Supplementary Materials for OG-UPCR

In this supplementary material, we provide a comprehensive overview of our work. This includes implementation details of OG-UPCR (Section 1), details of OCID (Section 2), metrics (Section 3) and additional experiments (Section 4). These experiments mainly focus on the analysis of the OCID module and the GRPA module. We also provide an analysis of the qualitative experimental results (Section 5).

1. Implementation Details of OG-UPCR

OG-UPCR is trained using Pytorch on a computer with both an Intel Xeon CPU @2.30GHZ and NVIDIA Quadro P6000 GPU. The network undergoes training utilizing the Adam optimizer [14] for a duration of 10 epochs on the 3DMatch dataset and ScanNet dataset. The batch size is set to 4, and the weight decay parameter is established at 10^{-6}. The learning rate is initialized at 10^{-4}. Table 5 shows the implementation details and hyperparameters of OG-UPCR. The parameters related to feature extraction are maintained consistently with those used in LLT [19] and PointMBF [21]. During testing, larger initial correspondences are selected to fully leverage OCID's ability to identify reliable correspondences. The top-k selection method is employed to select w_{thr}, ensuring that the same number of correspondences are used for a fair comparison with other methods. Additionally, the batch size is set to 1 to maintain the invariance of the results. The detailed architecture of the feature extraction is available in PointMBF [21].

Stage	Implementation details	Parameters
Feature extraction	Batch size	4
	Image size	128*128
	Feature dimension	32
	Training epochs	12
	Optimizer	Adam
	Learning rate	1e-4
	Weight decay	1e-6
	Momentum	0.9
	λ_{gs}	0.1
	λ_{corr}	1
Initial correspondences	Number of initial correspondences k (train)	200
	Number of initial correspondences k (test)	2500
OCID	Positive radius τ	0.05
	Overlapping region constraint O_{thr}	0.1
	Neighborhood radius R	0.1
	Number of iterations N	10
	The top-k algorithm for w_{thr} (test)	200
	Number of neighbor correspondences in consensus set K	30

Table 5. The implementation details and hyperparameters of OG-UPCR.

2. Details of OCID

As shown in Fig. 8, we present two examples to represent the principle of random overlapping region prediction. First, a subset is randomly selected from the initial correspondences of the input, and the weighted SVD algorithm [5] is applied to obtain the coarse transformation result. Based on this coarse transformation, the overlap ratio O_p between the point clouds is predicted. If the O_p exceeds the threshold O_{thr}, the subset is considered positive, and the corresponding weights within the subset are increased. Conversely, if the O_p is smaller than O_{thr}, the subset is considered negative, and its weights are penalized. After several iterations of this process, the weights of the initial correspondences are clearly differentiated, making it straightforward to use the top-k algorithm to identify reliable correspondences for geometric fitting. We also provide a pseudocode for random overlapping region prediction, as shown in Algorithm. 1.

3. Metrics

By convention [8, 9, 19, 21], we employ three metrics to evaluate RGB-D point cloud registration: rotation error, translation error and chamfer error.

3.1. Rotation Error

The rotation error is defined as the geodesic distance between the estimated and ground-truth rotation matrices, quantifying the difference between the predicted and ground-truth rotations, which can be expressed as:

$$E_{\text{rotation}} = \arccos\left(\frac{\text{trace}\left(\mathbf{R}_{pr}\mathbf{R}_{gt}^{T}\right) - 1}{2}\right). \tag{1}$$

3.2. Translation Error

The translation error is defined as the Euclidean distance between the estimated and ground-truth translation vectors, quantifying the difference between the predicted and ground-truth translation vectors, which can be expressed as:

$$E_{\text{translation}} = \left\|\mathbf{t}_{pr} - \mathbf{t}_{gt}\right\|_2. \tag{2}$$

