Mathematical Foundations of Data Sciences

Gabriel Peyré
CNRS & DMA
École Normale Supérieure
gabriel.peyre@ens.fr
www.gpeyre.com
www.numerical-tours.com

September 24, 2017

Chapter 5

Multiresolution Mesh Processing

This chapter shows how computations on a mesh can be performed in a multiscale manner, by considering meshes of increasing resolutions. This leads to the notion of subdivision surfaces and wavelet transform, which are two different tools to interpolates and decompose functions on meshes. Both methods rely on a special kind of meshes whose triangulations can be obtained by applying a regular refinement rule.

5.1 Semi-regular Meshes

5.1.1 Nested Multiscale Grids.

In order to perform multiscale mesh processing, one needs to pack the vertices V of a topological mesh $M = (V, E, F)$ in sets of increasing resolution. As explained in section 3.1.2, it is important to remember that this construction is purely combinatorial, in that no geometrical information (such as actual positions of the vertices in $\mathbb{R}^3$) is required to build the set of multi-resolution meshes. In fact these multiscale grids can be used to actually process the geometrical realization $\mathcal{M}$ of the mesh M as three real valued functions (the three coordinates of the points).

We thus consider a set of nested indexes

$$V_0 \subset V_{-1} \subset \ldots \subset V_L = V$$

which are split according to

$$V_j = V_{j+1} \cup H_{j+1}.$$

Next section describes how to actually compute this set of nested grids using a triangular split, but most of the mathematical tools are in fact valid for arbitrary set of indices, as long as they are embedded in one each other through scales.

For mesh processing, an index $\ell \in V_j$ corresponds to a vertex $x_\ell \in \mathcal{V} \subset \mathbb{R}^3$. The signals to be processed are vectors $f \in \mathbb{R}^n$ of size $n = |V_L|$ defined on the grid V_L. We sometimes write $f \in \ell^2(V_L)$ instead of $f \in \mathbb{R}^n$ to emphasis the domain on which f is indexed. This chapter describes transforms for signals $f \in \ell^2(V_L)$ sampled on the finest grid V_L.

5.1.2 Semi-regular Triangulation.

The combinatorial structure of a triangular mesh is defined in section 3.1.2. This chapter considers only a certain class of meshes $M = (V, E, F)$ that can be obtained by a regular split of faces, starting from an initial coarse triangulation. This splitting leads to a set of multiresolution meshes $M_j = (V_j, E_j, F_j)$ for $J \leqslant j \leqslant 0$, where the full mesh is $M_J = M$.

Starting from this coarse triangulation, one defines by subdivision a multiscale triangulation $(V_j, E_j, F_j)_{L \leqslant j \leqslant 0}$ where

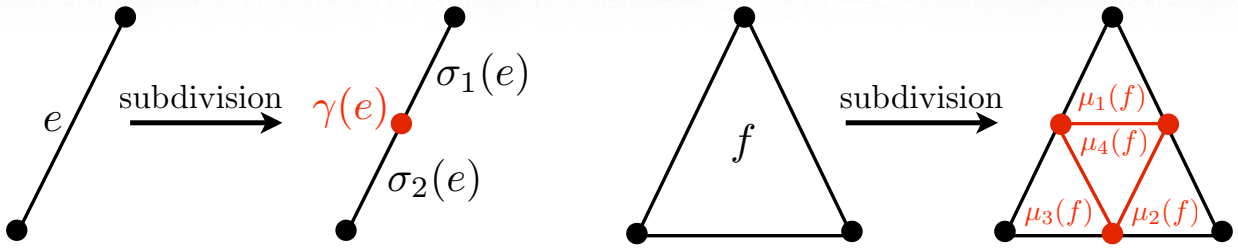

Figure 5.1: Edge-splitting subdivision.

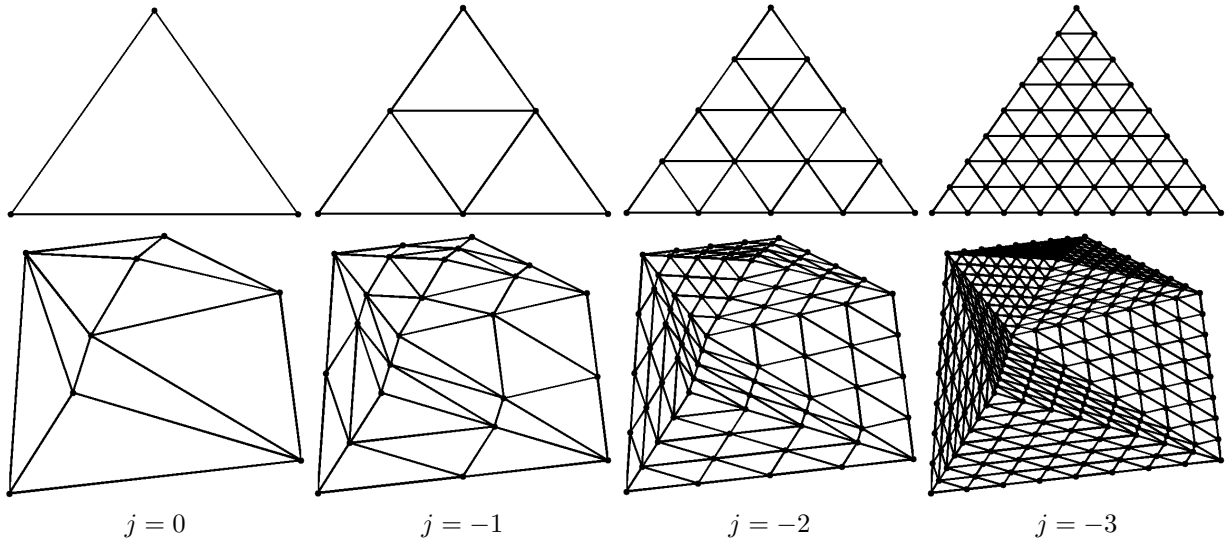

Figure 5.2: Regular subdivision 1:4 of a single triangle. Regular subdivision of a planar triangulation M_0.

– For each edge $e \in E_j$, a central index $\gamma(e) \in V_{j-1}$ is added to the vertices

$$V_{j-1} = V_j \cup \{\gamma(e)\ ;\ e \in E_j\}\,.$$

– Each edge is subdivided into two finer edges

$$\forall\, e = (a,b) \in E_j, \quad \sigma_1(e) = (a, \gamma(e)) \quad \text{and} \quad \sigma_2(e) = (b, \gamma(e)).$$

The subdivided set of edges is then

$$E_{j-1} = \{\sigma_i(e)\ ;\ i = 1,2 \quad \text{and} \quad e \in E_j\}\,.$$

– Each face $f = (a,b,c) \in F_j$ is subdivided into four faces

$$\left\{ \begin{array}{l} \mu_1(f) = (a, \gamma(a,b), \gamma(a,c)),\ \mu_2(f) = (b, \gamma(b,a), \gamma(b,c)), \\ \mu_3(f) = (c, \gamma(c,a), \gamma(c,b)),\ \mu_4(f) = (\gamma(a,b), \gamma(b,c), \gamma(c,a)). \end{array} \right.$$

The subdivided set of faces is then

$$F_{j-1} = \{\mu_i(f)\ ;\ i = 1,2,3,4 \quad \text{and} \quad f \in F_j\}\,.$$

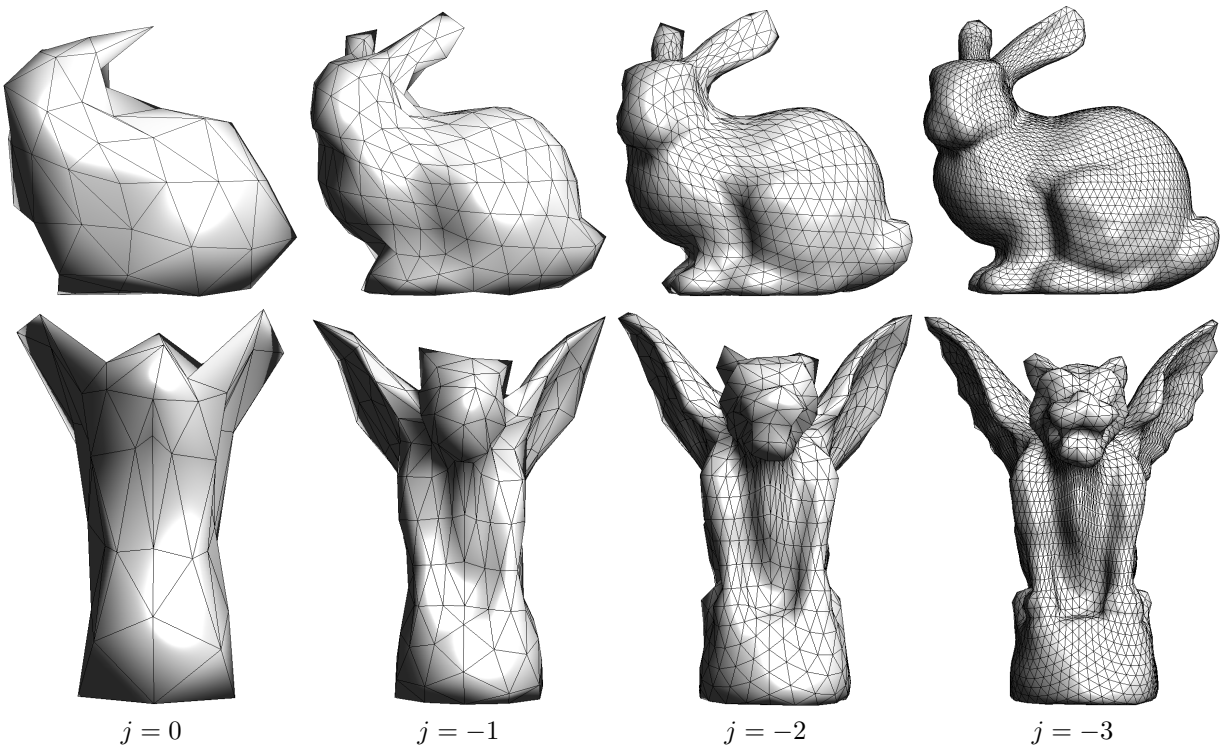

Figure 5.3: Examples of semi-regular meshes $(V_j)_j$ for increasing scale j (from left to right).

