Kernel Regression for Image Processing and Reconstruction

Hiroyuki Takeda, *Student Member, IEEE*, Sina Farsiu, *Member, IEEE*, and Peyman Milanfar, *Senior Member, IEEE*

***Abstract*—In this paper, we make contact with the field of nonparametric statistics and present a development and generalization of tools and results for use in image processing and reconstruction. In particular, we adapt and expand kernel regression ideas for use in image denoising, upscaling, interpolation, fusion, and more. Furthermore, we establish key relationships with some popular existing methods and show how several of these algorithms, including the recently popularized bilateral filter, are special cases of the proposed framework. The resulting algorithms and analyses are amply illustrated with practical examples.**

***Index Terms*—Bilateral filter, denoising, fusion, interpolation, irregularly sampled data, kernel function, kernel regression, local polynomial, nonlinear filter, nonparametric, scaling, spatially adaptive, super-resolution.**

I. Introduction

THE ease of use and cost effectiveness have contributed to the growing popularity of digital imaging systems. However, inferior spatial resolution with respect to the traditional film cameras is still a drawback. The apparent aliasing effects often seen in digital images are due to the limited number of CCD pixels used in commercial digital cameras. Using denser CCD arrays (with smaller pixels) not only increases the production cost, but also results in noisier images. As a cost-efficient alternate, image processing methods have been exploited through the years to improve the quality of digital images. In this paper, we focus on regression methods that attempt to recover the noiseless high-frequency information corrupted by the limitations of imaging systems, as well as the degradations processes such as compression.

We study regression, as a tool not only for interpolation of regularly sampled frames (up-sampling), but also for restoration and enhancement of noisy and possibly irregularly sampled images. Fig. 1(a) illustrates an example of the former case, where we opt to upsample an image by a factor of two in each direction. Fig. 1(b) illustrates an example of the latter case, where an irregularly sampled noisy image is to be interpolated onto a high resolution grid. Besides inpainting applications [1], interpolation of irregularly sampled image data is essential for applications such as multiframe super-resolution, where several low-resolution images are fused (interlaced) onto a high-resolution grid [2]. Fig. 2 represents a block diagram representation of such super-resolution algorithm. We note that "denoising" is a special case of the regression problem where samples at all desired pixel locations are given [illustrated in Fig. 1(c)], but these samples are corrupted, and are to be restored.

Contributions of this paper are the following. 1) We describe and propose *kernel regression* as an effective tool for both denoising and interpolation in image processing, and establish its relation with some popular existing techniques. 2) We propose a novel adaptive generalization of kernel regression with excellent results in both denoising and interpolation (for single or multiframe) applications.

This paper is structured as follows. In Section II, we will briefly describe the kernel regression idea for univariate data, and review several related concepts. Furthermore, the classic framework of kernel regression for bivariate data and intuitions on related concepts will also be presented. In Section III, we extend and generalize this framework to derive a novel data-adapted kernel regression method. Simulation on both real and synthetic data are presented in Section IV, and Section V concludes this paper. We begin with a brief introduction to the notion of kernel regression.

II. Classic Kernel Regression and Its Properties

In this section, we formulate the classical kernel regression method, provide some intuitions on computational efficiency, as well as weaknesses of this method, and motivate the development of more powerful regression tools in the next section.

A. Kernel Regression in 1-D

Classical parametric image processing methods rely on a specific model of the signal of interest and seek to compute the parameters of this model in the presence of noise. Examples of this approach are presented in diverse problems ranging from denoising to upscaling and interpolation. A generative model based upon the estimated parameters is then produced as the best estimate of the underlying signal.

In contrast to the parametric methods, nonparametric methods rely on the data itself to dictate the structure of the model, in which case this implicit model is referred to as *a regression function* [3]. With the relatively recent emergence of machine learning methods, kernel methods have become well-known and used frequently for pattern detection and discrimination problems [4]. Surprisingly, it appears that the corresponding ideas in nonparametric estimation—what we

Manuscript received December 15, 2005; revised August 1, 2006. This work was supported in part by the U.S. Air Force under Grant F49620-03-1-0387 and in part by the National Science Foundation Science and Technology Center for Adaptive Optics, managed by the University of California at Santa Cruz under Cooperative Agreement AST-9876783. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Tamas Sziranyi.

The authors are with the Electrical Engineering Department, University of California Santa Cruz, Santa Cruz CA 95064 USA (e-mail: htakeda@soe.ucsc.edu; farsiu@soe.ucsc.edu; milanfar@soe.ucsc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Software implementation available at http://www.soe.ucsc.edu/~htakeda.

Digital Object Identifier 10.1109/TIP.2006.888330

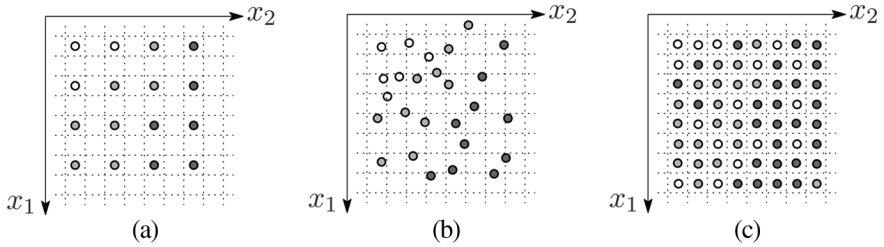


Fig. 1. (a) Interpolation of regularly sampled data. (b) Reconstruction from irregularly sampled data. (c) Denoising.

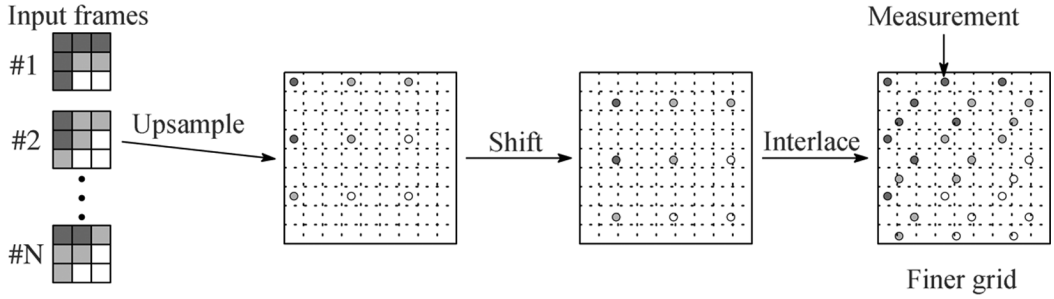


Fig. 2. Image fusion often yields us irregularly sampled data.

call here *kernel regression*—are not widely recognized or used in the image and video processing literature. Indeed, in the last decade, several concepts related to the general theory we promote here have been rediscovered in different guises, and presented under different names such as *normalized convolution* [5], [6], *bilateral filter* [7], [8], *edge-directed interpolation* [9], and *moving least squares* [10]. Later in this paper, we shall say more about some of these concepts and their relation to the general regression theory. To simplify this introductory presentation, we treat the 1-D case where the measured data are given by

$$y_i = z(x_i) + \varepsilon_i, \quad i = 1, 2, \ldots, P \tag{1}$$

where $z(\cdot)$ is the (hitherto unspecified) regression function and ε_is are the independent and identically distributed zero mean noise values (with otherwise no particular statistical distribution assumed). As such, kernel regression provides a rich mechanism for computing point-wise estimates of the function with minimal assumptions about global signal or noise models.

