ABSTRACT
The prevalence of accessible depth sensing and 3D laser scanning techniques has enabled the convenient acquisition of 3D dynamic point clouds, which provide efficient representation of arbitrarily-shaped objects in motion. Nevertheless, dynamic point clouds are often perturbed by noise due to hardware, software or other causes. While many methods have been proposed for the denoising of static point clouds, dynamic point cloud denoising has not been studied in the literature yet. Hence, we address this problem based on the proposed spatio-temporal graph modeling, exploiting both the intra-frame similarity and inter-frame consistency. Specifically, we first represent a point cloud sequence on graphs and model it via spatio-temporal Gaussian Markov Random Fields on defined patches. Then for each target patch, we pose a Maximum a Posteriori estimation, and propose the corresponding likelihood and prior functions via spectral graph theory, leveraging its similar patches within the same frame and corresponding patch in the previous frame. This leads to our problem formulation, which jointly optimizes the underlying dynamic point cloud and spatio-temporal graph. Finally, we propose an efficient algorithm for patch construction, similar/corresponding patch search, intra- and inter-frame graph construction, and the optimization of our problem formulation via alternating minimization. Experimental results show that the proposed method outperforms frame-by-frame denoising from state-of-the-art static point cloud denoising approaches.

CCS CONCEPTS
• Computing methodologies → Point-based models; Maximum a posteriori modeling; • Mathematics of computing → Graph algorithms.

KEYWORDS
Dynamic point clouds, denoising, spatio-temporal GMRF modeling, spectral graph theory

1 INTRODUCTION
The maturity of depth sensing and 3D laser scanning techniques has enabled convenient acquisition of 3D dynamic point clouds, a natural representation for arbitrarily-shaped objects varying over time [23]. A dynamic point cloud consists of a sequence of static point clouds, each of which is composed of a set of points defined on irregular grids, as shown in Fig. 1. Each point has geometry information (i.e., 3D coordinates) and possibly attribute information such as color. We focus on the geometry of point clouds in this paper due to its vital role. Because of the efficient representation, dynamic point clouds have been widely deployed in various fields, such as 3D immersive tele-presence, navigation for autonomous vehicles, gaming and animation [30].

Point clouds are often perturbed by noise, which comes from hardware, software or other causes. Hardware wise, noise occurs due to the inherent limitations of the acquisition equipment. Software wise, in the case of generating point clouds with existing algorithms, points may locate somewhere completely wrong due to imprecise triangulation (e.g., a false epipolar matching). The noise corruption directly affects the subsequent applications of dynamic point clouds.

However, the denoising of dynamic point clouds hasn’t been studied in the literature yet, while many approaches have been proposed for static point cloud denoising. Existing denoising methods for static point clouds mainly include moving least squares (MLS)-based methods, locally optimal projection (LOP)-based methods, sparsity-based methods, and non-local similarity-based methods. MLS-based methods [1, 12, 20] approximate a smooth surface for the input point clouds and project the points to the estimated surface.
LOP-based methods [14, 16, 17] also apply surface approximation but the operator is non-parametric. Sparsity-based methods [3, 18] assume sparse representation of point normals, and solve the global minimization problem to obtain the sparse reconstruction of the point normals. Non-local similarity-based methods [8, 32] exploit self-similarities among surface patches in a point cloud. Besides, several other approaches have been proposed for static point cloud denoising [11, 15, 24, 31], in which the key idea is to detect noise in point clouds via certain characteristics and then delete them.

Whereas it is possible to apply existing static point cloud denoising methods to each frame of a dynamic point cloud sequence separately, the inter-frame correlation would be neglected, which may lead to inconsistent denoising results in the temporal domain. Hence, we propose joint denoising of dynamic point clouds by exploiting the inter-frame correlation, which not only enforces the temporal consistency but also provides additional information for denoising. Since point clouds are irregular, it is challenging to acquire the temporal correspondence between neighboring frames. We address this issue by representing dynamic point clouds naturally on graphs, where each vertex represents a point, each edge captures the relationship between neighboring points, and the corresponding graph signal refers to the coordinates of points. We then propose a graph-based method to search the temporal correspondence and estimate the underlying clean dynamic point cloud.

Specifically, since it is computationally inefficient to consume an entire frame of point cloud, we first divide each frame into overlapping patches. Each irregular patch is defined as a local point set consisting of a centering point and its k-nearest neighbors. Then we propose a spatial-temporal model under Gaussian Markov Random Fields (GMRF) [22], which play a crucial role in describing both the intra-frame and inter-frame correlations over patches. Next, we estimate the underlying current frame via Maximum a Posteriori (MAP) estimation, given the previous and current noisy frames. We propose the likelihood function and prior distribution, based on the GMRF modeling and graph-signal smoothness prior [26]. This leads to the proposed problem formulation of dynamic point cloud denoising, where the underlying frame and its graph representation (the graph Laplacian in particular) are jointly optimized.

Based on the above problem formulation, we propose an efficient algorithm to address the denoising problem of dynamic point clouds. For each target patch in the current frame, we first search for its similar patches in the same frame to exploit the intra-frame correlation, and search for its corresponding patch in the previous frame to explore the inter-frame correlation. Similar to [32], the similarity metric between two patches depends on the distance from each point in the two patches to the tangent plane at each patch center of both patches. Based on the similar patches and corresponding patch, we address the problem formulation by designing an efficient alternating minimization algorithm to solve the underlying frame and graph Laplacian alternately. In particular, since the computational complexity of solving the graph Laplacian would be high and the numerical computation might be unstable, we propose to construct the intra-frame graph and inter-frame graph based on the patch similarity manually from each update of the underlying frame. Experimental results show that the proposed method outperforms separate denoising of each frame from state-of-the-art static point cloud denoising methods on five widely used dynamic point cloud sequences.

