IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 5, 2021, accepted January 20, 2021, date of publication January 25, 2021, date of current version February 5, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3054171*

# Joint Geometry and Color Point Cloud Denoising Based on Graph Wavelets

**MUHAMMAD ABEER IRFAN, (Graduate Student Member, IEEE), AND ENRICO MAGLI, (Fellow, IEEE)**
Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Turin, Italy
Corresponding author: Muhammad Abeer Irfan (abeer.irfan@polito.it)

**ABSTRACT** A point cloud is an effective 3D geometrical presentation of data paired with different attributes such as transparency, normal and color of each point. The imperfect acquisition process of a 3D point cloud usually generates a significant amount of noise. Hence, point cloud denoising has received a lot of attention. Most of the existing techniques perform point cloud denoising based only on the geometry information of the neighbouring points; there are very few works considering the problem of denoising of color attributes of a point cloud, and taking advantage of the correlation between geometry and color. In this article, we introduce a novel non-iterative set-up for the denoising of point cloud based on spectral graph wavelet transform (SGW) that jointly exploits geometry and color to perform denoising of geometry and color attributes in graph spectral domain. The designed framework is based on the construction of joint geometry and color graph that compacts the energy of smooth graph signals in the low-frequency bands. The noise is then removed from the spectral graph wavelet coefficients by applying data-driven adaptive soft-thresholding. Extensive simulation results show that the proposed denoising technique significantly outperforms state-of-the-art methods using both subjective and objective quality metrics.

**INDEX TERMS** Point cloud denoising, color denoising, graph signal processing, spectral graph wavelets.

## I. INTRODUCTION

Point clouds are considered as an efficient and useful technique to render volumetric data in 3D space. Point clouds are unorganized collections of points in space, where a single point consists of 3D geometric information along with attribute data, i.e., color, transparency, normals. Point clouds have now been widely employed in several different fields such as navigation of unmanned vehicles, culture and heritage reconstruction, 3D immersive telepresence, and 3D broadcasting [1].

With the advent of computer vision technology and optical components, in addition to laser scanning sensors, low-cost RGB-D cameras have been developed such as Astra, Astra S, Astra Pro, Intel RealSense [2]–[4] and Microsoft Kinect [5], [6]. Point cloud acquisition is quite easy using such cameras or generated by stereo matching algorithms [7] are typically suffered from non-negligible noise in geometry and color due to various angles, reflective materials or type of surfaces of the objects, light intensities, as well as the limitation of sensors [8]. The geometry, along with the color attribute of an object or scene, plays a vital role for several augmented and virtual reality applications [9]. The color attribute of a point cloud is considered an essential feature for applications such as segmentation [10]–[12] and retrieval of 3D models [13], [14]. Noisy colors may lead to wrong segmentation and inaccurate retrieval of point cloud models. In the recent literature, a lot of work has been performed for denoising the noise in the point coordinates (i.e., geometry noise) to improved their quality [15]–[18]. However, only one work is available for the color denoising of a point cloud using Graph Laplacian regularizer (GLR) coupled with alternating direction method of multipliers [19]; still, there are various applications employing both the geometry and color attribute of a point cloud.

In this article, we present a novel non-iterative algorithm for the point cloud geometry and color denoising problem, which jointly uses geometry and color attributes of points. We note that the color attribute is a powerful and very informative feature that is indeed correlated with the geometry, also observed in [12], [19], [20] and discussed in Sec. III. Knowledge of the color can be exploited to improve the

The associate editor coordinating the review of this manuscript and approving it for publication was Hongjun Su.

denoising process for geometry noise, and indeed geometry and color can be jointly employed to remove geometry *and* color noise. The proposed technique applies Spectral Graph Wavelet (SGW) [21], offering a trade-off between spectral-domain and vertex localization. This enables us to take the benefit of global smoothness, i.e., most of the energy lies in the lower frequency bands, while avoiding to over-smooth discontinuities that represents the large magnitude coefficients in the high frequency bands. Extensive simulation results on synthetic and real-world point clouds show that our proposed algorithm for color-only and geometry-only denoising outperforms state-of-the-art techniques using both subjective and objective quality metrics.

The main contributions of our work are as follow:

1) We provide a statistical analysis to assess the correlation between color and geometry of a point cloud.
2) We propose a 3D point cloud geometry denoising problem using SGW, for which we have constructed a joint geometry/color graph. We use data-driven adaptive thresholding for denoising the point cloud in contrast to the algorithm described in [22] where the prior knowledge of standard deviation is required for thresholding. The proposed algorithm is non-iterative, in contrast with [23] that employs an iterative technique using convex minimization.
3) We use the same framework for solving the color denoising problem of a point cloud in a computationally efficient way with respect to the approach discussed in [24] where the point cloud color is denoised using an iterative technique.

The rest of the paper is organized as follows. In Sec. II, an overview of related works is presented. Sec. III contains the statistical analysis. Sec. IV reviews the basics of graph signals and describes the graph construction. Sec. V explains the proposed algorithms which incorporate color-only and geometry-only denoising. Performance evaluation metrics are discussed in Sec. VI. The subjective and objective experimental results are presented in Sec. VII. Finally, conclusions are drawn in Sec. VIII.

## II. RELATED WORK

In the literature, point cloud denoising techniques can be classified into two categories: outlier removal and noise removal, i.e., surface smoothing techniques. Outlier removal techniques can be classified into two main approaches: statistical and model-based.

**Statistical approaches:** The primary purpose of the statistical methods is to remove the outliers based on the distribution of each point with respect to its neighbors or the number of neighbors to each point. The statistical outlier removal approach proposed in [25] computes the mean distance of each point from all its neighbors. The mean and standard deviation identify likely intervals of the global distances, and all the points whose mean distance is outside the defined range are counted as noise and eliminated from the point cloud. An alternate extensively used approach called radius outlier removal (ROR) is based on the number of neighbors; here the number of neighbors of each point is computed in a defined radius and the points whose number of neighbors are less than a threshold are considered outliers and are eliminated from the point cloud [26].

**Model-based approaches:** These approaches employ the notion that noisy points are generally distant from the surface of the object. Due to the unknown underlying surface, the general idea is to approximate the surface with some model, e.g., sphere, plane, square, and so forth, and then compute the distance of each point to the surface of the model. The points having significant distance are considered as noise and removed [27]. A progressive plane algorithm is described in [28], whereby using the average normal and 3D coordinates of a given point set, a plane is estimated. A least-square plane fitting technique is used for computing the distances between each point and the plane and construct a progressive plane; a hybrid algorithm is introduced in [29] based on [28]. The issue with these methods is that in complex geometry, details can be missed because it is hard to fit complex regions with simple models.

Surface smoothing techniques mainly contain moving least squares (MLS), locally optimal projection (LOP)-based methods, sparsity-based methods, and graph based-methods.

**MLS-based methods:** MLS-based methods typically use an estimated smooth surface from the given input to fit the point cloud, and then the points are projected onto the fitted surface. The MLS projection operator in [30] is used by [31] to compute the optimal MLS surface of the point cloud and considered as a reference surface, and then the points are moved around the surface towards it.

Spherical fitting denoising based on MLS algebraic point set surfaces is proposed in [32], along with its variant [33]. This method overcomes unstable reconstruction in case of high curvature and enhances stability at a low sampling rate in comparison to the MLS-based approach. Several extensions of MLS such as robust MLS [34] and robust implicit MLS [35] have also been proposed. These MLS-based methods can provide a smooth surface from significantly noisy input but are usually prone to over-smoothing, and are very sensitive to outliers [17], [18].

**LOP-based methods:** Unlike MLS-based methods, these methods do not measure specific surface parameters; LOP [36] enforces uniform distribution over the given input point cloud and provides a set of points that represent the underlying surface. Its variant weighted LOP (WLOP) [37] provides a more uniformly distributed output by modifying the repulse term according to the local density. Moreover, anisotropic WLOP [38] re-models WLOP by using an anisotropic weighting function for better preservation of sharp features. Due to the use of local operators, LOP-based methods are also affected by over-smoothing [17], [18].

**Sparsity-based methods:** Sparsity-based methods seek a sparse representation of some features of geometry based on local planarity assumption. These methods solve a global minimization problem in $l_1$ [39] or $l_0$ [17] norm for the sparse reconstruction of surface normals and minimize a cost function to update the point positions with surface normals. Moving robust principal components analysis [16] is an efficient method for solving a minimization problem to update the position of each point. Weighted $l_1$ minimization helps to preserve sharp features. These approaches provide state-of-the-art performance [40]; however, the normal approximation can be affected by high-level noise, which leads to over-sharpening and over-smoothing [17].

**Graph-based methods:** Recently, graph-based techniques have been used for denoising point clouds. The conventional approach is to construct a $k$ nearest neighbor ($k$-NN) graph as it makes geometric structure explicit [41]. The points in a given point cloud are considered as nodes, and each node is connected through edges to its $k$ nearest neighbors with weights that reflect inter-node similarities based on geometric information [42], [43]; an iterative regularization technique (IBR) is employed to enforce smoothness on the geometry-only graph signal [42]. Manifold denoising through spectral graph wavelet (MSGW) [44] also employed a geometry-only graph. The robust denoising of piece-wise smooth manifolds (RPSM) is performed using a local tangent space-based graph [45]. The disadvantage of these approaches is that holes are typically formed in the denoised output (see Fig. 1) as the correct position of a point is estimated based on the *noisy* geometry. This can lead to errors in estimating the local surface as the information of the manifold is based only on geometry. Data-driven methods [7], [46] for point cloud denoising have presented good performance, still may not be convenient to apply where no dataset of ground-truth point clouds is available. An extension of the IBR is used in [23], where the convex optimization is performed for denoising the geometry of a point cloud using the geometry/color graph signal. In contrast to an iterative method, a non-iterative technique for point cloud denoising by exploiting the same geometry/color graph signal is presented in [22] using the manual setting of threshold corresponding to the noise level. In this article, we employ the same method as in [44], but we carefully exploit both geometry and color to move each point to its actual position, preventing the artifacts caused by other denoising techniques employing only geometry information and perform adaptive thresholding for the noise removal contrary to the thresholding technique described in [22].

More particularly, the proposed technique constructs a joint geometry/color graph, followed by employing the SGW to the respective graph signal. Point cloud denoising based on SGW is very effective, as the constructed graph compacts most of the energy of smooth graph signals in lower frequencies. Moreover, the joint geometry and color graph offers better spectral separation between noise and signal, helping in noise removal. Mostly recent point cloud denoising are iterative based techniques and may also be susceptible to various parameters selection [16], [32], [35], [42]. On the contrary, the proposed framework is non-iterative and computationally less expensive, which scales linearly with the size of data; and independent of parameter selection.

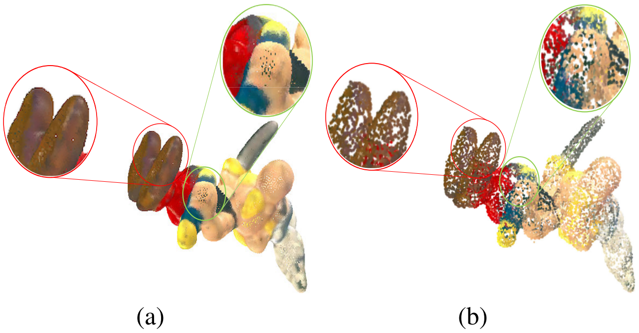


**FIGURE 1.** Asterix model: (a) Noise-free input, (b) denoised with IBR [42]. The noisy points are relocated to their closest neighbors rather than their original locations, generating holes in the resulting denoised point cloud.

The proposed algorithm is different from the technique described in [44] in the following ways:

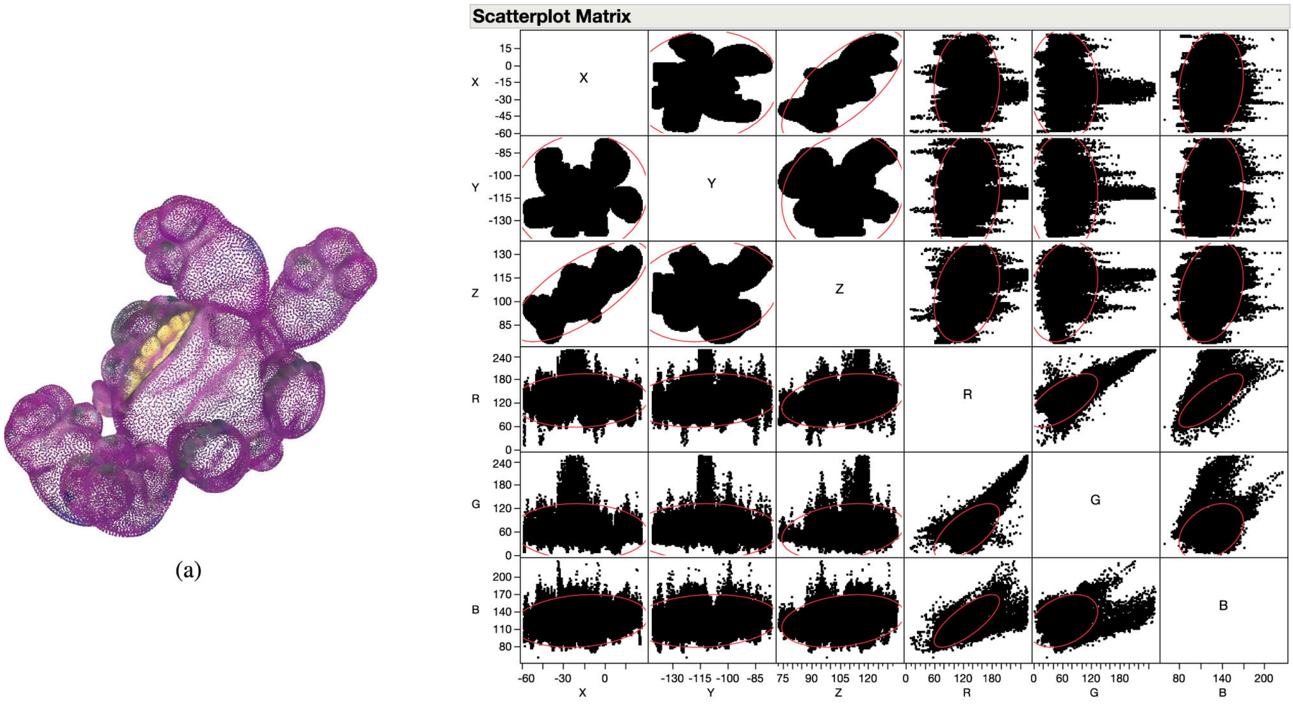
- The construction of the graph in [44] is based on geometry-only, and hence the resulting denoised point clouds have artifacts. In the proposed algorithm, we exploit the correlation between geometry and color in the graph construction, which strongly reduces the artifacts.
- In the algorithm described in [44], prior knowledge of standard deviation is needed for performing the soft-thresholding, which can work with synthetic point clouds. Still, in real-world point clouds, the standard deviation is unknown. In our technique, we used data-driven adaptive soft-thresholding, in which the prior knowledge of standard deviation is not required.
- The proposed algorithm is also used for color denoising of the point cloud with a different set of weights based on the correlation of the geometry and color. The algorithm in [44] does not perform color denoising.

Extensive experimental findings depict that the proposed method effectively utilizes the correlation between geometry and color to move each point to its original position, avoiding creating holes caused by other recent approaches. We present that the proposed method outperforms state-of-the-art techniques using both subjective and objective quality metrics.

