# Adaptive Kernel-based Image Denoising employing Semi-Parametric Regularization

Pantelis Bouboulis, Konstantinos Slavakis, *Member, IEEE,* and Sergios Theodoridis, *Fellow, IEEE*

***Abstract*—The main contribution of this paper is the development of a novel approach, based on the theory of Reproducing Kernel Hilbert Spaces (RKHS), for the problem of Noise Removal in the spatial domain. The proposed methodology has the advantage that it is able to remove any kind of additive noise (impulse, gaussian, uniform, e.t.c.) from any digital image, in contrast to the most commonly used denoising techniques, which are noise-dependent. The problem is cast as an optimization task in a RKHS, by taking advantage of the celebrated Representer Theorem in its semi-parametric formulation. The semi-parametric formulation, although known in theory, has so far found limited, to our knowledge, application. However, in the image denoising problem its use is dictated by the nature of the problem itself. The need for edge preservation naturally leads to such a modeling. Examples verify that in the presence of gaussian noise the proposed methodology performs well compared to wavelet based technics and outperforms them significantly in the presence of impulse or mixed noise.**

***Index Terms*—Kernel, Denoising, Reproducing Kernel Hilbert Spaces, semi-parametric representer theorem**

## I. Introduction

THE problem of noise removal from a digitized image is one of the most important ones in digital image processing. So far, various techniques have been proposed to deal with it. Among the most popular methodologies are, for example, the wavelet-based image denoising methods (which dominate the research in recent years, see for example [1], [2], [3], [4], [5]), the image denoising methods based on Partial Differential Equations ([6]), neighborhood filters ([7], [8]), some methods for impulse detection (see [9], [10], [11], [12]), methods based on fractal theory [13] and, more recently, methods of non linear modeling using kernel regression and/or local expansion approximation techniques ([14]). In many cases, the denoising techniques are focused on a particular noise model (gaussian, impulse, e.t.c.) [15], [2], [3], [5], [11], [10]. Thus, they cannot treat effectively more complex models, which are often met in practical applications. In this paper, we propose a different approach. We treat noise in a unified framework. Our only assumption is that the image is corrupted by zero mean additive noise, without any additional information with respect to the noise pdf. To remove the noise, we employ the well known (especially in pattern analysis) theory of kernels.

P. Bouboulis and S. Theodoridis are with the Department of Informatics and Telecommunications, University of Athens, Greece, e-mail: {bouboulis,stheodor}@di.uoa.gr.

K. Slavakis is with the Department of Telecommunications Science and Technology, University of Peloponnese, Tripolis, Greece, email: slavakis@uop.gr.

In kernel methodology, the notion of the Reproducing Kernel Hilbert Space (RKHS) plays a crucial role. A RKHS, introduced in [16], [17], [18], is a rich construct (roughly, a smooth space with an inner product), which has been proven to be a very powerful tool. Kernel based methods are utilized in an increasingly large number of scientific areas, especially where non-linear models are required. For example, in pattern analysis, a classification task of a set $\mathcal{X} \subset \mathbb{R}^m$ is usually reformed by mapping the data into a higher dimensional space (possibly of infinite dimension) $\mathcal{H}$, which is a Reproducing Kernel Hilbert Space (RKHS). The advantage of such a mapping is to make the task more tractable, by employing a linear classifier in the feature space $\mathcal{H}$, exploiting Cover's theorem (see [19], [20], [21], [22]). This is equivalent with solving a non-linear problem in the original space. Similar approaches have been used in principal components analysis, in Fisher's linear discriminant analysis, in clustering, regression and in many other subdisciplines (see [19], [20] for more). Recently, processing in RKHS is gaining in popularity within the Signal Processing community in the context of adaptive filtering and beam forming [23], [24], [25], [26].

Though there has been some work exploring the use of kernels in the denoising problem, the methodology presented here is fundamentally different. In [14], the notion of kernel regression has been adopted. The original image is formulated as a local (e.g. Taylor) approximation series around a center $\boldsymbol{x}_i$ and data adaptive kernels are used, as weighted factors, to penalize distances away from $\boldsymbol{x}_i$. In a relatively similar context, kernels have been employed by other well known denoising methods (such as [7]). In [27] and [28] a support vector regression approach is considered for the gaussian noise case and in [29] the kernel principal components of an image are extracted and this expansion is truncated to produce the denoising effect. However, in the latter approaches the reported results are rather poor.

In our case, the main idea is to assume that the original (noise-free) image lies in a RKHS and thus it can be formulated as a linear combination of specific kernel functions that justify the existence of a Reproducing Kernel Hilbert Space. Therefore, in our methods, the kernels are used in order to explicitly model the denoised image, in contrast to the SKR method of [14], where kernels are used as weighted factors to a Taylor approximation. In contrast, as already been stated, in our case, kernels can only be reproducing kernels (i.e., a kernel that generates a RKHS, not any kernel function). The methodology presented in this paper is derived from the discipline of machine learning (learning with kernels), rather than that of the non parametrical statistical estimation, where

kernel regression has its roots. The approach we have used, gives us the power to model and work in infinite dimensional Hilbert spaces, instead of a simple family of kernels. Hence, our approach lies closer to the support vector regression and kernel principal component analysis methods derived in [27], [30], than to the kernel regression employed in [14] and elsewhere. We exploit a useful property of RKHS, the so called *representer theorem*. It states that the minimizer of any optimization task in the Hilbert space $\mathcal{H}$, with a cost function of a certain type, has finite representation in $\mathcal{H}$. We recast the image denoising problem as an optimization task of this type and use a semi-parametric variant of the representer theorem to obtain its solution algorithmically. The semi-parametric part of the methodology is used to explicitly model, and thus preserve, the sharp edges of the image, which are not respected if only the kernel expansion is considered. This is done by learning edge models using a rich set of basis functions, similarly to the modeling approach taken by the K-SVD algorithm [31]. The denoising procedure is performed inside a pixel-centered region that moves from one pixel to the next and the parameters of the model are controlled adaptively at each region to preserve the fine details and local characteristics of the image. Note that at the optimization step the $L_1$ norm is employed as the cost function. This implies that the proposed method will give enhanced results when impulse or Laplace - distributed noises (with potentially spatially varying statistics) are considered. This is verified by the tests' results.

The paper is structured as follows. In Section II, we briefly describe the key mathematical preliminaries behind the notion of RKHS and state the representer theorem. In Section III, we present the kernelized approach to the image denoising problem. The framework, the details of the implementation as well as the algorithmic scheme can be found there. Experiments on images corrupted by various types of synthetic noise models (impulse, gaussian, uniform, mixed) are detailed in Section IV and Section V concludes the paper.

## II. Mathematical Preliminaries

### A. Reproducing Kernel Hilbert Spaces

We start with some basic definitions regarding the property of positive definite matrices and functions, which play a fundamental role in the study of RKHS. Throughout this paper, we apply the usual convention to use bold face letters for vectors and normal letters for scalars. Therefore $\boldsymbol{x}$, $\boldsymbol{y}$ denote vectors whilst $x$, $y$ denote scalars.

**Definition II.1.** *(Gram Matrix) Let $\mathcal{X}$ be a set. Given a function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{X}$, the matrix*[1] *$K = (K_{i,j})^N$ with elements $K_{i,j} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for $i, j = 1, \ldots, N$, is called the Gram matrix (or kernel matrix) of $\kappa$ with respect to $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$.*

**Definition II.2.** *(Positive Definite Matrix) A real symmetric matrix $K = (K_{i,j})^N$ satisfying*

$$\boldsymbol{v}^T \cdot K \cdot \boldsymbol{v} = \sum_{i=1,j=1}^{N,N} v_i v_j K_{i,j} \geq 0,$$

*for all $v_i \in \mathbb{R}$, $i = 1, \ldots, N$, is called Positive Definite. In matrix analysis literature, this is the definition of the positive semidefinite matrix, but since this is a rather cumbersome term and the distinction between positive definite and positive semidefinite matrices is not important in this paper, we employ the term positive definite in the way presented here. Furthermore, the term positive definite was introduced for the first time by Mercer in kernel context (see [32]).*

[1]The term $(K_{i,j})^N$ denotes a square $N \times N$ matrix.

**Definition II.3.** *(Positive Definite Kernel) Let $\mathcal{X}$ be a nonempty set. Then a function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which for all $N \in \mathbb{N}$ and all $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{X}$ gives rise to a positive definite Gram matrix $K$ is called a Positive Definite Kernel.*

In the following, we will frequently refer to a positive definite kernel simply as kernel. The reason that the kernels are so popular is that they can be regarded as a "generalized dot product". In fact, any dot product is a kernel (of course the opposite is not true). Several properties of dot products (such as the Cauchy-Schwartz inequality) do have natural generalizations to kernels (see [18], [33] and [20]).

Having dealt with the definitions of positivity, we are ready to move on and discuss the main issue of this section. Consider a Hilbert space $\mathcal{H}$ of real valued functions $f$ defined on a set $\mathcal{X}$, with a corresponding inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$. We will call $\mathcal{H}$ as a *Reproducing Kernel Hilbert Space* - RKHS, if there exists a kernel $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following two properties:

1) For every $\boldsymbol{x} \in \mathcal{X}$, $\kappa(\boldsymbol{x}, \cdot)$ belongs to $\mathcal{H}$.
2) $\kappa$ has the so called *reproducing property*, i.e.

$$f(\boldsymbol{x}) = \langle f, \kappa(\boldsymbol{x}, \cdot) \rangle_{\mathcal{H}}, \text{ for all } f \in \mathcal{H}, \tag{1}$$

in particular $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \langle \kappa(\boldsymbol{x}, \cdot), \kappa(\boldsymbol{y}, \cdot) \rangle_{\mathcal{H}}$.

It has been shown (see [17], [33]) that to every positive definite kernel $\kappa$ there corresponds one and only one class of functions $\mathcal{H}$ with a uniquely determined inner product in it, forming a Hilbert space and admitting $\kappa$ as a reproducing kernel. In fact, the kernel $\kappa$ produces the entire space $\mathcal{H}$, i.e.

$$\mathcal{H} = \overline{\text{span}\{\kappa(\boldsymbol{x}, \cdot) | \boldsymbol{x} \in \mathcal{X}\}},$$

where the overbar denotes the closure of the respective space. There are several kernels that are used in practice (see [20]). In this work, we focus on one of the most widely used, the Gaussian Kernel:

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|^2}{2\sigma^2}\right), \sigma > 0,$$

due to some additional properties that it admits.

One of the powerful properties of Kernel-theory is the introduction of non-linearity via a computationally elegant way known to the machine learning community as the *kernel trick* [20]:

> "Given an algorithm which is formulated in terms of dot products, one can construct an alternative algorithm by replacing each one of the dot products with a positive definite kernel $\kappa$."