In summary, the main contributions of our work include:

• To the best of our knowledge, we are the first to address dynamic point cloud denoising problem in the literature. The key idea is to exploit the inter-frame correlation of irregular point clouds for the temporal consistency.
• We propose a spatial-temporal model of dynamic point clouds under GMRF, and derive the MAP estimation from graph-signal priors, which finally casts dynamic point cloud denoising as an optimization problem.
• We propose an efficient algorithm to solve the optimization problem. Experimental results validate the effectiveness of our method.

2 RELATED WORK

To the best of our knowledge, there has been no research on dynamic point cloud denoising yet in the literature. Previous works on static point cloud denoising can be divided into four classes: moving least squares (MLS)-based methods, locally optimal projection (LOP)-based methods, sparsity-based methods, and non-local methods.

MLS-based methods. MLS-based methods aim to approximate a smooth surface from the input point cloud and minimize the geometric error of the approximation. Alexa et al. obtain a polynomial function on a local reference domain to best fit neighboring points in terms of MLS [1]. Other similar solutions are algebraic point set surfaces (APSS) [12] and robust implicit MLS (RIMLS) [20]. However, the results may be over-smoothing and may not perform well in terms of removing outliers.

LOP-based methods. LOP-based methods also apply surface approximation for denoising point clouds. But unlike MLS-based methods, the operator is non-parametric, thus it performs well in cases of ambiguous orientation. For example, Lipman et al. define a set of points that represent the estimated surface by minimizing the sum of Euclidean distances to the data points [17]. The two branches of [17] are weighted LOP (WLOP) [16] and anisotropic LOP (WLLOP) [14], [16] produces a set of denoised, outlier-free and more evenly distributed particles over the original dense point cloud to keep the sample distance of neighboring points. [14] modifies WLOP with an anisotropic weighting function so as to preserve sharp features better. However, LOP-based methods may also lead to over-smoothing results.

Sparsity-based methods. Sparsity-based methods are based on the theory of sparse representation of the point normals. With sparsity regularization, they solve a global minimization problem to obtain sparse reconstruction of the point normals. Then the positions of points are updated by solving another global minimization problem based on a local planar assumption, such as [18] and [3]. However, when locally high noise-to-signal ratios yield redundant features, these methods may not perform well and lead to over-smoothing or over-sharpening [29].

Non-local methods. Non-local methods exploit self-similarities among surface patches in a point cloud. These methods are inspired
We represent dynamic point clouds on undirected graphs. An undirected graph $G = (V, E)$ is composed of a vertex set $V$ of cardinality $|V| = n$, an edge set $E$ connecting vertices, and a weighted adjacency matrix $A$. $A \in \mathbb{R}^{n \times n}$ is a real and symmetric matrix, where $a_{i,j} \geq 0$ is the weight assigned to the edge $(i,j)$ connecting vertices $i$ and $j$. Edge weights often measure the similarity between connected vertices.

The graph Laplacian matrix is defined from the adjacency matrix. Among different variants of Laplacian matrices, the combinatorial graph Laplacian used in [13, 25] is defined as $L := D - A$, where $D$ is the degree matrix—a diagonal matrix where $d_{i,i} = \sum_{j=1}^{n} a_{i,j}$.

Graph signal refers to data that resides on the vertices of a graph. In our case, the coordinates of each point in the input dynamic point cloud are the graph signal. A graph signal $z \in \mathbb{R}^n$ defined on a graph $G$ is smooth with respect to the topology of $G$ if

$$\sum_{i,j} a_{i,j} (z_i - z_j)^2 < \epsilon,$$

where $\epsilon$ is a small positive scalar, and $i \sim j$ denotes two vertices $i$ and $j$ are one-hop neighbors in the graph. In order to satisfy (1), $z_i$ and $z_j$ have to be similar for a large edge weight $a_{i,j}$, and could be quite different for a small $a_{i,j}$. Hence, (1) enforces $z$ to adapt to the topology of $G$, which is thus coined graph-signal smoothness prior.

As $x^T L x = \sum_{i,j} a_{i,j} (z_i - z_j)^2$ [27], (1) is concisely written as $x^T L x < \epsilon$ in the sequel. This term will be employed as the prior for the MAP estimation of dynamic point clouds.

4 PROBLEM FORMULATION

In this section, we elaborate on the proposed problem formulation. We start from the modeling of a dynamic point cloud sequence via spatio-temporal GMRFs, and propose such modeling on patch basis. Then we pose a MAP estimation of the underlying dynamic point cloud, and come up with the likelihood function and prior distribution. Finally, we arrive at the problem formulation from the MAP estimation.

4.1 Spatial-Temporal Modeling

A dynamic point cloud sequence $P = \{ U_t, U_{t+1}, \ldots, U_{t+k} \}$ consists of $m$ frames of point clouds. The coordinates $U_t = [u_{1,t}, u_{2,t}, \ldots, u_{n,t}]^T \in \mathbb{R}^{n \times 3}$ denote the position of each point in the point cloud at frame $t$, in which $u_{i,t} \in \mathbb{R}^3$ represents the coordinates of the $i$-th point at frame $t$. Let $U_t$ denote the ground truth coordinates of the $t$-th frame, and $\hat{U}_{t-1}$, $\hat{U}_t$ denote the noise-corrupted coordinates of the $(t-1)$-th and $t$-th frame respectively. Then we formulate the dynamic point cloud denoising problem as

$$\hat{U}_t = f(U_{t-1}, U_t) + E_t,$$

where $E_t \in \mathbb{R}^{n \times 3}$ is a zero-mean signal-independent noise. For point clouds acquired from equipments, the noise distribution is related to the acquisition equipments. Several previous works [19, 28] have shown through statistics that the noise in point clouds approximates Gaussian distribution for 3D scanning equipments such as Microsoft Kinect, 3D laser scanner, etc. As these are popular sensors, we assume the noise follows Gaussian distribution.

