ViGG: Robust RGB-D Point Cloud Registration using Visual-Geometric Mutual Guidance

Supplementary Material

A. Experimental Setup

For 3DMatch and ScanNet datasets, we directly use the depth images provided to generate point clouds, thus the mapping between pixels and 3D points is determined by the depth images. For KITTI dataset, we use the provided calibration parameters to determine the approximate mappings between pixels and 3D points. Specifically, we first transform the point clouds into camera coordinate system using the calibrated transformation matrix between camera and LiDAR, and remove the points that are behind the camera. Then, we project the point clouds onto image plane using the camera intrinsics, and remove the points that are located outside the image. The mapping between pixels and 3D points is then determined by nearest neighbor searching on the image plane.

Following previous works[33, 47], we downsample the point cloud with a voxel size of 2.5 cm on indoor datasets 3DMatch and ScanNet, and a voxel size of 30 cm on outdoor dataset KITTI. Besides, the inlier threshold is set to 10 cm for indoor dataets, and 60 cm for outdoor dataset. For the estimated transformation $\mathbf{T}$, the rotation and translation errors are calculated as follows:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}, \tag{1}$$

$$\mathrm{RE} = \mathrm{acos}\frac{\mathrm{trace}(\mathbf{R}^{\mathrm{T}}\bar{\mathbf{R}}) - 1}{2}, \quad \mathrm{TE} = \|\mathbf{t} - \bar{\mathbf{t}}\|_2, \tag{2}$$

where $\bar{\mathbf{R}}$ and $\bar{\mathbf{t}}$ are the ground truth rotation matrices and translation vectors.

It should be noted that for the experiments on KITTI dataset, due to the large number of points in the downsampled point cloud, all the robust estimator methods we compared have adopted a subsampling preprocess to control time and memory costs. After feature extraction, they randomly sample N points from the downsampled point cloud before performing feature matching and transformation estimation. In the papers that provide the preprocess code, SC^2-PCR++[12] randomly samples 8,000 points for evaluation, while VBReg[26] randomly samples 12,000 points for evaluation. However, we observe that this strategy obviously increase the registration errors. This is because randomly sampling a small number of points from a large point cloud increases the distance between points, leading to inaccurate correspondences. As this is not a fair comparison for registration error evaluation, we adjust it to a more appropriate subsampling strategy.

	Rotation(deg)		Translation(cm)		RR
	Acc@0.25/0.5/1	Err	Acc@7.5/15/30	Err	
SC^2-PCR++(default)	44.0 / 79.1 / 95.9	0.28	55.0 / 90.1 / 97.5	7.0	98.4
SC^2-PCR++	55.3 / 85.2 / 96.0	0.23	67.2 / 92.6 / 98.0	5.9	98.6
VBReg(default)	44.3 / 82.3 / 94.6	0.27	59.5 / 89.7 / 96.2	6.6	97.3
VBReg	52.4 / 84.1 / 95.1	0.24	64.5 / 91.4 / 97.1	6.1	97.5

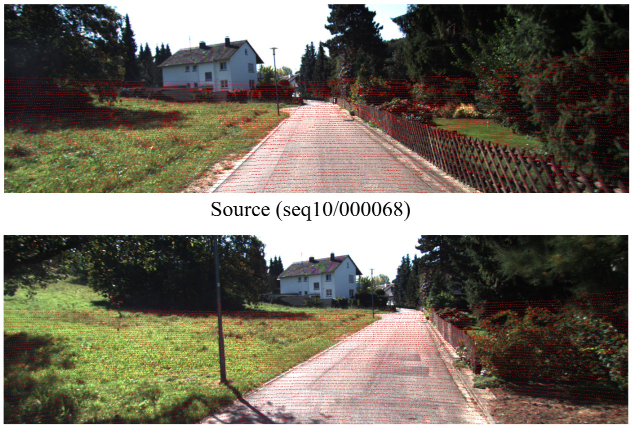
Table S1. Registration performance with different subsampling strategy on KITTI dataset. FPFH is used to extract features, default denotes using the subsampling strategy provided by their authors. We adjust the subsampling strategy of robust estimator methods to a more appropriate one for a fairer and more referable comparison.

In contrast, we use the entire downsampled point cloud for feature matching, and randomly sample 10,000 correspondences for subsequent estimation to control the computational costs. As shown in Tab. S1, our new subsampling strategy does not affect registration recall but achieves lower registration errors. For a fairer and more referable comparison, we apply our new subsampling strategy to all robust estimator methods, and also sample the source points in ViGG's visual-guided geometric matching module to obtain 10,000 correspondences for transformation estimation when evaluating on KITTI dataset.

