3D to 2D feature matching for next generation 3D mapping algorithms

Anougmar Mohamed Noufel
MT-MA1
mohamed.anougmar@epfl.ch

Gessler Frédéric
IN-MA3
frederic.gessler@epfl.ch

Mouzakidou Kyriaki
EDOC-EDCE
kyriaki.mouzakidou@epfl.ch

***Abstract*—Since the last decade, machine learning algorithms are studied for the purpose of high accuracy 3D matching. As the 2D and 3D descriptors are widely available, a real challenge is to find an association between these representations. However, a novel method to learn a local cross-domain descriptor for 2D image and 3D point cloud matching that works on both is proposed in [1].**

This project aims at reproducing the result of this research on a topological data containing real and synthetic data. The challenge lies in dealing with such data characteristics because of the structure, scale, sample density, etc., which increases drastically the training time. This data is provided from SPAD (Single Photon Avalanche Diod) sensors thanks to the latest developments of LIDAR (Light Detection And Ranging) on chip.

I. Introduction

In this project, we aim at understanding a new method to learn a local cross domain descriptor for 2D image and 3D point cloud matching, which maps both 2D and 3D input into a shared latent space representation using a neural network dual auto-encoder. We first focus on reproducing the paper results, then we start on testing this algorithm to real-mapping data of EPFL provided by TOPO laboratory. The TOPO training dataset is synthetically generated from high fidelity 3D model of EPFL.

The report is organized as follows. In section II we introduce our frame of work: the LCD DNN. In section III we attempt to reproduce the experimental results of the LCD paper, taking the opportunity to introduce the considered computer vision tasks: 2D matching, 3D alignment (registration and depth estimation are left for future work). Section IV introduces the topological dataset and demonstrates the abilities of LC descriptors on the EPFL dataset. We introduce and evaluate strategies for learned descriptors for topological data in section V. Section VI outlines future work and section VII concludes the report.

II. Learned Cross Descriptors

In computer vision, many tasks rely on robust *descriptors* which can recognize low level features of the local patch in a 2D or 3D scenes, such as color, edge shape, etc. Such descriptors indeed aid at identifying patches in different projections of the same scene that correspond to the same ”object”. To be considered robust, a descriptor should be able to recognize these low level features between 2 scenes of the same ”object” regardless of the orientation differences, and even without they all being contained in both scenes. In relation to the research, the robustness of the descriptors is evaluated when being used for image matching and 3D registration.

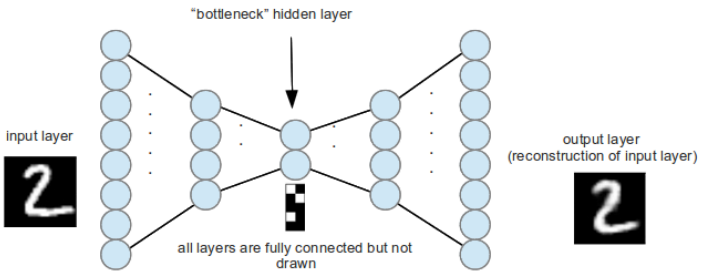

Fig. 1: Auto encoder architecture. *Credit: Ngiah Ho, François Chollet*

Such descriptors were traditionally hand-crafted by computer vision experts, but with the recent success of Deep Learning in computer vision tasks, deep neural networks can effectively learn robust descriptors both in 2D and 3D.

Despite those advances in both 2D and 3D learned descriptors, learned single domain descriptors from one domain to another have no relationship, due notably to the very different representations of both domains (typically mesh vs. point cloud), making it impossible to match a 2D patch to a local 3D point cloud of the same local point in the same scene. Such **cross domain** descriptors have great applications, such as 2D to 3D content retrieval or depth estimation, and in our case topology-related tasks.

LCD is the first descriptor learning Deep Neural Network to be cross domain, being able to learn descriptors that match 2D and 3D point, by being trained on 2D-3D matches. LCD is based on two auto-encoders, one for images and one for point clouds that are trained **jointly**, a key insight in achieving domain crossing invariance.

