Patch-based de-noising algorithms and patch manifold smoothing have emerged as efficient de-noising methods. This paper provides a new insight on these methods, such as the Non Local Means or the image graph de-noising, by showing its use for filtering a selected pattern.

Filtering via a Reference Set

A.Haddad[†], D. Kushnir[‡], R.R. Coifman[‡]
Technical Report YALEU/DCS/TR-1441
February 21, 2011

Keywords: *NL-Means, diffusion processes, diffusion geometry, graph filtering, patch manifold.*

Filtering via a Reference Set

Ali Haddad[1] Dan Kushnir [2] Ronald Coifman [3]

Abstract

Patch-based de-noising algorithms and patch manifold smoothing have emerged as efficient de-noising methods. This paper provides a new insight on these methods, such as the Non Local Means [1] or the image graph de-noising [8], by showing its use for filtering a selected pattern.

Keywords: NL-Means, diffusion processes, diffusion geometry, graph filtering, patch manifold.

1 Introduction

One of the most efficient patch-based de-noising algorithm is the so-called Non Local Means (NL-Means) of Buades et al [1]. The fundamental idea is to consider a natural image as a collection of sub-images or patches. Each noisy patch is approximated by a weighted average of patches where the weights reflect the similarity between any two patches. The averaging reduces the noise. One way to interpret the patch-based de-noising methods is to adopt the manifold point of view. We lift the grayscale-valued image and embed it in a higher dimensional space by considering the manifold of patches. The grayscale value is seen as a function defined on the manifold. Although a patch is high dimensional (for example a patch of size 7×7 is of dimension 49), in practice the intrinsic dimension is much smaller. The weighted average of patches can be seen as a smoothing on the manifold. For example, in [8] the heat equation on the manifold is used for de-noising.

In this work, our primary interest is not de-noising but filtering. More precisely, we are interested in decomposing the image into two components; one corresponding to a specific pattern that is provided by the user and a second one corresponding to the residual. The pattern of interest (it could be several patterns) is defined by a set of patches of fixed size. This set is a subset of the collection of patches provided by the image. It is also a subset of the manifold formed by the patches. From now on, we refer to it as the "reference set". We propose to modify the NL-Means algorithm and its graph-based diffusion variants taking into account the given reference set. This set has to represent all the patches extracted from the specific pattern, in the sense that any patch related to the pattern of interest should not be "too far" from the reference set with respect to some metric between patches. NL-Means uses the Euclidian metric between patches. In the graph-based method described in [8], each pixel is associated to a feature vector that is built by convolution of the patch at that pixel with a filter bank. The distance is defined as the Euclidian distance between feature vectors. In both methods, the affinity between pixels is a nonnegative decreasing function of the metric (e.g. a gaussian function). NL-Means averages the value at each pixel by weighting the sum by the affinities. We propose first to keep those weights but restrict the averaging over pixels whose patches are in the reference set. In addition to de-noising the image, we do two tasks; de-noising and filtering according to the reference set. This defines an operator that we name "non local filter operator". We show the limitations of this operator and propose a variant that we name "projected non local filter operator". We then introduce the graph-based diffusion approach relative to the reference set. The pixels form the nodes of a graph. The edges are defined once we define a notion of affinity or neighborhood between nodes. This is defined relative to the reference set. This definition is crucial for computing efficiently the eigenfunctions of the Laplace-Beltrami operator of the whole set. We show that there exists an extension formula between the eigenfunctions on the reference set and the eigenfunctions on the whole set.

[1]ali.haddad@yale.edu.
[2]dan.kushnir@yale.edu
[3]ronald.coifman@yale.edu
Department of Mathematics, Yale University.
New Haven, CT 06511

This paper is organized as follows. In section 2, we present the non local filter and the projected non local filter operators and motivate our preference for the latter. In section 3, the general framework of our graph-based diffusion approach relative to the reference set is explained. Details of the extension formula and its use for projection are given. In section 4, we demonstrate our methods in filtering and show that the eigenfunctions of the Laplace-Beltrami operator yield an embedding of the manifold of images into a low-dimensional meaningful space [2].

2 Non Local Filter using a reference set

In this section we first present the NL-Means algorithm and then present two modified versions that aim at filtering. The first method we propose, the non local filter operator, is a trivial extension of NL-Means algorithm. This method performs decently when the desired texture is not mixed with other textures. We illustrate this point in section 4. We then show how to overcome the limitations and propose another algorithm that we name projected non local filter operator.
The NL-Means algorithm has been introduced for de-noising purposes. Any pixel, say i, is associated to a fixed-size rectangle centered at it that we call *patch*(i). The value at location i, say $v(i)$, is replaced by a weighted average of all the pixels in the image, say $NL(v)(i)$, defined as follows:

$$NL(v)(i) = \sum_{j \in I} a(i,j)v(j) \tag{1}$$

where the weights $\{a(i,j)\}_j$ - all non negative - depend on the similarity between the pixels i and j and satisfy $\sum_{j \in I} a(i,j) = 1$. The similarity weight is

$$a(i,j) = \frac{1}{Z(i)} e^{-\frac{||patch(i)-patch(j)||_2^2}{\epsilon}} \tag{2}$$

where $Z(i)$ is the normalizing factor $Z(i) = \sum_j e^{-\frac{||patch(i)-patch(j)||_2^2}{\epsilon}}$. The patches *patch*$(i)$ and *patch*(j) are seen as column vectors. The parameter ϵ controls the decay of the exponential function. The similarity weight is close to 1 if the patches at pixels i and j are close in L^2 sense. This method can be interpreted as follows: The set of patches is a dictionary of the image that is redundant. Any corrupted patch is then replaced by a weighted average. The noise is attenuated by the averaging. The NL-Means has proved to be one of the most efficient algorithm for de-noising.