Figure 5.1 shows the notations related to the subdivision process. Figure 5.2 shows an example of recursive splitting of a triangle and a coarse triangulation. Figure 5.3 shows examples of semi-regular triangulation using a geometric realization (position of the vertices) to create a 3D surface.

The set of vertices can be classified as

- **Regular vertices** are those who belong neither to the coarse mesh V_0 nor to a boundary of a mesh M_j. These vertices have always 6 neighbors.
- **Extraordinary vertices** are the initial vertices of V_0. They exhibit arbitrary connectivity.
- **Boundary vertices** are those belonging to a mesh boundary. Boundary vertices not in V_0 always have 4 immediate neighbors.

Obviously not every meshes can be obtained from such a subdivision process. In practice, an arbitrary mesh, obtained from CAD design or range scanning usually does not have any multiscale structure. It is thus necessary to remesh it in order to modify the connectivity of the mesh. During this process, the position of the vertices in $\mathbb{R}^3$ is modified in order for the geometrical realization to stay close from the original piecewise linear surface. One can see [2] fur a survey of various semi-regular remeshing methods.

5.1.3 Spherical Geometry Images

Starting from some input surfaces $\mathcal{S} \subset \mathbb{R}^3$, one typically wants to compute a semi-regular meshes $(M_j)_{j \geqslant L}$ that approximate $\mathcal{S}$. In most case, the surface $\mathcal{S}$ is actually given as an arbitrary triangulated mesh and this process corresponds to a semi-regular remeshing. Many algorithm have been devised for surface remeshing and we describe here a method [26] that works for surfaces that have the topology of a sphere. It means that the surface has genus 0, without boundary and without handles.

This methods works by computing several intermediate surface-wise parameterization.

- *Spherical parameterization:* each points of the original triangulation of $\mathcal{S}$ is mapped onto the unit sphere.

This create a bijective parameterization

$$\varphi_S : S^2 \to \mathcal{S}.$$

This is a non-linear process that differ from the planar parameterization introduced in section 3.3.7. We do not give the details of such a process, but it requires minimizing the smoothness of the mapping φ_S^{-1} under the constraint that it maps points of $\mathcal{S}$ to unit length vectors (point on the sphere S^2). The algorithm is explained in details in [26].

– *Spherical-tetraedron flattening:* one flatten each quadrant (1/8) of the sphere in order to have a mapping

$$\varphi_T : \text{Octaedron} \to S^2.$$

One can use for instance a mapping between spherical barycentric coordinate on each quadrant and Euclidean barycentric coordinates on each face of the octahedron.

– *Tetraedron unfolding:* One maps each equilateral face of the octaedron on a rectangular triangle that corresponds to 1/8th of the square $[0,1]^2$

$$\varphi_U : [0,1]^2 \to \text{Octaedron}.$$

– *Regular sampling:* the geometry image is obtained by regularly sampling the square on a uniform grid

$$x_\ell = \varphi_S \circ \varphi_T \circ \varphi_U(\ell/n) \quad \text{for} \quad \ell_i = 0, \ldots, n-1.$$

The mapping $\ell \mapsto x_\ell \in \mathbb{R}^3$ is the geometry image, which can be stored as a 3-channel (color) image.

From such a geometry image x_ℓ, one can easily compute a semi-regular mesh by simply performing a regular 1:4 subdivision of the octaedron. Figure 5.4 shows the steps of the construction of a geometry image, and the resulting semi-regular mesh.

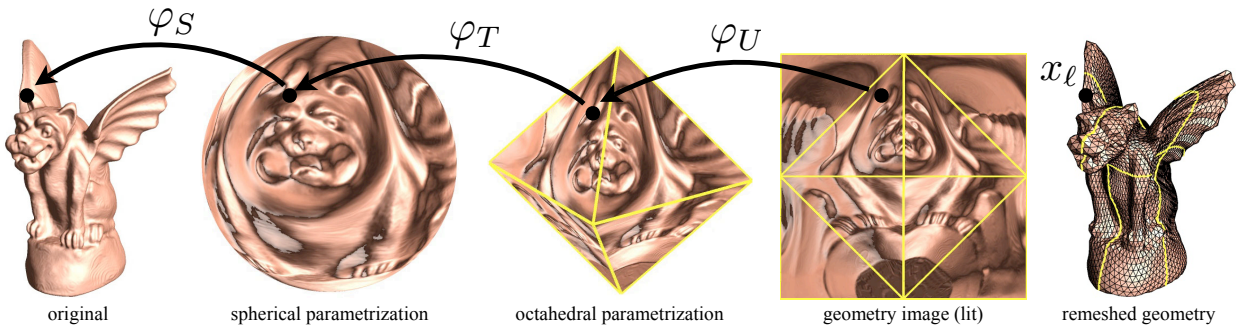

Figure 5.4: Spherical geometry image construction, taken from [26].

5.2 Subdivision Curves

Before getting into the detail of subdivision surfaces, we describe the subdivision process in the simpler setting of 1D signals. This leads to the construction of subdivision of 1D functions and subdivision curves.

In this 1D setting, the grid point indexes are dyadic sub-grids of $\mathbb{Z}$

$$\forall\, j \geqslant L, \quad V_j = \left\{ \ell 2^{j-L} \;;\; 0 \leqslant \ell < s_0 2^{-j} \right\},$$

where $s_0 = |V_0|$ is the size of the initial vector f_0 to be subdivided.

Each subdivision steps computes, from a set $f_j(\ell) \in \ell^2(V_j)$ of coarse values, a refined vector $f_{j-1} \in \ell^2(V_{j-1})$ defined by

$$\begin{cases} \forall\, k \in H_j,\ f_{j-1}(k) = \sum_t f_j((k-1)/2 + t) h(t), \\ \forall\, \ell \in V_j,\ f_{j-1}(\ell) = \sum_t f_j(\ell + t) \tilde{h}(t). \end{cases}$$

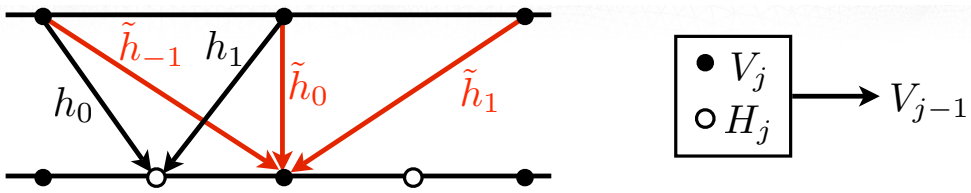

Figure 5.5: 1D subdivision scheme with filters h and $\tilde{h}$. The red curve represent the original signal f^0.

where the set of weights h and $\tilde{h}$ acts as local averaging operators. This averaging should be corrected at the boundary, and we use here cyclic boundary conditions which identifies 0 and $s_0 2^{-j}$ in V_j. Figure 5.5 shows a graphical display of these averaging operators.

One can write this subdivision steps as convolution by introducing the global set of weights

$$g = [\ldots, \tilde{h}(-1), h(0), \tilde{h}(0), h(1), \tilde{h}(1), \ldots]$$

since one has

$$f_{j-1} = (f_j \uparrow 2) * g \quad \text{where} \quad a \uparrow 2 = [\ldots, 0, a(-1), 0, a(0), 0, a(1), 0, \ldots].$$

This corresponds to the traditional description of the wavelet low-pass filtering [20].

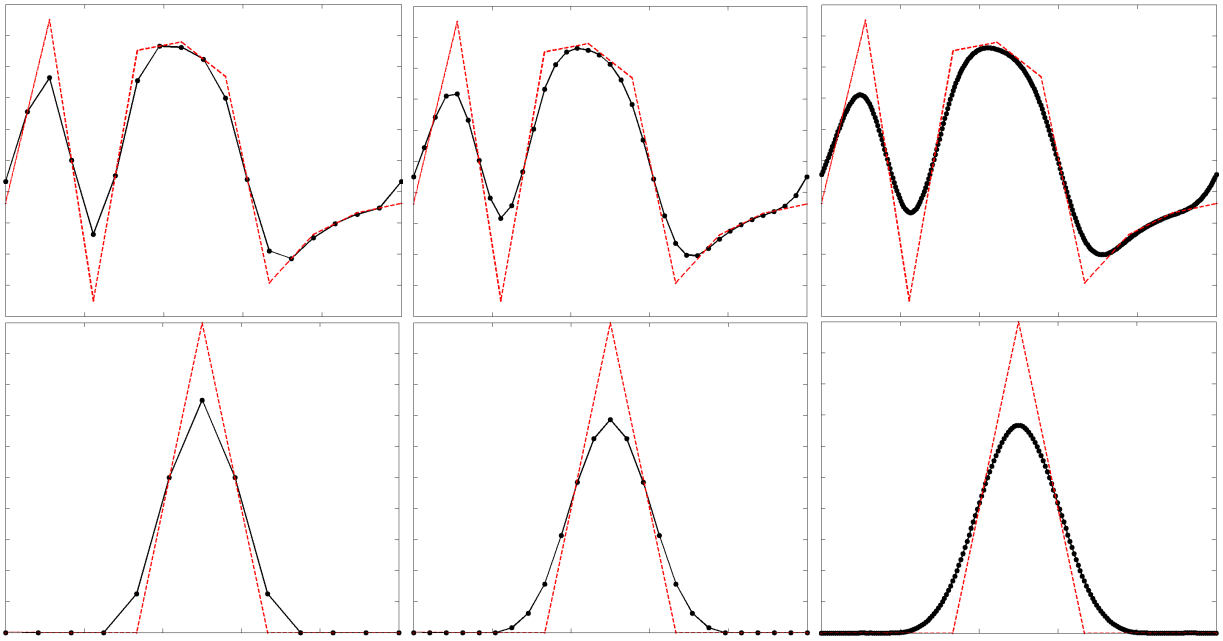

Figure 5.6: 1D subdivision of a signal. Bottom row shows the subdivision from an impulse signal, converging to the scaling function φ.