Auto-encoders [2] are a Deep Neural Network architecture composed of an encoder and a decoder, that are able learn a compressed representation of an (large) input.

In LCD, each auto-encoder is associated to a loss that encodes ”how good is the auto-encoder at deconstructing and reconstructing an input”, which is essentially the domain-specific distance between the input layer and the output layer. For 2D, the appropriate loss is the photo-metric loss which

computes the mean squared error (MSE) between an input 2D point patch and its reconstructed one, whereas the Chamfer loss is used for the 3D auto-encoder. The Chamfer loss computes the distance between an input point set and its reconstructed point set using the Chamfer distance. But to ensure that both auto-encoders learn representations in the same space, the distance between descriptors is added to what the authors call the triplet loss. This loss minimizes the distance between an anchor and a positive ($\mathcal{F}(d_a, d_p)$), while maximizes the distance between the anchor and a negative ($\mathcal{F}(d_a, d_n)$) considering a margin m.

$$\mathcal{L}_{triplet} = max(\mathcal{F}(d_a, d_p) - \mathcal{F}(d_a, d_n) + m, 0) \quad (1)$$

where $\mathcal{F}$ is the distance function. (d_a, d_p, d_n) is a triplet of an anchor, a positive, and a hardest negative, respectively.

III. Reproducing experimental results

A. Training

The LCD paper indicates that the training should last 17 hours on single V100 GPU, for 250 epochs. Hence, all our training experiments were conducted on SCITAS' izar cluster, which makes the same V100 GPUs available, and which conveniently enables the sharing of datasets across the group.

To issue the training, we simply create the appropriate virtual environment on the cluster and create a job submission SLURM file. However, a first major roadblock in reproducing the model is that the latest commit (5be8134) for the paper companion repository, the latest at the time, had a major performance issue resulting in a projected training time of circa 20 hours per epoch. Preliminary investigations revealed that the problem stem from the I/O, as many micro-experiments confirmed (such as assessing the dataset location). Later discussion with the author [3] confirmed that the Data Loader was not up to date and that reaching this performance entailed loading all the dataset **in memory** (as opposed to caching only a sub sample), requiring a node with 100GB of RAM.

This fix brought down the training time to 20 minutes, which proved tractable but still $5\times$ slower than expected. Further attempts to profile the code with our own timer based on CUDA events and post-batch synchronization, seemed to indicate that the bottleneck was no longer the I/O. Sub GPU utilization was a plausible hypothesis on the cause of low performance, but raising the batch size (with the appropriate learning rate scaling of course) did not yield better performance. The situation is still unclear at this moment but the data loader later posted [4] by the author seemed to hint that the author only used half of the dataset. Constrained by time, we decided to move on and explain the difference by dataset inconsistencies.

B. 2D matching

Since no explicit script is provided by the author for the 2D image matching, we created one for extracting and matching 2D features in pairs of images.

It is worth mentioning that before using the LCD descriptor, keypoints need to be detected on the images. The so called SuperPoint detector that the author of the paper used was proprietary and for this reason we decided to use ORB [5], a fast and robust handcrafted feature detector.

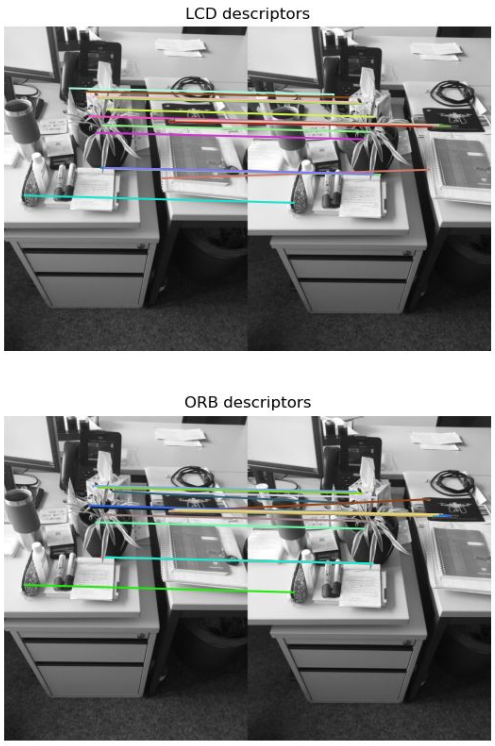

Fig. 2: 2D image matching of data similar to the trained ones.