Here we wish to adapt the NL-Means algorithm in order to filter the image given a specific pattern we want to extract. For example we would like to extract one specific texture from a natural image that might contain several textures. In that sense, we build the dictionary based on the patches extracted from the texture we are interested in. In other words, we only consider a sub-dictionary of the whole dictionary of the image and use it to reconstruct the image. Let $\mathcal{R}$ be the given set of pixels that defines the reference set. As we already said, this set should "represent" the pattern we want to extract; any patch similar to the pattern has to be close (in L^2 sense) to the set of patches centered at any pixels of $\mathcal{R}$. Our first modification of NL-Means is the non local filter operator that is defined as follows:

$$NLF(v)(i) = \sum_{j \in \mathcal{R}} a(i,j)v(j) \tag{3}$$

where $a(i,j) \geq 0$ and $\sum_{j \in \mathcal{R}} a(i,j) = 1$. The similarity weight is defined as in (2). The matrix formulation is

$$NLF(v) = Av. \tag{4}$$

where $A = (a(i,j))_{i \in I\, j \in \mathcal{R}}$ is row-stochastic. In practice, A is sparse. The weights $a(i,j)$ are thresholded or only computed between nearest neighbors. Here we do the following. For a given pixel $i \in I$, its kernel density relative to $\mathcal{R}$ is $Z(i)$. If the density is below a threshold, say δ, then the reference set does not represent the patch at pixel i. The distance between *patch*(i) and patches coming from $\mathcal{R}$ is big. So, we discard this pixel by setting to 0 all the weights $a(i,\cdot)$. In other words, whenever $\sum_{j \in \mathcal{R}} e^{-\frac{||patch(i)-patch(j)||_2^2}{\epsilon}} < \delta$, $NLF(v)(i)$ is defined as 0. This non local filter is very simple to implement. The only parameters are the width of the Gaussian, ϵ, we are using to define the weights and the threshold δ. The smaller ϵ is, the bigger the similarity coefficients are. The smaller δ is the bigger the support of NLF is. We demonstrate some successful applications of this algorithm in section 4. However this simple model shows its limit when dealing with complicated situation. In particular it cannot resolve intersections. Indeed, in the case where the texture we are interested in intersects another texture at a pixel i the density kernel $Z(i)$ might be small. The $patch(i)$ is roughly speaking the sum of two patches: one coming from the desired texture and one coming from the crossing texture. The distance between $patch(i)$ and the patches coming from the reference set $\mathcal{R}$ is of order the norm of the crossing texture component of $patch(i)$. We illustrate this limitation in our numerical experiments.

To overcome this flaw, we look at the set of reference patches as a set of clouds. Each cloud, say $\mathcal{C}_l$, is characterized by its center of mass, say m_l, and its covariance matrix, say C_l, that characterize the tangent space at m_l. For example one can partition the set of reference patches into K bins (or clouds) using any clustering method. The center of mass of bins can be interpreted as the main visual perceptions of the pattern of interests. They are often called "textons" [4] [3]. The projection of $patch(i)$ onto the tangent space at m_l is given by:

$$P_{C_l}(i) = m_l + \sum_{j=1}^{n_l} < patch(i) - m_l \, , \, u_{l,j} > \, u_{l,j} \tag{5}$$

where n_l is the number of principal eigenvalues of C_l and $u_{l,1}, \ldots u_{l,n_l}$ are the associated normalized eigenfunctions. The affinity matrix defined in (2) has to be modified. It becomes an affinity between any patch to any center of mass m_l. The affinity is measured in the tangent space of $\mathcal{C}_l$:

$$a(i,l) = \frac{1}{Z(i)} \phi(i,l) e^{-\frac{||P_{C_l}(i) - m_l||_2^2}{\epsilon}} \tag{6}$$

where $Z(i)$ is such that $\sum_{l=1}^{K} a(i,l) = 1$. The coefficient $\phi(i,l)$ is here only to ensure the operator is define locally. In other words, one could define $\phi(i,l)$ by:

$$\phi(i,l) = e^{-||patch(i) - m_l||^2/\eta} \tag{7}$$

Other choices are possible (e.g. compute $a(i,l)$ for k nearest neighbors). Similarly to the non local filter operator, one can threshold the coefficients $a(i,l)$ or the kernel density $Z(i)$ whenever one of these quantities is insignificant. We then propose the following projected non local filter operator:

$$PNLF(i) = \sum_{l=1}^{K} a(i,l) \, P_{C_l}(i). \tag{8}$$

where the coefficients $a(i,l)$ are given by (6). Notice that $PNLF(i)$ is a patch. To build the filtered image one can associate to each pixel i the coordinate of $PNLF(i)$ at the center of the patch.

There are two main advantages of using $PNLF$ instead of NLF. First, instead of using all the patches coming from $\mathcal{R}$ as a sub-dictionary, $PNLF$ learns a condensed dictionary of textons. This reduces considerably the complexity of the problem. Second, by defining the affinity kernel by means of the projection, we are able to resolve intersections: the projection kills the orthogonal (with respect to the tangent space) component of any patch. Numerical results confirm our intuition.

To conclude this section, we would like to emphasize the similarity between our approach and the one presented in [7]. While we are considering the projection of a patch onto a tangent space, the latter considers the parallel transport of a patch onto a tangent space.

3 Graph-based Diffusion Filter

In this section we provide a framework based upon diffusion processes for filtering a natural image given a pattern of interest. We still consider the reference set $\mathcal{R}$ which consists of pixels whose patches correspond to a given pattern. The similarity weights defined previously are organized in a similarity matrix A which is either $A = (a(i,j))_{i \in I, \, j \in \mathcal{R}}$ in the case of NLF or $A = (a(i,j))_{i \in I, \, j \in \{1 \ldots K\}}$ where $a(i,j)$ is given by either (2) or (6) in the case of $PNLF$. Without loss of generality we consider the first case. In the second case the set $\mathcal{R}$ is replaced by the set $\{1, \ldots, K\}$. We now look at the set of pixels I as a graph.