While the specific form of $z(x_i)$ may remain unspecified, if we assume that it is locally smooth to some order N, then in order to estimate the value of the function at any point x given the data, we can rely on a generic local expansion of the function about this point. Specifically, if x is near the sample at x_i, we have the N-term Taylor series

$$\begin{aligned} z(x_i) &\approx z(x) + z'(x)(x_i - x) + \frac{1}{2!}z''(x)(x_i - x)^2 \\ &\quad + \cdots + \frac{1}{N!}z^{(N)}(x)(x_i - x)^N \qquad (2) \\ &= \beta_0 + \beta_1(x_i - x) + \beta_2(x_i - x)^2 \\ &\quad + \cdots + \beta_N(x_i - x)^N. \qquad (3) \end{aligned}$$

The above suggests that if we now think of the Taylor series as a local representation of the regression function, estimating the parameter β_0 can yield the desired (local) estimate of the regression function based on the data.[1] Indeed, the parameters $\{\beta_n\}_{n=1}^N$ will provide localized information on the nth *derivatives* of the regression function. Naturally, since this approach is based on *local* approximations, a logical step to take is to estimate the parameters $\{\beta_n\}_{n=0}^N$ from the data while giving the nearby samples higher weight than samples farther away. A least-squares formulation capturing this idea is to solve the following optimization problem:

$$\min_{\{\beta_n\}} \sum_{i=1}^{P} \left[y_i - \beta_0 - \beta_1(x_i - x) - \beta_2(x_i - x)^2 - \cdots - \beta_N(x_i - x)^N\right]^2 \frac{1}{h}K\left(\frac{x_i - x}{h}\right) \tag{4}$$

where $K(\cdot)$ is the *kernel function* which penalizes distance away from the local position where the approximation is centered, and the *smoothing parameter* h (also called the "bandwidth") controls the strength of this penalty [3]. In particular, the function K is a symmetric function which attains its maximum at zero, satisfying

$$\int_{R^1} tK(t)dt = 0, \quad \int_{R^1} t^2K(t)dt = c \tag{5}$$

where c is some constant value.[2] The choice of the particular form of the function K is open, and may be selected as a Gaussian, exponential, or other forms which comply with the above constraints. It is known [11] that for the case of classic regression the choice of the kernel has only a small effect on the accuracy of estimation, and, therefore, preference is given

[1]Indeed the local approximation can be built upon bases other than polynomials [3].

[2]Basically, the only conditions needed for the regression framework are that the kernel function be nonnegative, symmetric, and unimodal. For instance, unlike the kernel *density* estimation problems [11], even if the kernel weights in (6) do not sum up to one, the term in the denominator will normalize the final result.

to the differentiable kernels with low computational complexity such as the Gaussian kernel. The effect of kernel choice for the data-adaptive algorithms presented in Section III is an interesting avenue of research, which is outside the scope of this paper and part of our ongoing work.

Several important points are worth making here. First, the above structure allows for tailoring the estimation problem to the *local* characteristics of the data, whereas the standard parametric model is generally intended as a more global fit. Second, in the estimation of the local structure, higher weight is given to the nearby data as compared to samples that are farther away from the center of the analysis window. Meanwhile, this approach does not specifically require that the data to follow a regular or equally spaced sampling structure. More specifically, so long as the samples are near the point x, the framework is valid. Again this is in contrast to the general parametric approach which generally either does not directly take the location of the data samples into account, or relies on regular sampling over a grid. Third, and no less important as we will describe later, the proposed approach is both useful for *denoising*, and equally viable for *interpolation* of sampled data at points where no actual samples exist. Given the above observations, the kernel-based methods are well suited for a wide class of image processing problems of practical interest.

Returning to the estimation problem based upon (4), one can choose the order N to effect an increasingly more complex local approximation of the signal. In the nonparametric statistics literature, locally constant, linear, and quadratic approximations (corresponding to $N = 0, 1, 2$) have been considered most widely [3], [12]–[14]. In particular, choosing $N = 0$, a locally linear, adaptive, filter is obtained, which is known as the *Nadaraya–Watson* estimator (NWE) [15]. Specifically, this estimator has the form

$$\hat{z}(x) = \frac{\sum_{i=1}^{P} K_h(x_i - x) y_i}{\sum_{i=1}^{P} K_h(x_i - x)}, \quad K_h(t) = \frac{1}{h} K\left(\frac{t}{h}\right). \tag{6}$$

The NWE is the simplest manifestation of an adaptive filter resulting from the kernel regression framework. As we shall see later in Section III, the *bilateral filter* [7], [8] can be interpreted as a generalization of NWE with a modified kernel definition.

Of course, higher order approximations $(N > 0)$ are also possible. The choice of order in parallel with the smoothness (h) affects the bias and variance of the estimate. Mathematical expression for bias and variance can be found in [16] and [17], and, therefore, here, we only briefly review their properties. In general, lower order approximates, such as NWE, result in smoother images (large bias and small variance) as there are fewer degrees of freedom. On the other hand, overfitting happens in regressions using higher orders of approximation, resulting in small bias and large estimation variance. We also note that smaller values for h result in small bias and consequently large variance in estimates. Optimal order and smoothing parameter selection procedures are studied in [10].

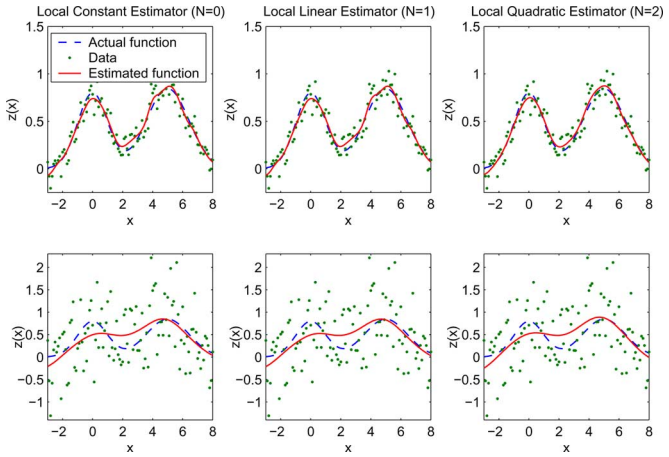


Fig. 3. Examples of local polynomial regression on an equally spaced data set. The signals in the first and second rows are contaminated with the Gaussian noise of SNR $= 9$ [dB] and -6.5 [dB], respectively. The dashed lines, solid lines, and dots represent the actual function, estimated function, and the noisy data, respectively. The columns from left to right show the constant, linear, and quadratic interpolation results. Corresponding RMSEs for the first row experiments are 0.0364, 0.0364, 0.0307 and for the second row are as 0.1697, 0.1708, 0.1703.

The performance of kernel regressors of different order are compared in the illustrative examples of Fig. 3. In the first experiment, illustrated in the first row, a set of moderately noisy[3] regularly sampled data are used to estimate the underlying function. As expected, the computationally more complex high-order interpolation $(N = 2)$ results in a better estimate than the lower-ordered interpolators ($N = 0$ or $N = 1$). The presented quantitative comparison of the root-mean-square errors (RMSE) supports this claim. The second experiment, illustrated in the second row, shows that for the heavily noisy data sets (variance of the additive Gaussian noise 0.5), the performance of lower order regressors is better. Note that the performance of the $N = 0$ and $N = 1$-ordered estimators for these equally spaced sampled experiments are identical. In Section II-D, we study this property in more detail.

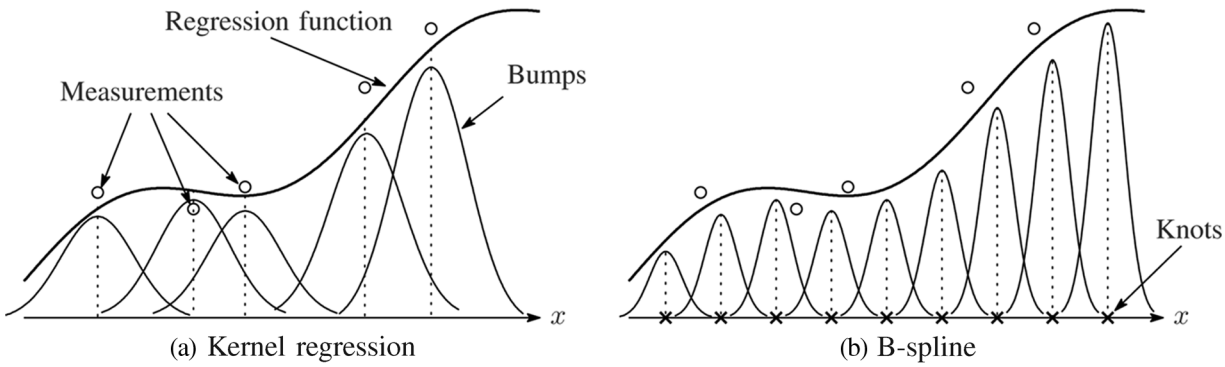
B. Related Regression Methods

In addition to kernel regression methods we are advocating, there are several other effective regression methods such as B-spline interpolation [18], orthogonal series [13], [19], cubic spline interpolation [20], and spline smoother [13], [18], [21]. We briefly review some of these methods in this section.