Spatio-temporal GMRF modeling. In particular, we model the relationship in consecutive frames of a dynamic point cloud via spatio-temporal GMRF models. A spatial GMRF is a restrictive multivariate Gaussian distribution that satisfies additional conditional independence assumptions. A graph is often used to represent the conditional independence assumption. Here is the formal definition:

Definition: A random vector $x = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n$ is a GMRF with respect to a graph $G = (V, E)$ if its density has the form

$$p(x) = (2\pi)^{-\frac{n}{2}} |Q|^\frac{1}{2} \exp \left( -\frac{1}{2} (x - \mu)^T Q (x - \mu) \right),$$

and

$$Q_{i,j} \neq 0 \iff \{i, j\} \in E, \forall i \neq j.$$

Spatio-temporal GMRF models are extensions of spatial GMRF models to account for additional temporal variation. In our case, we represent a dynamic point cloud of $m$ frames on a sequence of $m$ subgraphs. Each subgraph describes the intra-frame connectivities within each frame, and temporal connectivities exist between neighboring subgraphs to describe the inter-frame connectivities.

Patch representation. Further, as it is computationally expensive to consume an entire point cloud, we model both intra-frame and inter-frame dependencies on patch basis. Unlike images or videos defined on regular grids, point clouds reside on irregular domain with uncertain local neighborhood, thus the definition of a patch is nontrivial. We define a patch $P_{t,l} \in \mathbb{R}^{(k+1) \times 3}$ in the point cloud at frame $t$ as a local point set of $k+1$ points, consisting of a centering point $c_{t,l} \in \mathbb{R}^3$ and its $k$-nearest neighbors in terms of Euclidean distance. Then the entire set of patches at frame $t$ is

$$P_t = S_t U_t - C_t,$$

where $S_t \in \{0,1\}^{(k+1)M \times n}$ is a sampling matrix to select points from point cloud $U_t$ so as to form $M$ patches of $(k+1)$ points each, and $C \in \mathbb{R}^{(k+1)M \times 3}$ contains the coordinates of patch centers for each point.

Based on the patch representation, we model the intra-frame dependency by building graph connectivities among similar patches within a frame, and model the inter-frame dependency by constructing graph connectivities between corresponding patches over consecutive frames. The details of searching similar patches within a frame and corresponding patches between frames will be discussed in section 5.2.
4.2 MAP Estimation of Dynamic Point Clouds

Under the spatio-temporal GMRF modeling, we pose a MAP estimation for the underlying patches $P_t$ in the point cloud $U_t$ at frame $t$: given the observed noisy previous frame $\hat{P}_{t-1}$ and current noisy frame $\hat{P}_t$, find the most probable signal $P_t$,

$$P_t^{\text{MAP}}(\hat{P}_{t-1}, \hat{P}_t) = \arg \max_{P_t} f(\hat{P}_{t-1}, \hat{P}_t | P_t) g(P_t), \quad (6)$$

where $f(\hat{P}_{t-1}, \hat{P}_t | P_t)$ is the likelihood function, and $g(P_t)$ is the signal prior. Because $P_t$ are patches that cover the entire $U_t$, Eq. (6) also gives the MAP estimation of $U_t$:

$$U_t^{\text{MAP}}(\hat{P}_{t-1}, \hat{P}_t) = \arg \max_{U_t} f(\hat{P}_{t-1}, \hat{P}_t | P_t) g(P_t). \quad (7)$$

The proposed likelihood function, $f(\hat{P}_{t-1}, \hat{P}_t | P_t)$ is the probability of obtaining the observed point clouds $\hat{P}_{t-1}$ and $\hat{P}_t$, given the desired current frame $P_t$. We have

$$f(\hat{P}_{t-1}, \hat{P}_t | P_t) = f(\hat{P}_{t-1} | P_t) f(\hat{P}_t | P_t)$$

$$= f(\hat{P}_{t-1} | P_t) f(\hat{P}_t | P_t), \quad (8)$$

where $f(\hat{P}_{t-1} | P_t, \hat{P}_t)$ is equivalent to $f(\hat{P}_{t-1} | P_t)$ because we assume the noise of the $(t-1)$-th frame and $t$-th frame are independent.

For the second term in Eq. (8), according to the linear relationship of $P_t$ and $U_t$ as in Eq. (5) and assuming zero-mean Gaussian distribution for the noise, we have

$$\arg \max_{U_t} f(\hat{U}_t | U_t) = \arg \max_{U_t} f(\hat{U}_t - U_t | U_t)$$

$$= \arg \max_{U_t} f(\hat{U}_t - U_t | U_t)$$

$$= \arg \max_{U_t} \alpha_1 \exp \left( -\lambda_1 \| \hat{U}_t - U_t \|_2^2 \right) \cdot \alpha_2 \exp \left( -\lambda_2 \| \hat{P}_{t-1} - P_t \|_2^2 \right), \quad (9)$$

where $\alpha_1$ is a normalization factor to keep the integral of the probability function equal to 1, and $\lambda_1$ is a variance-related parameter.