Figure 5.6 shows several steps of subdivision, starting from an initial vector of size $|V_0| = 10$.

One can apply this subdivision of functions to a pair of signals

$$(X_0, Y_0) : V_0 \to \mathbb{R}^2$$

which is a control polygon composed of points located in the plane. The subdivision curve converges to the limiting curve

$$(X_j, Y_j) \overset{j \to -\infty}{\longrightarrow} (X(t), Y(t))_{t=0}^{1} \subset \mathbb{R}^2.$$

An interesting property is that this curve is included in the convex hull of the control polygon

$$(X(t), Y(t))_t \subset \mathrm{Conv}(X_0, Y_0).$$

Figure 5.7 shows examples of subdivision curves.

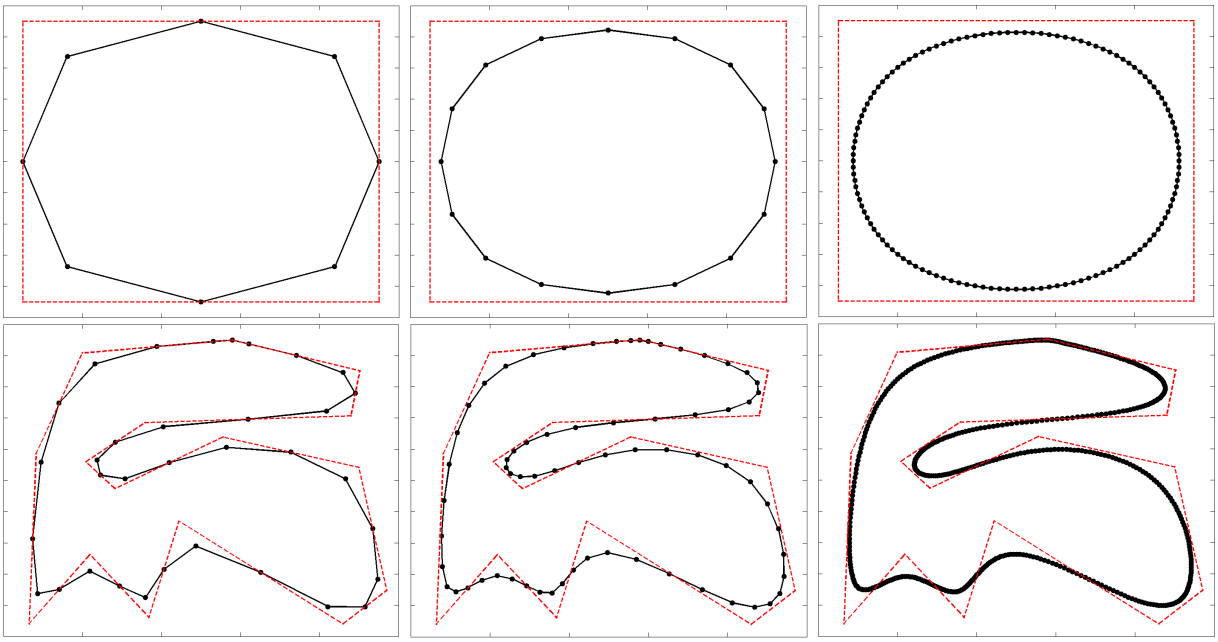

Figure 5.7: Two examples of subdivision curves. The red curve represent the original curve (X^0, Y^0).

5.3 Subdivision Surfaces

Subdivision schemes allows to compute a set of progressively refined vectors on a semi-regular mesh. More precisely, from an initial vector $f_0 \in \mathbb{R}^{|V_0|}$ defined on the coarse mesh M_0, local interpolation kernels computes iteratively vectors $f_j \in \mathbb{R}^{|V_j|}$ of finer resolution. When applied to 3 function $(f_0^i)_{i=1,2,3}$ defining the geometrical position of points in $\mathbb{R}^3$, this hierarchical construction defines a subdivision surface. These subdivisions surfaces are used extensively in computer aided geometry and computer graphics. One can see [9] for a survey of subdivision surfaces and their applications.

5.3.1 Interpolation Operators

In order to refine a vector $f_j \in \mathbb{R}^{|V_j|}$ defined on the vertex V_j of the mesh M_j, one uses two interpolators

$$P_j : \ell^2(V_j) \longrightarrow \ell^2(H_j) \qquad \text{and} \qquad \tilde{P}_j : \ell^2(V_j) \longrightarrow \ell^2(V_j). \tag{5.1}$$

A new refined function $f_{j-1} \in \mathbb{R}^{|V_{j-1}|}$ defined on the vertices $V_{j-1} = V_j \cup H_j$ of M_{j-1} is defined by applying these two refinement operators:

$$\forall\, \ell \in V_{j-1}, \quad f_{j-1}(\ell) = \begin{cases} (P_j f_j)(\ell) & \text{if} \quad \ell \in V_j, \\ (\tilde{P}_j f_j)(\ell) & \text{if} \quad \ell \in H_j. \end{cases}$$

Since $V_j \subset V_{j-1}$, the operator $\tilde{P}_j$ only modify slightly the value at vertex in V_j. On the other hand, the operator P_j creates new value at the vertices of H_j that are inserted between V_j and V_{j-1}.

In practical applications, these interpolating operators are local, meaning that the value of $(P_j f_j)(\ell)$ and $(\tilde{P}_j f_j)(\ell)$ depends only on values $f_j(\ell')$ for $\ell' \in V_j$ being close to $\ell \in V_{j-1}$, typically in the 1-ring or 2-ring vertex neighborhood.

A particularly important setting for subdivision scheme is when one apply the subdivision steps in parallel to three vectors (X_j, Y_j, Z_j) starting from three initial vectors describing the position in 3D space of a coarse mesh M_0. This allows to defines finer and finer spacial localization for the vertex of the refined meshes M_j. Figure 5.9 shows an example of such a subdivision surface. In order for the resulting infinitely refined surface to have good properties such as being continuous and even smooth, one needs to design carefully the interpolation operators. Next section gives examples of such operators.

5.3.2 Some Classical Subdivision Stencils

In order to define the interpolation operators P_j and $\tilde{P}_j$ of equation (5.1), one needs to use a naming convention for the neighborhoods of vertices.

For a vertex $\ell \in V_j$, the one ring neighborhood V_ℓ has already been defined in equation (3.1). It is the set of vertices adjacent to ℓ. In a regular point (that does not belongs to V_0 and not on a boundary of the mesh), its size is $|V_\ell| = 6$ since a point has 6 neighbors. This 1-ring is used to define $\tilde{P}_j$.

For a vertex $k \in H_j \subset V_{j-1}$, the butterfly neighborhood is a set of vertices in V_j close to k. This neighborhood is used to define P_j. The two immediate neighbors are

$$(v_k^1, v_k^2) \overset{\text{def.}}{=} \{v \in V_j \,;\, (v,k) \in E_{j-1}\}\,.$$

Two other vertices (w_k^1, w_k^2) are defined using the two faces adjacent to edge $(v_k^2, v_k^2) \in E_j$

$$f_k^1 = (v_k^1, v_k^2, w_k^1) \in F_j \quad \text{and} \quad f_k^2 = (v_k^1, v_k^2, w_k^2) \in F_j.$$

For edges E_j on the boundary of M_j, one one face is available, in which case we implicitly assume that $f_1 = f_2$ (reflecting boundary conditions). The four last vertices are defined using faces adjacent to f_1 and f_2:

$$\forall\, i,j = 1,2, \quad f_k^{i,j} \overset{\text{def.}}{=} (z_k^{i,j}, v_k^j, w_k^j) \in F_j \quad \text{with} \quad f_k^{i,j} \neq f_j.$$

Once again, reflecting boundary condition are applied for faces on the boundary of the mesh. The butterfly neighborhood is depicted on figure 5.8.

Linear Interpolating Scheme The simplest subdivision rule compute values along edge mide point using a simple linear interpolation as follow

$$\begin{cases} \forall\, k \in H_j, & (P_j f_j)(k) = \frac{1}{2}(f(v_k^1) + f(v_k^2)), \\ \forall\, \ell \in V_j, & (\tilde{P}_j f_j)(\ell) = f_j(\ell). \end{cases} \tag{5.2}$$

Since $\tilde{P}_j$ is the identity operator, this scheme is called interpolating. It means that value of f_0 on points of the coarse triangulation are kept during iteration of the subdivision.

