Multiscale Representation of Surfaces by Tight Wavelet Frames with Applications to Denoising

Bin Dong[a]*, Qingtang Jiang[b], Chaoqiang Liu[c], and Zuowei Shen[c]

[a]*Department of Mathematics, University of Arizona, Tucson, AZ 85721, USA*
dongbin@math.arizona.edu

[b]*Department of Mathematics and Computer Science,*
University of Missouri–St. Louis, St. Louis, MO 63121, USA
jiangq@umsl.edu

[c]*Department of Mathematics, National University of Singapore*
10 Lower Kent Ridge Road, Singapore, 119076
matliucq@nus.edu.sg (C.Q. Liu), matzuows@nus.edu.sg (Z.W. Shen)

Abstract

In this paper, we introduce a new multiscale representation of surfaces using tight wavelet frames. Both triangular and quadrilateral (quad) surfaces are considered. The multiscale representation for triangulated surfaces is generalized from the non-tensor-product tight wavelet frame representation of functions (of two variables) that were introduced by [1], while the tensor-product tight frames of continuous linear B-spline from [62] are used for quad surfaces representation. As one of many possible applications of such representation, we consider surface denoising as an example.

1 Introduction

1.1 Multiscale Representation of Surfaces

Wavelet, or more generally, multiscale representation of functions is well studied in the past thirty years. However, when one deals with surfaces instead of functions (such as images), most of the existing theories and numerical algorithms cannot be directly applied. The major difficulty is that there is not a trivial way of associating each surface with a function that preserves important geometric features of the surface, and such association is often non-unique.

Many attempts have been made in the literature, some of which are quite successful in computer graphics and (medical) shape/surface analysis. One attempt was to map each surface to a simple surface that is easily parameterized, such as the unit sphere. Then, the multiscale representation can be given by the spherical wavelets [2] via lifting scheme [3]. As proposed in [4,5], the authors first used conformal mapping to map the surface onto the unit sphere. Then, they applied spherical wavelet decomposition to each (x, y, z)-components of the mapping. Another similar method was proposed by [6], where the surface was first map to the unit sphere using a conformal mapping developed by [7]; then, instead of

*Corresponding author

decomposing the function on the sphere, the sphere was cut open and transformed to a square via a method introduced by [8]. In this way, one associates each surface with a vector-valued image, and standard wavelet transforms can be applied to each of (x, y, z)-components of the vector-valued image. The major drawback of the aforementioned approaches is that the mapping between surfaces and the unit sphere is non-unique and not trivial to find. A low quality mapping associated to a given surface may hamper the quality of the underlying multiscale representation.

Alternatively, one can apply the idea of spherical wavelet representation directly on the surface itself by regarding it as the vector-valued function with domain being the surface and function being the (x, y, z)-coordinates of the vertices. With the idea of the lifting scheme, biorthogonal wavelets with high symmetry for surface multiresolution processing have been constructed in [9–14]. If the biorthogonal wavelets have certain smoothness, they will have big supports. In other words, the corresponding multiscale algorithms have large templates, and this is not desirable for surface processing. Loop's scheme-based biorthogonal wavelets have been considered in [15] with the biorthogonal dual wavelets constructed in [16]. However the corresponding highpass filters do not have desirable symmetry for surface processing with extraordinary vertices. This undesirable property will cause problems in designing the associated algorithms for extraordinary vertices. More recently in [17], 6-fold symmetric bi-frames with 4 framelets (frame generators) for triangulated surfaces were introduced. Such 6-fold symmetric bi-frames yield frame decomposition and reconstruction algorithms (for regular vertices) with high symmetry, which is required for the design of the corresponding frame multiresolution algorithms for extraordinary vertices on the triangular mesh. Compared with biorthogonal wavelets, the constructed bi-frames have better smoothness and smaller supports. In addition, frame multiresolution algorithms for extraordinary vertices are provided.

Although the bi-frames representation of surfaces by [17] has the properties of short-support and high-symmetry, it is relatively sensitive to perturbations of the coefficients, since most of the bi-frames constructed in [17] are not using the canonical duals. Canonical dual of a frame system has the least sensitivity to perturbations of the coefficients, while the frame functions may have very large supports which is unacceptable in many applications. In this paper, we shall propose a *tight* wavelet frame representation for triangular and quad surfaces. Tight wavelet frame systems are self-dual system, which means its canonical dual system is identical to its primal system. In other words, the proposed representation is not sensitive to perturbations of the coefficients. In addition, each of the tight frame functions in the system also has very short support and desirable symmetry. These properties are desirable for many surface processing applications such as surface denoising.

In recent years, multiscale representation on graphs has been well studied in the literature. Since triangulated or quad surfaces are special cases of graphs, these methods provide multiscale representation for surfaces as well. Successful examples include the wavelets on unweighted graphs by [18], the multiscale scheme on graphs based on lifting by [19], the Haar wavelet transform for rooted binary trees [20] and its generalization treelets [21], the diffusion wavelets [22] and diffusion polynomial frames [23], the wavelets on compact differentiable manifolds [24], the spectral graph wavelet transform by [25,26], Haar transform for coherent matrices [27], and orthogonal polynomial systems for weighted trees [28].

We finally note that, an alternative representation of surfaces is implicit representation using, for instance, level set functions. Although implicit representation of surfaces is commonly not as efficient as triangular meshes, it is free of parametrization which is a huge advantage for the applications where topology change occurs during the numerical computations. In [29], the authors introduced a new multiscale representation for implicitly represented surfaces via level set motions and nonlinear PDEs. The main idea of [29] is to generate a sequence of multiscaled approximation of the original surface via level set motions, such as mean/average curvature motion, and then define the wavelet coefficients as the tangents of the characteristic curves which are vector fields living on the surfaces at different scales. Based on such multiscale representation, they designed a surface inpainting algorithm to recover 3D geometry of blood vessels. For surface inpainting problems, topology changes may occur during the processing and thus multiscale representations for triangulated surfaces may not be suitable for inpainting problems. However, the drawback of the multiscale representation by [29] is the approximated reconstruction, instead of exact reconstruction, for the inverse transformation.

In this paper, we propose a multiscale representation of triangulated and quad surfaces by generalizing various types of existing tight wavelet frame systems for functions. We will discuss how convolutions of wavelet frame masks with standard grid data can be generalized to surface data, and how multiple level transforms can be defined in a similar way as classical wavelets. The new wavelet frame transform can be perfectly parallelized and computed rather efficiently.

1.2 Surface Denoising

As one of many potential applications of the proposed tight wavelet frame representation of surfaces, we shall consider surface denoising in this paper.

Variational and partial differential equations (PDEs) based image denoising models have had great success in the past thirty years (see e.g. [30–33]). The objective of image denoising is to remove noise and artifacts from an observed image, while preserving key image features such as sharp edges. Some of the models used for image denoising have been extended to denoising surfaces (see [34–39]). Based on ideas of nonlocal means introduced in [40] for image denoising, a nonlocal averaging algorithm for denoising triangulated surfaces were introduced by [41]. The nonlocal means was also generalized to implicit surface denoising in [42], which has the advantage over [41] in restoring surface topology. In [43], a nonlocal discrete regularization framework was introduced, which is the discrete analogue of the continuous Euclidean nonlocal regularization functionals by [44]. This method is applied to image and manifold processing using weighted graphs of arbitrary topologies. In [45], a variational model for triangulated surface denoising by minimizing the L_1-norm of the Gaussian curvature on the given surface were proposed, which is analogous to the well-known Rudin-Osher-Fatemi model [30] for image denoising. Other approaches for surface denoising include Laplacian smoothing based on Winer-filter-type shrinkage [46], bilateral filtering [47, 48].

In this paper, we propose an analysis based model based on the proposed new tight wavelet frame representation of surfaces. The analysis based model is a generalized from the model used in image restoration [49, 50], which can be solved efficiently using the split Bregman algorithm [49, 51]. Other than the analysis based model, the balanced model [52–56] and the synthesis based model [57–61] are also widely used with success in image restoration.

1.3 Organization of the paper

We start in Section 2 with a review of some basic concepts and theories of tight wavelet frames, and a review of the construction of non-tensor-product tight wavelet frames of [1]. We elaborate how to generalize the tight wavelet frame representation for functions to triangulated surfaces in Section 3. In Section 4, we present the analysis based model for surface denoising and the associated fast optimization algorithm. Numerical experiments will also be presented. In Section 5, we consider quad surface wavelet frame representation and surface denoising problem.

2 Tight Wavelet Frames on $\mathbb{R}^d$

In this section, we briefly introduce the concept of tight frames and tight wavelet frames on $\mathbb{R}^d$. The interested readers should consult [62–64] for theories of frames and wavelet frames, [65] for a short survey on the theory and applications of frames, and [66] for a more detailed survey. Then we recall examples of tensor-product tight frames from [62] and non-tensor-product tight wavelet frames of [1].

2.1 Tight Frames and Unitary Extension Principle (UEP)

A countable set $X \subset L_2(\mathbb{R}^d)$, with $d \in \mathbb{N}$, is called a tight frame of $L_2(\mathbb{R}^d)$ if

$$f = \sum_{g\in X} \langle f, g\rangle g \quad \forall f \in L_2(\mathbb{R}^d), \tag{2.1}$$

where $\langle\cdot,\cdot\rangle$ is the inner product of $L_2(\mathbb{R}^d)$. For given $\Psi := \{\psi_1, \cdots, \psi_L\} \subset L_2(\mathbb{R}^d)$, the corresponding quasi-affine system $X(\Psi)$ generated by Ψ is defined by the collection of the dilations and the shifts of Ψ as

$$X(\Psi) = \{\psi_{\ell,n,\boldsymbol{k}} :\ 1 \le \ell \le L; n \in \mathbb{Z}, \mathbf{k} \in \mathbb{Z}^d\}, \tag{2.2}$$

where $\psi_{\ell,n,\boldsymbol{k}}$ is defined by

$$\psi_{\ell,n,\boldsymbol{k}} := \begin{cases} 2^{\frac{nd}{2}}\psi_\ell(2^n \cdot -\boldsymbol{k}), & n \ge 0; \\ 2^{nd}\psi_\ell(2^n \cdot -2^n\boldsymbol{k}), & n < 0. \end{cases} \tag{2.3}$$

When $X(\Psi)$ forms a (tight) frame of $L_2(\mathbb{R}^d)$, each function ψ_ℓ, $\ell = 1, \cdots, L$, is called a (tight) framelet and the whole system $X(\Psi)$ is called a (tight) wavelet frame system.

The constructions of framelets Ψ, which are desirably (anti)symmetric and compactly supported functions, are usually based on a multiresolution analysis (MRA) that is generated by some refinable function ϕ with refinement mask $\boldsymbol{a}_0$ satisfying

$$\phi = 2^d \sum_{\boldsymbol{k}\in\mathbb{Z}^d} \boldsymbol{a}_0[\boldsymbol{k}]\phi(2\cdot -\boldsymbol{k}). \tag{2.4}$$

The idea of an MRA-based construction of framelets $\Psi = \{\psi_1, \cdots, \psi_L\} \subset L_2(\mathbb{R}^d)$ is to find masks $\boldsymbol{a}_\ell$, which are finite sequences, such that

$$\psi_\ell = 2^d \sum_{\boldsymbol{k}\in\mathbb{Z}^d} \boldsymbol{a}_\ell[\boldsymbol{k}]\phi(2\cdot -\boldsymbol{k}), \quad \ell = 1, 2, \cdots, L. \tag{2.5}$$

The sequences $\boldsymbol{a}_1, \cdots, \boldsymbol{a}_L$ are called wavelet frame masks, or the highpass filters of the system, and the refinement mask $\boldsymbol{a}_0$ is also known as the lowpass filter. The set $\{\boldsymbol{a}_0, \boldsymbol{a}_1, \cdots, \boldsymbol{a}_L\}$ is called a frame filter bank.

The unitary extension principle (UEP) of [62] provides a general theory of the construction of MRA-based tight wavelet frames. Roughly speaking, as long as $\{\boldsymbol{a}_1, \cdots, \boldsymbol{a}_L\}$ are finitely supported and their Fourier series satisfy

$$\sum_{\ell=0}^{L} \left|\widehat{\boldsymbol{a}}_\ell(\xi)\right|^2 = 1, \quad \text{and} \tag{2.6}$$

$$\sum_{\ell=0}^{L} \widehat{\boldsymbol{a}}_\ell(\xi)\overline{\widehat{\boldsymbol{a}}_\ell(\xi+\nu)} = 0, \tag{2.7}$$

for all $\nu \in \{0,\pi\}^d \backslash \{\mathbf{0}\}$ and $\xi \in [-\pi,\pi]^d$, the quasi-affine system $X(\Psi)$ (as well as the traditional wavelet system) with $\Psi = \{\psi_1, \cdots, \psi_L\}$ defined by (2.5) forms a tight frame in $L_2(\mathbb{R}^d)$.

2.2 Tight Wavelet Frames on $\mathbb{R}^2$

In this paper, we shall consider tight frames of $L_2(\mathbb{R}^d)$ with $d = 2$ since surfaces are essentially 2-dimensional objects. One common way to construct tight frames for $L_2(\mathbb{R}^2)$ (or $L_2(\mathbb{R}^d)$ in general) is by taking *tensor-products* of univariate tight frames. Given a set of univariate masks $\{a_\ell : \ell = 0, 1, \cdots, r\}$, define the 2-dimensional masks $\boldsymbol{a_i}[\boldsymbol{k}]$, with $\boldsymbol{i} := (i_1, i_2)$ and $\boldsymbol{k} := (k_1, k_2)$, as

$$\boldsymbol{a_i}[\boldsymbol{k}] := a_{i_1}[k_1] a_{i_2}[k_2], \quad 0 \le i_1, i_2 \le r; \ (k_1, k_2) \in \mathbb{Z}^2. \tag{2.8}$$

Then the corresponding 2-dimensional refinable function and framelets are defined by

$$\psi_{\boldsymbol{i}}(x,y) = \psi_{i_1}(x)\psi_{i_2}(y), \quad 0 \le i_1, i_2 \le r; (x,y) \in \mathbb{R}^2,$$

where we have let $\psi_0 := \phi$ for convenience. We denote

$$\boldsymbol{\Psi} := \{\psi_{\boldsymbol{i}}; \ 0 \le i_1, i_2 \le r; \ \boldsymbol{i} \neq (0,0)\}.$$

If the univariate masks $\{a_\ell\}$ are constructed from UEP, then it is easy to verify that $\{\boldsymbol{a_i}\}$ satisfies (2.6) and (2.7) and thus $X(\boldsymbol{\Psi})$ is a tight frame for $L_2(\mathbb{R}^2)$.

Example 2.1. *Let a_0, a_1, a_2 be the univariate tight frame filters constructed in [62]:*

$$a_0 = [\frac{1}{4}, \frac{1}{2}, \frac{1}{4}], \ a_1 = [\frac{\sqrt{2}}{4}, 0, -\frac{\sqrt{2}}{4}], \ a_2 = [-\frac{1}{4}, \frac{1}{2}, -\frac{1}{4}]. \tag{2.9}$$

a_0 is the refinement mask of univariate continuous linear spline supported on $[-1,1]$.

Taking tensor-products of these filters, we have 2-dimensional tight frame filters:

$$\begin{aligned}
&p = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \ q_1 = \frac{\sqrt{2}}{16}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \\
&q_2 = \frac{\sqrt{2}}{16}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, \ q_3 = \frac{1}{16}\begin{bmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{bmatrix}, \\
&q_4 = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}, \ q_5 = \frac{1}{16}\begin{bmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{bmatrix}, \\
&q_6 = \frac{\sqrt{2}}{16}\begin{bmatrix} 1 & -2 & 1 \\ 0 & 0 & 0 \\ -1 & 2 & -1 \end{bmatrix}, q_7 = \frac{\sqrt{2}}{16}\begin{bmatrix} -1 & 0 & 1 \\ 2 & 0 & -2 \\ -1 & 0 & 1 \end{bmatrix}, \\
&q_8 = \frac{1}{16}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}. \qquad \square
\end{aligned} \tag{2.10}$$

Tensor-product wavelet frames usually works well for quad surface processing, but not for triangulated surfaces. In addition, wavelet frames constructed from tensor-product usually have many highpass filters. For example, as shown in Example 2.1, eight 2-dimensional highpass filters are generated by the tensor-product process from a univariate tight frame with two highpass filters. Furthermore, wavelet frames constructed from tensor-product have in general larger support than necessary. In other words, the corresponding filters have more non-zero entries than necessary. This leads to higher computation cost to compute the wavelet frame decomposition and reconstruction algorithm than non-tensor-product wavelet frames.