The kernel trick is based on the use of the mapping $\Phi : \mathcal{X} \to \mathcal{H} : \Phi(\boldsymbol{x}) = \kappa(\boldsymbol{x}, \cdot)$, which maps any element of $\mathcal{X}$ to an element of $\mathcal{H}$. In addition this map has the interesting property:

$$\langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{y}) \rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}, \boldsymbol{y}). \tag{2}$$

Using the map $\Phi$, the kernel trick transforms a non linear problem defined on $\mathcal{X}$ to a linear one on the rich space $\mathcal{H}$. The next step is to solve the linear problem on $\mathcal{H}$ (usually this is an easier task), which, in turn, provides a non linear solution on $\mathcal{X}$.

Another powerful tool in kernel theory is the application of the representer theorem to *regularized risk minimization* problems (see [20] [19] and [34]):

**Theorem II.1** (Representer Theorem)**.** *Denote by $\Omega : [0,\infty) \to \mathbb{R}$ a strictly monotonic increasing function, by $\mathcal{X}$ a set and by $c : (\mathcal{X} \times \mathbb{R}^2)^N \to \mathbb{R} \cup \{\infty\}$ an arbitrary loss function. Then each minimizer $f \in \mathcal{H}$ of the regularized risk functional*

$$c\left((\boldsymbol{x}_1, z_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, z_N, f(\boldsymbol{x}_N))\right) \quad + \quad \Omega\left(\|f\|_{\mathcal{H}}\right) \tag{3}$$

*admits a representation of the form*

$$f(\boldsymbol{x}) = \sum_{n=1}^{N} \alpha_n \kappa(\boldsymbol{x}_n, \boldsymbol{x}). \tag{4}$$

In regression and classification tasks, $c$ often admits the form

$$c\left((\boldsymbol{x}_1, z_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, z_N, f(\boldsymbol{x}_N))\right) = \sum_{n=1}^{N} \mathcal{L}\left(z_n - f(\boldsymbol{x}_n)\right),$$

where $\mathcal{L}$ is a suitable loss function, $z_n$ are the actual values of the (possibly corrupted) signal at $\boldsymbol{x}_n$ and $f(\boldsymbol{x}_n)$ are the reconstructed values. Usually the regularization term $\Omega(f)$ takes the form $\Omega(f) = \frac{1}{2}\|f\|^2_{\mathcal{H}}$. In the case of a RKHS produced by the gaussian Kernel (which implies an infinite dimensional space [20]) it can be shown that[2]

$$\|f\|_{\mathcal{H}} = \int_{\mathcal{X}} \sum_{n} \frac{\sigma^{2n}}{n! 2^n} (O^n f(\boldsymbol{x}))^2 d\boldsymbol{x}, \tag{5}$$

with $O^{2n} = \Delta^n$ and $O^{2n+1} = \nabla\Delta^n$, $\Delta$ being the Laplacian and $\nabla$ the gradient operator (see [35]). The implication of this is that the regularization term "penalizes" the derivatives of the minimizer. This results to a very smooth solution of the regularized risk minimization problem. In fact, this penalization occurs in a more influential fashion than the *total variation* scheme, which is often used in wavelet-based denoising (see for example [36], [37], [38], [39], [40]). Indeed, while the total variation penalizes only the first order derivatives, the term $\|f\|^2_{\mathcal{H}}$ penalizes derivatives of any order, resulting in very smooth estimates.

The representer theorem plays a central role in solving practical problems of statistical estimation. Its significance is clear. Although we are solving an optimization problem and we search for an estimate of a function $f$, in a rich space $\mathcal{H}$ (possibly infinite-dimensional), the optimal solution lies in the span of a *finite* number of particular kernels; i.e., those centered on the training points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$. In addition, it has been found that for suitable choices of loss functions many of the coefficients $\alpha_n$ in (4) are often equal to 0. That is, the solution can be sparse, which is in line with our desire to guard against overfitting [19]. In the *Support Vector Machines* literature equation (4) is called the *support vector expansion*. This theorem can be generalized by the addition of some real valued functions (which may indicate some additional a priori knowledge of the problem), as follows:

**Theorem II.2** (Semi-parametric Representer Theorem)**.** *Suppose that, in addition to the assumptions of the previous theorem, we are given $\Omega_2 : [0,\infty) \to \mathbb{R}$ another strictly monotonic increasing function and a set of $M$ real-valued functions $\{\psi_k\}_{k=1}^{M} : \mathcal{X} \to \mathbb{R}$, with the property that the $N \times M$ matrix $(\psi_p(\boldsymbol{x}_n))_{n,p}$ has rank M. Then any $\tilde{f} := f + \psi$, with $f \in \mathcal{H}$ and $\psi \in \mathfrak{H} = \overline{span\{\psi_k\}}$, where $\|\cdot\|$ is a norm defined in $\mathfrak{H}$, minimizing the regularized risk functional*

$$c\left((\boldsymbol{x}_1, z_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, z_N, f(\boldsymbol{x}_N))\right) \quad + \quad \Omega\left(\|f\|_{\mathcal{H}}\right) \quad + \quad \Omega_2\left(\|\psi\|\right) \tag{6}$$

*admits a representation of the form (e.g. [20])*

$$\tilde{f}(\boldsymbol{x}) = \sum_{n=1}^{N} \alpha_n \kappa(\boldsymbol{x}_n, \boldsymbol{x}) + \sum_{k=1}^{M} \beta_k \psi_k(\boldsymbol{x}). \tag{7}$$

## III. Application of RKHS theory to the denoising problem

As it is usually the case, we model the noisy image as

$$\hat{f}(x,y) = f(x,y) + \eta(x,y), \tag{8}$$

for $x, y \in [0,1]$, where $f$ is the input image and $\eta$ the additive noise [41]. Given $\hat{f}$, the objective of any denoising methodology is to obtain an estimate of the original image. Usually, this is carried out by exploiting some extra knowledge about the noise term. In contrast, our method needs no additional information with respect to the pdf of $\eta$.

### A. Problem formulation in RKHS

Let $f_{i,j}$ and $\hat{f}_{i,j}$ be the restrictions of $f$ and $\hat{f}$ on the $N \times N$ orthogonal region centered at the pixel $(i,j)$ of each image accordingly ($N$ is an odd number, in order to have a central pixel, see figure 1). Our task is to estimate $f_{i,j}$, given the samples of $\hat{f}_{i,j}$. For simplicity, we drop the $i,j$ indices and consider $f_{i,j}$ and $\hat{f}_{i,j}$ (which from now on will be written as $f$ and $\hat{f}$) as functions defined on $[0,1] \times [0,1]$ (and zero elsewhere). The pixel values of the digitized image are given by $f(x_n, y_m)$ and $z_{n,m} = \hat{f}(x_n, y_m)$, where $x_n = n/(N-1)$, $y_m = m/(N-1)$ for $n, m = 0, 1, \ldots, N-1$. (Note that from this point on, we drop the $\boldsymbol{x}_n$ notation and use, instead, the notation $(x_n, y_m)$, to account for each pixel.)

The idea is to consider our image $f$ as a function in a RKHS $\mathcal{H}$. We assume that the RKHS $\mathcal{H}$ is generated by the Gaussian kernel:

$$\kappa\left((x,y),(x',y')\right) = exp\left(-\frac{|x-x'|^2 + |y-y'|^2}{2\sigma^2}\right),$$

for $\sigma > 0$. Then to obtain $f$ we may solve the regularized risk minimization problem:

$$\underset{f \in \mathcal{H}, h \in \mathbb{R}}{\text{minimize}} \;\; c(f,h) = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} |f(x_n, y_m) + h - z_{n,m}| \quad + \quad \frac{\lambda}{2}\|f\|^2_{\mathcal{H}}. \tag{9}$$

[2] In the cases where $\mathcal{X} = \mathbb{R}^m$, $m > 0$.

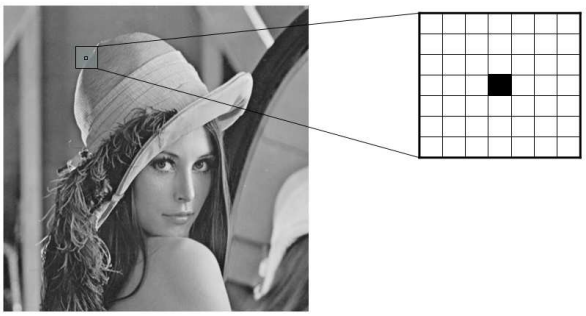


Fig. 1. An orthogonal $N \times N$ region centered at a pixel.

This is reasonable, since we want the denoised image to be smooth (recall that usually a RKHS is comprised of smooth functions). Note, that a threshold $h$ has explicitly been used, as it is commonly used in the support vector regression (SVR) rationale. It turns out that this is important in order to counteract the effect of the regularizer, which also affects the leveling of the solution (i.e. the regularizer penalizes the values of the function and its derivatives, see (5)). To solve this problem, we use the celebrated representer theorem (theorem II.1), which, now, ensures that the minimizer $\tilde{f}$ will have a finite representation in $\mathcal{H} + R$ (where $R = \{g : \mathbb{R}^2 \to \mathbb{R} : g(x, y) = h, \text{ for } h \in \mathbb{R}\}$), i.e.

$$\tilde{f}_{\min}(\cdot, \cdot) = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \alpha_{n,m} \kappa\left((x_n, y_m), (\cdot, \cdot)\right) + h, \qquad (10)$$

where the notation $\tilde{f}_{\min}(\cdot, \cdot)$ denotes that $\tilde{f}_{\min}$ is a function of two variables (the same is true for $\kappa\left((x_n, y_m), (\cdot, \cdot)\right)$). Note that in (9) the cost function used is the $l_1$ norm. This has a two fold advantage. It guards against outliers and also, it is in line with our desire to obtain as sparse solutions as possible, as this is well documented in compressed sensing literature [42].

