A. Appendix Section

A.1. Graph Signal Processing theory

Graph Shift A graph G can be represented in the form of $(\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes $\{v_0, v_1, \ldots, v_{N-1}\}, N = |\mathcal{V}|$ and $\mathcal{A} \in \mathbb{C}^{N\times N}$ is the graph shift, or the weighted adjacency matrix. A graph shift can reflect the connections of the graph for the edge weight is a quantitative representation between nodes. When a graph shift acts on a graph signal, it can represent the diffusion of the graph signal. A graph shift is usually normalized for proper scaling, ensuring $||\mathcal{A}||_{spec} = 1$.

Graph Signal Given a graph $G = (\mathcal{V}, \mathcal{A})$, a graph signal on this graph can be seen as a map assigning each node v_i with a value $x_i \in \mathbb{C}$. If the order of the nodes is fixed, then the graph signal is defined as a N dimensional vector $x = (x_1, x_2, \ldots, x_N)$.

Graph Fourier Transform In general, a fourier transform is the expansion of a signal on a set of bases. When performing graph fourier tansform, the signal refers to the graph signal and the bases are the eigenbasis of the graph shift, or the Jordan eigenbasis if a complete eigenbasis is not available. To be specific, a graph shift $\mathcal{A}$ is eig-decomposed into $\mathcal{A} = V\Lambda V^{-1}$. The eigenvalues represent the frequencies on the graph. The fourier transform of a graph signal x, therefore, is defined as $\hat{x} = V^{-1}x$ and the inverse transform is $x = V\hat{x}$. $\hat{x}$ describes the content of different frequency components in the graph signal x.

Graph Filtering A graph filter is a type of system that accepts a graph signal as input and then generates another graph signal as output. If the graph signal is described as $x \in \mathbb{C}^n$, then any matrix $A \in \mathbb{C}^{n\times n}$ can be seen as a graph filter and $Ax \in \mathbb{C}^n$ is the output signal. For instance, a graph shift can be a graph filter, replacing the signal value at a node with a weighted linear combination of values at its neighbors. In fact, every linear, shift-invariant graph filter can be formulated as a polynomial in the graph shift

$$\mathcal{H} = \sum_{l=0}^{l=L-1} h_l \mathcal{A}_l$$

where $\mathcal{H}$ is the graph filter represented with the graph shift $\mathcal{A}$. h_l is the coefficient and L is the length of the graph filter.

A.2. Problem Formulation

For two input point clouds, P^s and P^t, a correspondence set C is created using either hand-crafted or learned descriptors. Each correspondence, represented by $c \in C$, consists of a pair of points (p^s, p^t), where p^s and p^t are points in P^s and P^t respectively. These correspondences can be modelled as a graph, G, where each node denotes a correspondence, and the edge weights measure the compatibility between nodes. Our approach aims to decrease the size of the correspondence graph G and employ the sampled correspondences for computing the 6-DoF pose transformation between P^s and P^t, instead of utilising the original correspondence set C.

A.3. Correspondence Generation

Though our focus is not primarily on the creation of correspondences, it is crucial to understand how they are generated due to our approach's reliance on them. As such, we include this section. Point cloud descriptors aim to characterise local geometry. Hand-crafted designs[51] or deep learning methods[17][1][69][30] have been employed in previous studies to create descriptors. As long as pointwise descriptors are defined, the matchability score can be calculated to form correspondences.

We denote the descriptor for point x_i as f_i. Then the matchability score is defined using L_2 euclidean distance:

$$d(x_i, x_j) = ||f_i - f_j||_2$$

Then for each point x_i, find the corresponding point x_{k_i} with the highest matchability score, namely the nearest neighbor. Now a correspondence is generated:

$$c_i = (x_i, x_{k_i})$$

A.4. Graph Construction

As show in Fig. 2, given a set of input correspondences C, we first construct a compatibility graph. Each node of the graph is a correspondence denoted as $c_i = (x_i, y_i, z_i, u_i, v_i, w_i)$. We further denote $p_i^s = (x_i, y_i, z_i)$ and $p_i^t = (u_i, y_i, z_i)$. As can be seen, c_i is tuple of six elements, composed of coordinates of the source point p_i^s and the target point p_i^t. We first define the *distance* between correspondences as $S_{dist}(c_i, c_j) = \left| ||p_i^s - p_j^s|| - ||p_i^t - p_j^t|| \right|$. With this, we compute the pair-wise compatibility between correspondences, or the edge weight in the correspondence graph.