Butterfly Interpolating Scheme The linear scheme creates function that are piecewise linear on each face of the coarse triangulation F_0. In order to create smooth surface, one needs to use more points in the butterfly neighborhood as follow

$$\begin{cases} \forall\, k \in H_j, & (P_j f_j)(k) = \frac{1}{2}\sum_{i=1}^2 f(v_k^i) + \frac{1}{8}\sum_{i=1}^2 f(w_k^i) - \frac{1}{16}\sum_{i,j=1}^2 f(z_k^{i,j}), \\ \forall\, \ell \in V_j, & (\tilde{P}_j f_j)(\ell) = f_j(\ell). \end{cases} \tag{5.3}$$

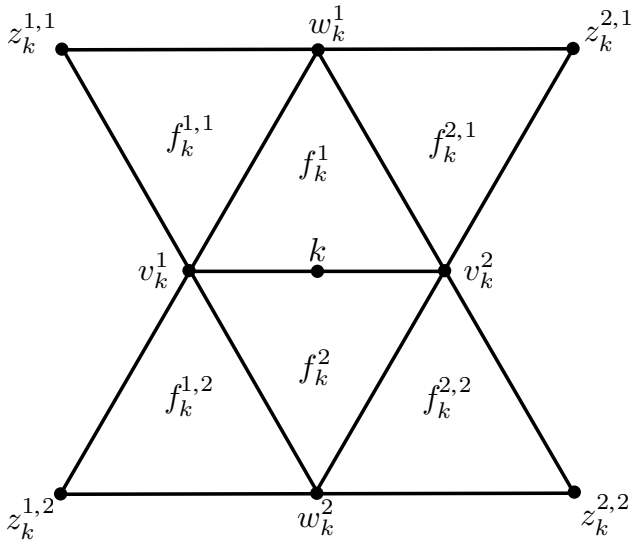

Figure 5.8: The butterfly neighborhood of a vertex $k \in H_j$.

Loop Approximating Scheme In order to gain flexibility in the subdivision design, one can also modify points in V_j during the iterations. This means that $\tilde{P}_j$ is not any more the identity, and that all the values will evolves during the iterations. The question of wether these iterated modification actually converge to a limit value is studied in the next section.

The Loop subdivision rule is defined as

$$\begin{cases} \forall\, k \in H_j, & (P_j f_j)(k) = \frac{3}{8} \sum_{i=1}^2 f(v_k^i) + \frac{1}{8} \sum_{i=1}^2 f(w_k^i), \\ \forall\, \ell \in V_j, & (\tilde{P}_j f_j)(\ell) = (1 - |V_\ell| \beta_{|V_\ell|}) f_j(\ell) + \beta_{|V_\ell|} \sum_{\ell' \in V_\ell} f_j(\ell'). \end{cases}$$

where the weights depends on the number of neighbors and are defined as

$$\beta_m \overset{\text{def.}}{=} \frac{1}{m} \left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4} \cos(2\pi/m) \right)^2 \right).$$

Other schemes. It is possible to define subdivision schemes using rules that do not involve a regular 1:4 splitting of each coarse face. For instance, in dual schemes such as the one depicted in figure 5.11, the faces of F_j are not included in F_{j-1} but only in F_{j-2}.

5.3.3 Invariant Neighborhoods

In order to study the convergence of subdivision schemes, one needs to consider independently each vertex $x \in V_{j_0(x)}$, where $j_0(x)$ is the coarser scale at which x appears

$$j_0(x) = \max \{ j \,;\, x \in V_j \}.$$

Original vertices satisfy $j_0(x) = 0$ and are the only one (except boundary vertices) that have a non-regular connectivity.

The vertex x belongs to the mesh $M_{j_0(x)}$ which is going to be refined through scales $j < j_0(x)$. In order to analyze this refinement, one needs to define an invariant neighborhood $V_j^x \subset V_j$ of x for each scale $j \leqslant j_0(x)$. These neighborhood are the set of points that are required to compute the operators P_j and $\tilde{P}_j$.

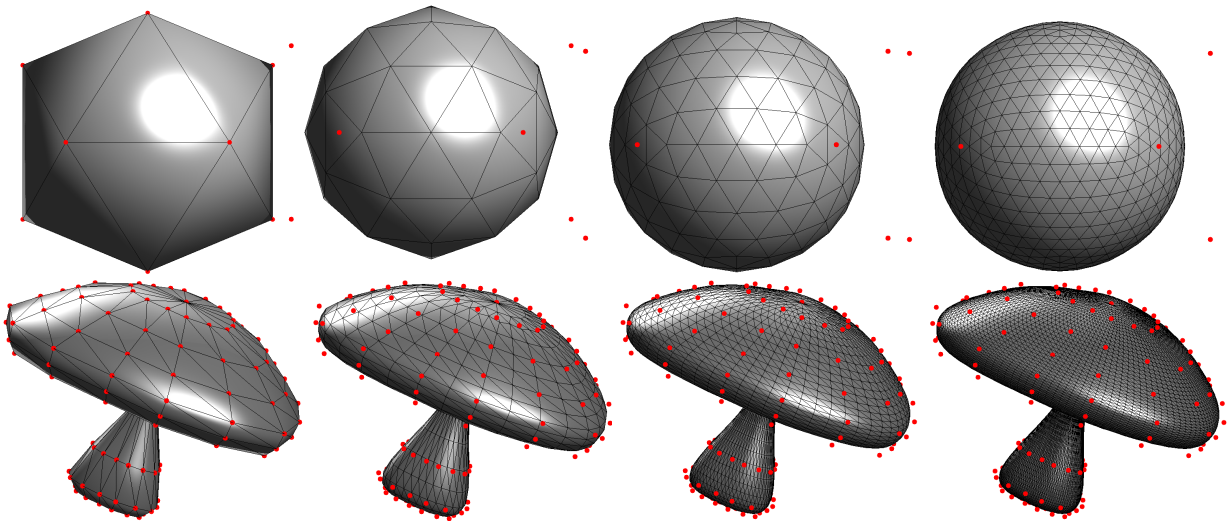

Figure 5.9: Examples of iterative subdivision using Loop scheme. The points (X_0, Y_0, Z_0) of the initial coarse mesh M_0 are shown in red.

More precisely, given a vector $f \in \ell^2(V_{j-1})$, the neighborhoods are required to satisfy

$$\begin{cases} \forall\, \ell \in V_{j-1}^x \cap V_j, & (\tilde{P}_j f)(\ell) \text{ depends only on } V_j^x \\ \forall\, k \in V_{j-1}^x \cap H_j, & (P_j f)(k) \text{ depends only on } V_j^x. \end{cases}$$

We further impose that all the invariant neighborhoods have the same size

$$\forall\, j \leqslant j_0(x), \quad \#V_j^x = m_x.$$

Figure 5.12 shows an example of invariant neighborhood which corresponds to the 2-ring $V_\ell^{(2)}$, as defined in (3.2).

Thanks to the invariance of these neighborhood systems, one can restrict the predictors around x and define

$$P_j^x : V_j^x \longrightarrow V_{j-1}^x \cap V_j \qquad \text{and} \qquad \tilde{P}_j^x : V_j^x \longrightarrow V_{j-1}^x \cap H_j.$$

The subdivision matrix $S_j^x \in \mathbb{R}^{m_x \times m_x}$ is then defined as matrix of the following mapping

$$(\tilde{P}_j^x, P_j^x) : V_x^j \longrightarrow V_x^{j-1}.$$

All the subdivision schemes studied in this chapter are invariant, meaning that the subdivision rule does not change through the scales j. This impose that the subdivision matrices are constant $S_j^x = S^x$. In fact, in all the examples given in the previous section, they only depends on the number $|V_x|$ of neighbors in the one ring of x.

5.3.4 Convergence of Subdivisions

The value at $x \in V_{j_0(x)}$ of a function $f_j \in \ell^2(V_j)$ obtained by subdividing at scale $j \leqslant j_0(x)$ an initial vector $f_0 \in \ell^2(V_0)$ can be computed as

$$f_j(x) = \left(S^x f_{j+1}^x\right)(x) = \left(\left(S^x\right)^{j_0(x)-j} f_{j_0(x)}^x\right)(x),$$

where the vector $f_i^x \in \mathbb{R}^{m_x}$ is the restriction of f_i to the set V_j^x.

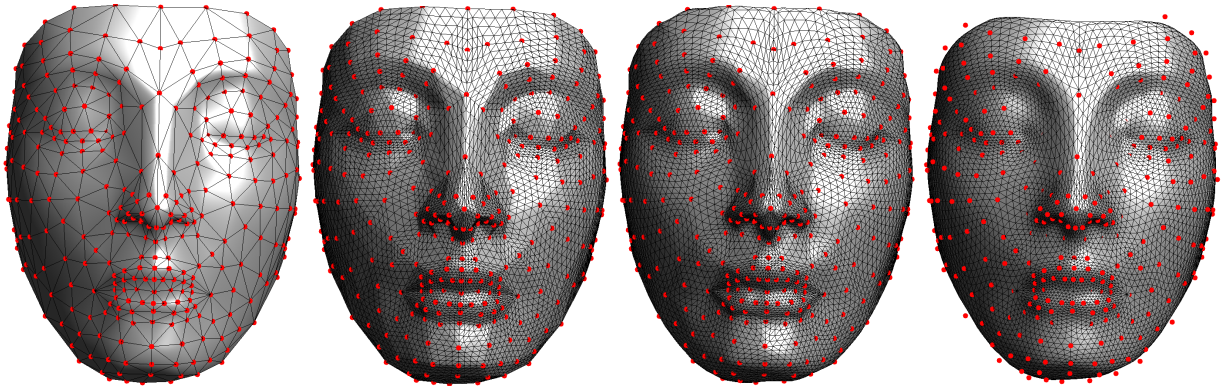

Figure 5.10: Examples of subdivision schemes. The points (X_0, Y_0, Z_0) of the initial coarse mesh M_0 are shown in red. Since the linear and butterfly scheme are interpolating, these points actually belongs to the limiting surface.