For the first term in Eq. (8), since the variation between adjacent frames is often trivial, we assume the current frame is a perturbed version of the previous frame. In particular, we propose to adopt a weighting parameter $w_i$ to represent the perturbation at the $i$-th patch, leading to

$$f(\hat{P}_{t-1} | P_t) = \alpha_2 \exp \left( -\lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 \right). \quad (10)$$

where $\alpha_2$ is a normalization factor, and $\lambda_2$ is a variance-related parameter. In the proposed algorithm, $w_i$ is a variable depending on $\hat{P}_{t-1,i}$ and $P_{t,i}$, which describes the similarity between $\hat{P}_{t-1,i}$ and $P_{t,i}$.

The proposed prior distribution. Since $U_t$ follows GMRF modeling, assuming zero mean, we have its prior distribution from Eq. (3) as:

$$g(P_t) = \beta \exp \left( -\frac{1}{2} P_t^T Q_t P_t \right). \quad (11)$$

where $Q_t$ is the precision matrix of the $t$-th frame, and $\beta$ is a normalization factor.

However, it is challenging to estimate $Q_t$ statistically from small amounts of data. Instead, as introduced in [33], the precision matrix can be interpreted by the graph Laplacian, i.e., $Q_t = \delta L_t$ by a scalar $\delta$. Hence, we replace $Q_t$ in Eq. (11) by the graph Laplacian $L_t$, leading to

$$g(P_t) = \beta \exp \left( -\frac{1}{2} P_t^T L_t P_t \right). \quad (12)$$

4.3 Final Problem Formulation

Combining Eq. (7), Eq. (8), Eq. (9), Eq. (10), Eq. (12), we have

$$U_t^{\text{MAP}}(\hat{P}_{t-1}, \hat{P}_t) = \arg \max_{U_t} f(\hat{P}_{t-1}, \hat{P}_t | P_t) g(P_t)$$

$$= \arg \max_{U_t} \alpha_1 \exp \left( -\lambda_1 \| \hat{U}_t - U_t \|_2^2 \right) \cdot \alpha_2 \exp \left( -\lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 \right)$$

$$= \arg \max_{U_t} \alpha_1 \exp \left( -\lambda_1 \| \hat{U}_t - U_t \|_2^2 \right) \cdot \alpha_2 \exp \left( -\lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 \right)$$

$$= \arg \max_{U_t} \alpha_1 \exp \left( -\lambda_1 \| \hat{U}_t - U_t \|_2^2 \right) \cdot \alpha_2 \exp \left( -\lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 \right)$$

$$= \arg \max_{U_t} \alpha_1 \exp \left( -\lambda_1 \| \hat{U}_t - U_t \|_2^2 \right) \cdot \alpha_2 \exp \left( -\lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 \right)$$

$$= \arg \max_{U_t} \alpha_1 \exp \left( -\lambda_1 \| \hat{U}_t - U_t \|_2^2 \right) \cdot \alpha_2 \exp \left( -\lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 \right)$$

$$= \arg \max_{U_t} \alpha_1 \exp \left( -\lambda_1 \| \hat{U}_t - U_t \|_2^2 \right) \cdot \alpha_2 \exp \left( -\lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 \right)$$

Due to the dependency of $L_t$ and $w_i$ on $U_t$, $w_i$ and $L_t$ are optimization variables as well as $U_t$.

Taking logarithm of Eq. (13) and multiplying by $-1$, we arrive at the final problem formulation:

$$\min_{U_t, L_t, w_i} \lambda_1 \| U_t - \hat{U}_t \|_2^2 + \lambda_2 \sum_{i=1}^M w_i \| \hat{P}_{t-1,i} - P_{t,i} \|_2^2 + P_t^T L_t P_t,$$

s.t. $P_t = S_t U_t - C_t.$

(14)
As $U_l$, $L_l$ and $w_l$ are optimization variables, Eq. (14) is nontrivial to solve. We develop an efficient algorithm to address this problem formulation in the next section.

5 THE PROPOSED ALGORITHM

As demonstrated in Fig. 2, for a given dynamic point cloud, we perform denoising on each frame sequentially. The proposed algorithm consists of four major steps: 1) patch construction, in which we form overlapped patches from chosen patch centers; 2) similar/corresponding patch search, in which we search similar patches for each patch in the current frame, and search the corresponding patch in the previous frame; 3) graph construction, in which we build a spatio-temporal graph with intra-connectivities among similar patches and inter-connectivities among corresponding patches; 4) optimization, in which we solve the proposed problem formulation in Eq. (14) via alternating minimization, thus performing step 2-4 iteratively. Note that, the inter-frame reference is bypassed for denoising the first frame as there is no previous frame. We discuss the four steps separately in detail.

5.1 Patch Construction

As each patch is formed around a patch center, we first select $M$ points from $U_l$ as the patch centers, denoted as $\{c_{l,1}, c_{l,2}, ..., c_{l,M}\} \in \mathbb{R}^{M \times 3} \subseteq U_l$. In order to keep the patches distributed as uniformly as possible, we first choose a random point in $U_l$ as $c_{l,1}$, and add a point which holds the farthest distance to the previous patch centers as the next patch center, until there are $M$ points in the set of patch centers. We then search the $k$-nearest neighbors of each patch center in terms of Euclidean distance, which leads to $M$ patches in $U_l$.

5.2 Similar/Corresponding Patch Search

For each constructed patch in $U_l$, we search for its similar patches locally in $U_l$, and its corresponding patch in $U_{l-1}$. A metric is necessary to measure the similarity between patches. It remains a challenging problem as the patches are irregular.