Since each finitely supported mask corresponds to a Laurant polynomial in Fourier domain, many existing constructions are based on matrix completions with matrix's entries being Laurant polynomials, so that the UEP conditions (2.6) and (2.7) are satisfied. More recently in [1], the authors introduced a much simpler way of constructing non-tensor-product tight wavelet frames. Their construction is supported by the generic theory they developed in [1] on dual Gramian analysis on Hilbert spaces. We refer the interested readers to [1] for details. Here, we shall recall the refinement masks of the non-tensor-product tight wavelet frames on $\mathbb{R}^2$ constructed in [1] using linear bivariate box spline with directions $(1,0)^\top$, $(0,1)^\top$ and $(1,1)^\top$.

Example 2.2. *The refinement mask of this box spline is*

$$\boldsymbol{a}_0 = \frac{1}{8}\begin{bmatrix} 1 & 1 & \\ 1 & 2 & 1 \\ & 1 & 1 \end{bmatrix}$$

and the six wavelet frame masks are

$$\boldsymbol{a}_1 = \frac{1}{8}\begin{bmatrix} -1 & -1 & \\ 1 & 2 & 1 \\ & -1 & -1 \end{bmatrix}, \quad \boldsymbol{a}_2 = \frac{1}{8}\begin{bmatrix} 1 & -1 & \\ -1 & 2 & -1 \\ & -1 & 1 \end{bmatrix},$$

$$\boldsymbol{a}_3 = \frac{1}{8}\begin{bmatrix} -1 & 1 & \\ -1 & 2 & -1 \\ & 1 & -1 \end{bmatrix}, \quad \boldsymbol{a}_4 = \frac{\sqrt{3}}{12}\begin{bmatrix} -1 & -1 & \\ 1 & 0 & -1 \\ & 1 & 1 \end{bmatrix},$$

$$\boldsymbol{a}_5 = \frac{\sqrt{6}}{24}\begin{bmatrix} 1 & 1 & \\ 2 & 0 & -2 \\ & -1 & -1 \end{bmatrix}, \quad \boldsymbol{a}_6 = \frac{\sqrt{2}}{8}\begin{bmatrix} 1 & -1 & \\ 0 & 0 & 0 \\ & 1 & -1 \end{bmatrix}. \qquad \square$$

2.3 Fast Transform

We focus on the case $d = 2$. In the discrete setting, the data considered is a 2-dimensional array. We denote by

$$\mathcal{I}_2 := \mathbb{R}^{N_1 \times N_2}$$

the set of all 2-dimensional arrays of size $N_1 \times N_2$. We will further assume that $N_1 = N_2 = N$. Note that these assumptions are not essential, and all arguments and results in this paper can be easily extended to more general cases. We denote the 2-dimensional fast (discrete) frame transform (see, e.g., [66]) with levels of decomposition J as

$$\boldsymbol{W}\boldsymbol{u} = \{\boldsymbol{W}_{j,\boldsymbol{i}}\boldsymbol{u} : 0 \le j \le J-1, 0 \le i_1 \le r_1, 0 \le i_2 \le r_2\}, \qquad \boldsymbol{u} \in \mathcal{I}_2. \tag{2.11}$$

We denote the wavelet frame bands (high frequency bands) as $\mathbb{B} = \{\boldsymbol{i} : 0 \leq i_1 \leq r_1, 0 \leq i_2 \leq r_2\} \setminus \{\boldsymbol{0}\}$. The fast frame transform $\boldsymbol{W}$ is a linear operator with $\boldsymbol{W}_{j,\boldsymbol{i}}\boldsymbol{u} \in \mathcal{I}_2$ denoting the frame coefficients of $\boldsymbol{u}$ at level j and band $\boldsymbol{i}$. Furthermore, we have

$$\boldsymbol{W}_{j,\boldsymbol{i}}\boldsymbol{u} := \boldsymbol{a}_{j,\boldsymbol{i}}[-\cdot] \circledast \boldsymbol{u}, \tag{2.12}$$

where $\circledast$ denotes the convolution operator with a certain boundary condition, e.g., periodic boundary condition, and $\boldsymbol{a}_{j,\boldsymbol{i}}$ is defined as

$$\boldsymbol{a}_{j,\boldsymbol{i}} = \tilde{\boldsymbol{a}}_{j,\boldsymbol{i}} \circledast \tilde{\boldsymbol{a}}_{j-1,\boldsymbol{0}} \circledast \cdots \circledast \tilde{\boldsymbol{a}}_{0,\boldsymbol{0}} \quad \text{with} \quad \tilde{\boldsymbol{a}}_{j,\boldsymbol{i}}[\boldsymbol{k}] = \begin{cases} \boldsymbol{a}_{\boldsymbol{i}}[2^{-j}\boldsymbol{k}], & \boldsymbol{k} \in 2^j\mathbb{Z}^2; \\ 0, & \boldsymbol{k} \notin 2^j\mathbb{Z}^2. \end{cases} \tag{2.13}$$

Notice that $\boldsymbol{a}_{0,\boldsymbol{i}} = \boldsymbol{a}_{\boldsymbol{i}}$. The wavelet frame transform (2.12) is known as the undecimated wavelet frame transform in the literature. When (2.6) holds, $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$. That is $\boldsymbol{u}$ can be recovered from $\boldsymbol{W}\boldsymbol{u}$ by the tight frame inverse transform $\boldsymbol{W}^\top$.

In the literature of wavelets and wavelet frames, there are generally two types of wavelet frame transforms that are most frequently used, namely the decimated and undecimated wavelet/frame transforms (2.12). Decimation on triangulated surfaces can be done in a similar manner as regular cartesian mesh. However, if the triangulation of a given surface is not generated by subdividing a coarser mesh, then the mesh is not readily available for decimated wavelet/frame transform and the surface needs to be remeshed/sampled to have a subdivision connectivity. The undecimated wavelet/frame transform, on the other hand, always operates on the original triangulation. We only need to define dilation of masks on triangular meshes, which is in general easier to do. The same issue arises when decimated wavelet/frame transforms are applied to quad surfaces which in general do not have a subdivision connectivity. Therefore, we shall focus on the undecimated wavelet frame transform of a given triangulated/quad surface $\boldsymbol{S}$.

3 Tight Wavelet Frame Representation for Triangulated Surfaces

In this section, we introduce our tight wavelet frame representation for triangulated surfaces. Given a triangulated surface $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{T}\}$, where $\boldsymbol{V}$ is the set of vertices and $\boldsymbol{T}$ the set of triangles. The major challenge of generalizing the fast algorithm given in the previous section to triangulated surfaces is the definition of convolution of the wavelet frame masks (e.g. the ones in Example 2.2) on triangles.

To properly generalize convolution on triangulated surfaces, we introduce a so-called data-matrix generation procedure that converts a given surface to a data-matrix. Then, we introduce one-level, followed by multiple-level wavelet frame transforms. The data-matrix generation operation is associated with the support of the masks. More precisely, let

$$\text{supp}(\boldsymbol{a}) = \bigcup_{\boldsymbol{i}} \left\{[j,k] : \boldsymbol{a}_{\boldsymbol{i}}[j,k] \neq 0\right\} \tag{3.1}$$

be the support of a frame filter bank $\boldsymbol{a} = \{\boldsymbol{a}_{\boldsymbol{i}}\}_{\boldsymbol{i}}$. Then each column of the data-matrix is a set of vertices neighboring a vertex. The size of the neighborhood matching the support of the frame filter bank. For the simplicity of presentation, in Section 3.1 and Section 3.2, we shall focus on the specific example of Example 2.2 and define our wavelet frame representation for triangulated surfaces using this tight frame. In Section 3.3, we will discuss how we can define wavelet frame representations for surfaces using masks other than those given in Example 2.2.

3.1 One-Level Tight Wavelet Frame Transform

Notice from the masks given in Example 2.2, the masks generally have seven nonzero elements that match with the standard 6-valence triangulation (see Fig.1). Based on such observation, given a surface $\boldsymbol{S}$, we first conduct a *data-matrix generation* procedure associated with the support of these masks before the actual wavelet frame transforms are performed. Such procedure converts a given surface $\boldsymbol{S}$ to a data-matrix. This process is to ready the surface for convolution with wavelet frame masks. We note that the data-matrix generation operator and the tight wavelet frame transforms given below do apply to general triangulated surfaces whose vertices may or may not have exactly 6 immediate neighbors.

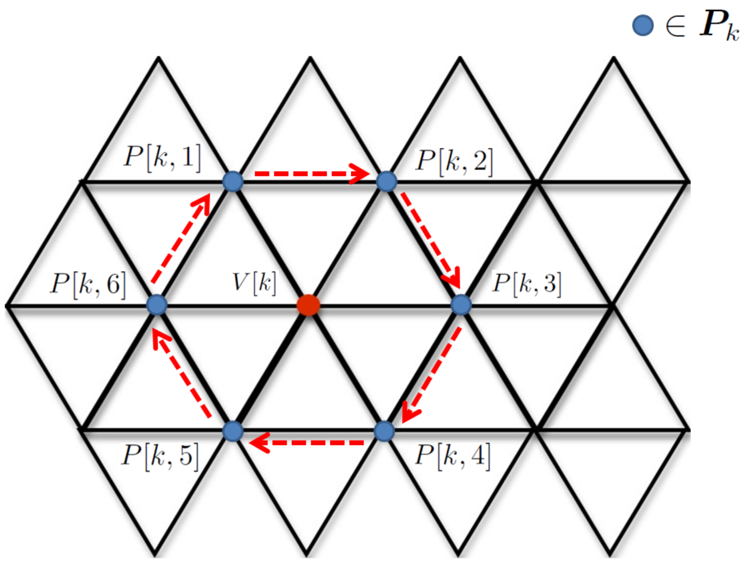

Figure 1: This figure illustrates how the neighboring vertices (forming the set $\boldsymbol{P}$) of a given vertex $V \in \boldsymbol{V}$ are ordered.

Data-Matrix Generation

Given a triangulated surface $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{T}\}$, we perform the following operations that produce a data-matrix $\boldsymbol{D}$. We shall denote the following operations as $\mathcal{C}$, i.e. $\mathcal{C}(\boldsymbol{S}) = \{\boldsymbol{D}, \boldsymbol{T}\}$. Note that during this process, as well as the actual wavelet frame transforms that we shall define afterwards, we do not alter the original triangulation $\boldsymbol{T}$. Therefore, we omit $\boldsymbol{T}$ and simply write $\mathcal{C}(\boldsymbol{S}) = \boldsymbol{D}$.

1. Given $\boldsymbol{V}$, we let $\boldsymbol{V} =: \{V[k] \mid k = 1, 2, \cdots N\}$, where

$$V[k] = (V_1[k], V_2[k], V_3[k])^\top$$

is the (x, y, z)-coordinates of the vertex $V[k]$.

2. For each $k = 1, 2, \cdots, N$, if $V[k]$ is regular (meaning it has valence 6), we find the six (immediate) neighboring vertices of the vertex $V[k]$. The set of neighboring vertices is denoted as

$$\boldsymbol{P}_k = \{P[k,1],\ P[k,2],\ P[k,3],\ P[k,4],\ P[k,5],\ P[k,6]\},$$

where $P[k,l] = (P_1[k,m], P_2[k,m], P_3[k,m])^\top$. These six vertices are ordered clockwise as shown in Fig.1. We shall refer to these six vertices, or in general the set of immediate neighboring vertices, as the 1-ring of $V[k]$.

- ☐ If $V[k]$ is an extraordinary vertex, which means its valence is not 6, we generate $\boldsymbol{P}_k$ with elements $P[k,l], 1 \leq l \leq 6$ as follows. Let $\widetilde{\boldsymbol{P}}_k = \{\widetilde{P}[k,j] : \ j = 1, 2, \cdots, J\}$ with $K \neq 6$ be the set of vertices on the 1-ring of $V[k]$. Define

$$\widehat{P}[k] = \frac{1}{K} \sum_{j=1}^{J} \widetilde{P}[k,j]. \tag{3.2}$$

  Then we set $\boldsymbol{P}_k = \{P[k,j] : \ 1 \leq j \leq 6\}$ with

$$P[k,j] = \widehat{P}[k], \ 1 \leq j \leq 6.$$

3. For each $i = 1, 2, 3$, the data-matrix $\boldsymbol{D}_i$ is a matrix in $\mathbb{R}^{7 \times N}$ defined by

$$\boldsymbol{D}_i = \begin{bmatrix} V_i[1] & \cdots & V_i[k] & \cdots & V_i[N] \\ P_i[1,1] & \cdots & P_i[k,1] & \cdots & P_i[N,1] \\ P_i[1,2] & \cdots & P_i[k,2] & \cdots & P_i[N,2] \\ P_i[1,3] & \cdots & P_i[k,3] & \cdots & P_i[N,3] \\ P_i[1,4] & \cdots & P_i[k,4] & \cdots & P_i[N,4] \\ P_i[1,5] & \cdots & P_i[k,5] & \cdots & P_i[N,5] \\ P_i[1,6] & \cdots & P_i[k,6] & \cdots & P_i[N,6] \end{bmatrix}.$$

Finally we let $\boldsymbol{D} = \{\boldsymbol{D}_1, \boldsymbol{D}_2, \boldsymbol{D}_3\}$.

Inversion of Data-Matrix Generation

Let $\boldsymbol{D} = \mathcal{C}(\boldsymbol{S})$, the left inversion operator, denoted as $\mathcal{C}^{-1}$, is simply defined as

$$\mathcal{C}^{-1}(\{\boldsymbol{D}, \boldsymbol{T}\}) = \Big\{ \big\{ \big(\boldsymbol{D}_1[1,k]\big)_{k=1}^N, \big(\boldsymbol{D}_2[1,k]\big)_{k=1}^N, \big(\boldsymbol{D}_3[1,k]\big)_{k=1}^N \big\}, \boldsymbol{T} \Big\}. \tag{3.3}$$

Note that $\mathcal{C}^{-1}(\{\boldsymbol{D}, \boldsymbol{T}\})$ simply extracts the first rows of $\boldsymbol{D}_i$ for each $i = 1, 2, 3$, which can also be written as

$$\mathcal{C}^{-1}(\{\boldsymbol{D}, \boldsymbol{T}\}) = \Big\{ \boldsymbol{\delta}_0 \boldsymbol{D}, \boldsymbol{T} \Big\} = \Big\{ \{\boldsymbol{\delta}_0 \boldsymbol{D}_1, \boldsymbol{\delta}_0 \boldsymbol{D}_2, \boldsymbol{\delta}_0 \boldsymbol{D}_3\}, \boldsymbol{T} \Big\}$$

where

$$\boldsymbol{\delta}_0 = [1, 0, \ldots, 0] \in \mathbb{R}^7.$$

It is obvious that $\mathcal{C}^{-1}(\mathcal{C}(\boldsymbol{S})) = \boldsymbol{S}$. For simplicity, we shall denote

$$\boldsymbol{D}_i[j, \cdot] := (\boldsymbol{D}_i[j,k])_{k=1}^N$$

as long as the dimension of $\boldsymbol{D}_i$ is clear.

Now, with the data-matrix $\boldsymbol{D}$ corresponding to a given surface $\boldsymbol{S}$, we can define the associated wavelet frame transform using the tight wavelet frame filters given in Example 2.2. In this paper, instead of (2.12), we use the convolution with masks as the frame transform for simplicity:

$$\boldsymbol{W}_{j,\boldsymbol{i}} \boldsymbol{u} := \boldsymbol{a}_{j,\boldsymbol{i}} \circledast \boldsymbol{u}. \tag{3.4}$$

One-Level Tight Wavelet Frame Transform of $\boldsymbol{S}$

Let $\boldsymbol{D}$ be the associated data-matrix of the given surface $\boldsymbol{S}$. According to the order of the set of neighboring vertices of a vertex as shown in Fig.1, we arrange the coefficients of each of masks $\boldsymbol{a}_i, 0 \leq i \leq 6$ given by Example 2.2 into a row vector, and put all vectors of the masks together to form the following 7×7 matrix:

$$M = \frac{1}{8}\begin{bmatrix} 2 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & -1 & -1 & 1 & -1 & -1 & 1 \\ 2 & 1 & -1 & -1 & 1 & -1 & -1 \\ 2 & -1 & 1 & -1 & -1 & 1 & -1 \\ 0 & -\frac{2\sqrt{3}}{3} & -\frac{2\sqrt{3}}{3} & -\frac{2\sqrt{3}}{3} & \frac{2\sqrt{3}}{3} & \frac{2\sqrt{3}}{3} & \frac{2\sqrt{3}}{3} \\ 0 & \frac{\sqrt{6}}{3} & \frac{\sqrt{6}}{3} & -\frac{2\sqrt{6}}{3} & -\frac{\sqrt{6}}{3} & -\frac{\sqrt{6}}{3} & \frac{2\sqrt{6}}{3} \\ 0 & \sqrt{2} & -\sqrt{2} & 0 & -\sqrt{2} & \sqrt{2} & 0 \end{bmatrix}. \tag{3.5}$$

We call M the *mask-matrix*. Observe that the order of the coefficients of each mask should match that for the neighborhood of a vertex $\boldsymbol{V}[k]$ which forms the column of the data-matrix. Then we define the following wavelet frame transform of a given surface $\boldsymbol{S}$ and $\boldsymbol{D} = \mathcal{C}(\boldsymbol{S})$:

$$\mathcal{W}(\boldsymbol{S}) = M\mathcal{C}(\boldsymbol{S}) = \{\{M\boldsymbol{D}_1, M\boldsymbol{D}_2, M\boldsymbol{D}_3\}, \boldsymbol{T}\} = \{\boldsymbol{\alpha}, \boldsymbol{T}\} \quad \text{Decomposition} \tag{3.6}$$

where $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3\}$ with $\boldsymbol{\alpha}_i = M\boldsymbol{D}_i$. Note that the reconstruction $\mathcal{W}^{-1}$ can be easily defined as

$$\mathcal{W}^{-1}\boldsymbol{\alpha} = \boldsymbol{\delta_0} M^{-1}\boldsymbol{\alpha}.$$

This is not the conventional tight frame inverse transform W^T. For surface processing, a simple reconstruction algorithm is desirable. We will elaborate this point in later sections.

