Learning-Based Matching of 3D Submaps from Dense Stereo for Planetary-Like Environments

Hsuan-Cheng Liao[1*], Riccardo Giubilato[1+], Wolfgang Stürzl[1+], Rudolph Triebel[1+]

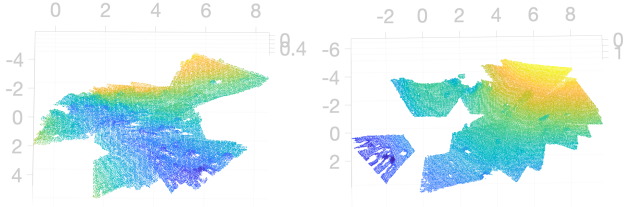
***Abstract*— An autonomous robot typically requires a minimum capability of perceiving the surroundings and locating itself when it is deployed to an unknown environment. Such a task is generally known as Simultaneous Localization and Mapping (SLAM), for which pairwise submap matching is a common foundation for subsequent processes to construct a global map around the robot. While the task has been extensively studied and successfully accomplished with different advanced solutions, their applied domains are rather constrained within indoor or structured regions. In this paper, we enhance a seminal learning-based approach, 3DFeat-Net, with more sophisticated architectures, and evaluate them in extremely unorganized planetary-like environments. Our work demonstrates that the proposed enhancement performs better than classical feature-based algorithms, and therefore outlines a promising direction for future work.**

I. INTRODUCTION

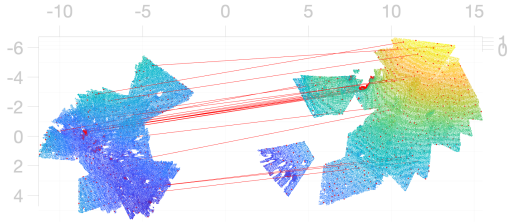
To safely navigate and operate in an unknown and unstructured environment, an autonomous robot has to possess the capability to perform Simultaneous Localization and Mapping (SLAM). To solve the problem, one usually decouples it and starts from submap matching, which essentially refers to the task to compare pairs of local submaps and predict the relative transformations between them [1]. The submap matching task is substantially related to the ordinary problem of point cloud registration, which has been researched extensively in recent years using neural networks [2], [3], [4]. Yet, the field of point cloud registration tends to have a focus on organized data. This type of data usually exhibits stronger details such as corners or edges [5], [6], [7], and sometimes even only consists of object models without cluttered backgrounds [8]. On the contrary, we particularly aim at planetary-like environments. To our understanding, we are among the very few to apply learning-based approaches to such challenging domains.

This work is based upon our previous research [9], [10], [11], [12], in which a 6D SLAM framework with various loop-closure and relocalization algorithms has been proposed for perception and navigation tasks in planetary-like environments. An example of our submap matching scenarios is presented in Fig. 1. We use stereo cameras on our autonomous robot, Lightweight Rover Unit (LRU), thanks to their mechanical simplicity. Nonetheless, they introduce other challenges such as triangulation noises and arbitrary point cloud boundaries while producing submaps. Furthermore, from Fig. 1, it can be observed that not only individual submaps bear weak visual cues, but the overlapping regions of two submaps might exhibit even fewer features.

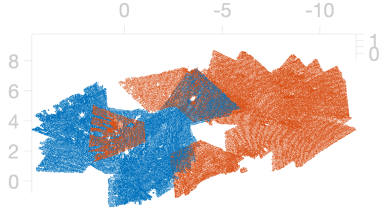
(a) Our autonomous robot, Lightweight Rover Unit (LRU), and a rectified view from its stereo cameras at Mount Etna [9].



(b) A submap pair collected at Mount Etna, colored from yellow (low regions) to blue (high regions).



(c) Point correspondences based on our predicted descriptors.



(d) The submap pair registered by RANSAC.

Fig. 1: An example of our submap matching pipeline.

Our previous research, being the state-of-the-art approaches in these challenging scenarios, has mostly engaged classical keypoint detectors and feature descriptors, whereas this work tackles the submap matching task with neural networks similar to 3DFeat-Net [13]. Specifically, Yew *et. al.* [13] innovates a weakly-supervised framework to train a feature detector and descriptor. Leveraging PointNet [14] and PointNet++ [15] convolutional operations, 3DFeat-Net [13] takes raw point clouds as inputs, minimizes a triplet

[1]The authors are with Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Weßling, Germany
*h.liao@eu.denso.com
+{firstname.lastname}@dlr.de

loss, and outputs a set of salient descriptors for each input point cloud. Subsequently, given a pair of submaps and their predicted descriptors, they employ RANdom SAmple Consensus (RANSAC) [16] to generate the final relative pose and align the submaps. Although there are more recent work showcasing end-to-end neural networks for point cloud registration such as PointNetLK [2], Deep Closest Point (DCP) [3] and RPM-Net [4], 3DFeat-Net [13] and our work differ in two ways. First, most of the end-to-end methods assume explicit ground truths and employ supervised learning schemes. This is not possible in our case as there is no annotated data available. Second, there has not been clear evidence showing their capability to align partially observed and extremely featureless point clouds. Therefore, sharing similar assumptions and seeing promising results from 3DFeat-Net [13], we enhance the framework by first adapting its original descriptor module and then increasing the number of the adapted modules across the architecture. With three different forms of enhancement, we propose *3DFeat-Net-mod*, *3DFeat-Net-seq* and *3DFeatNet-par*. We show in both qualitative and quantitative experiments that our adaptations strengthen the original architecture and the improved networks perform better than the classical approaches in extremely challenging terrains.