Having stated the problem, our goal now becomes to estimate the values of the parameters, $\alpha_{n,m}$, $n, m = 0, \ldots, N-1$, $h$ in (10). To this end, (10) is substituted in (9) and the respective optimization has to be carried out. However, note that the cost function, defined by the $l_1$ norm, is not differentiable. Hence the notion of the subgradient (see appendix A) has to be mobilized. In this paper the well known Polyak's Projected Subgradient Method (see [43]) has been employed. Polyak's algorithm solves for the optimal value of $\boldsymbol{x}$ iteratively and it can be summarized in the following recursion:

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \gamma_n \cdot \frac{\nabla c(\boldsymbol{x}_n)}{\|\nabla c(\boldsymbol{x}_n)\|}, \qquad (11)$$

where $c(\boldsymbol{x})$ is the cost function of the minimization problem, $\gamma_n$ is an arbitrary sequence such that $\sum_{n=0}^{\infty} \gamma_n = \infty$, $\sum_{n=0}^{\infty} \gamma_n^2 < \infty$ and $\nabla c(x)$ is any subgradient of $c$ at $\boldsymbol{x}$. To implement the algorithm in the case of (9), we need to compute any of the subgradients $\nabla c(f, h)$. Taking into account that $f(x, y) = \langle f, \kappa\left((x, y), (\cdot, \cdot)\right)\rangle_{\mathcal{H}}$, we can deduce (after some algebra) that a suitable choice is:

$$\nabla c(f, h) = \begin{pmatrix} \nabla_f c(f, h) \\ \nabla_h c(f, h) \end{pmatrix}, \qquad (12)$$

where $\nabla_f c(f, h)$ and $\nabla_h c(f, h)$ are defined as follows (see Appendix A):

$$\nabla c_f(f, h) = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \operatorname{sign}\left(f(x_n, y_m) + h - z_{n,m}\right) \cdot \cdot \kappa\left((x_n, y_m), (\cdot, \cdot)\right) + \lambda \cdot f, \qquad (13)$$

$$\nabla c_h(f, h) = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \operatorname{sign}\left(f(x_n, y_m) + h - z_{n,m}\right). \qquad (14)$$

Under the above formulation, the proposed denoising algorithm can be summarized in the following three steps:

- For each pixel $(i, j)$ do:
  - Form the $N \times N$ "pixel centered" region $\hat{f}$.
  - Solve the minimization problem (9) for that particular region.
  - Move to the next pixel.

Note that each pixel is assigned to $N^2$ different values (since it belongs to the each one of the $N^2$ regions of its neighboring pixels). The actual value that we assign to each pixel is the mean of these values.

Figure 2 shows the results obtained by the application of the previous algorithm on Lena. One can immediately see that the result of the denoising process is a blurry image. The noise has been removed successively, but in the process most of the fine details have been lost. The same problem can be observed in other kernel-based denoising approaches such as the one in [29] (see figure 4). This is where the semi-parametric representer theorem comes into the scene as a means for sparse modeling of regions with edges. Note that the same problem of over-smoothening was also observed in [14], when gaussian kernels were considered, albeit the two methodologies are fundamentally different. In our case over-smoothening results from the fact that RKHS are extremely smooth spaces and therefore not well suited to represent strong edges.

**Remark III.1.** *We have used the notation $\mathcal{H} + R$, in a rather "naive" way. In Appendix B, a more rigorous elaboration is provided.*

(a) (b)

Fig. 2. (a) Lena corrupted by 20% of impulse noise, (b) the denoised result without semi-parametric modeling (PSNR=28.6 dB). Most of the fine details have been lost.

### *B. Semi-parametric formulation*

In this section, we adopt the semi-parametric modeling, as the means to remedy the smoothing effects associated with the problem formulation of the previous section. Moreover, we will attack the problem not by ad-hoc techniques, but by a theoretically sound modeling. We consider a set of real valued two dimensional functions $\{\psi_k,\ k=1,\ldots,K\}$, that can adequately model edges. Various types of functions can be used. In our experiments we used bivariate polynomials of order 1, functions of the form $\text{Erf}(a\cdot x+b\cdot y+c)$, where Erf is the error function, i.e.

$$\text{Erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt,$$

(which can approximate ridges - see figure 3(a), (b)) as well as functions of the form $\text{Exp}(-(a\cdot x+b\cdot y+c)^2)$ (see figure 3(c)) for several suitable choices of $a$, $b$ and $c$. The regularized risk minimization problem is now reformulated as follows:

$$\begin{aligned}&\underset{f\in\mathcal{H},\ \boldsymbol{\beta}\in\mathbb{R}^K,\boldsymbol{h}\in\mathbb{R}^4}{\text{minimize}} \quad c(f,\boldsymbol{h},\boldsymbol{\beta})=\\ &\frac{1}{N^2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}\Big|f(x_n,y_m)+h_0+h_1x_n+h_2y_m+h_3x_ny_m\\ &+\sum_{k=1}^{K}\beta_k\psi_k(x_n,y_m)-z_{n,m}\Big| \ + \ \frac{\lambda}{2N^2}\|f\|_{\mathcal{H}}^2\\ &+ \ \frac{\mu}{2K}\sum_{k=1}^{K}\beta_k^2 \ + \ \frac{\mu_1}{2}\sum_{l=1}^{3}h_l^2,\end{aligned}\tag{15}$$

where $\boldsymbol{\beta}=(\beta_1,\ldots,\beta_K)$, $\boldsymbol{h}=(h_0,h_1,h_2,h_3)$. In this case, the minimizer $\tilde{f}$ belongs to the space $\mathcal{H}+\Psi+\mathcal{P}$, where $\Psi=\overline{\text{span}\{\psi_k,\ k=1,\ldots,K\}}$ and $\mathcal{P}$ is the space of the bivariate polynomials of order 1 (see Appendix B). In other words, we recast problem (9), to account for some extra parameters, i.e. $\beta_k,\ k=1,\ldots,K,\ h_i,\ i=0,\ldots,3$ (that contribute to the preservation of the fine details of the image), which are also regularized. This approach (learning edge models from a rich set of basis functions) is similar to the modeling taken by the K-SVD algorithm [31].

The semi-parametric theorem II.2 ensures that the minimizer will have a finite representation of the form:

$$\begin{aligned}\tilde{f}(x,y)=&\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\alpha_{n,m}\kappa((x_n,y_m),(x,y))\\ &+\sum_{k=1}^{M}\beta_k\psi_k(x,y)+h_0+h_1x+h_2y+h_3xy.\end{aligned}\tag{16}$$

Once more, we can solve this problem using Polyak's Projected Subgradient Method. The necessary selected subgradients are given below:

$$\begin{aligned}\nabla c(f,\boldsymbol{h},\boldsymbol{\beta})=&(\nabla c_f(f,\boldsymbol{h},\boldsymbol{\beta}),\nabla c_{h_0}(f,\boldsymbol{h},\boldsymbol{\beta}),...,\nabla c_{h_3}(f,\boldsymbol{h},\boldsymbol{\beta}),\\ &\nabla c_{\beta_1}(f,\boldsymbol{h},\boldsymbol{\beta}),\ldots,\nabla c_{\beta_K}(f,\boldsymbol{h},\boldsymbol{\beta}))^T,\end{aligned}\tag{17}$$

where

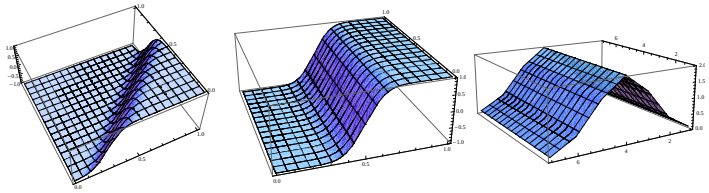
$$\begin{aligned}\nabla c_f(f,\boldsymbol{h},\boldsymbol{\beta})=&\frac{1}{N^2}\Big(\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}sign\left(e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta})\right)\cdot\\ &\cdot\kappa\left((x_n,y_m),(\cdot,\cdot)\right)+\lambda\cdot f\Big),\end{aligned}\tag{18}$$

$$\nabla c_{h_0}(f,\boldsymbol{h},\boldsymbol{\beta})=\frac{1}{N^2}\left(\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}sign\left(e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta})\right)\right),\tag{19}$$

$$\begin{aligned}\nabla c_{h_1}(f,\boldsymbol{h},\boldsymbol{\beta})=&\frac{1}{N^2}\left(\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}sign\left(e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta})\right)\cdot x_n\right)\\ &+\mu_1\cdot h_1,\end{aligned}\tag{20}$$

$$\begin{aligned}\nabla c_{h_2}(f,\boldsymbol{h},\boldsymbol{\beta})=&\frac{1}{N^2}\left(\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}sign\left(e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta})\right)\cdot y_m\right)\\ &+\mu_1\cdot h_2,\end{aligned}\tag{21}$$

$$\begin{aligned}\nabla c_{h_3}(f,\boldsymbol{h},\boldsymbol{\beta})=&\frac{1}{N^2}\left(\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}sign\left(e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta})\right)\cdot x_ny_m\right)\\ &+\mu_1\cdot h_3,\end{aligned}\tag{22}$$



(a) (b) (c)

Fig. 3. Some of the functions $\psi_k$ that are used to represent edges. (a) Erf$(8x-8y-2)$, (b) Erf$(8x-4)$, (c) Exp$(-(8x-4)^2)$.

and

$$\nabla c_{\beta_k}(f,\boldsymbol{h},\boldsymbol{\beta}) = \frac{1}{N^2}\Big(\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} sign\left(e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta})\right)\cdot \cdot \psi_k(x_n,y_m)\Big) + \frac{\mu}{K}\cdot\beta_k, \quad (23)$$

for $k=1,\ldots,K$, where the $e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta})$ term is given by:

$$e_{n,m}(f,\boldsymbol{h},\boldsymbol{\beta}) = f(x_n,y_m) + h_0 + h_1x_n + h_2y_m + h_3x_ny_m + \sum_{k=1}^{K}\beta_k\psi_k(x_n,y_m) - z_{n,m}.$$

### C. The algorithm

The choice of the regularization parameters $\mu$, $\mu_1$ (especially the first one) plays an important role in the edge-preservation properties of the algorithm. Roughly speaking, we adjust $\mu$ and $\mu_1$ so that they take small values around edges and large values in smoother areas (the regularization parameter $\lambda$ is kept fixed). The reason for this approach is quite obvious. Small values for the regularization parameters, $\mu$ and $\mu_1$, enhance the contribution of the semi-parametric part, which is desirable around edges. On the other hand, in smoother regions, the effect of the semi-parametric part of the algorithm needs to be suppressed. Thus, larger values for $\mu$, $\mu_1$ are adopted. As the algorithm moves from one pixel to the next (with user-defined step sizes, $s$), it decides whether the corresponding pixel-centered region contains edges or not (in order to compute a proper value for $\mu$ and $\mu_1$) and it solves the corresponding minimization problem; that is, either (9) or (15), depending on the "degree of smoothness" of the specific region. More specifically, we consider $L$ different types of smoothness, where $L$ is a user defined parameter. The smoothness of each region is determined by the value of the mean gradient in the region. We consider $L$ distinct cases for the regions, depending on how large the respective mean gradient is. Once the type of region has been decided, we assign values to $\mu$ and $\mu_1$ accordingly. This is accomplished using the values of the vectors ($L$ elements each) $\boldsymbol{\mu}$ and $\boldsymbol{\mu}_1$. The elements $\mu_i$ and $\mu_{1,i}$ contain the regularization values associated with any region of type $i$. The input parameters of the algorithm and their usage are shown in table I. In effect, the proposed method gives higher weight to kernel smoothing in smooth regions and performs sparse modeling for edge regions.