The matrix $M\boldsymbol{D}_i \in \mathbb{R}^{7\times N}$, for each $i = 1, 2, 3$, has rows correspond to different wavelet frame bands. The first row corresponds to the low frequency band and the triangulated surface

$$\left\{\left((M\boldsymbol{D}_1)[1,\cdot], (M\boldsymbol{D}_2)[1,\cdot], (M\boldsymbol{D}_3)[1,\cdot]\right)^\top, \boldsymbol{T}\right\}$$

is a smooth approximation of the original surface $\boldsymbol{S}$; and

$$\{(M\boldsymbol{D}_1)[k,\cdot], (M\boldsymbol{D}_2)[k,\cdot], (M\boldsymbol{D}_3)[k,\cdot]\}, \quad \text{for } 2 \leq k \leq 7,$$

is the wavelet frame coefficients of $\boldsymbol{S}$ in band k.

Remark 3.1.

1. Since throughout the data-matrix generation process and the wavelet frame transformations, the triangulation $\boldsymbol{T}$ is not changed, we hence denote $\mathcal{C}$, $\mathcal{C}^{-1}$, $\mathcal{W}$ simply as

$$\mathcal{C}(\boldsymbol{S}) = \boldsymbol{D},\ \mathcal{C}^{-1}(\boldsymbol{D}) = \boldsymbol{S},\ \mathcal{W}(\boldsymbol{S}) = M\mathcal{C}(\boldsymbol{S}) = \{M\boldsymbol{D}_1, M\boldsymbol{D}_2, M\boldsymbol{D}_3\} = \boldsymbol{\alpha}.$$

   Therefore, throughout the rest of this paper, we shall drop $\boldsymbol{T}$ wherever the specific triangulation of a surface is irrelevant.

2. When implementing the wavelet frame transform $\mathcal{W}$, we do not need to compute the entire data-matrix $\boldsymbol{D}$. Each vertex, or in other words, each column of $\boldsymbol{D}$, can be handled independently. Therefore, the transform $\mathcal{W}$ can be computed in a paralleling fashion. $\square$

3.2 Multiple-Level Tight Wavelet Frame Transform

Same as the previous subsection, we consider the masks given by Example 2.2. Dilations of the masks required to define multiple-level tight wavelet frame transform are given by (2.13). To generalize such mask dilations to a triangulated surface, we need to introduce data-matrix generation associated to each of the dilation level.

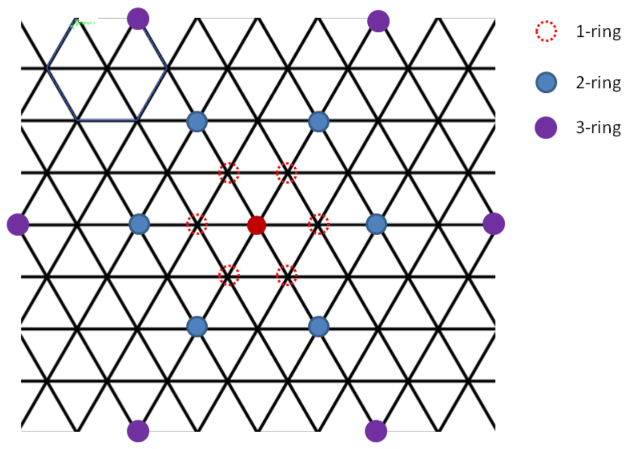

Figure 2: This figure shows examples of dilated neighboring vertices $\boldsymbol{P}_k^l$ of a given vertex (solid red dot) with dilation level $l = 0, 1, 2$.

Data-Matrix Generation with Dilations

The procedure is similar to that described in Section 3.1. Given a triangulated surface $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{T}\}$, we perform the following operations that produce a data-matrix $\boldsymbol{D}^l$ that corresponds to the dilation level l, where l is a positive integer. Such operation is denoted as $\mathcal{C}^l$, i.e. $\mathcal{C}^l(\boldsymbol{S}) = \boldsymbol{D}^l$. For convenience of notation, we let $\mathcal{C}^0 = \mathcal{C}$ and $\boldsymbol{D}^0 = \boldsymbol{D}$, where $\mathcal{C}$ is the data-matrix generation operation without dilation (given in Section 3.1) and $\mathcal{C}(\boldsymbol{S}) = \boldsymbol{D}$. For a vertex V on $\boldsymbol{S}$, the m-ring of V is the set of vertices on $\boldsymbol{S}$ which can be connected with V by m (minimum) edges. See Fig.2.

1. Let $\boldsymbol{V} = \{V[k] \mid k = 1, 2, \cdots N\}$, where
$$V[k] = (V_1[k], V_2[k], V_3[k])^\top$$
is the (x, y, z)-coordinates of the vertex $V[k]$. Let
$$\boldsymbol{P}_k^0 = \{P[k,1],\ P[k,2], \cdots,\ P[k,J]\},$$
where $P[k,j] = (P_1[k,m], P_2[k,m], P_3[k,m])^\top$, be the set of K immediate neighbors of $V[k]$ with valence K (i.e. 1-ring of $V[k]$).

2. For $l \geq 1$ and for each $k = 1, 2, \cdots, N$, we generate $\boldsymbol{P}_k^l$, which is the l-dilation of $\boldsymbol{P}_k^0$ living on the 2^l-ring of $V[k]$ (illustration is given in Fig.2). The point set $\boldsymbol{P}_k^l$ is obtained through the following iterative procedure:

(a) Let $\widetilde{\boldsymbol{P}}_k^j$, $0 \le j \le l-1$ be the set of K vertices neighboring $V[k]$ in the 2^j-ring that we have already found, with $\widetilde{\boldsymbol{P}}_k^0 = \boldsymbol{P}_k^0$.

(b) For each vertex $\widetilde{P}^j[k,m] \in \widetilde{\boldsymbol{P}}_k^j$, we find one of the vertices in 2^{j+1}-ring of $V[k]$, denoted by $\widetilde{P}^{j+1}[k,m]$, such that the angle between the vectors $\widetilde{P}^{j+1}[k,m] - \widetilde{P}^j[k,m]$ and $\widetilde{P}^j[k,m] - V[k]$ is the smallest.

(c) Let $\widetilde{\boldsymbol{P}}_k^{j+1}$ denote the set of $\widetilde{P}^{j+1}[k,m], m = 1, \cdots, K$ found in step (b).

(d) Repeat (a) to (c) until we obtain $\widetilde{\boldsymbol{P}}_k^l$.

(e) If $K = 6$, let $\boldsymbol{P}_k^l = \widetilde{\boldsymbol{P}}_k^l$. Otherwise, generate $\boldsymbol{P}_k^l$ from $\widetilde{\boldsymbol{P}}_k^l$ by (3.2) in Section 3.1.

3. For each $i = 1, 2, 3$, the data-matrix $\boldsymbol{D}_i^l$ is a matrix in $\mathbb{R}^{7 \times N}$ defined by

$$\boldsymbol{D}_i^l = \begin{bmatrix} V_i[1] & \cdots & V_i[k] & \cdots & V_i[N] \\ P_i^l[1,1] & \cdots & P_i^l[k,1] & \cdots & P_i^l[N,1] \\ P_i^l[1,2] & \cdots & P_i^l[k,2] & \cdots & P_i^l[N,2] \\ P_i^l[1,3] & \cdots & P_i^l[k,3] & \cdots & P_i^l[N,3] \\ P_i^l[1,4] & \cdots & P_i^l[k,4] & \cdots & P_i^l[N,4] \\ P_i^l[1,5] & \cdots & P_i^l[k,5] & \cdots & P_i^l[N,5] \\ P_i^l[1,6] & \cdots & P_i^l[k,6] & \cdots & P_i^l[N,6] \end{bmatrix}.$$

With

$$\boldsymbol{D}^l = \{\boldsymbol{D}_1^l, \boldsymbol{D}_2^l, \boldsymbol{D}_3^l\},$$

we finally define $\mathcal{C}^l(\boldsymbol{S}) = \boldsymbol{D}^l$. □

We note that, by construction, we have $\mathcal{C}^{-1}(\boldsymbol{D}^l) = \boldsymbol{S}$ for all $l = 0, 1, \cdots$, where $\mathcal{C}^{-1}$ is given in (3.3), the operation of extracting the first rows of $\boldsymbol{D}^j$.

Multiple-Level Tight Wavelet Frame Transform of $\boldsymbol{S}$

Let $\boldsymbol{D}^j$, $0 \le j \le l-1$ with $l \ge 1$, be the associated data-matrix of the given surface $\boldsymbol{S}$ at dilation level j. Let $M \in \mathbb{R}^{7 \times 7}$ be given by (3.5). Then we define the l-level tight wavelet frame decomposition of a given surface $\boldsymbol{S}$ as follows:

$$\mathcal{W}^l(\boldsymbol{S}) = \{W^j(\boldsymbol{S}) |\ 0 \le j \le l-1\} =: \{\boldsymbol{\alpha}^j, 0 \le j \le l-1\} =: \boldsymbol{\alpha}, \tag{3.7}$$

where

$$W^j(\boldsymbol{S}) = \begin{cases} M\mathcal{C}^j \circ \mathcal{L}^{j-1} \circ \cdots \circ \mathcal{L}^0(\boldsymbol{S}) & \text{for } j = l-1 \\ \mathcal{R}\left(M\mathcal{C}^j \circ \mathcal{L}^{j-1} \circ \cdots \circ \mathcal{L}^0(\boldsymbol{S})\right) & \text{for } 0 \le j \le l-2 \end{cases}$$

with $\mathcal{L}^j := \mathcal{C}^{-1} \circ (M\mathcal{C}^j)$ and the operator $\mathcal{R} : \mathbb{R}^{3 \times 7 \times N} \mapsto \mathbb{R}^{3 \times 6 \times N}$ defined by

$$\mathcal{R}(\boldsymbol{D}) := \left\{ \begin{bmatrix} \boldsymbol{D}_1[2,\cdot] \\ \boldsymbol{D}_1[3,\cdot] \\ \vdots \\ \boldsymbol{D}_1[7,\cdot] \end{bmatrix}, \begin{bmatrix} \boldsymbol{D}_2[2,\cdot] \\ \boldsymbol{D}_2[3,\cdot] \\ \vdots \\ \boldsymbol{D}_2[7,\cdot] \end{bmatrix}, \begin{bmatrix} \boldsymbol{D}_3[2,\cdot] \\ \boldsymbol{D}_3[3,\cdot] \\ \vdots \\ \boldsymbol{D}_3[7,\cdot] \end{bmatrix} \right\}.$$

Note that $\mathcal{R}(\boldsymbol{D})$ simply extracts the 2nd to 7th row of $\boldsymbol{D}_i$ for each $i = 1, 2, 3$. By definition of the transform (3.7), the wavelet frame coefficients $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}^j |\ 0 \leq j \leq l-1\}$ satisfy

$$\boldsymbol{\alpha}^j \in \begin{cases} \mathbb{R}^{3\times 7\times N} & \text{for } j = l-1 \\ \mathbb{R}^{3\times 6\times N} & \text{for } 0 \leq j \leq l-2 \end{cases},$$

where $\{\boldsymbol{\alpha}_1^{l-1}[1,\cdot], \boldsymbol{\alpha}_2^{l-1}[1,\cdot], \boldsymbol{\alpha}_3^{l-1}[1,\cdot]\}$ is the low frequency approximation of $\boldsymbol{S}$, while the rest are wavelet frame coefficients at different scales $0 \leq j \leq l-1$.

3.3 Tight Wavelet Frame Transforms with Generic Masks

Here, we discuss how we can generalize our wavelet frame transform for surfaces to masks other than those given in Example 2.2. In fact, the ideas and the actual transformations are rather similar to the case we considered above.

First let us consider the one-level frame transform. Same as before, we generate the mask-matrix first. Suppose $\boldsymbol{a} = \{\boldsymbol{a}_0, \boldsymbol{a}_1, \cdots, \boldsymbol{a}_r\}$ is a tight frame filter bank. Let $\mathrm{supp}(\boldsymbol{a})$ be its support defined by (3.1). By shifting the indices of the coefficients of all $\boldsymbol{a}_i$, we may assume that $[0,0] \in \mathrm{supp}(\boldsymbol{a})$ and $[0,0]$ is the "center" of $\mathrm{supp}(\boldsymbol{a})$. Next, we select an ordering for the set $\{[j,k] :\ [j,k] \in \mathrm{supp}(\boldsymbol{a})\}$ with $[0,0]$ being the first term in this order. With this ordering, write the coefficients $\{\boldsymbol{a}_i[j,k] :\ [j,k] \in \mathrm{supp}(\boldsymbol{a})\}$ as a row vector

$$A_i = [\boldsymbol{a}_i[0,0], \cdots].$$

Then we have the mask-matrix

$$M = \begin{bmatrix} A_0 \\ A_1 \\ \vdots \\ A_r \end{bmatrix}.$$

After we fix the order of the indices in $\mathrm{supp}(\boldsymbol{a})$, we select the vertices $P[k,m], 1 \leq m \leq \#\mathrm{supp}(\boldsymbol{a})$, the nearest neighbors of the vertex $V[k]$, based on such ordering. The selected $P[k,m]$ should match with the indices in $\mathrm{supp}(\boldsymbol{a})$. (When $V[k]$ is an extraordinary vertex, we then select the neighborhood of $V[k]$ as in Section 3.1.) Then we arrange each of the x, y, z components $P_i[j,m]$ of $P[j,m] = (P_1[j,m], P_2[j,m], P_3[j,m])^\top$ as a column vector to form the data-matrix $\boldsymbol{D}_i$. Then one-level frame transform can be written as the product of M and $\boldsymbol{D}_i$.

For the multi-level frame transform, it will be better to consider the indices $[j,k]$ in $[-s,s]\times[-s,s]$, where s is a positive integer such that

$$\mathrm{supp}(\boldsymbol{a}) \subseteq [-s,s]\times[-s,s].$$

Then we give an order of $[j,k] \in [-s,s]\times[-s,s]$ with $[0,0]$ being the first term to form the mask-matrix M. After that we define $\mathcal{C}^l$ as in Section 3.2. Here, we skip the details. Instead, we consider a specific case where the masks are 3×3 with possibly all 9 non-zero entries. Same as before, we need to first define the neighboring vertex set $\boldsymbol{P}_k^l$ for each vertex $V[k]$. Now the cardinality of the set $\boldsymbol{P}_k^l$ is eight instead of six. Then, the data-matrix generation operator $\mathcal{C}^l$ can be defined similarly as before. Illustration of the set $\mathcal{P}_k^l$ for $l=0$ and $l=1$, i.e. the 1-ring and 2-ring, are shown in Fig.3. The resulting data-matrix

$\boldsymbol{D}^l = \mathcal{C}^l(\boldsymbol{S})$ is in $\mathbb{R}^{9\times N}$ and takes the form

$$\boldsymbol{D}_i^l = \begin{bmatrix} V_i[1] & \cdots & V_i[k] & \cdots & V_i[N] \\ P_i^l[1,1] & \cdots & P_i^l[k,1] & \cdots & P_i^l[N,1] \\ P_i^l[1,2] & \cdots & P_i^l[k,2] & \cdots & P_i^l[N,2] \\ P_i^l[1,3] & \cdots & P_i^l[k,3] & \cdots & P_i^l[N,3] \\ P_i^l[1,4] & \cdots & P_i^l[k,4] & \cdots & P_i^l[N,4] \\ P_i^l[1,5] & \cdots & P_i^l[k,5] & \cdots & P_i^l[N,5] \\ P_i^l[1,6] & \cdots & P_i^l[k,6] & \cdots & P_i^l[N,6] \\ P_i^l[1,7] & \cdots & P_i^l[k,7] & \cdots & P_i^l[N,7] \\ P_i^l[1,8] & \cdots & P_i^l[k,8] & \cdots & P_i^l[N,8] \end{bmatrix}.$$

Finally, the corresponding multiple-level tight wavelet frame transforms take the exact same form as those given in Section 3.2.

