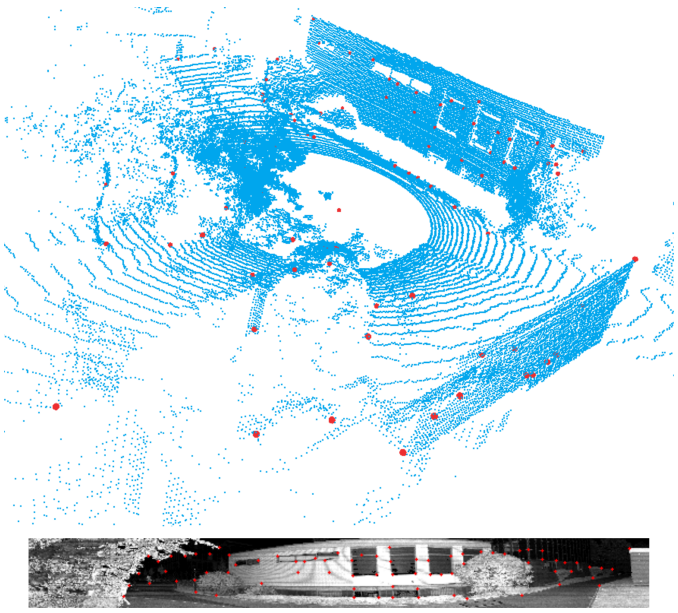

Fig. 1: An example of salient points extraction of our odometry pipeline. We first project the intensity channel of the LiDAR scans into a cylindrical image (bottom) and then extract keypoints using SuperPoint (red dots). We can directly identify the corresponding 3D salient points on the point cloud (top). We then use these points in our odometry pipeline to provide state-of-the-art pose estimation at high frequency.

The main drawback of these algorithms is that they process all points in the scans to estimate the relative pose. The task is particularly challenging on modern 3D LiDARs, as they typically provide a large amount of data (around one million points per second). Furthermore, the data is redundant for the motion estimation problem as, under ideal conditions, three points are sufficient to determine the rigid transformation between two point clouds. Even though these conditions are usually not met in real-world scenarios, it is evident that most of the computations in LiDAR odometry systems might be used on little informative points of the scans.

In this work, we investigate the possibility of exploiting the intensity channel of the LiDAR scans to extract salient points in the scene and then use them to compute the odometry more efficiently. Our subset of the point cloud is small enough to allow fast processing and, at the same time, contains enough information to perform the task at an accuracy on par or better than standard point cloud registration algorithms.

The main contribution of this paper is a sparse LiDAR odometry pipeline that relies only on intensity images for feature selection. Moreover, we do not use a LiDAR-based detector for identifying these points but instead rely on the SuperPoint [9] architecture, a well-known feature extractor network for RGB images. To boost the odometry beyond the SuperPoint performance, we propose a self-supervised procedure for fine-tuning the architecture online to the scene at hand while computing the relative pose. This procedure requires neither labels nor ground truth poses. It only relies on the relative poses estimated by the odometry pipeline itself. The fine-tuning allows the system to adapt the feature selection process to the structure of the scene in which it is operating, improving the pose estimation accuracy.

In sum, we make three key claims: our approach is able to (i) compute solid frame-to-frame odometry relying solely on intensity images for feature selection, (ii) improve its odometry performance on the fly by fine-tuning the salient point detector during navigation, and (iii) it runs online at 30 frame-per-second and even faster. These claims are backed up by our experimental evaluation.

II. Related Work

3D laser-based odometry is a widely investigated topic in robotics. It builds upon point cloud registration methods to incrementally estimate the pose of a mobile platform by aligning successive scans.

The Iterative Closest Point algorithm (ICP) [3] is one of the most used methods for point cloud registration. It is an iterative algorithm that refines an initial relative pose estimate. At each iteration, the algorithm searches for corresponding points between the clouds. Then, it performs an optimization step to update the pose by minimizing the point-to-point distance of these correspondences.

Chen et al. [4] modifies the objective of the original ICP algorithm to capture the knowledge that points collected by range sensors are discrete samples of continuous surfaces. This more robust cost function, named point-to-plane, improves the ICP convergence speed and accuracy. Segal et al. [24] developed a probabilistic formulation of ICP called Generalized-ICP (G-ICP). In this framework, the author introduces a novel objective considering the point distributions in both scans. This further improves estimation accuracy and is considered the gold standard for point cloud registration.

Della Corte et al. [7] proposes a general methodology for 3D photometric registration that can be used on both RGB-D and LiDAR data. The approach does not require explicit data association and can exploit multiple information sources, such as range, depth, color and normals.