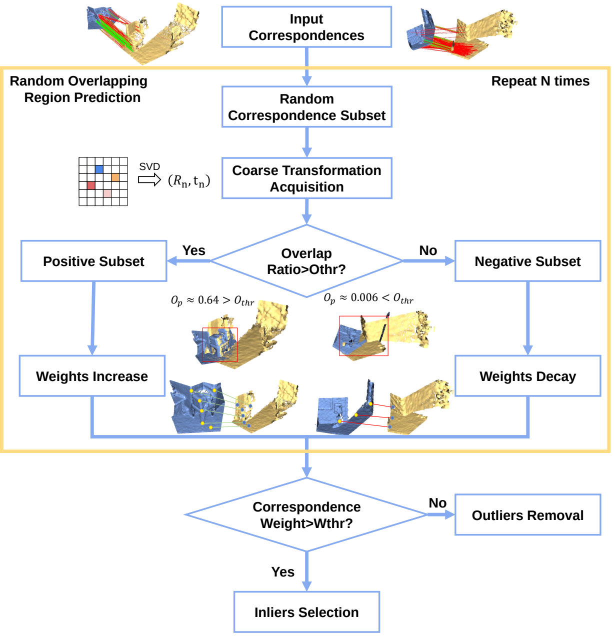

Figure 8. Flowchart of random overlapping region prediction.

3.3. Chamfer Error

The chamfer error is defined as the closest pair of points between the registered point clouds P and Q, evaluating the performance of their registration, which can be expressed as:

$$E_{\text{cham}} = |P|^{-1} \sum_{(p,q)\in\Lambda_{P,Q}} \|x_p - x_q\| + |Q|^{-1} \sum_{(q,p)\in\Lambda_{Q,P}} \|x_q - x_p\|, \tag{3}$$

where $\Lambda_{P,Q} = \left\{ \left(\mathbf{p}, \text{argmin}_{\mathbf{q}\in Q} \|\mathbf{p}-\mathbf{q}\|\right) : \mathbf{p} \in P \right\}$.

4. Additional Experiments

4.1. Comparison of Performance Decay

As illustrated in Fig. 9, we provide a comparison plot that demonstrates the performance decline from the easier ScanNet dataset to the more challenging ScanNet1500. The plot clearly highlights the significant performance degradation of the unsupervised methods LLT [19] and PointMBF [21] in scenarios with low overlap and viewpoint variations. In contrast, our method and the supervised method BUFFER [1] maintain similar performance, proving the advantage of our approach.

4.2. Further Analysis of OCID

To demonstrate the superiority of the proposed OCID, we compare it with similar correspondence optimization strategies, including Weighted Kabsch [9, 12, 19, 21], RANSAC [10], SC2PCR [4], and MAC [22], by replacing OCID with these modules. The experimental results presented in Table 6 show that while SC2PCR and MAC perform well, they are time-consuming. In contrast, our method employs the overlapping constraint to obtain reliable correspondences, resulting in reduced processing time and improved accuracy. Additionally, the proposed OCID can optimize the traditional RANSAC algorithm, enhancing its performance.

Algorithm 1 Random Overlapping Region Prediction

Input:

$$C_{\text{initial}} = \{(\mathbf{p}, \mathbf{q}, w)_m : 0 \leq m < 2k\}$$

Parameter: Positive radius τ, Overlap threshold O_{thr}, Number of iterations N.

Output:

$$C_f = \left\{(\mathbf{p}, \mathbf{q}, w)_f : 0 \leq f < 2k\right\}$$

```
Let $i = 0$, $\tau = 0.05$, $O_{thr} = 0.1$, $N = 10$.
while $i \leq N$ do
    Randomly select a partial subset $C_n \in C_{\text{initial}}$.
    $\mathbf{T}_s$ = Weighted SVD($C_n$).
    Let $O_p$ = compute overlap($P, Q, \mathbf{T}_s, \tau$).
    if $O_p < O_{thr}$ then
        $w_n = w_n \cdot w_p$
    else
        $w_n = w_n \cdot w_r$
    end if
    $i = i + 1$
end while
if $w_n < w_{thr}$ then
    Remove outliers
end if
return $C_f$
```

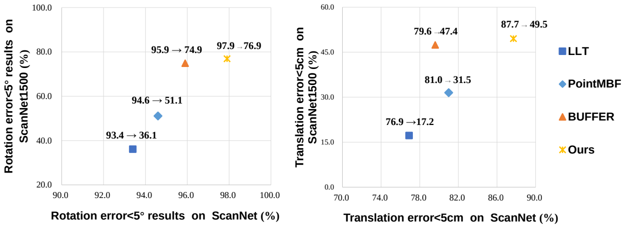

Figure 9. Comparison plot of performance decay. The model is trained on 3DMatch and tested on ScanNet and ScanNet1500. Our method maintains comparable performance with the supervised BUFFER, while other methods experience significant decay.