To summarize, our contributions include:

- A framework to apply learning-based approaches to the submap matching task in planetary-like environments.
- An extension from an existing powerful framework, 3DFeat-Net [13].
- Thorough experiments to demonstrate the capacity of the improved architectures against the original model and selected classical approaches.

The remainder of the paper is organized as follows: in Section II, an overview of existing work on submap matching is provided. Our adaptations to the original 3DFeat-Net [13] are illustrated in Section III, and evaluation results in Section IV. Finally, several concluding remarks and prospective research directions are suggested in V.

II. RELATED WORK

Feature-based keypoint detection and description approaches have been reported to be accurate and robust, serving as the fundamental blocks in many computer vision applications such as object recognition [17], tracking [18] and segmentation [19]. We review in the following first the handcrafted algorithms and then the learning-based approaches.

A. Handcrafted Keypoint Detectors

A comprehensive list of classical approaches to detect keypoints can be found in [20], [21]. To give an example, the Harris 3D detector [22] is a gradient-based method that finds keypoints with eigenvalue decomposition and Harris value filtering. On the other hand, Scale-Invariant Feature Transform (SIFT) [23] establishes a scale-space of the curvature using the Difference-of-Gaussian (DoG) operator. Although being strong in recognizing objects in different scales, scale-invariant methods are often computationally heavier than others. Moreover, in spite of the reported success in the above papers, most of them are limited to organized domains or clear object models. For particularly rough terrains, our research group at DLR has innovated a curvature-based approach to locate keypoints [9], [10], [11].

B. Handcrafted Feature Descriptors

Before downstream tasks, a feature descriptor is normally employed to derive meaningful embeddings from local regions of detected keypoints. For example, the spatial distribution-based Spin Image (SI) descriptor [24] constructs a local reference axis at the keypoints and bins the axis-perpendicular and axis-parallel distances from the keypoints to their neighbors into histograms. Apart from the spatial distribution-based approaches, several papers consolidate geometric attributes such as normals or curvatures into feature vectors as well. These include Point Feature Histogram (PFH) [25], its extension Fast Point Feature Histogram (FPFH) [26] and Signature of Histograms of Orientations (SHOT) [27]. More examples can be found in [28].

C. Learned Keypoint Detectors

Despite the prosperity of the learning-based research community, we find yet not many algorithms dedicated to 3D keypoint detection. Most of the methods for 3D perception tasks make use of specific mechanisms to represent input point clouds instead. For example, VoxelNet [29] partitions point clouds into voxels before further grouping, random sampling and feature encoding to achieve object detection. 3DMatch [30] also voxelizes raw point clouds into subregions around keypoints and trains a 3D CNN with a contrastive loss for matching correspondences between partial object inputs. A common shortcoming of this type of approaches is that voxelization usually results in a large destruction of data quality and characteristics.

To the best of our knowledge, 3DFeat-Net [13] and Unsupervised Stable Interest Point (USIP) [31] are among the few directly related work. In particular, 3DFeat-Net [13] contains a detector module that predicts attention weights for all points, thereby having the ability to highlight interest points. On the other hand, USIP [31] is trained directly for locating repeatable and accurate keypoints using a Feature Proposal Network (FPN). Although the results are shown promising, their target domains are limited to the rather organized Oxford RobotCar [5], KITTI [6], ETH [7] and ModelNet [8] suites.

D. Learned Feature Descriptors

Other than the aforementioned voxelized [29] and point-direct methods [13], [14], [15], there are also graph-based [19], multi-view [32] and kd-tree representations [33]. Various extents of success have been reported, yet usually at the expense of higher computational effort due to the more sophisticated feature representation schemes. For more details, we defer readers to the original papers, in which

multiple downstream tasks such as object classification and segmentation are performed.

III. METHODOLOGY

Prompted by the advances in deep learning frameworks for computer vision tasks, we adopt neural networks to address the complicated submap matching problem. However, we do identify several difficulties in our specific case. First, there are neither sufficient amount of data nor explicit ground-truth alignments for an end-to-end learning scheme. , as Tthe approximate ground truths in our datasets are only calibrated estimates from the LRU SLAM sessions. Second, such approximate ground truths provide merely model-level annotations but not point-to-point correspondences. Taking the difficulties into consideration, we extend our neural network architectures from 3DFeat-Net [13], which shares the premises similarly. In this section, we first brief the original framework, and then detail our adaptations.

A. The 3DFeat-Net Framework

3DFeat-Net [13] adopts a weakly-supervised learning scheme highlighting a three-branch Siamese architecture. The Siamese network takes as an input a tuple of one anchor point cloud, one positive-sample point cloud, and one negative-sample point cloud. The determination of the positivity/negativity of a sample is based on the overlapping ratio between the sample and the anchor. Details can be found in Section IV-A.