In the orthogonal series methods, instead of using Taylor series, the regression function z can be represented by a linear combination of other basis functions, such as Legendre polynomials, wavelet bases, Hermite polynomials [10], and so on. In the 1-D case, such a model, in general, is represented as

$$z(x) = \sum_{j=0}^{N} \beta_j \varphi_j(x). \tag{7}$$

[3]Variance of the additive Gaussian noise is 0.1. Smoothing parameter is chosen by the cross validation method (Section II-E).



(a) Kernel regression (b) B-spline

Fig. 4. Comparison of the position of knots in (a) kernel regression and (b) classical B-spline methods.

The coefficients $\{\beta_j\}_{j=0}^{N}$ are the unknown parameters we want to estimate. We refer the interested reader to [13, pp. 104–107] which offers further examples and insights.

Following the notation used in the previous section, the B-spline regression is expressed as the linear combination of shifted spline functions $B^q(\cdot)$

$$z(x) = \sum_k \beta_k B^q(x-k) \tag{8}$$

where the q^{th}-order B-spline function is defined as a $q+1$ times convolution of the zero-order B spline [18]

$$B^q(x) = \underbrace{B^0(x) * B^0(x) * \cdots * B^0(x)}_{q+1}$$

where

$$B^0(x) = \begin{cases} 1, & -\frac{1}{2} < x < \frac{1}{2} \\ \frac{1}{2}, & |x| = \frac{1}{2} \\ 0, & \text{else} \end{cases}. \tag{9}$$

The scalar k in (8), often referred to as the *knot*, defines the center of a spline. Least squares is usually exploited to estimate the B-spline coefficients $\{\beta_k\}$.

The B-spline interpolation method bears some similarities to the kernel regression method. One major difference between these methods is in the number and position of the knots as illustrated in Fig. 4. While in the classical B-Spline method the knots are located in equally spaced positions, in the case of kernel regression the knots are implicitly located on the sample positions. A related method, the *nonuniform rational B-spline* is also proposed [22] to address this shortcoming of the classical B-Spline method by irregularly positioning the knots with respect to the underlying signal.

Cubic spline interpolation technique is one of the most popular members of the spline interpolation family which is based on fitting a polynomial between any pair of consecutive data. Assuming that the second derivative of the regression function exists, cubic spline interpolator is defined as

$$z(x) = \beta_0(i) + \beta_1(i)(x_i - x) + \beta_2(i)(x_i - x)^2 + \beta_3(i)(x_i - x)^3, \quad x \in [x_i, x_{i+1}] \tag{10}$$

where under the following boundary conditions:

$$z(x)|_{x=-x_i} = z(x)|_{x=+x_i}, \quad z'(x)|_{x=-x_i} = z'(x)|_{x=+x_i}$$
$$z''(x)|_{x=-x_i} = z''(x)|_{x=+x_i}, \quad z''(x_1) = z''(x_P) = 0 \tag{11}$$

all the coefficients $(\beta_n(i)\text{s})$ can be uniquely defined [20].

Note that an estimated curve by cubic spline interpolation passes through all data points which is ideal for the noiseless data case. However, in most practical applications, data is contaminated with noise and, therefore, such perfect fits are no longer desirable. Consequently, a related method called spline smoother has been proposed [18]. In the spline smoothing method, the afore-mentioned hard conditions are replaced with soft ones, by introducing them as Bayesian priors which penalize rather than constrain nonsmoothness in the interpolated images. A popular implementation of the spline smoother [18] is given by

$$\hat{z}(x) = \arg\min_{z(x)} \left[\sum_{i=1}^{P} \{y_i - z(x_i)\}^2 + \lambda \|z''\|_2^2 \right] \tag{12}$$

where

$$\|z''\|_2^2 = \int \{z''(x)\}^2 \, dx$$

and $z(x_i)$ can be replaced by either (8) or any orthogonal series (e.g., [23]), and λ is the regularization parameter. Note that assuming a continuous sample density function, the solution to this minimization problem is equivalent to NWE (6), with the following kernel function and smoothing parameter:

$$K(t) = \frac{1}{2} \exp\left(-\frac{|t|}{\sqrt{2}}\right) \sin\left(\frac{|t|}{\sqrt{2}} + \frac{\pi}{4}\right)$$
$$h(x_i) = \left(\frac{\lambda}{Pf(x_i)}\right)^{\frac{1}{4}} \tag{13}$$

where f is the density of samples [13], [24]. Therefore, spline smoother is a special form of kernel regression.

In Section IV, we compare the performance of the spline smoother with the proposed kernel regression method, and, later in Section V, we give some intuitions for the outstanding performance of the kernel regression methods.

The authors of [9] propose another edge-directed interpolation method for upsampling regularly sampled images. The interpolation is implemented by weighted averaging the four immediate neighboring pixels in a regular upsampling scenario where the filter coefficients are estimated using the classic covariance estimation method [25].

The normalized convolution method presented in [5] is very similar to the classic kernel regression method (considering a different basis function), which we show is a simplified version of the adaptive kernel regression introduced in Section III. An edge-adaptive version of this work is also very recently proposed in [6].

We note that other popular edge-adaptive denoising or interpolation techniques are available in the literature, such as the PDE-based regression methods [26]–[28]. A denoising experiment using the anisotropic diffusion (the second-order PDE) method of [26] is presented in Section IV; however, a detailed discussion and comparison of all these diverse methods is outside the scope of this paper.

C. Kernel Regression Formulation in 2-D

Similar to the 1-D case in (1), the data measurement model in 2-D is given by

$$y_i = z(\mathbf{x}_i) + \varepsilon_i, \quad i = 1, \cdots, P \tag{14}$$

where the coordinates of the measured data y_i is now the 2×1 vector $\mathbf{x}_i$. Correspondingly, the local expansion of the regression function is given by

$$\begin{aligned} z(\mathbf{x}_i) &= z(\mathbf{x}) + \{\nabla z(\mathbf{x})\}^T (\mathbf{x}_i - \mathbf{x}) \\ &\quad + \frac{1}{2}(\mathbf{x}_i - \mathbf{x})^T \{\mathcal{H} z(\mathbf{x})\} (\mathbf{x}_i - \mathbf{x}) + \cdots \\ &= z(\mathbf{x}) + \{\nabla z(\mathbf{x})\}^T (\mathbf{x}_i - \mathbf{x}) \\ &\quad + \frac{1}{2}\mathrm{vec}^T \{\mathcal{H} z(\mathbf{x})\} \mathrm{vec}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} + \cdots \end{aligned} \tag{15}$$

where ∇ and $\mathcal{H}$ are the gradient (2×1) and Hessian (2×2) operators, respectively, and $\mathrm{vec}(\cdot)$ is the vectorization operator, which lexicographically orders a matrix into a vector. Defining $\mathrm{vech}(\cdot)$ as the half-vectorization operator of the "lower-triangular" portion of a symmetric matrix, e.g.,

$$\begin{aligned} \mathrm{vech}\left(\begin{bmatrix} a & b \\ b & d \end{bmatrix}\right) &= [a \;\; b \;\; d]^T \\ \mathrm{vech}\left(\begin{bmatrix} a & b & c \\ b & e & f \\ c & f & i \end{bmatrix}\right) &= [a \;\; b \;\; c \;\; e \;\; f \;\; i]^T \end{aligned} \tag{16}$$

and considering the symmetry of the Hessian matrix, (15) simplifies to

$$z(\mathbf{x}_i) = \beta_0 + \boldsymbol{\beta}_1^T (\mathbf{x}_i - \mathbf{x}) + \boldsymbol{\beta}_2^T \mathrm{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} + \cdots \tag{17}$$