The nodes are the pixels. The similarity between any two pixels i and j, say $w(i,j)$, is defined relative to the reference set; the similarity is strong if these pixels have lots of common neighbors in $\mathcal{R}$. We then naturally define $w(i,j)$ by $W_I = AA^T = (w(i,j))_{i,j\in I}$, i.e.,

$$w(i,j) = \sum_{r\in\mathcal{R}} a(i,r)a(j,r)\,. \tag{9}$$

The kernel W_I we have built is relative to the reference set. It defines the local geometries of the graph. Many works propose to use the first few eigenvectors of the Laplace-Beltrami operator for clustering purposes. For further details, the reader is referred to the fundamental paper of S. Lafon and R. Coifman [5] and J. Shi and J. Malik [9]. The construction of the kernel W_I implies that the eigenfunctions of the Laplace-Beltrami operator are supported on the region of the image that is similar to the pattern of interest.

The way W_I is defined allows us the following result:

Proposition 3.1 *The eigenfunctions of $W_I = AA^T$, say ψ_j, corresponding to nonzero eigenvalues $\lambda_j > 0$ are*

$$\psi_j = \frac{1}{\lambda_j^{1/2}} A\phi_j$$

where ϕ_j are the eigenfunctions of $W_{\mathcal{R}} = A^TA$ corresponding to the eigenvalues λ_j. The denominator $\lambda_j^{1/2}$ is a normalization term; $||\psi_j||_2 = ||\phi_j||_2$. Moreover, the sets $\{\phi_j\}$ and $\{\psi_j\}$ are orthogonal.

The main advantage of considering $W_{\mathcal{R}} = (w_{\mathcal{R}}(i,j))$ instead of W_I is that in practice the size of $W_{\mathcal{R}}$ is much smaller than the size of W_I. This provides us with a fast way of computing the eigenfunctions of W_I. However, we are interested in the eigenfunctions of the Laplace-Beltrami operator. The proposition needs to be slightly modified. Here are the details.

In order to get rid of the non-uniform sampling, the Laplace-Beltrami operator is preferred to the Markov operator. We first estimate the diagonal density matrix D_1 which i^{th} diagonal term is given by

$$d_1(i) = \sum_{j\in\mathcal{R}} w_{\mathcal{R}}(i,j)$$

where $i \in \mathcal{R}$. The kernel $W_{\mathcal{R}}$ is then normalized by the density as follows:

$$W_1 = D_1^{-1} W_{\mathcal{R}} D_1^{-1} \tag{10}$$

where W_1 is the normalized kernel. Its generic term is given by $w_1(i,j) = \frac{w_{\mathcal{R}}(i,j)}{d_1(i)d_1(j)}$. The kernel W_1 is made row stochastic by considering $W_2 = D_2^{-1}W_1$ where D_2 is a diagonal matrix whose diagonal term

$$d_2(i) = \sum_{j\in\mathcal{R}} w_1(i,j).$$

Its eigenvalues - all non negative and bounded by 1- are sorted in decreasing order with $\lambda_1 = 1$. The first eigenfunction $\phi_1 = \mathbf{1}$, a column vector of just ones. The Laplace-Beltrami operator is given by $I - W_2$. It is convenient to consider the kernel W_2 since it shares the same eigenfunctions as the Laplace-Beltrami operator. The kernel W_2 is adjoint to a symmetric kernel

$$\tilde{W}_2 = D_2^{-1/2} W_1 D_2^{-1/2}.$$

They share the same eigenvalues λ_j. The eigenfunctions of W_2, denoted ϕ_j - **that we assume of unit norm** - are related to those of $\tilde{W}_2$, denoted $\tilde{\phi}_j$ according to

$$D_2^{1/2}\phi_j = \tilde{\phi}_j. \tag{11}$$

This implies $\tilde{W}_2 = \tilde{A}^T\tilde{A}$, where $\tilde{A} = (\frac{a(i,j)}{d_1(j)d_2(j)^{1/2}})_{i\in I,\, j\in\mathcal{R}}$. In other words, the similarity weights $a(i,j)$ are normalized by the factor $d_1(j)d_2^{1/2}(j)$. We are using $\tilde{A}$ instead of A to compute the Laplace-Beltrami eigenfunctions; the similarity weights between the pixels in I are given by $\tilde{A}\tilde{A}^T$. The set $\left\{\tilde{\phi}_j\right\}$ is orthonormal in $l^2(\mathcal{R},\, dx)$ and the set $\{\phi_j\}$ is orthonormal in $l^2(\mathcal{R},\, d_2(x)dx)$. Using proposition 3.1 and equation 11 we get:

$$\tilde{\psi}_j = \frac{1}{\lambda_j^{1/2}}\tilde{A}\tilde{\phi}_j = \frac{1}{\lambda_j^{1/2}}\tilde{A}D_2^{1/2}\phi_j \tag{12}$$

where $\tilde{\psi}_j$ are the eigenfunctions of $\tilde{A}\tilde{A}^T$ corresponding to the eigenvalue $\lambda_j > 0$. The set $\left\{\tilde{\psi}_j\right\}$ is orthonormal. We normalize the kernel $\tilde{A}D_2^{1/2}$ so that the trivial eigenfunction $\phi_1 = \mathbf{1} \in \mathbb{R}^{|\mathcal{R}|}$ is associated to the trivial eigenfunction $\mathbf{1} \in \mathbb{R}^{|I|}$. To do so, we define the diagonal matrix D whose diagonal term $d(i) = \sum_{j\in\mathcal{R}}(\tilde{A}D_2^{1/2})(i,j) = \sum_{j\in\mathcal{R}}\frac{a(i,j)}{d_1(j)}$. Let $A_1 = D^{-1}\tilde{A}D_2^{1/2} = D^{-1}AD_1^{-1}$. The matrix A_1 is row stochastic. The eigenfunctions ψ_j are then given by