The point clouds in the tuple are processed respectively by the three branches of the Siamese network sharing the same weights. Each point cloud goes through a clustering process, a detector module and a descriptor module. At first, K clusters are generated from an input point cloud P. For each of the clusters in $\{C_1, C_2, ..., C_K\}$, the detector module predicts an attention weight w_k and an orientation value θ_k. Then, the descriptor module rotates the cluster C_k to a canonical configuration with the calculated orientation value θ_k and further extracts a descriptor $f_k \in \mathbb{R}^d$ for the cluster C_k. In particular, the detector and descriptor make use of the approximating function from PointNet [14], which is defined as:

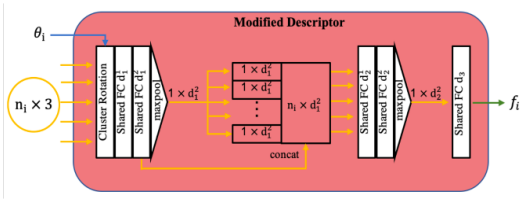
$$f(\{x_1, x_2, ..., x_N\}) \approx g(h(x_1), h(x_2), ..., h(x_N)), \quad (1)$$

where $h(.) : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$ is a shared transforming function operating on each input point x_i and $g(.) : \mathbb{R}^{d'} \times \cdots \times \mathbb{R}^{d'} \rightarrow \mathbb{R}$ is a symmetric function taking all transformed elements as inputs. In practice, $h(.)$ is implemented by multi-layer perceptrons with 1×1 kernels and $g(.)$ by a max-pooling layer.

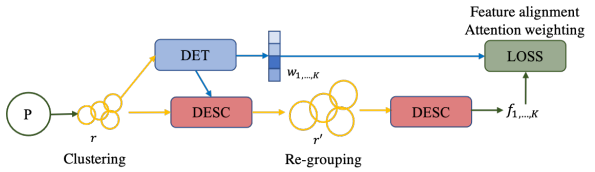
Having obtained three sets of K descriptors from the input tuple, P_{anc}, P_{pos} and P_{neg}, the neural network is trained to optimize the triplet loss, defined as:

$$\mathfrak{L}_{triplet} = \max(0, \mathfrak{D}_{anc,pos} - \mathfrak{D}_{anc,neg} + \gamma), \quad (2)$$

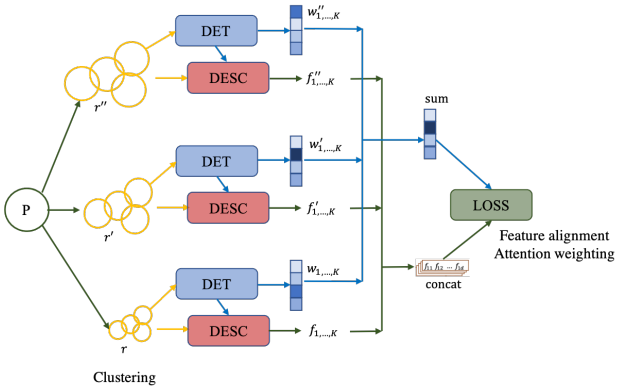
where γ is the margin to be enforced between the positive and negative samples, and the difference between two point



(a) The modified descriptor in 3DFeat-Net-mod



(b) An overview of 3DFeat-Net-seq



(c) An overview of 3DFeat-Net-par

Fig. 2: Our adaptations to the original 3DFeat-Net.

clouds $P^{(s)}$ and $P^{(r)}$ is defined as:

$$\mathfrak{D}_{s,r} = \sum_{i=1}^{K} \left(w_i'^{(s)} \cdot \min_{j=1,...,K} \left\| f_i^{(s)} - f_j^{(r)} \right\| \right), \quad (3)$$

where $w_i'^{(s)}$ is the i-th normalized attention weight calculated by:

$$w_i'^{(s)} = \frac{w_i^{(s)}}{\sum_{l=1}^{K} w_l^{(s)}}. \quad (4)$$

Essentially, optimizing the triplet loss minimizes the difference between the anchor and the positive sample, and maximizes the difference between the anchor and the negative sample. Noticeably, the feature alignment step done by min in (3) naturally mines the hard negatives for $\mathfrak{D}_{anc,neg}$.

At inference time, only one branch of the Siamese network is activated. Additionally, keypoint detection and feature extraction are done separately in two stages. In the first stage, for an input point cloud P with N points, all points are passed to the grouping process as cluster centroids. Instead of $K \times M \times d$, this results in a tensor of size $N \times M \times d$, where M is the number of points in each cluster and d the depth. The tensor is given to the detector module to predict an attention weight and orientation value for each of the N

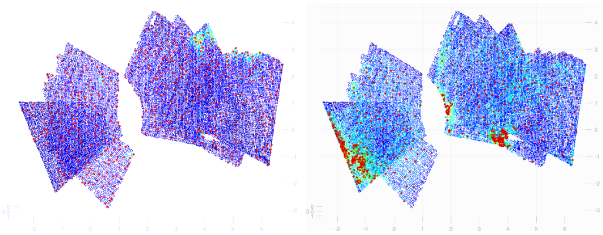


Fig. 3: A qualitative comparison of attention allocation and keypoint detection from 3DFeat-Net *(left)* and 3DFeat-Net-mod *(right)* on a featureless submap. It can be observed that the modified version has a better capability of locating salient regions, rendered in yellow, and nominating keypoints, marked as red dots.

points. Then, a non-maximal suppression algorithm with a fixed radius r_{nms} around each point is performed to retain a predefined number of points that have gained the strongest attention. Subsequently, a filtering step is used to reject points with an attention weight $w_i < \beta * \max(w_1, w_2, ..., w_N)$. Now, the remaining are the detected keypoints, whose total number, say Z, depends on β. In the second stage, the descriptor module extracts a feature vector from every detected keypoint and its local supporting region. Hence, the resulting tensor of the inference pipeline has the size $Z \times 1 \times d$, where d is the dimension of the feature vector set at the training phase.