All the above mentioned approaches operate on all points to estimate the relative transformation between the point clouds. They are highly time-consuming and cannot be employed online with modern LiDARs, as they provide a massive amount of points per frame. The research community proposes different strategies to reduce the 3D point cloud data while maintaining the pose estimate accuracy to overcome this limitation. These approaches rely on subsampling [19][32], NDT-representations [29][23][6], or distinctive features [35][22][18] to extract few meaningful points that are then used for the registration. Our approach belongs to the latter category, as it extracts salient features from the intensity channel of the scans. While also previous works [35][18] extract distinct points from the scans, it does so by evaluating the smoothness of the local surface around each point. The selected points correspond to planar patches and corners, which is substantially different to our approach that instead select salient features using only intensity values.

Recently, researchers have developed many data-driven algorithms to extract salient points and descriptors directly from point clouds. Choy et al. [5] presents a fully convolutional feature network for extracting dense descriptors on point clouds. They then rely on random sampling to identify the salient points. Building upon this architecture, Bai et al. [1] proposes D3Feat, which computes a dense map for interest points and descriptors simultaneously. Deng et al. [8] proposes 3DFeat-Net, a two-stage network that first computes the keypoints and then calculates the descriptors for the selected points only. Shi et al. [26] proposes a graph neural network to extract and match keypoints between two point clouds.

All these methods require ground truth poses for training. Moreover, to the best of our knowledge, none of the above data-driven approaches were used to compute the odometry online, but rather an offline registration between the point clouds. In contrast to these approaches, our method does not require ground truth data and can compute the odometry at the sensor frame rate.

Similarly to our method, the approach of Yoon et al. [34] does not require ground truth and it is trained using only the LiDAR scans. The authors present an unsupervised learning algorithm that combines odometry estimation with keypoints prediction and uncertainty estimation using a single learning objective. However, the algorithm cannot compute the odometry at sensor frame rate.

Wang et al. [33] proposes a SLAM pipeline that combines geometric and intensity information to enhance estimation accuracy. The approach exploits the intensity as an additional cue in the feature selection step and the map representation. Nevertheless, the system heavily relies on geometric information to detect distinctive points from the LiDAR scans. Conversely, our work aims to show that salient points can be extracted by relying solely on intensity values.

Di Giammarino et al. [10] shows that the intensity channel of LiDAR scans can be effectively used to detect loop closures using traditional visual place recognition algorithms. In contrast, our work focus on exploiting the intensity images to compute an accurate frame-to-frame odometry.

A method related to our approach is the work by Streiff et al. [30], where the authors train a convolutional neural network to extract salient points and descriptors using an image representation of the scans. Their method exploits both the range and the point coordinates, and it is trained from scratch using ground truth trajectories. In contrast, our method relies on a pre-trained network on RGB images, exploits only intensity information, and can be trained online using the estimated relative pose in a self-supervised fashion. Thus, no external

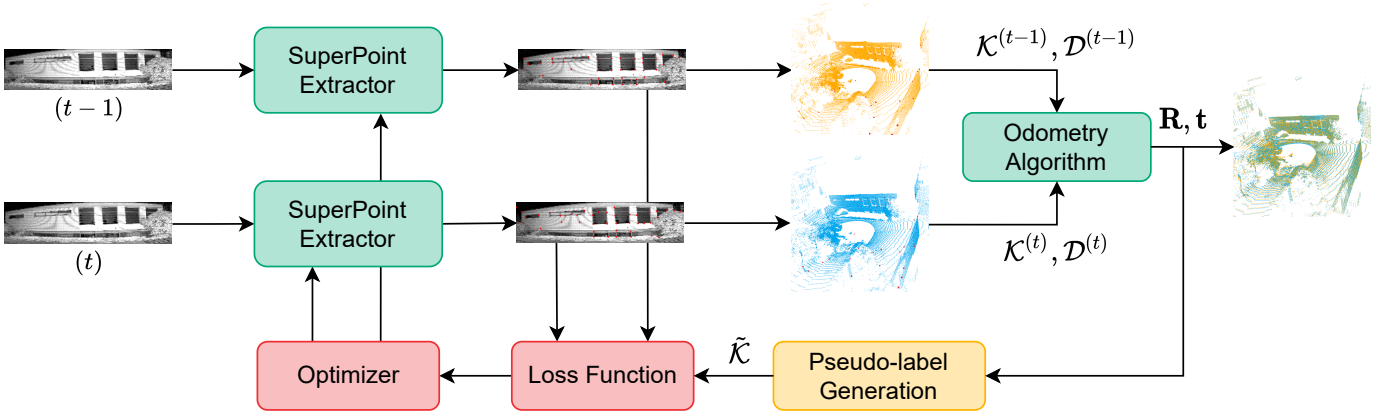

Fig. 2: Our approach in a nutshell. The SuperPoint architecture processes intensity images in successive frames to obtain salient point locations in image coordinates. We then extract the corresponding 3D points from the scan. These keypoints $\mathcal{K}$, together with the corresponding descriptors $\mathcal{D}$, are then passed to our odometry algorithm to estimate the relative pose $\mathbf{R}, \mathbf{t}$ that aligns the scans. We then use this pose to generate pseudo-labels which we exploit to train SuperPoint online.

ground truth is needed in our case, which is a significant advantage of our approach.