## III. STATISTICAL ANALYSIS

In this section, we report the statistical analysis results to study the correlation between geometry and color in a point cloud. The idea of using joint geometry/color graph construction is based on the notion that on a smooth surface, the color is typically smooth; hence the correct location of a point also depends on the color attribute and not only the geometry of the neighboring points, as well as on the geometry of the neighbors.

A point cloud is represented as $\mathcal{P} = \{p_1, p_2, p_3, \ldots\ldots, p_N\}$ with $p_i \in \mathbb{R}^6$ containing 3D geometry and RGB color information for point $p_i$. The six-dimensional feature of each point is $p_i = [X_i, C_i]$, where $X_i = [x_i\ y_i\ z_i] \in \mathbb{R}^3$ is the geometric



(a)

(b)

**FIGURE 2.** (a) 4arms_monster ground-truth model; (b) Scatterplot Matrix of the correlation result of the geometry and color of (a) shown in Tab. 1.

**TABLE 1.** Correlation coefficient $\rho$ between geometry and color of *4arms_monster* ground-truth model.

| | $x$ | $y$ | $z$ | $R$ | $G$ | $B$ |
|---|---|---|---|---|---|---|
| $x$ | 1.00 | 0.07 | 0.71 | 0.07 | 0.06 | 0.16 |
| $y$ | 0.07 | 1.00 | 0.14 | 0.15 | 0.01 | 0.12 |
| $z$ | 0.71 | 0.14 | 1.00 | 0.28 | 0.25 | 0.25 |
| $R$ | 0.07 | 0.15 | 0.28 | 1.00 | 0.65 | 0.77 |
| $G$ | 0.06 | 0.01 | 0.25 | 0.65 | 1.00 | 0.36 |
| $B$ | 0.16 | 0.12 | 0.25 | 0.77 | 0.36 | 1.00 |

**TABLE 2.** Correlation coefficient $\rho$ between geometry and color of a planar patch in *Arco_valentino* model.

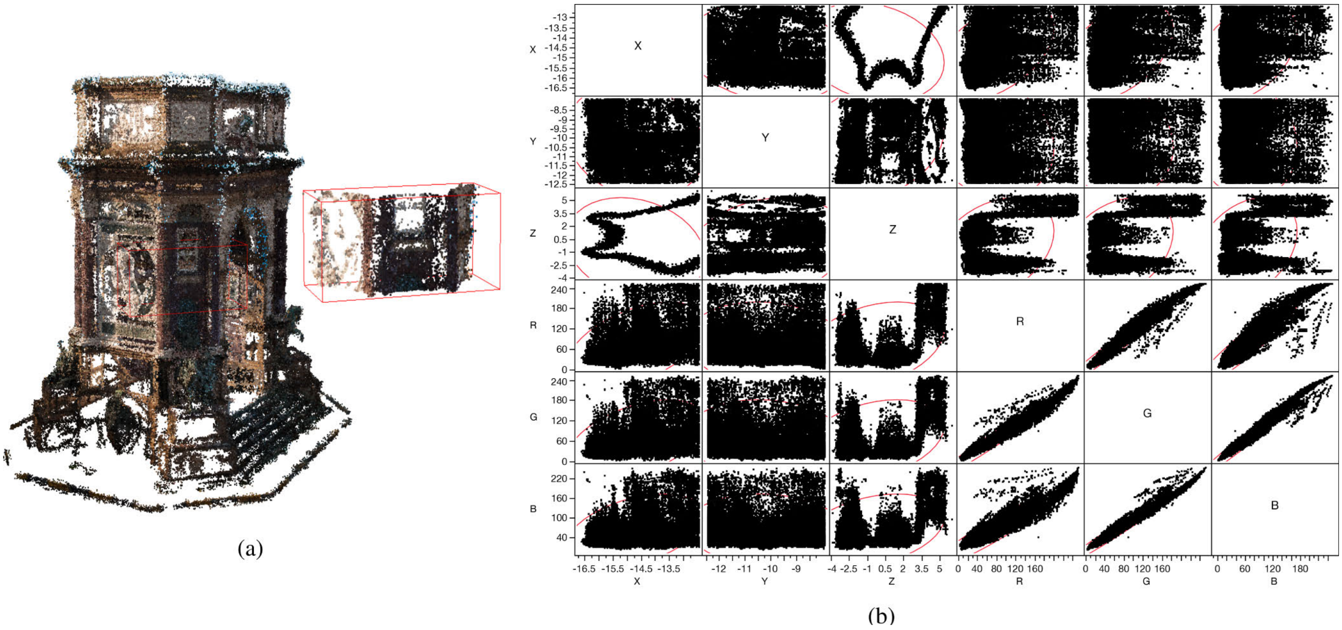
| | $x$ | $y$ | $z$ | $R$ | $G$ | $B$ |
|---|---|---|---|---|---|---|
| $x$ | 1.00 | -0.17 | -0.21 | 0.49 | 0.47 | 0.41 |
| $y$ | -0.17 | 1.00 | 0.20 | 0.12 | 0.11 | 0.09 |
| $z$ | -0.21 | 0.20 | 1.00 | 0.35 | 0.34 | 0.34 |
| $R$ | 0.49 | 0.12 | 0.35 | 1.00 | 0.98 | 0.94 |
| $G$ | 0.47 | 0.11 | 0.34 | 0.98 | 1.00 | 0.99 |
| $B$ | 0.41 | 0.09 | 0.34 | 0.94 | 0.99 | 1.00 |

coordinate vector and $C_i = [R_i \; G_i \; B_i] \in \mathbb{R}^3$ are the color attributes. We compute the linear correlation coefficient $\rho$ between each independent variable of $X_i$ and $C_i$. If there is an exact linear relationship between two variables, $\rho$ is equal to 1 or -1 depending on whether the variables are positively or negatively related. If there is no linear relationship, the correlation tends towards zero.

Fig. 2-a shows the synthetic point cloud used for this analysis. In Fig. 2-b, each scatterplot box is illustrated with a bivariate normal density ellipse.

The results in Tab. 1 and Fig. 2 show the relationship of individual geometry coordinate to itself (diagonal values of $\rho$ in Tab. 1 and to other variables. They show that there exists some correlation between each geometry coordinate and every individual color attribute of a point. In order to quantify the relationship between the geometry and color more precisely and gain some intuition, it is useful to compute the correlation for a real-world point cloud. A planar patch is taken into account from an *Arco_valentino* shown in Fig. 3-a, for which the correlation coefficient between geometry and color is computed to obtain more meaningful results. The results in Tab. 2 and the scatterplot shown in Fig.3-b indicate that the data points for the individual geometry and color component are tightly clustered, suggesting that the geometry and color are highly correlated. This justifies exploiting this correlation for point cloud denoising.

In Fig. 2-b and Fig. 3-b, each pair-wise correlation of an individual attribute is shown. The lower and upper triangle across the diagonal is similar. Each pair-wise scatterplot matrix represents the bi-variate normal distribution and is computed using Pearson product-moment correlation (PPMC), which measures the strength of the linear



(a)

(b)

**FIGURE 3.** **(a) Arco_valentino model; (b) Scatterplot Matrix of the correlation result of the geometry and color of (a) shown in Tab. 2.**

relationship between the corresponding two variables. The red ellipses in each scatterplot box encompass approximately 95% of the points. These ellipses reflect the correlation between the variables.

## IV. PROPOSED METHOD - GRAPH CONSTRUCTION

### A. GRAPH NOMENCLATURE

An undirected weighted graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ contains a confined set of $m$ vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$ denoted as $(i, j, w_{i,j})$, where $i, j \in \mathcal{V}$ and an individual edge has a positive weight $w_{i,j} \in \mathbb{R}^+$ that indicates the affinity between node $i$ and $j$. The respective *adjacency matrix* $W(i, j) = w_{i,j}$ is a real symmetric $m \times m$ matrix. Let the diagonal degree matrix $\mathbf{D}$ with entries $\mathbf{D}_{i,j} = \sum_j w_{i,j}$. The combinatorial graph Laplacian matrix is computed as $\mathbf{L} = \mathbf{D}$ - $\mathbf{W}$ with the given $\mathbf{W}$ and $\mathbf{D}$. The graph signal $g(\mathcal{G})$ for a given graph $\mathcal{G}$ is defined on the vertices of a graph, as $g : \mathcal{V} \rightarrow \mathbb{R}^D$ for some dimension $D$.

### B. JOINT GEOMETRY AND COLOR *k*-NN GRAPH

The generalized procedure is to construct a k-NN graph based on Euclidean distance to make an explicit geometric structure [41]. The proposed denoising algorithm constructs a $k$-NN graph taking into account both the color similarity and coordinates proximity of point $p_i \in \mathcal{P}$.

Construction of $k$-NN graph based on color-only is not a good option, since points with the same color and geometrically far may have distinct semantic content, which results in the generation of a wrong graph. However, the construction of geometry-only based $k$-NN graph causes adverse effects as anticipated in Sec. I. The joint geometry and color graph is effective as it uses additional statistics about the point cloud.

The weight of the edge between node $i$ and $j$ for a given point cloud $\mathcal{P}$ has to be determined in order to construct such a graph. A common choice is the threshold Gaussian kernel [47]:

$$w_{i,j} = \begin{cases} \exp\left(-\frac{\|X_i - X_j\|^2}{2\theta_X^2} - \frac{\|C_i - C_j\|^2}{2\theta_C^2}\right) & \text{if } p_j \in \phi_k(i) \\ & \text{or } p_i \in \phi_k(j) \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Here, $\theta_X$ and $\theta_C$ represent the respective contribution of geometry and color attribute in the joint geometry/color graph constructionfi. $\phi_k(i)$ and $\phi_k(j)$ represent the set of $k$ nearest neighbors to point $p_i$ and to point $p_j$, respectively. The resulting $k$-NN graph is denoted as $\mathcal{G}$.

## V. PROPOSED METHOD - DENOISING

### A. SPECTRAL GRAPH WAVELET PRELIMINARIES

In this section, we set up a mathematical notation and the definition of SGW [21]. The Graph Fourier Transform (GFT) $\hat{g}$ for a function $g(\mathcal{G})$ is defined as $\hat{g}(l) = \langle \chi_l, g \rangle = \sum_{i=1}^{m-1} g(i)\chi_l(i)$, where the eigenvectors of the graph Laplacian $\mathbf{L}$ is represented by $\chi_l$ and eigenvalues are denoted as $\lambda_l$ for $l = 0, \ldots, m-1$. SGW [21] establishes a scaling operator in the Graph Fourier domain based on $\lambda_l$. Particularly, SGW are determined using a kernel $f$, the wavelet operator $T_f = f(L)$ acts on a given function $g$ by modulating each Fourier mode as:

$$\widehat{T_f\ g}(l) = f(\lambda_l)\hat{g}(l)\,. \tag{2}$$

The inverse transform is defined as:

$$(T_f\ g)(n) = \sum_{l=0}^{m-1} f(\lambda_l)\hat{g}(l)\chi_l(n). \tag{3}$$

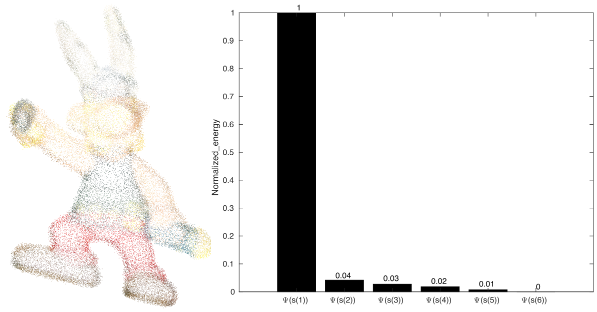
At scale $s$, the wavelet operator is defined as $T_f^s = f(sL)$. SGW is then calculated by localizing the operators $T_f^s$ by enforcing them to the impulse $\delta$ on a single vertex: $\psi_{s,n} = T_f^s \delta_n$. For a given graph signal $g$, the wavelet coefficients are determined by taking the inner product with these wavelets as $\Psi_g(s,n) = \langle \psi_{s,n}, g \rangle$. $\Psi_g(s,n)$ can be estimated from the wavelet operators $T_f^s$ using the orthonormality of $\chi$ as:

$$\Psi_g(s,n) = (T_f^s g)(n) = \sum_{l=0}^{m-1} f(s\lambda_l)\hat{g}(l)\chi_l(n). \tag{4}$$

The naive technique for calculating SGW from Eq. 4 needs explicit computation of the entire $\chi$, which scales inadequately for large graphs. Another method for direct computation of SGW transform is to diagonalize $\mathbf{L}$, which is possible only for the small graphs having less than a thousand vertices [21]. For the real-world and synthetic point clouds consisting of a large number of points, the SGW can be computed with a fast algorithm via low order polynomials, which estimate the scaled generating kernels. The wavelet coefficients can be determined at every single scale by applying a polynomial of $\mathbf{L}$ to the underlying data. This leads to lower computational cost when the graph is sparse. To define the transform in graph spectral domain, direct calculation of Laplacian operator via diagonalization is infeasible for problems with size exceeding a few thousand vertices. Hence, the wavelet and scaling coefficients of a given input $[m \times D]$ graph signal are efficiently calculated by employing Chebyshev polynomial approximation [21], [48] to avoid the need for explicit diagonalization of the graph Laplacian and then mapped these coefficients to $[m{*}(I-1) \times D]$, where $I$ denotes the number of wavelet decomposition levels.

### B. GEOMETRY DENOISING

A primary aim of the proposed algorithm is to perform geometry denoising by taking advantage of the constructed joint geometry/color $k$-NN graph $\mathcal{G}$. The given noisy coordinates of each point $p_i$ can be described as $p_i = [X_i + n_i,\ \ C_i]$, $X_i$ being the undetermined original position of a point $p_i$ and $n_i$ the geometry noise, with $X_i,\ n_i \in \mathbb{R}^3$. The goal is to approximate $X_i$ for each point in a given point cloud. This can be achieved by using the proposed denoising algorithm based on SGW. After an undirected graph $\mathcal{G}$ construction using Eq. 1 and defining the graph Laplacian $L$ from $\mathbf{W}$, we compute the SGW transform by employing low-order polynomials to establish compact vertex localization of SGW coefficients. For the respective noisy signal $X_i$, SGW coefficients are computed for each SGW band, i.e., $\Psi_{X_i}(s(i))$ for $1 \leq i \leq I$, preserving all the wavelet and scaling coefficients, which corresponds to a low-frequency wavelet band $s$. Denoising of $\Psi_{X_i}(s(i), n)$ is then performed by an adaptive data-driven wavelet soft-thresholding [49] based on the characteristic that low-frequency spectral wavelet bands contained most of the energy of the signal as anticipated in Sec. I. The distribution of the energy of signal for a noisy *Asterix* model is shown in Fig. 4-b. Finally, the denoised point cloud $\mathcal{Q}$ is obtained by taking an inverse spectral wavelet transform of the denoised SGW coefficients $\Psi^*_{X_i}(s(i))$. The data-driven adaptive threshold $\tau$ is computed as in Sec. V-D.



(a) (b)

**FIGURE 4. (a) Noisy Asterix model with $\mu = 0, \sigma = 0.2$ (b) Normalized energy of $\Psi_{X_i}(s(i), n)$, where $1 \leq i \leq I$ and $n = 1 \ldots N$.**

In order to compare the proposed geometry denoising algorithm with MSGW [44] and IBR [42], outlier removal is needed. Outliers have distinct features in that they have a sparse neighborhood; therefore, outliers detection and removal are density-based [27], [42]. We follow the ROR method, in which a sphere with a radius $r$ is formed having each point $p_i$ as its centre. The sphere contains $u_i$ number of points. We then compute $\bar{u} = \frac{\sum_{i=1}^{N} u_i}{N}$, where $N$ is the total number of points. A point $p_j$ is considered as an outlier if $u_j < \bar{u}$.