It is noteworthy that such framework outputting a set of salient points and descriptors for each raw point cloud bears a good resemblance to the classical pipelines [21] and differs from end-to-end learning schemes [2], [3], [4], which usually require a larger amount of labelled data. Interested readers are deferred to the original 3DFeat-Net paper for details [13].

B. Our Adaptations

We depict in the following the main enhancement in our work. As the first step, we modify the structure of the original descriptor module as shown in Fig. 2a. Precisely, we add one more shared layer after the concatenation of the contextual feature and individual point descriptors. This is inspired by the Universal Approximation Theorem [34], which states that a neural network with two layers can approximate any well-behaved function for some bounded inputs. Additionally, with even more layers, the neural network can still approximate the well-behaved function by using the same weights in the first layers and converging to the identity function in the later layers. Nevertheless, to prevent overfitting and considering the sizes of our datasets, we increment simply one layer and call the modified network *3DFeat-Net-mod*. Due to space limit, we do not show performance of the original architecture later in Section IV, but only highlight the effect of this amendment with Fig. 3.

A further adaptation is inspired by PointNet++ [15], which suggests that the size of receptive fields plays a critical role in general perception tasks. The original 3DFeat-Net [13] allows for only one level of predefined clustering radius, and by experiments we find it incapable of grasping contextual information for the planetary-like submaps. Hence, we propose two forms of further enhancement. For the first form, we place two enhanced descriptor modules in sequence and a re-grouping process in between. Such operation resembles the multi-resolution grouping scheme in [15]. More exactly, having obtained a set of feature vectors $\{f_1, f_2, ...f_K\}$ from the first descriptor module with a smaller base scale, we pass them to the second descriptor, which re-groups these $K \times 1 \times d$ feature vectors into $K \times M \times d$ clusters using a larger base scale. Finally, the resulting output for each point cloud has size $K \times 1 \times d'$. It is noteworthy that we remove the rotation operation at the second descriptor. This is different from PointNet [14] and PointNet++ [15], which continuously predict orientation values and transforms vectors in feature spaces. We find such transformation unnecessary as these values are difficult to train and can be approximated by the kernel operations between the layers. Our sequential network, called *3DFeat-Net-seq* is shown in Fig. 2b.

The second form of enhancement is to perform grouping of clusters with three different radii in parallel, similar to the multi-scale grouping scheme in [15]. The parallel network, *3DFeat-Net-par*, is depicted in Fig. 2c. Such architecture returns three attention weights and three feature vectors of size d, d' and d'' for one input point cloud. To compute the triplet loss at last, we sum the attention weights together and concatenate the vectors into one final descriptor of size $d+d'+d''$. Similarly, at inference time, the attention weights are summed and the encoded descriptors are concatenated to compose the final outputs. With the described techniques, we achieve at more complete model designs, preserving finer details and observing larger fields of submaps at the same time.

IV. EVALUATION RESULTS AND DISCUSSIONS

A. Experiment Datasets and Settings

We evaluate our architectures in ten planetary-like datasets, including five indoor sets from our laboratory (Lab1, Lab2, Lab3, Lab4 and Lab5), four outdoor sets from Mount Etna (Etna1, Etna2, Etna3 and Etna4), and one outdoor set from Morocco (Morocco1). As introduced, these datasets are collected throughout our previous work and detailed in [9], [10], [11], [12]. Fig. 4 exhibits three sample pairs with their approximate ground-truth alignments. It gives an impression on the difficulty of our target scenarios compared to orginary point cloud datasets [5], [6], [7], [8].

To foster learning, we split each of our datasets with a 7:3 ratio for training and inference. To form a tuple of point clouds for training the Siamese network, we determine positive and negative samples for each anchor submap in two steps. First, we exploit the approximate ground-truth relative poses from the SLAM sessions to transform all submaps onto a common coordinate system. Second, we make each submap the anchor and traverse through other submaps to do the following classification: a submap is a positive sample if it has an overlapping ratio above 0.3 with the anchor, and a

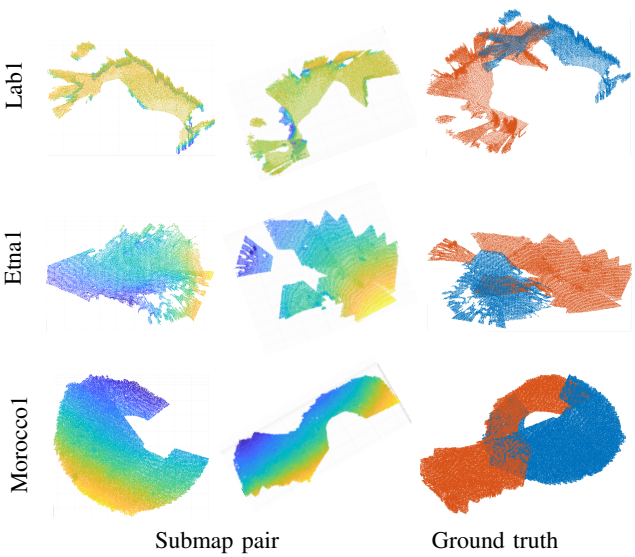


Fig. 4: Sample submap pairs and their ground-truth alignments from different environments. Individual submaps are colored from yellow (low regions) to blue (high regions).

submap is a negative sample if the overlapping ratio is below 0.05. We adopt the overlapping ratio defined as:

$$IoU(P^{(s)}, P^{(r)}) = \frac{Area(P^{(s)}) \cap Area(P^{(r)})}{Area(P^{(s)}) \cup Area(P^{(r)})}, \quad (5)$$

where $P^{(s)}$ and $P^{(r)}$ are the two submaps under invesitgation. Although they are essentially 3D point clouds, using 2D area measurements on the $x-y$ plane is still valid given our assumption that all submaps are oriented upright along the gravitational direction.