$$\psi_j = \frac{1}{\lambda_j^{1/2}}A_1\phi_j \tag{13}$$

They form an orthonormal family in $l^2(I,\, d^2(x)dx)$. To summarize, We built a row stochastic kernel $A_1 = (\frac{a(i,j)}{d(i)d_1(j)})$ from $l^2(\mathcal{R},\, d_2(x)dx)$ to $l^2(I,\, d^2(x)dx)$ s.t. $\psi_j = \frac{1}{\lambda_j^{1/2}}A_1\phi_j$; it maps the orthonormal family $\{\phi_j\}$ onto the orthonormal family $\{\psi_j\}$. As a matter of fact, relation (13) allows us to interpret ψ_j as an extension of ϕ_j. While the kernel A_1 is an averaging operator similar to Nyström interpolation, the coefficient $1/\lambda_j^{1/2} \geq 1$ counter-balances the averaging effect by amplifying the magnitude. Subsection 4.1 illustrates this point while [2] shows how to use this general framework in the context of extendable Independent Component Analysis.

The orthogonality gives us an easy way to project onto any vectorial space spanned by some eigenfunctions. In practice, we consider the eigenfunctions whose eigenvalues λ_j are above a certain threshold. Consider the space spanned by the first L eigenfunctions, $\psi_1, \ldots, \psi_L$. Given a function $f \in l^2(I,\, d^2(x)dx)$, its projection is given by

$$P(f) = \sum_{j=1}^{L}\alpha_j\psi_j, \tag{14}$$

where $\alpha_j = \sum_{i\in I} f(i)\psi_j(i)\, d^2(i)$, corresponding to the inner-product between f and ψ_j in $l^2(I,\, d^2(x)dx)$. Subsection 4.2 compares the graph-based diffusion filter to the non local filters (NLF and PNLF).

4 Numerical Results

4.1 Filtering on Smooth Manifolds

Our technique for filtering patterns in an image by using a reference set can be generalized to extending functions on the manifold. In particular, we demonstrate the extension of the manifold of images, by using a reference set. It is shown below that a reference set that captures the essential degrees of freedom in an image database can be used to extend the manifold of images. This idea is discussed in detail in [2] in the context of extendable Independent Component Analysis. We demonstrate this concept via a simple example involving images of size 100 x 100 pixels of rotating earth. The images contain earth image rotated with respect to angles $\theta = -45$ to $\theta = 45$ in longitude and $\sigma = -30$ to $\sigma = 60$ in latitude. Each image is represented in a vector of dimension 100^2. The data set contains 900 images out of which we sample uniformly 450 images as a reference set.
We performed the following steps: we reduced the data dimension to $\mathbb{R}^{20}$ via random projections. We then constructed the affinity matrix A between the reference set and the rest of the data with $\epsilon = 0.1$, where the patches correspond to column vectors of the projected images in $\mathbb{R}^{20}$. Finally, we computed the embedding of the reference set and its extension. In Figure 1 we show a sample of our image data. In Figure 2 we show the embedding of the reference set into the eigenspace spanned by $\Phi = [\phi_1; \phi_2]$, and the embedding of the full data set using the extension to $\Psi = [\psi_1; \psi_2]$. The embedding demonstrates the invariant similarity between earth images with similar rotation angles that is captured by the eigenvectors. In particular, the extension Ψ is shown to preserve this similarity.

4.2 Few examples

In this subsection we present some numerical results for the non local filters and the graph-based diffusion filter as well. We test our methods on two specific images. On one hand we use the well-known image

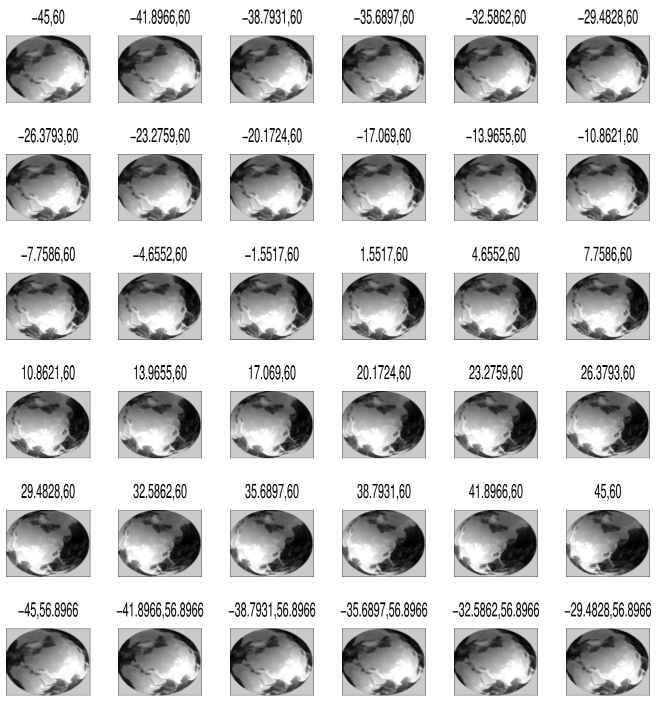

Figure 1: A sample of Earth images for $\theta = -45$ to $\theta = 45$, and $\sigma = 60$ to $\sigma = 56.89$.