### C. COLOR DENOISING

For the color denoising problem, we assume that the geometry of the point cloud is noise-free or has been denoised using one of the existing denoising algorithms [19], [42], [44], [45].

We perform color denoising by exploiting the graph $\mathcal{G}$ from geometry and color information of the noisy point cloud. Each vertex of $\mathcal{G}$ is associated with the geometry $X_i$ and color $C_i$ information of the corresponding point $p_i$. The input noisy point cloud consists of both geometry and color. Each point can be expressed as $p_i = [X_i,\ C_i\ +\ n_i]$, $X_i$ being the geometry, $C_i$ being the unknown true color and $W_i$ the color noise. The objective is to estimate $C_i$ for each point of the point cloud. After the construction, the graph $\mathcal{G}$ using Eq. 1 with the appropriate choice of $\theta_X$ and $\theta_C$ for the color denoising, we compute the SGW coefficients for each SGW band, i.e., $\Psi_{C_i}(s(i))$, where $1 \leq i \leq I$ for the corresponding noisy signal $C_i$ and then denoising is performed by following the procedure described in Sec. V-B.

### D. SELECTION OF DATA-DRIVEN ADAPTIVE THRESHOLD

Here, we focus on the estimation of parameters $\sigma_X$ and $\sigma_C$ for geometry and color denoising, respectively, which in turn yields an estimation of individual thresholds $\tau(\sigma_X)$ and $\tau(\sigma_C)$ for geometry and color denoising application. These estimated thresholds are adaptive to various sub-band characteristics

The geometry/color noise variance $\sigma^2$ is required to be estimated initially, which can be done using robust median estimator as in [50], [51]:

$$\hat{\sigma} = \frac{\text{Median}(|\Psi_{X_i}(s(i))|)}{0.6745}, \quad \text{for } 5 \le i \le I. \tag{5}$$

The variance of the transform coefficients $\sigma^2_{\Psi_{X_i}}$ can be found empirically:

$$\hat{\sigma}^2_{\Psi_{X_i}} = \frac{1}{n}\sum_{i=1}^{3n} \Psi^2_{X_i}, \tag{6}$$

where $n = m \times (I-1)$. The threshold is thus computed using:

$$\tau(\hat{\sigma}_X) = \frac{\hat{\sigma}^2}{\hat{\sigma}_X}, \tag{7}$$

where

$$\hat{\sigma}_X = \sqrt{\max(\hat{\sigma}^2_{\Psi_{X_i}} - \hat{\sigma}^2), 0.} \tag{8}$$

To compute $\tau(\sigma_C)$, we need to replace $\Psi_{X_i}$ by $\Psi_{C_i}$ in Eq. 5, Eq. 6, and Eq. 8, and replace $\sigma_X$ by $\sigma_C$ in Eq. 7.

## VI. EVALUATION METRICS

### A. COLOR DENOISING

The metrics used for the objective evaluation of the proposed color denoising algorithm are mean-squared-error (MSE) and peak signal-to-noise ratio (PSNR):

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} \|p_i - \widehat{p}_i\|^2, \tag{9}$$

$$\text{PSNR} = 10\log_{10}\left(\frac{255^2}{\text{MSE}}\right). \tag{10}$$

where $p_i$ and $\widehat{p}_i$ represent the points in ground-truth and color denoised point cloud, respectively, and $N$ is the number of points in point cloud $\mathcal{P}$.

### B. GEOMETRY DENOISING

For image denoising, quality metrics are based on a one-to-one correspondence between ground-truth and denoised data samples. However, in the case of point clouds, such constraint would be practically too restrictive. The Hausdorff distance overcomes the notion of a vertex to vertex distance [52]–[55]. A triangular mesh $\mathcal{M}$ consists of a set $p_i$ of points and a set $\mathcal{T}$ of triangles defining how the vertices from $p_i$ are associated together denoted as $\mathcal{M} = (p_i, \mathcal{T})$. We also consider a denoised point cloud $\mathcal{Q}$. We are interested in measuring the distance between sets of points in the two point clouds.

#### 1) HAUSDORFF DISTANCE

The distance $d(\widehat{X}_i, \mathcal{M})$ between a given point $\widehat{X}_i \in \mathcal{Q}$ and any point $X_i \in \mathcal{M}$ is defined as:

$$d(\widehat{X}_i, \mathcal{M}) = \min_{X_i \in \mathcal{M}} \|X_i - \widehat{X}_i\|_2. \tag{11}$$

The Hausdorff distance between a point $\widehat{X}$ and $\mathcal{M}$ is denoted as $d_H(\widehat{X}, \mathcal{M})$ and is given by:

$$d_H(\widehat{X}, \mathcal{M}) = \max_{\widehat{X}_i \in \widehat{p}_i} d(\widehat{X}_i, \mathcal{M}), \tag{12}$$

$d_H(\widehat{X}, \mathcal{M})$ and $d_H(\mathcal{M}, \widehat{X})$ are referred to as forward and backward distance, respectively. These distances are not symmetrical, i.e., $d_H(\widehat{X}, \mathcal{M}) \neq d_H(\mathcal{M}, \widehat{X})$. The symmetrical Hausdorff distance $d_S(\widehat{X}, \mathcal{M})$ can be computed as:

$$d_S(\widehat{X}, \mathcal{M}) = \max\,(d_H(\widehat{X}, \mathcal{M}), d_H(\mathcal{M}, \widehat{X})). \tag{13}$$

The distance between any point $X_i$ belonging to $\mathcal{M}$ and $\widehat{X}$ can be computed analytically, as it can be reduced to the minimum of the distances between $X_i$ and all the triangles $T \in \mathcal{T}$. If the orthogonal projection $\widehat{X}_i$ of $X_i$ on the plane of $T$ is inside the triangle, the point-to-triangle distance is nothing but a point-to-plane distance. When the projection lies outside $T$, the point-to-triangle distance is the distance between $X_i$ and the closest point $\widehat{X}_i'$ of $T$, which lies necessarily on one of the sides of $T$ [52]–[54].

The point-to-mesh distance in Eq. 11 can also be used to calculate the mean distance $d_m$ between $\widehat{X}$ and $\mathcal{M}$:

$$d_m(\widehat{X}, \mathcal{M}) = \frac{1}{N}\sum_{\widehat{X}_i \in \widehat{X}} d(\widehat{X}_i, \mathcal{M}), \tag{14}$$

$$\zeta = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(d_i - d_m)^2}, \tag{15}$$

where $d_i = \|p_i - q_i\|^2$; $\zeta$ represents the standard deviation of the distance between the point $q_i$ and the corresponding point $p_i$. We computed the Hausdorff distance using the cloud-to-mesh (C2M) metric in CloudCompare [56]. The ground-truth 3D models act as the reference meshes to their respective denoised point clouds. The outputs of C2M metric are $d_H$, $d_m$.

We further verify the results by employing the same objective quality metrics used in [57]. Assume $\mathcal{P}$ and $\mathcal{Q}$ represent the geometry of the noise-free and denoised point cloud respectively, where $\mathcal{P} = \{p_i\}_{i=1}^{N_1}$ $\mathcal{Q}=\{q_i\}_{i=1}^{N_2}$, such that $p_i, q_i \in \mathbb{R}^3$. We define distance metrics as follows.

Mean-square-error (MSE): It is calculated as an average of the squared Euclidean distance between each point in $\mathcal{P}$ and its respective closest point in $\mathcal{Q}$, and also between each point in $\mathcal{Q}$ and its corresponding nearest point in $\mathcal{P}$:

$$\text{MSE} = \frac{1}{2N_1}\sum_{p_i \in \mathcal{P}} \min_{q_i \in \mathcal{Q}} \|p_i - q_i\|_2^2 + \frac{1}{2N_2}\sum_{q_i \in \mathcal{Q}} \min_{p_i \in \mathcal{P}} \|q_i - p_i\|_2^2. \tag{16}$$

Mean city-block distance (MCD): MCD uses $l_1$ norm instead of $l_2$ norm.

$$\text{MCD} = \frac{1}{2N_1}\sum_{p_i \in \mathcal{P}} \min_{q_i \in \mathcal{Q}} \|p_i - q_i\| + \frac{1}{2N_2}\sum_{q_i \in \mathcal{Q}} \min_{p_i \in \mathcal{P}} \|q_i - p_i\|. \tag{17}$$

**TABLE 3. Parameter values used for the proposed geometry denoising algorithm, MSGW [44], IBR [42] and RPSM [45].**

| Parameter | Proposed | MSGW [44] | IBR [42] | RPSM [45] |
|---|---|---|---|---|
| $k$ (Synthetic Point cloud) | 30 | 60 | 20 | 10 |
| $k$ (Natural Point cloud) | 10 | 20 | 20 | – |
| $\theta_X$ | 0.8 | – | – | – |
| $\theta_C$ | 0.2 | – | – | – |
| $\gamma$ (Synthetic Point cloud) | – | – | $0.50\ (\sigma = 0.2,\ 0.3)$<br>$0.70\ (\sigma = 0.4)$ | – |
| $\gamma$ (Natural Point cloud) | – | – | 0.1 | – |
| $\tau$ (Synthetic Point cloud) | – | $0.2\ (\sigma = 0.2)$<br>$0.3\ (\sigma = 0.3)$<br>$0.4\ (\sigma = 0.4)$ | – | – |
| $\tau$ (Natural Point cloud) | – | 0.4 | – | – |

**TABLE 4. Parameters values used for color denoising using various algorithms.**

| Parameters | Proposed | Tikhonov & TV | GLR [19] | GTV [19] |
|---|---|---|---|---|
| $k$ (Synthetic point cloud ) | 8 | 20 | 8 | 8 |
| $k$ (Natural point cloud) | 8 | 25 | – | – |
| $\theta_X$ | 5 | 0.88 | 3 | 2 |
| $\theta_C$ | 50 | 3.5 | 100 | 100 |
| $\gamma$ (Synthetic point cloud) | – | $0.05\ (\sigma = 10)$<br>$0.07\ (\sigma = 15)$<br>$0.10\ (\sigma = 20)$<br>$0.13\ (\sigma = 25)$<br>$0.19\ (\sigma = 30)$<br>$0.26\ (\sigma = 40)$ | $0.3\ (\sigma = 10,\ 15)$<br>$0.5\ (\sigma = 20\ ,30)$ | $15\ (\sigma = 10,\ 15)$<br>$25\ (\sigma = 20\ ,30)$ |
| $\gamma$ (Natural point cloud) | – | 1.5 | – | – |
| $\rho\ ,t$ | – | – | – | 5 ,0.1 |

For performance comparison between the proposed algorithm and RPSM [45]. Due to the large memory requirement of RPSM, we performed experiments on sub-sampled point clouds, for which the sub-sampled ground-truth mesh is not available. Hence, we only compute MSE and MCD using Eq. 16 and Eq. 17 on these sub-sampled point clouds.

## VII. EXPERIMENTAL RESULTS

This section focuses on the analysis of the experimental results, both visual and objective, on static real-world point clouds available in the JPEG PLENO (GTI-UTM) database [58] and Greyc dataset of synthetic point clouds [59]. Each point cloud in real-world and synthetic dataset consisting of geometry and color attributes.

### A. EXPERIMENTAL SETUP

We defined $I = 6$ wavelet decomposition levels, and maintain the wavelet coefficients of a scale $s(1)$ and then preserve the wavelet coefficients that exceeds the threshold $\tau(\sigma_X)$ which is calculated using Eq. 7 for the corresponding noise level in $s(i)$ for $2 \leq i \leq I$. To denoise the point cloud locally by estimating the SGW coefficients, the order of Chebyshev polynomial approximation is set to $k/2$ for a $k$-NN graph. The whole set of parameters used in this article required for geometry-only and color-only denoising scenario are given in Tab. 3 and Tab. 4, respectively.

### B. VISUAL INSPECTION OF GEOMETRY DENOISING ALGORITHM

The visual inspection of geometry denoising algorithms has been performed for both synthetic and natural point clouds.

#### 1) GEOMETRY DENOISING OF NATURAL POINT CLOUDS

We exhibit a visual comparison of the proposed point cloud denoised algorithm with the denoising technique that constructs a graph from geometry only and performs regularization as in [42], along with the algorithm described in [44]. The proposed algorithm is applied to real-world natural point clouds, for which we do not have a noiseless reference; thereby, the results are only qualitative. Fig. 5-a and Fig. 6-a present the point clouds with real noise; it can be observed that the points with similar color are typically dispersed in a small neighborhood in the acquisition process, which may blur the details. The resulting output after outlier removal is shown in Fig. 5-b and Fig. 6-b, a prior step required for denoising algorithms such as MSGW [44] and IBR [42]. The resulting denoised point cloud using the proposed method is reported in Fig. 5-c and Fig. 6-c. Here, it can be seen that the noisy points are relocated close to their true position by taking advantage of the correlation of their geometry and color, thus preserving the details which were hidden in the noisy input point cloud. The adaptive thresholding is very helpful for denoising the real-world point cloud, for which prior knowledge of the standard deviation of the noise is not required and is estimated using Eq. 5. Fig. 5-d and Fig. 6-d show the denoised point cloud using MSGW; it appears that the technique is not preserving the details effectively in the geometry irrespective of employing the outlier removal step before performing denoising.

Furthermore, prior knowledge of noise-level ($\sigma$) is required to apply soft-thresholding in [22], which is unknown for real-world point clouds and geometry denoising depends on selection of the $\sigma$ for soft-thresholding. Fig. 5-e and Fig. 6-e depict the resulting denoised output using IBR [42] applied to the outlier-free input; it can be noted in the same region that, considering no color information, the noisy points are not relocated to their original location with respect to the proposed algorithm. However, IBR performs better than MSGW in denoising but enlarging gaps due to over-regularization and typically providing a noisier result near object boundaries. This is an iterative-based algorithm,

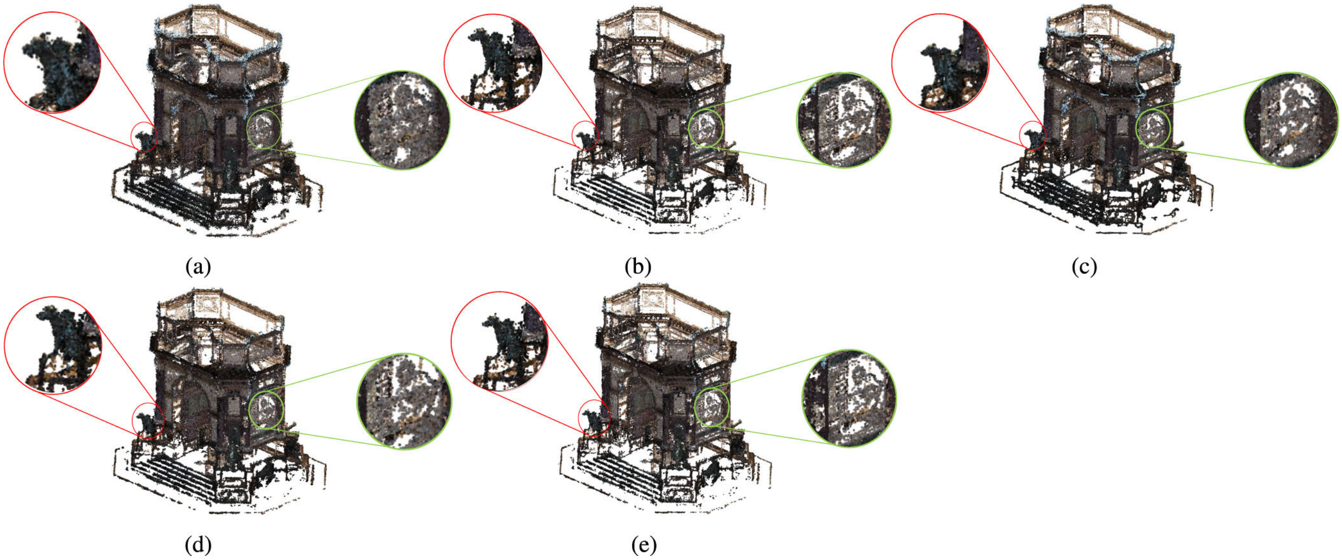


**FIGURE 5.** Arco_Valentino model. (a) Noisy input (b) outlier-free input, denoised results by (c) proposed algorithm, (d) MSGW [44] applied to outlier-free input, and (e) IBR performed after the outlier removal step [42].