III. Feature-based Point Cloud Registration

Point cloud registration methods compute and iteratively refine an initial estimate of the relative pose between two point clouds. During each iteration, these algorithms search for corresponding points between the clouds and perform an optimization step based on the correspondences to minimize the distance between associated points. We can roughly divide these algorithms into two categories based on the data-association strategy to find corresponding points. Geometric approaches rely on simple heuristics based on nearest neighbor search [3][4][24][25]. They are effective only when a good initial estimate of the relative pose is available.

Feature-based methods in contrast compute a descriptor vector that incorporates the surrounding geometry and appearance of each point in the clouds [22][28]. As these descriptors are informative enough to identify corresponding points, these approaches usually rely on a RANSAC scheme [11] combined with the Umeyama algorithm [31] to compute a robust initial alignment between the point clouds. As such, these methods do not require an externally provided initial estimate. However, the descriptor computation and RANSAC runtime limit the real-time application of this type of algorithm, especially on dense point clouds.

Our method relies on feature-based registration components and builds on top of them, but we do not claim a novelty for that part.

IV. Our Approach

Our approach aims to compute an accurate frame-to-frame odometry using sparse keypoint sets extracted from the intensity channel of 3D LiDAR scans. At the same time, we exploit the odometry estimate to fine-tune the interest point detector in a self-supervised fashion. We provide an overview of the approach in Fig. 2. To identify the interest points, we project the LiDAR point cloud into an image where pixels contain both the coordinates and the intensity values of the corresponding points (see Sec. IV-A). Keypoint/descriptor pairs are extracted from the intensity channel of this image using the SuperPoint architecture [9] (see Sec. IV-B). The relative pose is estimated using this sparse point clouds in a feature-based registration fashion (see Sec. IV-C). Note that, to initially compute the odometry, we do not pre-train the network on LiDAR intensity but instead rely on the original model trained on RGB images[1]. Using the information given by the relative poses, we generate pseudo-ground truth salient point locations and back-propagate them through the network (Sec. IV-D). The overall system runs online at 30 frame-per-second.

A. Intensity Image Generation

The key idea of the proposed approach is to use interest points extracted from an image representation of LiDAR scans to compute the odometry of the vehicle. We represent the point cloud as a set of points $\mathcal{P} = \{(\mathbf{p}, i)\}$, where $\mathbf{p} = (x, y, z)$ are the spatial coordinates and i is the corresponding intensity value. We project the point cloud into a cylindrical image $\mathcal{I} \in \mathbb{R}^{H \times W \times 4}$, where each pixel contains the nearest point $(\mathbf{p}, i)$ projected into that pixel. More specifically, we convert $\mathbf{p}$ to image coordinates (u, v) using the mapping $\Pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$ [2]:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\left(1 - \arctan\left(y, x\right)\pi^{-1}\right) W \\ \left(1 - \left(\arcsin\left(zr^{-1}\right) + f_{up}\right) f^{-1}\right) H \end{pmatrix}, \quad (1)$$

where $r = \sqrt{x^2 + y^2 + z^2}$ is the range, $\mathrm{f} = \mathrm{f}_{\text{up}} + \mathrm{f}_{\text{down}}$ is the vertical field-of-view of the sensor, and W, H are the width and height of the resulting image $\mathcal{I}$, respectively. Given the image $\mathcal{I}$, we can extract the individual channels $\mathcal{I}_x$, $\mathcal{I}_y$, $\mathcal{I}_z$, $\mathcal{I}_i$ by simple slicing operations.

B. Interest Point Detection using SuperPoint

For each generated image $\mathcal{I}$, we extract keypoint-descriptor pairs from the intensity channel $\mathcal{I}_i$ using the SuperPoint architecture. An example of interest point detection is shown

[1]The pre-trained model can be found at https://github.com/magicleap/SuperPointPretrainedNetwork

in Fig. 1. This fully convolutional network has a VGG-like [27] encoder that maps the input image $\mathcal{I}_i \in \mathbb{R}^{H \times W}$ to an intermediate tensor $\mathcal{B} \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8} \times C}$ with smaller spatial dimension and larger depth. After the encoder, the architecture splits into two decoders. One detects the salient point locations, and the other computes point descriptors. The peculiarity of these decoders is that they are both composed of a single Conv-ReLU-Conv block, while the remaining up-sampling operations have no trainable parameters. For more details about the SuperPoint architecture, we refer to the original paper of DeTone et al. [9]. Due to its lightweight structure, this neural network can be trained effectively on small memory GPUs, which makes it suitable for online training.