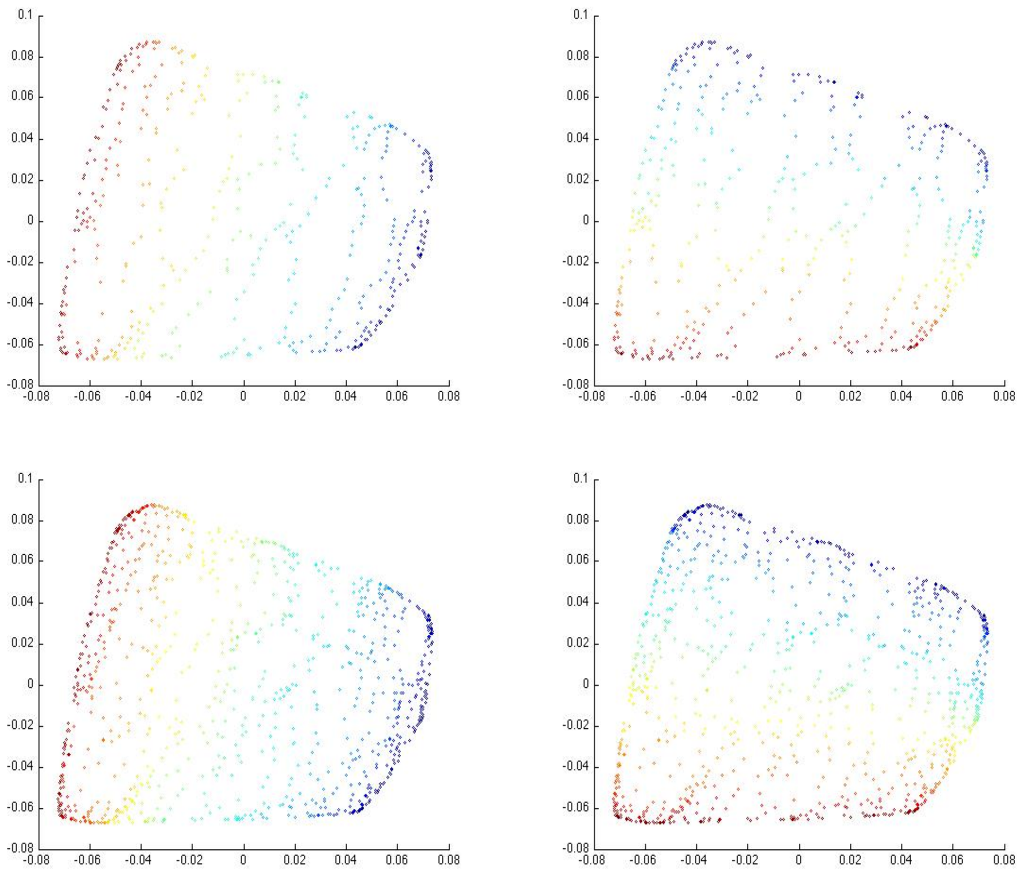

Figure 2: Upper row: data embedding into (Φ_1, Φ_2). Bottom row: data embedding into (Ψ_1, Ψ_2) computed via the extension. Color on left sub-figures is according to θ, and on the right sub-figures according to σ

of “Barbara” to extract the scarf. On the other hand, we use an image coming from seismic data. It corresponds to different layers in the ground and we are interested in extracting one specific layer, e.g. the horizontal or diagonal layer.

The image of “Barbara” contains a few different patterns, e.g. the scarf, the chair or the knee. We focus on the pattern that defines the scarf. It is a one-dimensional periodic pattern. The patches are extracted from the grayscale image that was rescaled such that its values are in the interval $[0, 1]$. The size of the patches is fixed to 5×5. The kernel A is computed using $\epsilon = 0.1$. We select a small region inside the scarf that corresponds to the reference set. This is shown by the highlighted polygon in Figure 3. For the Non Local Filter algorithm, the kernel A is thresholded as explained in section 2; if the kernel density $Z(i)$ is less than δ, then $a(i, \cdot) = 0$. Here we set manually $\delta = 0.008$. In the graph-based diffusion filter , the thresholding is done at the very last step; if $d(i) < \delta$ then $A_1(i, \cdot) = 0$. We choose $\delta = 10^{-5}$. The projection is done using 8 eigenfunctions. The reference set contains about 400 pixels. In both methods, the scarf is almost completely detected; only the region located below the right hand is missed. This is mainly due to the change of contrast in this region of the image. Both methods detect also some part of the knee and the chair. This is expected since both objects contain regions corresponding to the selected pattern. The results are shown in Figure 3. To improve the detection of the scarf, we slightly modify our data. The rescaled image is normalized as follows. The value at each pixel is normalized by the norm of its corresponding patch. The same tests are then performed using different parameters; $\epsilon = 0.01$, $\delta = 0.05$ for the non local filter, and $\delta = 10^{-4}$ for the graph-based diffusion filter. Figure 4 shows that the little part of the scarf below the right hand is now detected. The results in Figure 4 are better than those from Figure 3 because the normalization got rid of the change of contrast along the scarf. If we decrease δ, Figure 5 shows that more of the knee and the chair are seen. The reference set contains about 300 pixels. These results could have been obtained using the second operator we proposed; the projected non local filter. This is illustrated in the left image of Figure 6. In this experiment, we cluster (using k-means) the reference set into 100 bins. For each bin, we compute the center of mass and the projection of any patches. The patch size is fixed to 5×5. The parameters are $\eta = 0.02$ and $\epsilon = 1/3$. The coefficients $a(i, l)$ given by relation (6) are set to 0 whenever $Z(i) < 10^{-5}$. The image obtained in the right of Figure 6 was obtained using patches of size 17×17. The threshold parameter was set to 0. One clearly see from this image the pattern we selected while the rest of the image is smooth. The image has been decomposed into two components; one corresponding to the pattern of interest (the one provided by the scarf) and the other component corresponding to a sketch of the image; the bigger the patch size is the smoother the sketch is. It is easier to extract the pattern once the simple part (the sketch) of the image is removed; textures are then seen as zero-mean oscillatory periodic functions.
The second example we are considering is an image provided by seismic data. It is made of three different layers; horizontal, almost vertical and diagonal. We are interested in extracting the horizontal one. We rescale the image to fit between 0 and 1. We then extract the patches of size 5×5 and define the reference set $\mathcal{R}$ by manually selecting a region within the polygon highlighted in Figure 7. This set contains about 1000 pixels. The ϵ for the non local filter and the graph-based diffusion filter is set to 0.01. The kernel A is computed as in relation (2). Results displayed in Figure 7 are similar. Both methods cannot detect the horizontal layer at the intersection of the diagonal and the horizontal layers. The number of eigenfunctions used for the projection is set to 8. Increasing this number does not change significantly the results. We overcome this problem by modifying the weights $a(i, j)$ and by increasing the size of the patches. Increasing the size of the patches while keeping $a(i, j)$ as in relation 2 does not resolve the intersection; the patches are almost orthogonal since high dimensional. We consider the projected non local filter operator. We set the patch size to 31×61. As explained at the end of section 2, the projections of any patch centered at any pixels of the intersection onto the K clusters of the reference set are significant. We choose $K = 25$. Each projection is computed using the top 8 eigenfunctions. The result is displayed on the left panel of Figure 9. The right panel is the graph-based diffusion projection onto the top 8 eigenfunctions as explained in section 3 where the kernel A is chosen as in relation 6. In both tests, the horizontal layer is detected across the intersection. Figure 10 displays a sample of 8 textons (each of them corresponding to a center of mass of a cluster).