After the keypoints were detected, patches of 64 pixel size were created around the keypoints forming characteristic features. By using the pre-trained model of the author, 256-length descriptors (embeddings) were created for each patch. We chose to perform the experiment using the 256-length descriptors, since the longer the embedding, the more efficient the description of the feature.

In order to evaluate the accuracy and the validity of the LCD descriptors, we also computed the ORB descriptors for the same keypoints and evaluated their matching. Fig.2 shows how the LCD descriptor matching performs in a pair of images similar to those used from the author in the pre-trained network. The results confirm the validity of the LCD neural network in images of same scale and content with those used by the author.

C. 3D alignment

3D alignment is the only experiment currently published by the authors. For brevity's sake we ported the alignment code almost verbatim to a notebook, and swapped the 3D visualisation backend to use Pyntcloud, an interactive point cloud library.

Unfortunately, we could not locate other sample pairs, as the location of the original 3D datasets was unclear, and instead moved on to the tests with our dataset.

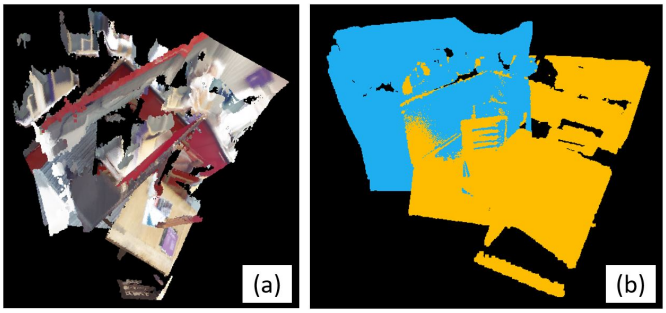

Fig. 3: (a) Original point clouds put in the same scene: both scene patches are not aligned and end up overlapping and (b) the same scene patches, aligned and colored.

IV. Topological dataset

A. Data description and preparation

The main goal of this project is to assess the effectiveness of the Learned Cross Descriptors, which the authors state should be generic (applicable to any scene), on topological data. Two data sets are made available by the laboratory: 'La Comballaz' and 'EPFL'. In this work we only evaluate the 'EPFL' dataset. A problem with the 'La Comballaz' dataset is that the sky is present in some scenes, leading to restricted keypoint detection. If we remove the sky-pixels and depending on the overlap of the images, the feature matching will be limited in the small effective area of the images. Both datasets, despite different file formats, are both presented in classical 2D RGBA grid for images and the same grid that maps pixel to a point in 3D space. This tremendously simplifies our job as we need matches for the training and validation tasks. The format needed by the encoders are 64x64 patches for images, and point clouds of 1024 RGB points. Hence, to generate training matches data, we can tile both grids, generating image patches and the getting the corresponding point cloud by concatenating the RGB values from the first grid to the XYZ values from the second grid and flattening to XYZRGB. All the point cloud coordinates have to be normalized to the -1, 1 cube (inferred from the original code).

The goal of the application in topological data is to prove that 2-dimensional and 3-dimensional data can be matched using one common cross-domain descriptor under one pipeline. Initially it was considered that this descriptor is data-dependent and thus it has to be re-trained each time on the needed application. That is why the synthetic datasets were generated from TOPO laboratory, in order to train such a network. However, it is important that Network similarly handles the synthetic and the real datasets, and that is what we are also called to show with this application.

