Nonlocal processing of 3D colored point clouds

François Lozes, Abderrahim Elmoataz, Olivier Lézoray

Nonlocal processing of 3D colored point clouds

François Lozes, Abderrahim Elmoataz, Olivier Lézoray
Université de Caen Basse-Normandie,GREYC UMR CNRS 6072
ENSICAEN - Image Team, 6 Bd. Maréchal Juin, F-14050 Caen, France.
{*francois.lozes,abderrahim.elmoataz-billah,olivier.lezoray*}*@unicaen.fr*

Abstract

In this paper we present a methodology for nonlocal processing of 3D colored point clouds using regularization of functions defined on weighted graphs. To adapt it to nonlocal processing of 3D data, a new definition of patches for 3D point clouds is introduced and used for nonlocal filtering of 3D data such as colored point clouds. Results illustrate the benefits of our non-local approach to filter noisy 3D colored point clouds (either on spatial or colorimetric information).

1. Introduction

With the advent of nonlocal processing of images [2], patch-based approaches have been recently very popular. However, the notion of patches is difficult to extend to 3D point clouds. Indeed, for images, the notion of patch relies on the spatial organization of pixels in a grid. With point clouds, the data to process is not organized on any Cartesian grid and the neighbors of a point have to be defined. Even with such neighbors defined (e.g., by creating a graph), one has no insurance that the number of neighbors of two different points will be the same and this makes difficult the comparison of these two neighborhoods. Therefore, there is actually no natural expression of what is a patch on a point cloud. Recently, some authors have experimented nonlocal denoising point clouds [6]. However this approach does not take into account any orientation information of the patch and is used only to denoise spatial information and not any spectral information attached to the points. In this paper, we propose an approach relying on the framework of Partial difference Equations (PdEs) to perform graph regularization [3] in order to filter 3D colored point clouds. To do so we introduce an innovative way to define patches for 3D points clouds in order to perform nonlocal processing.

2. Weighted Graphs

In this section, we recall definitions on graphs and operators on graphs. This constitutes the basis of the framework of PdEs to regularize a function on a graph.

2.1 Definitions

A weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$ consists in a finite set $\mathcal{V} = \{v_1, \ldots, v_N\}$ of N vertices and a finite set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ of weighted edges. We assume $\mathcal{G}$ to be undirected, with no self-loops and no multiple edges. Let (u, v) be the edge of $\mathcal{E}$ that connects vertices u and v of $\mathcal{V}$. Its weight, denoted by $w(u, v)$, represents the similarity between its vertices. Similarities are usually computed by using a positive symmetric function $w : \mathcal{V} \times \mathcal{V} \to \mathbb{R}^+$ satisfying $w(u, v) = 0$ if $(u, v) \notin \mathcal{E}$. The notation $u \sim v$ is also used to denote two adjacent vertices. Let $\mathcal{H}(\mathcal{V})$ be the Hilbert space of real-valued functions defined on the vertices of a graph. A function $f : \mathcal{V} \to \mathbb{R}$ of $\mathcal{H}(\mathcal{V})$ assigns a real value $f(v_i)$ to each vertex $v_i \in \mathcal{V}$. The space $\mathcal{H}(\mathcal{V})$ is endowed with the usual inner product $\langle f, h\rangle_{\mathcal{H}(\mathcal{V})} = \sum_{u\in\mathcal{V}} f(u)h(u)$, where $f, h : \mathcal{V} \to \mathbb{R}$. Similarly, one can define $\mathcal{H}(\mathcal{E})$, the space of real-valued functions defined on the edges of $\mathcal{G}$.

2.2 Difference operators on weighted graphs

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$ be a weighted graph, and let $f : \mathcal{V} \to \mathbb{R}$ be a function of $\mathcal{H}(\mathcal{V})$. The *difference operator* [3] of f, noted $d_w : \mathcal{H}(\mathcal{V}) \to \mathcal{H}(\mathcal{E})$, is defined on an edge $(v_i, v_j) \in \mathcal{E}$ by: $(d_w f)(v_i, v_j) = \sqrt{w(v_i, v_j)}(f(v_j) - f(v_i))$. The *directional derivative* (or *edge derivative* of f, at a vertex $v_i \in \mathcal{V}$, along an edge $e = (v_i, v_j)$, is defined as: $\partial_{v_j} f(v_i) = (d_w f)(v_i, v_j)$. The *adjoint* of the difference operator, noted $d_w^* : \mathcal{H}(\mathcal{E}) \to \mathcal{H}(\mathcal{V})$, is a linear operator defined by: $\langle d_w f, H\rangle_{\mathcal{H}(\mathcal{E})} = \langle f, d_w^* H\rangle_{\mathcal{H}(\mathcal{V})}$ for all $f \in \mathcal{H}(\mathcal{V})$ and all $H \in \mathcal{H}(\mathcal{E})$.