To further explore the effect of OCID module, we conduct experiments on the real low-overlap point cloud dataset RESSO [3]. As presented in Table 7, integrating the OCID

Method	Rotation(°)		Translation(cm)		Chamfer(mm)		Time(s)
	Accuracy < 5	Error Med	Accuracy < 5	Error Med	Accuracy < 5	Error Med	
Weighted Kabsch	65.1	2.1	42.3	6.8	57.1	0.5	**0.044**
RANSAC	62.9	2.3	37.2	7.5	54.3	0.7	1.262
SC2PCR	79.7	1.6	49.3	5.1	69.9	**0.3**	0.509
MAC	78.1	1.8	48.6	5.5	68.2	0.4	5.291
OCID+RANSAC	79.7	**1.5**	53.7	4.5	71.6	**0.3**	0.995
OCID(Ours)	**81.3**	**1.5**	**53.5**	**4.3**	**72.4**	**0.3**	0.125

Table 6. Comparison of correspondence optimization strategies. The model is trained on ScanNet and tested on ScanNet1500.

module enhances the performance of RANSAC [10] and PLADE [3] in low-overlap scenarios.

Method	6(a)	6(b)	6(c)	6(d)	6(f)	6(g)	6(h)	6(j)	7(a)	7(b)	7(c)
RANSAC	15	16	16	13	50	67	43	0	43	33	**100**
RANSAC+OCID	62	42	47	33	**80**	**78**	57	75	**100**	50	**100**
PLADE	**77**	53	37	**67**	70	**78**	64	50	**100**	**100**	**100**
PLADE+OCID	**77**	**63**	**53**	**67**	**80**	**78**	**79**	**100**	**100**	**100**	**100**

Table 7. Registration results on RESSO.

4.3. Effect of Hyperparameter

Iteration Number N. In order to explore the effect of the proposed hyperparameter iteration number N on the method in the random overlapping region prediction module, we conduct quantitative experiments with different iteration values, as shown in Table 8. The experimental results demonstrate that as the number of iterations N increases, the overall accuracy of the method improves. However, an excessive number of iterations leads to increased computational time. To balance accuracy and efficiency, we select $N = 10$ to optimize the overall model performance.

Iteration	Rotation(°)		Translation(cm)		Chamfer(mm)		Time(s)
	Accuracy < 5	Error Med	Accuracy < 5	Error Med	Accuracy < 1	Error Med	
5	81.1	**1.5**	53.0	4.5	72.0	**0.3**	**0.076**
10	81.3	**1.5**	53.5	**4.3**	72.4	**0.3**	0.125
30	81.3	**1.5**	53.7	**4.3**	73.1	**0.3**	0.321
100	**81.9**	**1.5**	**54.1**	**4.3**	**73.5**	**0.3**	0.928

Table 8. Pairwise registration accuracies and errors under different hyperparameter N. The model is trained on ScanNet and tested on ScanNet1500. The best result is shown in bold.

Overlapping Region Constraint O_{thr}. Following the previous work [3, 11], we further explore the effect of the hyperparameter overlap constraint O_{thr}. As shown in Table 9, the registration performance first increases and then decreases as O_{thr} is changed, and according to this change, we choose 0.10 as the threshold for the overlap constraint.

Reward and Penalty Factors w_r/w_p. To explore the effect of the reward and penalty factors w_r/w_p, we performed a sensitivity analysis. As shown in Table 10, performance improves regardless of the reward and penalty factors value. Optimal performance is achieved when w_r/w_p is set to $2/0.5$.