B. 2D matching

We performed the 2D matching task, as described previously in the reproducing task, but this time in real and synthetic pairs of images of the EPFL dataset. We used the pre-trained network, in order to assess whether it is data-invariant or not, and compared the results with the corresponding ones from the ORB descriptor. Three test cases on the EPFL dataset are presented in Fig. 5: (1) 2 real images with 80-90 %, (2) 2 synthetic images with 80-90 % and (3) a real and its corresponding synthetic image. In the first line of Fig. 5, with red frames we can see the common overlapping area between each pair of images. The second line shows the matches found using the LCD descriptor, while the third line shows the one found using ORB.

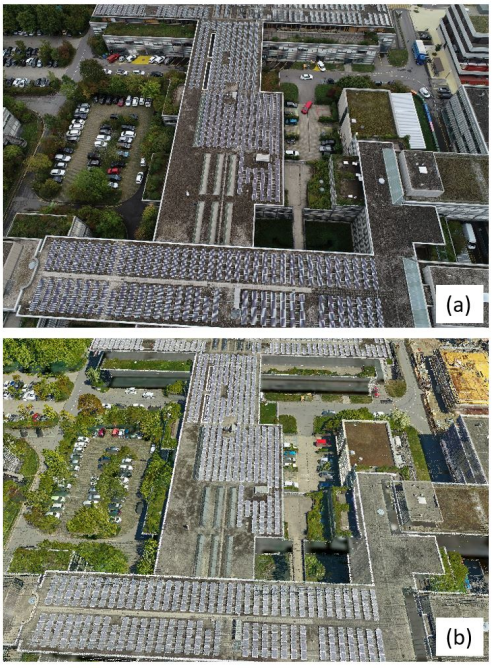

Fig. 4: (a) Real data sample and (b) synthetic data sample

The first notable result was that LCD succeeded to match features of the real EPFL dataset even though it depicts different scenes at different scales than the ones it was trained with. Secondly, we also observed that even when the images are synthetic, LCD recognizes the corresponding features and matches successfully, while ORB seems to be disorganized. However, in test (3) where the 2 images depict exactly the same scene, LCD descriptor fails to correctly match the features. This is probably attributed to the different lighting of the scenes, which seems to be a weakness of the LCD descriptor.

C. 3D alignment

We perform the same alignment task on the EPFL dataset, on two point clouds of close "flight snapshots". As shown in figure 6, LC descriptors perform well.

D. Reconstruction tasks

To properly assess the performance of the LC descriptors, we now present reconstruction experiments that consist in encoding the input in some domain (2D or 3D) and decoding the given descriptors to get a visual understanding of how well the LCD descriptor perform.