We obtain the expression of d_w^* by: $(d_w^* H)(v_i) = \sum_{v_j \sim v_i} \sqrt{w(v_i, v_j)}(H(v_j, v_i) - H(v_i, v_j))$. The *divergence operator*, defined by $-d_w^*$, measures the net outflow of a function of $\mathcal{H}(\mathcal{E})$ at each vertex of the graph. The *weighted gradient operator* of a function $f \in \mathcal{H}(\mathcal{V})$, at a vertex $v_i \in \mathcal{V}$, is the vector operator defined by: $(\boldsymbol{\nabla_w f})(\mathbf{v_i}) = [\partial_{v_j} f(v_i) : v_j \sim v_i]^T, \forall (v_i, v_j) \in \mathcal{E}$. The $\mathcal{L}_2$ norm of this vector represents the *local variation* of the function f at a vertex of the graph. It is defined by [3]: $\|(\boldsymbol{\nabla_w f})(\mathbf{v_i})\|_p = \Big[\sum_{v \sim v_i} w(v_i, v_j)^{p/2} \big|f(v_j) - f(v_i)\big|^p\Big]^{1/p}$.

2.3 Nonlocal regularization on graphs

In this section, one considers a general function $f^0 : \mathcal{V} \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$ defined on graphs of the arbitrary topologies and we want to regularize this function. The regularization of such a function corresponds to an optimization problem that can be formalized by the minimization of an energy as a weighted sum of two energy terms: $\frac{1}{2} \sum_{v_i \in \mathcal{V}} ||\boldsymbol{\nabla_w f}(\mathbf{v_i})||_2^2 + \frac{\lambda}{2} \|f - f^0\|_2^2$ The first term is the regularization term, meanwhile the second is the fitting term. The parameter $\lambda \geq 0$ is a fidelity parameter that specifies the trade-off between the two competing terms. Works of [3] have shown that solutions minimizing such an energy on weighted graphs can be obtained by the following iterative algorithm $\forall u \in \mathcal{V}$:

$$\begin{cases} f^{(0)}(v_i) = f^0(v_i) \\ f^{(n+1)}(v_i) = \dfrac{\lambda f^0(v_i) + \sum_{v \sim v_i} w(v_i, v_j) f^{(n)}(v_j)}{\lambda + \sum_{v_j \sim v_i} w(v_i, v_j)} \end{cases} \tag{1}$$

Equation (1) describes a family of discrete diffusion processes, which is parameterized by the structure of the graph (topology and weight function w), the parameter λ. As shown in [3], modifying both graph topology and graph weights enables to perform both local and nonlocal filtering within the same framework of PdEs. This iterative algorithm stops when a number of iterations is reached, or when the difference $||f^{n+1} - f^n||$ is small.

3. Processing of 3D point clouds

In contrast to existing works, we propose to make the most of weighted graphs to define the notion of patches for 3D colored point clouds. In addition to providing an innovative definition of patches for 3D point clouds, our approach can be used to denoise spatial or/and spectral properties of the point cloud. We review the principle of our approach that relies on four steps.

3.1 Graph creation from 3D point clouds

Let us consider a point cloud P as a set of data points $\{\mathbf{p}_1, \ldots, \mathbf{p}_n\} \in \mathbb{R}^3$. There are many ways of associating a graph, that encodes proximity between points, to such a data set. To each data point we first associate a vertex of a proximity graph $\mathcal{G}$ to define a set of vertices $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$. Then, determining the edge set $\mathcal{E}$ of the proximity graph $\mathcal{G}$ requires defining the neighbors of each vertex v_i according to its embedding $\mathbf{p}_i$ using the Euclidean distance. We will denote as $\mathcal{D}(v_i, v_j) = \|\mathbf{p}_i - \mathbf{p}_j\|_2$ the Euclidean distance between vertices and as $\mathcal{B}(v_i\ ;\ r) = \{\mathbf{p}_j \in \mathbb{R}^n \mid \mathcal{D}(v_i, v_j) \leq r\}$ the closed ball of radius r centered on $\mathbf{p}_i$. We consider two types of graphs: i) the ϵ-ball graph: $v_i \sim v_j$ if $\mathbf{p}_j \in \mathcal{B}(v_i\ ;\ \epsilon)$, ii) The k nearest neighbor graph (k-NNG): $v_i \sim v_j$ if the distance between $\mathbf{p}_i$ and $\mathbf{p}_i$ is among the k-th smallest distances from $\mathbf{x}_i$ to other data points. The first step consists in associating a k-NNG graph to the 3D point cloud.