Then, a comparison of (15) and (17) suggests that $\beta_0 = z(\mathbf{x})$ is the pixel value of interest and the vectors $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are

$$\boldsymbol{\beta}_1 = \nabla z(\mathbf{x}) = \left[\frac{\partial z(\mathbf{x})}{\partial x_1}, \;\; \frac{\partial z(\mathbf{x})}{\partial x_2}\right]^T \tag{18}$$

$$\boldsymbol{\beta}_2 = \frac{1}{2}\left[\frac{\partial^2 z(\mathbf{x})}{\partial x_1^2}, \;\; 2\frac{\partial^2 z(\mathbf{x})}{\partial x_1 \partial x_2}, \;\; \frac{\partial^2 z(\mathbf{x})}{\partial x_2^2}\right]^T. \tag{19}$$

As in the case of univariate data, the $\boldsymbol{\beta}_n$s are computed from the following optimization problem:

$$\min_{\{\boldsymbol{\beta}_n\}} \sum_{i=1}^{P} \Big[y_i - \beta_0 - \boldsymbol{\beta}_1^T (\mathbf{x}_i - \mathbf{x}) - \boldsymbol{\beta}_2^T \mathrm{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} - \cdots\Big]^2 K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) \tag{20}$$

with

$$K_{\mathbf{H}}(\mathbf{t}) = \frac{1}{\det(\mathbf{H})} K(\mathbf{H}^{-1}\mathbf{t}) \tag{21}$$

where K is the 2-D realization of the kernel function, and $\mathbf{H}$ is the 2×2 smoothing matrix, which will be studied more carefully later in this section. It is also possible to express (20) in a matrix form as a weighted least-squares optimization problem [10], [29]

$$\begin{aligned} \hat{\mathbf{b}} &= \arg\min_{\mathbf{b}} \|\mathbf{y} - \mathbf{X}_{\mathbf{x}}\mathbf{b}\|^2_{\mathbf{W}_{\mathbf{x}}} \\ &= \arg\min_{\mathbf{b}} (\mathbf{y} - \mathbf{X}_{\mathbf{x}}\mathbf{b})^T \mathbf{W}_{\mathbf{x}} (\mathbf{y} - \mathbf{X}_{\mathbf{x}}\mathbf{b}) \end{aligned} \tag{22}$$

where

$$\mathbf{y} = [y_1, y_2, \cdots, y_P]^T, \;\; \mathbf{b} = \left[\beta_0, \boldsymbol{\beta}_1^T, \cdots, \boldsymbol{\beta}_N^T\right]^T \tag{23}$$

$$\mathbf{W}_{\mathbf{x}} = \mathrm{diag}\left[K_{\mathbf{H}}(\mathbf{x}_1 - \mathbf{x}), K_{\mathbf{H}}(\mathbf{x}_2 - \mathbf{x}) \;\cdots, K_{\mathbf{H}}(\mathbf{x}_P - \mathbf{x})\right] \tag{24}$$

$$\mathbf{X}_{\mathbf{x}} = \begin{bmatrix} 1 & (\mathbf{x}_1 - \mathbf{x})^T & \mathrm{vech}^T\left\{(\mathbf{x}_1 - \mathbf{x})(\mathbf{x}_1 - \mathbf{x})^T\right\} & \cdots \\ 1 & (\mathbf{x}_2 - \mathbf{x})^T & \mathrm{vech}^T\left\{(\mathbf{x}_2 - \mathbf{x})(\mathbf{x}_2 - \mathbf{x})^T\right\} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 1 & (\mathbf{x}_P - \mathbf{x})^T & \mathrm{vech}^T\left\{(\mathbf{x}_P - \mathbf{x})(\mathbf{x}_P - \mathbf{x})^T\right\} & \cdots \end{bmatrix} \tag{25}$$

with "diag" defining a diagonal matrix.

Regardless of the estimator order (N), since our primary interest is to compute an estimate of the image (pixel values), the necessary computations are limited to the ones that estimate the parameter β_0. Therefore, the least-squares estimation is simplified to

$$\hat{z}(\mathbf{x}) = \hat{\beta}_0 = \mathbf{e}_1^T \left(\mathbf{X}_{\mathbf{x}}^T \mathbf{W}_{\mathbf{x}} \mathbf{X}_{\mathbf{x}}\right)^{-1} \mathbf{X}_{\mathbf{x}}^T \mathbf{W}_{\mathbf{x}} \mathbf{y} \tag{26}$$

where $\mathbf{e}_1$ is a column vector with the first element equal to one, and the rest equal to zero. Of course, there is a fundamental difference between computing β_0 for the $N = 0$ case, and using a high-order estimator ($N > 0$) and then effectively discarding all estimated $\boldsymbol{\beta}_n$s except β_0. Unlike the former case,

the latter method computes estimates of pixel values assuming a N^{th}-order locally polynomial structure is present.

D. Equivalent Kernels

In this section, we present a computationally more efficient and intuitive solution to the above kernel regression problem. Study of (26) shows that $\mathbf{X}_{\mathbf{x}}^T\mathbf{W}_{\mathbf{x}}\mathbf{X}_{\mathbf{x}}$ is a $(N+1)\times(N+1)$ block matrix, with the following structure:

$$\mathbf{X}_{\mathbf{x}}^T\mathbf{W}_{\mathbf{x}}\mathbf{X}_{\mathbf{x}} = \begin{bmatrix} s_{11} & \mathbf{s}_{12} & \mathbf{s}_{13} & \cdots \\ \mathbf{s}_{21} & \mathbf{s}_{22} & \mathbf{s}_{23} & \cdots \\ \mathbf{s}_{31} & \mathbf{s}_{32} & \mathbf{s}_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{27}$$

where $\mathbf{s}_{lm}$ is an $l\times m$ matrix (block). The block elements of (27) for orders up to $N=2$ are as follows:

$$s_{11} = \sum_{i=1}^{P} K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) \tag{28}$$

$$\mathbf{s}_{12} = \mathbf{s}_{21}^T = \sum_{i=1}^{P} (\mathbf{x}_i - \mathbf{x})^T K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) \tag{29}$$

$$\mathbf{s}_{22} = \sum_{i=1}^{P} (\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) \tag{30}$$

$$\mathbf{s}_{13} = \mathbf{s}_{31}^T = \sum_{i=1}^{P} \text{vech}^T\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} \cdot K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) \tag{31}$$

$$\mathbf{s}_{23} = \mathbf{s}_{32}^T = \sum_{i=1}^{P} (\mathbf{x}_i - \mathbf{x})\text{vech}^T\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} \cdot K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) \tag{32}$$

$$\mathbf{s}_{33} = \sum_{i=1}^{P} \text{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} \cdot \text{vech}^T\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}). \tag{33}$$

Considering the above shorthand notations, (26) can be represented as a local *linear filtering process*

$$\hat{z}(\mathbf{x}) = \sum_{i=1}^{P} W_i(\mathbf{x}; N, \mathbf{H}) y_i \tag{34}$$

where

$$W_i(\mathbf{x}; 0, \mathbf{H}) = \frac{K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x})}{s_{11}} \tag{35}$$

$$W_i(\mathbf{x}; 1, \mathbf{H}) = \frac{\left\{1 - \mathbf{s}_{12}\mathbf{s}_{22}^{-1}(\mathbf{x}_i - \mathbf{x})\right\} K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x})}{s_{11} - \mathbf{s}_{12}\mathbf{s}_{22}^{-1}\mathbf{s}_{21}} \tag{36}$$

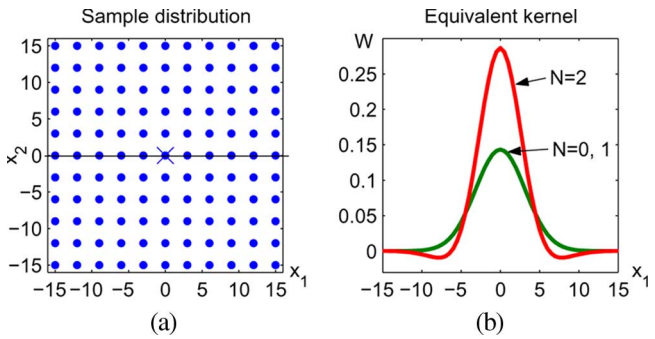


Fig. 5. (a) Uniformly sampled data set. (b) Horizontal slice of the equivalent kernels of orders $N=0$, 1, and 2 for the regularly sampled data in (a). The kernel $K_{\mathbf{H}}$ in (20) is modeled as a Gaussian, with the smoothing matrix $\mathbf{H} = \text{diag}[10, 10]$.