C. Odometry Algorithm

Our approach relies on the intensity values returned by the LiDAR sensor to compute features and, on top of that, the relative pose between scans $\mathcal{P}^{(t)}$ and $\mathcal{P}^{(t-1)}$. The algorithm is a feature-based point cloud registration method as presented in Sec. III, applied to sparse point sets extracted from intensity images. We first extract the 3D points coordinates of the keypoints $\mathcal{K}^{(t)}$, $\mathcal{K}^{(t-1)}$ from the images $\mathcal{I}^{(t)}$ and $\mathcal{I}^{(t-1)}$ using the locations given by SuperPoint. Further, we obtain the corresponding descriptors from the second decoder of the network. As the number of extracted salient points per frame is small, we perform an exhaustive search among the descriptors sets $\mathcal{D}^{(t)}$, $\mathcal{D}^{(t-1)}$ to establish the first set of correspondences. As we compute the descriptors using only the intensity channels, it might happen that points with matching descriptors do not correspond to the same spatial location.

To filter out these outliers, we rely on RANSAC [11] as classical feature-based registration methods do. In particular, at each iteration, we sample three random correspondences and compute the relative transformation between the two frames using the Umeyama algorithm [31]. We consider a point correspondence an inlier if the associated points are closer than a threshold τ. We then select the transformation with the largest number of inliers.

Once inlier correspondences are determined, we refine the relative pose by minimizing the point-to-point error using non-linear ICP:

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum_{j,k \in \mathcal{C}} \rho(\|\mathbf{p}_k^{(t-1)} - \mathbf{R}\,\mathbf{p}_j^{(t)} + \mathbf{t}\|^2), \qquad (2)$$

where $\mathbf{R}, \mathbf{t}$ are the relative rotation and translation, $\mathcal{C}$ is the set of inlier correspondences. $\mathbf{p}_j^{(t)}, \mathbf{p}_k^{(t-1)}$ are the coordinates of the corresponding points and ρ is the Geman-McClure robust cost [13].

D. Self-Supervision Optimization of the Features

Given the relative pose estimated by the odometry algorithm, we design a procedure for fine-tuning the interest point detector in a self-supervised way so that the odometry estimation improves. To this end, we first identify the subset of keypoints that constitute the consensus set of the transformation given by RANSAC and use them as labels for the salient points' locations during the training. If we naively apply the above procedure, the network will constantly reduce the number of points extracted, as the inliers are just a subset of the keypoints set.

Instead, we want to identify new salient points to keep a reasonable size of the consensus set to enhance the robustness and, potentially, the pose estimate accuracy. To this end, we select points with a small residual that are not in $\mathcal{K}^{(t)}$, using the same threshold τ that we used in RANSAC. As in this case, we do not have correspondences between points, we projectively compute the residuals using the image representation of the target cloud $\mathcal{I}^{(t-1)}$ and the projection function Π. A point $p_j^{(t)}$ in the source cloud is first transformed and then converted to image coordinates using:

$$\begin{pmatrix} u_j \\ v_j \end{pmatrix} = \Pi(\mathbf{R}\,\mathbf{p}_j^{(t)} + \mathbf{t}). \qquad (3)$$

We can then evaluate the point-to-point error via:

$$E(\mathbf{p}_j^{(t)}, \mathcal{I}^{(t-1)}, \mathbf{R}, \mathbf{t}) = \|\mathbf{p}_j^{(t-1)} - \mathbf{R}\,\mathbf{p}_j^{(t)} + \mathbf{t}\|^2, \qquad (4)$$

where $p_j^{(t-1)}$ are the point coordinates stored in the pixel $(u_j,\, v_j)$ of $\mathcal{I}^{(t-1)}$. We can efficiently compute the residual for all the points through this method without the explicit need of a nearest neighbor search for point-to-point correspondences.

However, the number of candidate keypoints is directly related to the relative motion of the LiDAR. In the limit case in which the sensor is not moving, all the scan points could be selected. Furthermore, it is not straightforward for the salient point detector to identify these points, as it does have access just to the intensity values, not the point coordinates. We filter out points with low gradient magnitude on the intensity image to tackle these problems. In this way, we sparsify the candidate keypoints set and give a clear hint to the network about the salient points' locations. Overall, a point is selected as a new keypoint if the projective residual is less than τ and the magnitude of the intensity gradient is above a certain threshold γ. Once we identify the new keypoints $\tilde{\mathcal{K}}$ in both LiDAR scans, we generate the corresponding pseudo labels for both the source and target intensity image.