In order to analyze the limiting function resulting from an infinite number of subdivision, one can use the eigen vector decomposition of the matrix S^x

$$S^x = \tilde{\Phi} V \Lambda \Phi^{\mathrm{T}} \quad \text{where} \quad \begin{cases} \Phi^{\mathrm{T}} = \tilde{\Phi}^{-1}, \\ \Lambda = \mathrm{diag}(\lambda_i), \lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_{m_x}. \end{cases}$$

Since the subdivision matrix S_x is not symmetric, some of the eigenvalues might be complex, and we shall ignore this difficulty here. The fact that P_j and $\tilde{P}_j$ are predictor implies that the subdivision matrix has to satisfy $S^x 1 = 1$, meaning that $\tilde{\varphi}_1 = 1$ is an eigenvector associated to the eigenvalue 1. In the following we further makes the following assumption

$$1 = \lambda_1 < \lambda \stackrel{\text{def.}}{=} \lambda_2 = \lambda_3 < \lambda_4. \tag{5.4}$$

This hypothesis is satisfied by all the subdivision rules introduced in the previous section.

If one write $\Phi = (\varphi_i)_{i=1}^m$ and $\Phi = (\varphi_i)_{i=1}^m$, one has the following decomposition of a vector $f \in \mathbb{R}^{m_x}$

$$f = \sum_{i=1}^{m_x} \langle f, \varphi_i \rangle \tilde{\varphi}_i \quad \text{and} \quad (S^x)^k(x) = \sum_{i=1}^{m_x} \lambda_i^k \langle f, \varphi_i \rangle \tilde{\varphi}_i.$$

One thus has the following asymptotic expansion

$$\frac{1}{\lambda^k} (f - \langle f, \varphi_1 \rangle 1) = \langle f, \varphi_2 \rangle \tilde{\varphi}_2 + \langle f, \varphi_3 \rangle \tilde{\varphi}_3 + o(1). \tag{5.5}$$

This expression describes the asymptotic behavior of the subdivision scheme at zero order (position) and first order (tangents).

Theorem 25 (Convergence of the subdivision scheme)**.** *If the subdivision matrix S^x of a point x satisfies (5.4) then the subdivision process converges at x to the value*

$$f^j(x) \stackrel{j \to -\infty}{\longrightarrow} \langle f^x_{j_0(x)}, \varphi_1 \rangle.$$

The smoothness of the resulting function is more difficult to analyze. A particularly important setting is when one computes the subdivision of 3 function $p_0 = (X_0, Y_0, Z_0) \in \ell^2(V_0)^3$ corresponding to the position

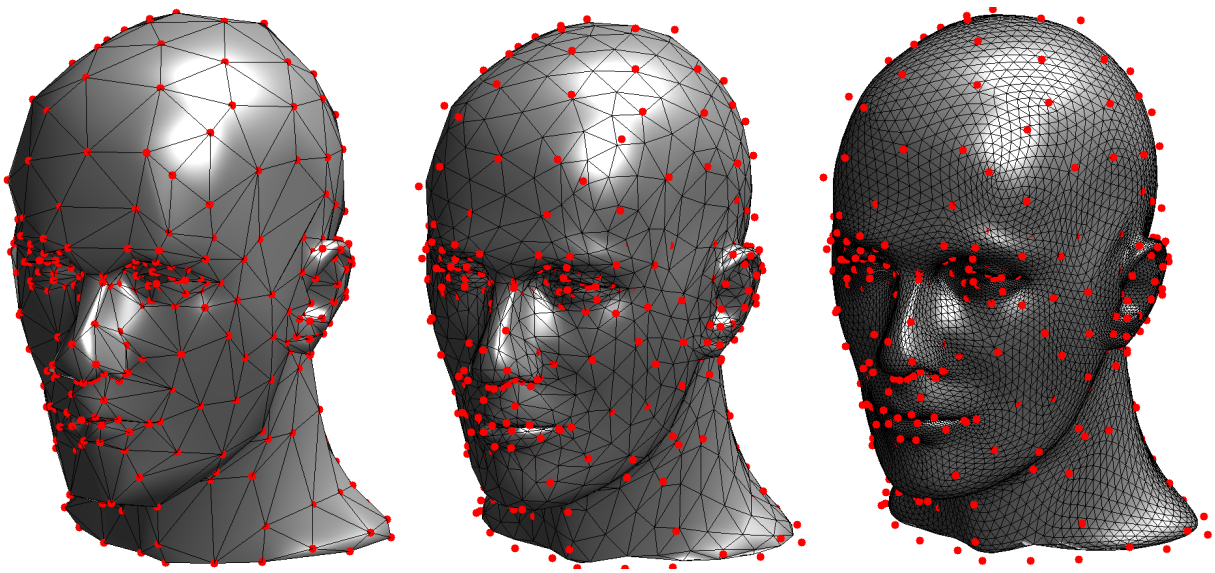

Figure 5.11: Surface after 0, 1 and 3 step of $\sqrt{3}$ subdivision [18].

in $\mathbb{R}^3$ (geometrical realization) of a coarse mesh M_0. In this case, the subdivided functions $p_j = (X_j, Y_j, Z_j)$ gives refined 3D meshes that converge uniforlmy to a continuous surfaces

$$p(x) = (X(x), Y(x), Z(x)) = (\langle X^x_{j_0}, \varphi_1\rangle, \langle Y^x_{j_0}, \varphi_1\rangle, \langle Z^x_{j_0}, \varphi_1\rangle).$$

Condition (5.4) nearly implies that the resulting surface is smooth. Indeed, the asymptotic expansion (5.5) shows that for a point x' near x in the subdivision domain, the differential vector can be well approximated as a projection on a 2D plane

$$p(x) - p(x') + o(1) \in \mathrm{Span}(\tau^x_2, \tau^x_3) \quad \text{where} \quad \tau^i(x) \stackrel{\text{def.}}{=} (\langle X^x_{j_0}, \varphi_i\rangle, \langle Y^x_{j_0}, \varphi_i\rangle, \langle Z^x_{j_0}, \varphi_i\rangle).$$

If the vectors τ^x_2 and τ^x_3 are linearly independent, they form a basis of the tangent plane at $p(x)$.

Example of the Loop subdivision. For the Loop interpolation operators defined in equation (5.3.2), the invariant neighborhood V^x_j correspond to the 2-ring of x in the triangulation G_j, as shown in figure 5.12. For a vertex with k neighbors, $|V_x| = k$, the size of these invariant neighborhood is $m_x = 3k + 1$. A particular neighboring for $k = 3$ is depicted in figure 5.12, together with an indexing in $\{0, \ldots, 3k = 9\}$ of the points in V^x_j and V^x_{j-1}. For this indexing, the subdivision matrix reads

$$\begin{pmatrix}
7 & & & & & & 3 & 3 & 3 \\
1 & 1 & & & 1 & & 1 & 10 & 1 & 1 \\
1 & & 1 & & 1 & 1 & & 1 & 10 & 1 \\
1 & & & 1 & & 1 & 1 & 1 & 1 & 10 \\
1 & & & & 1 & & & 3 & 3 & \\
1 & & & & & 1 & & & 3 & 3 \\
1 & & & & & & 1 & 3 & & 3 \\
1 & & & & & & & 3 & 1 & 1 \\
1 & & & & & & & 1 & 3 & 1 \\
1 & & & & & & & 2 & 1 & 3
\end{pmatrix}$$

where the 0's have been omitted and where the rows should be rescaled to sum to 1. The eigenvalues of this matrix satisfy $\lambda_1 = 1$ and $\lambda_2 = \lambda_3 = 1/3 > \lambda_4$.

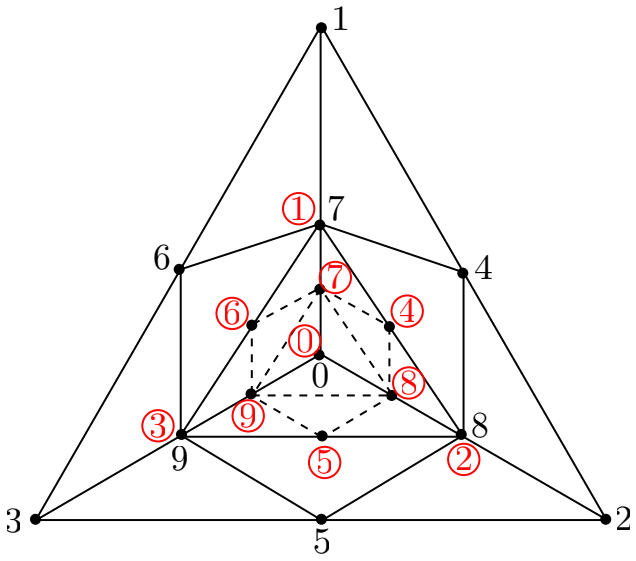

Figure 5.12: Invariant neighborhood V_j^x and V_{j-1}^x (indexing with red circles) of the Loop subdivision scheme for a vertex of valence $|V_\ell| = 0$. The number in $\{0, \dots, 9\}$ refers to the numbering of the vertices in V_j^x and V_{j-1}^x

5.4 Wavelets on Meshes

5.4.1 Multiscale Biorthogonal Bases on Meshes

The transforms considered in this section are multiscale and indexed by the set of nested grids $(V_j)_{L<j\leqslant J}$. This corresponds to computing a set of coefficients $(d_j)_{L<j\leqslant J} \cup f_J$ from an initial input signal f. These coefficients corresponds to inner products with basis vectors

$$\begin{cases} d_j \in \ell^2(H_j) & \text{where} \quad \forall\, k \in H_j,\ d_j(k) = \langle f,\, \psi_{j,k} \rangle, \\ f_J \in \ell^2(V_J) & \text{where} \quad \forall\, \ell \in V_J,\ f_J(\ell) = \langle f,\, \varphi_{J,\ell} \rangle. \end{cases}$$

By analogy with the wavelet setting, the vectors $\psi_{j,k} \in \mathbb{R}^n$ corresponds to primal wavelets and are intended to capture the details present in the signal f at a scale j, whereas the scaling vectors $\varphi_{J,k} \in \mathbb{R}^n$ capture the missing coarse approximation of f at scale J. This decomposition is stopped at any coarse scale $L < J \leqslant 0$.