5 Conclusion

We proposed different patch-based filters for extracting a specific pattern out of an image. Once the reference set is chosen by the user, the only parameters that need to be tuned are the threshold δ and the

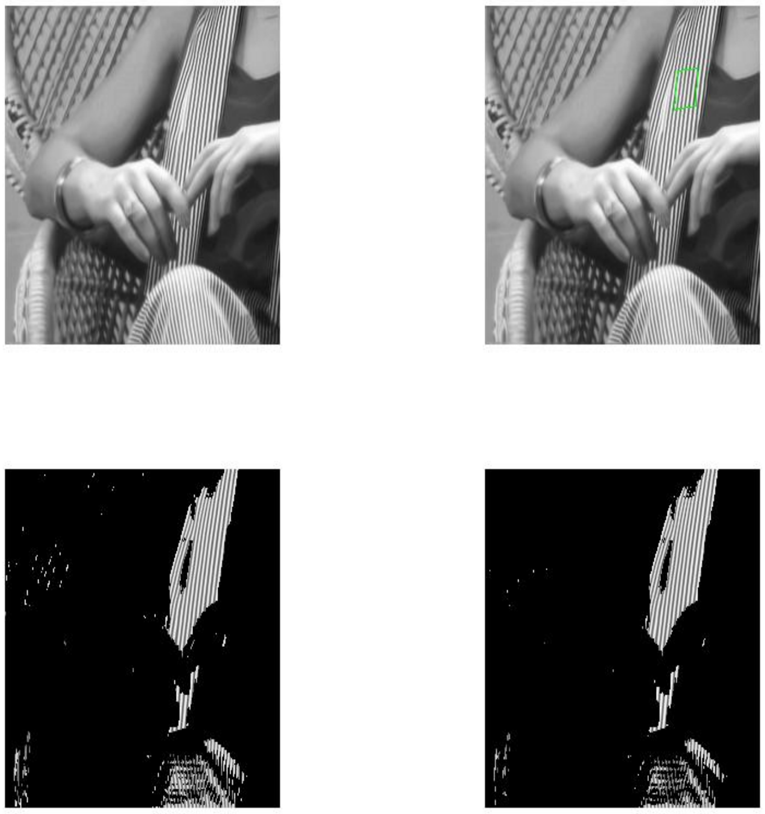

Figure 3: From left to right, Top to bottom: Original image. Reference set. Non Local Filter with $\delta = 0.008$. Graph-based Diffusion Filter with 8 eigenfunctions, $\epsilon = 0.1$, $\delta = 10^{-5}$.

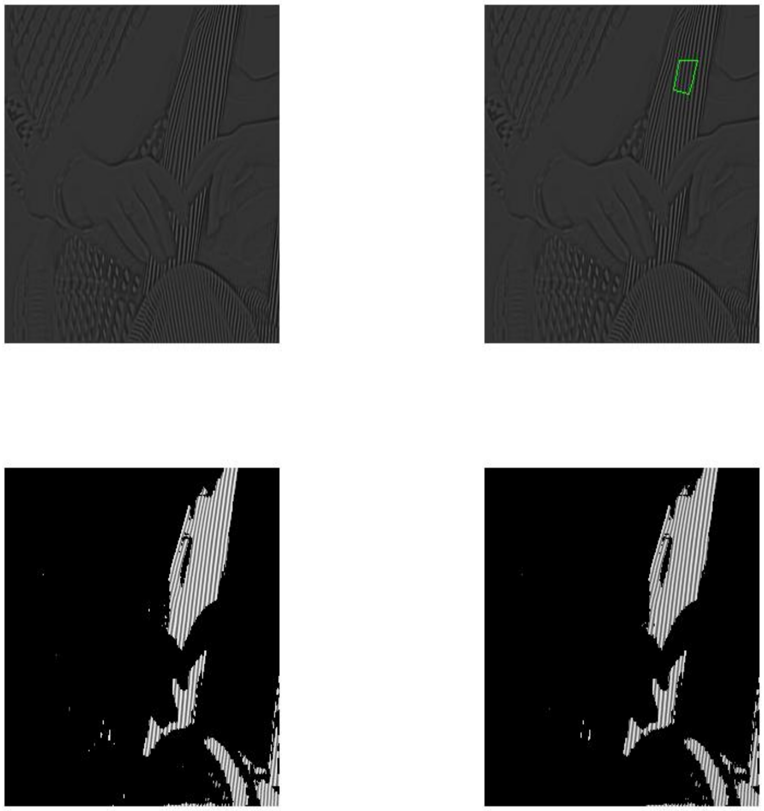

Figure 4: From left to right, Top to bottom: Normalized image used for patches. Reference set. Non Local Filter with $\delta = 0.05$. Graph-based diffusion Filter with 8 eigenfunctions, $\epsilon = 0.01$ and $\delta = 10^{-4}$.