Similarity Metric. We deploy a simplified method of [32] to measure the similarity between patch $\hat{p}_{t,n}$ and patch $\hat{p}_{t,m}$. The key idea is to compare the distance of the two patches, from each point to the tangent plane at the patch center.

Firstly, we structure the tangent planes of the two patches. A point cloud describes the surface of the object. We thus calculate the surface normals $n_n$ and $n_m$ for patch $\hat{p}_{t,n}$ and patch $\hat{p}_{t,m}$ respectively. Then we acquire the tangent planes of the patches at the patch center $c_{t,n}$ and $c_{t,m}$.

Secondly, we measure the difference of patches with the distance of the two patches from each point to the corresponding tangent plane. Specifically, we project each point in patch $\hat{p}_{t,n}$ and patch $\hat{p}_{t,m}$ to the tangent plane of patch $\hat{p}_{t,n}$. For the $i$-th point $v_n^i$ in patch $\hat{p}_{t,n}$, we find the point $v_m^i$ in $\hat{p}_{t,m}$ whose projection on the tangent plane is closest to that of $v_n^i$. We then define $d_n(v_n^i)$ and $d_m(v_m^i)$ as the distance of the two points to their projections on the tangent plane. $d_n(v_n^i) - d_m(v_m^i)$ is regarded as the difference of the two patches in point $v_n^i$ and $v_m^i$. Then we acquire the average difference between the two patches at all the $(k + 1)$ points:

$$D_{mn} = \sqrt{\frac{1}{k+1} \sum_{i=1}^{k+1} [d_n(v_n^i) - d_m(v_m^i)]^2}. \quad (15)$$

Similarly, projecting each point in patch $\hat{p}_{t,n}$ and patch $\hat{p}_{t,m}$ to the tangent plane of patch $\hat{p}_{t,m}$, we acquire an average difference $D_{nn}$.

$$D_{mn} = \sqrt{\frac{1}{2} [D_{m}^2 + D_{n}^2].} \quad (16)$$

Finally, we measure the patch similarity with a thresholded Gaussian function using the above mean difference:

$$s_{mn} = \begin{cases} \exp\left(-\frac{D_{mn}^2}{2\epsilon^2}\right), & D_{mn} < r, \\ 0, & D_{mn} > r, \end{cases} \quad (17)$$

where $r$ is a threshold determined by the density of the point cloud, and $\epsilon$ is a variance-related parameter. The larger $s_{mn}$ is, the more similar $\hat{p}_{t,n}$ and $\hat{p}_{t,m}$ are.

Local Patch Search. Given the similarity measure, we search for similar patches within the current frame. The number of the similar patches depends on the size of the point cloud. As to the corresponding patch in the previous frame, we only search one patch as the corresponding patch. Given a target patch in the $t$-th frame $\hat{p}_{t,i}, i \in [1,M]$, we choose the most similar patch to $\hat{p}_{t,i}$ in the $(t-1)$-th frame as the corresponding patch $\hat{p}_{t-1,i}$.

In order to reduce the computation complexity, we set a local window in the $(t-1)$-th frame for the corresponding patch search, which contains patches centering at the K-nearest neighbors of the target patch center. Thus we evaluate the patch similarity between the target patch and these K-nearest patches instead of all the patches in the $(t-1)$-th frame. Once we acquire the patch $\hat{p}_{t-1,i}$, we deploy its similarity measure in Eq. (17) to the patch $\hat{p}_{t,i}$ as the weighting parameter $w_l$ in Eq. (10). Similarly, we set a local window for similar patch search in the $t$-th frame.

5.3 Graph Construction

Having searched intra-frame similar patches and inter-frame corresponding patches, we construct a spatio-temporal graph over the patches. Though this graph is supposed to be learned via Eq. (14), the computational complexity of solving the optimization problem would be high and the numerical computation might be unstable. Instead, we propose to manually build intra-frame graph connectivities and inter-frame graph connectivities based on the patch similarity, as shown in Fig. 3.

Intra-frame graph construction. Given a target patch $\hat{p}_{t,i}$ in the $t$-th frame, we construct a bipartite graph between $\hat{p}_{t,i}$ and each of its similar patches $\hat{p}_{t,m}$. Specifically, each point in $\hat{p}_{t,i}$ is connected with its nearest neighbors in $\hat{p}_{t,m}$, where the distance is in terms of their projections on the tangent plane decided by the surface normal of $\hat{p}_{t,i}$ at the patch center. Similarly, each point in $\hat{p}_{t,m}$ is connected with the nearest points in $\hat{p}_{t,i}$ in terms of their projections on the tangent plane decided by the surface normal of $\hat{p}_{t,m}$ at the patch center. The intra-frame connectivities are undirected and share the same weight $s_{nm}$ as in Eq. (17). We build intra-frame connectivities over all the patches in this way, which leads to the graph Laplacian $L_l \in \mathbb{R}^{(k+1)M \times (k+1)M}$, where $(k+1)$ is the number of points in each patch and $M$ is the number of patches in the $t$-th frame.