Furthermore, we observe that using FPFH to provide initial correspondences can achieve better registration performance than FCGF for robust estimator methods on KITTI dataset, which has also been reported in their papers[12, 26, 50]. Therefore, we only report the registration performance using FPFH for robust estimator methods and our ViGG on KITTI dataset.

B. Detailed Analysis About KITTI Dataset

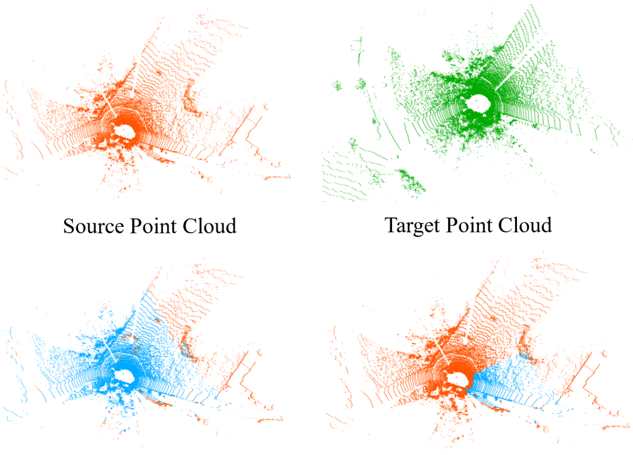
In main paper, we demonstrate that our method achieves significantly lower registration errors compared to other methods. However, our method exhibits slightly lower registration recall. This is primarily because the camera's field of view is much smaller than that of the LiDAR, and as shown in Fig. S1, a quite portion of the image lacks valid mapping to 3D points. Consequently, the valid overlap area between image pairs is small. As shown in Fig. S2, the overlap points within the camera's field of view are much fewer than those in the entire point cloud, leading to less information and fewer potential correspondences, making it unfair for visual matching. KITTI dataset only deploys forward-facing cameras to collect image data, for our proposed ViGG, it is promising that deploying additional cameras in multiple



Source (seq10/000068)

Target (seq10/000056)

Figure S1. 3D points and image pixels are captured by sensors with different ranges, not the entire image is available for registration. We mark the pixels with valid 3D point mapping in red.



Source Point Cloud

Target Point Cloud

Overlap

Overlap in Camera's View

Figure S2. The overlap in the camera's field of view is much smaller than that of the entire point cloud, making registration using visual matches more challenging. We mark the overlapped points of source point cloud in blue.

directions to expand the overall field of view can effectively improve visual matching, thereby achieving better registration recall.

To further demonstrate our method's effectiveness, we evaluate registration performance within a unified field of view. As shown in Fig. S3, we clip the entire point cloud and retain only the points within the camera's field of view, using the clipped point cloud pairs for registration, which provides a fairer evaluation for visual matches. The results in Tab. S2 show that in this camera-LiDAR fair case, our method achieves significant improvements in both registration error and recall, highlighting its effectiveness. Additionally, compared to the results in Tab. 3 of main paper, our method achieves better performance when using the entire point cloud for correspondence extraction. This demonstrates that our method can effectively leverage the points without pixel-point mapping, whereas other RGB-D registration methods can only effectively process points that have valid pixel-point mapping.

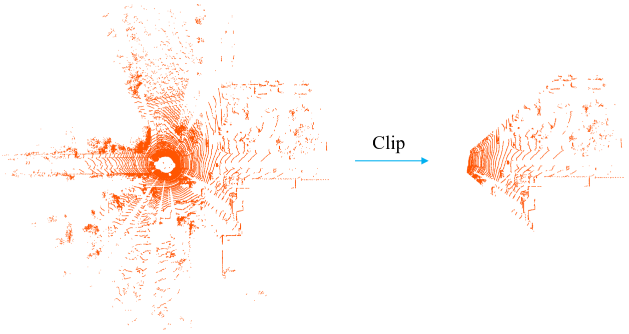


Figure S3. We clip the entire point cloud to retain only the points within the camera's field of view for further experiments.