[see (37), shown at the bottom of the page] and

$$\mathbf{S}_{12} = \mathbf{s}_{12} - \mathbf{s}_{13}\mathbf{s}_{33}^{-1}\mathbf{s}_{32}, \quad \mathbf{S}_{22} = \mathbf{s}_{22} - \mathbf{s}_{23}\mathbf{s}_{33}^{-1}\mathbf{s}_{32}$$
$$\mathbf{S}_{13} = \mathbf{s}_{13} - \mathbf{s}_{12}\mathbf{s}_{22}^{-1}\mathbf{s}_{23}, \quad \mathbf{S}_{33} = \mathbf{s}_{33} - \mathbf{s}_{32}\mathbf{s}_{22}^{-1}\mathbf{s}_{23}. \tag{38}$$

Therefore, regardless of the order, the classical kernel regression is nothing but local weighted averaging of data (linear filtering), where the order determines the type and complexity of the weighting scheme. This also suggests that higher order regressions $(N>0)$ are equivalents of the zero-order regression $(N=0)$ but with a more complex kernel function. In other words, to effect the higher order regressions, the original kernel $K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x})$ is modified to yield a newly adapted "equivalent" kernel [3], [17], [30].

To have a better intuition of "equivalent" kernels, we study the example in Fig. 5, which illustrates a uniformly sampled data set, and a horizontal cross section of its corresponding equivalent kernels for the regression orders $N=0$, 1, and 2. The direct result of the symmetry condition (5) on $K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x})$ with uniformly sampled data is that all odd-order "moments" ($\mathbf{s}_{2j,2k+1}$ and $\mathbf{s}_{2k+1,2j}$) consist of elements with values very close to zero. Therefore, as noted in Fig. 5(b), the kernels for $N=0$, and $N=1$ are essentially identical. As this observation holds for all regression orders, *for the regularly sampled data*, the $N=2q-1$ order regression is preferred to the computationally more complex $N=2q$ order regression, as they produce the same results. This property manifests itself in Fig. 5, where the $N=0$ or $N=1$-ordered equivalent kernels are identical.

In the next experiment, we compare the equivalent kernels for an irregularly sampled data set shown in Fig. 6(a). The $N=2$-ordered equivalent kernel for the sample marked with "×," is shown in Fig. 6(b). Fig. 6(c) and (d) shows the horizontal and vertical cross sections of this kernel, respectively. This figure demonstrates the fact that the equivalent kernels tend to adapt themselves to the density of available samples. Also, unlike the uniformly sampled data case, since the odd-order moments are

$$W_i(\mathbf{x}; 2, \mathbf{H}) = \frac{\left[1 - \mathbf{S}_{12}\mathbf{S}_{22}^{-1}(\mathbf{x}_i - \mathbf{x}) - \mathbf{S}_{13}\mathbf{S}_{33}^{-1}\text{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\}\right] K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x})}{s_{11} - \mathbf{S}_{12}\mathbf{S}_{22}^{-1}\mathbf{s}_{21} - \mathbf{S}_{13}\mathbf{S}_{33}^{-1}\mathbf{s}_{31}} \tag{37}$$

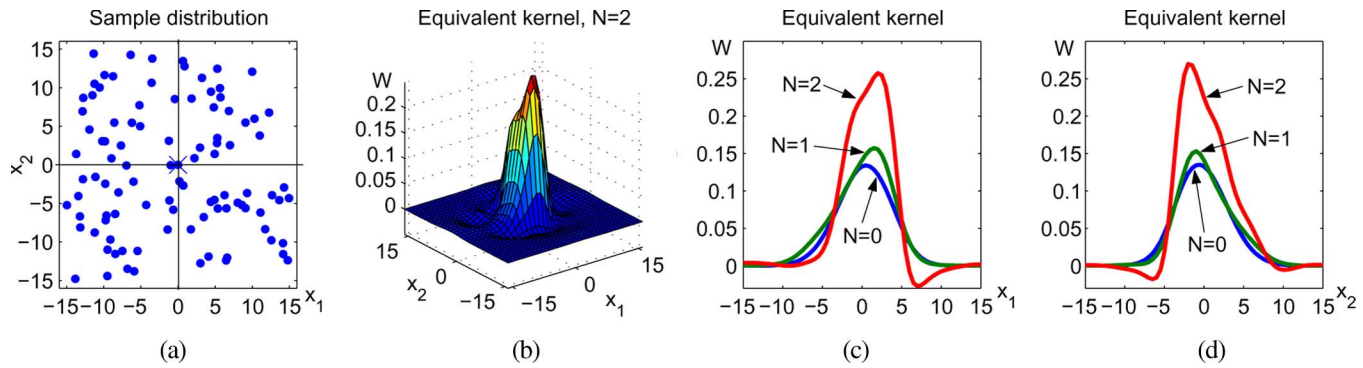


Fig. 6. Equivalent kernels for an irregularly sampled data set shown in (a); (b) is the second-order ($N = 2$) equivalent kernel. The horizontal and vertical slices of the equivalent kernels of different orders ($N = 0, 1, 2$) are compared in (c) and (d), respectively. In this example, the kernel $K_{\mathbf{H}}$ in (20) was modeled as a Gaussian, with the smoothing matrix $\mathbf{H} = \mathrm{diag}[10, 10]$.

nonzero, the $N = 0$ and $N = 1$ equivalent kernels are no longer identical.

E. Smoothing Matrix Selection

The shape of the regression kernel as defined in (21), and, consequently, the performance of the estimator depend on the choice of the smoothing matrix $\mathbf{H}$ [16]. For the bivariate data cases, the smoothing matrix $(\mathbf{H})$ is 2×2, and it extends the support (or footprint) of the regression kernel to contain "enough" samples. As illustrated in Fig. 7, it is reasonable to use smaller kernels in the areas with more available samples, whereas larger kernels are more suitable for the more sparsely sampled areas of the image.

The cross validation "leave-one-out" method [3], [13] is a popular technique for estimating the elements of the local $\mathbf{H}_i$s. However, as the cross validation method is computationally very expensive, we can use a simplified and computationally more efficient model of the smoothing kernel as

$$\mathbf{H}_i = h\mu_i\mathbf{I} \tag{39}$$

where μ_i is a scalar that captures the local density of data samples (nominally set to $\mu_i = 1$) and h is the *global smoothing parameter*.

The *global smoothing parameter* is directly computed from the cross validation method, by minimizing the following cost function

$$\zeta_{\mathrm{cv}}(h) = \frac{1}{P}\sum_{i=1}^{P}\{\hat{z}_{\not i}(\mathbf{x}_i) - y_i\}^2 \tag{40}$$

where $\hat{z}_{\not i}(\mathbf{x}_i)$ is the estimated pixel value without using the ith sample at $\mathbf{x}_i$. To further reduce the computations, rather than leaving a single sample out, we leave out a set of samples (a whole row or column) [31]–[33].