Note that, we do not connect points within each patch explicitly in order to avoid bringing the coordinates close to each other in a patch. However, connectivities may exist among some points if they are nearest neighbors in overlapping patches.
We propose an efficient alternating minimization approach as follows. We first rewrite Eq. (14) for efficient optimization. We define a matrix $W_{t+1,t}$ to describe the weights $w_{ij}$ between corresponding patches:

$$W_{t+1,t} = \text{diag} \left[ \sqrt{w_{11}}, \sqrt{w_{22}}, \ldots, \sqrt{w_{MM}} \right].$$

Then we rewrite Eq. (14) in the following form:

$$\min_{U_t, L_t, W_{t+1,t}} \lambda_1 \| U_t - \hat{U}_t \|^2 + \lambda_2 \| W_{t+1,t} - (S_t U_t - C_t) - W_{t+1,t-1} \hat{P}_{t-1} \|^2 + (S_t U_t - C_t)^T L_t (S_t U_t - C_t).$$

Eq. (19) is nontrivial to solve with three optimization variables. We propose an efficient alternating minimization approach as follows. Firstly, we initialize $U_t$ with the noisy observation $\hat{U}_t$, based on which we calculate $L_t$ from the proposed inter-frame graph construction and $W_{t+1,t}$ from the proposed inter-frame graph construction. Secondly, we fix both $L_t$ and $W_{t+1,t}$, and take derivative of Eq. (19) with respect to $U_t$ and set the derivative to 0. This leads to the closed-form solution of $U_t$:

$$\left( S_t^T L_t S_t + \lambda_1 I + \lambda_2 S_t^T W_{t+1,t-1} W_{t+1,t-1} S_t \right) U_t = S_t^T L_t C_t + \lambda_1 \hat{U}_t + \lambda_2 S_t^T W_{t+1,t-1} (C_t + \hat{P}_{t-1}).$$

Then we update $L_t$ and $W_{t+1,t}$ from the solved $U_t$. The iterations are repeated until convergence, i.e., when the difference of $U_t, L_t$, and $W_{t+1,t}$ from their values in the previous iteration is trivial.

Note that, we first perform denoising on the first frame with only intra-correlations. Then for the next frame, in order to take advantage of the previously reconstructed frame for better reference, we take $\hat{P}_{t-1}$ as patches in the denoised previous frame instead of those in the observed noisy previous frame. Hence, the final solution of $U_t$ in Eq. (19) serves as the reference frame for the denoising of the next frame. A framework of the proposed algorithm is shown in Algorithm 1.

**Algorithm 1: 3D Dynamic Point Cloud Denoising**

**Input:** A noisy dynamic point cloud sequence $\hat{P} = \{\hat{U}_1, \hat{U}_2, \ldots, \hat{U}_m\}$

**Output:** Denoised dynamic point cloud sequence $P = \{U_1, U_2, \ldots, U_m\}$

1. for $\hat{U}_t$ in $\hat{P}$ do
2. Initialize $U_t$ with $\hat{U}_t$
3. Select $M$ points (set $C_t$) as patch centers;
4. for $c_t$ in $C_t$ do
5. Find $k$-nearest neighbors of $c_t$;
6. Build patch $\hat{p}_{t,i}$;
7. Add $\hat{p}_{t,i}$ to $\hat{P}_t$;
8. end
9. repeat
10. for $\hat{P}_{t+1,t}$ in $\hat{P}_t$ do
11. for $\hat{p}_{t+1,t}$ in $\hat{P}_t$’s adjacent patches do
12. Calculate the similarity metric $s_{ij}$ between $\hat{p}_{t+1,t}$ and $\hat{p}_{t,i}$ as in Eq. (17);
13. Connect nearest points in $\hat{p}_{t+1,t}$ and $\hat{p}_{t,i}$ with the edge weight $s_{ij}$;
14. end
15. Search the corresponding patch $\hat{p}_{t+1,t}$ of $\hat{p}_{t+1,t}$ in the previous frame, which is the most similar to $\hat{p}_{t+1,t}$ in terms of the metric metric $s_{(t-1)|t}$ as in Eq. (17);
16. Connect corresponding points in $\hat{p}_{t+1,t}$ and $\hat{P}_{t+1,t}$ with the edge weight $s_{(t-1)|t}$;
17. end
18. Compute the intra-frame graph Laplacian $L_t$;
19. Compute the weight matrix $W_{t+1,t}$ between corresponding patches;
20. Solve Eq. (20) to update $U_t$;
21. until convergence;
22. $U_t$ serves as the input for the denoising of the next frame.
23. end

**6 EXPERIMENTAL RESULTS**

**6.1 Experimental Setup**

We evaluate our algorithm by testing on dynamic point clouds from MPEG [9] and JPEG Pleno [10], including soldier, longdress, loot, redandblack and UlliWegner. We randomly choose 6 consecutive frames as the sample data: frame 601-606 in soldier, frame 1201-1206 in loot, frame 1201-1206 in longdress, frame 1501-1506...
Table 1: Experimental Comparison with noise variance 0.03

|        | Noisy | APSS | RIMLS | MRPCA | Baseline | Ours |
|--------|-------|------|-------|-------|----------|------|
| Soldier | 1.4984 | 1.4125 | 1.3572 | 1.3488 | 1.2805 | 1.2726 |
| Longdress | 1.4746 | 1.3985 | 1.3360 | 1.3247 | 1.2475 | 1.2406 |
| Loot | 1.4715 | 1.3571 | 1.3279 | 1.3011 | 1.2208 | 1.2165 |
| Redandblack | 1.4895 | 1.3892 | 1.3499 | 1.3221 | 1.2506 | 1.2432 |
| UlliWegner | 1.3355 | 1.2632 | 1.2063 | 1.1895 | 1.1091 | 1.1075 |

Table 2: Experimental Comparison with noise variance 0.05

|        | Noisy | APSS | RIMLS | MRPCA | Baseline | Ours |
|--------|-------|------|-------|-------|----------|------|
| Soldier | 2.1433 | 1.8047 | 1.8105 | 1.8116 | 1.7815 | 1.7538 |
| Longdress | 2.1260 | 1.8007 | 1.7955 | 1.7922 | 1.7329 | 1.7213 |
| Loot | 2.1286 | 1.7668 | 1.7883 | 1.7703 | 1.7079 | 1.6938 |
| Redandblack | 2.1110 | 1.7915 | 1.8064 | 1.7959 | 1.7439 | 1.7297 |
| UlliWegner | 2.0865 | 1.7830 | 1.7807 | 1.8024 | 1.7444 | 1.7298 |