In order to compute the mean gradient in each one of the regions, and therefore assign each region to a specific type $i$, a preprocessing step is first required. The regularization parameters $\mu$ and $\mu_1$ are fixed (i.e., they are given initial values) and the minimization problem (15) is solved for all regions, moving from one pixel to the next (at this step, each region has size $N_0\times N_0$, where $N_0$ is a user-defined initial value). This first estimate of the denoised image (which contains much less noise from the original noisy one) is used to compute the gradients during the second step. The process is repeated, however this time, each region is assigned to a specific type $i$, $1\leq i\leq L$ (according to the "degree" of smoothness). Thus, the respective values $\mu_i$ and $\mu_{1,i}$ are used for the regularization. The second image estimate contains even less noise and the iteration continues (usually no more than 3 steps are needed).

To assign a region to a specific type, we use the respective mean gradient (obtained after the preprocessing step) and the information contained in the vector $\boldsymbol{p}$ ($L$ elements that sum up to 1). If the mean gradient of the region is larger than the $100\cdot(1-p_1)\%$ of the mean gradients of all regions, then the region is of type 1 (strongest edge). If the region is not of type 1 and its mean gradient is larger than the $100\cdot(1-p_2)\%$ of the mean gradients of all regions, then the region is of type 2, e.t.c. In smooth regions (i.e., type 6 or 5) we solve (9), while in regions which contain edges we solve the minimization problem (15), using the corresponding $\mu_i$ and $\mu_{1,i}$ depending on the "smoothness degree". The actual size of each pixel-centered region is defined by the vector $\boldsymbol{N}$ (see table I). The latter is a vector of dimension $L$. Its $i$-th element, i.e., $N_i$, defines the size $N_i\times N_i$ of the respective window for regions of the $i$-th type. These are user-defined variables and the values used in the context of this paper are shown in Tables II-VI. This concept of variable size windows has previously been used in the context of median filtering (e.g. [44]).

The computation of the mean gradient is performed at the beginning of each iteration. In particular, at each pixel, we compute the mean gradient of a $N_{\max}\times N_{\max}$ window centered at the specific pixel, where $N_{\max}=\max\{N_1,\ldots,N_L\}$. As a consequence, a mean gradient is assigned to each pixel of the image. Everytime the algorithm tries to assign a region to a specific type, it exploits the value of the mean gradient of the pixel at the center of that particular region.

The last algorithmic issue is how the final values of pixels are computed. Keep in mind that, each region centered at a specific pixel assigns values to all its neighbors. This means that each pixel is assigned to as much as $N^2$ discrete values (possibly less, if the step sizes are taken to be larger than 1). There are two solutions to this problem. The final value

(a) (b) (c)

Fig. 4. (a) Lena ($256 \times 256$) corrupted by impulse noise, (b) the denoised image according to kernel PCA denoising presented in [29] (PSNR=26.14 dB), (c) the denoised image according to the proposed method (PSNR=27.43 dB). The difference in quality is increased significantly if the $512 \times 512$ version of Lena is used.

of the pixel can be computed either as the mean of all the aforementioned values, or alternatively, as the value assigned to it by its corresponding region (i.e., the region centered at the pixel in question). For impulse noise removal, the latter solution seems to result to slightly better performance.

Finally, the problem of the selection of functions $\psi$ suitable to represent edges must be addressed. As mentioned in section III-B, we employed bivariate polynomials of order 1, functions of the form $\text{Erf}(a \cdot x + b \cdot y + c)$ and functions of the form $\text{Exp}(-(a \cdot x + b \cdot y + c)^2)$, for several suitable choices of $a$, $b$ and $c$. In particular, for regions of size $5 \times 5$, 44 functions are used (mainly rotations and translations of the ones shown in figure 3), for regions of size $7 \times 7$, 52 functions are used and for $9 \times 9$ regions, 76 functions are considered. Also, we should emphasize that the choice of the collection of $\psi$ functions is far from critical. It is possible that a larger set of suitable functions would enhance the results, but it would increase the computing time significantly. In other words, these functions can be considered as rich enough "basis" to account for the different orientations as well as locations of the edges within each window. The choice of their number and the respective parameters has been the result of extensive experimentation. The exact values of the parameters are not critical, as long as a rich enough representation has been achieved. Relevant details can be found in http://cgi.di.uoa.gr/∼stheodor/ker_den/index.htm. The algorithm is given below in a more detail.

**Kernel Denoising Algorithm**

1) Input: $\hat{f}$ (the noisy image), $N_0$, $\lambda$, $\mu_0$, $L$, $\boldsymbol{\mu}$, $\boldsymbol{\mu}_2$, $\boldsymbol{p}$, $\boldsymbol{c}$, $\boldsymbol{N}$, $\boldsymbol{m}$, $\nu$, $\boldsymbol{s}$.
2) (Initialization step) For each pixel do:
   a) Take the $N_0 \times N_0$ neighborhood of the pixel
   b) Solve the optimization problem (15) using the parameters $\lambda$, $\mu_0$.
   c) Put the solution to the denoised image $f_1$.
   d) Move to the next pixel using the step $s_1$
3) Set $N_{\max} = \max\{N_1, \dots N_L\}$
4) for $r = 2$ to $\nu$ do:
   a) At each pixel, compute the mean gradient of all pixels in the corresponding $N_{\max} \times N_{\max}$ neighborhood of image $f_{r-1}$.
   b) Sort the values of the gradients in a descending order.
   c) For each pixel do:
      i) Compute the type $i$ of the region according to the mean gradient and the information stored in $\boldsymbol{p}$ (see table I).
      ii) Take the $N_i \times N_i$ neighborhood of the current pixel of the image $f_{r-1}$
      iii) Set $\mu = mu_i$ and $\mu_1 = mu_{1,i}$ (see table I).
      iv) Take the (noisy) region of image $\hat{f}$ centered at the specified pixel and solve the optimization problem with the parameters $\lambda$, $\mu$, $\mu_2$ using the Polyak's Projected Subgradient Method. The problem to be solved is either (9) or (15), according to the type of the region. This information is stored in $\boldsymbol{m}_i$.
      v) Put the solution to the denoised image $f_r$.
      vi) Move to the next pixel using the step $s_r$.
5) Output: The denoised image $f_\nu$.

## IV. Experiments

The kernelized algorithm was implemented in C. The source code along with all the images used in the paper can be found at http://cgi.di.uoa.gr/∼stheodor/ker_den/index.htm.[3], for the sake of reproducibility of results [45]. Experiments were conducted on several test images contained in the Waterloo Image Repository (see [46]), which were corrupted with various types of synthetic noise. The results were compared with those obtained using several state of the art models (BiShrink[4] - [3], [4], [15], ProbShrink[5] - [47], BLS-GSM[6] - [2], K-SVD[7] -[31], SKR[8] - [14], BM3D[9] - [5]). We note that in all cases the input parameters were carefully adjusted to obtain the best possible results with respect to PSNR (in most cases this approach led to better visual quality too). The

[3]In this page the interested reader can found many more test images.
[4]Using the code provided at http://taco.poly.edu/WaveletSoftware/denoise2.html
[5]Using the code provided at http://www.pudn.com/downloads150/sourcecode/windows/mul
[6]Using the code provided at http://decsai.ugr.es/∼javier/denoise
[7]Using the code provided at http://www.cs.technion.ac.il/∼elad/software.
[8]Using the code provided at http://users.soe.ucsc.edu/∼htakeda/kernelToolbox.htm.
[9]Using the code provided at http://www.cs.tut.fi/∼foi/GCF-BM3D

TABLE I
DESCRIPTION OF THE INPUT PARAMETERS OF THE ALGORITHM.

| Parameter | Type | Usage |
|---|---|---|
| $L$ | integer | The number of distinct types of regions depending on the "degree of smoothness". |
| $N_0$ | odd integer | The size of the pixel-centered region for the initial step. |
| $\mu_0$ | real | The value for $\mu$ and $\mu_2$ for the initial step of the algorithm. |
| $\boldsymbol{p}$ | vector ($L$ elements) that sum up to 1 | This vector is used to detect the type of region. If the mean gradient of the region is larger than the $100 \cdot (1 - p_1)\%$ of the mean gradients of all regions, then the region is of type 1 (strongest edge). If the region is not of type 1 and its mean gradient is larger than the $100 \cdot (1 - p_2)\%$ of the mean gradients of all regions, then the region is of type 2 e.t.c. |
| $\boldsymbol{N}$ | vector ($L$ elements) | This vector contains the values of $N$ (the size of the region) that will be considered at each pixel according to its type. If the region is of type $i$ then $N = N_i$. |
| $\boldsymbol{\mu}$ | vector ($L$ elements) | This vector contains the values of $\mu$ that will be used to each region according to its type. If the region is of type $i$ then $\mu = mu_i$. |
| $\boldsymbol{\mu_2}$ | vector ($L$ elements) | This vector contains the values of $\mu_2$ that will be used to each region according to its type. If the region is of type $i$ then $\mu_2 = mu_{2,i}$. |
| $\nu$ | integer | the number of iteration steps. |
| $\boldsymbol{s}$ | vector ($\nu$ elements) | the step size from one pixel to the next for each iteration. |
| $\lambda$ | real | the regularization parameter of the optimization problem. |
| $\sigma$ | real | the parameter of the gaussian kernel. |

results show that the reproducing kernel approach performs almost as well as BiShrink in the presence of Gaussian noise. However, it outperforms significantly kernel or wavelet-based methods when impulse or mixed noise is considered. This enhanced performance is obtained at the cost of higher complexity, which is basically contributed by the optimization step, which is of the order of $O(N^2)$ per pixel. In terms of absolute execution times, the proposed algorithm seems to have the same performance as the SKR algorithm ([14]). Thus on a $512 \times 512$ image the algorithm may need up to ten minutes to complete. Currently, more efficient optimization algorithms are considered. Moreover, the whole setting is open to a straightforward parallelization, when a parallel processing environment is available. This is also currently under consideration.

It should be noted that, although the kernel based algorithm presented in this paper, at a first look, has many input parameters (as shown in Table I), most of them were kept constant. In particular, $L = 6$, $\boldsymbol{\mu} = (0.01, 0.1, 0.5, 5, 50, 100)$, $\boldsymbol{\mu_2} = (0, 0, 0, 0.1, 1, 3)$, $\mu_0 = 0.1$, $\boldsymbol{m} = (2, 2, 1, 1, 1, 0)$, $\nu = 3$, $\boldsymbol{s} = (3, 3, 1)$, $\lambda = 1$, $\sigma = 3$. Loosely speaking, this means that 6 types of regions are considered: the first two are regions that contain strong edges (thus smaller values for $\mu$ are taken and a large number of iterations is used as the values of $\boldsymbol{m}$ indicate), the next three are regions with soft edges and the last type is for smooth regions. In addition for all the examples we set $c = (1, 1, 1, 1, 1, 1)$. Therefore the only parameters that need to be selected by the user are the vectors $\boldsymbol{N}$ and $\boldsymbol{p}$. The values of $\boldsymbol{N}$ depend on the amount of the additive noise (as it is the case in median filters; the larger the noise the larger the values of the elements of $\boldsymbol{N}$), while the values of $\boldsymbol{p}$ depend on the percentage of edges in the image. Although $\boldsymbol{N}$ contains six elements, only two values need to be set. One that corresponds to strong and softer edges (the first three elements of $\boldsymbol{N}$) and one that corresponds to smooth regions (the last three elements - see tables II - VI). In the presented tests the vector $\boldsymbol{N}$ typically takes the values $(5, 5, 5, 5, 5, 5)$, $(5, 5, 5, 7, 7, 7)$, $(7, 7, 7, 7, 7, 7)$ and $(7, 7, 7, 9, 9, 9)$, while $\boldsymbol{p}$ is $(0.1, 0.1, 0.1, 0.1, 0.2, 0.2)$ for images with a medium amount of edges (such as lena and peppers) and $(0.2, 0.2, 0.1, 0.1, 0.2, 0.4)$ for images which contain many edges (such as barbara and boat). Once more, we emphasize the low sensitivity of the algorithm to the input parameters.