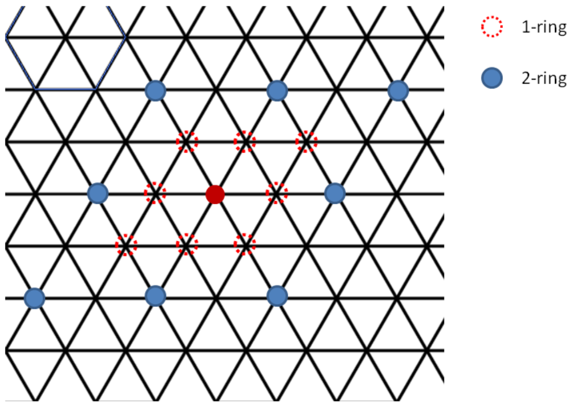

Figure 3: This figure shows examples of dilated neighboring vertices $\boldsymbol{P}_k^l$ of a given vertex (solid red dot) with dilation level $l = 0, 1$. The masks considered here are supported on $[-1,1]\times[-1,1]$.

Note that M does not necessary need to be a square matrix. The following is an example with canonical highpass filters considered in [67].

Example 3.1. *The scaling function is linear bivariate box spline B_{111} with directions $(1,0)^\top$, $(0,1)^\top$ and $(1,1)^\top$. The mask $\boldsymbol{a}_0$ for B_{111} and the first three highpass filters $\boldsymbol{a}_1$, $\boldsymbol{a}_2$, $\boldsymbol{a}_3$ are given in Example 2.2. The remaining 4 highpass filters are*

$$\boldsymbol{a}_4 = \frac{1}{8}\begin{bmatrix} 1 & -1 & \\ -1 & 0 & 1 \\ & 1 & -1 \end{bmatrix}, \quad \boldsymbol{a}_5 = \frac{1}{8}\begin{bmatrix} -1 & -1 & \\ 1 & 0 & -1 \\ & 1 & 1 \end{bmatrix},$$

$$\boldsymbol{a}_6 = \frac{1}{8}\begin{bmatrix} -1 & 1 & \\ -1 & 0 & 1 \\ & -1 & 1 \end{bmatrix}, \quad \boldsymbol{a}_7 = \frac{1}{8}\begin{bmatrix} 1 & 1 & \\ 1 & 0 & -1 \\ & -1 & -1 \end{bmatrix}.$$

With the order of $P[k,i], 1 \le i \le 6$ given in Fig.1, the corresponding mask-matrix, denoted by $\widetilde{M}$, is given by

$$\widetilde{M} = \frac{1}{8}\begin{bmatrix} 2 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & -1 & -1 & 1 & -1 & -1 & 1 \\ 2 & 1 & -1 & -1 & 1 & -1 & -1 \\ 2 & -1 & 1 & -1 & -1 & 1 & -1 \\ 0 & 1 & -1 & 1 & -1 & 1 & -1 \\ 0 & -1 & -1 & -1 & 1 & 1 & 1 \\ 0 & -1 & 1 & 1 & 1 & -1 & -1 \\ 0 & 1 & 1 & -1 & -1 & -1 & 1 \end{bmatrix}. \quad \square \tag{3.8}$$

In the next example, let us look at tight frame filters from [1]. The scaling function is the bivariate box spline with directions $(1,0)^\top$, $(0,1)^\top$, $(1,1)^\top$ and $(1,-1)^\top$.

Example 3.2. *The refinement mask of this box spline and the corresponding wavelet frame masks are*

$$\boldsymbol{a}_0 = \frac{1}{16}\begin{bmatrix} & 1 & 1 & \\ 1 & 2 & 2 & 1 \\ 1 & 2 & 2 & 1 \\ & 1 & 1 & \end{bmatrix}, \boldsymbol{a}_1 = \frac{1}{16}\begin{bmatrix} & 1 & -1 & \\ 1 & -2 & 2 & -1 \\ -1 & 2 & -2 & 1 \\ & -1 & 1 & \end{bmatrix},$$

$$\boldsymbol{a}_2 = \frac{1}{16}\begin{bmatrix} & -1 & -1 & \\ -1 & 2 & 2 & -1 \\ -1 & 2 & 2 & -1 \\ & -1 & -1 & \end{bmatrix}, \boldsymbol{a}_3 = \frac{1}{16}\begin{bmatrix} & -1 & 1 & \\ -1 & -2 & 2 & 1 \\ 1 & 2 & -2 & -1 \\ & 1 & -1 & \end{bmatrix},$$

$$\boldsymbol{a}_4 = \frac{1}{16}\begin{bmatrix} & -1 & -1 & \\ -1 & -2 & -2 & -1 \\ 1 & 2 & 2 & 1 \\ & 1 & 1 & \end{bmatrix}, \boldsymbol{a}_5 = \frac{1}{16}\begin{bmatrix} & -1 & 1 & \\ -1 & 2 & -2 & 1 \\ -1 & 2 & -2 & 1 \\ & -1 & 1 & \end{bmatrix},$$

$$\boldsymbol{a}_6 = \frac{1}{16}\begin{bmatrix} & 1 & 1 & \\ 1 & -2 & -2 & 1 \\ -1 & 2 & 2 & -1 \\ & -1 & -1 & \end{bmatrix}, \boldsymbol{a}_7 = \frac{1}{16}\begin{bmatrix} & 1 & -1 & \\ 1 & 2 & -2 & -1 \\ 1 & 2 & -2 & -1 \\ & 1 & -1 & \end{bmatrix},$$

$$\boldsymbol{a}_8 = \frac{\sqrt{2}}{16}\begin{bmatrix} & 1 & -1 & \\ -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \\ & -1 & 1 & \end{bmatrix}, \boldsymbol{a}_9 = \frac{\sqrt{2}}{16}\begin{bmatrix} & -1 & -1 & \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ & -1 & -1 & \end{bmatrix},$$

$$\boldsymbol{a}_{10} = \frac{\sqrt{2}}{16}\begin{bmatrix} & 1 & -1 & \\ -1 & 0 & 0 & 1 \\ -1 & 0 & 0 & 1 \\ & 1 & -1 & \end{bmatrix}, \boldsymbol{a}_{11} = \frac{\sqrt{2}}{16}\begin{bmatrix} & -1 & -1 & \\ 1 & 0 & 0 & 1 \\ -1 & 0 & 0 & -1 \\ & 1 & 1 & \end{bmatrix}.$$

Arranging the nonzero coefficients of each of $\boldsymbol{a}_i$ as a row vector with certain order, we have a 12 *by* 12 *matrix M:*

$$M = \frac{1}{16}\begin{bmatrix} 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & -2 & 2 & -2 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ & & & & & \vdots & & & & & & \\ 0 & 0 & 0 & 0 & -\sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & \sqrt{2} & -\sqrt{2} & \sqrt{2} & \sqrt{2} \end{bmatrix}. \square$$

4 Triangulated Surface Denoising: Model, Algorithm and Simulations

In this section, we propose the analysis based model for surface denoising using the tight wavelet frame transformation defined in Section 3.1 with masks given by Example 2.2.

4.1 Triangulated Surface Denoising: Model and Algorithm

Let $\widetilde{\boldsymbol{S}} = \{\widetilde{\boldsymbol{V}}, \boldsymbol{T}\}$ be the observed noisy surface. In our experiments, the noise are added to the original noise-free surface $\overline{\boldsymbol{S}} = \{\overline{\boldsymbol{V}}, \boldsymbol{T}\}$ by randomly perturbing each vertex in $\overline{\boldsymbol{V}}$ in the normal direction. The amount of perturbation satisfy a Gaussian distribution with 0 mean. The triangulation $\boldsymbol{T}$ is assumed to be unchanged, which is reasonable if the noise level is moderate.

Given the noisy surface $\widetilde{\boldsymbol{S}} = \{\widetilde{\boldsymbol{V}}, \boldsymbol{T}\}$, we propose the following analysis based model:

$$\min_{\boldsymbol{S}} \ \frac{1}{2}\left\|\boldsymbol{S} - \widetilde{\boldsymbol{S}}\right\|_2^2 + \lambda\|\mathcal{W}\boldsymbol{S}\|_1, \tag{4.1}$$

where $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{T}\}$,

$$\left\|\boldsymbol{S} - \widetilde{\boldsymbol{S}}\right\|_2^2 = \sum_{i=1}^{3}\sum_{j=1}^{N}\left|V_i[j] - \widetilde{V}_i[j]\right|^2$$

and

$$\|\mathcal{W}\boldsymbol{S}\|_1 = \|\boldsymbol{\alpha}\|_1 = \sum_{l=1}^{N}\sum_{i=1}^{3}\Big(\sum_{j=2}^{7}\left|\alpha_i[j,l]\right|^2\Big)^{\frac{1}{2}}. \tag{4.2}$$

Note that the minimization with respect to $\boldsymbol{S}$ in (4.1) is in fact only with respect to $\boldsymbol{V}$, since $\boldsymbol{T}$ is assumed to be unchanged. In addition, the addition/subtraction of two surfaces with the same triangulation, i.e. $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{T}\}$ and $\widetilde{\boldsymbol{S}} = \{\widetilde{\boldsymbol{V}}, \boldsymbol{T}\}$, is defined as

$$\boldsymbol{S} \pm \widetilde{\boldsymbol{S}} = \{\boldsymbol{V} \pm \widetilde{\boldsymbol{V}}, \boldsymbol{T}\}.$$

The analysis based model (4.1) can be solved using the following split Bregman algorithm [49, 51]:

$$\begin{cases} \boldsymbol{S}^{(k+1)} = \arg\min\limits_{\boldsymbol{S}} \frac{1}{2}\left\|\boldsymbol{S} - \widetilde{\boldsymbol{S}} + \boldsymbol{c}^{(k)}\right\|_2^2 + \frac{\mu}{2}\|\mathcal{W}\boldsymbol{S} - \boldsymbol{d}^{(k)} + \boldsymbol{b}^{(k)}\|_2^2, \\ \boldsymbol{d}^{(k+1)} = \arg\min\limits_{\boldsymbol{d}} \lambda\|\boldsymbol{d}\|_1 + \frac{\mu}{2}\|\boldsymbol{d} - \mathcal{W}\boldsymbol{S}^{(k+1)} - \boldsymbol{b}^{(k)}\|_2^2, \\ \boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} + \mathcal{W}\boldsymbol{S}^{(k+1)} - \boldsymbol{d}^{(k+1)}, \\ \boldsymbol{c}^{(k+1)} = \boldsymbol{c}^{(k)} + \delta\big(\boldsymbol{S}^{(k+1)} - \widetilde{\boldsymbol{S}}\big). \end{cases} \tag{4.3}$$

Here, the variables $\boldsymbol{b}, \boldsymbol{c}$ and $\boldsymbol{d}$ have the same structure as $\boldsymbol{\alpha} = \mathcal{W}\boldsymbol{S}$, and the addition and subtraction of them are defined as $\boldsymbol{\alpha} \pm \boldsymbol{d} = \{\boldsymbol{\alpha}_1 \pm \boldsymbol{d}_1, \boldsymbol{\alpha}_2 \pm \boldsymbol{d}_2, \boldsymbol{\alpha}_3 \pm \boldsymbol{d}_3\}$ (the addition and subtraction operations of $\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}$ and $\boldsymbol{\alpha}$ are defined in the same way). Note that the ℓ_2-norm of $\boldsymbol{\alpha}$ is defined as

$$\|\boldsymbol{\alpha}\|_2^2 = \sum_{i=1}^{3}\sum_{j=1}^{7}\sum_{l=1}^{N} \left|\alpha_i[j,l]\right|^2.$$

The ℓ_2-norms of $\boldsymbol{b}$ and $\boldsymbol{d}$ are defined similarly.

The second optimization problem of (4.3) has a closed form solution which is defined by the so-called isotropic shrinkage of wavelet frame coefficients [50]:

$$\boldsymbol{d}^{(k+1)} = \boldsymbol{\mathcal{T}}_{\lambda/\mu}\left(\mathcal{W}\boldsymbol{S}^{(k+1)} + \boldsymbol{b}^{(k)}\right), \tag{4.4}$$

where

$$\boldsymbol{\mathcal{T}}_{\nu}(\boldsymbol{\beta}) = \left(\mathcal{T}_\nu(\boldsymbol{\beta}_1), \mathcal{T}_\nu(\boldsymbol{\beta}_2), \mathcal{T}_\nu(\boldsymbol{\beta}_3)\right)^\top,$$

and $\mathcal{T}_\nu(\boldsymbol{\beta}_i)$ is defined as

$$\mathcal{T}_\nu(\boldsymbol{\beta}_i)[j,l] = \frac{\boldsymbol{\beta}_i[j,l]}{r_l^i}\max\left(r_l^i - \nu, 0\right) \tag{4.5}$$

with $r_l^i = \sqrt{\sum_{j=2}^{7}\left|\beta_i[j,l]\right|^2}$.

When $\mathcal{W}$ is linear, the first optimization of (4.3) has the solution (see [66])

$$\boldsymbol{S}^{(k+1)} = \left(I + \mu\mathcal{W}^T\mathcal{W}\right)^{-1}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathcal{W}^T(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right). \tag{4.6}$$

When the frame is tight, we have $\mathcal{W}^T\mathcal{W} = I$. However, when $\mathcal{W}$ is applied to a surface which has extraordinary vertices in general , $\mathcal{W}^T$ often does not have a closed form. However, for surface denoising, it is desirable that $\mathcal{W}^T$ has a simple closed form.

Here, we will replace $\mathcal{W}^T$ by an operator $\mathcal{V}$ which has a simple expression and satisfies

$$\mathcal{V}\mathcal{W} = I.$$

Suppose $\mathbf{v}$ is a row vector satisfying

$$\mathbf{v}M = \boldsymbol{\delta}_0 = [1, 0, \cdots, 0].$$

For M given by (3.5), M is nonsingular, and $\mathbf{v}$ is the first row of M^{-1}, which is

$$\mathbf{v} = [1, 1, 1, 1, 0, 0, 0]. \tag{4.7}$$

Recall that $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3\}$ are the frame coefficients defined by (3.6) after the frame decomposition. We may define $\mathcal{V}$ as

$$\mathcal{V}(\boldsymbol{\alpha}) = \mathbf{v}\boldsymbol{\alpha} = \{\mathbf{v}\boldsymbol{\alpha}_1, \mathbf{v}\boldsymbol{\alpha}_2, \mathbf{v}\boldsymbol{\alpha}_3\}. \tag{4.8}$$

Obviously, we have $\mathcal{V}\mathcal{W}\boldsymbol{S} = \boldsymbol{S}$.

Next, we use the following formula to approximate the solution of the first sub optimization problem of (4.3):

$$\begin{aligned}\boldsymbol{S}^{(k+1)} &= \frac{1}{1+\mu}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathcal{V}(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right)\\ &= \frac{1}{1+\mu}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathbf{v}(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right).\end{aligned} \tag{4.9}$$

Putting (4.9) and (4.4) together, we have the following algorithm for surface denoising

$$\begin{cases} \boldsymbol{S}^{(k+1)} = \frac{1}{1+\mu}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathbf{v}(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right), \\ \boldsymbol{d}^{(k+1)} = \boldsymbol{\mathcal{T}}_{\lambda/\mu}\left(\mathcal{W}\boldsymbol{S}^{(k+1)} + \boldsymbol{b}^{(k)}\right), \\ \boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} + \mathcal{W}\boldsymbol{S}^{(k+1)} - \boldsymbol{d}^{(k+1)}, \\ \boldsymbol{c}^{(k+1)} = \boldsymbol{c}^{(k)} + \delta\big(\boldsymbol{S}^{(k+1)} - \widetilde{\boldsymbol{S}}\big) \end{cases} \qquad \textbf{(Algorithm-Tri-1)} \tag{4.10}$$

where $\mathbf{v}$ is given by (4.7).