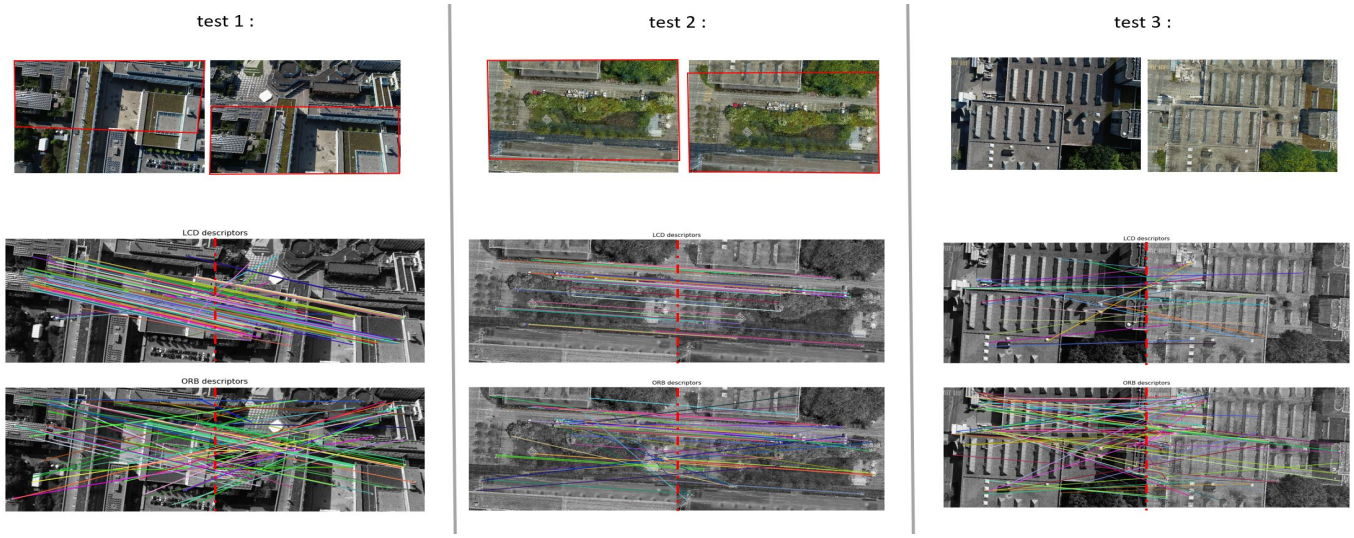

Fig. 5: 2D feature matching tests on the 'EPFL' datasets.

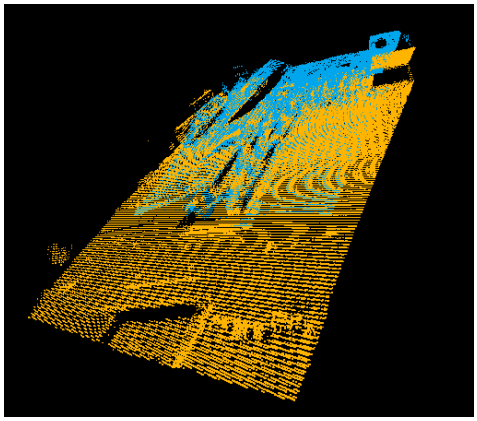

Fig. 6: Alignment task on the EPFL dataset.

First, as a comparison point we take samples from the 3DMatch training set and introduce the reconstruction task.

V. Adapting LCD to TOPO data

Anticipating poor performance of the LCD descriptors on synthetic data, due to the inherent down sampling, noisiness and scale of the patches, we re-trained the model, once on synthetic data and once on a mix of real and synthetic data (thus containing both synthetic/point-cloud and real/point-cloud matches). As shown in the previous section, LCD descriptors actually work well on the synthetic data, so the re-training experiment is only useful to show that the training can converge on the topological dataset.

VI. Future work

Regarding next steps to reach, different adaptations, tests, and experiments are to be considered in order to pursue this project.

Future work concerns variations on patch sizes so as to observe the affect of the scale of the image on the matching, and also performing the matching between a whole scene rather than just a pair of images or point-clouds. Moreover, it would be interesting to focus the study on keypoint detection prior to the description. It could also be interesting to have new proposals to try different methods for data pre-processing, in particular to apply outlier elimination using the RANSAC method which may leads to a more robust matching.

VII. Conclusion

In this work, we trained a 2D-3D learned cross domain descriptor on a topological data set. We showed that these types of descriptors works also on huge real and synthetic data sets, by adapting the parts of code needed in order to achieve this task.

During the project, we faced many challenges that slowed down our progress, including:

- Unavailability of baseline codes: as previously explained, the code for the matching tasks, and even the code for data loading was out of date at the time our project started. This slowed down reproducing the training as well as the matching tasks.
- The wide diversity of formats of topological data: we had to adapt each time to different formats and shapes of the topological data.
- Different view parameters: in the provided EPFL real-data set, some images were distorted.

References

[1] Quang-Hieu Pham, Mikaela Angelina Uy, Binh-Son Hua, Duc Thanh Nguyen, Gemma Roig, and Sai-Kit Yeung. LCD: Learned cross-domain descriptors for 2D-3D matching. In *AAAI Conference on Artificial Intelligence*, 2020.

[2] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[3] Experimental setup for training. https://github.com/hkust-vgd/lcd/issues/4. Accessed: 2020-12-16.

[4] Add original dataloader. https://github.com/hkust-vgd/lcd/commit/1ad5ce9ecb2ca7233f7f9403685e6088108a5a7f. Accessed: 2020-12-16.

[5] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571, 2011.