### A. Impulse Noise

Two types of impulse noise are considered. The first, which we call *type 1*, is the typical (bipolar) impulse noise with pdf:

$$p(z) = \begin{cases} p, & \text{if } z = a, \text{ or } z = -a \\ 1 - 2p, & \text{otherwise,} \end{cases} \tag{24}$$

for some $0 \leq p \leq \frac{1}{2}$, $a > 0$. This means that approximately $200p\%$ of the pixels will be corrupted. The second type of impulse noise (*type II*) has uniformly distributed impulses, i.e. $200p\%$ of the pixels will be corrupted with additive uniform noise in the range $[-a, a]$, for some $0 \leq p \leq \frac{1}{2}$, $a > 0$. In both types of impulse noise the proposed algorithm gives excellent results (both visually and in terms of PSNR). The wavelet-based techniques are known not to be able to deal with impulse noise effectively. Results show that the proposed kernel-based noise removal algorithm can achieve an improvement of more than 5dB (some times up to 10dB) in terms of PSNR, over most wavelet-based methods and much better visual quality (even in cases where the difference of PSNRs is relatively small). The same is true for most other methods such as K-SVD and SKR. Moreover, the kernelized approach gives significantly enhanced results over the traditional median filter, especially in visual quality. The only method that yields competitive results (in terms of PSNR) is the SKR variant that uses the $L_1$ norm, although in most cases the difference is more than 1 dB and the noise is not removed effectively (see figures 6, 7). The denoised images obtained by the BM3D algorithm, which are superior to the ones obtained by other kernel based or wavelet based methods, yield significant loss in details and a difference of 1-3 dBs in comparison to the proposed methodology. In the case of Barbara, although BM3D results in higher PSNR than the proposed kernel based methodology, there is a significant loss in fine details as it can been seen, for example, in figure 8. In figure 8(c), it can be observed that the face of Barbara is distorted and the texture of the chair (behind her), as well the texture of the table clothe, are blurred compared to the image obtained by the proposed kernel based methodology, as it becomes evident in figure 8(b). There are, of course, some other regions (patterns in the pants and scarf) that BM3D restores better. Tables II, III and figures 5-9 report the results of the kernelized denoising algorithm on Lena, Peppers, Barbara and Boat images corrupted by various types of impulse noise.

### B. Gaussian Noise

The pdf of the zero-mean gaussian noise is given by:

$$p(z) = \frac{1}{\sqrt{2\pi}s} e^{-\frac{z^2}{2s^2}}, \tag{25}$$

where $z$ represents the gray level and $s$ is the standard deviation (the average value of $z$ is 0). It is well known that $70\%$ of the values of $z$ will be in the range $[-s, s]$ and $95\%$ will be in the range $[-2s, 2s]$. Most wavelet based methods were developed especially for this kind of noise (mainly because of its mathematical tractability in both spatial and frequency domains). The BM3D algorithm developed in [5] is reported to be one of the best approaches in gaussian noise removal. Its performance shows an average improvement of approximately 2 dBs (sometimes more) over the kernel-based approach. In many cases, our method seems to have similar behavior (in terms of PSNR) with another very well known wavelet-based algorithm called BiShrink[10] (see [3], [4]). Table IV and figures 10-11 report the results of the experiments conducted on several images corrupted by gaussian noise.

### C. Uniform Noise

The pdf of the zero-mean uniform noise is given by

$$p(z) = \begin{cases} \frac{1}{2a}, & \text{if } -a \leq z \leq a \\ 0, & \text{otherwise,} \end{cases} \tag{26}$$

for $a > 0$. Its variance is $\sigma^2 = \frac{a^2}{3}$.

The kernelized denoising algorithm performs relatively well in the presence of uniform noise, but wavelet-based methods clearly give better results both visually and in terms of PSNR (see Table V).

### D. Mixed Noise

We included in the simulated experiments several images corrupted by mixed noise of various types as specified below:
mixed 1: 20% of impulse noise (type II, $a = 128$) + gaussian noise with $s = 10$.
mixed 2: 30% of impulse noise (type II, $a = 128$) + gaussian noise with $s = 20$.
mixed 3: uniform noise in the interval $[-10, 10]$ + gaussian noise with $s = 10$.
mixed 4: uniform noise in the interval $[-10, 10]$ + 10% impulse noise (type II, $a = 128$).
mixed 5: uniform noise in the interval $[-10, 10]$ + 10% impulse noise (type II, $a = 128$) + gaussian noise with $s = 10$.

The results are reported in table VI and figures 12-14. The kernelized denoising method can effectively remove any of these types of mixed noise. In the presence of noise with impulse components, the proposed algorithm seems to outperform all other techniques in most cases both in visually quality and in terms of PSNR.

[10]This is also true for ProbShrink ([47]).

## V. Conclusions

A novel denoising algorithm was presented based on the use of Reproducing Kernel Hilbert Spaces. The semiparametric Representer Theorem was exploited in order to cope with the problems associated with the smoothing around edges, which is a common problem in almost all denoising algorithms. The comparative study against wavelet based techniques, showed that significantly enhanced results are obtained in the case of impulse noise. In the case of gaussian noise, the proposed algorithm performs quite well (in terms of PSNR the results are similar with BiShrink). In addition the kernelized approach can effectively treat any type of mixed noise, resulting at significantly better results than wavelet-based methods, especially if impulse components are present. The previously reported enhanced performance is achieved at a higher computational complexity.

## Appendix A
## Differentiability of Operators

Since gradients and subgradients of operators defined in Hilbert spaces play a crucial role in several parts of this paper, it is important to present their formal definitions and their key properties.

**Definition A.1.** *Consider an operator $T : H \to \mathbb{R}$, where $(H, \langle\cdot,\cdot\rangle_H)$ is a Hilbert space. $T$ is said to be Fréchet differentiable at $x_0$, if there exists a $y \in H$ such that*

$$\lim_{\|h\|_H \to 0} \frac{T(x_0 + h) - T(x_0) - \langle y, h\rangle_H}{\|h\|_H} = 0, \tag{27}$$

*where $\|\cdot\|_H = \sqrt{\langle\cdot,\cdot\rangle_H}$ is the induced norm. Usually, the element $y \in H$ is called the gradient of $T$ at $x$ and the notation $y = \nabla T(x_0)$ is used to refer to it*[11].

For convex functions defined on Hilbert spaces the gradient at $x_0$ satisfies the well known first order condition:

$$T(z) \geq T(x_0) + \langle \nabla T(x_0), z - x_0\rangle.$$

for all $z$. This condition has a simple geometric meaning when $T$ is finite at $x_0$: it says that the graph of the affine function $h(z) = T(x_0) + \langle \nabla T(x_0), z - x_0\rangle$ is a non-vertical supporting hyperplane to the convex set $\operatorname{epi} T$[12] at $(x_0, T(x_0))$. In other words, (a) $h(z)$ defines an osculant hyperplane of the graph of $T$ at $(x_0, T(x_0))$ and (b) all the points of the graph of $T$ lie at the same side of the hyperplane. This is one of the reasons why the notion of gradient is so important in optimization problems. If $T$ is not differentiable at $x$, we can still construct such a hyperplane using a subgradient.

**Definition A.2.** *Let $T : H \to \mathbb{R}$ be a convex function defined on a Hilbert space $(H, \langle\cdot,\cdot\rangle_H)$. A vector $x^* \in H$ is said to be a subgradient of $T$ at $x_0$ if*

$$T(z) \geq T(x_0) + \langle x^*, z - x_0\rangle_H.$$

[11]In the literature the notation $T'(x)$ is also used to refer to the gradient of $T$ at $x$.

[12]$\operatorname{epi} T$ denotes the epigraph of $T$, i.e. the set $\{(x, y) : x \in H, y \in \mathbb{R} : T(x) \leq y\}$.

(a) (b) (c)

Fig. 5. (a) Lena corrupted by 20% of impulse noise of type II, (b) denoising using the kernel approach (PSNR=35.27 dB), (c) denoising with BiShrink (PSNR=22.83 dB).

(a) (b) (c)

Fig. 6. (a) Lena corrupted by 40% of impulse noise of type II, (b) denoising using the kernel approach (PSNR=31.78 dB), (c) denoising with SKR -$L_1$ (PSNR=31.33 dB).

(a) (b) (c)

Fig. 7. (a) Lena corrupted by 50% of impulse noise of type II, (b) denoising using the kernel approach (PSNR=30.71 dB), (c) denoising with SKR - $L_1$ (PSNR=29.05 dB).

(a) (b) (c)

Fig. 8. (a) Barbara corrupted by 40% of impulse noise of type II, (b) denoising using the kernel approach (PSNR=24.81 dB), (c) denoising with BM3D (PSNR=27.05 dB).

(a) (b) (c)

Fig. 9. (a) Boat corrupted by 40% of impulse noise of type II, (b) denoising using the kernel approach (PSNR=29.14 dB), (c) denoising with BM3D (PSNR=27.26 dB).

(a) (b) (c)

Fig. 10. (a) Lena corrupted by gaussian noise with $s = 10$, (b) denoising using the kernel approach (PSNR=33.98 dB), (c) denoising with BiShrink (PSNR=34.33 dB).

(a) (b) (c)

Fig. 11. (a) Boat corrupted by gaussian noise with $s = 20$, (b) denoising using the kernel approach (PSNR=29.46 dB), (c) denoising with BM3D (PSNR=31.65 dB).