The training of the neural network is conducted with a batch size of 4 triplets. All input submaps are first downsampled to $N = 4096$. Such random input dropout is shown beneficial for a better generalizability [13]. Then, we sample $K = 512$ centroids and group $M = 64$ neighboring points around every centroid using various base scales. The parameters for each neural network architecture is set as follows:

- 3DFeat-Net-mod (inference time $\approx$ 0.8 second)
  - Base scale: 0.5 m
  - Descriptor MLP sizes: [32, 64] | [256, 128] | [32]
- 3DFeat-Net-seq (inference time $\approx$ 1.2 seconds)
  - First base scale: 0.3 m
  - First descriptor MLP sizes: [32, 64] | [64, 128] | None
  - Second base scale: 1.0 m
  - Second descriptor MLP sizes: [128, 256] | [128, 64] | [32]
- 3DFeat-Net-par (inference time $\approx$ 1.5 seconds)
  - First base scale: 0.3 m
  - Second base scale: 1.0 m
  - Third base scale: 2.0 m
  - All descriptor MLP sizes: [32, 64] | [128, 64] | [32]

The depths of each multi-layer perceptron (MLP) are expressed within one pair of square brackets. For example, the descriptor module in 3DFeat-Net-mod has three MLPs, see 2a. The first one has two layers of depth 32 and 64. The second one has another two layers of depth 256 and 128. Finally, the third layer serving as a final operator has the depth 32, which will be the length of the feature vector. For the first-level descriptor in 3DFeat-Net-seq, we do not need a third MLP to refine the vectors as the outputs are passed sequentially to the second-level descriptor. On the contrary, in 3DFeat-Net-par, all three levels of descriptors require a refining third MLP because they work in parallel. The three feature vectors of length 32 will be concatenated to form a final product of length 96 before being matched and weighed at the triplet loss.

We fix the discriminating margin γ in the triplet loss to 1. As [13] suggests, we employ the ADAM optimizer with a learning rate of 1e-5 and apply further data augmentation at the training phase, including individual point jittering, submap shifting and submap 1D rotation. The rotation is only applied on the z-axis based on our assumption that all submaps are obtained in an upright orientation using the LRU on-board sensors. The training took around 16 hours on a Nvidia GeForce GTX Titan X with the Maxwell architecture.

For inference, we set the non-maximum suppression radius r_{nms} to 0.5 meter and minimum response ratio β to 1e-6. Apart form the non-maximum suppression step, we include 512 additional points that have the highest attention weights to rely more on regions with focused attention. This is found helpful in the submap matching task as more keypoints are located at important areas.

B. Keypoint Detection

In Fig. 5, we compare the three networks against three handcrafted approaches, namely Harris 3D [22], SIFT 3D [23], and curvature-based detectors [9], [10], [11]. It is discovered that the keypoints predicted by the Harris 3D and SIFT 3D detectors are rather unconcentrated. On the other hand, the curvature-based detector clearly marks down rocks, craters and regions with high curvatures. However, there are two drawbacks. First, the keypoints are too focused in local regions. This might cause an issue later in the submap matching task because global contexts are greatly neglected. Second, with an inflexible curvature threshold, the number of filtered keypoints varies largely in different submaps. This might also hinder its stability.

The outputs from 3DFeat-Net-mod show a certain degree of success, exhibiting groups of keypoints around the holes in the point clouds and distributing a small portion of focus over the submaps. However, their are also clusters of keypoints on the submap boundaries, which might be misleading for the subsequent feature descriptor and registration estimator. Above all, 3DFeat-Net-seq and 3DFeat-Net-par not only successfully identify distinctive areas as the curvature-based method does, but also suggest keypoints across the entire point clouds. Therefore, we speculate that 3DFeat-Net-seq and 3DFeatNet-par have a better potential to recognize finer

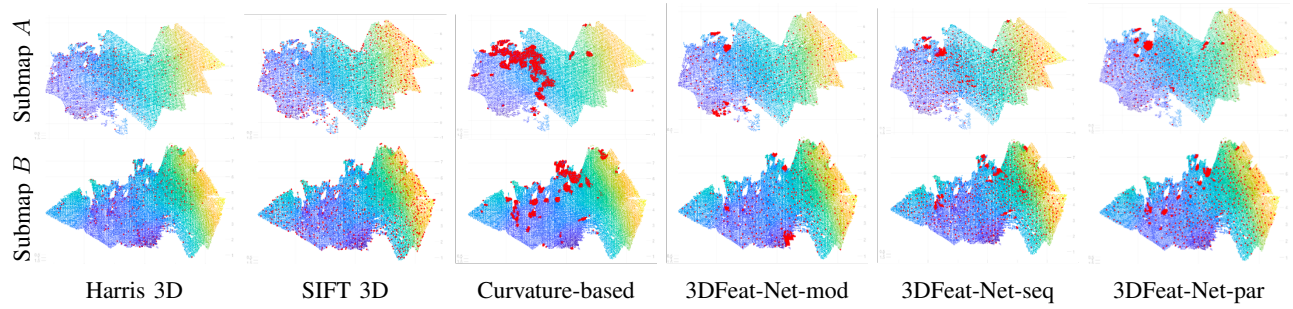


Fig. 5: Qualitative results of keypoint detection on a submap pair from Etna1. The learning-based approaches presents better balances between repeatable points in salient regions and scattered points across entire submaps.