In order to reconstruct the function f from this set of transformed coefficients, one needs to use a set of bi-orthogonal basis vectors

$$f = \sum_{L<j\leqslant J, k\in H_j} d_j(k)\tilde{\psi}_{j,k} + \sum_{\ell\in V_J} f_J(\ell)\tilde{\varphi}_{J,\ell}.$$

If this reconstruction formula holds for any scale $L < J \leqslant 0$, the set of vectors

$$(\psi_{j,k}, \varphi_{j,\ell})_{k\in H_j, \ell\in V_j}^{L<j\leqslant 0} \qquad \text{and} \qquad (\tilde{\psi}_{j,k}, \tilde{\varphi}_{j,\ell})_{k\in H_j, \ell\in V_j}^{L<j\leqslant 0}, \tag{5.6}$$

is said to be a pair of primal and dual multiscale bases (together with their scaling functions).

The following paragraph shows how one can modify such a pair of multiscale bases while still maintaining the biorthogonality property. This lifting process is useful to design multiscale bases with various properties on complicated domains.

5.4.2 The Lifting Scheme

The lifting scheme is a construction of multiscale biorthogonal bases introduced by Sweldens [34, 35]. It extends the traditional construction of wavelets in two main directions:

– As explained in [11], it allows to implement already existing filter banks more efficiency by splitting the computation into elementary blocks. This computational gain is described at the end of the section together with the factorization of wavelets into lifting steps.
– It allows to define multiscale transforms over domains that are not translation invariant. This section gives two examples of such transforms: a non-separable 2D wavelet transform and wavelets on triangulated meshes.

In order to build wavelets on triangulation, one can specialize the lifting scheme to a particular setting where only two lifting steps are applied.

Forward lifting scheme. The forward algorithm performs the transform

$$(f_{j-1}(\ell))_{\ell\in V_{j-1}} \quad \longrightarrow \quad (d_j(k))_{k\in H_j} \cup (f_j(\ell))_{\ell\in V_j}$$

by applying the following steps
– *Splitting:* this corresponds selecting the coefficient of $f_{j-1}(\ell)$ that are in V_j or in H_j

$$(f_{j-1}(\ell))_{\ell\in V_{j-1}} = (f_j(\ell))_{\ell\in V_j} \cup (f_j(\ell))_{\ell\in H_j}.$$

These two sets of coefficients are treated differently in the two remaining steps of the transform.
– *Predict step:* creates wavelets coefficients d_j by computing local differences between each coefficient in V_j and its neighbors in H_j

$$\forall\, k \in H_j, \quad d_j(k) = f_{j-1}(k) - \sum_{\ell\in V_j} p_j(k,\ell) f_{j-1}(\ell).$$

The coefficients $p_j(k,\ell)$ are weights that determine the predict operator

$$P_j : \left\{ \begin{array}{ccc} \ell^2(V_j) & \longrightarrow & \ell^2(H_j) \\ g & \longmapsto & h = P_j g \end{array} \right. \quad \text{where} \quad h(k) = \sum_{k\in H_j} p_j(k,\ell) g(\ell).$$

– *Update step:* enhance the properties of each remaining low pass coefficients $f_{j-1}(\ell)$ for $\ell \in V_j$ by pooling locally the wavelets coefficients $d_j(k)$ for k around ℓ

$$\forall\, \ell \in V_j, \quad f_j(\ell) = f_{j-1}(\ell) + \sum_{k\in H_j} u_j(\ell,k) d_j(k).$$

The coefficients $u_j(\ell,k)$ are weights that determine the update operator

$$U_j : \left\{ \begin{array}{ccc} \ell^2(H_j) & \longrightarrow & \ell^2(V_j) \\ h & \longmapsto & g = U_j h \end{array} \right. \quad \text{where} \quad g(\ell) = \sum_{k\in H_j} u_j(\ell,k) h(k).$$

Figure 5.13, top row, shows the block diagram associated to this forward lifting wavelet transform.

The iterations of the forward lifting transform can also be written in vector and operator format

$$\left\{ \begin{array}{l} d_j = f_{j-1}^{H_j} - P_j f_{j-1}^{V_j}, \\ f_j = f_{j-1}^{V_j} + U_j d_j = (\mathrm{Id}_{V_j} - U_j P_j) f_{j-1}^{V_j} + U_j f_{j-1}^{H_j}, \end{array} \right.$$

where g^A is the restriction of some vector g to the set A.

Backward lifting scheme. The backward transform algorithm does the reverse computation

$$(d_j(k))_{k\in H_j} \cup (f_j(\ell))_{\ell\in V_j} \quad \longrightarrow \quad (f_{j-1}(\ell))_{\ell\in V_{j-1}}$$

One of the main feature of the lifting scheme is that this is achieved by simply reversing the order of the lifting steps and interchanging +/- signs.

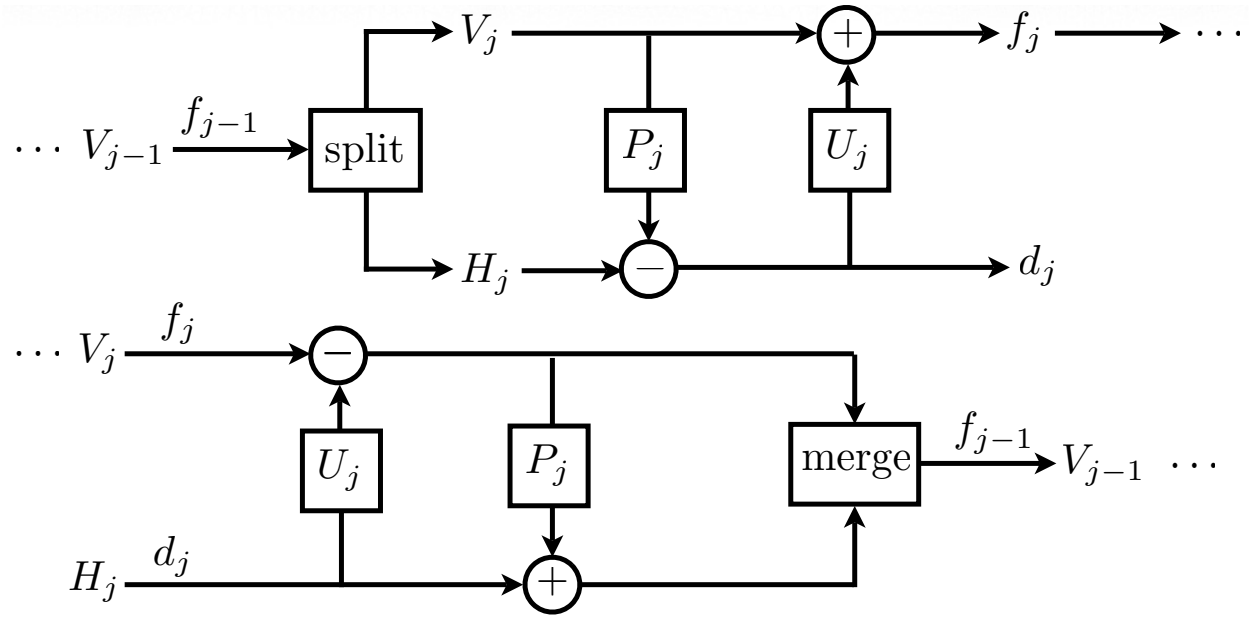

Figure 5.13: Block diagrams for the forward and backward lifting scheme.

– *Inverse update step:*

$$\forall\, \ell \in V_j, \quad f_{j-1}(\ell) = f_j(\ell) - \sum_{k \in H_j} u_j(\ell, k) d_j(k).$$

– *Inverse predict step:*

$$\forall\, k \in H_j, \quad f_{j-1}(k) = d_j(k) + \sum_{\ell \in V_j} p_j(k, \ell) f_{j-1}(\ell).$$

– *Merging:* makes the union of the coefficients computed in the two previous steps

$$(f_{j-1}(\ell))_{\ell \in V_{j-1}} = (f_j(\ell))_{\ell \in V_j} \cup (f_j(\ell))_{\ell \in H_j}.$$

Figure 5.13, top row, shows the block diagram associated to this backward lifting wavelet transform.

The lifting scheme is more general than the algorithm described in this section since several passes of predict/update steps can be applied to further enhance the properties of the resulting transform. However, the steps beyond the two initial ones are difficult to analyze, except in the notable exception of points sampled evenly on a 1D axes, where a factorization algorithm [11] allows to recover traditional wavelet filters.