*The set of all subgradients of $f$ at $x_0$ is called the subdifferential of $T$ at $x_0$ and is denoted by $\partial T(x)$.*

As an example, we consider the operator $T(f) = \|f(\boldsymbol{x}_0) - y_0\|$, defined on a RKHS $\mathcal{H}$, where $\boldsymbol{x}_0 \in \mathbb{R}^n$ and $y_0 \in \mathbb{R}$ (in other words we take a simple form of a cost function that employs the $l_1$ norm). Using the kernel properties (see section II) we take $f(\boldsymbol{x}_0) = \langle f, \kappa(\boldsymbol{x}_0, \cdot)\rangle$. Thus $T(f) = \|\langle f, \kappa(\boldsymbol{x}_0, \cdot)\rangle - y_0\|$. This operator is non differentiable at any $f$, such that $f(\boldsymbol{x}_0) = y_0$. The subgradients of $T$ at $f$ are given below:

$$\nabla T(f)(\cdot) = \begin{cases} \text{sign}\,(f(\boldsymbol{x}_0) - y_0) \cdot \kappa(\boldsymbol{x}_0, \cdot), \\ \quad \text{if } f : f(\boldsymbol{x}_0) \neq y_0, \\ \lambda\kappa(\boldsymbol{x}_0, \cdot), \text{ for any } -1 \leq \lambda \leq 1, \\ \quad \text{if } f : f(\boldsymbol{x}_0) = y_0. \end{cases} \tag{28}$$

More on the subject can be found in [48], [49], [50], [51].

## APPENDIX B
## SUMS OF HILBERT SPACES

Another concept, that is used in key parts of the paper, is that of the summation of Hilbert Spaces. Here, we give a more rigorous analysis of this important subject. Note, that there are two distinct cases of Hilbert space's summation: the *direct sum* and the *ordinary sum*.

**Definition B.1.** *Two Hilbert spaces $(H_1, \langle\cdot,\cdot\rangle_{H_1})$ and $(H_2, \langle\cdot,\cdot\rangle_{H_2})$ can be combined into another Hilbert space, called the (orthogonal) direct sum, and denoted by $H = H_1 \oplus H_2$, consisting of the set of all ordered pairs $(x_1, x_2)$ where $x_i \in$*

Fig. 12. (a) Lena corrupted by gaussian noise with $s = 10$ and 20% of impulse noise of type II ($a = 128$), (b) denoising using the kernel approach (PSNR=32.27 dB), (c) denoising with BiShrink (PSNR=25.30 dB), (d) denoising with K-SVD (PSNR=27.51 dB), (e) denoising with SKR $L_1$ (PSNR=31.14 dB), (f) denoising with BM3D (PSNR=30.65 dB).

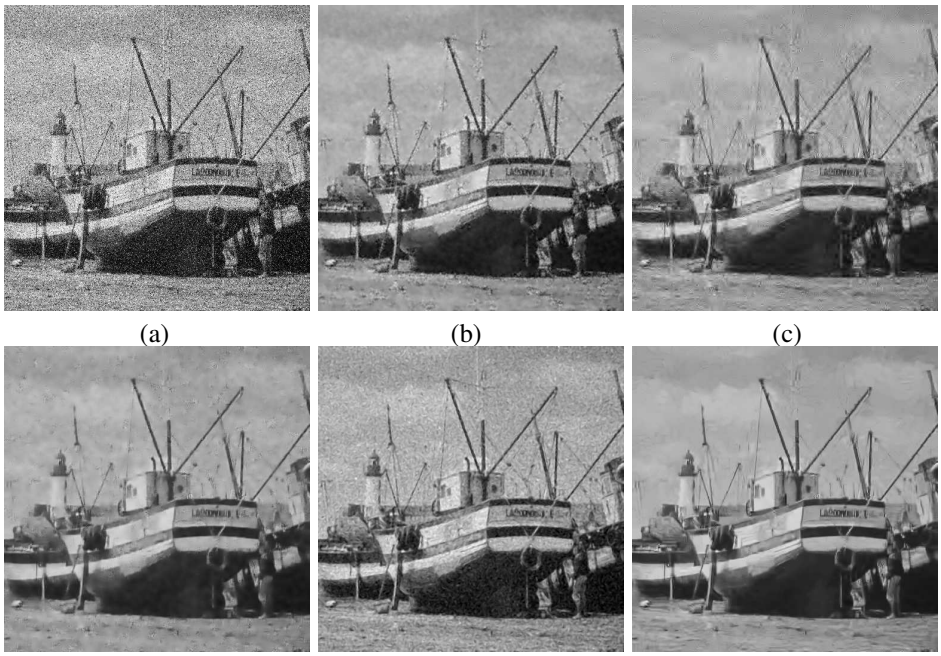


Fig. 13. (a) Boat corrupted by gaussian noise with $s = 20$ and 30% of impulse noise of type II ($a = 128$), (b) denoising using the kernel approach (PSNR=26.90 dB), (c) denoising with BiShrink (PSNR=25.62 dB), (d) denoising with K-SVD (PSNR=26.56 dB), (e) denoising with SKR $L_1$ (PSNR=26.18 dB), (f) denoising with BM3D (PSNR=27.6 dB).

(a) (b) (c)

Fig. 14. (a) Lena corrupted by gaussian noise with $s = 10$, uniform noise in the interval $[-10, +10]$ and 10% of impulse noise of type II ($a = 128$), (b) denoising using the kernel approach (PSNR=32.74 dB), (c) denoising with BM3D (PSNR=31.77 dB).

(a) (b) (c)

Fig. 15. (a) Boat corrupted by uniform noise in the interval $[-10, +10]$ and 10% of impulse noise of type II ($a = 128$), (b) denoising using the kernel approach (PSNR=32.28 dB), (c) denoising with BM3D (PSNR=30.5 dB).

$H_i$, $i = 1, 2$, *and inner product defined by*

$$\langle (x_1, x_2), (y_1, y_2) \rangle_{H_1 \oplus H_2} = \langle x_1, y_1 \rangle_{H_1} + \langle x_2, y_2 \rangle_{H_2}. \quad (29)$$

The direct sum can be generalized to infinite sums of Hilbert spaces (see e.g. [49]).

**Definition B.2.** *Consider two Hilbert spaces $(H_1, \langle \cdot, \cdot \rangle_{H_1})$ and $(H_2, \langle \cdot, \cdot \rangle_{H_2})$ subsets of the larger space $F$. We may define the sum $H = H_1 + H_2 \subseteq F$, as follows:*

$$\begin{aligned} &x \in H, \text{ iff there are } x_1 \in H_1 \text{ and } x_2 \in H_2, \\ &\text{such that } x = x_1 + x_2. \end{aligned} \quad (30)$$

*Then $H$ is a Hilbert space with inner product defined by*

$$\begin{aligned} \langle x, y \rangle_{H_1 + H_2} = \min\{ &\langle x_1, y_1 \rangle_{H_1} + \langle x_2, y_2 \rangle_{H_2}, \\ &\text{for all } x_1, y_1 \in H_1, x_2, y_2 \in H_2, \\ &\text{such that } x = x_1 + x_2, y = y_1 + y_2 \}. \end{aligned} \quad (31)$$

The sum of Hilbert spaces can be easily generalized to include a finite number of Hilbert spaces (see [18]). In contrast with the direct sum, this generalization is not valid, if an infinite number of spaces is considered. Note that in the special case, where for each $x \in H$ there is a unique decomposition $x_1 + x_2$, $x_1 \in H_1$, $x_2 \in H_2$, the two sums coincide (i.e. there is a 1-1 mapping between $H_1 + H_2$ and $H_1 \oplus H_2$). Moreover, it is easy to see that in both cases we can define gradients and subgradients of operators defined in the combined space $H$.

In sections III-A and III-B the sums $\mathcal{H} + R$ and $\mathcal{H} + \Psi + \mathcal{P}$ were used in a rather superficial way. However, in light of the information given before, one may easily see that each element of $\mathcal{H} + R$ can be uniquely decomposed into $f + g$, where $f \in \mathcal{H}$ and $g \in R$ (in fact $f$ lies in a finite dimensional subspace of $\mathcal{H}$) and thus to conclude that the sum $\mathcal{H} + R$ can be identified to the space $\mathcal{H} \oplus R$. Hence, we don't have to use the cumbersome inner product (and respective norm) given in (31) to compute the gradients. Instead, we employ the more elegant inner product associated to the direct sum definition in (29).

With the same rationale, the space $\mathcal{H} + \Psi + \mathcal{P}$ can be also identified to the space $\mathcal{H} \oplus \Psi \oplus \mathcal{P}$. Indeed, It is easy to see that if we select the functions $\psi_k$ properly (i.e. so that they cannot be decomposed into a finite sum of gaussian kernels and/or bivariate polynomials of order 1), then each $\tilde{f} \in \mathcal{H} + \Psi + \mathcal{P}$ is uniquely decomposed into $f + \psi + p$ where $f \in \mathcal{H}$, $\psi \in \Psi$, $p \in \mathcal{P}$.

## REFERENCES


[1] D. K. Hammonf and E. P. Simonceli, "A machine learning framework for adaptive combination of signal denoising methods," *ICIP 2007*.

[2] J. Portilla, V. Strela, M. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain," *IEEE Trans. Im. Proc.*, vol. 12, no. 11, pp. 1338–1351, 2003.

[3] L. Sendur and I. Selesnick, "Bivariate shrinkage with local variance estimationn," *IEEE Signal Process Lett.*, vol. 9, no. 12, pp. 438–441, 2002.

[4] ——, "Bivariate shrinkage functions for wavelet-based denoising exploiting interscale dependency," *IEEE Trans. Signal Process.*, vol. 50, no. 11, pp. 2744–2756, 2002.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3d transform-domain collaborative filtering," *IEEE Tran. Im. Proc.*, vol. 16, no. 8, pp. 2080–2095, 2007.

[6] K. Seongjai, "PDE-based image restoration : A hybrid model and color image denoising," *IEEE Trans. Im. Proc.*, vol. 15, no. 5, pp. 1163–1170, 2006.

[7] A. Buades, B. Coll, and J. M. Morrel, "A review of image denoising algorithms, with a new one," *SIAM Multiscale Modelling and Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[8] C. Kervrann and J. Boulanger, "Local adaptivity to variable smoothness for exemplarbased image regularization and representation," *IJVC*, vol. 79, no. 1, pp. 45–69, 2008.

[9] E. Abreu, M. Lightstone, S. Mitra, and K. Arakawa, "A new efficient approach for the removal of impulse noise from highly corrupted images," *IEEE Trans. Im. Proc.*, vol. 5, pp. 1012–1025, 1996.

[10] R.Garnett, T. Huegerich, and C. Chui, "A universal noise removal algorithm with an impulse detector," *IEEE Trans. Im. Proc.*, vol. 14, no. 11, pp. 1747–1754, 2005.

[11] E. Besdok, "Impulsive noise suppression from images by using anfis interpolant and lillietest," *EURASIP J. Appl. Si. Pr.*, vol. 16, pp. 2423–2433, 2004.

[12] C. F. J. Cai, R. Chan, "Minimization of detail-preserving regularization functional for impulse noise removal," *JMIV*, vol. 29, no. 1, pp. 79–91, 2007.

[13] M. Ghazel, G. H. Freeman, and E. Vrscay, "Fractal image denoising," *IEEE Trans. Im. Proc.*, vol. 12, no. 12, pp. 1560–1578, 2003.