To train the SuperPoint architecture, we use the same loss functions of the original paper [9]. The loss is optimized on a frame-to-frame basis using Stochastic Gradient Descent [15]. Thanks to the lightweight structure of SuperPoint and our efficient strategy to generate pseudo labels, we can perform the fine-tuning of the network while the system is computing the odometry. Moreover, the overall pipeline runs at the sensor frame rate.

Nevertheless, as we will show in the experiments, the pre-trained model on RGB images already provides competitive pose accuracy. As such, we can activate the fine-tuning on-the-fly depending on the specific operating conditions of the system. For instance, in highly dynamic environments, it might be convenient to turn on the self-supervision to push most of the keypoints towards the static parts of the scene.

E. Implementation Details

We report the evaluations for both the pipeline that uses the pre-trained model on RGB images, named Sparse LiDAR

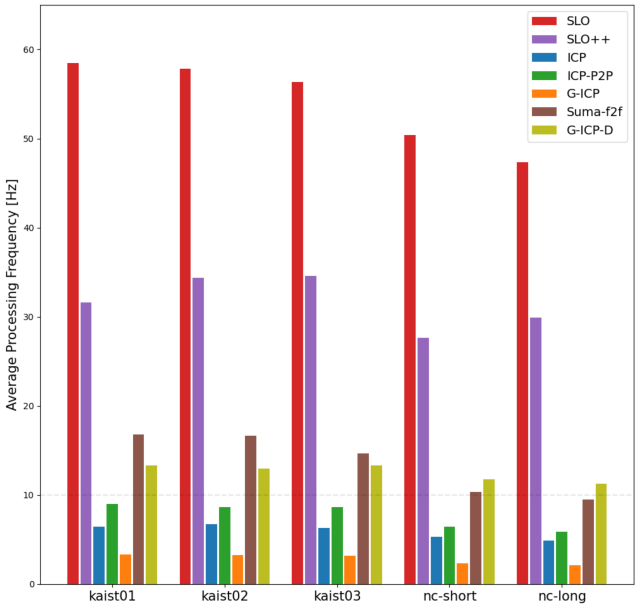

Fig. 3: Average processing frequency for all the approaches on the selected datasets. The dashed line indicates the sensor frame rate.

Odometry (SLO), and the online fine-tuned variant (SLO++). In both cases, τ is set to $0.3\,m$, while in SLO++ γ is set to the mean magnitude of the gradient in the current image. Furthermore, for SLO++ the loss is optimized using Stochastic Gradient Descent with a batch size of 2, meaning that we consider only the pair of frames used to compute the odometry at each step. We would like to point out that we do not perform multiple epochs on the datasets but rather discard the pseudo-labels and the images once we perform an optimization step. As such, the network sees each sample just once.

V. Experimental Evaluation

We present our experiments to showcase the capabilities of our method and to support our key claims that our approach can (i) compute solid frame-to-frame odometry relying solely on intensity images for feature selection, (ii) improve its odometry performance on-the-fly by fine-tuning the salient point detector during navigation, and (iii) it runs online at 30 frame-per-second and even faster.

A. Odometry Baselines

In the following experiments, we compare the proposed method against state-of-the-art algorithms for point cloud registration that are typically used to compute a frame-to-frame odometry using LiDAR clouds. In particular, we select the three standard point cloud registration algorithms namely ICP [3], point-to-plane ICP [4] (ICP-P2P) and G-ICP [24]. Since the latter method is well known to be highly demanding in terms of runtime, we implement an additional baseline, denoted as G-ICP-D, where the input point cloud is down-sampled with a voxel size of $0.2\,m$.

All the approaches mentioned above rely on nearest neighbor search to find correspondences. For outlier rejection, we filter out associations with a point-to-point distance above $2.0\,m$ and use a Geman-McClure robust cost [13] with the scale parameter of 0.5.

We also implement a fifth baseline, reported as Suma-f2f, based on the work of Behley et al. [2]. This baseline computes the associations projectively using Eq. (1) and estimates the relative pose using a point-to-plane ICP with a Huber robust cost [14]. We keep the same outlier rejection strategy and thresholds as the original algorithm [2].

Fig. 4: Comparison between intensity images obtained from Newer College (top), Mulran (middle) and KITTI scans (bottom).

B. Datasets and Metrics

We perform the comparisons using the MulRan [16] and Newer College [21] datasets. Unfortunately, we could not perform the comparisons on the popular KITTI Odometry sequences [12], as the intensity values for the LiDAR used in the recordings were too noisy due to the over ten years old model of the LiDAR. Early multi-beam LiDAR sensors return different values of the intensity even though the beams' rays were hitting the same surface [17]. Descriptors extracted on such noisy images are not reliable enough to perform a high-quality feature matching. To better showcase this effect, we report in Fig. 4 the intensity images obtained by projecting a sample scan from KITTI, MulRan and Newer College datasets.

