Supplementary Material for
Generalized Shuffled Linear Regression (ICCV 2021)

Feiran Li[1]   Kent Fujiwara[2]   Fumio Okura[1]   Yasuyuki Matsushita[1]

[1]Osaka University   [2] LINE Corporation

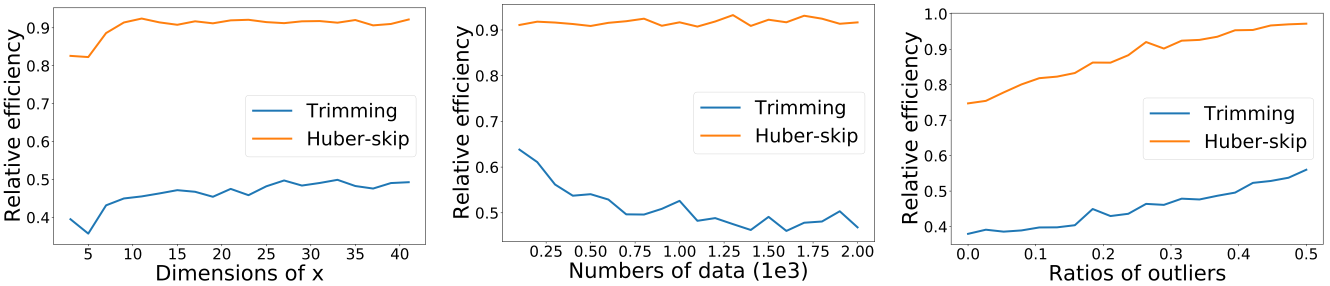

Figure 1: Comparisons of efficiencies between the trimming operator and the Huber-skip estimator under different setups.

1. More analysis of the Huber-skip estimator

We have used the Huber-skip estimator for determining the number of inliers k as described in Sec. 3.3 of the main paper. In this supplementary material, we present more experiments to demonstrate the properties of the adopted Huber-skip estimator.

1.1. Comparisons of efficiency between the trimming operator and the Huber-skip estimator

We conduct Monte Carlo experiments to study the statistical efficiencies of the 50%-trimming operator and the Huber-skip estimator on the least trimmed square problem. We follow the common setup used in [2, 3] to test their respective efficiencies under the following conditions:

- Different dimensions of the regression variable $\mathbf{x}$: We increase the dimension of $\mathbf{x}$ from 3 to 43 with a step size of 2.
- Different numbers of data: We increase the number of data from 100 to 2000 with a step size of 100.
- Different ratios of outliers: We increase the ratio of outliers from 0 to 0.5 with a step size of 0.025.

All the data is randomly generated and corrupted by the Gaussian noise $\mathcal{N}(\mathbf{0}, 0.1\mathbf{I})$. The control variables in each trial are set to 20 for the dimension of the regression variable, 2000 for the number of data, and 0.4 for the ratio of outliers. Each sub-setup (*e.g.*, efficiency w.r.t. a specific dimensional regression variable) is repeated 100 times and their mean value is used as the final result. The results are presented in Fig. 1. As shown, the Huber-skip estimator can achieve significantly higher efficiencies than the trimming operator in all conditions.

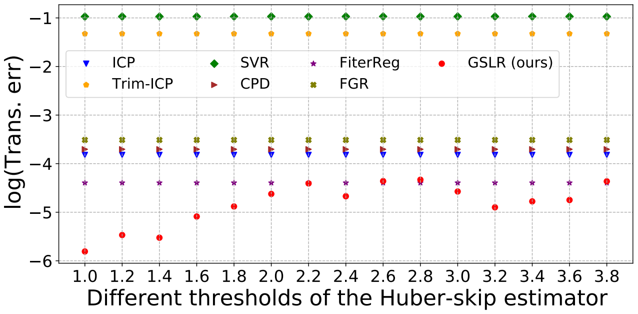

Figure 2: Registration accuracy w.r.t. distinct thresholds of the Huber-skip estimator.

1.2. Threshold of the Huber-skip estimator

In general, the Huber-skip estimator performs stably under varying thresholds since it only affects the statistical efficiency. For demonstration, we conduct tests on the point cloud registration task. The experimental setup is the same as the experiment with different initializations except that the ground-truth rotation is fixed to 15° in Euler angles. The two point clouds are registered with a varying threshold in the range of 1 to 4 with a step size 0.2. The results are

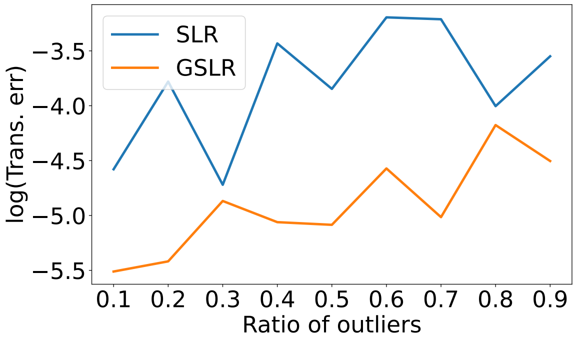

Figure 3: Comparisons between SLR and GSLR on registering outlier-contaminated point clouds.