	Rotation(deg)		Translation(cm)		RR
	Acc@0.25/0.5/1	Err	Acc@7.5/15/30	Err	
learning-free					
FPFH+MAC	14.2 / 44.9 / 72.4	0.56	24.0 / 55.3 / 75.5	13.1	79.6
FPFH+SC^2-PCR++	8.1 / 36.2 / 69.7	0.64	11.0 / 46.8 / 75.0	15.9	80.7
Ours(SIFT+FPFH)	**74.1 / 89.2 / 94.6**	**0.16**	**84.9 / 91.9 / 94.2**	**4.0**	**94.6**
learning-based					
FPFH+VBReg	6.8 / 29.4 / 60.5	0.76	9.5 / 37.1 / 59.3	20.5	64.9
GeoTransformer	44.3 / 74.4 / 89.2	0.28	49.9 / 80.9 / 85.9	7.5	86.3
Ours(LG+FPFH)	**74.8 / 91.4 / 96.2**	**0.16**	**86.5 / 93.7 / 96.4**	**3.9**	**96.9**

Table S2. Registration results on KITTI with unified field of view for camera and LiDAR.

	Rotation(deg)		Translation(cm)		RR
	Acc@2/5/10	Err	Acc@5/10/25	Err	
learning-free					
FPFH+MAC	30.0 / 57.8 / 64.8	3.7	30.6 / 47.0 / 61.6	11.2	60.8
FPFH+SC^2-PCR++	28.6 / 58.2 / 65.4	3.6	27.8 / 47.4 / 61.0	11.2	60.6
Ours(SIFT+FPFH)	**38.8 / 63.2 / 69.4**	**2.8**	**35.2 / 54.8 / 66.4**	**8.5**	**66.4**
learning-based					
FCGF+MAC	31.8 / 64.6 / 72.8	3.1	31.2 / 52.4 / 67.4	9.4	68.0
FCGF+SC^2-PCR++	31.6 / 63.6 / 71.6	3.1	30.8 / 51.6 / 67.0	9.1	67.4
FCGF+VBReg	31.6 / 63.6 / 71.8	3.1	30.8 / 50.4 / 65.0	9.7	67.0
GeoTransformer	38.2 / 68.8 / 75.0	2.7	35.2 / 56.4 / 69.6	7.9	70.6
PEAL	40.8 / 70.8 / 79.0	2.5	36.2 / 59.0 / 73.4	7.3	75.0
Ours(LG+FCGF)	**45.2 / 77.0 / 83.8**	**2.2**	**41.4 / 62.0 / 79.4**	**6.4**	**80.4**

Table S3. Registration results on 3DMatch dataset with a frame spacing of 120.

C. Registration under Larger Frame Spacing

In main paper, we follow the previous studies to use pairs that are 20 frames apart for registration. Additionally,

	10000				5000				2500				1000			
	RE	TE	RR	TC	RE	TE	RR	TC	RE	TE	RR	TC	RE	TE	RR	TC
SC^2-PCR++	0.23	5.9	98.6	278.7	0.26	6.6	98.6	234.0	0.31	7.8	98.4	209.4	0.42	10.4	96.8	159.5
VBReg	0.24	6.1	97.5	811.7	0.27	7.2	97.5	390.6	0.34	8.1	96.4	234.1	0.52	12.0	95.3	181.5
Ours(SIFT)	0.08	3.1	96.8	101.3	0.08	3.2	96.8	89.6	0.10	3.6	96.6	85.6	0.11	4.0	96.2	82.4
Ours(LG)	0.07	3.0	98.9	89.7	0.08	3.1	98.7	77.6	0.10	3.3	98.4	73.6	0.11	3.9	97.7	70.1

Table S4. Registration under different sampling settings on KITTI. TC is the total time cost (ms) of the estimation step after feature extraction.

we use pairs that are 60 frames apart for evaluation under lower overlap. However, we notice that on 3DMatch dataset, 60 frames apart is also easy for our method, making the improvement not obvious. Therefore, we further evaluate registration performance on 3DMatch dataset using pairs that are 120 frames apart, which have extremely low overlap.

As shown in Tab. S3, our method has an obvious improvement in learning-free setting, and also outperforms other methods in learning-based setting. Low overlap brings great challenge for feature matching and point cloud registration. However, by effectively leveraging the advantages of both visual and geometric features, our method can better handle low overlap cases and achieve robust registration.

D. Efficiency Comparison with Robust Estimator Methods

Sampling parts of the correspondences or points for estimation is a commonly used strategy in robust estimator methods, which can reduce the computational cost while bringing only slight accuracy degradation. For registration of large point clouds, sampling process is important. For example, process 10,000 correspondences using VBReg[26] will cost 24 GB GPU memory, making registration computationally expensive.