5.4.3 Imposing vanishing moments.

The operator P_j is called a predictor since the values of $P_j f_{j-1}^{V_j}$ should typically be close to $f_{j-1}^{H_j}$ for the wavelet coefficients d_j to be small. Such predictors have already been constructed in equations (5.2), (5.3) and (5.3.2).

The operator U_j is called an update operator since the additional term $U_j d_j$ should enhance the properties of $f_{j-1}^{V_j}$. This update steps does not appears in the theory of subdivision surface and this section considers a local update operator which guaranty the conservation of the mean value when switching from f_{j-1} to f_j.

Polynomial vectors. In order to select predict and update operator that have good properties, one follow the insight gained from the analysis of the wavelet approximation of signal on the real line. In order to do

so, one need analyze the effect of a lifting wavelet transform on polynomials. The most basic constraint enforces one vanishing moment by imposing orthogonality with the constant vector $\Phi_0 = 1$. This constraint does not require to known the spacial location x_ℓ of each index $\ell \in V_L$. In order to impose higher order vanishing moments, one needs to assume some sampling pattern, for instance

$$\forall\, \ell \in V_L, \quad f(\ell) = \bar{f}(x_\ell) \qquad \text{where} \qquad x_\ell \in \mathbb{R}^q$$

and where $\bar{f}$ is a function defined on $\mathbb{R}^q$. For instance, the points x_ℓ might corresponds to a regular sampling of the line (this is the traditional wavelet setting) or to an irregular sampling of a 2D surface embedded in $\mathbb{R}^3$. The next paragraphs describe several situations with different sampling grids. Once the precise locations of the samples are known, one can for instance select Φ_s as some monomials of degree $(s_1, \ldots, s_q)$ over $\mathbb{R}^q$.

Vanishing moment and polynomials reproduction. Having defined these polynomial vectors, one requires that the following constraints are fulfilled.

– *Vanishing moments:* the wavelet coefficients of a low order polynomial should be 0, which implies that

$$\forall\, k \in H_j,\ \langle \Phi_s,\, \psi_{j,\ell} \rangle = 0. \tag{5.7}$$

– *Polynomial reproduction:* coarse coefficients f_j computed from a polynomial f_{f-1} should also be polynomials, which implies that

$$\forall\, \ell \in V_j,\ \langle \Phi_s,\, \varphi_{j,\ell} \rangle = \Phi_s(\ell) \tag{5.8}$$

In order for the wavelets and scaling function to satisfy conditions (5.7) and (5.8), the predict operator P_j and update operator U_j should be designed carefully. One can impose these constraint from the fine scale $j = L$ until the coarse scale $j = 0$. Indeed, if $(\varphi_{j-1,\ell}, \psi_{j-1,\ell})_{k,\ell}$ satisfy conditions (5.7) and (5.8), then, for the scale j

$$\forall\, s \in S, \quad \begin{cases} (5.7) & \iff & P_j \Phi_s^{V_j} = \Phi_s^{H_j}, \\ (5.8) & \iff & {U_j}^{\mathrm{T}} \left(\Phi_s^{V_j} + {P_j}^{\mathrm{T}} \Phi_s^{H_j} \right) = \Phi_s^{H_j}. \end{cases}$$

where $\Phi_s^A \in \ell^2(A)$ is the restriction of Φ_s to A.

In contrast, the constraint (5.8) on the update operator P_j is more involved and the next section shows how to handle it on a triangulation situations for only one vanishing moment $|S| = 1$.

5.4.4 Lifted Wavelets on Meshes

The lifted wavelet bases can be used to process signals $f \in \ell^2(V_L)$ where $\ell \in V_L$ index a sampling x_ℓ of an arbitrary surface. The construction of biorthogonal wavelets on triangulated mesh has been first proposed by Lounsbery et al. [19] and re-casted into the lifting scheme framework by Schroeder and Sweldens [29, 30].

Designing predict operators. The constraints (5.7) on the predictor P_j is easily solved. For instance, for each k, one selects only $|S|$ non vanishing weights $(p_j(k, \ell))_\ell$ and solves a small $|S| \times |S|$ linear system. Furthermore, in the case of a regular triangulation with edges of constant length, predictors with several vanishing moments have been already defined in (5.2), (5.3) and (5.3.2). Figure 5.14 shows the weights for these predictors.

One can choose any of these operators, and creates respectively linear, butterfly and Loop wavelets bases. All these predictors have one vanishing moment since they satisfy $P_j 1^{H_j} = 1^{V_j}$. In fact they have more vanishing moments if one consider polynomials Φ_s sampled at points $x_\ell \in \mathbb{R}^2$ of an hexagonal tiling with constant edge length. In practice, if the triangulation under consideration have edges with smoothly varying length, the resulting predictor are efficient to predict the value of smooth functions on the triangulation.

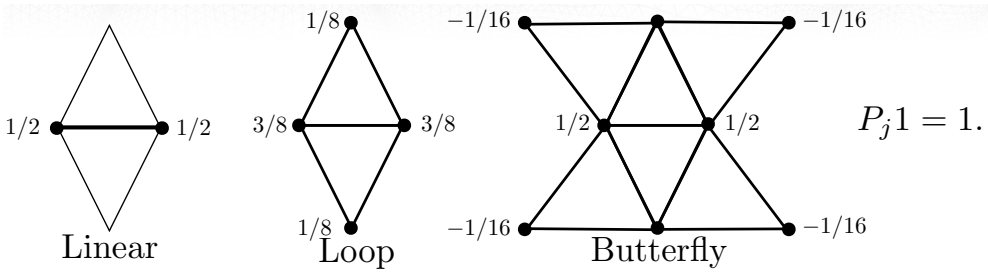

Figure 5.14: Predict operators on a triangulation.

Designing update operators. In order to ensure the reproduction of constant polynomials, we design the update operator so that it depends only on the direct neighbors in H_j of each point in V_j

$$\forall\, \ell \in V_j, \quad V_\ell = \{\gamma(\ell, \ell')\ ;\ (\ell, \ell') \in E_j\}\,.$$

One then looks for a valid update operator in the following form

$$\forall\, h \in \ell^2(H_j),\ \forall\, \ell \in V_j, \quad (U_j h)(\ell) = \lambda_\ell \sum_{k \in V_\ell} h(k), \tag{5.9}$$

where each λ_ℓ should be fixed in order for condition (5.8) to be satisfied.

In an semi-regular triangulation, $|V_\ell| = 6$ excepted maybe for some points in the coarse grid $\ell \in V_0$. In this setting, the values of λ_ℓ can be computed by a recursion through the scales. In an ideal triangulation where $|V_\ell| = 6$ for all ℓ, one can use a constant weight $\lambda_\ell = \lambda$.

For the predictors defined in (5.2), (5.3) and (5.3.2), one has

$$P_j^{\mathrm{T}} 1^{H_j} = 3 \times 1^{V_j} \qquad \text{and} \qquad U_j^{\mathrm{T}} 1^{V_j} = 6\lambda 1^{H_j}$$

so setting $\lambda_\ell = 1/24$ solves equation (5.8). Figure 5.15 shows examples of butterfly wavelets on a planar semi-regular triangulation.

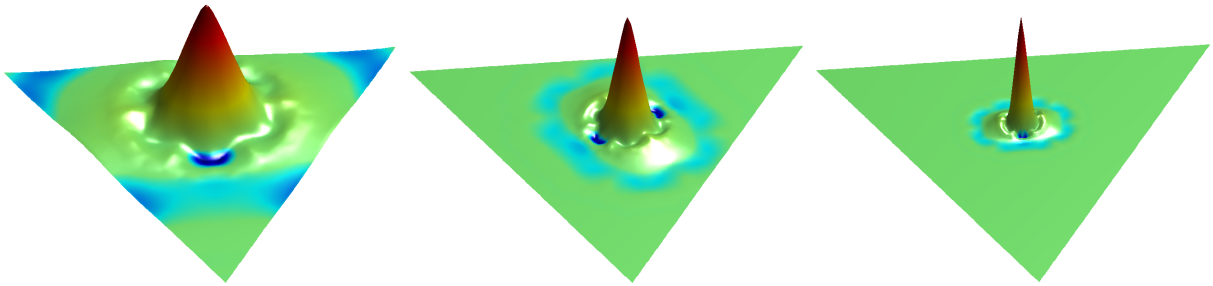

Figure 5.15: Example of wavelets $\psi_{j,k}$ on a semi-regular triangulation. The height over the triangle (together with the color) indicates the value of the wavelet vector.

5.4.5 Non-linear Mesh Compression

These wavelets can be used to perform an approximation of a function $f \in \ell^2(V_L)$ defined on the fine triangulation. For instance a wavelet approximation can be applied to each coordinate $f_i, i = 1, 2, 3$ of the

actual position $x_\ell = (f_1(\ell), f_2(\ell), f_3(\ell)) \in \mathbb{R}^3$ of the surface points, as done in [16, 17]. This leads to a scheme to approximate and compress a 3D surface using the lifted biorthogonal wavelets associated to the semi-regular triangulation. This is possible because these wavelets depend only on the combinatorial grids V_j and not on the precise position of the samples x_ℓ in 3D.

In order to perform a wavelet approximation in this biorthogonal basis, one uses a non-linear thresholding at $T > 0$

$$f = \sum_{(j,k)\in I_T} \langle f, \psi_{j,k}\rangle \tilde{\psi}_{j,k}$$

$$\text{where} \quad I_T = \left\{ (j,k) \;;\; k \in H_j \quad \text{and} \quad |\langle f, \psi_{j,k}\rangle| > T |\operatorname{supp}(\psi_{j,k})|^{-1/2} \right\}.$$

Note that for each coefficient the threshold T is scaled according to the size of the support of the wavelet in order to approximately normalize the wavelets in $\ell^2(V_L)$ norm.