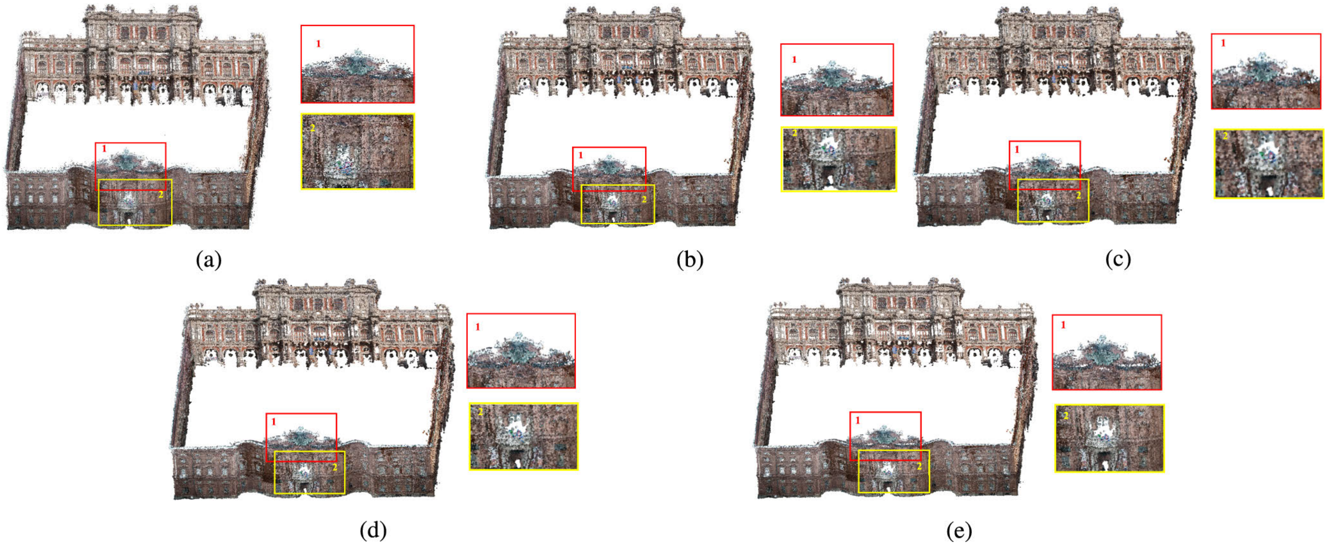


**FIGURE 6.** Palazzo_Carignano model. (a) Noisy input (b) outlier-free input, denoised results by (c) proposed algorithm, (d) MSGW [44] applied to outlier-free input, and (e) IBR performed after the outlier removal step [42].

and for real-world point cloud, it is computationally very expensive; furthermore, the outlier removal step is essential for both MSGW and IBR, which turns these into more computational complex algorithms.

### 2) GEOMETRY DENOISING OF SYNTHETIC POINT CLOUDS

The experiments have also been performed on noise-free point clouds from the *Greyc* dataset [59], corrupted with Gaussian synthetic geometry noise with $\mu = 0$ and $\sigma = 0.2$, 0.3, and 0.4. In Fig. 7, the results are presented in two rows for *4arms_monstre* and *Asterix* models, respectively. For each point cloud model in Fig. 7, the first row of figures is their natural representation. The artifacts (i.e., holes formation) can be clearly seen in the resulting denoised point clouds using MSGW and IBR with reference to the proposed algorithm. Moreover, an alternate view is provided in the second and fourth rows; they are the zoomed-in editions of the similar point clouds shown with the identical size as in the first and third row. This sort of visual representation is more sparse and allows to differentiate better the noise removal and fine details at the boundaries. Fig. 7-a shows the ground-truth point cloud models. Fig. 7-b presents the noisy point clouds with $\sigma = 0.3$. The resulting denoised point clouds using the proposed algorithm are shown in Fig. 7-c; it can be observed that the geometry noise has been regularized, and the noisy points are proximate to their actual

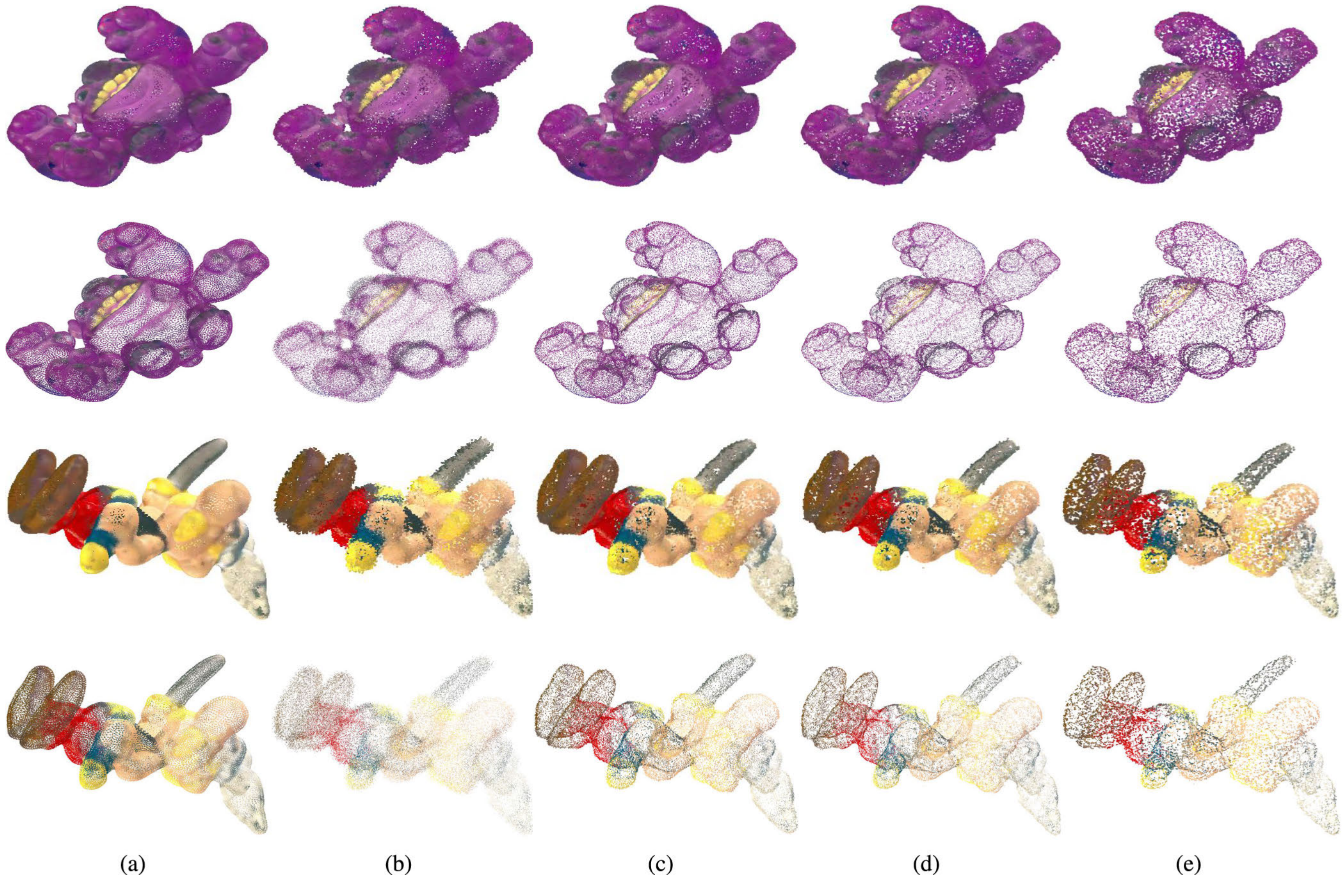


**FIGURE 7.** Point cloud models with color: (a) ground-truth, (b) noisy input ($\mu = 0$ and $\sigma = 0.3$), denoised results by (c) proposed algorithm, (d) MSGW [44], and (e) IBR [42].

positions with the less adverse effect of holes formation. The denoised output of MSGW is shown in Fig. 7-d; it can be clearly seen that the geometry noise is not properly removed and also causes the opening of holes in the output point clouds. Fig. 7-e depicts the resulting denoised point clouds obtained by the IBR algorithm. It is evident that the geometry is not entirely regularized with respect to the proposed algorithm, while still better than MSGW; nevertheless, the resulting output of the IBR algorithm has large holes; the reason is the $\gamma$ value, higher $\gamma$ is appropriate for denoising well, but it causes severe artifacts. In general, it can be noted from the visual results of both real-world and synthetic point clouds that the denoised point clouds using the proposed algorithm have superior quality and fewer artifacts.

A comparative study has also been conducted with respect to RPSM [45], and the outcomes are shown in Fig. 8. In this specific case, we have applied the proposed algorithm on sub-sampled point clouds from the same dataset because of the large memory requirement of RPSM as anticipated in Sec. VI-B. The sub-sampling performed here is on a spatial basis, where the average minimum distance between the two points in each point cloud is set to 0.80. The number of points in each point cloud model of the Greyc dataset [59] is around 20,000 on average. The results of the proposed algorithm and RPSM [45] on sub-sampled *4arms_monstre* and *Asterix* are presented in two rows for each point, respectively. The proposed algorithm and RPSM [45] are applied to the sub-sampled noisy inputs shown in Fig. 8-b; the corresponding ground-truth point clouds are shown in Fig. 8-a. The denoised outputs of the proposed algorithm are shown in Fig. 8-c; it can be observed that the proposed algorithm performs better at geometry denoising with very few artifacts, and the resulting denoised outputs of RPSM [45] are shown in Fig. 8-d; it can be seen that RPSM [45] over-regularizes the sub-sampled point clouds which tends to generate the holes in the resulting denoised point cloud.

### C. VISUAL INSPECTION OF COLOR DENOISING ALGORITHM

The visual inspection of color denoising algorithms has been performed for both synthetic and real-world point clouds. The proposed color denoising technique has been compared with the color denoising method described in [24], where the color of the point clouds is denoised using a different set of parameters $\theta_X$ and $\theta_C$. Graph gradient was used to measure the degree of smoothness of a geometry/color graph signal.

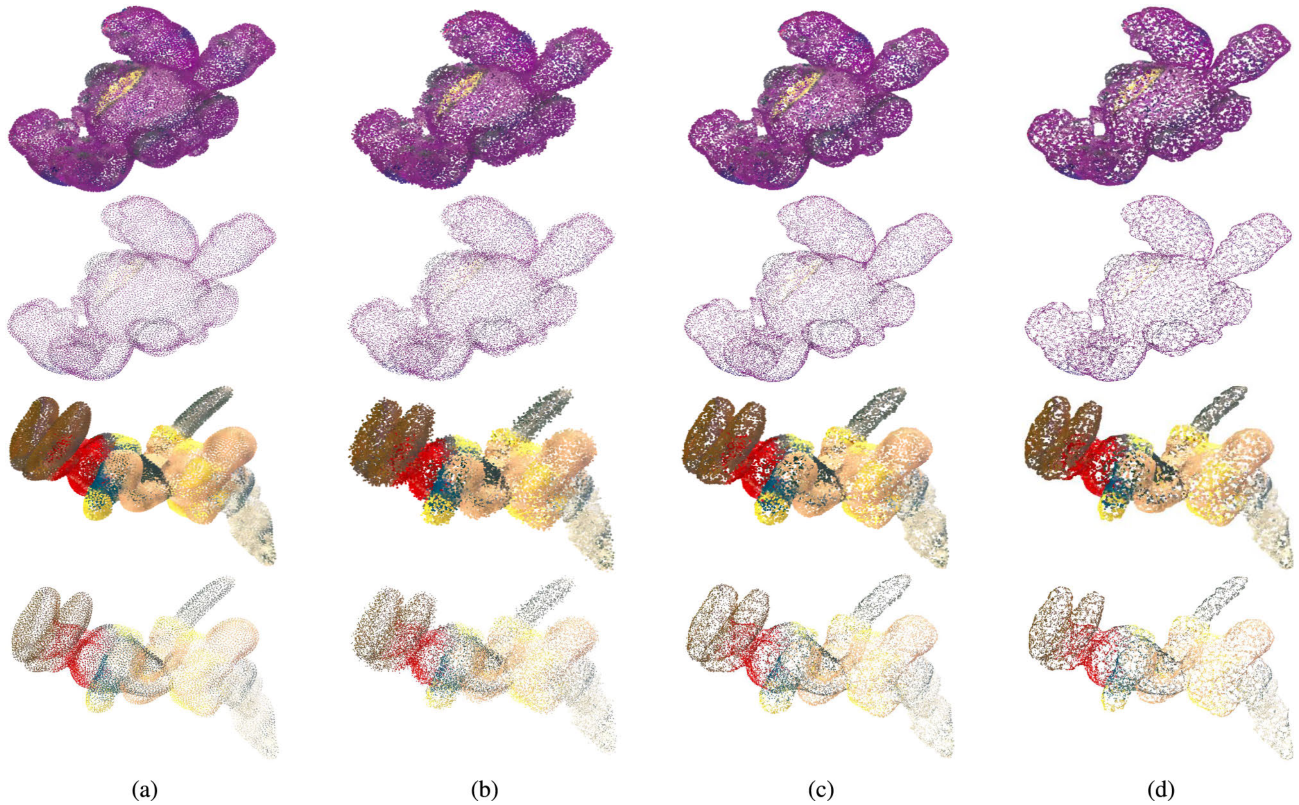


**FIGURE 8.** Sub-sampled point cloud models with color: (a) ground-truth, (b) noisy input ($\mu = 0$ and $\sigma = 0.3$), denoised results by (c) proposed algorithm, and (d) RPSM [45].