Following [11], the *local density parameter*, μ_i is estimated as follows:

$$\mu_i = \left\{\frac{\hat{f}(\mathbf{x}_i)}{\exp\left(\frac{1}{P}\sum_{i=1}^{P}\log\hat{f}(\mathbf{x}_i)\right)}\right\}^{-\alpha} \tag{41}$$

where the sample density, $\hat{f}(\mathrm{x})$, is measured as

$$\hat{f}(\mathrm{x}) = \frac{1}{P}\sum_{i=1}^{P}K_{\mathbf{H}_i}(\mathbf{x}_i - \mathbf{x}) \tag{42}$$

and α, the *density sensitivity* parameter, is a scalar satisfying[4] $0 < \alpha \leq 1$. Note that $\mathbf{H}_i$ and μ_i are estimated in an iterative fashion. In the first iteration, we initialize with $\mu_i = 1$ and iterate until convergence. Fortunately, the rate of convergence is very fast, and in none of the presented experiments in this paper did we need to use more than two iterations.

In this section, we studied the classic kernel regression method and showed that it is equivalent to an adaptive locally linear filtering process. The price that one pays for using such computationally efficient classic kernel regression methods with diagonal matrix $\mathbf{H}_i$ is the low-quality of reconstruction in the edge areas. Experimental results on this material will be presented later in Section IV. In the next section, we gain even better performance by proposing data-adapted kernel regression methods which take into account not only the spatial sampling density of the data, but also the actual (pixel) values of those samples. These more sophisticated methods lead to locally adaptive *nonlinear* extensions of classic kernel regression.

III. Data-Adapted Kernel Regression

In the previous section, we studied the kernel regression method, its properties, and showed its usefulness for image restoration and reconstruction purposes. One fundamental improvement on the afore-mentioned method can be realized by noting that the local polynomial kernel regression estimates, independent of the order N, are always local *linear* combinations of the data. As such, though elegant, relatively easy to analyze, and with attractive asymptotic properties [16], they suffer from an inherent limitation due to this local linear action on the data. In what follows, we discuss extensions of the kernel regression method that enable this structure to have nonlinear, more effective, action on the data.

Data-adapted kernel regression methods rely on not only the sample location and density, but also on the radiometric properties of these samples. Therefore, the effective size and shape of

[4]In this paper, we choose $\alpha = 0.5$, which is proved in [34] to be an appropriate overall choice for the density sensitivity parameter.

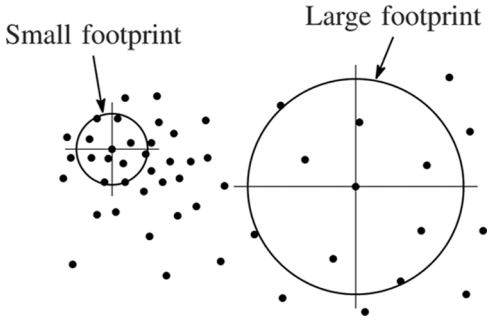


Fig. 7. Smoothing (kernel size) selection by sample density.

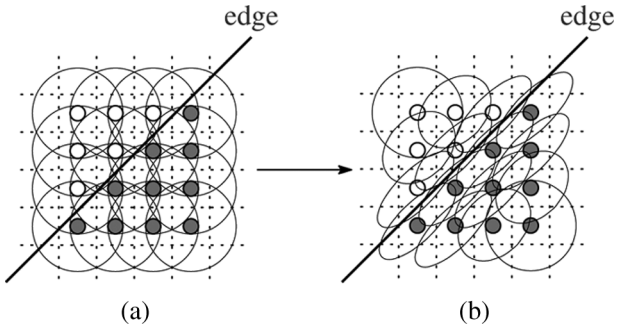


Fig. 8. Kernel spread in a uniformly sampled data set. (a) Kernels in the classic method depend only on the sample density. (b) Data-adapted kernels elongate with respect to the edge.

the regression kernel are adapted locally to image features such as edges. This property is illustrated in Fig. 8, where the classical and adaptive kernel shapes in the presence of an edge are compared.

Data-adapted kernel regression is structured similarly to (20) as an optimization problem

$$\min_{\{\boldsymbol{\beta}_n\}} \sum_{i=1}^{P} \Big[y_i - \beta_0 - \boldsymbol{\beta}_1^T(\mathbf{x}_i - \mathbf{x}) - \boldsymbol{\beta}_2^T \operatorname{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} - \cdots \Big]^2 \cdot K_{\text{adapt}}(\mathbf{x}_i - \mathbf{x}, y_i - y) \tag{43}$$

where the data-adapted kernel function K_{adapt} now depends on the spatial sample locations $\mathbf{x}_i$s and density, as well as the radiometric values y_i of the data.

A. Bilateral Kernel

A simple and intuitive choice of the K_{adapt} is to use separate terms for penalizing the spatial distance between the pixel of interest $\mathbf{x}$ and its neighbors $\{\mathbf{x}_i\}$, and the radiometric "distance" between the corresponding pixels y and $\{y_i\}$

$$K_{\text{adapt}}(\mathbf{x}_i - \mathbf{x}, y_i - y) \equiv K_{\mathbf{H}_s}(\mathbf{x}_i - \mathbf{x}) K_{h_r}(y_i - y) \tag{44}$$

where $\mathbf{H}_s$ $(= h_s\mathbf{I})$ is the spatial smoothing matrix and h_r is the radiometric smoothing scalar. The properties of this adaptive method, which we call *bilateral kernel regression* (for reasons that will become clear shortly), can be better understood by studying the special case of $N = 0$, which results in a data-adapted version of NWE

$$\hat{z}(\mathbf{x}) = \frac{\sum_{i=1}^{P} K_{\mathbf{H}_s}(\mathbf{x}_i - \mathbf{x}) K_{h_r}(y_i - y) y_i}{\sum_{i=1}^{P} K_{\mathbf{H}_s}(\mathbf{x}_i - \mathbf{x}) K_{h_r}(y_i - y)}. \tag{45}$$

Interestingly, this is nothing but the recently well-studied and popular *bilateral filter* [7], [8]. We note that, in general, since the pixel values (y) at an arbitrary position $(\mathbf{x})$ might not be available from the data, the direct application of (44) is limited to the denoising problem. This limitation, however, can be overcome by using an initial estimate of y by an appropriate interpolation technique (e.g., for the experiments of Section IV, we used the second-order classic kernel regression method). Also, it is worth noting that the bilateral kernel choice, along with higher order choices for $N(> 0)$, will lead to generalizations of the bilateral filter, which have not been studied before. We report on these generalizations in [44].

In any event, breaking K_{adapt} into spatial and radiometric terms as utilized in the bilateral case weakens the estimator performance since it limits the degrees of freedom and ignores correlations between positions of the pixels and their values. In particular, we note that, for very noisy data sets, $(y_i - y)$s tend to be large, and, therefore, most radiometric weights are very close to zero, and effectively useless. The following section provides a solution to overcome this drawback of the bilateral kernel.

B. Steering Kernel

The filtering procedure we propose next takes the above ideas one step further, based upon the earlier nonparametric framework. In particular, we observe that the effect of computing $K_{h_r}(y_i - y)$ in (44) is to implicitly measure a function of the local gradient estimated between neighboring values and to use this estimate to weight the respective measurements. As an example, if a pixel is located near an edge, then pixels on the same side of the edge will have much stronger influence in the filtering. With this intuition in mind, we propose a two-step approach where first an initial estimate of the image gradients is made using some kind of gradient estimator (say the second-order classic kernel regression method). Next, this estimate is used to measure the dominant orientation of the local gradients in the image (e.g., [35]). In a second filtering stage, this orientation information is then used to adaptively "steer" the local kernel, resulting in elongated, elliptical contours spread along the directions of the local edge structure. With these locally adapted kernels, the denoising is effected most strongly along the edges, rather than across them, resulting in strong preservation of details in the final output. To be more specific, the data-adapted kernel takes the form

$$K_{\text{adapt}}(\mathbf{x}_i - \mathbf{x}, y_i - y) \equiv K_{\mathbf{H}_i^{\text{steer}}}(\mathbf{x}_i - \mathbf{x}) \tag{46}$$

where $\mathbf{H}_i$s are now the data-dependent full matrices which we call *steering* matrices. We define them as