3.2 Tangent plane estimation

Second step consists in computing an approximation of the tangent plane of a point $\mathbf{p}_i$ (or the vertex v_i). Classically (see [5]), the PCA of the covariance matrix of the neighbors of v_i in a local ϵ-ball graph around v_i is considered. Let $\overline{\mathbf{p}}(v_i)$ be the centroid of the neighbors of v_i. The covariance matrix of the local frame is $\mathbf{C} = \big[\mathbf{p}_j - \overline{\mathbf{p}}(v_i)\big] \cdot \big[\mathbf{p}_j - \overline{\mathbf{p}}(v_i)\big]^T$ with $v_j \sim v_i$. From this matrix, eigenvalues $\lambda_0 < \lambda_1 < \lambda_2$ and eigenvectors $\mathbf{t_0}(v_i), \mathbf{t}_1(v_i), \mathbf{t}_2(v_i)$ are computed. Eigenvectors $\mathbf{t}_1(v_i)$ and $\mathbf{t}_2(v_i)$ form an orthogonal basis of the tangent plane, and $\mathbf{t}_0(v_i)$ is normal to this tangent plane (Figure 1).

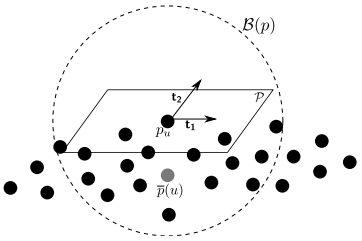

Figure 1: Estimation of the tangent plane for a given point $\mathbf{p}_i$ according to a set of neighbors. The centroid the set of neighbors of p_i in $\mathcal{B}(v_i\ ;\ r)$ is $\overline{\mathbf{p}}(v_i)$. Computed tangent vectors are $\mathbf{t}_1(v_i)$ and $\mathbf{t}_2(v_i)$.

3.3 Orientation estimation

Third step consists in estimating orientations. Indeed, patches are oriented from principal directions. This means that directions of first and second axis of the patch basis will coincide respectively with major and minor principal directions. To compute these principal directions at point $\mathbf{p}_i$, we use the arguments of [1]. Principal directions and principal curvatures can be estimated from the PCA of the covariance matrix of the normals of the neighbors of $\mathbf{p}_i$ projected on the tangent plane of $\mathbf{p}_i$. Let $\mathbf{n}(v_j)$ be the normal associated to node v_j (this is estimated [4] from the tangent plane normal as $\pm\mathbf{t_0}(v_i)$) and $\boldsymbol{\xi}(v_j) = \left[\mathbf{n}(v_j)\cdot\mathbf{t_1}(v_i); \mathbf{n}(v_j)\cdot\mathbf{t_2}(v_i)\right]^T$ be the projection of the normal $\mathbf{n}(v_j)$ on the tangent plane $(\mathbf{t_1}(v_i), \mathbf{t_2}(v_i))$. Then the covariance matrix C_n of these projected normals of all the neighbors of a given point (in the local $epsilon$-ball graph) is computed as $\mathbf{C}_n = \left[\boldsymbol{\xi}(v_j) - \overline{\boldsymbol{\xi}}(v_i)\right]\cdot\left[\boldsymbol{\xi}(v_j) - \overline{\boldsymbol{\xi}}(v_i)\right]^T$ with $v_j \sim v_i$ and $\overline{\boldsymbol{\xi}}(v_i)$ the centroid of the projected normals. From this matrix, eigenvalues $\lambda_0 < \lambda_1$ and eigenvectors $\mathbf{c}_0(v_i), \mathbf{c}_1(v_i)$ are computed. Then, $\mathbf{c}_0(v_i), \mathbf{c}_1(v_i)$ are respectively estimations of the minor and major principal directions.