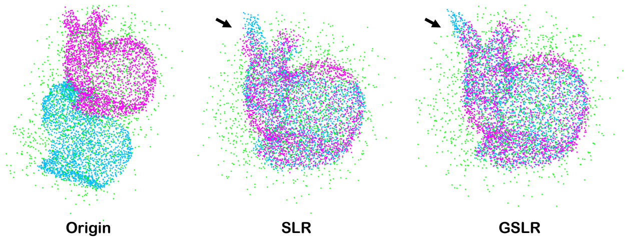

Figure 4: Registration results of SLR and GSLR on outlier-contaminated partially overlapped point clouds. Green points are the outliers.

presented in Fig. 2, where the estimations of competing methods are reported for reference. As shown, GSLR can consistently provide more accurate estimations compared to the others under varying thresholds. Although a smaller threshold may provide slightly more accurate results by rejecting more matches, it may not be a good choice for tasks like shape matching, which desires to keep as many bijective correspondences as possible. Therefore, we consider the threshold of 3.5 suggested in [3] as a reasonable choice for general applications.

2. Complementary experiments and details of experimental setup

In this section, we provide more experiments as well as the details of our experimental setup to complement Sec. 5 of the main paper.

2.1. More comparisons between SLR and GSLR

We carry out experiments on the registration task to further demonstrate the different capacities of SLR and GSLR. The experimental setup is the same as the outlier-contaminated one mentioned in the main paper. Figure 3 reports the results. As demonstrated, GSLR can present more accurate estimations compared to SLR, especially when the ratios of outliers are large. Furthermore, we also conduct a test on registering point clouds that not only contain noise and outliers but also are partially overlapped. As presented in Fig. 4, SLR obviously mis-aligns the point clouds while GSLR can effectively maintain the accuracy.

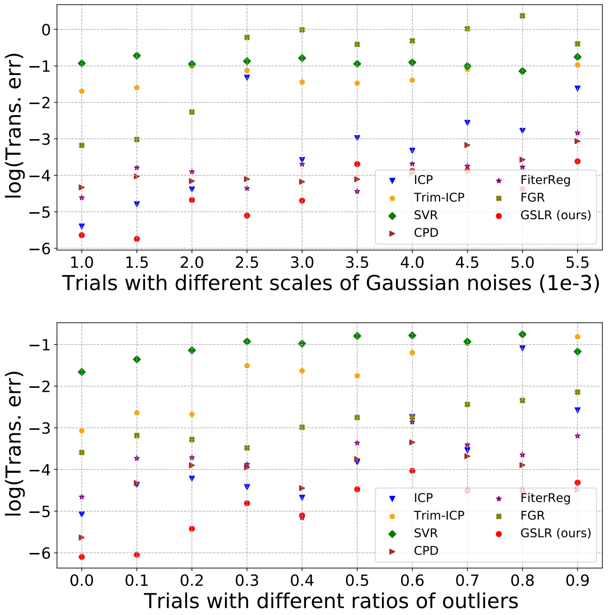

Figure 5: Registration accuracies w.r.t. different scales of noise and ratios of outliers.

2.2. Registration w.r.t. varying scales of noise and ratios of outliers

We have also tested the registration accuracies w.r.t. different scales of noise and ratios of outliers. Both of the experiments are under the same setup as mentioned in Sec. 1.2. As presented in Fig. 5, our GSLR can consistently provide more accurate estimations compared to other approaches.

2.3. More details of the experimental setup

Point cloud registration For each pair of synthetic point clouds used in the outlier-contaminated and distinctly initialized experiments, we perturb them with isotropic Gaussian noises $\mathcal{N}(\mathbf{0}, 1.5\text{e-}3\mathbf{I})$ and $\mathcal{N}(\mathbf{0}, 2.5\text{e-}3\mathbf{I})$. Similarly, for the partial-overlapping tests, we set the scaling of covariances associated to the cropped point clouds to 1e-3, 1.5e-3, 2e-3, and 2.5e-3, respectively. For the competing methods, we follow the suggestion of the CPD paper [4] and set the trimming ratio of Trim-ICP, as well as the weights of the uniform distributions in CPD and FilterReg to 0.5, representing a breakdown point of 0.5. For FGR on the synthetic data, we set the voxel size to the average resolution of the two input point clouds. For the real-world data, we set the voxel size to 0.1 to let FGR perform better, leading to approximately $200k$ points in each point cloud. Regarding the codes used in the experiments, ICP and Trim-ICP are implemented by ourselves in Python, FGR is from the Open3D library [6],

	Mean		Median	
	R	**t**	**R**	**t**
ICP	1.05	4.80e-3	1.04	4.67e-3
Trim-ICP	1.07	3.69e-3	1.09	3.49e-3
SVR	1.05	3.00e-3	0.99	2.67e-3
CPD	1.58e-2	4.22e-4	1.52e-2	4.00e-4
FilterReg	2.19e-2	4.84e-4	1.69e-2	4.99e-4
FGR	8.53e-2	1.57e-3	7.21e-2	1.10e-3
GSLR (ours)	**6.22e-3**	**1.63e-4**	**5.64e-3**	**1.18e-4**

Table 1: Rotational and translational errors on outlier-contaminated point clouds as mentioned in Fig. 3 of the main paper.