To evaluate the pose accuracy of the different approaches, we use the metric introduced by the KITTI Odometry benchmark [12], which considers relative translation and rotation errors averaged over different trajectory lengths. Furthermore, we report the average processing time for all the compared algorithms. For all the experiments, we use a Dell XPS 15 laptop with an Intel Core i7-10750H, 16 GB of RAM, and an NVIDIA GeForce GTX 1650 Ti GPU with 4 GB memory. The machine is running Ubuntu 20.04.

C. Runtime

The first experiment has been conducted to back up our claim that our approach runs fast enough to support online processing at sensor frame rate. Furthermore, we want to compare the performance of the different algorithms in terms of runtime. We report in Fig. 3 the average processing frequency of all the baselines for the selected datasets. As we can see, only SLO, SLO++, G-ICP-D and Suma-f2f can effectively compute the odometry at sensor frame rate. In particular, SLO

Approach	kaist01	kaist02	kaist03	short	long
SLO	10.46/3.72	10.25/3.93	11.65/4.04	26.25/23.21	14.15/15.79
SLO++	**10.04**/3.65	**9.86**/3.81	**11.06**/3.92	**24.87**/22.09	**13.61**/15.48
G-ICP-D	13.40/2.78	10.73/2.38	14.30/2.95	28.63/18.48	35.32/24.02
Suma-f2f	24.89/11.68	25.29/13.58	23.79/12.44	43.53/46.75	37.60/39.77
ICP	8.75/2.82	9.99/3.41	9.74/3.15	17.56/13.68	15.21/14.43
ICP-P2P	17.98/6.10	16.98/6.06	18.34/5.98	9.41/8.27	6.52/7.25
G-ICP	**5.78**/1.85	**5.17**/1.71	**5.65**/1.69	**6.58**/5.54	**4.56**/4.21

TABLE I: Relative errors averaged over trajectories of 100 to 800 meters length: relative translational error in % / relative rotational error in degrees per 100 meters. In **blue** the best performance in terms of translational error among the real-time capable methods, in **red** among all the compared approaches.

runs at more than 50 Hz, and it is the fastest method among the compared algorithms.

As we can see from the result, our fine-tuning strategy does not impact the real-time capability of the odometry system. In fact, SLO++ runs at up to 30 Hz, more than two times the average processing frequency of Suma-f2f, which is the closest baseline. Furthermore, it processes scans 3 times faster than a typical LiDAR data stream (10 Hz).

D. Odometry Performance

The experiments presented in this section are designed to show the odometry performance of our approach. The results support the claim that the proposed method can compute accurate odometry using salient points extracted on intensity images only.

We report quantitative results on the kaist sequences of MulRan and short/long sequences of Newer College in Tab. I and Fig. 5. In terms of the translation and rotation error, both SLO and SLO++ perform on par with state-of-the-art methods for frame-to-frame LiDAR odometry. In particular, their performances in terms of odometry accuracy are similar to point-to-point ICP on all the presented datasets. This result shows that the keypoints extracted by SuperPoint on the intensity images correspond to stable 3D points in the scene. They are usually objects' corners and edges, which have well-defined geometric properties in 3D and are easily recognizable in the 2D image.

On the kaist sequences, our approach performs even better than point-to-plane ICP. This is related to the fact that there are not many planar structures in those scenarios, and the algorithm tends to be stuck in local minima.

The best performing method is G-ICP. This algorithm exploits the local geometric structure around each point to bootstrap convergence and accuracy. It does that by computing the local covariance of the points in a small neighborhood. As such, it is the most demanding method in terms of computational resources and runtime. Conversely, G-ICP-D can effectively run at the sensor frame rate. However, the resulting estimated odometry is much less accurate due to the reduced number of points used to estimate the local geometric properties around each point.