Remark 4.1. Observe that the last three components of $\mathbf{v}$ in (4.7) are zero, and hence, $\boldsymbol{a}_4, \boldsymbol{a}_5, \boldsymbol{a}_6$ do not play any role in the algorithm (4.10). Thus to save the computation cost, one may replace $\mathcal{W}$ in (4.10) by $\mathcal{W}_1$ which is defined by

$$\mathcal{W}_1(\boldsymbol{S}) = M_1\mathcal{C}(\boldsymbol{S}) = \{M_1\boldsymbol{D}_1, M_1\boldsymbol{D}_2, M_1\boldsymbol{D}_3\}, \tag{4.11}$$

where M_1 is the submatrix of M consisting of the first 4 rows of M, and replace $\mathbf{v}$ by $\mathbf{v}_1 = [1, 1, 1, 1]$. Thus the surface denoising algorithm (4.10) can be simplified as

$$\begin{cases} \boldsymbol{S}^{(k+1)} = \frac{1}{1+\mu}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathbf{v}_1(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right), \\ \boldsymbol{d}^{(k+1)} = \boldsymbol{\mathcal{T}}_{\lambda/\mu}\left(\mathcal{W}_1\boldsymbol{S}^{(k+1)} + \boldsymbol{b}^{(k)}\right), \\ \boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} + \mathcal{W}_1\boldsymbol{S}^{(k+1)} - \boldsymbol{d}^{(k+1)}, \\ \boldsymbol{c}^{(k+1)} = \boldsymbol{c}^{(k)} + \delta\big(\boldsymbol{S}^{(k+1)} - \widetilde{\boldsymbol{S}}\big). \end{cases} \tag{4.12}$$

where $\boldsymbol{b}^{(k)}$ and $\boldsymbol{d}^{(k)}$ have the same structure as $\mathcal{W}_1\boldsymbol{S}$. $\square$

4.2 Simulations

In this subsection, we show experimental results on the surface denoising with our wavelet frame based algorithm. As in [47], the Gaussian noise (with $\sigma = 1/5$ of the mean edge length) is added to the normal vectors. We also compare our method with the bilateral filtering [47] (with five iterations). From Fig.4, we see that our method has a better performance than the bilateral filtering in denoising the bunny. For the fandisk, our method preserves certain features better (see Fig.5), while the edges are slightly smeared out comparing to bilateral filtering. The smearing effect can be reduced by proper modification of the data-matrix generation process and the shrinkage operator. Next we propose another way of generating data-matrices which preserves edges better.

Alternative generation process of data-matrix

We can generate the data around an extraordinary in a different way:

□ If $V[k]$ is an extraordinary vertex, i.e. its valence is not 6, we generate $\boldsymbol{P}_k$ with elements $P[k, l], 1 \leq l \leq 6$ in an alternative way provided below. Let $\widetilde{\boldsymbol{P}}_k = \{\widetilde{P}[k, m] \mid m = 1, 2, \cdots, K\}$ with $K \neq 6$. If $K = 3$, we let $P[k, 4], P[k, 5], P[k, 6]$ be the middle points of the three edges with vertices $P[k, 1], P[k, 2], P[k, 3]$. When $K = 4$, let $P[k, 5], P[k, 6]$ be the middle points of the two longest edges among the fours edges $\overline{P[k, 1]P[k, 2]}, \overline{P[k, 2]P[k, 3]}, \overline{P[k, 3]P[k, 4]}, \overline{P[k, 4]P[k, 1]}$. For $K \geq 5$, we look at the angles of $\overrightarrow{V[k]P[k, l]}$ and $\overrightarrow{V[k]P[k, l']}$ and choose three pairs of $P[k, l]$ and $P[k, l']$ such that the angles are the largest with each vertex $P[k, l]$ selected only once (except for the case $K = 5$).

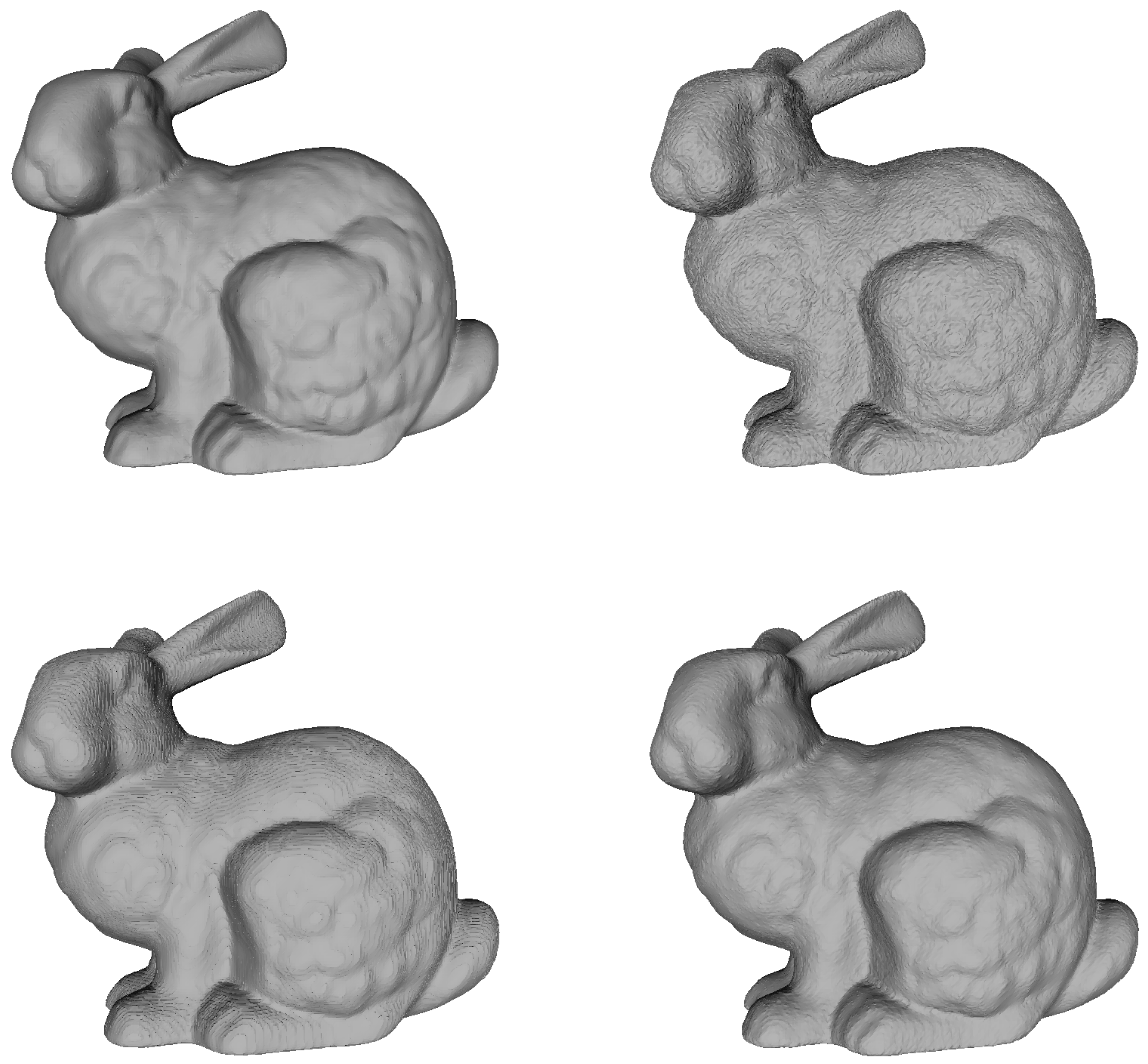

Figure 4: The bunny model (top-left) is artificially corrupted by Gaussian noise ($\sigma = 1/5$ of the mean edge length) (top-right), then smoothed by bilateral filtering (bottom-left) and Algorithm-Tri-1 (bottom-right).

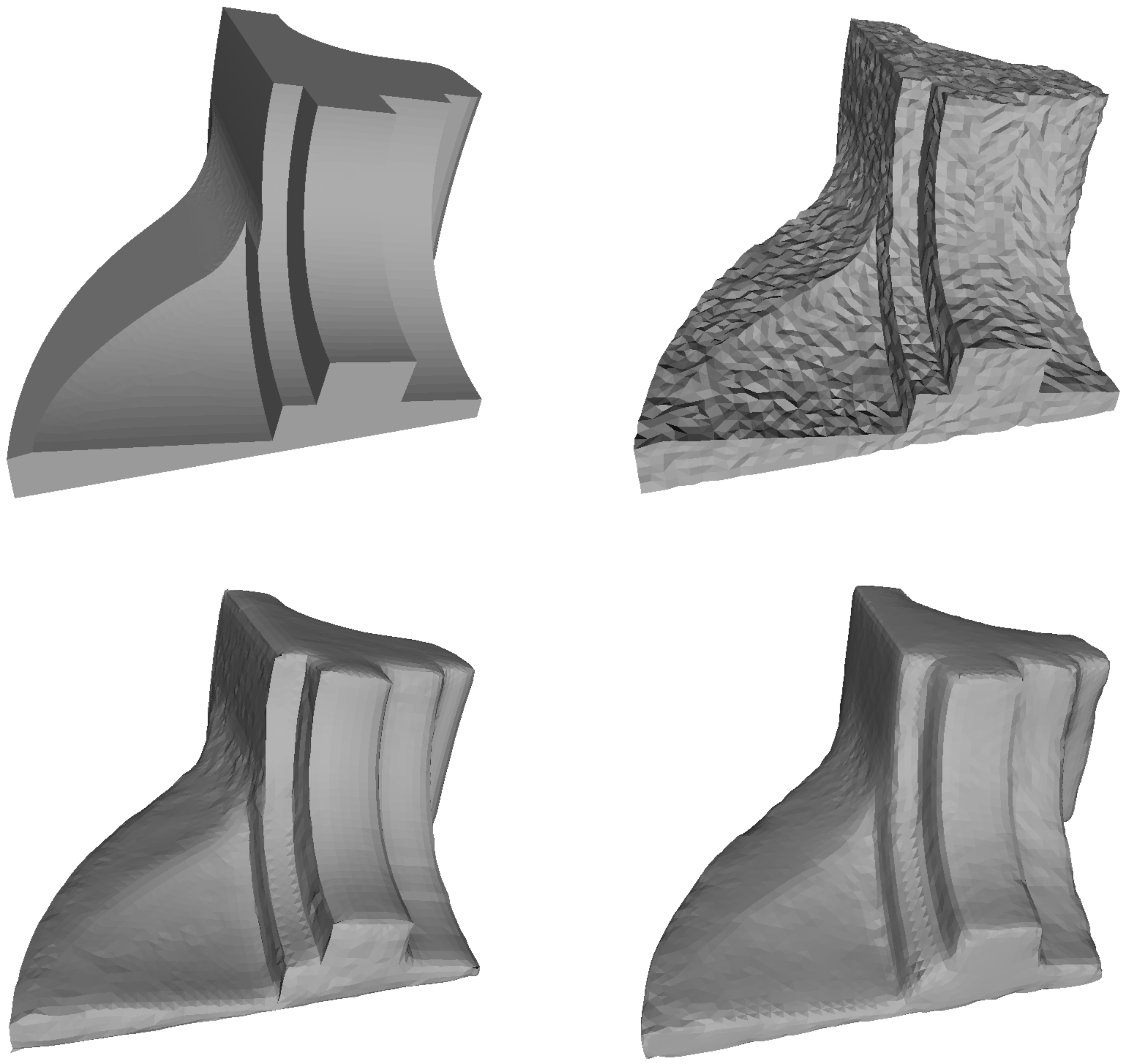

Figure 5: The fandisk model (top-left) is artificially corrupted by Gaussian noise ($\sigma = 1/5$ of the mean edge length) (top-right), then smoothed by bilateral filtering (bottom-left) and Algorithm-Tri-1 (bottom-right).

In this case when $\mathcal{W}$ is applied to a surface which has in general extraordinary vertices, $\mathcal{W}$ is nonlinear. If we use the shrinkage $\boldsymbol{\mathcal{T}}_\nu(\boldsymbol{\beta}) = (\mathcal{T}_\nu(\boldsymbol{\beta}_1), \mathcal{T}_\nu(\boldsymbol{\beta}_2), \mathcal{T}_\nu(\boldsymbol{\beta}_3))^\top$ defined by

$$\mathcal{T}_\nu(\boldsymbol{\beta}_i)[j,l] = \frac{\boldsymbol{\beta}_i[j,l]}{s_l^j} \max\left(s_l^j - \nu, 0\right)$$

with $s_l^j = \sqrt{\sum_{i=1}^3 \left|\beta_i[j,l]\right|^2}$ in (4.10), the denoising algorithm with the above new data-matrix generation preserves edge features better. We shall refer to this new algorithm as **Algorithm-Tri-2**. See Fig.6, where edge features are better preserved by our new method. Such shrinkage can be derived by the second optimization problem of (4.3) with a modified ℓ_1 norm of $\boldsymbol{\alpha}$ defined by

$$\|\mathcal{W}\boldsymbol{S}\|_1 = \|\boldsymbol{\alpha}\|_1 = \sum_{l=1}^N \sum_{j=2}^7 \Big(\sum_{i=1}^3 \left|\alpha_i[j,l]\right|^2\Big)^{\frac{1}{2}}.$$

5 Quad Surface Denoising: Representation, Model, Algorithm and Simulations

Quad surfaces or quad meshes have been widely used in many areas including CAD and animation movie production. Quad surfaces are preferred in some graphics applications because they can be aligned to two dominant local directions in a geometry (see [68]). Wavelet frame representation and the denoising model and algorithm for triangulated surfaces presented in Sections 3 and 4 can be easily modified to handle quad surfaces. In this section we consider quad surface tight wavelet frame representation and denoising.

5.1 Wavelet Frame Representation for Quad Surfaces

We focus on the quad surface processing by tensor-product tight frames of linear splines with the 2-dimensional tight frame filter bank $\boldsymbol{p}, \boldsymbol{q}_1, \cdots, \boldsymbol{q}_8$ given in (2.10) of Example 2.1.

Let $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{Q}\}$ be a given quad surface. According to the nonzero coefficients of the masks $\boldsymbol{p}, \boldsymbol{q}_1, \cdots, \boldsymbol{q}_8$, for each vertex $V[k] = (V_1[k], V_2[k], V_3[k])^\top$ in $\boldsymbol{S}$, if $V[k]$ is regular (meaning it has valence 4), we find the 8 (nearest) neighboring vertices of the vertex $V[k]$. The set of neighboring vertices is denoted by

$$\boldsymbol{P}_k = \{P[k,1],\ P[k,2],\ P[k,3],\ P[k,4],\ P[k,5],\ P[k,6],\ P[k,7],\ P[k,8]\},$$

where as in the previous sections, $P[k,l] = (P_1[k,l], P_2[k,l], P_3[k,l])^\top$. These eight vertices are selected and ordered in a way as illustrated in Fig.7.

- ☐ If $V[k]$ is an extraordinary vertex, that this its valence $K \neq 4$, we generate $\boldsymbol{P}_k$ with elements $P[k,l], 1 \leq l \leq 8$ as follows. Let $Q[k,m], m = 1, 2, \cdots, K$ denote the vertices which can be connected with $V[k]$ by one edge, and let $W[k,m],\ m = 1, 2, \cdots, K$ be the vertices which can be connected with $V[k]$ by a quad and face $V[k]$ on the quad, see Fig.8. Let $\widehat{\boldsymbol{v}}_1$ and $\widehat{\boldsymbol{v}}_2$ be the vertices defined by (3.2) based on $V[k], Q[k,m], m = 1, \cdots, K$ and based on $V[k], W[k,m], m = 1, \cdots, K$ respectively. Finally, we set $\boldsymbol{P}_k = \{P[k,l] :\ 1 \leq l \leq 8\}$ with

$$P[k, 2l-1] = \widehat{\boldsymbol{v}}_1,\ P[k, 2l] = \widehat{\boldsymbol{v}}_2,\ 1 \leq l \leq 4.$$

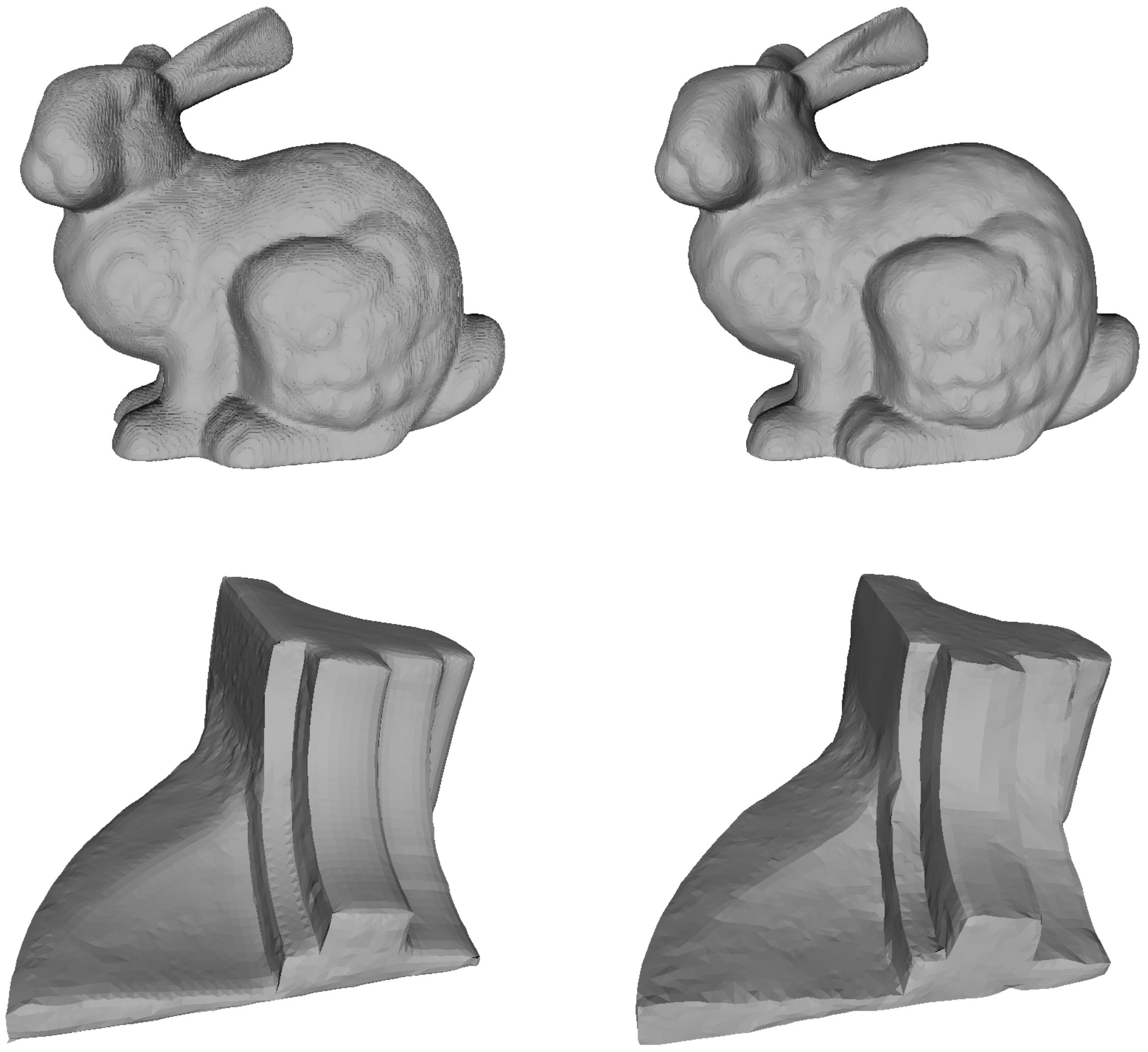

Figure 6: The bunny model is smoothed by bilateral filtering (top-left) and Algorithm-Tri-2 (top-right); the fandisk model is smoothed by bilateral filtering (bottom-left) and Algorithm-Tri-2 (bottom-right).