[14] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Tran. Im. Proc.*, vol. 16, no. 2, pp. 349–366, 2007.

[15] S. Cai and K. Li, "Wavelet software at Brookly Poly." [Online]. Available: http://taco.poly.edu/WaveletSoftware/index.html

[16] J. Mercer, "Sturm-liuville series of normal functions in the theory of integral equations," *Philos. Trans. Roy. Soc. London, Ser. A*, vol. 211, pp. 111–198, 1911.

[17] E. H. Moore, "On properly positive hermitian matrices," *Bull. Amer. Math. Soc.*, vol. 23, 1916.

[18] N. Aronszajn, "La théorie générale des noyaux reproduisants et les applications. premier partie," *Proc. Cambridge Philos. Soc. Math.*, vol. 67, 1939.

[19] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, 4th edition*. Academic Press, 2009.

[20] B. Schölkopf and A. J. Smola, *Learning with Kernels*. MIT Press, 2002.

[21] M. E. Mavroforakis and S. Theodoridis, "A geometric approach to support vector machine (svm) classification," *IEEE Trans. Neural Netw.*, vol. 17, no. 3, pp. 671–682, 2006.

[22] S. Theodoridis and M. E. Mavroforakis, "Reduced convex hulls: A geometric approach to support vector machines," *IEEE Signal Proc. Mag.*, vol. 24, no. 3, pp. 119–122, 2007.

[23] K. Slavakis and S. Theodoridis, "Sliding window generalized kernel affine projection algorithm using projection mappings," *EURASIP J. Appl. Si. Pr.*, 2008, article id 830381.

[24] K. Slavakis, S. Theodoridis, and I. Yamada, "On line classification using kernels and projection based adaptive algorithm," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 2781–2797, 2008.

[25] J. Xu, A. Paiva, I. Park, and J. Principe, "A reproducing kernel hilbert space framework for information theoretic learning," *IEEE Trans. Signal Process.*, vol. 56, no. 12, pp. 5891–5902, 2008.

[26] R. M. Y. Angel, S. Mannor, "Dekernel recursive least squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, 2004.

[27] D. Li, "Support vector regression based image denoising," *Image Vision Comput.*, vol. 27, pp. 623–627, 2009.

[28] S. Zhang and Y. Chen, "Image denoising based on wavelet support vector machine," *IICCIAS 2006*.

[29] K. Kim, M. O. Franz, and B. Scholkopf, "Iterative kernel principal component analysis for image modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 9, pp. 1351–1366, 2005.

[30] K. Tan, S. Chen, and D. Zhang, "Robust image denoising using kernel-induced-measures."

[31] M. Elad and M. Aharon, "Image denoising via sparse and redundunt representations over learned dictionaries," *IEEE Tran. Im. Proc.*, vol. 15, no. 12, pp. 3736–3745, 2006.

[32] J. Mercer, "Functions of positive and negative type and their connection with the theory of integral equations," *Phil. Trans. Roy. Soc. Ser. A*, vol. 209, pp. 415–446, 1909.

[33] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, no. 3, pp. 337–404, 1950.

[34] G. S. Kimeldorf and G. Wahba, "Some results on tchebycheffian spline functions," *Journal of Mathematical Analysis and Applications*, vol. 33, pp. 82–95, 1971.

[35] A.Yuille1 and N. Grzywacz1, "A mathematical analysis of the motion coherence theory," *IJCV*, vol. 3, no. 2, pp. 155–175, 1989.

[36] S. Durand and J. Froment, "Reconstruction of wavelet coefficients using total variation minimization," *SIAM J. Sci. Comput.*, vol. 24, pp. 1754–1767, 2003.

[37] P. L. Combettes and J.-C. Pesquet, "Image restoration subject to a total variation constraint," *IEEE Trans. Im. Proc.*, vol. 13, no. 9, pp. 1213–1222, 2004.

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

[39] S. Osher and L. I. Rudin, "Feature-oriented image enhancement using shock filters," *SIAM J. Numer. Anal.*, vol. 27, pp. 919–940, 1990.

[40] L. I. Rudin and S. Osher, "Total variation based image restoration with free local constraints," *Proc. IEEE Int. Conf. Image Processing*, vol. 1, pp. 31–35, 1994.

[41] R. C. Gonzalez and R. Woods, *Digital Image Processing*. Prentice Hall, 2002.

[42] M. Nikolova, "Semi-explicit solution and fast minimization scheme for an energy with l1-fitting and tikhonov-like regularization," *Journal of Mathematical Imaging and Vision*, vol. 34, no. 1, pp. 32–47, 2009.

[43] B. T. Polyak, *Introduction to Optimization*. New York: Optimization Software, 1987.

[44] R. Suoranta and K. P. Estola, "Robust median filter with adaptive window length," *IEEE International Sympoisum on Circuits and Systems*, pp. 108–111, 1991.

[45] P. Vandewalle, J. Kovacevic, and M. Vetterli, "Reproducible research in signal processing - what, why, and how," *IEEE Signal Processing Magazine*, vol. 26, no. 3, pp. 37–47, May 2009. [Online]. Available: http://rr.epfl.ch/17/

[46] "Waterloo bragzone image repository," 2002. [Online]. Available: http://links.uwaterloo.ca/Repository.html

[47] A. Pizurica and W. Phillips, "Estimating the probability on the presence of a signal of interest in multiresolution single and multiband image denoising," *IEEE Trans. Im. Proc.*, vol. 15, no. 3, pp. 654–665, 2006.

[48] L. Debnath and P. Mikusinski, *Introduction to Hilbert Spaces with applications, second edition*. Academic Press, 1999.

[49] S. Lang, *Real and Functional Analysis, third edition*. Springer, 1993.

[50] Liusternik and Sobolev, *Elements of Functional Analysis*. Frederick Ungar Publishing Co, 1961.

[51] R. T. Rockafellar, *Convex Analysis*. Princeton University Press, 1970.

TABLE II

RESULTS OF THE KERNELIZED DENOISING METHOD ON VARIOUS IMAGES (WITH DIMENSIONS $512 \times 512$) CORRUPTED BY IMPULSE NOISE OF TYPE I, FOR $a = 100$.

| Image | Noise | noisy PSNR | Kernel Denoising | | BiShrink | Median |
|---|---|---|---|---|---|---|
| | | | $\boldsymbol{N}$ | denoised PSNR | | |
| Lena | 20% | 15,77 dB | $(5, 5, 5, 5, 5, 5)$ | 34,31 dB | 24,57 dB | 30.7 dB |
| | 30% | 14.01 dB | $(7, 7, 7, 7, 7, 7)$ | 32.07 dB | 26.19 dB | 28.3 dB |
| | 40% | 12.76 dB | $(7, 7, 7, 7, 7, 7)$ | 30.28 dB | 26.19 dB | 26.97 dB |
| | 50% | 11.78 dB | $(7, 7, 7, 9, 9, 9)$ | 28.53 dB | 24.43 dB | 25.1 dB |
| Peppers | 20% | 16,10 dB | $(5, 5, 5, 5, 5, 5)$ | 32,27 dB | 23,24 dB | 30.81 dB |
| | 30% | 14.31 dB | $(5, 5, 5, 7, 7, 7)$ | 30,48 dB | 25.37 dB | 29.00 dB |
| | 40% | 13.08 dB | $(7, 7, 7, 7, 7, 7)$ | 29,19 dB | 24.29 dB | 27.64 dB |
| | 50% | 12.13 dB | $(7, 7, 7, 9, 9, 9)$ | 27,88 dB | 23.30 dB | 25.77 dB |
| Barbara | 20% | 15,81 dB | $(5, 5, 5, 5, 5, 5)$ | 26,21 dB | 23,45 dB | 23.85 dB |
| | 30% | 14.07 dB | $(5, 5, 5, 7, 7, 7)$ | 24,92 dB | 23.47 dB | 22.41 dB |
| | 40% | 12.78 dB | $(7, 7, 7, 7, 7, 7)$ | 23,80 dB | 22.78 dB | 22.00 dB |
| | 50% | 11.81 dB | $(7, 7, 7, 9, 9, 9)$ | 22,88 dB | 22.28 dB | 21.88 dB |
| Boat | 20% | 15.92 dB | $(5, 5, 5, 5, 5, 5)$ | 31,04 dB | 24.04 dB | 228.37 dB |
| | 30% | 14.14 dB | $(5, 5, 5, 7, 7, 7)$ | 28,86 dB | 24.73 dB | 26.00 dB |
| | 40% | 12.90 dB | $(7, 7, 7, 7, 7, 7)$ | 27,62 dB | 24.91 dB | 25.05 dB |
| | 50% | 11.92 dB | $(7, 7, 7, 9, 9, 9)$ | 25,88 dB | 23.30 dB | 23.31 dB |

TABLE III

RESULTS OF THE KERNELIZED DENOISING METHOD ON VARIOUS IMAGES (WITH DIMENSIONS $512 \times 512$) CORRUPTED BY IMPULSE NOISE OF TYPE II, FOR $a = 128$.