The Suma-f2f algorithm has the worst performance in terms of odometry accuracy on all the selected datasets. This is due to the projective data associations, which turns out to be less reliable than nearest neighbor search, especially if used on a frame-to-frame basis.

	kaist0	kaist1	kaist2	nc-short	nc-long
δ_t	4.01%	3.8%	5.1%	5.2%	3.8%

TABLE II: Ratio of improvement δ_t between SLO and SLO++ for all the selected datasets. The value is reported in percentage

E. Online Fine-Tuning

The last experiment evaluates the performance gain of the online fine-tuning. The results support the claim that the proposed self-supervised strategy for fine-tuning the salient point detector improves the accuracy of the pose estimate. To quantify this improvement, we define a straightforward metric that aims to measure the ratio of reduction in translation error as:

$$\delta_t = \frac{e_{\text{SLO}} - e_{\text{SLO++}}}{e_{\text{SLO}}}, \tag{5}$$

where e_{SLO}, $e_{\text{SLO++}}$ represent the relative translation error for SLO and SLO++ according to the KITTI metric. A positive value of Eq. (5) represents a translation error reduction in percentage. We report the value of δ_t for all the selected datasets in Tab. II. As we can see from the results, SLO++ provides a reduction in translation error between 4 and 5% in all cases. This confirms that the system improves over time and learns how to extract better keypoints as it processes incoming scans.

In summary, our evaluation suggests that both SLO and SLO++ provide competitive performances in terms of odometry and can process incoming scans at sensor frame rate. Furthermore, the two algorithms are not mutually exclusive, as we can switch on the self-supervision at any time without impacting the real-time capability of the odometry system. As such, one can compromise between a fast odometry system or sacrifice some processing speed to get a better ego-motion estimation.

VI. Conclusion

In this paper, we presented a novel approach to LiDAR odometry that outperforms key state-of-the-art methods in terms of runtime while being as accurate as standard ICP-based methods, such as point-to-point and point-to-plane ICP. Our approach exploits the intensity channel of LiDAR scans

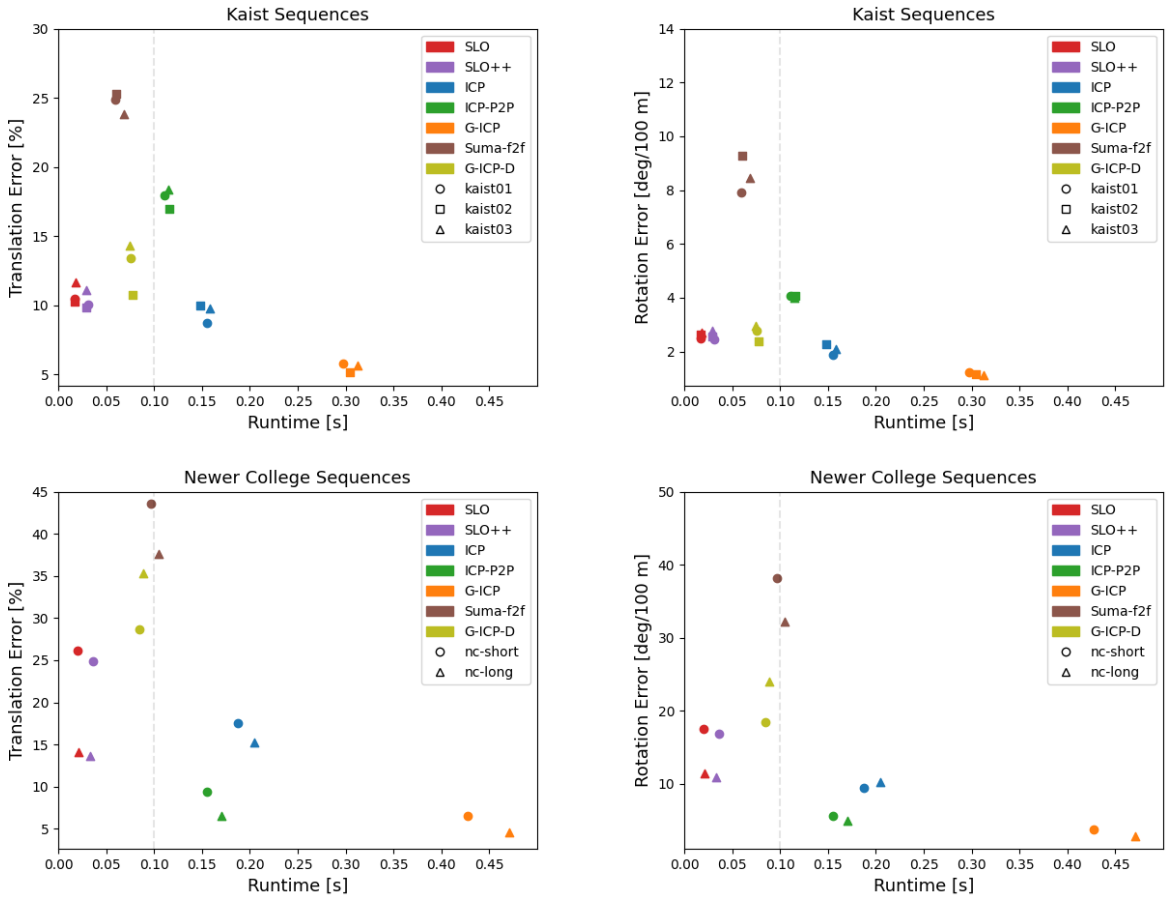

Fig. 5: Relative translational and rotational error reported in relation with the runtime of the approaches. The dashed line represent the sensor frame rate.