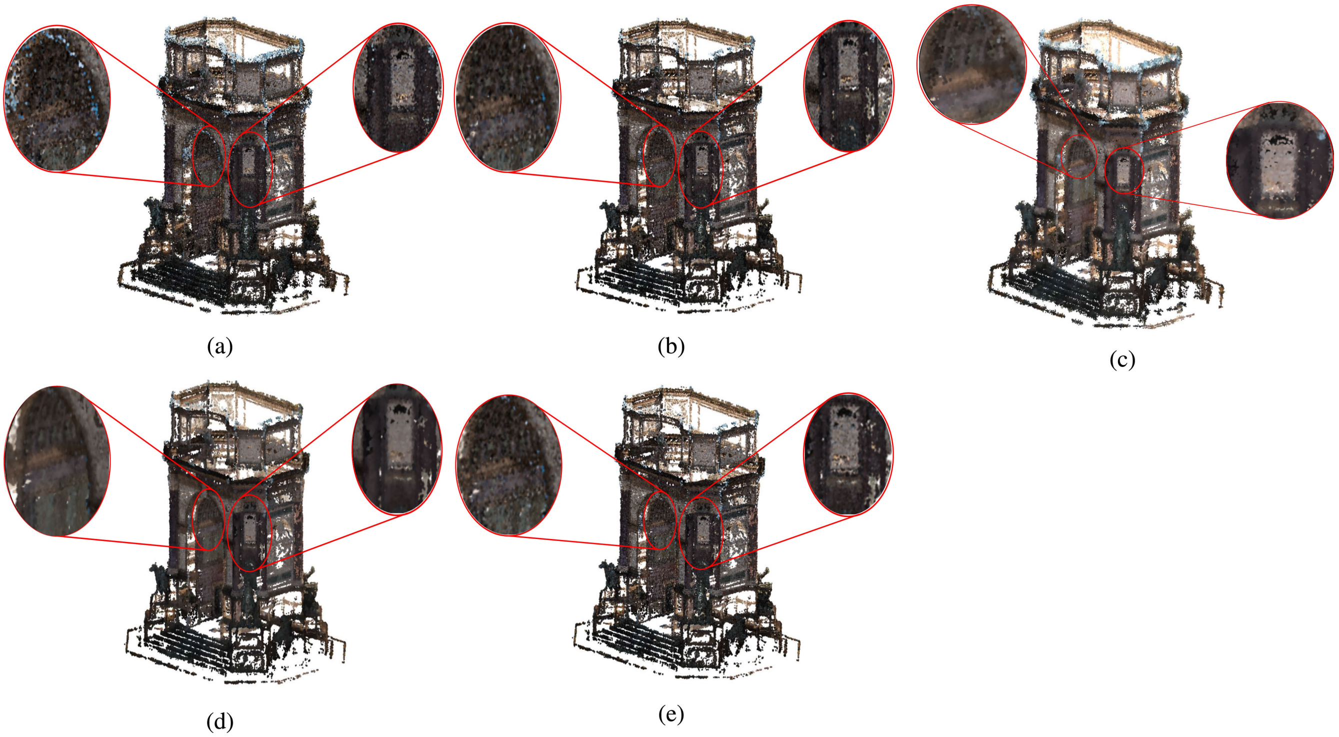


**FIGURE 9.** Arco_Valentino model illustration. (a) noisy input, (b) outlier-free input, color denoised results by (c) proposed algorithm (d) using Tikhonov regularization, and (d) using TV.

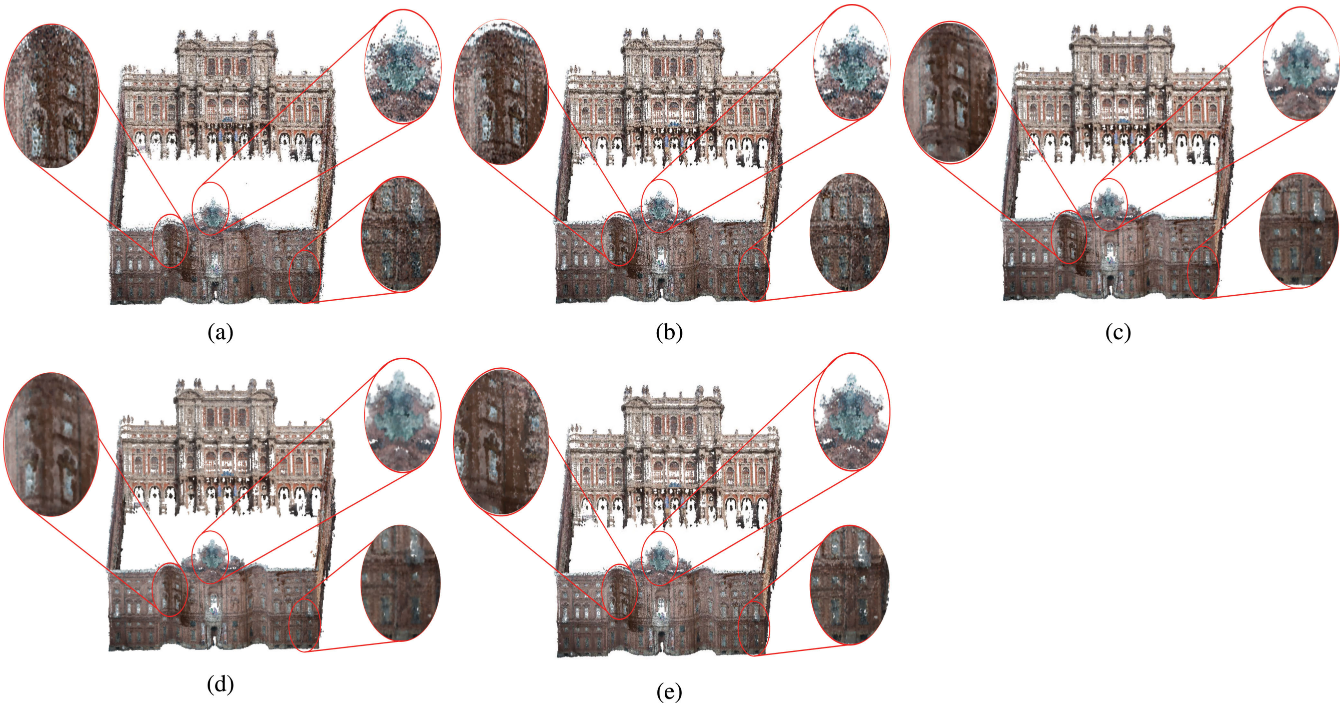


**FIGURE 10.** Palazzo_Carignano_Dense model illustration. (a) noisy input, (b) outlier-free input, color denoised results by (c) proposed algorithm (d) using Tikhonov regularization, and (d) using TV.

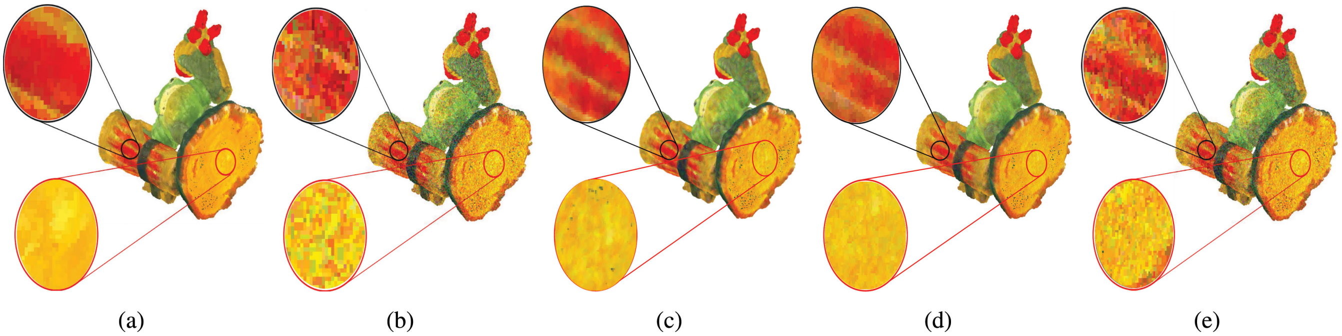


**FIGURE 11.** Green_monster model: (a) ground-truth (b) noisy point cloud with noise level of $\mu = 0$ and $\sigma = 30$, color denoised results by (c) proposed algorithm (d) using Tikhonov regularization, and (e) using TV.

The color denoising technique [24] is an iterative convex optimization technique that enforces the regularity of the denoised color attributes on a defined graph.

### 1) COLOR DENOISING OF REAL-WORLD POINT CLOUDS

The visual results of the proposed color denoising algorithm on real-world point clouds are presented here. There is not much literature available for color denoising of the point cloud as anticipated in Sec. I. Here, the qualitative results show the comparison between the proposed algorithm using SGW with Tikhonov and TV-based regularization. Fig. 9-a and 10-a represent the real-world noisy point clouds; it can be observed that the details are not very clear due to the existence of a large amount of color noise. In certain areas, two different regions are overlapped with each other as an effect of color noise. Fig. 9-b and 10-b show the resulting point clouds after outlier removal, which is required to get better color denoising results using Tikhonov and TV regularization. Fig. 9-c and 10-c depict the point cloud denoised using the proposed algorithm. Here, the colors are much smoother and natural by exploiting the relation of the color of the points within proximity. Due to the noise in the point cloud, details are missing, and one cannot see the contours in the real-world point cloud. The denoised point clouds look sharper in comparison to the input noisy point clouds. The color denoising procedure helps to preserve object boundaries. Fig. 9-d and Fig. 10-d show the denoised result using Tikhonov regularization; it can be seen that the details are quite similar to the proposed algorithm but a little over-smoothed. Fig. 9-e and Fig. 10-e show the denoised point cloud using TV; it can be seen that the color is still noisy, and there is a lack of details in the output point clouds.

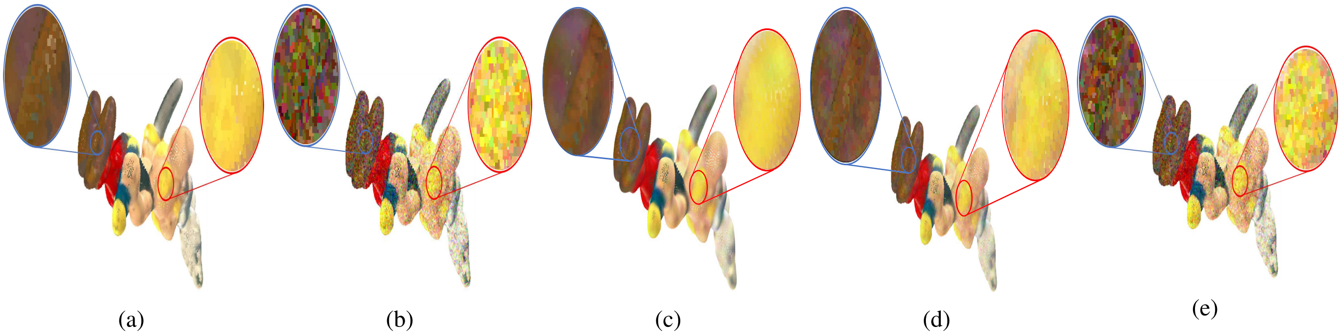


**FIGURE 12.** Asterix model: (a) ground-truth (b) noisy point cloud with noise level of $\mu = 0$ and $\sigma = 30$, color denoised results by (c) proposed algorithm (d) using Tikhonov regularization, and (e) using TV.

**TABLE 5.** MSE and MCD comparison of various algorithms for *Greyc* dataset.

| Gaussian Noise | Methods | 4arms monstre | Asterix | Cable car | Dragon | Duck | Green Dinasour | Green monster | Horse | Jaguar | Long Diansour | Mario | Mario car | Pokeman ball | Rabbit | Red horse | Statue | **Average** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MSE | | | | | | | | | | | | | | | | | | |
| $\sigma = 0.2$ | Proposed | **0.33** | **0.26** | **0.21** | **0.22** | **0.61** | 0.38 | **0.17** | **0.38** | **0.13** | **0.16** | **0.12** | **0.13** | **0.35** | **0.35** | 0.26 | **0.35** | **0.28** |
| | MSGW | 0.36 | 0.27 | 0.23 | **0.22** | 0.84 | **0.36** | **0.17** | 0.42 | 0.14 | 0.17 | 0.14 | 0.14 | 0.36 | 0.37 | **0.23** | 0.37 | 0.30 |
| | IBR | 0.38 | 0.30 | 0.29 | 0.30 | 0.63 | 0.40 | 0.23 | 0.45 | 0.20 | 0.24 | 0.19 | 0.19 | 0.41 | 0.40 | 0.30 | 0.37 | 0.33 |
| $\sigma = 0.3$ | Proposed | **0.34** | **0.27** | **0.23** | **0.23** | **0.68** | 0.39 | **0.18** | **0.39** | **0.13** | **0.17** | **0.13** | **0.15** | **0.38** | **0.36** | 0.28 | **0.36** | **0.29** |
| | MSGW | 0.37 | 0.28 | 0.24 | 0.24 | 0.92 | **0.38** | **0.18** | 0.44 | 0.15 | 0.18 | 0.15 | **0.15** | 0.40 | 0.40 | **0.25** | 0.38 | 0.32 |
| | IBR | 0.39 | 0.30 | 0.30 | 0.31 | 0.71 | 0.42 | 0.23 | 0.45 | 0.20 | 0.25 | 0.20 | 0.19 | 0.44 | 0.42 | 0.32 | 0.38 | 0.34 |
| $\sigma = 0.4$ | Proposed | **0.35** | **0.28** | **0.25** | **0.26** | **1.12** | **0.39** | **0.21** | **0.40** | **0.15** | **0.18** | **0.15** | **0.16** | **0.40** | **0.38** | **0.29** | **0.36** | **0.33** |
| | MSGW | 0.38 | **0.28** | 0.30 | 0.31 | 1.35 | 0.42 | **0.21** | 0.44 | 0.16 | 0.21 | 0.17 | 0.17 | 0.41 | 0.41 | 0.32 | 0.39 | 0.37 |
| | IBR | 0.39 | 0.31 | 0.30 | 0.32 | 1.15 | 0.42 | 0.24 | 0.45 | 0.21 | 0.26 | 0.21 | 0.20 | 0.46 | 0.42 | 0.33 | 0.38 | 0.38 |
| MCD | | | | | | | | | | | | | | | | | | |
| $\sigma = 0.2$ | Proposed | **0.49** | **0.38** | **0.30** | **0.31** | **0.85** | 0.55 | **0.25** | **0.56** | **0.19** | **0.24** | **0.18** | **0.19** | **0.51** | **0.52** | 0.38 | **0.50** | **0.40** |
| | MSGW | 0.53 | 0.39 | 0.33 | 0.33 | 1.24 | **0.53** | **0.25** | 0.61 | 0.20 | 0.25 | 0.20 | 0.20 | 0.52 | 0.54 | **0.34** | 0.53 | 0.44 |
| | IBR | 0.55 | 0.43 | 0.42 | 0.44 | 0.94 | 0.59 | 0.33 | 0.65 | 0.29 | 0.35 | 0.28 | 0.27 | 0.61 | 0.58 | 0.44 | 0.53 | 0.48 |
| $\sigma = 0.3$ | Proposed | **0.50** | **0.39** | **0.32** | **0.32** | **1.22** | 0.57 | **0.25** | **0.57** | **0.20** | **0.25** | **0.20** | **0.21** | **0.56** | **0.52** | 0.40 | **0.51** | **0.44** |
| | MSGW | 0.54 | 0.40 | 0.35 | 0.35 | 1.36 | **0.55** | 0.26 | 0.63 | 0.21 | 0.26 | 0.22 | **0.21** | 0.58 | 0.58 | **0.36** | 0.54 | 0.46 |
| | IBR | 0.56 | 0.44 | 0.43 | 0.46 | 1.06 | 0.61 | 0.34 | 0.66 | 0.30 | 0.36 | 0.29 | 0.28 | 0.65 | 0.60 | 0.46 | 0.55 | 0.50 |
| $\sigma = 0.4$ | Proposed | **0.52** | **0.40** | **0.40** | **0.41** | **1.64** | **0.56** | **0.27** | **0.58** | **0.21** | **0.26** | **0.23** | **0.23** | **0.58** | **0.55** | **0.42** | **0.53** | **0.49** |
| | MSGW | 0.54 | 0.41 | 0.43 | 0.45 | 1.95 | 0.61 | 0.31 | 0.64 | 0.24 | 0.30 | 0.24 | 0.25 | 0.61 | 0.60 | 0.47 | 0.54 | 0.54 |
| | IBR | 0.57 | 0.45 | 0.44 | 0.47 | 1.71 | 0.62 | 0.35 | 0.68 | 0.30 | 0.37 | 0.30 | 0.29 | 0.67 | 0.60 | 0.47 | 0.55 | 0.55 |

TV is not very effective at enforcing color smoothness in comparison to the proposed algorithm. Tikhonov and TV are iterative-based and parameter oriented techniques, which tend to be computationally expensive.