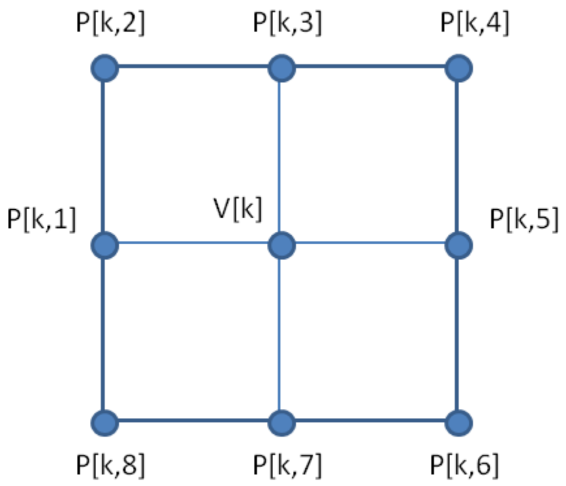

Figure 7: This figure illustrates how the neighboring vertices (forms set $\boldsymbol{P}_k$) of a given vertex $V[k] \in \boldsymbol{V}$ in a quad surface are ordered.

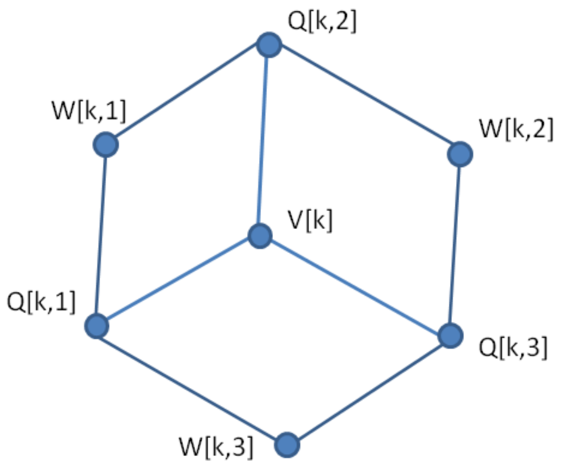

Figure 8: This figure illustrates how $Q[k,m], W[k,m], 1 \leq m \leq K$ $(K = 3)$ are defined.

The data-matrix $\boldsymbol{D}_i$ for each of he x, y and z coordinates is a matrix in $\mathbb{R}^{9\times N}$ defined by

$$\boldsymbol{D}_i = \begin{bmatrix} V_i[1] & \cdots & V_i[k] & \cdots & V_i[N] \\ P_i[1,1] & \cdots & P_i[k,1] & \cdots & P_i[N,1] \\ P_i[1,2] & \cdots & P_i[k,2] & \cdots & P_i[N,2] \\ P_i[1,3] & \cdots & P_i[k,3] & \cdots & P_i[N,3] \\ P_i[1,4] & \cdots & P_i[k,4] & \cdots & P_i[N,4] \\ P_i[1,5] & \cdots & P_i[k,5] & \cdots & P_i[N,5] \\ P_i[1,6] & \cdots & P_i[k,6] & \cdots & P_i[N,6] \\ P_i[1,7] & \cdots & P_i[k,7] & \cdots & P_i[N,7] \\ P_i[1,8] & \cdots & P_i[k,8] & \cdots & P_i[N,8] \end{bmatrix},$$

and we denote $\boldsymbol{D} = \{\boldsymbol{D}_1, \boldsymbol{D}_2, \boldsymbol{D}_3\}$ and $\mathcal{C}(\boldsymbol{S}) = \boldsymbol{D}$.

According to the order of the set of neighboring vertices as shown in Fig.7, we arrange the coefficients of each of masks $\boldsymbol{p}, \boldsymbol{q}_1, \cdots, \boldsymbol{q}_8$ into a row vector, and put the vectors of all masks together to form the following mask-matrix:

$$E = \frac{1}{16}\begin{bmatrix} 4 & 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1 \\ 0 & 2\sqrt{2} & \sqrt{2} & 0 & -\sqrt{2} & -2\sqrt{2} & -\sqrt{2} & 0 & \sqrt{2} \\ 0 & 0 & -\sqrt{2} & -2\sqrt{2} & -\sqrt{2} & 0 & \sqrt{2} & 2\sqrt{2} & \sqrt{2} \\ 4 & -2 & -1 & 2 & -1 & -2 & -1 & 2 & -1 \\ 0 & 0 & -2 & 0 & 2 & 0 & -2 & 0 & 2 \\ 4 & 2 & -1 & -2 & -1 & 2 & -1 & -2 & -1 \\ 0 & 0 & \sqrt{2} & -2\sqrt{2} & \sqrt{2} & 0 & -\sqrt{2} & 2\sqrt{2} & -\sqrt{2} \\ 0 & 2\sqrt{2} & -\sqrt{2} & 0 & \sqrt{2} & -2\sqrt{2} & \sqrt{2} & 0 & -\sqrt{2} \\ 4 & -2 & 1 & -2 & 1 & -2 & 1 & -2 & 1 \end{bmatrix}. \tag{5.1}$$

Then, the tight wavelet frame transform of a quad surface $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{Q}\}$ with filters in (2.10) is written as

$$\mathcal{W}(\boldsymbol{S}) = E\mathcal{C}(\boldsymbol{S}) = \Big\{\{E\boldsymbol{D}_1, E\boldsymbol{D}_2, E\boldsymbol{D}_3\}, \boldsymbol{Q}\Big\} = \big\{\boldsymbol{\alpha}, \boldsymbol{Q}\big\} \quad \text{Decomposition.} \tag{5.2}$$

We recall another tight frame filter bank from [69] with the same scaling function but having fewer highpass filters:

$$p = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \ q_1 = \frac{\sqrt{2}}{16}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \ q_2 = \frac{\sqrt{2}}{8}\begin{bmatrix} -1 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix},$$
$$q_3 = -\frac{1}{16}\begin{bmatrix} 1 & -2 & 1 \\ 2 & -4 & 2 \\ 1 & -2 & 1 \end{bmatrix}, \ q_4 = \frac{\sqrt{2}}{8}\begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}, \ q_5 = -\frac{1}{4}\begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \tag{5.3}$$

The corresponding mask-matrix is

$$E = \frac{1}{16}\begin{bmatrix} 4 & 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1 \\ 0 & 2\sqrt{2} & \sqrt{2} & 0 & -\sqrt{2} & -2\sqrt{2} & -\sqrt{2} & 0 & \sqrt{2} \\ 0 & 0 & -2\sqrt{2} & 0 & -2\sqrt{2} & 0 & 2\sqrt{2} & 0 & 2\sqrt{2} \\ 4 & -2 & -1 & 2 & -1 & -2 & -1 & 2 & -1 \\ 0 & 0 & -2\sqrt{2} & 0 & 2\sqrt{2} & 0 & -2\sqrt{2} & 0 & 2\sqrt{2} \\ 8 & 0 & 0 & -4 & 0 & 0 & 0 & -4 & 0 \end{bmatrix}. \tag{5.4}$$

Then, the associated tight wavelet frame transform $\mathcal{W}(\boldsymbol{S})$ for a given quad surface $\boldsymbol{S}$ is defined as in (5.2) with E defined in (5.4).

5.2 Quad Surface Denoising: Model and Algorithm

Given the noisy surface $\widetilde{\boldsymbol{S}} = \{\widetilde{\boldsymbol{V}}, \boldsymbol{Q}\}$, we use the following analysis based model based on the wavelet frame decomposition defined in (5.2):

$$\min_{\boldsymbol{S}} \ \frac{1}{2}\left\|\boldsymbol{S} - \widetilde{\boldsymbol{S}}\right\|_2^2 + \|\mathcal{W}\boldsymbol{S}\|_1, \tag{5.5}$$

where

$$\|\mathcal{W}\boldsymbol{S}\|_1 = \|\boldsymbol{\alpha}\|_1 = \sum_{i=1}^{3}\sum_{l=1}^{N}\sum_{k=2}^{9}\big|\alpha_i[k,l]\big|.$$

To solve (5.5), we use the split Bregman algorithm. If $\mathcal{W}$ is linear, then following the discussion in Section 4.1, we can reach the following algorithm

$$\begin{cases} \boldsymbol{S}^{(k+1)} = \left(1+\mu\mathcal{W}^T\mathcal{W}\right)^{-1}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathcal{W}^T(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right), \\ \boldsymbol{d}^{(k+1)} = \boldsymbol{\mathcal{T}}_{\lambda/\mu}\left(\mathcal{W}\boldsymbol{S}^{(k+1)} + \boldsymbol{b}^{(k)}\right), \\ \boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} + \mathcal{W}\boldsymbol{S}^{(k+1)} - \boldsymbol{d}^{(k+1)}, \\ \boldsymbol{c}^{(k+1)} = \boldsymbol{c}^{(k)} + \delta\big(\boldsymbol{S}^{(k+1)} - \widetilde{\boldsymbol{S}}\big). \end{cases} \tag{5.6}$$

Since $\mathcal{W}$ is not linear in general, and $\mathcal{W}^T$ does not have a closed form, we replace $\mathcal{W}^T$ by an operator $\mathcal{U}$ which has a simple expression and satisfies $\mathcal{U}\mathcal{W} = I$.

Suppose $\mathbf{u}$ is a row vector satisfying

$$\mathbf{u}E = [1, 0, \cdots, 0]. \tag{5.7}$$

For E given by (5.1), $\mathbf{u}$ is the first row of E^{-1}, which is

$$\mathbf{u} = [1, 0, 0, 1, 0, 1, 0, 0, 1]. \tag{5.8}$$

Let $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3\}$ be the frame coefficients defined by (5.2) after the frame decomposition. We may define $\mathcal{U}$ as

$$\mathcal{U}(\boldsymbol{\alpha}) = \Big\{ \{\mathbf{u}\boldsymbol{\alpha}_1, \mathbf{u}\boldsymbol{\alpha}_2, \mathbf{u}\boldsymbol{\alpha}_3\}, \boldsymbol{Q} \Big\}. \tag{5.9}$$

Then we have $\mathcal{U}\mathcal{W}\boldsymbol{S} = \boldsymbol{S}$. Thus, we have the following algorithm for quad surface denoising:

$$\begin{cases} \boldsymbol{S}^{(k+1)} = \frac{1}{1+\mu}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathbf{u}(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right), \\ \boldsymbol{d}^{(k+1)} = \boldsymbol{\mathcal{T}}_{\lambda/\mu}\left(\mathcal{W}\boldsymbol{S}^{(k+1)} + \boldsymbol{b}^{(k)}\right), \\ \boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} + \mathcal{W}\boldsymbol{S}^{(k+1)} - \boldsymbol{d}^{(k+1)}, \\ \boldsymbol{c}^{(k+1)} = \boldsymbol{c}^{(k)} + \delta\big(\boldsymbol{S}^{(k+1)} - \widetilde{\boldsymbol{S}}\big), \end{cases} \qquad (\textbf{Algorithm-Quad-1}) \tag{5.10}$$

where $\mathbf{u}$ is given by (5.8).

Following similar discussion in Remark 4.1, (5.10) can be reduced to

$$\begin{cases} \boldsymbol{S}^{(k+1)} = \frac{1}{1+\mu}\left(\widetilde{\boldsymbol{S}} - \boldsymbol{c}^{(k)} + \mu\mathbf{u}_1(\boldsymbol{d}^{(k)} - \boldsymbol{b}^{(k)})\right), \\ \boldsymbol{d}^{(k+1)} = \boldsymbol{\mathcal{T}}_{\lambda/\mu}\left(\mathcal{W}_1\boldsymbol{S}^{(k+1)} + \boldsymbol{b}^{(k)}\right), \\ \boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} + \mathcal{W}_1\boldsymbol{S}^{(k+1)} - \boldsymbol{d}^{(k+1)}, \\ \boldsymbol{c}^{(k+1)} = \boldsymbol{c}^{(k)} + \delta\big(\boldsymbol{S}^{(k+1)} - \widetilde{\boldsymbol{S}}\big), \end{cases} \tag{5.11}$$

with $\mathbf{u}_1 = [1,1,1,1]$, and

$$\mathcal{W}_1(\boldsymbol{S}) = E_1\mathcal{C}(\boldsymbol{S}) = \{\{E_1\boldsymbol{D}_1, E_1\boldsymbol{D}_2, E_1\boldsymbol{D}_3\}, \boldsymbol{Q}\}\,, \tag{5.12}$$

where E_1 is a submatrix of E:

$$E_1 = \frac{1}{16}\begin{bmatrix} 4 & 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1 \\ 4 & -2 & -1 & 2 & -1 & -2 & -1 & 2 & -1 \\ 4 & 2 & -1 & -2 & -1 & 2 & -1 & -2 & -1 \\ 4 & -2 & 1 & -2 & 1 & -2 & 1 & -2 & 1 \end{bmatrix}. \tag{5.13}$$

5.3 Quad Surface Denoising with Wavelet Frame Transforms of Generic Masks

As in Section 3.3, we can generalize our wavelet frame transforms for quad surfaces to masks other than those given in (2.10) and (5.3). Here we focus on the separable 2-dimensional tight frame filters of B-splines.

First let us recall univariate tight frames constructed in [62]. Let $m \in \mathbb{N}$. The tight frame filters with even-length are given by

$$\widehat{q}_n(\omega) = i^n\sqrt{\binom{2m}{n}}\sin^n\frac{\omega}{2}\cos^{2m-n}\frac{\omega}{2}, \quad 0 \le n \le 2m.$$

The scaling function corresponding to the lowpass filter q_0 is the $2m$-order B-spline supported on $[-m, m]$. When $m = 1$, q_0, q_1, q_2 are the filters in (2.9) (except for the $-$ sign for q_2).

The tight frame filters of odd-length are given by

$$\widehat{g}_n(\omega) = i^n\sqrt{\binom{2m-1}{n}}\sin^n\frac{\omega}{2}\cos^{2m-1-n}\frac{\omega}{2}e^{-i\frac{\omega}{2}}, \quad 0 \le n \le 2m-1.$$

The scaling function corresponding to the lowpass filter g_0 is the $(2m-1)$-order B-spline supported on $[1-m, m]$.