O_{thr}	Rotation(°)		Translation(cm)		Chamfer(mm)	
	Accuracy < 5	Error Med	Accuracy < 5	Error Med	Accuracy < 1	Error Med
0.05	81.0	**1.5**	53.0	4.4	72.0	**0.3**
0.10	**81.3**	**1.5**	**53.5**	**4.3**	**72.4**	**0.3**
0.15	80.7	**1.5**	53.3	4.5	71.9	**0.3**
0.20	80.3	**1.5**	53.1	4.5	71.4	**0.3**
0.25	80.1	**1.5**	53.0	4.5	71.2	**0.3**
0.30	79.9	1.6	52.4	4.5	70.9	**0.3**

Table 9. Pairwise registration accuracies and errors under different hyperparameter O_{thr}. The model is trained on ScanNet and tested on ScanNet1500. The best result is shown in bold.

Reward Factor	Penalty Factor	Rotation(°)		Translation(cm)		Chamfer(mm)	
		Accuracy <5	Error Med	Accuracy <5	Error Med	Accuracy <1	Error Med
4/3	3/4	79.7	1.6	51.3	4.8	71.5	**0.3**
3/2	2/3	80.7	**1.5**	52.6	4.6	71.5	**0.3**
2	1/2	**81.3**	**1.5**	**53.5**	**4.3**	72.4	**0.3**
3	1/3	80.9	**1.5**	53.4	4.4	72.3	**0.3**

Table 10. Pairwise registration accuracies and errors under different reward and penalty factors w_r/w_p. The model is trained on ScanNet and tested on ScanNet1500. The best result is shown in bold.

4.4. Further Analysis of GRPA

As shown in Fig. 10, traditional differentiable rendering directly projects 3D points into the image space without adaptive optimization, while 3D Gaussian Splatting offers partial adaptability to photometric inconsistencies arising from viewpoint changes. Based on this, we apply an adaptive photometric correction to each pixel post-rendering, making the model more robust to view pairs with large viewpoint variations.

To further explore the effect of GRPA module on our model, we visualize the point cloud feature descriptors before and after its integration. As shown in Fig. 11, the results clearly indicate that after applying the GRPA module, the regions of the scene affected by viewpoint changes exhibit more consistent feature extraction for registration. In contrast, without the GRPA module, the features are significantly more cluttered. These observations highlight the effectiveness of the GRPA module for pairwise registration tasks involving inconsistent photometric information caused by viewpoint variations.

4.5. In-depth Comparison to Sparse Views Methods

We conduct an in-deppth comparison to sparse views methods for pose estimation. However, A direct comparison with methods such as Mast3R [15] is not feasible, as they aim to establish correspondences between 2D pixel fields, whereas our approach focuses on estimating correspondences between point clouds to compute the transformation matrix. For a fair evaluation, a viable solution is to project our results into the image space. As presented in Table 11, our method significantly outperforms them, pri-

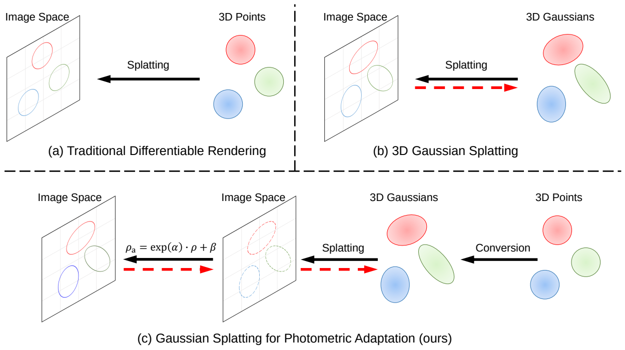
Image Space
3D Points
Splatting
(a) Traditional Differentiable Rendering
Image Space
3D Gaussians
Splatting
(b) 3D Gaussian Splatting
Image Space
Image Space
3D Gaussians
3D Points
$\rho_a = \exp(\alpha) \cdot \rho + \beta$
Splatting
Conversion
(c) Gaussian Splatting for Photometric Adaptation (ours)

Figure 10. (a) Traditional differentiable renderer [16] directly projects 3D points, whereas (b) 3D Gaussian Splatting [13] optimizes its own parameters to adapt to photometric changes during projection. (c) In contrast, the proposed GRPA further optimizes each pixel rendered based on 3D Gaussians [13] to enhance the model's robustness against photometric inconsistencies.

marily because point clouds inherently represent geometric shapes and are less sensitive to textureless regions, aligning with motivation of Mast3R [15].