### 2) COLOR DENOISING OF SYNTHETIC POINT CLOUDS

The proposed algorithm for color denoising has been applied to noise-free point clouds affected by synthetic color noise; Gaussian distribution is used to add noise to the color attribute of every point in a reference point cloud while keeping the geometry noise-free. Fig. 11-a and 12-a present the ground-truth *Green_Monster* and *Asterix* point cloud models, respectively having noise-free geometry and color. Fig. 11-b and 12-b show the point cloud affected by Gaussian noise distribution with $\mu = 0$ and $\sigma = 30$; adding noise to the color affects the details and causes blurring of the boundaries. Fig. 11-c and 12-c depict the denoised output of the proposed algorithm. The color of the output point cloud is denoised by exploiting the correlation of color within the proximity, the points in the $k$-neighborhood have a high probability of having a similar color as the surface has smooth color. Fig. 11-d and 12-d depict the denoised output using Tikhonov regularization; it can be clearly seen, particularly in the highlighted areas of both point cloud models, that there is still noise in color. Fig. 11-e and 12-e illustrate the denoised output using TV. The output point clouds are still noisy, and the details are not preserved. The TV technique has the least effective in terms of color denoising.

## D. OBJECTIVE EVALUATION ON GREYC COLOR MESH DATASET

The quantitative evaluation has also been performed on the Greyc noise-free synthetic point clouds dataset [59].

### 1) COLOR DENOISING

The color attribute of each point cloud is corrupted with Gaussian noise applied to each point in a point cloud with $\sigma = 20, 30, 40$. The MSE and PSNR comparisons between the proposed color denoising algorithm and the color denoising using Tikhonov and TV regularization are shown in Tab. 8 and Tab. 9. The results of both metrics show that the proposed

**TABLE 6.** MSE and MCD comparison between proposed algorithm and RPSM [45] on sub-sampled *Greyc* dataset for different noise levels.

| Gaussian Noise | Methods | 4arms monstre | Asterix | Cable car | Dragon | Duck | Green Dinasour | Green monster | Horse | Jaguar | Long Diansour | Mario | Mario car | Pokeman ball | Rabbit | Red horse | Statue | **Average** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MSE | | | | | | | | | | | | | | | | | | |
| $\sigma = 0.2$ | Proposed | **0.61** | **0.49** | **0.85** | **0.67** | **0.83** | **0.74** | **0.59** | **0.73** | **0.50** | **0.59** | **0.55** | **0.58** | **0.51** | **0.74** | **0.76** | **0.56** | **0.64** |
| | RPSM [45] | 0.90 | 0.74 | 1.2457 | 1.02 | 1.33 | 1.00 | 0.87 | 1.19 | 0.78 | 0.73 | 1.02 | 0.86 | 0.73 | 0.94 | 1.18 | 1.03 | 0.97 |
| $\sigma = 0.3$ | Proposed | **0.61** | **0.49** | **0.86** | **0.67** | **0.87** | **0.75** | **0.58** | **0.74** | **0.48** | **0.59** | **0.56** | **0.58** | **0.50** | **0.75** | **0.77** | **0.57** | **0.65** |
| | RPSM [45] | 0.95 | 0.76 | 1.34 | 1.06 | 1.35 | 1.03 | 0.93 | 1.24 | 0.83 | 0.80 | 1.07 | 0.92 | 0.79 | 1.00 | 1.25 | 1.06 | 1.02 |
| $\sigma = 0.4$ | Proposed | **0.62** | **0.50** | **0.87** | **0.68** | **0.90** | **0.77** | **0.59** | **0.75** | **0.51** | **0.58** | **0.56** | **0.59** | **0.51** | **0.77** | **0.79** | **0.58** | **0.66** |
| | RPSM [45] | 0.98 | 0.79 | 1.41 | 1.13 | 1.39 | 1.12 | 0.97 | 1.28 | 0.89 | 0.85 | 1.12 | 0.98 | 0.90 | 1.05 | 1.30 | 1.11 | 1.08 |
| MCD | | | | | | | | | | | | | | | | | | |
| $\sigma = 0.2$ | Proposed | **0.89** | **0.70** | **1.24** | **0.98** | **1.24** | **1.09** | **0.86** | **1.07** | **0.72** | **0.85** | **0.80** | **0.83** | **0.74** | **1.08** | **1.11** | **0.83** | **0.94** |
| | RPSM [45] | 1.44 | 1.04 | 1.93 | 1.56 | 1.91 | 1.60 | 1.33 | 1.79 | 1.18 | 1.24 | 1.48 | 1.25 | 1.51 | 1.63 | 1.79 | 1.56 | 1.51 |
| $\sigma = 0.3$ | Proposed | **0.90** | **0.71** | **1.26** | **0.98** | **1.29** | **1.10** | **0.86** | **1.08** | **0.71** | **0.86** | **0.82** | **0.84** | **0.74** | **1.09** | **1.13** | **0.84** | **0.95** |
| | RPSM [45] | 1.51 | 1.08 | 2.05 | 1.63 | 1.94 | 1.67 | 1.41 | 1.87 | 1.24 | 1.32 | 1.55 | 1.33 | 1.60 | 1.72 | 1.90 | 1.60 | 1.59 |
| $\sigma = 0.4$ | Proposed | **0.91** | **0.72** | **1.27** | **0.99** | **1.33** | **1.12** | **0.87** | **1.10** | **0.74** | **0.84** | **0.83** | **0.86** | **0.75** | **1.12** | **1.16** | **0.85** | **0.96** |
| | RPSM [45] | 1.55 | 1.11 | 2.15 | 1.71 | 2.00 | 1.77 | 1.47 | 1.92 | 1.32 | 1.40 | 1.63 | 1.41 | 1.73 | 1.78 | 1.97 | 1.66 | 1.66 |

**TABLE 7.** C2M metric comparison of the proposed geometry denoising algorithm with the IBR and MSGW.

| Geometry Noise | Methods | Parameters | 4arms monstre | Asterix | Cable car | Dragon | Duck | Green Dinasour | Green monster | Horse | Jaguar | Long Diansour | Mario | Mario car | Pokeman ball | Rabbit | Red horse | Statue |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\sigma = 0.2$ | Proposed | $d_H$ | **0.68** | **0.40** | **0.82** | **0.91** | **3.13** | 1.50 | **0.75** | **0.94** | **0.83** | **1.02** | **0.69** | **0.58** | **0.45** | **0.92** | 1.27 | **0.89** |
| | | $d_m$ | **0.19** | **0.12** | **0.12** | **0.12** | **0.44** | 0.17 | **0.05** | **0.24** | **0.06** | **0.07** | **0.07** | **0.08** | **0.16** | **0.23** | 0.17 | **0.25** |
| | | $\zeta$ | **0.10** | **0.07** | **0.09** | **0.09** | **0.37** | 0.10 | **0.04** | **0.12** | **0.05** | **0.05** | **0.06** | **0.07** | **0.09** | **0.12** | 0.13 | **0.09** |
| | IBR | $d_H$ | 0.96 | 0.61 | 1.16 | 1.29 | 3.83 | 1.50 | 1.18 | 0.94 | 0.83 | 1.12 | 1.19 | 0.82 | 1.00 | 1.31 | 1.28 | 1.26 |
| | | $d_m$ | 0.30 | 0.21 | 0.22 | 0.24 | 0.60 | 0.30 | 0.09 | 0.37 | 0.14 | 0.17 | 0.14 | 0.14 | 0.35 | 0.33 | 0.24 | 0.29 |
| | | $\zeta$ | 0.11 | 0.09 | 0.11 | 0.09 | 0.43 | 0.10 | 0.61 | 0.13 | 0.05 | 0.06 | 0.08 | 0.07 | 0.15 | 0.13 | 0.10 | 0.12 |
| | MSGW | $d_H$ | 0.68 | 0.86 | 1.17 | 0.91 | 3.53 | **1.50** | 1.06 | 0.94 | 0.83 | 1.02 | 1.38 | 1.31 | 0.63 | 0.93 | **1.27** | 0.90 |
| | | $d_m$ | 0.25 | 0.13 | 0.16 | 0.17 | 0.76 | **0.16** | 0.07 | 0.29 | 0.10 | 0.10 | 0.09 | 0.09 | 0.20 | 0.28 | **0.13** | 0.26 |
| | | $\zeta$ | 0.11 | 0.08 | 0.12 | 0.13 | 0.66 | **0.09** | 0.05 | 0.13 | 0.07 | 0.07 | 0.07 | 0.07 | 0.10 | 0.13 | **0.09** | 0.09 |
| $\sigma = 0.3$ | Proposed | $d_H$ | **0.68** | **0.86** | **0.82** | **0.91** | **3.13** | 1.50 | **0.75** | **0.93** | **0.85** | **1.02** | **0.97** | **0.83** | **0.45** | **0.94** | 1.27 | **0.89** |
| | | $d_m$ | **0.20** | **0.13** | **0.13** | **0.13** | **0.50** | 0.17 | **0.05** | **0.25** | **0.07** | **0.07** | **0.08** | **0.09** | **0.16** | **0.24** | 0.19 | **0.26** |
| | | $\zeta$ | **0.09** | **0.07** | **0.10** | **0.10** | **0.41** | 0.11 | **0.05** | **0.12** | **0.06** | **0.06** | **0.07** | **0.08** | **0.10** | **0.13** | 0.15 | **0.09** |
| | IBR | $d_H$ | 0.96 | 0.91 | 1.165 | 1.29 | 3.98 | 1.51 | 1.59 | 0.96 | 0.92 | 1.04 | 1.43 | 1.18 | 0.10 | 1.31 | 1.29 | 1.28 |
| | | $d_m$ | 0.30 | 0.21 | 0.23 | 0.25 | 0.66 | 0.31 | 0.13 | 0.38 | 0.15 | 0.18 | 0.15 | 0.15 | 0.36 | 0.34 | 0.24 | 0.30 |
| | | $\zeta$ | 0.12 | 0.09 | 0.13 | 0.10 | 0.24 | 0.10 | 0.67 | 0.14 | 0.06 | 0.07 | 0.12 | 0.13 | 0.16 | 0.14 | 0.10 | 0.12 |
| | MSGW | $d_H$ | 0.68 | 0.96 | 1.17 | 0.91 | 3.87 | **0.15** | 1.58 | 0.93 | 0.90 | 1.03 | 1.39 | 1.69 | 0.90 | 1.31 | **1.27** | 1.27 |
| | | $d_m$ | 0.26 | 0.14 | 0.17 | 0.18 | 0.79 | **0.16** | 0.08 | 0.29 | 0.14 | 0.15 | 0.13 | 0.14 | 0.22 | 0.30 | **0.13** | 0.27 |
| | | $\zeta$ | 0.11 | 0.09 | 0.13 | 0.14 | 0.68 | **0.10** | 0.07 | 0.13 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.13 | **0.09** | 0.10 |
| $\sigma = 0.4$ | Proposed | $d_H$ | **0.96** | **0.86** | **1.06** | **0.91** | **3.54** | **1.50** | **1.18** | **0.94** | **1.18** | **1.03** | **1.38** | **1.17** | **0.64** | **0.94** | **1.27** | **1.27** |
| | | $d_m$ | **0.20** | **0.13** | **0.20** | **0.21** | **0.75** | **0.17** | **0.10** | **0.25** | **0.10** | **0.12** | **0.11** | **0.12** | **0.17** | **0.25** | **0.21** | **0.27** |
| | | $\zeta$ | **0.12** | **0.08** | **0.15** | **0.15** | **0.64** | **0.12** | **0.10** | **0.13** | **0.08** | **0.09** | **0.09** | **0.10** | **0.10** | **0.13** | **0.16** | **0.11** |
| | IBR | $d_H$ | 1.21 | 1.17 | 1.36 | 1.58 | 4.02 | 1.52 | 1.75 | 1.32 | 1.24 | 1.76 | 1.53 | 1.44 | 1.11 | 1.61 | 1.37 | 1.28 |
| | | $d_m$ | 0.32 | 0.22 | 0.23 | 0.27 | 0.69 | 0.32 | 0.18 | 0.39 | 0.16 | 0.19 | 0.16 | 0.16 | 0.37 | 0.36 | 0.25 | 0.30 |
| | | $\zeta$ | 0.13 | 0.09 | 0.15 | 0.11 | 0.26 | 0.15 | 0.13 | 0.16 | 0.09 | 0.11 | 0.12 | 0.13 | 0.16 | 0.15 | 0.10 | 0.13 |
| | MSGW | $d_H$ | 1.16 | 1.14 | 1.22 | 1.30 | 4.11 | 1.51 | 1.25 | 1.32 | 1.23 | 1.45 | 1.44 | 1.38 | 0.91 | 1.51 | 1.28 | 1.28 |
| | | $d_m$ | 0.22 | 0.14 | 0.21 | 0.22 | 0.95 | 0.18 | 0.12 | 0.25 | 0.14 | 0.17 | 0.16 | 0.15 | 0.18 | 0.26 | 0.22 | 0.28 |
| | | $\zeta$ | 0.13 | 0.10 | 0.16 | 0.16 | 0.78 | 0.13 | 0.11 | 0.14 | 0.08 | 0.10 | 0.11 | 0.12 | 0.11 | 0.13 | 0.17 | 0.12 |

technique performed better than the Tikhonov and TV regularization for all the point cloud models for all the noise-level except *cable_car, Horse*, which performed better for noise level of $\sigma = 20$, 30 and *Pokemon_ball* performed better only for noise-level $\sigma = 20$. The average MSE and PSNR (last column in Tab. 8 and Tab. 9 and Fig. 14) shows that the gain is larger with the increase in the noise-level, showing that the proposed algorithm of color denoising using SGW performed better.