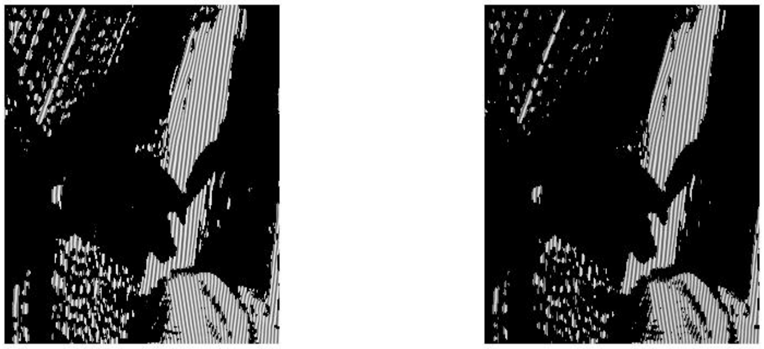

Figure 5: From left to right. Non Local Filter with $\delta = 0.001$. Graph-based Diffusion Filter with 8 eigenfunctions, $\epsilon = 0.01$ and $\delta = 3\,10^{-6}$.

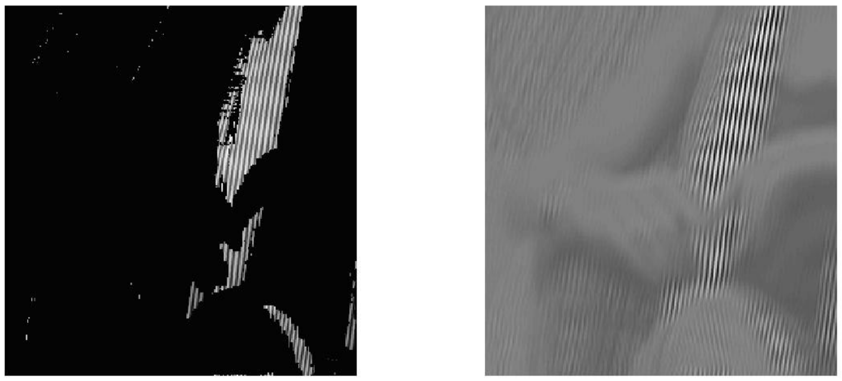

Figure 6: left: Projected non local filter. Patch 5×5, $\delta = 10^{-5}$, $\epsilon = 0.333$, $\eta = 0.02$. Right: No thresholding. Patch of size 17×17.

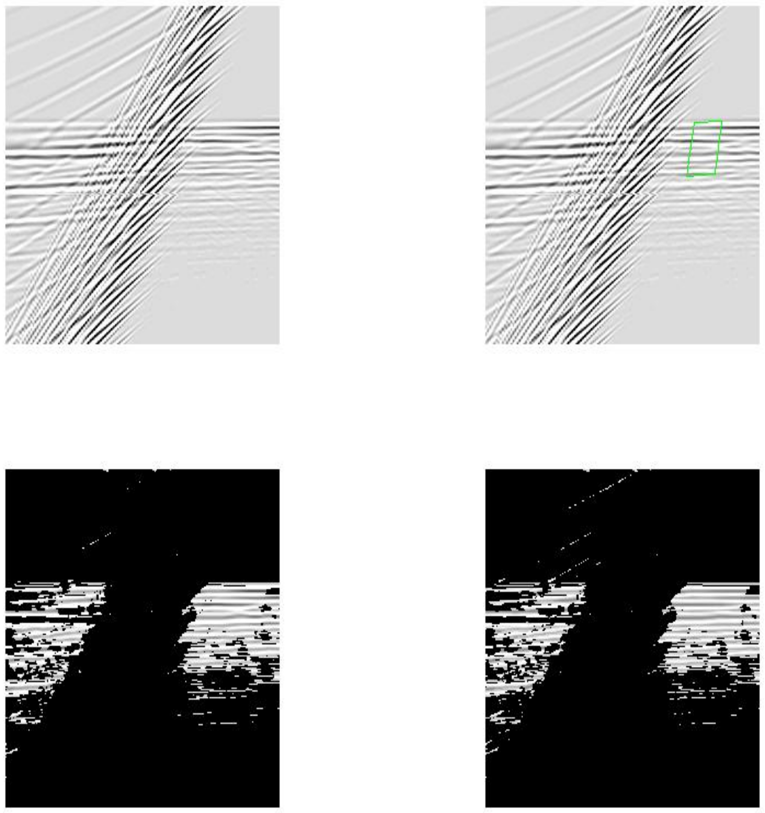

Figure 7: From left to right, Top to bottom: Original image. Reference set. non local filter with $\delta = 0.001$. Graph-based Diffusion Filter with 8 eigenfunctions, $\epsilon = 0.01$, $\delta = 10^{-5}$.