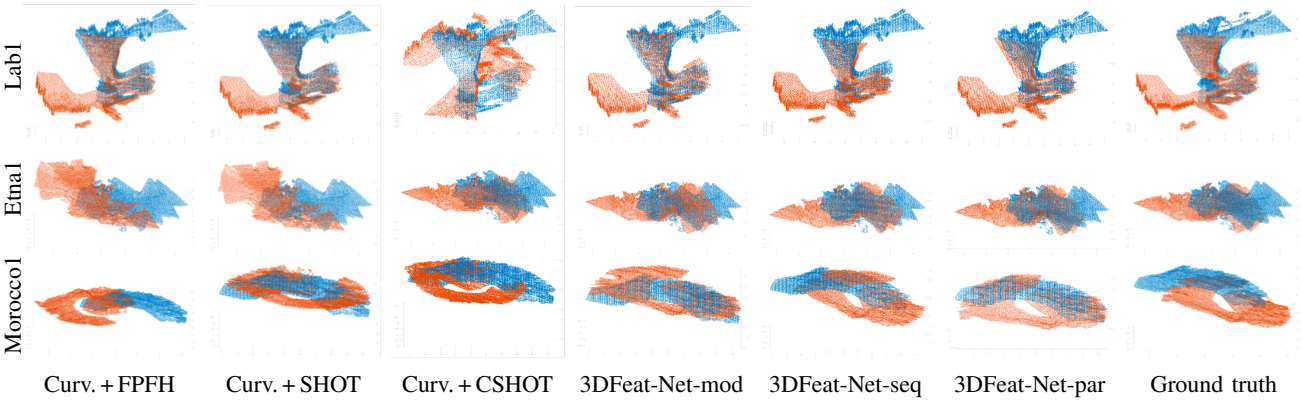


Fig. 6: Qualitative results of submap matching on three submap pairs from Lab1, Etna1, and Morocco1 respectively. 3DFeat-Net-seq and 3DFeat-Net-par have the closest solutions to the ground truths.

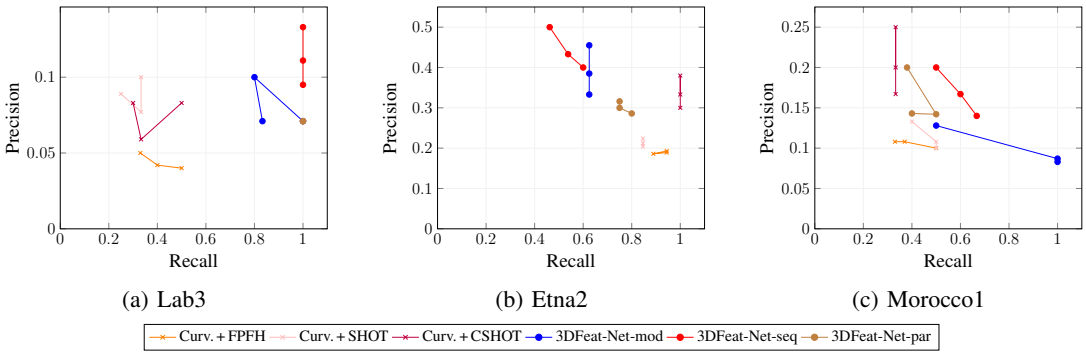


Fig. 7: Quantitative results of submap matching validation on various datasets. Despite subtle differences for the level of difficulty, the $Precision - Recall$ curves of the learning-based approaches generally lie at the top right from those of the classical approaches.

structures and associate them with larger contexts within a submap. Based on our analysis on the keypoints detection task, we bring forth the curvature-based detector and three learning-based approaches for further investigation.

C. Feature Description and Submap Matching

1) Qualitative Results: In Fig. 6, we show three sets of submap matching results given by six approaches, alongside with the approximate ground-truth alignments. In terms of submap properties, the easiest pair from Lab1 bears the most noticeable and substantially shared structures. The intermediate pair from Etna1 also contains a handful of rocks in common, whereas the hardest pair from Morocco1 exhibits rather plain surfaces and obscure characteristics.

We find all approaches align the Lab1 pair rather well except for the CSHOT descriptor. In fact, the descriptor struggles to align any of the three target pairs accurately. It is presumed that since most of our original submaps are not abundant in colors, the lengthy CSHOT descriptor having an additional operation in the *CIELab* color space may negatively affect the expressiveness and conciseness of the feature output. On the contrary, its simpler variant, the SHOT descriptor, aligns the easy Lab1 pair nicely. The

performance is similar to that of the FPFH descriptor. Both SHOT and FPFH succeeds in the first pair, but fails in the other two. Such results are consistent with [13], in which the authors find common handcrafted approaches occasionally work well in easier scenarios yet not in most of the harder cases.