3.4 Patch construction

Final step consists in constructing the patches. Given a point $\mathbf{p}_i$, defining a patch for this point comes to construct a square grid around $\mathbf{p}_i$ on its tangent plane. We fix the patch length as $l = 2\times\max_{v_j\sim v_i}\|\mathbf{p}_j - \mathbf{p}_i\|$. A square lattice of n^2 cells is then constructed around $\mathbf{p}_i$ with respect to the basis obtained from principal directions $\mathbf{c}_0(v_i), \mathbf{c}_1(v_i)$. Each cell has a side length of l/n. Then, all the neighbors v_j of v_i are projected on the tangent plane of $\mathbf{p}_i$ giving rise to projected points $\mathbf{p}'_j$. To fill the patch with values, these projected points $\mathbf{p}'_j$ are affected to the cells the center of which is the closest. The value of the cell is then deduced from the average of the values $f^0(v_j)$ associated to the vertices v_j that where affected to the patch cell. This value can be a spatial value (the projected points' coordinates) or a spectral one (the points' colors). The set of values inside the patch of the vertex v_i are denoted as $\boldsymbol{\mathcal{P}}(v_i)$. Figure 2 summarizes the method.

4. Experiments

The proposed framework is used to denoise coordinates of noisy 3D point clouds and to filter colored point clouds. Let $f^0 : \mathcal{V} \to \mathbb{R}^3$ be the function

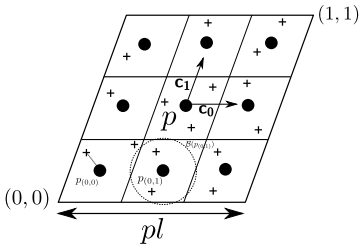

Figure 2: Interpolation of the content of the patch. l is the patch length. $\mathbf{c}_0(v_i)$ and $\mathbf{c}_1(v_i)$ are the principal directions at a point $\mathbf{p}_i$. Elements marked by a "+" symbol correspond to the projected neighbors of $\mathbf{p}_i$ on the patch. These projections are used to deduce values of each patch cell.

that associates either 3D coordinates or RGB colors to each node $u \in \mathcal{V}$. From this function, a graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, w)$ is first created. We consider 10 nearest neighbor graphs for all the experiments with 10 iterations of algorithm (1). The graph is weighted with $w(v_i, v_j) = exp(-\frac{||\mathbf{F(f^0, v_i)} - \mathbf{F(f^0, v_j)}||^2}{\sigma^2})$. If ones considers local weights, $\mathbf{F(f^0, v_i)} = f^0(v_i)$. In the case of nonlocal weights (based on patches), one has $\mathbf{F(f^0, v_i)} = \boldsymbol{\mathcal{P}}(v_i)$. We first consider the case of 3D point clouds where $f^0(v_i) = \mathbf{p}_i$. Figure 3 presents results on two points clouds corrupted by Gaussian noise and compares the effect of local versus nonlocal regularization. The benefit of our formulation of nonlocal denoising of 3D point clouds is evident with a final point cloud much more close to the shape of the original uncorrupted one. Second we consider the case of 3D colored point clouds where $f^0(v_i) = [R_i, G_i, B_i]^T$. Figure 4 presents results of color filtering on a given 3D colored point clouds (obtained from a laser scan of a Maya temple wall). It is important to notice that this point cloud is not a triangular mesh but a raw 3D colored point cloud. As attended, local filtering tends to remove many details while blurring borders. On the opposite, our formulation of nonlocal filtering achieves a much better job while preserving edges and providing uniform zones of similar textures.

5. Conclusion

In this paper we proposed a nonlocal processing of 3D colored point clouds. Using nonlocal regularization on graphs to filter point clouds, we demonstrated how to define the notion of patches for point clouds. To define a patch a given point $\mathbf{p}_i$, its tangent plane is first determined and its principal directions are estimated. Then, a grid lattice representing the patch is constructed around the point $\mathbf{p}_i$ and all the neighbors of $\mathbf{p}_i$ in the graph are