$$\mathbf{H}_i^{\text{steer}} = h\mu_i \mathbf{C}_i^{-\frac{1}{2}} \tag{47}$$

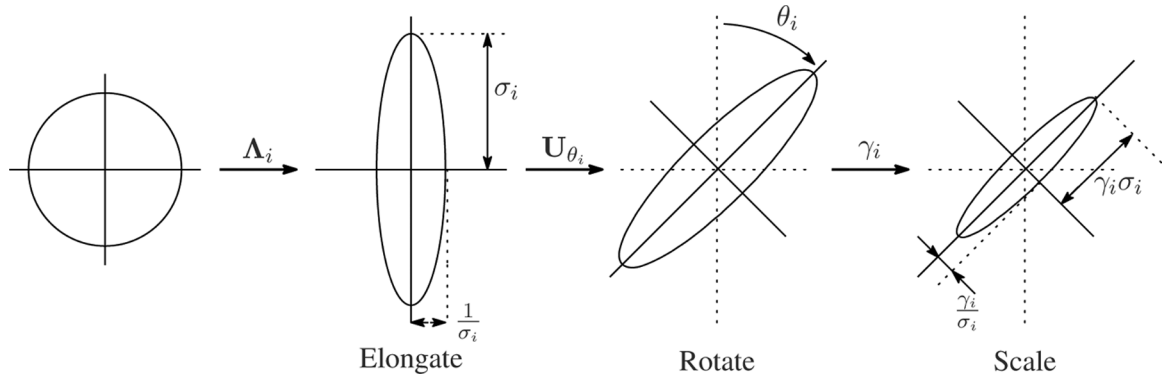


Fig. 9. Schematic representation illustrating the effects of the steering matrix and its component ($\mathbf{C}_i = \gamma_i \mathbf{U}_{\theta_i} \mathbf{\Lambda}_i \mathbf{U}_{\theta_i}^T$) on the size and shape of the regression kernel.

where $\mathbf{C}_i$s are (symmetric) covariance matrices based on differences in the local gray-values. A good choice for $\mathbf{C}_i$s will effectively spread the kernel function along the local edges, as shown in Fig. 8. It is worth noting that, even if we choose a large h in order to have a strong denoising effect, the undesirable blurring effect, which would otherwise have resulted, is tempered around edges with appropriate choice of $\mathbf{C}_i$s. With such steering matrices, for example, if we choose a Gaussian kernel, the steering kernel is mathematically represented as

$$K_{\mathbf{H}_i^{\text{steer}}}(\mathbf{x}_i - \mathbf{x}) = \frac{\sqrt{\det(\mathbf{C}_i)}}{2\pi h^2 \mu_i^2} \exp\left\{ -\frac{(\mathbf{x}_i - \mathbf{x})^T \mathbf{C}_i (\mathbf{x}_i - \mathbf{x})}{2h^2 \mu_i^2} \right\}. \tag{48}$$

The local edge structure is related to the gradient covariance (or equivalently, the locally dominant orientation), where a naive estimate of this covariance matrix may be obtained as follows:

$$\hat{\mathbf{C}}_i \approx \begin{bmatrix} \sum_{\mathbf{x}_j \in w_i} z_{x_1}(\mathbf{x}_j) z_{x_1}(\mathbf{x}_j) & \sum_{\mathbf{x}_j \in w_i} z_{x_1}(\mathbf{x}_j) z_{x_2}(\mathbf{x}_j) \\ \sum_{\mathbf{x}_j \in w_i} z_{x_1}(\mathbf{x}_j) z_{x_2}(\mathbf{x}_j) & \sum_{\mathbf{x}_j \in w_i} z_{x_2}(\mathbf{x}_j) z_{x_2}(\mathbf{x}_j) \end{bmatrix} \tag{49}$$

where $z_{x_1}(\cdot)$ and $z_{x_2}(\cdot)$ are the first derivatives along x_1 and x_2 directions and w_i is a local analysis window around the position of interest. The dominant local orientation of the gradients is then related to the eigenvectors of this estimated matrix. Since the gradients $z_{x_1}(\cdot)$ and $z_{x_2}(\cdot)$ depend on the pixel values ($\{y_i\}$), and since the choice of the localized kernels in turns depends on these gradients, it, therefore, follows that the "equivalent" kernels for the proposed data-adapted methods form a locally "nonlinear" combination of the data.

While this approach [which is essentially a local principal components method to analyze image (orientation) structure] [35]–[37] is simple and has nice tolerance to noise, the resulting estimate of the covariance may, in general, be rank deficient or unstable, and, therefore, care must be taken not to take the inverse of the estimate directly in this case. In such case, a diagonal loading or regularization methods can be used to obtain stable estimates of the covariance. In [35], we proposed an effective *multiscale* technique for estimating local orientations, which fits the requirements of this problem nicely. Informed by the above, in this paper, we take a parametric approach to the design of the steering matrix.

In order to have a more convenient form of the covariance matrix, we decompose it into three components (equivalent to eigenvalue decomposition) as follows:

$$\begin{aligned} \mathbf{C}_i &= \gamma_i \mathbf{U}_{\theta_i} \mathbf{\Lambda}_i \mathbf{U}_{\theta_i}^T, \\ \mathbf{U}_{\theta_i} &= \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{bmatrix} \\ \mathbf{\Lambda}_i &= \begin{bmatrix} \sigma_i & 0 \\ 0 & \sigma_i^{-1} \end{bmatrix} \end{aligned} \tag{50}$$

where $\mathbf{U}_{\theta_i}$ is a rotation matrix and $\mathbf{\Lambda}_i$ is the elongation matrix. Now, the covariance matrix is given by the three parameters γ_i, θ_i, and σ_i, which are the scaling, rotation, and elongation parameters, respectively. Fig. 9 schematically explains how these parameters affect the spreading of kernels. First, the circular kernel is elongated by the elongation matrix $\mathbf{\Lambda}_i$, and its semi-minor and major axes are given by σ_i. Second, the elongated kernel is rotated by the matrix $\mathbf{U}_{\theta_i}$. Finally, the kernel is scaled by the scaling parameter γ_i.

We define the scaling, elongation, and rotation parameters as follow. Following our previous work in [35], the dominant orientation of the local gradient field is the singular vector corresponding to the smallest (nonzero) singular value of the local gradient matrix arranged in the following form:

$$\mathbf{G}_i = \begin{bmatrix} \vdots & \vdots \\ z_{x_1}(\mathbf{x}_j) & z_{x_2}(\mathbf{x}_j) \\ \vdots & \vdots \end{bmatrix} = \mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^T, \quad \mathbf{x}_j \in w_i \tag{51}$$

where $\mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^T$ is the truncated singular value decomposition of $\mathbf{G}_i$, and $\mathbf{S}_i$ is a diagonal 2×2 matrix representing the energy in the dominant directions. Then, the second column of the 2×2 orthogonal matrix $\mathbf{V}_i$, $\mathbf{v}_2 = [\nu_1, \nu_2]^T$, defines the dominant orientation angle θ_i

$$\theta_i = \arctan\left(\frac{\nu_1}{\nu_2}\right). \tag{52}$$

That is, the singular vector corresponding to the smallest nonzero singular value of $\mathbf{G}_i$ represents the dominant orientation of the local gradient field. The elongation parameter σ_i

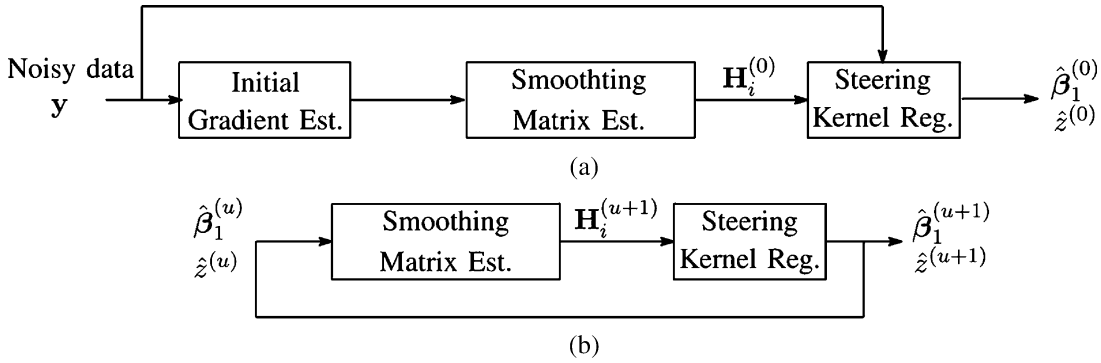


Fig. 10. Block diagram representation of the iterative steering kernel regression. (a) Initialization. (b) Iteration.