To compare the proposed color denoising using SGW with GLR [19] and GTV [19], [60], we added Gaussian noise with zero-mean and $\sigma = 10, 15, 20, 25$ to the color attribute of the

**TABLE 8.** **MSE comparison of color denoising algorithm for *Greyc dataset.***

| Gaussian Noise | Methods | 4arms monstre | Asterix | Cable car | Dragon | Duck | Green Dinasour | Green monster | Horse | Jaguar | Long Diansour | Mario | Mario car | Pokeman ball | Rabbit | Red horse | Statue | **Average** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\sigma = 20$ | Noisy | 398.25 | 361.28 | 373.52 | 393.64 | 339.57 | 397.24 | 365.38 | 383.29 | 368.57 | 387.52 | 321.09 | 375.24 | 309.14 | 335.40 | 377.39 | 398.13 | 367.79 |
| | Proposed | **69.43** | **98.20** | 166.40 | **71.44** | **128.27** | **63.03** | **85.39** | 166.60 | **76.68** | **66.00** | **87.33** | **78.41** | 156.10 | **89.09** | **80.64** | **78.93** | **97.63** |
| | Tikhonov | 77.75 | 103.56 | **145.23** | 82.80 | 130.65 | 63.04 | 91.64 | **150.11** | 85.83 | 77.90 | 95.91 | 89.56 | **149.53** | 93.75 | 82.52 | 84.77 | 100.29 |
| | TV | 303.05 | 279.02 | 289.48 | 298.13 | 286.05 | 301.50 | 279.04 | 299.68 | 279.21 | 293.61 | 242.14 | 283.15 | 241.15 | 254.99 | 293.32 | 302.99 | 282.91 |
| $\sigma = 30$ | Noisy | 869.27 | 781.84 | 816.58 | 867.14 | 722.06 | 875.25 | 796.19 | 850.03 | 797.43 | 850.89 | 692.54 | 810.94 | 645.11 | 713.61 | 811.93 | 882.35 | 798.95 |
| | Proposed | **93.26** | **124.45** | 224.75 | **99.77** | **166.98** | **81.85** | **131.05** | 226.10 | **125.85** | **85.24** | **137.52** | **109.78** | **194.21** | **133.74** | **128.09** | **103.39** | **135.38** |
| | Tikhonov | 109.74 | 148.53 | **214.89** | 120.77 | 189.47 | 89.12 | 138.24 | **217.08** | 130.09 | 112.13 | 157.44 | 136.40 | 250.35 | 148.58 | 133.63 | 123.06 | 151.22 |
| | Total Variation | 590.97 | 540.25 | 564.67 | 589.46 | 548.87 | 594.55 | 542.77 | 596.83 | 539.70 | 575.89 | 467.70 | 544.00 | 449.27 | 485.12 | 561.85 | 602.88 | 549.67 |
| $\sigma = 40$ | Noisy | 1506.10 | 1329.01 | 1384.60 | 1462.90 | 1241.90 | 1469.80 | 1347.81 | 1462.10 | 1355.51 | 1462.61 | 1171.90 | 1372.91 | 1092.10 | 1217.80 | 1376.21 | 1519.90 | 1360.82 |
| | Proposed | **108.93** | **203.40** | **269.15** | **137.18** | **221.99** | **107.32** | **181.40** | **274.05** | **139.83** | **119.45** | **194.93** | **149.54** | **263.86** | **204.24** | **221.17** | **129.75** | **182.89** |
| | Tikhonov | 137.31 | 210.90 | 281.89 | 149.67 | 262.51 | 133.94 | 184.53 | 279.98 | 175.90 | 144.71 | 228.37 | 185.25 | 385.53 | 213.93 | 242.06 | 154.26 | 210.67 |
| | Total Variation | 1119.50 | 995.38 | 1033.80 | 1081.70 | 979.85 | 1088.40 | 1000.40 | 1107.10 | 1000.90 | 1083.80 | 862.00 | 1005.10 | 816.02 | 903.14 | 1026.80 | 1131.90 | 1014.74 |

**TABLE 9.** **PSNR comparison of color denoising algorithm for *Greyc dataset.***

| Gaussian Noise | Methods | 4arms monstre | Asterix | Cable car | Dragon | Duck | Green Dinasour | Green monster | Horse | Jaguar | Long Diansour | Mario | Mario car | Pokeman ball | Rabbit | Red horse | Statue | **Average** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\sigma = 20$ | Noisy | 22.13 | 22.55 | 22.41 | 22.18 | 22.82 | 22.14 | 22.50 | 22.30 | 22.47 | 22.25 | 23.07 | 22.39 | 23.23 | 22.88 | 22.36 | 22.13 | 22.49 |
| | Proposed | **29.72** | **28.21** | 25.92 | **29.59** | **27.05** | **30.13** | **28.82** | 25.91 | **29.28** | **29.94** | **28.72** | **29.19** | 26.20 | **28.63** | **29.07** | **29.19** | **28.47** |
| | Tikhonov | 29.22 | 27.98 | **26.51** | 28.95 | 26.97 | 30.13 | 28.51 | **26.37** | 28.79 | 29.21 | 28.31 | 28.61 | **26.38** | 28.41 | 28.97 | 28.85 | 28.26 |
| | Total Variation | 23.32 | 23.67 | 23.52 | 23.39 | 23.57 | 23.34 | 23.67 | 23.36 | 23.67 | 23.45 | 24.29 | 23.61 | 24.31 | 24.07 | 23.46 | 23.32 | 23.63 |
| $\sigma = 30$ | Noisy | 18.74 | 19.20 | 19.01 | 18.75 | 19.55 | 18.71 | 19.12 | 18.84 | 19.11 | 18.83 | 19.73 | 19.04 | 20.03 | 19.60 | 19.04 | 18.67 | 19.12 |
| | Proposed | **28.43** | **27.18** | 24.61 | **28.14** | **25.90** | **29.01** | **26.96** | 24.59 | **27.13** | **28.83** | **26.75** | **27.73** | **25.25** | **26.87** | **27.06** | **27.99** | **27.03** |
| | Tikhonov | 27.73 | 26.42 | **24.81** | 27.31 | 25.36 | 28.63 | 26.72 | **24.76** | 26.99 | 27.63 | 26.16 | 26.78 | 24.14 | 26.33 | 26.88 | 27.23 | 26.49 |
| | Total Variation | 20.42 | 20.81 | 20.61 | 20.43 | 20.74 | 20.39 | 20.79 | 20.37 | 20.81 | 20.53 | 21.43 | 20.78 | 21.61 | 21.27 | 20.64 | 20.33 | 20.75 |
| $\sigma= 40$ | Noisy | 16.35 | 16.90 | 16.72 | 16.48 | 17.19 | 16.46 | 16.84 | 16.48 | 16.81 | 16.48 | 17.44 | 16.75 | 17.75 | 17.28 | 16.74 | 16.31 | 16.81 |
| | Proposed | **27.76** | **25.05** | **23.83** | **26.76** | **24.67** | **27.82** | **25.54** | **23.75** | **26.67** | **27.36** | **25.23** | **26.38** | **23.92** | **25.03** | **24.68** | **27.00** | **25.72** |
| | Tikhonov | 26.75 | 24.89 | 23.63 | 26.38 | 23.95 | 26.86 | 25.47 | 23.66 | 25.68 | 26.53 | 24.54 | 25.45 | 22.27 | 24.83 | 24.29 | 26.25 | 25.09 |
| | Total Variation | 17.64 | 18.15 | 17.99 | 17.79 | 18.22 | 17.76 | 18.13 | 17.69 | 18.13 | 17.78 | 18.78 | 18.11 | 19.01 | 18.57 | 18.01 | 17.59 | 18.09 |

noise-free point cloud models of *Greyc* dataset. Quantitative results in terms of PSNR are shown in Tab. 10, where the proposed algorithm shows the highest average PSNR value for all the noise level $\sigma = 10$, 15, 20 and 25. With the increase in the noise level, the proposed algorithm performs better than GLR and GTV, with an average PSNR increase of 0.49dB and 0.34dB, respectively. Besides PSNR, the average execution time (AET) per noise level for the proposed GLR and GTV algorithms is also shown in Tab. 10. The proposed method is almost 6 and over 200 times faster than GLR and GTV, respectively, as it is a non-iterative technique, which is computationally very cheap. AET has been measured in MATLAB 2019b on a 3.5Ghz MacBook Pro with Intel Core i7 processor and 16GB memory.

### 2) GEOMETRY DENOISING

The proposed geometry denoising method has also been verified via quantitative evaluation on the complete *Greyc* dataset. Each point cloud has been corrupted with zero-mean Gaussian synthetic geometry noise, applied to each point with $\sigma =$ 0.2, 0.3, and 0.4. The MSE and MCD comparisons between the proposed algorithm and the denoising approaches used in IBR [42] using TV regularization and MSGW are shown in Tab. 5. The results show that the proposed denoising technique performed better than MSGW and IBR for all the models for noise level $\sigma = 0.2$, 0.3, and 0.4 except *Green_Dinasour* and *Red_Horse* (where MSGW performed better) for $\sigma = 0.2, 0.3$.

For further verification of the performance of the proposed geometry denoising algorithm, we compute the C2M metric for each denoised point cloud by the proposed algorithm, MSGW, and IBR with respect to the corresponding reference ground-truth point cloud. The comparative C2M metric results are shown in Tab. 7, which further verifies the better performance of the proposed algorithm. To better understand the results, we mapped the C2M distance of denoised point cloud of *4arms_monstre* on the reference point cloud; this is shown in Fig. 13. The distances are represented by the color scale; blue, green, yellow, and red display the range of distances from minimum to maximum. The histogram of the proposed algorithm in Fig. 13-a shows that the points move closer to their actual position without moving too far from their corresponding original position of a reference point cloud. We can see in the histograms shown in Fig. 13-b and

**TABLE 10.** Color denoising comparison for Gaussian noise $\sigma = 10, 15, 20$, and 25 with GLR-based [19] and GTV-based [60] in terms of PSNR and AET (s).

| Model | $\sigma = 10$ | | | | $\sigma = 15$ | | | | $\sigma = 20$ | | | | $\sigma = 25$ | | | | AET (s) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Noise | Proposed | GLR | GTV | Noise | Proposed | GLR | GTV | Noise | Proposed | GLR | GTV | Noise | Proposed | GLR | GTV | Proposed | GLR | GTV |
| Asterix | 28.38 | 31.76 | **32.03** | 31.61 | 24.94 | **29.69** | 29.66 | 29.53 | 22.53 | **28.21** | 28.10 | 27.56 | 20.68 | **27.09** | 26.12 | 27.05 | **1.06** | 5.20 | 211.00 |
| Duck | 28.53 | **30.83** | 30.40 | 30.60 | 25.20 | **28.51** | 28.42 | 28.14 | 22.71 | **27.05** | 26.89 | 26.35 | 20.90 | **26.21** | 25.68 | 25.04 | **0.46** | 1.70 | 58.00 |
| Green_Dinosaur | 28.14 | 33.19 | 33.28 | **33.36** | 24.60 | 31.21 | **31.64** | 31.31 | 22.13 | 30.13 | 30.30 | **30.34** | 20.23 | 29.37 | 28.52 | **29.62** | **1.82** | 11.20 | 389.00 |
| Red_Horse | 28.29 | **32.79** | 32.15 | 32.20 | 24.74 | **30.59** | 29.72 | 30.01 | 22.35 | **29.07** | 27.85 | 28.57 | 20.47 | **28.25** | 25.66 | 27.33 | **3.48** | 19.50 | 851.00 |
| **Average** | 28.30 | **32.14** | 31.87 | 31.94 | 24.85 | **30.00** | 29.86 | 29.75 | 22.44 | **28.62** | 28.29 | 28.20 | 20.61 | **27.73** | 26.495 | 27.26 | **1.71** | 9.40 | 377.25 |

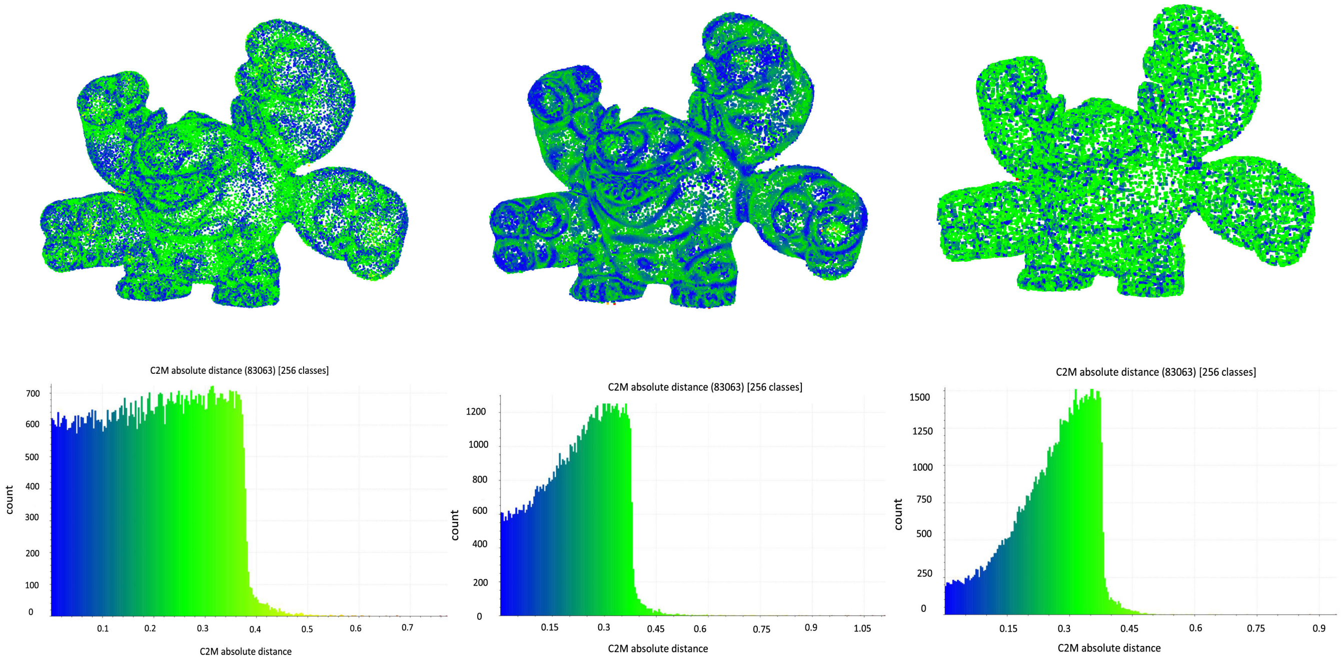


**FIGURE 13.** 4arms_model (noise level $\mu = 0$ and $\sigma = 0.4$) (a) C2M metric of denoised point cloud by proposed algorithm (b) C2M metric of denoised point cloud using MSGW, and (c) C2M metric of denoised point cloud using IBR.

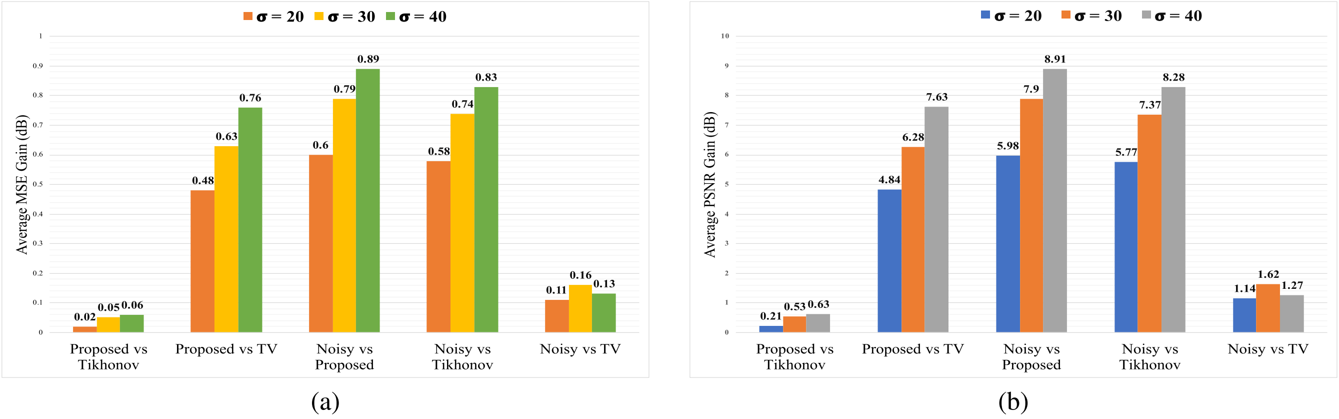


**FIGURE 14.** (a): Average gain in MSE (dB) for the color denoising algorithm (b): Average gain in PSNR (dB) for the color denoising algorithm.

Fig. 13-c that the points have deviated too much from their actual position, and also the number of points moved farther are higher for MSGW and IBR.