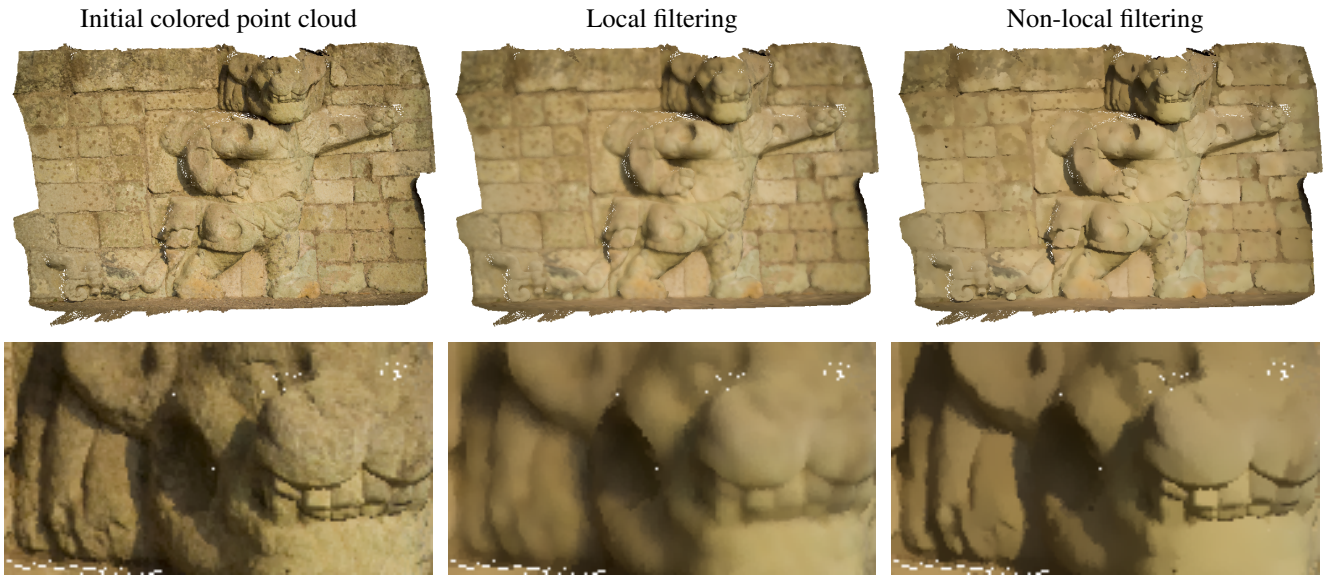

Figure 4: Local and nonlocal (3×3 patches) filtering of a 3D colored point cloud acquired by a laser scan of a Maya temple wall. First line present the full point cloud. Second line presents a cropped and zoomed area.

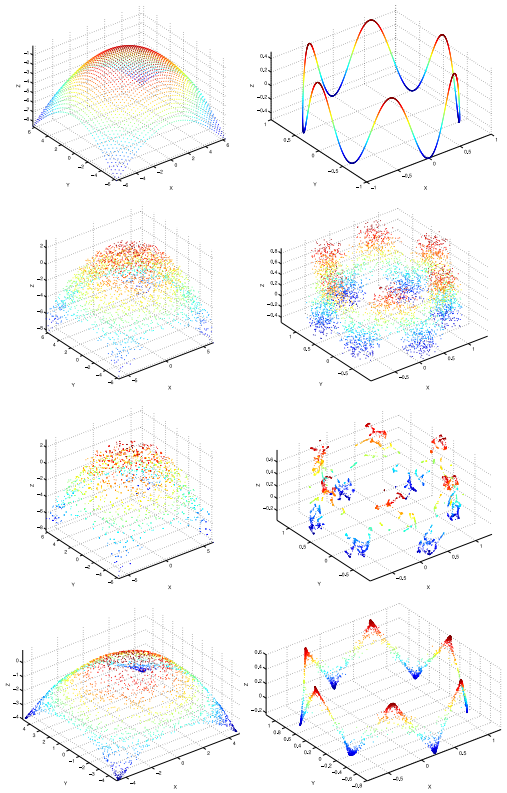

Figure 3: Denoising of two noisy 3D point clouds coordinates (in columns). From top to bottom lines: original, corrupted ($\sigma = 20$), locally denoised, and nonlocally (3×3 patches) denoised point clouds.

projected onto the grid. On this grid, an interpolation is finally used from projected points to fill the patch. Experimental results have shown the benefits of the approach that enables the nonlocal filtering of 3D colored point clouds within a unified formalism. Future works will consider the case of inpainting 3D colored point clouds.

References

[1] J. Berkmann and T. Caelli. Computation of surface geometry and segmentation using covariance techniques. *IEEE Trans. Pat. Anal. Mach.*, 16(11):1114–1116, 1994.

[2] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *CVPR*, pages 60–65, 2005.

[3] A. Elmoataz, O. Lezoray, and S. Bougleux. Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing. *IEEE Trans. on Im. Proc.*, 17(7):1047–1060, 2008.

[4] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *SIGGRAPH Comput. Graph.*, 26(2):71–78, 1992.

[5] M. Pauly. *Point primitives for interactive modeling and processing of 3D geometry*. Selected readings in vision and graphics. Hartung-Gorre, 2003.

[6] R.-F. Wang, W.-Z. Chen, S.-Y. Zhang, Y. Zhang, and X.-Z. Ye. Similarity-based denoising of point-sampled surfaces. *J. of Zhejiang. Univ. -Science A*, 9(6):807–815, 2008.