Taking tensor-products of $q_n, 0 \le n \le 2m$ and tensor-products of $g_n, 0 \le n \le 2m-1$, we have 2-dimensional tight frame filters:

$$\widehat{\boldsymbol{q}}_{jk}(\omega_1,\omega_2) = \widehat{q}_j(\omega_1)\widehat{q}_k(\omega_2), \quad 0 \le j,k \le 2m, \tag{5.14}$$

and

$$\widehat{\boldsymbol{g}}_{jk}(\omega_1,\omega_2) = \widehat{g}_j(\omega_1)\widehat{g}_k(\omega_2), \quad 0 \le j,k \le 2m-1. \tag{5.15}$$

To express the wavelet frame transform as the the product of a mask-matrix E and the data-matrix $\boldsymbol{D}$ of the surface, we first give an order of the coefficients of a filter. For the "even-length" filters $\boldsymbol{q}_{jk}$, their coefficients could be arranged as a row vector with an order given as follows. Starting with index $[0,0]$, we go over clockwise the first ring around $[0,0]$, then go over the second ring, and so on. For example, for $m = 2$, the coefficients $q[k_1,k_2], -2 \le k_1,k_2 \le 2$ of a filter $\boldsymbol{q}$ are arranged as the following row vector:

$$\big[\boldsymbol{q}[\boldsymbol{k}]\big]_{\{\mathbf{k}=[0,0],[-1,0],[-1,1],[0,1],[1,1],[1,0],[1,-1],[0,-1],[-1,-1],[-2,0],[-2,0],[-2,1],[-2,2],\atop [-1,2],[0,2],[1,2],[2,2],[2,1],[2,0],[2,-1],[2,-2],[1,-2],[0,-2],[-1,-2],[-2,-2],[-2,-1]\}}.$$

Then we have the matrix E from the from filters $\boldsymbol{q}_{jk}, 0 \le j,k \le 2m$:

$$E = \begin{bmatrix} \boldsymbol{q}_{0,0}[0,0] & \boldsymbol{q}_{0,0}[-1,0] & \boldsymbol{q}_{0,0}[-1,1] & \cdots & \boldsymbol{q}_{0,0}[-m,-1] \\ \boldsymbol{q}_{-1,0}[0,0] & \boldsymbol{q}_{-1,0}[-1,0] & \boldsymbol{q}_{-1,0}[-1,1] & \cdots & \boldsymbol{q}_{-1,0}[-m,-1] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{q}_{-m,-1}[0,0] & \boldsymbol{q}_{-m,-1}[-1,0] & \boldsymbol{q}_{-m,-1}[-1,1] & \cdots & \boldsymbol{q}_{-m,-1}[-m,-1] \end{bmatrix}. \tag{5.16}$$

For a vertex $V[k] = (V[k,1], V[k,2], V[k,3])^\top$ on a given quad surface $\boldsymbol{S}$, $(2m+1)^2-1$ vertices $P[k,l] = (P_1[k,l], P_2[k,l], P_3[k,l])^\top, 1 \le l \le (2m+1)^2-1$ on the m-ring neighborhood of $V[k]$ are selected and they are ordered with a way consistent with that for the filter coefficients. Then we have the data-matrix $\boldsymbol{D}_i \in \mathbb{R}^{(2m+1)^2 \times N}$ for each of the x, y, z components:

$$\boldsymbol{D}_i = \begin{bmatrix} V_i[1] & \cdots & V_i[k] & \cdots & V_i[N] \\ P_i[1,1] & \cdots & P_i[k,1] & \cdots & P_i[N,1] \\ P_i[1,2] & \cdots & P_i[k,2] & \cdots & P_i[N,2] \\ P_i[1,3] & \cdots & P_i[k,3] & \cdots & P_i[N,3] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ P_i[1,(2m+1)^2-1] & \cdots & P_i[k,(2m+1)^2-1] & \cdots & P_i[N,(2m+1)^2-1] \end{bmatrix}$$

for $i = 1,2,3$. Then the wavelet frame transform of a given quad surface $\boldsymbol{S} = \{\boldsymbol{V}, \boldsymbol{Q}\}$ with filters in (5.14) can be written in the same formulation (5.2) with E given by (5.16). Note that the wavelet frame transform with filters in (5.14) can be written in the same formulation as (5.2) with E given by (5.16).

Similarly, we can arrange coefficients of the "odd-length" filters $\boldsymbol{g}_{jk}$ in (5.15) as a row vector with a similar order to a matrix E, and arrange accordingly the vertices in a neighborhood of a vertex as a column vector to the data matrices $\boldsymbol{D}_i, i = 1,2,3$. Then, the wavelet frame transform of quad surface with filters in (5.15) can be written in the form of (5.2). The details are omitted here.

One can use algorithm (5.10) for surface denoising, where $\mathbf{u}$ is a row vector satisfying (5.7). Next we show that for tensor-product tight frame filters in (5.14) and (5.15), there is always a vector $\mathbf{u}$ such that (5.7) holds, which is equivalent to that there are constants c_{jk} such that

$$\sum_{j,k} c_{jk} \widehat{\boldsymbol{q}}_{jk}(\omega_1, \omega_2) = 1, \quad \omega_1, \omega_2 \in \mathbb{R}.$$

Proposition 1. *Let $\boldsymbol{q}_{jk}, 0 \le j,k \le 2m$ and $\boldsymbol{g}_{jk}, 0 \le j,k \le 2m-1$ be the tight frames given by* (5.14) *and* (5.15) *respectively. Then we have*

$$\sum_{k_1,k_2=0}^{m} (-1)^{k_1+k_2} \frac{\binom{m}{k_1}\binom{m}{k_2}}{\sqrt{\binom{2m}{2k_1}\binom{2m}{2k_2}}} \widehat{\boldsymbol{q}}_{2k_1,2k_2}(\omega_1,\omega_2) = 1, \quad \omega_1, \omega_2 \in \mathbb{R}; \tag{5.17}$$

and

$$\sum_{k_1,k_2=0}^{m-1} (-1)^{k_1+k_2} \binom{m-1}{k_1}\binom{m-1}{k_2} \left\{ \frac{\widehat{\boldsymbol{g}}_{2k_1,2k_2}(\omega_1,\omega_2)}{\sqrt{\binom{2m-1}{2k_1}\binom{2m-1}{2k_2}}} + \frac{\widehat{\boldsymbol{g}}_{2k_1+1,2k_2}(\omega_1,\omega_2)}{\sqrt{\binom{2m-1}{2k_1+1}\binom{2m-1}{2k_2}}} + \frac{\widehat{\boldsymbol{g}}_{2k_1,2k_2+1}(\omega_1,\omega_2)}{\sqrt{\binom{2m-1}{2k_1}\binom{2m-1}{2k_2+1}}} + \frac{\widehat{\boldsymbol{g}}_{2k_1+1,2k_2+1}(\omega_1,\omega_2)}{\sqrt{\binom{2m-1}{2k_1+1}\binom{2m-1}{2k_2+1}}} \right\} = 1, \quad \omega_1, \omega_2 \in \mathbb{R}. \tag{5.18}$$

Proof. It is straightforward to show that

$$\sum_{k=0}^{m}(-1)^k \frac{\binom{m}{k}}{\sqrt{\binom{2m}{2k}}}\widehat{q}_{2k}(\omega) = 1, \quad \omega \in \mathbb{R}.$$

Then expanding

$$\left(\sum_{k_1=0}^{m}(-1)^{k_1} \frac{\binom{m}{k_1}}{\sqrt{\binom{2m}{2k_1}}}\widehat{q}_{2k_1}(\omega_1)\right)\left(\sum_{k_2=0}^{m}(-1)^{k_2} \frac{\binom{m}{k_2}}{\sqrt{\binom{2m}{2k_2}}}\widehat{q}_{2k_2}(\omega)\right) = 1$$

leads to (5.17).

Similarly, (5.18) follows from

$$\sum_{k=0}^{m-1}(-1)^k \binom{m-1}{k}\left\{\frac{\widehat{g}_{2k}(\omega)}{\sqrt{\binom{2m}{2k}}} + \frac{\widehat{g}_{2k+1}(\omega)}{\sqrt{\binom{2m}{2k+1}}}\right\} = 1, \quad \omega \in \mathbb{R},$$

which can be proved straightforwardly. □

5.4 Simulations

In this subsection, we show one experimental result on quad surface denoising using our tight wavelet frame based algorithm. The top left of Fig.9 shows the quad mesh from [70]. The Gaussian noise (with $\sigma = 1/5$ of the mean edge length) is added to each vertex along the normal direction. We also compare our method with the bilateral filtering [47]. From Fig.9, we see that our method has a better performance.

Alternative generation process of data-matrix

As in Section 4.2, we consider generating the data around an extraordinary in a different way:

□ Suppose $V[k]$ is an extraordinary vertex, i.e. its valence is not 4. First we consider $K \geq 5$. Let $\boldsymbol{P}_k = \{\widetilde{P}[k,m] \mid m = 1, 2, \cdots, 2K\}$ be the vertices on the quads of which $V[k]$ is one vertex, see top-left picture of Fig.10. (a) We split each quad into two triangles with $V[k]$ be the vertex of both triangles, see top-middle picture of Fig.10. (b) Then, with this $2K$ triangles, as in Section 4.2, we select 4 pairs of $\widetilde{P}[k,m]$, see top-right picture of Fig.10. (c) After that, we have 8 triangles each having $V[k]$ as a vertex, see bottom-left picture of Fig.10. (d) Finally, we drop 4 edges to form 4 quads from these 8 triangles, see see bottom-right picture of Fig.10. There are two possible choices of dropping the 4 edges. We dropping these 4 edges whose sum is larger than the sum of other 4 edges.

When $K = 3$, let $\widetilde{P}[k,7], \widetilde{P}[k,8]$ be the middle points of the two longest edges among the six edges

$$\overline{\widetilde{P}[k,1]\widetilde{P}[k,2]}, \overline{\widetilde{P}[k,2]\widetilde{P}[k,3]}, \cdots, \overline{\widetilde{P}[k,5]P[k,6]}, \overline{\widetilde{P}[k,6]\widetilde{P}[k,1]}.$$

With these eight vertices $\widetilde{P}[k,1], \cdots, \widetilde{P}[k,8]$, we form 4 quads following Step (d) above.

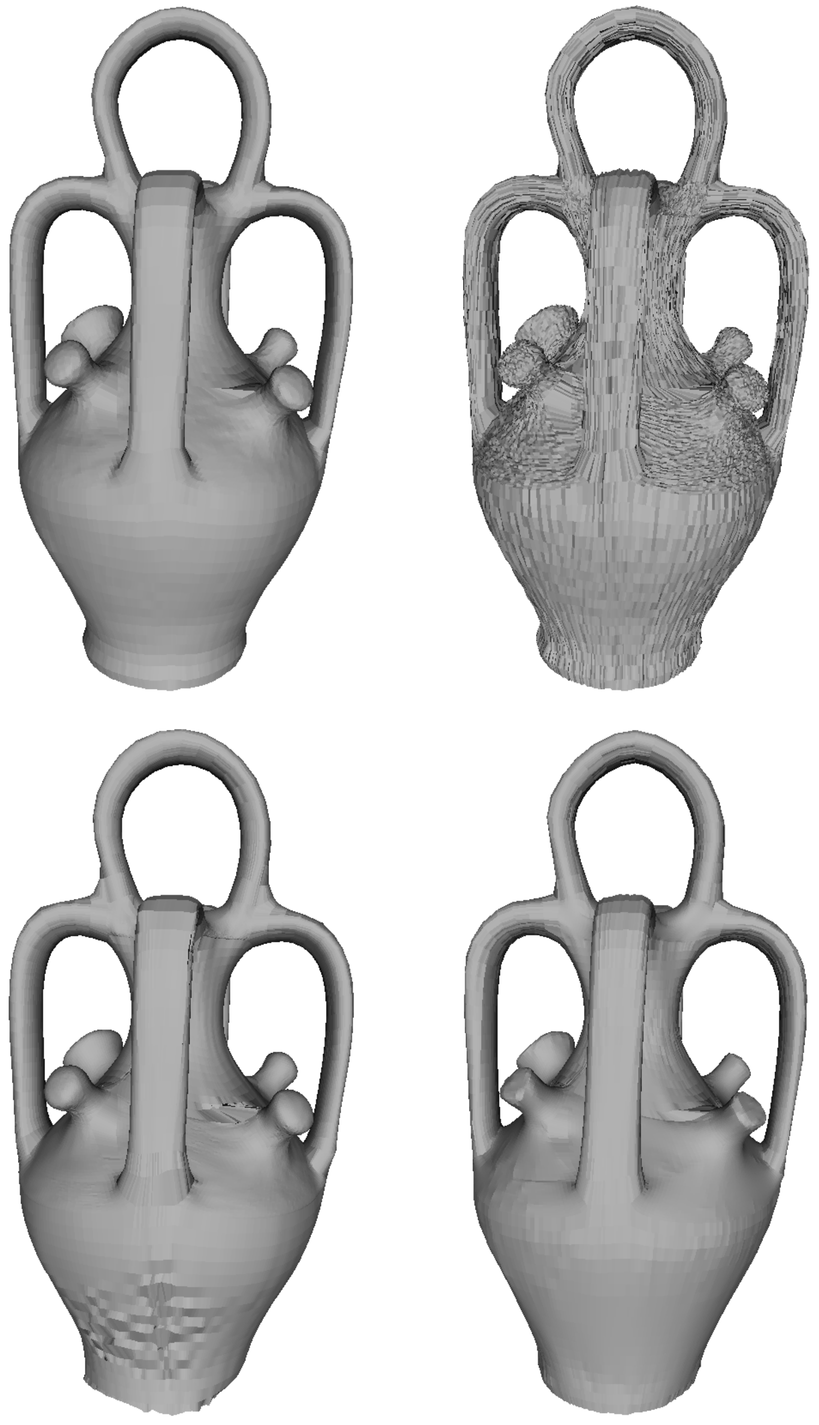

Figure 9: The botijo4 model (top-left) is artificially corrupted by Gaussian noise ($\sigma = 1/5$ of the mean edge length) (top-right), then smoothed by bilateral filtering (bottom-left) and Algorithm-Quad-1 (bottom-right).

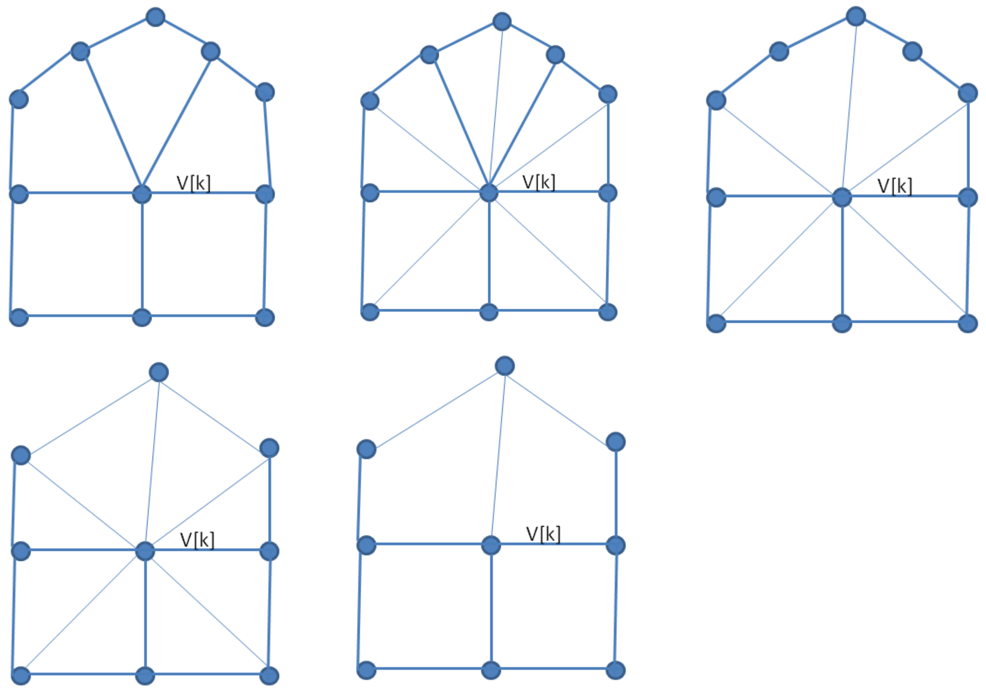

Figure 10: This figure shows how to generate data around an extraordinary vertex ($K = 5$).Top-left: 1-ring neighborhood of an extraordinary vertex; Top-middle: convert quad mesh to a triangular mesh; Top-right: selected 4 pairs of vertices; Bottom-left: selected 8 triangles; Bottom-right: 4 quads after 4 edges dropped.