Method	5°↑	10°↑	20°↑
Dust3R [18]	0.221	0.437	0.636
Mast3R [15]	0.159	0.359	0.573
RoMa [7]	0.270	0.492	0.673
NoPoSplat [20]	0.318	0.538	0.717
Ours	**0.489**	**0.690**	**0.797**

Table 11. Comparison with sparse views methods after projection into the image space on ScanNet1500. The best result is shown in bold.

4.6. Runtime Analysis

We report the mean and standard deviation of the runtime for each stage, comparing it to PointMBF. The correspondence estimation module includes the computation time for both initial correspondence acquisition and correspondence optimization. As shown in Table 12, OG-UPCR only requires approximately $57ms$ more per run compared to PointMBF [21], but the registration performance is significantly improved. Furthermore, the rendering module is 363.5% faster than traditional differentiable renderer.

Stage	Time(ms)	
	PointMBF	Ours
Feature extraction	19.76±31.57	19.07±32.55
Correspondence estimation	24.02±10.00	64.86±17.70
Geometric fitting	31.29±17.03	51.13±45.73
Rendering	3.49±8.53	0.96±0.15

Table 12. Runtime analysis of each stage compared to PointMBF.

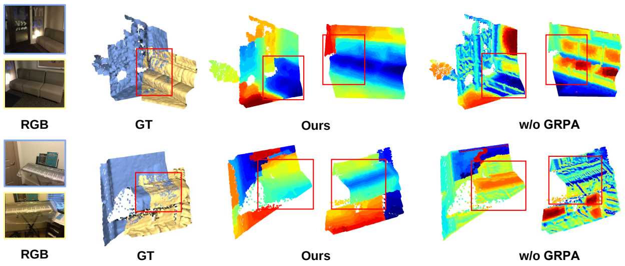

Figure 11. The red-boxed areas represent the overlapping regions. Due to changes in perspective, the photometric information in these areas becomes inconsistent, complicating the registration. The proposed GRPA module mitigates the effects of viewpoint variation, thereby helping the model extract more consistent features for registration. The features are visualized by mapping them to colors by t-SNE [2].

4.7. Computational Footprint

The GPU-memory statistics for each stage are presented in Table 13. The computational footprint is small due to only two frames to process.

Stage	Memory (MB)
Feature extraction	2985.98
Correspondence estimation	19.05
Geometric fitting	0.30
Gaussian Rendering	213.89

Table 13. GPU-memory statistics for each stage (batchsize=1).

5. Additional Qualitative Results

We classify the scenes into two categories: low-overlap scenes and illumination-variation scenes. Sufficient qualitative results are analyzed for both scenarios separately to fully demonstrate the effectiveness of our method.

As illustrated in Fig. 12, in the low-overlap scenes, the correspondences obtained by PointMBF contain a large number of outliers, leading to registration failure. In contrast, our method relies on the overlap constraint to obtain more accurate correspondences, resulting in successful registration.

Similarly, as shown in Fig. 13, when viewpoint changes drastically across different viewing angles, our method demonstrates greater robustness compared to PointMBF, yielding more inliers and ultimately achieving successful registration.

6. Limitations

Although the proposed OG-UPCR achieves superior performance in challenging scenarios, it still has certain limitations. First, it relies on KPconv [17] for point cloud feature extraction, which makes it vulnerable to large-scale rotation interference, while ensuring rotation equivariance and translation invariance. To address this issue, we plan to replace equivariant backbones [6, 23] in future work to further improve the robustness of the overall registration model.