$$W_{ij} = \begin{cases} 1 - \frac{S_{dist}(c_i,c_j)^2}{2\times d_{cmp}^2} & 1 - \frac{S_{dist}(c_i,c_j)^2}{2\times d_{cmp}^2} > t \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

where d_{cmp} and t are hyperparameters. We then define the adjacency matrix W_{SOG} of the graph as: $W_{SOG} = W \odot (W \times W)$.In this way, we build a graph on the given correspondence set C. And we denote this graph as G_{corr}.

A.5. Optimality Proof

We claim our stochastic method is an approximation to an optimal sample operator. To prove this, we first define the object function

$$\min_{\pi} \mathbb{E}_{\Psi\sim\pi}||S\Psi^T\Psi f(X) - f(X)||_2^2 \tag{8}$$

where Ψ is the sample operator relying on π and $S\Psi^T$ is the interpolation recovery operator. f is the LapLacian high-pass filter we use and X is our generalized degree signal. The optimization problem is then formulated as

$$\begin{aligned} &\min_{\pi}\mathbb{E}_{\Psi\sim\pi}||S\Psi^T\Psi f(X) - f(X)||_2^2 \\ &\text{s.t.} \sum \pi_i = 1, \pi > 0 \end{aligned} \tag{9}$$

To solve this, we use a Lagrange function

$$\begin{aligned} &L(\pi_i, \lambda, \mu) \\ =&\mathbb{E}_{\Psi\sim\pi}||S\Psi^T\Psi f(X) - f(X)||_2^2 \\ &+ \lambda(\sum \pi_i - 1) + \sum \mu_i\pi_i \\ =&||\mathbb{E}_{\Psi\sim\pi}(S\Psi^T\Psi f(X)) - f(X)||_2^2 + \\ &\mathbb{E}_{\Psi\sim\pi}||S\Psi^T\Psi f(X) - \mathbb{E}_{\Psi\sim\pi}(S\Psi^T\Psi f(X))||_2^2 + \\ &+ \lambda(\sum \pi_i - 1) + \sum \mu_i\pi_i \end{aligned} \tag{10}$$

The first item is zero, proved simply by

$$\begin{aligned} &\mathbb{E}_{\Psi\sim\pi}(S\Psi^T\Psi f(X))_i \\ =&\mathbb{E}_{\mathcal{M}}(\sum_{\mathcal{M}_j\in\mathcal{M}} S_{\mathcal{M}_j\mathcal{M}_j} f_{\mathcal{M}_j}(X)\delta_{\mathcal{M}_j,i}) \\ =&M\sum_k \frac{1}{M\pi_k} f_k(X)\pi_k\delta_{k,i} \\ =&f_i(X) \end{aligned} \tag{11}$$

where $\mathcal{M}$ is the sample set with M to be its size.

The second item can be deduced to

$$\begin{aligned} &\mathbb{E}_{\Psi\sim\pi}||(S\Psi^T\Psi f(X))_i - \mathbb{E}_{\Psi\sim\pi}(S\Psi^T\Psi f(X))_i||_2^2 \\ =&\mathbb{E}_{\mathcal{M}}(\sum_{\mathcal{M}_j,\mathcal{M}_{j'}\in\mathcal{M}} S_{\mathcal{M}_j\mathcal{M}_j} S_{\mathcal{M}_{j'}\mathcal{M}_{j'}} f_{\mathcal{M}_j}(X)^T f_{\mathcal{M}_{j'}}(X) \\ &\delta_{\mathcal{M}_j,i}\delta_{\mathcal{M}_{j'},i}) \\ =&M^2\sum_k \frac{f_k(X)^T f_k(X)}{M^2\pi_k^2}\pi_k\delta_{k,i} - f_i(X)^T f_i(X) \\ =&(\frac{1}{\pi_i} - 1) f_i(X)^T f_i(X) \end{aligned} \tag{12}$$