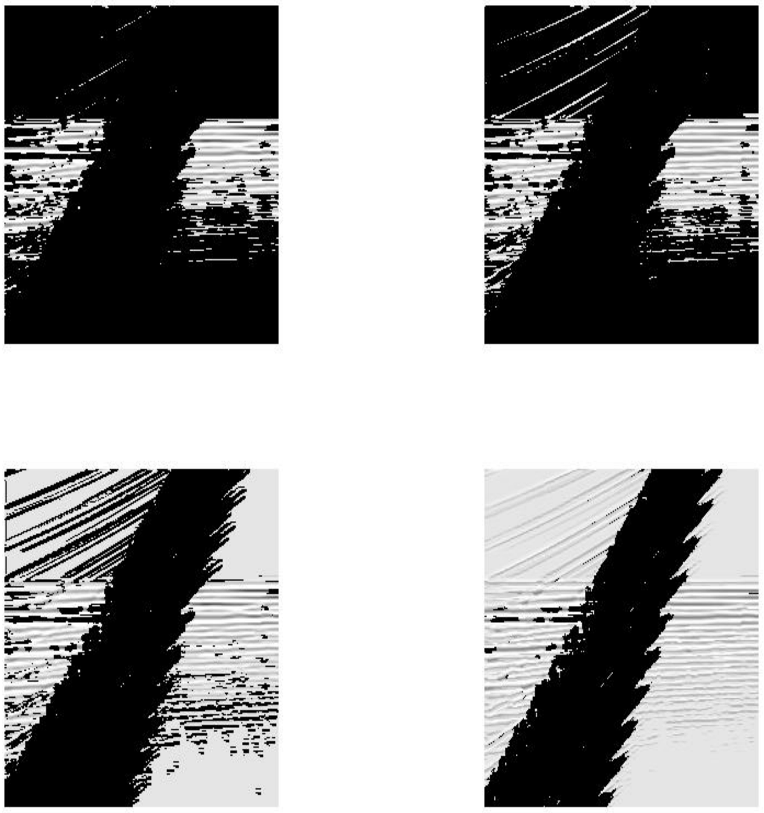

Figure 8: From left to right, top to bottom. non local filter with $\delta = 510^{-4}$. Graph-based Diffusion Filter with 40 eigenfunctions, $\epsilon = 0.01$ and $\delta = 10^{-3}$. Non Local Filter with $\delta = 10^{-5}$. Graph-based Diffusion Filter with 8 eigenfunctions, $\epsilon = 0.01$ and $\delta = 10^{-5}$.

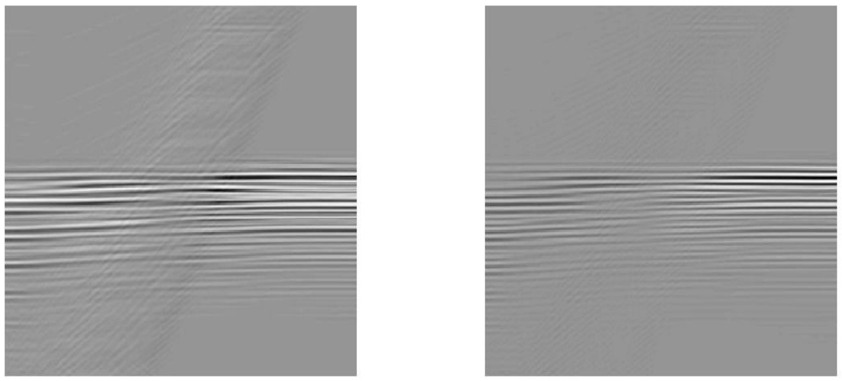

Figure 9: From left to right: Projected Non Local Filter with K=25, $n_l = 8$. Graph-based Diffusion Filter with 8 eigenfunctions.

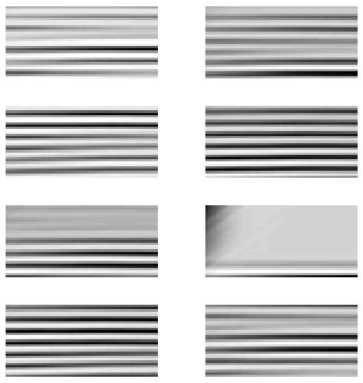
Figure 10: Textons using K=8.

projection parameters (number of clusters, number of principal components, number of eigenfunctions). The other parameters can be tuned as explained in section 4. We showed that considering patches as a characterization of local patterns and defining the similarity between patches based on the Euclidian norm allows us to build filters that perform well when textures are not overlapping. In addition, re-normalizing patches provides us with a much better similarity weight; it gets rid of the contrast changes. We also proposed a projected non local filter algorithm that aims at resolving intersections. This model is based on projections of patches onto tangent spaces at reference points (or center of mass of clusters). This suggests to replace patches by some local statistics. For example, one could consider local covariance matrices of patches as feature vectors and measure distances using the Fröbenius norm. Finally, these techniques extend easily to hyper-spectral imaging. Pixels correspond to points living in a high dimensional space. Patches are replaced by the bands representing the hyper-spectral data.

References

[1] A. Buydes, B. Cole, J.M. Morel. A review of denoising algorithms, with a new one. SIAM Multiscale Modeling and Simulation, vol 4 (2) pp:490-530, 2005.

[2] D. Kushnir, A. Haddad, R. R. Coifman, Anisotropic Diffusion on Sub-Manifolds with Application to Earth Structure Classification, submitted to Applied and Computational Harmonic Analysis (ACHA)

[3] Jitendra Malik, Serge Belongie, Jianbo Shi, Thomas K. Leung. Textons, Contours and Regions: Cue Integration in Image Segmentation. ICCV 1999, 918-925.

[4] B. Julesz. Textons, the elements of texture perception, and their interactions. Nature, 290(5802):917, March 1981.

[5] S. Lafon, R. Coifman. Diffusion Maps. Applied and computational harmonic analysis, vol 21 iss 1, 2006.

[6] P. Perona, J. Malik. Scale space and edge detection using anisotropic diffusion. IEEE Trans. Patt. Anal. Mach. Intell., 12 (1990), pp 629-639.

[7] A. Singer, H-T. WU. Vector Diffusion Maps and The Connection Laplacian. (preprint)

[8] A. Szlam, M. Magggioni, R.R. Coifman. Regularization on graphs with function-adapted diffusion processes. Journal of Machine Learning Research 9. 1711-1739, 2008.

[9] J. Shi, J. Malik. Normalized cuts and Image Segmentation. IEE Trans. Patt. Anal. Mach. Intell., vol 22, No. 8, August 2000.