Table 3: Experimental Comparison with noise variance 0.07

|        | Noisy | APSS | RIMLS | MRPCA | Baseline | Ours |
|--------|-------|------|-------|-------|----------|------|
| Soldier | 2.5417 | 1.9675 | 2.0450 | 1.9999 | 2.0111 | 1.9308 |
| Longdress | 2.5139 | 1.9630 | 2.0297 | 1.9754 | 1.9748 | 1.8933 |
| Loot | 2.5205 | 1.9359 | 2.0271 | 1.9487 | 1.9299 | 1.8613 |
| Redandblack | 2.5035 | 1.9726 | 2.0537 | 1.9849 | 2.0037 | 1.9096 |
| UlliWegner | 2.5600 | 2.0730 | 2.1060 | 2.0893 | 2.0405 | 2.0263 |

Table 4: Experimental Comparison with noise variance 0.1

|        | Noisy | APSS | RIMLS | MRPCA | Baseline | Ours |
|--------|-------|------|-------|-------|----------|------|
| Soldier | 3.0127 | 2.1404 | 2.3901 | 2.1874 | 2.1442 | 2.0902 |
| Longdress | 2.9761 | 2.1236 | 2.3748 | 2.1360 | 2.0975 | 2.0509 |
| Loot | 2.9853 | 2.1118 | 2.3338 | 2.1057 | 2.0431 | 1.9942 |
| Redandblack | 2.9622 | 2.1433 | 2.3266 | 2.1563 | 2.1339 | 2.0706 |
| UlliWegner | 3.0492 | 2.2737 | 2.4086 | 2.2988 | 2.2925 | 2.2430 |

6.2 Experimental Results

Objective results. We list the denoising results of different methods in Tab. 1,2,3,4 and mark the lowest MSE in bold. We see that our method outperforms all the four static point cloud denoising methods on the five datasets under all the noise levels. Specifically, we reduce the average MSE by 5.13% on average over APSS, 7.35% on average over RIMLS, 4.52% on average over MRPCA, and 1.81% on average over Baseline. This validates the effectiveness of our method. In particular, the improvement over Baseline validates that the temporal correlation we exploit is beneficial to dynamic point cloud denoising. Further, the MSE reduction over Baseline is 0.4%, 0.9%, 3%, 3% respectively with increasing noise levels. This indicates that the temporal correlation makes more impact at high noise levels, because the inter-frame difference is more negligible compared to the noise with large variance.

For easier comparison with state-of-the-art static point cloud denoising methods, we compute the average MSE on the five datasets under each noise level for different methods. The results are visualized in Fig. 4. We see that we achieve the best performance under various noise levels.

Subjective results. As illustrated in Fig. 5, the proposed method also has competitive visual results, especially in local details and temporal consistency. In order to demonstrate the temporal consistency, instead of the previous chosen frames as in Sec. 6.1, we choose another 6 consecutive frames that exhibit apparent movement in loot and soldier under noise variance 0.05. We show the visual comparison with APSS and MRPCA because they have comparatively better objective performance as presented in Fig. 4. We see that, our results preserve the local structure and keep the temporal consistency better. For example, in the loot dataset, the boundary of the left hand in our result is much cleaner than that in APSS, and smoother than that in MRPCA. Also, our result exhibits better temporal consistency in general.

7 CONCLUSION

While the denoising of static 3D point clouds has been widely studied, it remains a challenge to denoise dynamic point clouds. In order to address the problem, we propose a graph-based method to exploit both the intra-frame self-similarity and inter-frame consistency. Specifically, we propose spatio-temporal graph modeling of patches in dynamic point clouds, and pose a MAP estimation on the underlying patches. The key is to construct intra-frame connectivities among searched similar patches within the same frame, as well as inter-frame connectivities between searched corresponding patches over consecutive frames. We then accordingly cast dynamic point cloud denoising as an optimization program, which leverages the similar/corresponding patches and a graph-signal smoothness prior based on the constructed graph. Experimental results show that our method outperforms frame-by-frame denoising from state-of-the-art static point cloud denoising approaches.
Figure 5: Subjective comparison in local details and temporal consistency on Loot for frames 1075-1080 (row 1-4) and Soldier for frames 631-636 (row 5-8) under noise variance 0.05.
REFERENCES

[1] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. 2003. Computing and Rendering Point Set Surfaces. IEEE Transactions on Visualization and Computer Graphics 9, 1 (2003), 0–15.

[2] Jonathan Almonacid, Celia Cintas, Claudio Derieux, and Mithra Lewis. 2018. Point Cloud Denoising using Deep Learning. In Congreso Argentino de Ciencias de la Informática y Desarrollo de Investigación (CACID). 1–5.

[3] Haim Avron, Andrei Sharf, Chen Greif, and Daniel Cohen-Or. 2010. 1-l Sparse Reconstruction of Sharp Point Set Surfaces. ACM Transactions on Graphics (TOG) 29 (10 2010), 135.

[4] Antonio Buades, Bartomeu Coll, and J-M Morel. 2005. A non-local algorithm for image denoising. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 2. 60–65.

[5] Fan RK Chung. 1997. Spectral graph theory. In Conference Board of the Mathematical Sciences. American Mathematical Society.