| Image | Noise | noisy PSNR | Kernel Denoising | | BiShrink | K-SVD | SKR $L_1$ | SKR | BM3D |
|---|---|---|---|---|---|---|---|---|---|
| | | | $\boldsymbol{N}$ | PSNR | | | | | |
| Lena | 20% | 18.42 dB | $(5, 5, 5, 5, 5, 5)$ | 35.27 dB | 22.83 dB | 27.97 dB | 34.72 dB | 29.82 dB | 31.01 dB |
| | 30% | 16.60 dB | $(5, 5, 5, 7, 7, 7)$ | 33.20 dB | 25.90 dB | 27.88 dB | 33.13 dB | 28.90 dB | 28.74 dB |
| | 40% | 15.37 dB | $(7, 7, 7, 7, 7, 7)$ | 31.78 dB | 26.76 dB | 26.84 dB | 31.33 dB | 27.70 dB | 28.74 dB |
| | 50% | 14.41 dB | $(7, 7, 7, 9, 9, 9)$ | 30.71 dB | 26.41 dB | 26.73 dB | 29.05 dB | 26.45 dB | 28.20 dB |
| Peppers | 20% | 18.68 dB | $(5, 5, 5, 5, 5, 5)$ | 33.01 dB | 22.94 dB | 28.31 dB | 30.81 dB | 29.75 dB | 30.85 dB |
| | 30% | 16.93 dB | $(5, 5, 5, 7, 7, 7)$ | 31.77 dB | 25.21 dB | 28.06 dB | 30.30 dB | 28.17 dB | 29.70 dB |
| | 40% | 15.68 dB | $(7, 7, 7, 7, 7, 7)$ | 30.50 dB | 25.89 dB | 26.92 dB | 29.43 dB | 26.78 dB | 28.32 dB |
| | 50% | 14.68 dB | $(7, 7, 7, 9, 9, 9)$ | 29.72 dB | 25.50 dB | 26.15 dB | 27.96 dB | 25.26 dB | 27.36 dB |
| Barbara | 20% | 18.42 dB | $(5, 5, 5, 5, 5, 5)$ | 27.26 dB | 22.86 dB | 25.07 dB | 25.16 dB | 27.59 dB | 29.54 dB |
| | 30% | 16.67 dB | $(5, 5, 5, 7, 7, 7)$ | 26.09 dB | 24.34 dB | 25.92 dB | 24.66 dB | 26.15 dB | 28.26 dB |
| | 40% | 15.43 dB | $(7, 7, 7, 7, 7, 7)$ | 24.81 dB | 24.28 dB | 25.51 dB | 24.12 dB | 25.11 dB | 27.05 dB |
| | 50% | 14.45 dB | $(7, 7, 7, 9, 9, 9)$ | 24.29 dB | 23.81 dB | 24.40 dB | 23.41 dB | 23.86 dB | 26.64 dB |
| Boat | 20% | 18.56 dB | $(5, 5, 5, 5, 5, 5)$ | 32.36 dB | 22.59 dB | 26.46 dB | 31.85 dB | 28.35 dB | 29.45 dB |
| | 30% | 16.77 dB | $(5, 5, 5, 5, 5, 5)$ | 30.66 dB | 25.07 dB | 26.79 dB | 30.85 dB | 27.05 dB | 28.29 dB |
| | 40% | 15.52 dB | $(5, 5, 5, 7, 7, 7)$ | 29.14 dB | 25.40 dB | 26.08 dB | 29.51 dB | 25.85 dB | 27.26 dB |
| | 50% | 14.55 dB | $(7, 7, 7, 7, 7, 7)$ | 28.10 dB | 25.09 dB | 25.38 dB | 27.73 dB | 24.90 dB | 26.61 dB |

TABLE IV

RESULTS OF THE KERNELIZED DENOISING METHOD ON VARIOUS IMAGES CORRUPTED BY GAUSSIAN NOISE.

| Image | Noise | noisy PSNR | Kernel Denoising | | BiShrink | BLS-GSM | K-SVD | SKR $L_1$ | SKR | BM3D |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | $\boldsymbol{N}$ | PSNR | | | | | | |
| Lena | $s = 10$ | 28.12 dB | $(5, 5, 5, 5, 5, 5)$ | 33.98 dB | 34.33 dB | 35.60 dB | 35.47 dB | 32.66 dB | 35.32 dB | 35.93 dB |
| | $s = 20$ | 22.14 dB | $(5, 5, 5, 7, 7, 7)$ | 31.12 dB | 31.17 dB | 32.65 dB | 32.36 dB | 29.23 dB | 32.62 dB | 33.00 dB |
| | $s = 30$ | 18.72 dB | $(7, 7, 7, 7, 7, 7)$ | 29.11 dB | 29.35 dB | 30.50 dB | 30.30 dB | 26.60 dB | 30.71 dB | 31.21 dB |
| Peppers | $s = 10$ | 28.26 dB | $(5, 5, 5, 5, 5, 5)$ | 32.44 dB | 33.62 dB | 34.71 dB | 34.84 dB | 29.99 dB | 34.40 dB | 35.03 dB |
| | $s = 20$ | 22.32 dB | $(5, 5, 5, 7, 7, 7)$ | 30.38 dB | 30.67 dB | 31.90 dB | 31.93 dB | 27.99 dB | 31.84 dB | 32.51 dB |
| | $s = 30$ | 18.93 dB | $(7, 7, 7, 7, 7, 7)$ | 28.60 dB | 28.71 dB | 29.83 dB | 29.90 dB | 26.01 dB | 29.92 dB | 20.73 dB |
| Barbara | $s = 10$ | 28.11 dB | $(5, 5, 5, 5, 5, 5)$ | 27.60 dB | 32.46 dB | 34.02 dB | 34.79 dB | 25.41 dB | 33.37 dB | 35.37 dB |
| | $s = 20$ | 22.16 dB | $(5, 5, 5, 7, 7, 7)$ | 26.01 dB | 28.56 dB | 30.27 dB | 31.12 dB | 24.36 dB | 30.06 dB | 32.10 dB |
| | $s = 30$ | 18.73 dB | $(7, 7, 7, 7, 7, 7)$ | 24.06 dB | 26.46 dB | 28.05 dB | 28.61 dB | 23.17 dB | 27.85 dB | 30.01 dB |
| Boat | $s = 10$ | 28.13 dB | $(5, 5, 5, 5, 5, 5)$ | 31.78 dB | 33.29 dB | 34.52 dB | 34.77 dB | 30.95 dB | 34.03 dB | 35.08 dB |
| | $s = 20$ | 22.19 dB | $(5, 5, 5, 7, 7, 7)$ | 29.25 dB | 29.68 dB | 30.90 dB | 31.12 dB | 28.29 dB | 30.83 dB | 31.65 dB |
| | $s = 30$ | 18.73 dB | $(7, 7, 7, 7, 7, 7)$ | 27.34 dB | 27.79 dB | 28.90 dB | 28.93 dB | 25.95 dB | 28.79 dB | 29.72 dB |

TABLE V

RESULTS OF THE KERNELIZED DENOISING METHOD ON VARIOUS IMAGES CORRUPTED BY UNIFORM NOISE.

| Image | Noise | noisy PSNR | Kernel Denoising | | BiShrink | K-SVD | SKR $L_1$ | SKR | BM3D |
|---|---|---|---|---|---|---|---|---|---|
| | | | $\boldsymbol{N}$ | PSNR | | | | | |
| lena | $\pm 20$ | 26.88 dB | $(5, 5, 5, 5, 5, 5)$ | 33.00 dB | 33.66 dB | 34.33 dB | 31.20 dB | 34.73 dB | 34.99 dB |
| | $\pm 30$ | 23.36 dB | $(5, 5, 5, 7, 7, 7)$ | 30.81 dB | 31.84 dB | 32.34 dB | 28.82 dB | 33.20 dB | 33.51 dB |
| | $\pm 40$ | 20.85 dB | $(7, 7, 7, 7, 7, 7)$ | 29.41 dB | 30.51 dB | 31.00 dB | 26.82 dB | 32.03 dB | 32.01 dB |
| Peppers | $\pm 20$ | 27.00 dB | $(5, 5, 5, 5, 5, 5)$ | 31.78 dB | 33.05 dB | 33.89 dB | 29.13 dB | 33.96 dB | 34.42 dB |
| | $\pm 30$ | 23.50 dB | $(5, 5, 5, 7, 7, 7)$ | 30.28 dB | 31.31 dB | 32.10 dB | 27.65 dB | 32.38 dB | 32.94 dB |
| | $\pm 40$ | 21.05 dB | $(7, 7, 7, 7, 7, 7)$ | 28.81 dB | 29.96 dB | 30.68 dB | 26.14 dB | 31.17 dB | 31.57 dB |
| Barbara | $\pm 20$ | 26.87 dB | $(5, 5, 5, 5, 5, 5)$ | 27.16 dB | 31.60 dB | 33.43 dB | 24.94 dB | 32.70 dB | 34.13 dB |
| | $\pm 30$ | 23.35 dB | $(5, 5, 5, 7, 7, 7)$ | 25.95 dB | 29.24 dB | 31.02 dB | 24.31 dB | 30.75 dB | 32.66 dB |
| | $\pm 40$ | 20.87 dB | $(7, 7, 7, 7, 7, 7)$ | 24.19 dB | 27.72 dB | 29.76 dB | 23.47 dB | 29.35 db | 30.90 dB |
| Boat | $\pm 20$ | 26.88 dB | $(5, 5, 5, 5, 5, 5)$ | 31.15 dB | 32.48 dB | 33.64 dB | 29.81 dB | 33.37 dB | 34.09 dB |
| | $\pm 30$ | 23.39 dB | $(5, 5, 5, 7, 7, 7)$ | 29.38 dB | 30.40 dB | 30.94 dB | 27.98 dB | 31.49 dB | 32.11 dB |
| | $\pm 40$ | 20.92 dB | $(7, 7, 7, 7, 7, 7)$ | 27.47 dB | 29.02 dB | 30.11 dB | 26.21 dB | 30.17 dB | 30.40 dB |

TABLE VI

RESULTS OF THE KERNELIZED DENOISING METHOD ON VARIOUS IMAGES CORRUPTED BY MIXED NOISE.

| Image | Noise | noisy PSNR | Kernel Denoising | | BiShrink | K-SVD | SKR $L_1$ | SKR | BM3D |
|---|---|---|---|---|---|---|---|---|---|
| | | | $\boldsymbol{N}$ | denoised PSNR | | | | | |
| Lena | mixed 1 | 17.98 dB | $(5, 5, 5, 7, 7, 7)$ | 32.27 dB | 25.30 dB | 27.51 dB | 31.14 dB | 30.02 dB | 30.65 dB |
| | mixed 2 | 15,68 dB | $(5, 5, 5, 7, 7, 7)$ | 29.20 dB | 27.11 dB | 27.58 dB | 26.90 dB | 28.32 dB | 29.10 dB |
| | mixed 3 | 26.89 dB | $(5, 5, 5, 7, 7, 7)$ | 33.42 dB | 33.68 dB | 34.21 dB | 31.97 dB | 34.70 dB | 34.10 dB |
| | mixed 4 | 21.18 dB | $(5, 5, 5, 5, 5, 5)$ | 34.48 dB | 23.72 dB | 29.56 dB | 33.67 dB | 31.96 dB | 32.14 dB |
| | mixed 5 | 20.35 dB | $(5, 5, 5, 7, 7, 7)$ | 32.74 dB | 25.99 dB | 29.61 dB | 31.36 dB | 31.48 dB | 31.77 dB |
| Boat | mixed 1 | 18.14 dB | $(5, 5, 5, 7, 7, 7)$ | 30.27 dB | 24.58 dB | 36.92 dB | 29.67 dB | 28.13 dB | 29.21 dB |
| | mixed 2 | 15.82 dB | $(5, 5, 5, 7, 7, 7)$ | 26.90 dB | 25.62 dB | 26.56 dB | 26.18 dB | 26.36 dB | 27.6 dB |
| | mixed 3 | 26.90 dB | $(5, 5, 5, 7, 7, 7)$ | 31.56 dB | 32.51 dB | 33.54 dB | 30.5 dB | 33.42 dB | 32.57 dB |
| | mixed 4 | 21.20 dB | $(5, 5, 5, 5, 5, 5)$ | 32.28 dB | 23.55 dB | 28.11 dB | 31.56 dB | 29.98 dB | 30.5 dB |
| | mixed 5 | 20.46 dB | $(5, 5, 5, 7, 7, 7)$ | 30.70 dB | 25.53 dB | 28.20 dB | 29.86 dB | 29.61 dB | 30.1 dB |