We find the learning-based methods more capable of handling tricky submaps. For example, 3DFeat-Net-mod matches the Etna1 pair much better than the above descriptors. We can see the registered pair has a coinciding hole and the result orients similarly to the approximate ground truth. However, for the hardest Morocco1 pair, 3DFeat-Net-mod still lacks the ability to extract representative features that enclose both local and contextual information. In contrast, it is solved by 3DFeat-Net-seq and 3DFeat-Net-par to a satisfactory extent. Although the submap registration results are not identical to the approximate ground truth, they are considered rather successful and can be refined by the Iterative Closest Point (ICP) algorithm for further optimizing the registration result [10].

2) Quantitative Comparisons: In this section, we conduct a quantitative study to provide more scientific evidence regarding the submap matching task. To such end, we use the *Precision-Recall* metric for examining the capability of the proposed approaches to classify submap pairs into two categories, validated or not validated groups. Specifically, having obtained two sets of feature vectors from a submap pair, we perform correspondence matching and execute RANSAC. To classify a submap pair as a validated match, we require that the inlier proportion of the found correspondences calculated by RANSAC is larger than a certain threshold. We set the inlier proportion threshold as 0.3 for Etna1, Etna2, Etna3 and Etna4, 0.2 for Morocco1, and 0.15 for the indoor datasets.

Mathematically, the *Precision-Recall* metric is defined as:

$$Precision = \frac{TP}{TP + FP}\,,\ Recall = \frac{TP}{TP + FN}\,, \quad (6)$$

where TP is the count of validated submap matches being correct, FP is the count of incorrect validated matches, and FN is the count of true matches not being validated. To form the set of correct ground truths, we transform pairs of submaps onto a common coordinate system and include the submap pairs with an overlapping ratio above 0.3. In addition, we use distance ratio matching to find correspondences as suggested by [23]. For each descriptor kind, we apply distance ratios of 0.85, 0.9 and 0.95. This is different from [13], which uses a one-directional nearest neighbor matching scheme. Our matching process has higher standards so that the subsequent RANSAC results are more stable and eligible.

Fig. 7 gives the *Precision-Recall* curves of all descriptors in several environments. It is observed that the learning-based approaches deliver generally a better performance than the handcrafted ones. The *Precision-Recall* curves of the learning-based approaches are located either above or on the right of the others, indicating they are more accurate in validating true submap matches and more robust against seemingly false submap matches.

We now inquire further the set of true positives from the above quantitative experiment and calculate the numerical errors between the predicted registration and the approximate ground truths. Clearly, taking our challenging scenarios into consideration, the errors will be much larger than those in [13], [26], [30]. However, we aim at making a numerical comparison among the proposed descriptors in complicated planetary-like environments. The performances are evaluated with two metrics, namely the Relative Translational Error (RTE) [35] and Relative Rotational Error (RRE) [36]. The RTE is calculated using the Euclidean distance between the predicted translational vector T_{est} and the ground-truth translation vector T_{gt}:

$$RTE = ||T_{gt} - T_{est}||_2. \quad (7)$$

The RRE is defined by the sum of the absolute differences in three rotational Euler angles:

$$RRE = |\alpha| + |\beta| + |\gamma|, \quad (8)$$

$$\alpha, \beta, \gamma = F(R_{gt}^{-1} R_{est}), \quad (9)$$

where $F(.)$ denotes the transformation from a rotation matrix to Euler angles, R_{est} and R_{gt} are the estimated and ground-truth rotation matrices respectively.

We highlight in Table I the numerical errors of two data sequences, from indoor and outdoor domains respectively. We report the best, the mean and the standard deviation of each method. The values are obtained with the distance ratio matching scheme using a ratio threshold of 0.95. We draw two observations from the outcome. First, there is a great performance distinction between the handcrafted approaches and the learning-based approaches. We find that in the challenging outdoor environments, the learning-based approaches achieve much better best and mean scores. Although the performance superiority is not as distinct in the indoor domain, we spot the second observation that either 3DFeat-Net-seq or 3DFeat-Net-par has the best scores in general. The above observations raise our confidence in the learning-based approaches.

V. CONCLUSION

In this work, we have extended 3DFeat-Net, an weakly-supervised learning framework, into 3DFeat-Net-mod, 3DFeat-Net-seq and 3DFeat-Net-par, and conducted experiments in planetary-like environments. Similar to 3DFeat-Net, our networks do not explicitly regress transformation matrices, but predict salient keypoints and extract expressive features from raw point clouds. Evaluation results have shown positive effects of our adaptations. Given the promising outcome, we name two directions for future work. First, tuning the clustering radius as a trainable parameter presumably make the receptive fields more flexible and robust against noisy and uneven point clouds. Second, with more data collected in the future, we aspire to design an end-to-end submap matching framework that has an even greater power to perceive challenging submaps and perform the matching task more effectively and efficiently.

TABLE I: Numerical errors of submap matching on Lab1 and Etna2.