To further demonstrate our method's efficiency, we evaluate the registration performance and time costs of robust estimator methods and our ViGG under different sampling quantities on KITTI dataset. As mentioned in Appendix A, we randomly sample a subset of correspondences to reduce computational costs. SC^2-PCR++, VBReg and our ViGG are all implemented in Python and run on GPU. As shown in Tab. S4, our method achieve both lowest registration errors and time costs in all sampling settings. Notably, ViGG using 10,000 points has already achieved a shorter time cost than SC^2-PCR++ and VBReg using 1,000 points, indicating that our ViGG is more suitable for handling real-time tasks. Furthermore, our method shows minor accuracy degradation, while SC^2-PCR++ and VBReg both have a larger accuracy degradation, particularly when sampling 1,000 correspondences. This demonstrates that our method can better balance time cost and accuracy. It should be noted that due to the additional image matching process, our method have extra time costs in the feature extraction step compared to geometry-only methods. However, since the image matching process requires minimal computation time (less than 50 ms on KITTI dataset using LightGlue) and can be further sped up by downsampling the image or choosing a more efficient image matching method, our method still has significant advantages in registration efficiency compared to robust estimator methods.

E. Comparison of Distribution-based and Fixed Search Zone

In the visual-guided geometric matching (VGM) module, we propose a distribution-based search zone method to dynamically determine the local search space. This method calculates a radius based on the estimated $\hat{\sigma}^2$, allowing the search zone size to adapt to both the noise level and the scale of the point cloud. Benefited from this, the VGM module can achieve optimal matching performance without requiring careful parameter tuning.

To further demonstrate this, we compare the performance of fixed and distribution-based search zones on the noise version of 3DMatch60 dataset and KITTI dataset. For the fixed search zone, we test three different radius values on each dataset. For the distribution-based search zone, we set $\gamma^2 = 10$ across all experiments. As shown in Tab. S5, the distribution-based search zone achieves optimal performance on both datasets. In contrast, the fixed search zones suffer performance degradation when improperly configured, requiring dataset-specific parameter tuning. Furthermore, we vary the value of γ^2 to our method's sensitivity. As shown in Tab. S6, obvious performance degradation only occurs when γ^2 is low (when set to 2), which may makes the search zone too small and causes the loss of corresponding points. When γ^2 is large, varying its value has almost no effect on registration performance in both two datasets. This indicates that our method is insensitive to the hyper-parameter γ^2.

These results demonstrate the strong generalizability of the distribution-based search zone, which is a significant advantage in practical applications. Notably, when encountering varying noise levels across data, determining suitable search zones with fixed radius becomes impractical, whereas the distribution-based search zone adapts effectively in this case.

	Radius	SIFT			LG		
		RE	TE	RR	RE	TE	RR
N3DM60							
	0.05	1.5	4.5	87.9	1.4	4.2	94.4
fixed	0.1	1.5	4.3	88.4	1.4	4.0	94.4
	0.2	1.6	4.5	88.3	1.5	4.1	94.2
dynamic	-	1.4	4.1	88.4	1.4	3.9	94.4
KITTI							
	0.3	0.08	3.6	96.6	0.08	3.6	98.6
fixed	0.6	0.08	3.1	96.8	0.08	3.0	98.7
	1.2	0.12	3.3	96.9	0.12	3.3	98.7
dynamic	-	0.08	3.1	96.8	0.07	3.0	98.9

Table S5. Registration performance using fixed and distribution-based (dynamic) search zone. Radius is measured in m.

γ^2	SIFT			LG		
	RE	TE	RR	RE	TE	RR
N3DM60						
2	1.5	4.2	87.6	1.4	4.1	94.4
5	1.4	4.1	88.6	1.4	3.9	94.4
10	1.4	4.1	88.4	1.4	3.9	94.4
20	1.4	4.2	88.4	1.4	4.0	94.5
KITTI						
2	0.10	4.1	96.8	0.09	3.6	98.6
5	0.08	3.3	96.8	0.07	3.1	98.9
10	0.08	3.1	96.8	0.07	3.0	98.9
20	0.08	3.1	96.9	0.07	3.0	98.9

Table S6. Registration performance using different values of γ^2.

	Post Method	N3DM60		KITTI	
		RE	TE	RE	TE
	Post Refi	2.8	7.6	0.29	7.3
SIFT	ICP	1.8	6.0	0.15	3.9
	VGM+SVD	1.4	4.1	0.08	3.1
	Post Refi	2.1	5.7	0.24	6.5
LG	ICP	1.7	5.8	0.14	3.9
	VGM+SVD	1.4	3.9	0.07	3.0

Table S7. Post estimation performance using the prior transformation estimated by GVCA. Post Refi denotes applying post-refinement with visual matches. N3DM60 denotes the noise 0.025 version of 3DMatch60.