References

[1] Sheng Ao, Qingyong Hu, Hanyun Wang, Kai Xu, and Yulan Guo. Buffer: Balancing accuracy, efficiency, and generalizability in point cloud registration. In *CVPR*, pages 1255–1264, 2023. 2

[2] Guikun Chen and Wenguan Wang. A survey on 3D gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024. 4

[3] Songlin Chen, Liangliang Nan, Renbo Xia, Jibin Zhao, and Peter Wonka. Plade: A plane-based descriptor for point cloud registration with small overlap. *IEEE TGRS*, 58(4): 2530–2540, 2019. 2, 3

[4] Zhi Chen, Kun Sun, Fan Yang, and Wenbing Tao. Sc2-pcr: A second order spatial compatibility for efficient and robust point cloud registration. In *CVPR*, pages 13221–13231, 2022. 2

[5] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration. In *CVPR*, pages 2514–2523, 2020. 1

[6] Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J Guibas. Vector neurons: A general framework for so (3)-equivariant networks. In *ICCV*, pages 12200–12209, 2021. 5

[7] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *CVPR*, pages 19790–19800, 2024. 4

[8] Mohamed El Banani and Justin Johnson. Bootstrap your own correspondences. In *ICCV*, pages 6433–6442, 2021. 1

[9] Mohamed El Banani, Luya Gao, and Justin Johnson. Unsupervisedr&r: Unsupervised point cloud registration via differentiable rendering. In *CVPR*, pages 7129–7139, 2021. 1, 2

[10] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2, 3

[11] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. Predator: Registration of 3D point clouds with low overlap. In *CVPR*, pages 4267–4276, 2021. 3

[12] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976. 2

[13] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4):1–14, 2023. 4

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 1

[15] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3D with mast3r. In *ECCV*, pages 71–91. Springer, 2024. 3, 4

[16] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3D deep learning with pytorch3d. *ArXiv preprint arXiv:2007.08501*, 2020. 4

[17] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, pages 6411–6420, 2019. 4

[18] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3D vision made easy. In *CVPR*, pages 20697–20709, 2024. 4

[19] Ziming Wang, Xiaoliang Huo, Zhenghao Chen, Jing Zhang, Lu Sheng, and Dong Xu. Improving RGB-D point cloud registration by learning multi-scale local linear transformation. In *ECCV*, pages 175–191. Springer, 2022. 1, 2

[20] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D gaussian splats from sparse unposed images. *arXiv preprint arXiv:2410.24207*, 2024. 4

[21] Mingzhi Yuan, Kexue Fu, Zhihao Li, Yucong Meng, and Manning Wang. Pointmbf: A multi-scale bidirectional fusion network for unsupervised RGB-D point cloud registration. In *ICCV*, pages 17694–17705, 2023. 1, 2, 4

[22] Xiyu Zhang, Jiaqi Yang, Shikun Zhang, and Yanning Zhang. 3D registration with maximal cliques. In *CVPR*, pages 17745–17754, 2023. 2

[23] Minghan Zhu, Maani Ghaffari, William A Clark, and Huei Peng. E2pn: Efficient se (3)-equivariant point network. In *CVPR*, pages 1223–1232, 2023. 5