to extract salient points in the point cloud and then uses them to compute the odometry. The system can estimate the ego-motion of the sensor accurately by relying solely on these features extracted on a 2D image. Furthermore, we propose a self-supervised strategy to fine-tune the salient point detector online. In this way, the system adapts on-the-fly to the current sensor data stream. We implemented and evaluated our approach on different datasets, provided comparisons to other existing methods, and supported all claims made in this paper. The experiments suggest that the proposed algorithm can improve its performance online while computing the odometry at a much higher frequency than the sensor frame rate.

REFERENCES

[1] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.L. Tai. D3feat: Joint learning of dense detection and description of 3d local features. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 6359–6367, 2020.

[2] J. Behley and C. Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[3] P. Besl and N. McKay. A Method for Registration of 3D Shapes. *IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI)*, 14(2):239–256, 1992.

[4] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[5] C. Choy, J. Park, and V. Koltun. Fully convolutional geometric features. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, pages 8958–8966, 2019.

[6] 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 Transactions on Graphics*, 36(4):76a, 2017.

[7] B. Della Corte, I. Bogoslavskyi, C. Stachniss, and G. Grisetti. A General Framework for Flexible Multi-Cue Photometric Point Cloud Registration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[8] H. Deng, T. Birdal, and S. Ilic. 3d local features for direct pairwise registration. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3244–3253, 2019.

[9] D. DeTone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. In *In Conference on Computer Vision and Pattern Recognition*, June 2018.

[10] L. Di Giammarino, I. Aloise, C. Stachniss, and G. Grisetti. Visual place recognition using lidar intensity information. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 4382–4389, 2021.

[11] M. Fischler and R. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 24(6):381–395, 1981.

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

[13] S. Geman and D. McClure. Bayesian image analysis: An application to single photon emission tomography. *Amer. Statist. Assoc*, pages 12–18, 1985.

[14] P.J. Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

[15] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 23:462–466, 1952.

[16] G. Kim, Y.S. Park, Y. Cho, J. Jeong, and A. Kim. Mulran: Multimodal range dataset for urban place recognition. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, Paris, May 2020.
[17] J. Levinson and S. Thrun. Unsupervised calibration for multi-beam lasers. In *Experimental Robotics*, pages 179–193. Springer, 2014.
[18] S. Liang, Z. Cao, C. Wang, and J. Yu. A novel 3d lidar slam based on directed geometry point and sparse frame. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):374–381, 2020.
[19] F. Moosmann and C. Stiller. Velodyne SLAM. In *Proc. of the IEEE Vehicles Symposium (IV)*, pages 393–398, 2011.
[20] F. Nardi, B.D. Corte, and G. Grisetti. Unified representation and registration of heterogeneous sets of geometric primitives. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):625–632, 2019.
[21] 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 *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 4353–4360, 2020.
[22] R.B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 3212–3217, 2009.
[23] J. Saarinen, T. Stoyanov, H. Andreasson, and A. 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] A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Proc. of Robotics: Science and Systems (RSS)*, 2009.
[25] J. Serafin and G. Grisetti. NICP: Dense Normal Based Point Cloud Registration. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 742–749, 2015.
[26] C. Shi, X. Chen, K. Huang, J. Xiao, H. Lu, and C. Stachniss. Keypoint Matching for Point Cloud Registration using Multiplex Dynamic Graph Attention Networks. *IEEE Robotics and Automation Letters (RA-L)*, 6:8221–8228, 2021.
[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint*, 1409.1556, 2014.
[28] B. Steder, R.B. Rusu, K. Konolige, and W. Burgard. Narf: 3d range image features for object recognition. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, volume 44, 2010.
[29] 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.
[30] D. Streiff, L. Bernreiter, F. Tschopp, M. Fehr, and R. Siegwart. 3d3l: Deep learned 3d keypoint detection and description for lidars. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 13064–13070. IEEE, 2021.
[31] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.
[32] 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.
[33] H. Wang, C. Wang, and L. Xie. Intensity-slam: Intensity assisted localization and mapping for large scale environment. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):1715–1721, 2021.
[34] D.J. Yoon, H. Zhang, M. Gridseth, H. Thomas, and T.D. Barfoot. Unsupervised learning of lidar features for use in a probabilistic trajectory estimator. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):2130–2138, 2021.
[35] J. Zhang and S. Singh. LOAM: Lidar Odometry and Mapping in Real-time. In *Proc. of Robotics: Science and Systems (RSS)*, 2014.