	Rotation(deg)		Translation(cm)		RR
	Acc@2/5/10	Err	Acc@5/10/25	Err	
20 frames apart					
GeoTransformer	92.3 / 99.0 / 99.5	0.8	84.7 / 96.3 / 99.3	2.3	99.4
ColorPCR	84.0 / 95.9 / 97.6	0.9	77.7 / 92.0 / 96.7	2.7	97.1
60 frames apart					
GeoTransformer	67.0 / 90.7 / 93.9	1.5	57.7 / 80.0 / 91.4	4.1	91.8
ColorPCR	50.1 / 75.9 / 83.0	2.0	45.6 / 66.8 / 78.8	5.7	79.5

Table S8. Registration performance of ColorPCR and GeoTransformer on 3DMatch dataset.

F. Comparison with ICP

ICP[6] is a commonly used registration method when a prior transformation is provided. After clique-based alignment, performing fine registration with ICP using the prior transformation and point cloud pairs is also a feasible approach for RGB-D registration tasks. However, ICP is sensitive to the accuracy of prior transformation and can easily fall into local optima, making it less robust. In contrast, our proposed visual-guided geometric matching (VGM) method can better handle visually noisy cases and achieve significantly more robust and accurate registration.

To make further demonstration, we first use the geometric-guided visual clique alignment (GVCA) module to obtain prior transformation, then evaluate the post estimation performance of post-refinement, ICP and our VGM with SVD. As shown in Tab. S7, our method achieve the best performance. On the KITTI dataset where the prior transformation is relatively accurate, our method shows a clear improvement over ICP. When dealing with less accurate prior transformations on N3DM60 dataset, ICP exhibits even worse accuracy than geometry-only methods, and is significantly worse than our method. In this visually noisy case, the inaccurate prior transformations bring great challenge to ICP. However, our VGM can obtain reliable correspondences and achieve accurate registration even when visual matches are noisy, which is crucial for handling various RGB-D registration tasks.

G. Issues About Comparing with ColorPCR

ColorPCR[30] is the newest RGB-D registration method, which uses color information to enhance GeoTransformer networks and extract more distinctive superpoint features for matching. We reproduce the remarkable performance of ColorPCR on their provided color point cloud datasets Color3DMatch and Color3DLoMatch using the code and pretrained model. However, Color3DMatch and Color3DLoMatch datasets are also 50 franes-merged, which cannot be used for RGB-D registration methods as said in main paper. Therefore, we test ColorPCR on the RGB-D 3DMatch benchmark used in main paper. Unexpectedly, we observe serious performance degradation. As shown in Tab. S8, ColorPCR achieves worse registration performance than GeoTransformer[33].

After checking the code and discussing with the authors, we suppose the issue lies in the input color point cloud data. ColorPCR first reconstructs the entire scene using the color and depth images from original 3DMatch dataset, then applies the scene's color to the merged point cloud for generating color point cloud data. However, we follow PointMBF[48] to process 3DMatch dataset for RGB-D reg-

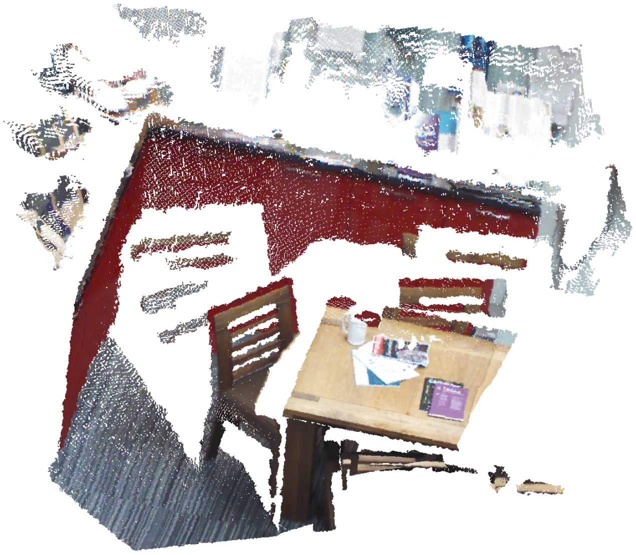


Figure S4. Color point cloud generated from 3DMatch dataset. We generate the color point clouds directly using the color and depth images for ColorPCR.

istration evaluation, the registration is performed on frames that are 20 or 60 frames apart, with only a pair of color and depth images available for each registration process. As shown in Fig. S4, we use a frame of color and depth images to directly generate the color point cloud, which may introduce more color errors and negatively affect the performance of ColorPCR. As a result, the RGB-D 3DMatch benchmark used in our main paper may not be suitable for evaluating ColorPCR, and therefore, we do not include ColorPCR in the main paper's comparison.