The objective evaluation has also been performed for the comparison between the proposed algorithm and RPSM [45] on the sub-sampled point cloud models of the *Greyc* dataset [59]. Tab. 6 shows the MSE and MCD comparison; it can be seen that the proposed algorithm outperformed RPSM [45] in terms of both metrics for $\sigma = 0.2,\ 0.3,$ and 0.4. The average MSE and MCD (last column in Tab. 5 and Tab. 6) shows that the gain is larger as the noise level increases, indicating that the proposed denoising method is better at removing geometry noise.

## VIII. CONCLUSION

In this article, we proposed a novel and efficient framework for point cloud denoising based on SGW. Unlike any other existing method, the proposed algorithm takes advantage of the correlation between the color and geometry attribute of a point cloud, which is statistically analysed in Sec. III. Such correlation is encoded in a $k$-NN graph, which can be used to denoise the color and geometry noise of the point cloud by simply adapting the potential values of parameters ($\theta_X$ and $\theta_C$) required for $k$-NN graph construction in each denoising scenario. An adaptive data-driven wavelet soft-thresholding then performs denoising.

We have provided a large set of experimental results using real-world as well as synthetic point clouds, which support the conclusion that the joint use of color and geometry is beneficial in both the color-only and geometry-only denoising scenarios, providing denoised point clouds having higher subjective and objective quality.

We showed results on both real-world and synthetic point clouds for color and geometry denoising for the visual inspection. The qualitative results for geometry denoising by the proposed algorithm are very good, avoiding the artifacts typically caused by other techniques. The proposed color denoising algorithm has also performed very well. We also performed an extensive quantitative analysis using multiple point clouds in the *Greyc* dataset [59], evaluating the performance of the proposed color denoising algorithm of point clouds using MSE and PSNR metrics. For geometry denoising, we computed MSE, MCD, and C2M metric. Both the subjective and objective results show that the proposed techniques perform very well for point cloud denoising, outperforming state-of-the-art techniques.

## REFERENCES

[1] C. Tulvan, R. Mekuria, Z. Li, and S. Laserre, *Use Cases for Point Cloud Compression*, Standard IEC JTC1/SC29/WG11 MPEG (2016), 2016.

[2] F. L. Siena, B. Byrom, P. Watts, and P. Breedon, ‘‘Utilising the Intel RealSense camera for measuring health outcomes in clinical research,’’ *J. Med. Syst.*, vol. 42, no. 3, p. 53, Mar. 2018.

[3] X. Gong, M. Chen, and X. Yang, ‘‘Point cloud segmentation of 3D scattered parts sampled by RealSense,’’ in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Jul. 2017, pp. 1–6.

[4] R. Das and K. B. S. Kumar, ‘‘GeroSim: A simulation framework for gesture driven robotic arm control using Intel RealSense,’’ in *Proc. IEEE 1st Int. Conf. Power Electron., Intell. Control Energy Syst. (ICPEICES)*, Jul. 2016, pp. 1–5.

[5] Q. Li, Y. Wang, A. Sharf, Y. Cao, C. Tu, B. Chen, and S. Yu, ‘‘Classification of gait anomalies from Kinect,’’ *Vis. Comput.*, vol. 34, no. 2, pp. 229–241, Feb. 2018.

[6] K. Kamal, S. Mathavan, T. Zafar, I. Moazzam, A. Ali, S. U. Ahmad, and M. Rahman, ‘‘Performance assessment of Kinect as a sensor for pothole imaging and metrology,’’ *Int. J. Pavement Eng.*, vol. 19, no. 7, pp. 565–576, Jul. 2018.

[7] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang, ‘‘SurfaceNet: An end-to-end 3D neural network for multiview stereopsis,’’ in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2307–2315.

[8] D. J. Tan, F. Tombari, and N. Navab, ‘‘Real-time accurate 3D head tracking and pose estimation with consumer RGB-D cameras,’’ *Int. J. Comput. Vis.*, vol. 126, nos. 2–4, pp. 158–183, Apr. 2018.

[9] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, ‘‘State of the art on 3D reconstruction with RGB-D cameras,’’ *Comput. Graph. Forum*, vol. 37, no. 2, pp. 625–652, 2018.

[10] F. Verdoja, D. Thomas, and A. Sugimoto, ‘‘Fast 3D point cloud segmentation using supervoxels with geometry and color for 3D scene understanding,’’ in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2017, pp. 1285–1290.

[11] Q. Zhan, Y. Liang, and Y. Xiao, ‘‘Color-based segmentation of point clouds,’’ *Laser Scanning*, vol. 38, no. 3, pp. 155–161, 2009.

[12] C. D. Mutto, P. Zanuttigh, and G. M. Cortelazzo, ‘‘Fusion of geometry and color information for scene segmentation,’’ *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 5, pp. 505–521, Sep. 2012.

[13] Y.-J. Liu, Y.-F. Zheng, L. Lv, Y.-M. Xuan, and X.-L. Fu, ‘‘3D model retrieval based on color + geometry signatures,’’ *Vis. Comput.*, vol. 28, no. 1, pp. 75–86, Jan. 2012.

[14] D. Slater and G. Healey, ‘‘Combining color and geometric information for the illumination invariant recognition of 3-D objects,’’ in *Proc. IEEE Int. Conf. Comput. Vis.*, 1995, pp. 563–568.

[15] G. Rosman, A. Dubrovina, and R. Kimmel, ‘‘Patch-collaborative spectral point-cloud denoising,’’ *Comput. Graph. Forum*, vol. 32, no. 8, pp. 1–12, 2013.

[16] E. Mattei and A. Castrodad, ‘‘Point cloud denoising via moving RPCA,’’ *Comput. Graph. Forum*, vol. 36, no. 8, pp. 123–137, 2017.

[17] Y. Sun, S. Schaefer, and W. Wang, ‘‘Denoising point sets via l0 minimization,’’ *Comput. Aided Geometric Des.*, vols. 35–36, pp. 2–15, May 2015.

[18] Y. Zheng, G. Li, S. Wu, Y. Liu, and Y. Gao, ‘‘Guided point cloud denoising via sharp feature skeletons,’’ *Vis. Comput.*, vol. 33, nos. 6–8, pp. 857–867, Jun. 2017.

[19] C. Dinesh, G. Cheung, and I. V. Bajic, ‘‘3D point cloud color denoising using convex graph-signal smoothness priors,’’ in *Proc. IEEE 21st Int. Workshop Multimedia Signal Process. (MMSP)*, Sep. 2019, pp. 1–6.

[20] E. K. Matti and S. Nebiker, ‘‘Geometry and colour based classification of urban point cloud scenes using a supervised self-organizing map,’’ *Photogrammetrie-Fernerkundung-Geoinf.*, vol. 2014, no. 3, pp. 161–173, 2014.

[21] D. K. Hammond, P. Vandergheynst, and R. Gribonval, ‘‘Wavelets on graphs via spectral graph theory,’’ *Appl. Comput. Harmon. Anal.*, vol. 30, no. 2, pp. 129–150, Mar. 2011.

[22] M. A. Irfan and E. Magli, ‘‘Point cloud denoising using joint geometry/color graph wavelets,’’ in *Proc. IEEE Workshop Signal Process. Syst. (SiPS)*, Oct. 2020, pp. 1–6.

[23] M. A. Irfan and E. Magli, ‘‘3D point cloud denoising using a joint geometry and color k-NN graph,’’ in *Proc. 28th Eur. Signal Process. Conf. (EUSIPCO)*, Jan. 2021, pp. 585–589.

[24] M. A. Irfan and E. Magli, ‘‘Exploiting color for graph-based 3D point cloud denoising,’’ *J. Vis. Commun. Image Represent.*, vol. 75, Jan. 2021, Art. no. 103027.

[25] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, ‘‘Towards 3D point cloud based object maps for household environments,’’ *Robot. Auto. Syst.*, vol. 56, no. 11, pp. 927–941, Nov. 2008.

[26] R. B. Rusu and S. Cousins, ‘‘3D is here: Point cloud library (PCL),’’ in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 1–4.

[27] X. Gao, W. Hu, and Z. Guo, ‘‘Graph-based point cloud denoising,’’ in *Proc. IEEE 4th Int. Conf. Multimedia Big Data (BigMM)*, Sep. 2018, pp. 1–6.

[28] W. Huang, Y. Li, P. Wen, and X. Wu, ‘‘Algorithm for 3D point cloud denoising,’’ in *Proc. 3rd Int. Conf. Genetic Evol. Comput.*, Oct. 2009, pp. 574–577.

[29] F. Yan and Z. Jinlei, ‘‘Research on scattered points cloud denoising algorithm,’’ in *Proc. IEEE Int. Conf. Signal Process., Commun. Comput. (ICSPCC)*, Sep. 2015, pp. 1–5.

[30] D. Levin, ‘‘Mesh-independent surface interpolation,’’ in *Geometric Modeling for Scientific Visualization*. Berlin, Germany: Springer, 2004, pp. 37–49.

[31] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, ‘‘Computing and rendering point set surfaces,’’ *IEEE Trans. Vis. Comput. Graphics*, vol. 9, no. 1, pp. 3–15, Jan. 2003.

[32] G. Guennebaud and M. Gross, ‘‘Algebraic point set surfaces,’’ *ACM Trans. Graph.*, vol. 26, no. 3, p. 23, Jul. 2007.

[33] G. Guennebaud, M. Germann, and M. Gross, "Dynamic sampling and rendering of algebraic point set surfaces," *Comput. Graph. Forum*, vol. 27, no. 2, pp. 653–662, 2008.

[34] R. B. Rusu, N. Blodow, Z. Marton, A. Soos, and M. Beetz, "Towards 3D object maps for autonomous household robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2007, pp. 3191–3198.

[35] A. C. Öztireli, G. Guennebaud, and M. Gross, "Feature preserving point set surfaces based on non-linear kernel regression," *Comput. Graph. Forum*, vol. 28, no. 2, pp. 493–501, 2009.

[36] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer, "Parameterization-free projection for geometry reconstruction," *ACM Trans. Graph.*, vol. 26, no. 3, p. 22, Jul. 2007.

[37] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 176:1–176:7, Dec. 2009.

[38] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. Zhang, "Edge-aware point set resampling," *ACM Trans. Graph.*, vol. 32, no. 1, pp. 1–12, Jan. 2013.

[39] H. Avron, A. Sharf, C. Greif, and D. Cohen-Or, "$\ell_1$-sparse reconstruction of sharp point set surfaces," *ACM Trans. Graph.*, vol. 29, no. 5, p. 135, 2010.

[40] X.-F. Han, J. S. Jin, M.-J. Wang, W. Jiang, L. Gao, and L. Xiao, "A review of algorithms for filtering the 3D point cloud," *Signal Process., Image Commun.*, vol. 57, pp. 103–112, Sep. 2017.

[41] J. Wang, *Geometric Structure of High-Dimensional Data and Dimensionality Reduction*. Berlin, Germany: Springer-Verlag, 2012.

[42] Y. Schoenenberger, J. Paratte, and P. Vandergheynst, "Graph-based denoising for time-varying point clouds," in *Proc. 3DTV-Conf., True Vis.-Capture, Transmiss. Display 3D Video (3DTV-CON)*, Jul. 2015, pp. 1–4.

[43] C. Dinesh, G. Cheung, I. V. Bajic, and C. Yang, "Local 3D point cloud denoising via bipartite graph approximation & total variation," in *Proc. IEEE 20th Int. Workshop Multimedia Signal Process. (MMSP)*, Aug. 2018, pp. 1–6.

[44] S. Deutsch, A. Ortega, and G. Medioni, "Manifold denoising based on spectral graph wavelets," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 4673–4677.

[45] S. Deutsch, A. Ortega, and G. Medioni, "Robust denoising of piece-wise smooth manifolds," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 2786–2790.

[46] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "EC-Net: An edge-aware point set consolidation network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 386–402.

[47] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[48] G. M. Phillips, *Interpolation and Approximation By Polynomials*, vol. 14. New York, NY, USA: Springer-Verlag, 2003.

[49] S. G. Chang, B. Yu, and M. Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1532–1546, 2000.

[50] D. L. Donoho and I. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, Sep. 1994.

[51] D. L. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *J. Amer. Stat. Assoc.*, vol. 90, no. 432, pp. 1200–1224, Dec. 1995.

[52] P. Cignoni, C. Rocchini, and R. Scopigno, "METRO: Measuring error on simplified surfaces," *Comput. Graph. Forum*, vol. 17, no. 2, pp. 167–174, Sep. 1998.

[53] N. Aspert, D. Santa-Cruz, and T. Ebrahimi, "MESH: Measuring errors between surfaces using the Hausdorff distance," in *Proc. IEEE Int. Conf. Multimedia Expo*, vol. 1, Aug. 2002, pp. 705–708.

[54] D. Eberly. (1999). *Distance Between Point and Triangle in 3D*. Magic Software. [Online]. Available: http://www.magic-software.com/Documentation/pt3tri3.pdf

[55] V. Oniga and C. Chirilă, "Hausdorff distance for the differences calculation between 3D surfaces," *J. Geodesy Cadastre RevCAD*, vol. 15, no. 15, pp. 193–202, 2013.

[56] D. Girardeau-Montaut, "CloudCompare," Tech. Rep., 2016. [Online]. Available: http://www.cloudcompare.org/

[57] J. Zeng, G. Cheung, M. Ng, J. Pang, and C. Yang, "3D point cloud denoising using graph Laplacian regularization of a low dimensional manifold model," *IEEE Trans. Image Process.*, vol. 29, pp. 3474–3489, 2020.

[58] K. Fliegel, F. Battisti, M. Carli, M. Gelautz, L. Krasula, P. L. Callet, and V. Zlokolica, "3D visual content datasets," in *3D Visual Content Creation, Coding and Delivery*. Cham, Switzerland: Springer, 2019, pp. 299–325.

[59] A. Nouri, C. Charrier, and O. Lézoray, "Technical report: Greyc 3D colored database," Ph.D. dissertation, Normandie Université, Unicaen, EnsiCaen, CNRS, Paris, France, 2017.

[60] C. Couprie, L. Grady, L. Najman, J.-C. Pesquet, and H. Talbot, "Dual constrained TV-based regularization on graphs," *SIAM J. Imag. Sci.*, vol. 6, no. 3, pp. 1246–1273, Jan. 2013.

**MUHAMMAD ABEER IRFAN** (Graduate Student Member, IEEE) received the B.Sc. and M.Sc. degrees from the University of Engineering and Technology at Peshawar, Peshawar, Pakistan, in 2008 and 2013, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with the Politecnico di Torino, Turin, Italy.

He is currently a Lecturer with the University of Engineering and Technology at Peshawar. His research interests include image processing and graph signal processing and computer vision.

**ENRICO MAGLI** (Fellow, IEEE) received the M.Sc. and Ph.D. degrees from the Politecnico di Torino, Turin, Italy, in 1997 and 2001, respectively.

He is currently a Full Professor with the Politecnico di Torino. His research interests include deep learning, compressive sensing, image and video coding, and vision. He was a recipient of the 2011 Transactions Prize Paper Award by the IEEE Geoscience and Remote Sensing Society, the 2015 Best Student Paper Award (as the Senior Author) and the 2019 Best Paper Award by the IEEE International Conference on Image Processing (IEEE ICIP), and the 2010 and 2014 Best Associate Editor Award by the IEEE Transactions on Circuits and Systems for Video Technology. He is also an Associate Editor of the IEEE Transactions on Circuits and Systems for Video Technology and the *EURASIP Journal on Image and Video Processing* and a former Associate Editor of the IEEE Transactions on Multimedia. He was an IEEE Distinguished Lecturer from 2015 to 2016.