Method	Lab1		Etna2	
	RTE (m)	RRE (°)	RTE (m)	RRE (°)
Curv. + FPFH	0.128, 2.877 ± 2.491	**1.038**, 125.275 ± 138.205	0.482, 4.985 ± 2.565	53.938, 212.261 ± 92.142
Curv. + SHOT	0.996, 5.467 ± 3.269	3.769, 149.769 ± 158.523	2.890, 5.404 ± 1.308	48.164, 157.297 ± 84.687
Curv. + CSHOT	0.063, 2.491 ± 2.381	1.310, 60.394 ± 89.755	2.928, 5.821 ± 2.364	102.963, 226.862 ± 64.616
3DFeat-Net-mod	0.241, 2.496 ± 2.401	2.263, 107.951 ± 138.469	0.495, 1.768 ± 0.948	5.422, 53.221 ± 88.728
3DFeat-Net-seq	**0.104**, 3.847 ± 2.701	4.349, 109.648 ± 93.178	**0.428**, **1.723** ± **0.981**	2.755, 43.477 ± 32.109
3DFeat-Net-par	0.173, **2.401** ± **2.372**	1.091, **52.634** ± **79.636**	0.471, 1.991 ± 1.084	**0.679**, **10.171** ± **9.300**

ACKNOWLEDGMENT

This work was supported by the Helmholtz Association, project alliance ROBEX (contract number HA-304) and project ARCHES (contract number ZT-0033).

REFERENCES

[1] T. Taketomi *et al.*, "Visual slam algorithms: A survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, 2017.

[2] Y. Aoki *et al.*, "Pointnetlk: Robust and efficient point cloud registration using pointnet," 2019, pp. 7156–7165.

[3] Y. Wang and J. Solomon, "Deep closest point: Learning representations for point cloud registration," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 3522–3531.

[4] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," *2020 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11 821–11 830, 2020.

[5] W. Maddern *et al.*, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017.

[6] A. Geiger *et al.*, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[7] F. Pomerleau *et al.*, "Challenging data sets for point cloud registration algorithms," *The International Journal of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, 2012.

[8] Z. Wu *et al.*, "3d shapenets: A deep representation for volumetric shapes," 2015, pp. 1912–1920.

[9] R. Giubilato *et al.*, "Relocalization with submaps: Multi-session mapping for planetary rovers equipped with stereo cameras," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 580–587, 2020.

[10] C. Brand *et al.*, "Submap matching for stereo-vision based indoor/outdoor slam," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 5670–5677.

[11] M. J. Schuster *et al.*, "Distributed stereo vision-based 6d localization and mapping for multi-robot teams," *Journal of Field Robotics*, vol. 36, no. 2, pp. 305–332, 2019.

[12] C. Gentil *et al.*, "Gaussian process gradient maps for loop-closure detection in unstructured planetary environments," *ArXiv*, vol. abs/2009.00221, 2020.

[13] Z. J. Yew and G. H. Lee, "3dfeat-net: Weakly supervised local 3d features for point cloud registration," in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XV*, ser. Lecture Notes in Computer Science, V. Ferrari *et al.*, Eds., vol. 11219. Springer, 2018, pp. 630–646.

[14] C. R. Qi *et al.*, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 77–85, 2017.

[15] ——, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *ArXiv*, vol. abs/1706.02413, 2017.

[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381–395, Jun. 1981.

[17] C. R. Qi *et al.*, "Volumetric and multi-view cnns for object classification on 3d data," in *2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[18] Q. He *et al.*, "Svga-net: Sparse voxel-graph attention network for 3d object detection from point clouds," 2020.

[19] Y. Wang *et al.*, "Dynamic graph cnn for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, Oct. 2019.

[20] F. Tombari *et al.*, "Performance evaluation of 3d keypoint detectors." *Int. J. Comput. Vis.*, vol. 102, no. 1-3, pp. 198–220, 2013.

[21] K. Joshi and M. Patel, "Recent advances in local feature detector and descriptor: a literature survey," *International Journal of Multimedia Information Retrieval*, vol. 9, pp. 1–17, 12 2020.

[22] I. Sipiran and B. Bustos, "Harris 3d: A robust extension of the harris operator for interest point detection on 3d meshes," *The Visual Computer*, vol. 27, pp. 963–976, 11 2011.

[23] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–, 11 2004.

[24] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, p. 433–449, May 1999.

[25] R. B. Rusu *et al.*, "Aligning point cloud views using persistent feature histograms," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 3384–3391.

[26] ——, "Fast point feature histograms for 3d registration," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3212–3217.

[27] S. Salti *et al.*, "Shot: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, 08 2014.

[28] X. Han *et al.*, "3d point cloud descriptors in hand-crafted and deep learning age: State-of-the-art." *arXiv: Computer Vision and Pattern Recognition*, 2018.

[29] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *2018 IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4490–4499.

[30] A. Zeng *et al.*, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[31] J. Li and G. Lee, "Usip: Unsupervised stable interest point detection from 3d point clouds," 10 2019, pp. 361–370.

[32] H. Su *et al.*, "Multi-view convolutional neural networks for 3d shape recognition," in *Proc. ICCV*, 2015.

[33] W. Zeng and T. Gevers, "3dcontextnet: K-d tree guided hierarchical learning of point clouds using local and global contextual cues," in *Computer Vision – ECCV 2018 Workshops*, L. Leal-Taixé and S. Roth, Eds. Cham: Springer International Publishing, 2019, pp. 314–330.

[34] H. Mhaskar, "Approximation properties of a multilayered feedforward artificial neural network," *Advances in Computational Mathematics*, vol. 1, pp. 61–80, 1993.

[35] G. Elbaz *et al.*, "3d point cloud registration for localization using a deep neural network auto-encoder," *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2472–2481, 2017.

[36] Y. Ma *et al.*, "Fast and accurate registration of structured point clouds with small overlaps," *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 643–651, 2016.