[6] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. 2007. Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering. IEEE Transactions on Image Processing (TIP) 16, 8 (Aug 2007), 2080–2095.

[7] Julie Digne. 2012. Similarity based filtering of point clouds. In IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 73–79.

[8] Chinthaka Dinesh, Gene Cheung, Ivan V. Bajic, and Yang Cheng. 2018. Fast 3D Point Cloud Denoising via Bipartite Graph Approximation & Total Variation. (2018).

[9] Thomas Ehlers, Ingo Feldmann, Oliver Schreer, Peter Kauff, and Tanja Unger. 2018. HII Point cloud dataset of a boxing trainer. http://mpeg.fs.int-evry.fr/MPEG/ PCC/DataSets/pointCloud/CIF/. In ISO/IEC JTC1/SC29/WG11 (MPEG2018) input document M42921.

[10] Taos Myers Eugene d’Eon, Bob Harrison and Philip A. Chou. 2017. 8i Voxelized Full Bodies, version 2 - A Voxelized Point Cloud Dataset. ISO/IEC JTC1/SC29/ WG11/ WG1 (MPEG/JPEG) input document m40059/M74006 (January 2017).

[11] Xiang Gao, Wei Hu, and Zongming Guo. 2018. Graph-Based Point Cloud Denoising. In IEEE Fourth International Conference on Multimedia Big Data (BigMM), 1–6.

[12] Gael Guennebaud and Markus Gross. 2007. Algebraic Point Set Surfaces. In ACM SIGGRAPH 23.

[13] Wei Hu, Gene Cheung, Antonio Ortega, and Oscar C. Au. 2015. Multiresolution graph fourier transform for compression of piecewise smooth images. IEEE Transactions on Image Processing (TIP) 24 (January 2015), 419–433.

[14] Hui Huang, Shihao Wu, Mingjun Gong, Daniel Cohen-Or, and Hao Zhang. 2013. Edge-Aware Point Set Resampling. ACM Transactions on Graphics (TOG) 32, 1 (2013), 1–12.

[15] Wenming Huang, Yuanwang Li, Peizhi Wen, and Xiaojun Wu. 2009. Algorithm for 3D Point Cloud Denoising. In International Conference on Genetic and Evolutionary Computing.

[16] Huang Hui, Li Dan, Zhang Hao, Uri Ascher, and Daniel Cohen-Or. 2009. Consolidation of unorganized point clouds for surface reconstruction. In Acm Siggraph Asia.

[17] Yaron Lipman, Daniel Cohen-Or, David Levin, and Hillel Tal-Ezer. 2007. Parameterization-free projection for geometry reconstruction. Acm Transactions on Graphics 26, 3 (2007), 22.

[18] Enrico Mattei and Alexey Castrodad. 2017. Point Cloud Denoising via Moving RPCA. Computer Graphics Forum 36 (11 2017).

[19] Chuong V Nguyen, Shahram Izadi, and David Lovell. 2012. Modeling Kinect sensor noise for improved 3D reconstruction and tracking. In International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPV'T), 524–530.

[20] Cengiz Oztireli, Gael Guennebaud, and Markus Gross. 2009. Feature Preserving Point Set Surfaces based on Non-Linear Kernel Regression. 28 (2009), 495–501.

[21] Guy Rosman, Anastasia Dubrovin, and Ron Kimmel. 2013. Patch-Collaborative Spectral Point-Cloud Denoising. In Computer Graphics Forum, Vol. 32. Wiley Online Library, 1–12.

[22] Harvard Rue and Leonhard Held. 2005. Gaussian Markov random fields: theory and applications. Chapman and Hall/CRC.

[23] Radu Bogdan Rusu and Steve Cousins. 2011. 3D is here: Point Cloud Library (PCL). IEEE International Conference on Robotics and Automation (2011), 1–4.

[24] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Daila, and Michael Beetz. 2008. Towards 3D Point cloud based object maps for household environments. Robotics and Autonomous Systems 56, 11 (2008), 927–941.

[25] Godwin Shen, Woor Shik Kim, Sumi K. Naranj, Antonio Ortega, and Ho Choon Way. 2010. Edge-adaptive transforms for efficient depth map coding. In IEEE Picture Coding Symposium (PCS). 566–569.

[26] David I Shuman, Sumi K. Naranj, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Processing Magazine 30 (2013), 83–98.

[27] DA Spielman. 2004. Lecture 2 of spectral graph theory and its applications.

[28] Xianfang Sun, Paul L. Rosin, Ralph R. Martin, and Frank C. Langbein. 2008. Noise in 3D laser range scanner data. IEEE International Conference on Shape Modeling and Applications (2008), 37–45.

[29] Yiqing Sun, Scott Schaefer, and Wenping Wang. 2015. Denoising point sets via L 0 minimization.

[30] Christian Teluven, Rafael Melkoura, Zhi Li, and Sebastien Lasserre. 2016. Use Cases for Point Cloud Compression. In ISO/IEC JTC1/SC29/WG11 (MPEG) output document N16331.

[31] Fu Yan and Jinlei Zhai. 2015. Research on scattered points cloud denoising algorithm. In IEEE International Conference on Signal Processing.

[32] Jin Zeng, Gene Cheung, Michael Ng, Jiuhao Pang, and Cheng Yang. 2018. 3D point cloud denoising using graph laplacian regularization of a low dimensional manifold model. arXiv preprint arXiv:1803.07252 (2018).

[33] Cha Zhang, Dinesh Florencio, and Charles Loop. 2014. Point cloud attribute compression with graph transform. In IEEE International Conference on Image Processing (ICIP). 2066–2070.