	Mean		Median	
	R	**t**	**R**	**t**
ICP	1.69e-1	1.04e-2	1.19e-1	8.43e-3
Trim-ICP	8.56e-1	3.81e-2	8.54e-1	4.83e-2
SVR	2.15e-1	7.14e-3	1.71e-1	4.94e-3
CPD	8.99e-2	5.66e-3	9.06e-2	3.75e-3
FilterReg	1.11e-1	6.22e-3	9.88e-2	3.89e-3
FGR	1.07	3.74e-2	1.12	3.18e-2
GSLR (ours)	**6.49e-2**	**4.01e-3**	**3.76e-2**	**1.93e-3**

Table 2: Rotational and translational errors on partially overlapped point clouds as mentioned in Fig. 3 of the main paper.

	Mean		Median	
	R	**t**	**R**	**t**
ICP	5.15e-2	5.63e-1	1.17e-2	1.65e-1
Trim-ICP	9.22e-2	1.60	7.00e-2	1.54
SVR	1.02e-2	3.01e-1	**2.23e-3**	1.97e-1
CPD	**3.22e-3**	4.97e-1	2.65e-3	1.22e-1
FilterReg	6.54e-2	7.46e-1	6.30e-2	6.58e-1
FGR	6.54e-2	1.21	6.00e-2	1.06
GSLR (ours)	7.05e-3	**1.38e-1**	4.23e-3	**1.12e-1**

Table 3: Rotational and translational errors on real-world point clouds as mentioned in Fig. 5 of the main paper.

	Elapsed time (second)
FMAP	20.48
FMAP+ICP	20.72
FMAP+BCICP	246.08
FMAP+ZO-50	20.53
FMAP+ZO-100	24.53
FMAP+GSLR (ours)	233.74

Table 4: Average elapsed time of the shape matching task.

	Elapsed time (second)
ICP	9.76
Trim-ICP	9.47
SVR	2.01
CPD	25.02
FilterReg	0.31
FGR	0.29
GSLR (ours)	122.61

Table 5: Average elapsed time of the registration task.

and the others are from the *ProbReg* Python library[1].

In the main paper, we have only provided transformational errors due to space limitation. Therefore, we present the detailed rotational and translational results in Tables 1, 2, and 3. For the synthetic data, our GSLR formulation champions both rotational and translational estimations. For the real-world data, since the repeated patterns mainly lie in the parallel arches, FGR fails to present reasonable translations although the estimated rotation is acceptable. Among the other methods, although GSLR presents slightly larger errors on rotation compared to the others, it can provide much more accurate results on translation. Consequently, as shown in the main paper, GSLR outperforms the others in estimating the overall transformations.

Shape matching We set the parameters of the direct operator [5] and the wave kernel signature (WKS) [1] as suggested in their respective papers. For post-processing, every algorithm is iterated 5 times, which we observe as enough to achieve accurate results. All the source codes we use are released by their respective authors.

2.4. Elapsed time of point cloud registration and shape matching

We hereby present the elapsed time on the registration and shape matching tasks. For shape matching, we record the average runtime of the 30 trials on the TOSCA dataset. All the tests are conducted on a desktop with an Intel i9-9900K CPU and 32GB RAM. The results are presented in Table 4. For registration, we record the average runtime of the 60 trials of the differently initialized experiment. The results are shown in Table 5. As shown, for the shape matching task, our GSLR algorithm consumes similar time to the suboptimal BCICP approach. For registration, although our GSLR algorithm consumes more time than the others, we consider this higher runtime as a minor deficiency since there always exist applications that focus more on accuracies and do not require real-time ability. Moreover, as mentioned in the main paper, there already exist significantly faster

[1] ProbReg: `http://probreg.readthedocs.io/`. Last accessed on Jul. 24, 2021.

GPU-accelerated Hungarian algorithms, and applying them to GSLR is just an engineering practice.

References

[1] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *International conference on computer vision workshops (ICCV workshops)*, 2011. 3

[2] Frank Hampel. Robust inference. *Statistics Reference Online*, 2014. 1

[3] Frank R Hampel. The breakdown points of the mean combined with some rejection rules. *Technometrics*, 27(2):95–107, 1985. 1, 2

[4] Andriy Myronenko and Xubo Song. Point set registration: Coherent point drift. *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(12):2262–2275, 2010. 2

[5] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *Transactions on Graphics (TOG)*, 37(6):1–16, 2018. 3

[6] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. 2