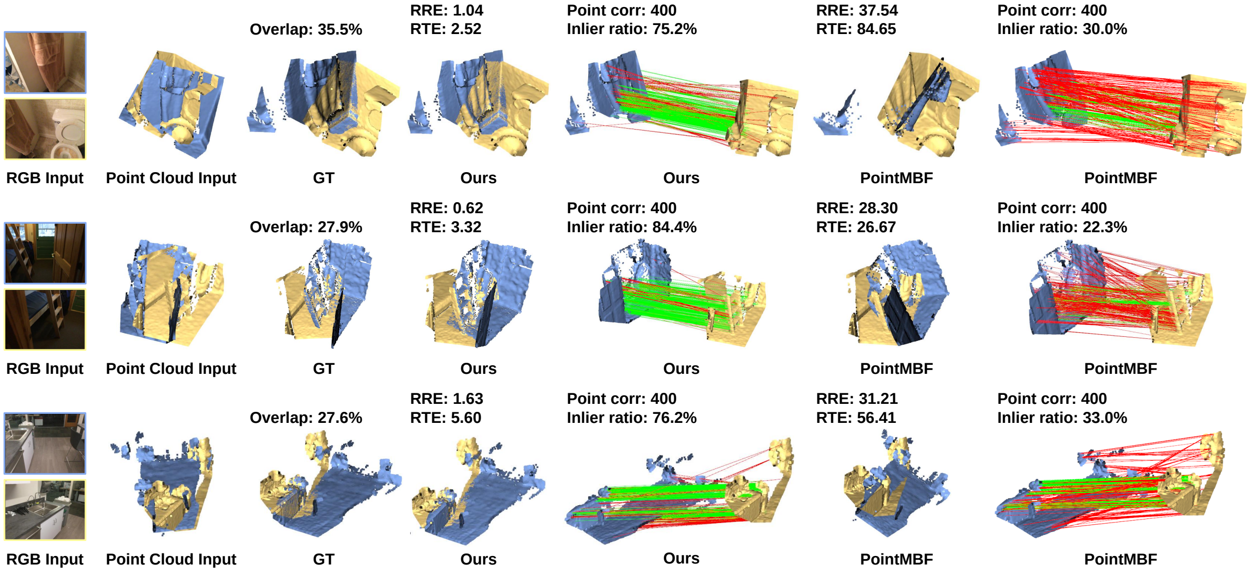

Figure 12. Qualitative results compared with PointMBF in the low-overlap scenario. The red lines denote the outliers, while the green lines denote the inliers.

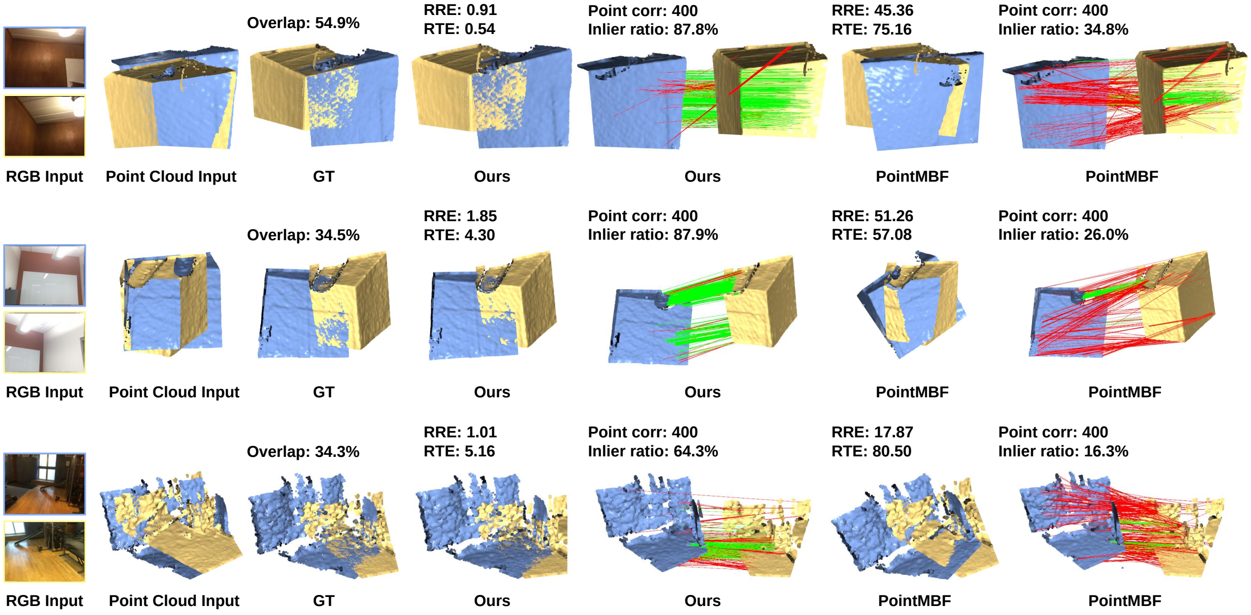

Figure 13. Qualitative results compared with PointMBF in the viewpoint variation scenario. The red lines denote the outliers, while the green lines denote the inliers.