Figure 5.16 shows an example of compression of the position of a vertex in 3D spaces as 3 functions defined on a semi-regular mesh. Figure 5.17 shows an example of compression of a spherical texture map which is a single function defined at each vertex of a semi-regular mesh obtained by subdividing an icosaedron.

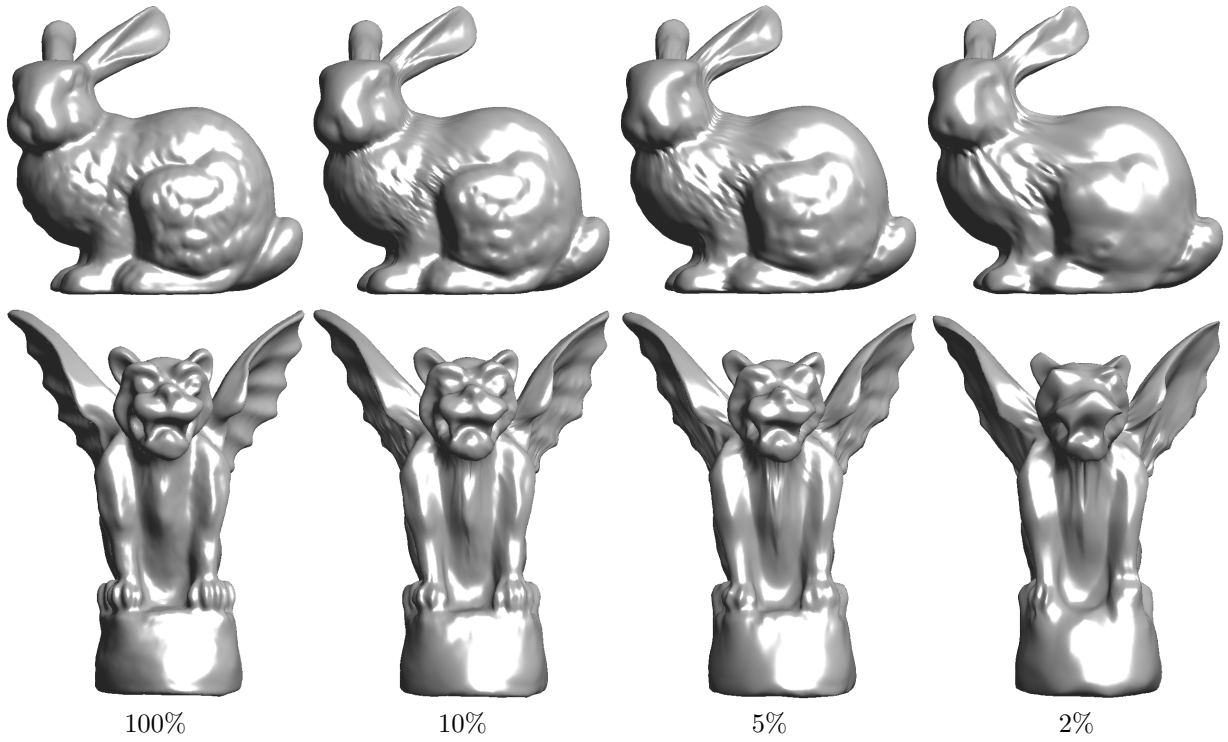

Figure 5.16: Non-linear wavelet mesh compression with a decreasing number of coefficients.

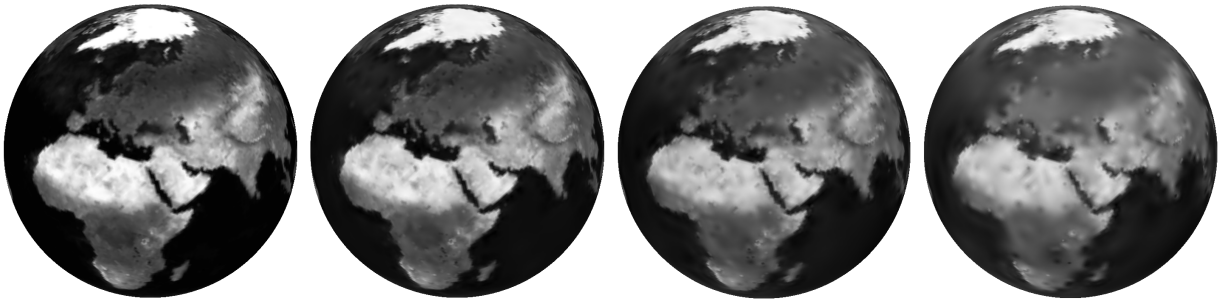

Figure 5.17: Non-linear spherical wavelet compression with a decreasing number of coefficients.

Bibliography

[1] P. Alliez and C. Gotsman. Recent advances in compression of 3d meshes. In N. A. Dodgson, M. S. Floater, and M. A. Sabin, editors, *Advances in multiresolution for geometric modelling*, pages 3–26. Springer Verlag, 2005.

[2] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene. Recent advances in remeshing of surfaces. In *AIM@SHAPE repport*. 2005.

[3] E. Candès and D. Donoho. New tight frames of curvelets and optimal representations of objects with piecewise C^2 singularities. *Commun. on Pure and Appl. Math.*, 57(2):219–266, 2004.

[4] E. J. Candès, L. Demanet, D. L. Donoho, and L. Ying. Fast discrete curvelet transforms. *SIAM Multiscale Modeling and Simulation*, 5:861–899, 2005.

[5] A. Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vis.*, 20:89–97, 2004.

[6] S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[7] F. R. K. Chung. Spectral graph theory. *Regional Conference Series in Mathematics, American Mathematical Society*, 92:1–212, 1997.

[8] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *SIAM Multiscale Modeling and Simulation*, 4(4), 2005.

[9] P. Schroeder et al. D. Zorin. Subdivision surfaces in character animation. In *Course notes at SIGGRAPH 2000*, July 2000.

[10] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. on Pure and Appl. Math.*, 57:1413–1541, 2004.

[11] I. Daubechies and W. Sweldens. Factoring wavelet transforms into lifting steps. *J. Fourier Anal. Appl.*, 4(3):245–267, 1998.

[12] D. Donoho and I. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 81:425–455, Dec 1994.

[13] Heinz Werner Engl, Martin Hanke, and Andreas Neubauer. *Regularization of inverse problems*, volume 375. Springer Science & Business Media, 1996.

[14] M. Figueiredo and R. Nowak. An EM Algorithm for Wavelet-Based Image Restoration. *IEEE Trans. Image Proc.*, 12(8):906–916, 2003.

[15] M. S. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In N. A. Dodgson, M. S. Floater, and M. A. Sabin, editors, *Advances in multiresolution for geometric modelling*, pages 157–186. Springer Verlag, 2005.

[16] I. Guskov, W. Sweldens, and P. Schröder. Multiresolution signal processing for meshes. In Alyn Rockwood, editor, *Proceedings of the Conference on Computer Graphics (Siggraph99)*, pages 325–334. ACM Press, August8–13 1999.

[17] A. Khodakovsky, P. Schröder, and W. Sweldens. Progressive geometry compression. In *Proceedings of the Computer Graphics Conference 2000 (SIGGRAPH-00)*, pages 271–278, New York, July 23–28 2000. ACMPress.

[18] L. Kobbelt. $\sqrt{3}$ subdivision. In Sheila Hoffmeyer, editor, *Proc. of SIGGRAPH'00*, pages 103–112, New York, July 23–28 2000. ACMPress.

[19] M. Lounsbery, T. D. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Trans. Graph.*, 16(1):34–73, 1997.

[20] S. Mallat. *A Wavelet Tour of Signal Processing, 3rd edition.* Academic Press, San Diego, 2009.

[21] Stephane Mallat. *A wavelet tour of signal processing: the sparse way.* Academic press, 2008.

[22] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Commun. on Pure and Appl. Math.*, 42:577–685, 1989.

[23] Y. Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1, Ser. A):127–152, 2005.

[24] Gabriel Peyré. *L'algèbre discrète de la transformée de Fourier.* Ellipses, 2004.

[25] J. Portilla, V. Strela, M.J. Wainwright, and Simoncelli E.P. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Trans. Image Proc.*, 12(11):1338–1351, November 2003.

[26] E. Praun and H. Hoppe. Spherical parametrization and remeshing. *ACM Transactions on Graphics*, 22(3):340–349, July 2003.

[27] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992.

[28] Otmar Scherzer, Markus Grasmair, Harald Grossauer, Markus Haltmeier, Frank Lenzen, and L Sirovich. *Variational methods in imaging.* Springer, 2009.

[29] P. Schröder and W. Sweldens. Spherical Wavelets: Efficiently Representing Functions on the Sphere. In *Proc. of SIGGRAPH 95*, pages 161–172, 1995.

[30] P. Schröder and W. Sweldens. Spherical wavelets: Texture processing. In P. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95.* Springer Verlag, Wien, New York, August 1995.

[31] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[32] A. Sheffer, E. Praun, and K. Rose. Mesh parameterization methods and their applications. *Found. Trends. Comput. Graph. Vis.*, 2(2):105–171, 2006.

[33] Jean-Luc Starck, Fionn Murtagh, and Jalal Fadili. *Sparse image and signal processing: Wavelets and related geometric multiscale analysis.* Cambridge university press, 2015.

[34] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computation Harmonic Analysis*, 3(2):186–200, 1996.

[35] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, 1997.