Again in this case, when $\mathcal{W}$ is applied to a surface which has in general extraordinary vertices, $\mathcal{W}$ is nonlinear. For quad models, the denoising algorithm with the new data-matrix generation preserves edge features better, see Fig.11. Here, we use the isotropic shrinkage $\boldsymbol{\mathcal{T}}_\nu(\boldsymbol{\beta}) = (\mathcal{T}_\nu(\boldsymbol{\beta}_1), \mathcal{T}_\nu(\boldsymbol{\beta}_2), \mathcal{T}_\nu(\boldsymbol{\beta}_3))^\top$ defined as

$$\mathcal{T}_\nu(\boldsymbol{\beta}_i)[j,l] = \frac{\boldsymbol{\beta}_i[j,l]}{s_l^j} \max\left(s_l^j - \nu, 0\right)$$

with $s_l^j = \sqrt{\sum_{i=1}^3 \left|\beta_i[j,l]\right|^2}$ in (5.10). The ℓ_1 norm of $\boldsymbol{\alpha}$ corresponding to this isotropic shrinkage is

$$\|\mathcal{W}\boldsymbol{S}\|_1 = \|\boldsymbol{\alpha}\|_1 = \sum_{l=1}^{N} \sum_{j=2}^{9} \Big(\sum_{i=1}^{3} \left|\alpha_i[j,l]\right|^2 \Big)^{\frac{1}{2}}.$$

We refer to the algorithm that uses the above new data-matrix generation and the shrinkage operator as **Algorithm-Quad-2**.

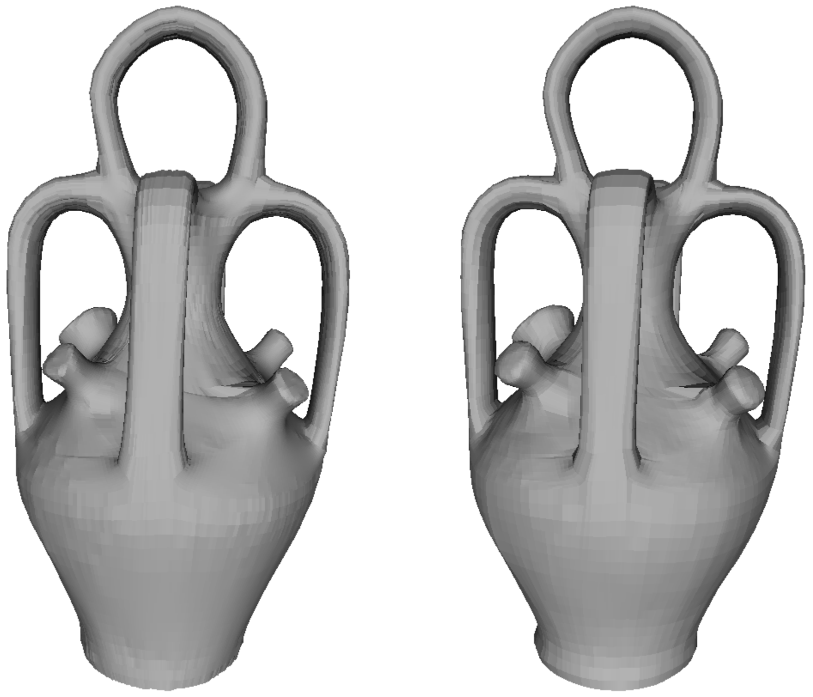

Figure 11: The botijo model smoothed by Algorithm-Quad-1 (left) and Algorithm-Quad-2 (right).

References

[1] Z.T. Fan, H. Ji, Z.W. Shen, "Dual Graminan analysis: duality and unitary extension principle", *Preprint*, 2013.

[2] P. Schröder, W. Sweldens, "Spherical wavelets: efficiently representing functions on the sphere", In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, ACM New York, NY, USA, 1995, pp. 161–172.

[3] W. Sweldens, “The lifting scheme: A construction of second generation wavelets”, *SIAM Journal on Mathematical Analysis*, 1998, vol. 29, pp. 511–546.

[4] D. Nain, S. Haker, A. Bobick, A.R. Tannenbaum, “Multiscale 3D shape analysis using spherical wavelets”, *Lecture Notes in Computer Science*, 2005, vol. 3750, pp. 459–467.

[5] D. Nain, S. Haker, A. Bobick, A.R. Tannenbaum, “Multiscale 3D shape representation and segmentation using spherical wavelets”, *IEEE Transactions on Medical Imaging*, 2007, vol. 26, pp. 598–618.

[6] B. Dong, Y. Mao, I. Dinov, Z. Tu, Y. Shi, Y. Wang, A. Toga, “Wavelet-based representation of biological shapes”, *Advances in Visual Computing*, 2009, pp. 955–964.

[7] X. Gu, Y. Wang, T.F. Chan, P.M. Thompson, S.T. Yau, “Genus zero surface conformal mapping and its application to brain surface mapping”, *IEEE Transactions on Medical Imaging*, 2004, vol. 23, pp. 949–958.

[8] E. Praun, H. Hoppe, “Spherical parametrization and remeshing”, *ACM Transactions on Graphics*, 2003, vol. 22, pp. 340–349.

[9] M. Bertram, “Biorthogonal Loop-subdivision wavelets”, *Computing*, 2004, vol. 72, pp. 29–39.

[10] M. Bertram, M.A. Duchaineau, B. Hamann, K.I. Joy, “Generalized B-spline subdivision-surface wavelets for geometry compression”, *IEEE Transacation on Visualization and Computer Graphics*, 2004, vol. 10, pp. 326–338.

[11] Q.T. Jiang, “Biorthogonal wavelets with 4-fold axial symmetry for quadrilateral surface multiresolution processing”, *Advances in Computational Mathematics*, 2011, vol. 34, pp. 127–165.

[12] Q.T. Jiang, “Biorthogonal wavelets with 6-fold axial symmetry for hexagonal data and triangle surface multiresolution processing”, *International Journal of Wavelets, Multiresolution and Information Processing*, 2011, vol. 9, pp. 773–812.

[13] H.W. Wang, K.H. Qin, K. Tang, “Efficient wavelet construction with Catmull–Clark subdivision”, *The Visual Computer*, 2006, vol. 22, pp. 874–884.

[14] H.W. Wang, K.H. Qin, H.Q. Sun, “$\sqrt{3}$-subdivision-based biorthogonal wavelets”, *IEEE Transactions on Visualization and Computer Graphics*, 2007, vol. 13, pp. 914–925.

[15] A. Khodakovsky, P. Schröder, W. Sweldens, “Progressive geometry compression”, In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000, pp. 271–278.

[16] B. Han, Z.W. Shen, “Wavelets from the Loop scheme”, *Journal of Fourier Analysis and Applications*, 2005, vol. 11, pp. 615–637.

[17] Q.T. Jiang, D. Pounds, “Highly symmetric bi-frames for triangle surface multiresolution processing”, *Applied and Computational Harmonic Analysis*, 2011, vol. 31, pp. 370–391.

[18] M. Crovella, E. Kolaczyk, “Graph wavelets for spatial traffic analysis”, In: INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications, IEEE Societies, 2003, vol. 3, pp. 1848–1857.

[19] M. Jansen, G.P. Nason, B.W. Silverman, "Multiscale methods for data on graphs and irregular multidimensional situations", *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2009, vol. 71, pp. 97–125.

[20] F. Murtagh, "The Haar wavelet transform of a dendrogram", *Journal of Classification*, 2007, vol. 24, pp. 3–32.

[21] A.B. Lee, B. Nadler, L. Wasserman, "Treelets: an adaptive multi-scale basis for sparse unordered data", *The Annals of Applied Statistics*, 2008, pp. 435–471.

[22] R.R. Coifman, M. Maggioni, "Diffusion wavelets", *Applied and Computational Harmonic Analysis*, 2006, vol. 21, pp. 53–94.

[23] M. Maggioni, H.N. Mhaskar, "Diffusion polynomial frames on metric measure spaces", *Applied and Computational Harmonic Analysis*, 2008, vol. 24, pp. 329–353.

[24] D. Geller, A. Mayeli, "Continuous wavelets on compact manifolds", *Mathematische Zeitschrift*, 2009, vol. 262, pp. 895–927.

[25] D.K. Hammond, P. Vandergheynst, R. Gribonval, "Wavelets on graphs via spectral graph theory", *Applied and Computational Harmonic Analysis*, 2011, vol. 30, pp. 129–150.

[26] M. Gavish, B. Nadler, R.R. Coifman, "Multiscale wavelets on trees graphs and high dimensional data: theory and applications to semi supervised learning", In: Proceedings of ICML, 2010.

[27] M. Gavish, R.R. Coifman, "Sampling, denoising and compression of matrices by coherent matrix organization", *Applied and Computational Harmonic Analysis*, 2012, vol. 33, pp. 354–369.

[28] C.K. Chui, F. Filbir, H.N. Mhaskar, "Representation of functions on big data: graphs and trees", *Applied and Computational Harmonic Analysis*, 2014, DOI: 10.1016/j.acha.2014.06.006.

[29] B. Dong, A. Chien, Z.W.Shen, S. Osher, "A new multiscale representation for shapes and its application to blood vessel recovery", *SIAM Journal on Scientific Computing*, 2010, vol. 32, pp. 1724–1739.

[30] L. Rudin, S. Osher, E. Fatemi, "Nonlinear total variation based noise removal algorithms", *Phys. D*, 1992, vol. 60, pp. 259–268.

[31] S. Osher, M. Burger, D. Goldfarb, J. Xu, W. Yin, "An iterative regularization method for total variation based image restoration", *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 2005, vol. 4, pp. 460–489.

[32] M. Burger, G. Gilboa, S. Osher, J. Xu, "Nonlinear inverse scale space methods", *Communications in Mathematical Sciences*, 2006, vol. 4, pp. 179–212.

[33] J. Xu, S. Osher, "Iterative regularization and nonlinear inverse scale space applied to wavelet-based denoising", *IEEE Transactions on Image Processing*, 2007, vol. 16, pp. 534–544.

[34] U. Clarenz, U. Diewald, M. Rumpf, "Anisotropic geometric diffusion in surface processing", In: Proceedings of the conference on Visualization'00, IEEE Computer Society Press Los Alamitos, CA, USA, 2000, pp. 397–405.

[35] M. Desbrun, M. Meyer, P. Schröder, A.H. Barr, “Anisotropic feature-preserving denoising of height fields and bivariate data”, In: Graphics Interface, 2000, vol. 11, pp. 145–152.

[36] M. Desbrun, M. Meyer, P. Schröder, A.H. Barr, “Implicit fairing of arbitrary meshes using diffusion and curvature flow”, In: Proceedings of SIGGRAPH 1999, 1999, pp. 317–324.

[37] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, W. Stuetzle, “Piecewise smooth surface reconstruction”, *Computer Graphics*, 1994, vol. 28, pp. 295–302.

[38] T. Tasdizen, R. Whitaker, P. Burchard, S. Osher, “Geometric surface processing via normal maps”, *ACM Transactions on Graphics*, 2003, vol. 22, pp. 1012–1033.

[39] T. Tasdizen, R. Whitaker, P. Burchard, S. Osher, “Geometric surface smoothing via anisotropic diffusion of normals”, In: Proceedings of the conference on Visualization’02, 2002, pp. 125–132.

[40] A. Buades, B. Coll, J.M. Morel, “On image denoising methods”, *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 2005, vol. 4, pp. 490–530.

[41] S. Yoshizawa, A. Belyaev, H.P. Seidel, “Smoothing by example: mesh denoising by averaging with similarity-based weights”, In: IEEE International Conference on Shape Modeling and Applications, SMI 2006, 2006, pp. 38–44.

[42] B. Dong, J. Ye, S. Osher, S., I. Dinov, “Level set based nonlocal surface restoration”, *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 2008, vol. 7, pp. 589–598.

[43] A. Elmoataz, O. Lezoray, S. Bougleux, “Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing”, *IEEE Transactions on Image Processing*, 2008, vol. 17, pp. 1047–1060.

[44] G. Gilboa, S. Osher, “Nonlocal operators with applications to image processing”, *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 2008, vol. 7, pp. 1005–1028.

[45] M. Elsey, S. Esedoglu, “Analogue of the total variation denoising model in the context of geometry processing”, *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 2009, vol. 7, pp. 1549–1573.

[46] I. Kakadiaris, L.X. Shen, M. Papadakis, I. Konstantinidis, D. Kouri, D. Hoffman, “Surface denoising using a tight frame”, In: Proceedings of the Computer Graphics International (CGI’04), pp. 553–560, 2004.

[47] S. Fleishman, I. Drori, D. Cohen-Or, “Bilateral mesh denoising”, In: Proceedings of AMC SIGGRAPH 2003, pp. 950–953.

[48] T. Jones, F. Durand, M. Desbrun, “Non-iterative, feature-preserving mesh smoothing”, In: Proceeding of ACM SIGGRAPH 2003, pp. 943–949.

[49] J.F. Cai, S. Osher, Z.W. Shen, “Split Bregman methods and frame based image restoration”, *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 2009, vol. 8, pp. 337–369.

[50] J.F. Cai, B. Dong, S. Osher, Z.W. Shen, “Image restorations: total variation, wavelet frames and beyond, *Journal of American Mathematical Society*, 2012, vol. 25(4), pp. 1033–1089.

[51] T. Goldstein, S. Osher, "The split Bregman algorithm for L_1 regularized problems", *SIAM Journal on Imaging Sciences*, 2009, vol. 2, pp. 323–343.

[52] R.H. Chan, T. Chan, L.X. Shen, Z.W. Shen, "Wavelet algorithms for high-resolution image reconstruction", *SIAM Journal on Scientific Computing*, 2003, vol. 24, 1408–1432.

[53] J.F. Cai, R.H. Chan, L.X. Shen, Z.W. Shen, "Convergence analysis of tight framelet approach for missing data recovery", *Advances in Computational Mathematics*, 2008, vol. 31, pp. 1–27.

[54] J.F. Cai, Z.W. Shen, "Framelet based deconvolution", *J. Comp. Math.*, 2010, vol. 28, pp. 289–308.

[55] J.F. Cai, R.H. Chan, Z.W. Shen, "Simultaneous cartoon and texture inpainting", *Inverse Problems and Imaging*, 2010, vol. 4, pp. 379–395.

[56] J.F. Cai, R.H. Chan, L.X. Shen, Z.W. Shen, "Restoration of chopped and nodded images by framelets", *SIAM J. Sci. Comput.*, 2008, vol. 24, pp. 1205–1227.

[57] I. Daubechies, G. Teschke, L. Vese, "Iteratively solving linear inverse problems under general convex constraints", *Inverse Problems and Imaging*, 2007, vol. 1, pp. 29–46.

[58] M. Fadili, J. Starck, "Sparse representations and Bayesian image inpainting", In: Proc. SPARS'05, Rennes, France, 2005, vol. 1.

[59] M. Fadili, J. Starck, F. Murtagh, "Inpainting and zooming using sparse representations", *The Computer Journal*, 2009, vol. 52, pp. 64–79.

[60] M. Figueiredo, R. Nowak, "An EM algorithm for wavelet-based image restoration", *IEEE Trans. Image Proc.*, 2003, vol. 12, pp. 906–916.

[61] M. Figueiredo, R. Nowak, "A bound optimization approach to wavelet-based image deconvolution", In: IEEE International Conference on Image Processing (ICIP 2005), Genoa, Italy, 2005, vol. 2, pp. 782–785.

[62] A. Ron, Z.W. Shen, "Affine Systems in $L_2(\mathbb{R}^d)$: the analysis of the analysis operator", *Journal of Functional Analysis*, 1997, vol. 148, pp. 408–447.

[63] I. Daubechies, "Ten Lectures on Wavelets", Society for Industrial and Applied Mathematics, CBMS-NSF Lecture Notes, SIAM, 1992, vol. 61.

[64] I. Daubechies, B. Han, A. Ron, Z.W. Shen, "Framelets: MRA-based constructions of wavelet frames", *Applied and Computational Harmonic Analysis*, 2003, vol. 14, pp. 1–46.

[65] Z.W. Shen, "Wavelet frames and image restorations", In: Proceedings of the International Congress of Mathematicians, 2010, vol. 4, pp. 2834–2863.

[66] B. Dong, Z.W. Shen, "MRA-Based Wavelet Frames and Applications", *IAS Lecture Notes Series, Summer Program on "The Mathematics of Image Processing", Park City Mathematics Institute*, 2010.

[67] Q.T. Jiang, "Hexagonal tight frame filter banks with idealized high-pass filters", *Advances in Computational Mathematics*, 2009, vol. 31, pp. 215–236.

[68] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. Silva, M. Tarini, D. Zorin, "Quad-mesh generation and processing: a survey", *Computer Graphics Forum*, 2013, vol. 32, pp. 51–76.

[69] B. Dong, Q.T. Jiang, Z.W. Shen, "Image restoration: wavelet frame shrinkage, nonlinear evolution PDEs, and beyond", *Preprint*, 2013.

[70] D. Bommes, M. Campen, H.-C. Ebke, P. Alliez, L. Kobbelt, "Integer-grid maps for reliable quad meshing", In: Proceedings of SIGGRAPH 2013, 2013, pp. 98:1?-98:12.