StickyLocalization: Robust End-To-End Relocalization on Point Clouds using Graph Neural Networks
Supplementary Material

Kai Fischer[1,3] Martin Simon[1,3] Stefan Milz[2,3] Patrick Mäder[3]
[1]Valeo Schalter und Sensoren GmbH [2]Spleenlab GmbH [3]Ilmenau University of Technology
{kai.fischer, martin.simon}@valeo.com {stefan.milz, patrick.maeder}@tu-ilmenau.de

In the course of our investigations we did extensive experiments on hyperparameter finetuneing, as well as comparisons regarding different components of our architecture. The baseline configuration for the following analyses is the parameter setup presented in the main section of the paper, denoting:

- Number of selected points for cloud $\mathcal{P}^{\mathrm{G}}$ and $\mathcal{P}^{\mathrm{L}}$: $n = 500$, $m = 2500$
- $l_{max} = 9$ with $h = 4$ heads per layer
- Pillar shaped descriptors including a maximum of $z = 128$ points within a radius of $d = 0.5m$ based on x, y, z and $intensity$ values

The following sections show comprehensive experiments by respectively alternating one of each of the mentioned parameters and subsequent evaluation in terms of the overall performance of the network. The presented results were achieved by validating the respective network composition on the hard version of the NuScenes test dataset. For each configuration a separate training was performed for 200 epochs following the training strategy mentioned in the main paper. Finally the respective best results for each method within this epoch span are used for our comparisons.

1. Selection and comparison of Hyperparameters

Number of selected points: One of the most influential parameters in the StickyLocalization architecture is the number of selected points m and n for the current frame and the submap respectively. Since the submap is constructed by accumulating multiple frames it is reasonable that a larger amount of points get extracted compared to the current measurement. In this context the ratio between the number of cloud and map points is denominated by the map factor f, where $n = f \cdot m$. A map factor of 1 for instance implies the same amount of points for both frames. We performed several trainings with different cloud and map sizes, for $m = \{250, 500, 1000\}$ and $f = \{1, 2, 5\}$ yielding a total of 9 trainings for the point selection analysis. The respective results for average E_t and E_r on the NuScenes test data (hard) are visualized in figure 1.

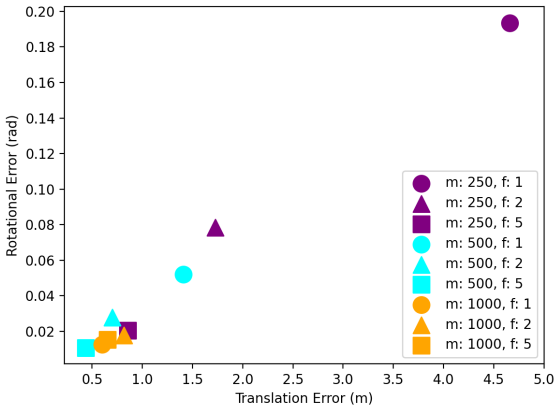

Figure 1: **Point selection analysis** validating the performance of StickyLocalization under varying number of selected points (purple: $m = 250$, cyan: $m = 500$, orange: $m = 1000$) and map factors (circle: $f = 1$, triangle: $f = 2$, square: $f = 5$)

According to the shown results, generally a higher amount of selected points, as well as higher map factors lead to better results concerning the validation metrics. For $m = 250$ the amount of extracted points seems to be insufficient for constructing valid point correspondences due to the sparsity of the clouds. Solely a map factor of 5 in this context yields enough point cloud overlap for the network to predict valid correspondences. In contrast $m = 500$ seems to produce enough density of the resulting filtered clouds to hold enough point matches for pose prediction, leading

to a way better overall performance. Best results among all experiments could be achieved using a parameterization of $m = 500$ and $f = 5$ which also displays a good trade off in terms of runtime efficiency, since a larger amount of points automatically results in a higher inference time of the network. A selected point size of $m = 1000$ shows the average best performance, yielding mean E_t and E_r below a value of $1.0m$ and $0.025rad$ respectively for all map factors. However the large amount of points not only leads to a much higher computation time, but is also disadvantageous with regard to the training process and the performance of the network, since the selected points receive multiple possible match candidates, resulting in less distinctive connections inside the *Graph Neural Network*.

Number of layers inside the *Graph Neural Network*: Another important parameter affecting the performance and runtime of our approach is the number of self-and cross-attention layers inside the *Graph Neural Network*. Here a higher amount of layers implies a deeper network structure, providing more learnable parameters and thus leading to a better generalizability of the network. On the other hand this also involves an increased GPU load impacting the runtime of the network. The performance of StickyLocalization with regard to pose estimation errors and inference runtime under the viewpoint of varying number of layers is displayed in table 1.

Table 1: **Ablation experiments** with different number of self-and cross-attention layers inside the *Graph Neural Network*

	$l_{max} = 3$	$l_{max} = 5$	$l_{max} = 7$	$l_{max} = 9$
E_t (m)	0.816	0.892	0.623	**0.436**
E_r (rad)	0.014	0.019	0.019	**0.011**
t_{inf} (s)	**0.101**	0.117	0.127	0.142

Generally more layers lead to a better performance of the network with respect to the transformation error metrics but simultaneously yielding a higher inference time. However we experienced performance drops in our experiments for layer size of $l_{max} = 5$ which presumably originate from an unfortunate weight initialization. In the course of our main experiments we therefore chose a number of $l_{max} = 9$ layers to ensure the best performance of our architecture. However for time critical applications the number of layers could be adapted to meet certain runtime requirements.

2. Comparison of different descriptor front-ends

Finally we performed experiments investigating the impact of different feature front-ends on the relocalization capability of StickyLocalization. In this context we used two alternative descriptors in comparison to our point pillar feature representation used in SL_{Pillar}:

- SL_{Voxel}: Contrary to the pillar construction with a radius of $d = 0.5m$ and infinite height, in this context we used voxelization for point aggregation with a voxel size of $0.5m$ in x, y and z direction with x, y, z and $intensity$ entries for each point. Subsequent encoding by the pillar- and positional-encoder according to the main section is performed on the voxel representation.
- $SL_{Pointnet}$ Point cloud encoding according to [1] for semantic point cloud segmentation, resulting in a 256 dimensional feature descriptor for each point by skipping the final MLP layers. In this context, point aggregation and pillar-/positional-encoder steps are disregarded and point selection is performed on the PointNet features.

The experimental validation for the alternative feature descriptors as front-end for StickyLocalization are listed in table 2.

Table 2: **Ablation experiments** with alternative feature descriptor front-ends for StickyLocalization

	SL_{Voxel}	$SL_{Pointnet}$	SL_{Pillar}
E_t (m)	0.806	0.584	**0.436**
E_r (rad)	0.015	0.014	**0.011**
t_{inf} (s)	0.142	**0.127**	0.142

Comparing the results of SL_{Voxel} and SL_{Pillar}, shows that the type of point aggregation has a non-negligible effect on the performance of the network. In this context, utilizing pillar shapes offers the opportunity to decode important textures of the local 3D environment like corners and poles, resulting in distinctive feature descriptors for the point cloud registration task.

Applying the PointNet front-end for StickyLocalization $SL_{PointNet}$ also yields decent performance on the relocalization experiments which shows the versatility and robustness of our *Graph Neural Network* middle-end. Although deployment of our pillar descriptors outperforms all compared feature front-ends in terms of transformation errors, a slight improvement of inference runtime for $SL_{PointNet}$ can be observed due to the efficiency of the PointNet architecture.

References

[1] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2