Therefore, the Lagrange function can be written as

$$\sum_i (\frac{1}{\pi_i} - 1)||f_i(X)||_2^2 + +\lambda(\sum \pi_i - 1) + \sum \mu_i\pi_i \tag{13}$$

By setting its derivative to zero and with the complementary slackness, we have the final result.

$$\mu_i\pi_i = 0, \pi_i = \frac{||f_i(X)||_2}{\sqrt{\lambda + \mu_i}} \tag{14}$$

A.6. Experimental Setup

Datasets. We consider three main datasets, i.e, the outdoor dataset KITTI[26], the indoor dataset 3DMatch[70] and its low-overlap version 3DLoMatch[30]. For KITTI, we follow the preprocess schedule of previous work[2] [13][71] and obtain a test set of 555 pairs of point clouds. 3DMatch is a scene-scale indoor dataset and 3DLoMatch is its subset with overlap rate ranges from 10% to 30%, bringing greater challenges for accurate registration.

Evaluation Criteria. We follow the common evaluation criteria in 3D registration, i.e, the rotation error (RE), the translation error (TE) and the recall or success rate (RR). RE measures the angular difference between the estimated rotation matrix and the ground truth or reference rotation matrix. TE is computed as the euclidean distance between the estimated translation vector and the ground-truth, and is given in centimetres. By referring to the settings in [18], registration is considered successful when RE$\leq$ 15° and TE$\leq$ 30cm on 3DMatch and 3DLoMatch datasets, and RE$\leq$ 5° and TE$\leq$ 60cm on the KITTI dataset. RR is then defined as the success ratio of all point cloud pairs.

Implementation Details. Our FastMAC consists of a sampling process implemented in PyTorch for cuda computation and then the original Maximal Clique Registration[71] process based on C++. Our method accepts initial correspondences as input, which are generated using Fast Point Features Histograms (FPFH) [51] and Fully Convolutional Geometric Features (FCGF) [17] as basic descriptors for both KITTI and 3DMatch&3DloMatch. Hyperparameters like d_{cmp} and t mentioned in Sec. 3 are set to default values, i.e, 0.1 and 0.999 respectively. The Maximal Clique Registration process remains exactly the same implemented in [71], with the same parameter value settings. All experiments were implemented with an Intel i5-13600KF CPU and a single NVIDIA RTX4070ti. When comparing with baseline methods, we use their default parameters in their released code to ensure fairness.

A.7. Additional Experiments

The following table demonstrates the extension result of our Time-performance Trade-off experiments. Results for RE and TE are further shown. FastMAC performs better than all previous methods when sample ratio is 50%. With lower sample rate, RR and TE still remain competitive, with time loss decreasing dramatically, indicating its potential for real-time application.

Methods	ratio	RR(%)	RE (°)	TE(cm)	Time(ms)
RANSAC 1K[23]		56.58	1.74	33.12	19.9
RANSAC 10K[23]		88.47	1.15	25.33	158.4
RANSAC 100K[23]		94.77	0.77	17.83	1549.7
DGR[18]		95.14	0.43	23.28	330.1
PointDSC[2]	100	96.40	0.61	13.42	131.0
PointDSC(50k)[2]		96.40	0.51	11.53	722.4
PointDSC(icp)[2]		96.76	0.51	11.20	130.5
SC2-PCR[13]		97.12	0.41	9.71	851.1
MAC[71]		97.25	0.36	8.00	573.0
FastMAC	50	**97.84**	**0.36**	**7.98**	**114.4**
	20	**97.48**	**0.38**	8.20	**28.1**
	10	**97.30**	0.43	8.70	**18.2**
	5	97.12	0.52	9.92	**16.1**
	1	71.56	1.02	15.24	**15.5**

Table 6. Comparison on RR, RE and TE with SOTA methods on the KITTI dataset with FCGF descriptor. The best results are marked in bold. $RANSAC - k$ denotes RANSAC with k iterations. PointDSC(50K): PointDSC with 50k iterations of RANSAC. PointDSC(icp): with icp refinement.