can be selected corresponding to the energy of the dominant gradient direction

$$\sigma_i = \frac{s_1 + \lambda'}{s_2 + \lambda'}, \quad \lambda' \geq 0 \tag{53}$$

where λ' is a "regularization" parameter for the kernel elongation, which dampens the effect of the noise, and restricts the ratio from becoming degenerate. The intuition behind (53) is to keep the shape of the kernel circular in flat areas $(s_1 \approx s_2 \approx 0)$, and elongate it near edge areas $(s_1 \gg s_2)$. Finally, the scaling parameter γ_i is defined by

$$\gamma_i = \left(\frac{s_1 s_2 + \lambda''}{M} \right)^{\frac{1}{2}} \tag{54}$$

where λ'' is again a "regularization" parameter, which dampens the effect of the noise and keeps γ_i from becoming zero,[5] and M is the number of samples in the local analysis window. The intuition behind (54) is that, to reduce noise effects while producing sharp images, large footprints are preferred in the flat (smooth) and smaller ones in the textured areas. Note that the local gradients and the eigenvalues of the local gradient matrix $\hat{\mathbf{C}}_i$ are smaller in the flat (low-frequency) areas than the textured (high-frequency) areas. As $\sqrt{s_1 s_2}$ is the geometric mean of the eigenvalues of $\hat{\mathbf{C}}_i$, γ_i makes the steering kernel area large in the flat, and small in the textured areas.

While it appears that the choice of parameters in the above discussion is purely ad-hoc, we direct the interested reader to a more careful statistical analysis of the distributional properties of the singular values (s_j) in [35], [38], and [39]. Our particular selections for these parameters are directly motivated by these earlier works. However, to maintain focus and conserve space, we have elected not to include such details in this presentation. We also note that the presented formulation is quite close and apparently independently derived normalized convolution formulation of [6].

Fig. 11 is a visual illustration of the steering kernel footprints on a variety of image structures (texture, flat, strong edge, corner, and weak edge) of the Lena image for both noiseless and noisy cases. Note that, in the noisy case, the shape and orientation of the kernel's footprints are very close to those of the noiseless case. Also, depending on the underlying features, in the flat areas, they are relatively more spread to reduce the noise effects, while in texture areas, their spread is very close to the noiseless case which reduces blurriness.

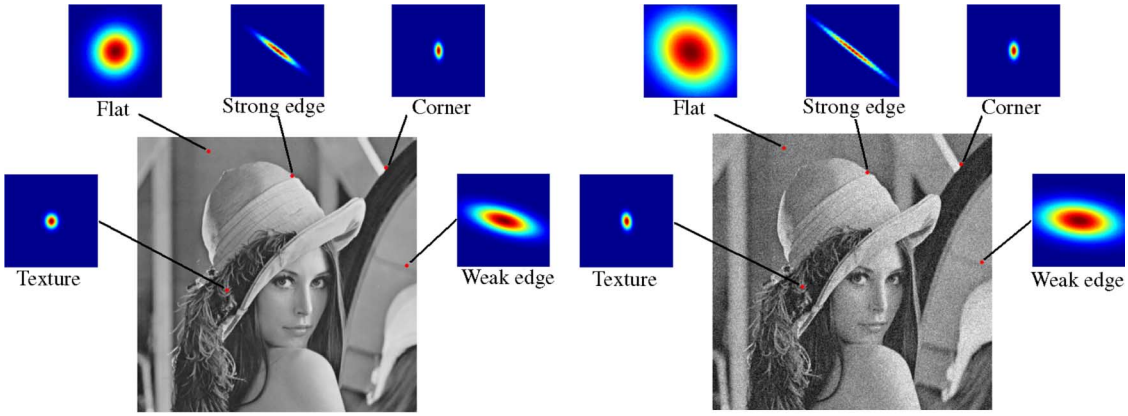
C. Iterative Steering Kernel Regression

The estimated smoothing matrices of the steering kernel regression method are data dependent, and, consequently, sensitive to the noise in the input image. As we experimentally demonstrate in the next section, steering kernel regression is most effective when an iterative regression/denoising procedure is used to exploit the output (less noisy) image of each iteration to estimate the radiometric terms of the kernel in the next iteration. A block diagram representation of this method is shown in Fig. 10, where u is the iteration number. In this diagram, the data samples are used to create the initial (dense) estimate[6] of the interpolated output image Fig. 10(a). In the next iteration, the reconstructed (less noisy) image is used to calculate a more reliable estimate of the gradient Fig. 10(b), and this process continues for a few more iterations. A quick consultation with Fig. 10(a) shows that, although our proposed algorithm relies on an initial estimation of the gradient, we directly apply the estimated kernels on the original (noninterpolated) samples which results in the populated (or denoised) image in the first iteration. Therefore, denoising and interpolation are done jointly in one step. Further iterations in Fig. 10(b) apply the modified kernels on the denoised pixels which results in more aggressive noise removal.

While we do not provide an analysis of the convergence properties of this proposed iterative procedure, we note that while increasing the number of iterations reduces the variance of the estimate, it also leads to increased bias (which manifests as blurriness). Therefore, in a few (typically around five) iterations, a minimum mean-squared estimate is obtained. An example of this observation is shown in Figs. 12–14. A future line of work will analyze the derivation of an effective stopping rule from first principles.

[5]The regularization parameters λ' and λ'' are used to prohibit the shape of the kernel from becoming infinitely narrow and long. In practice, it suffices to keep these numbers reasonably small, and, therefore, in all experiments in this paper, we fixed their values equal to $\lambda' = 1.0$ and $\lambda'' = 0.01$.

[6]Note that, in this paper, all adaptive kernel regression experiments are initialized with the outcome of the classic kernel regression method.



(a) Noiseless case (b) Noisy case (SNR = 5.64[dB])

Fig. 11. Footprint examples of steering kernels with covariance matrices $\{\mathbf{C}_i\}$ given by the local orientation estimate (50) at a variety of image structures. (a) The estimated kernel footprints in a noiseless image and (b) the estimated footprints for the same areas of a noisy image (after 7 iterations considering additive Gaussian noise with $\sigma = 25$ similar to the first experiment of Section IV).

It is worth pointing out that the iterative regression method has the luxury of using directly estimated gradients. Note that the discrete gradients used in (51) are usually approximated by convolving a band-pass filter with the image. However, the comparison between (15) and (17) shows that the vector $\boldsymbol{\beta}_1$ is the direct estimate of the image gradient. Indeed direct estimation of the gradient vector is more reliable but at the same time computationally more expensive. In Appendix I, computation of $\boldsymbol{\beta}_1$, directly in the regression context, is shown.

IV. Experiments

In this section, we provide experiments on simulated and real data. These experiments show diverse applications and the excellence of the proposed adaptive technique, and also attest to the claims made in the previous sections. Note that in all experiments in this paper we used Gaussian type kernel functions. While it is known [11] that, for classic kernel regression, the choice of kernel is not of critical importance, the analysis of this choice for the data-adapted regression methods remains to be done, and is outside the scope of this paper.

A. Denoising Experiment

In the first set of experiments, we compare the performance of several denoising techniques. We set up a controlled simulated experiment by adding white Gaussian noise with standard deviation of $\sigma = 25$ to the Lena image shown in Fig. 12(a). The resulting noisy image with signal-to-noise ratio (SNR)[7] of 5.64 [dB], is shown in Fig. 12(b). This noisy image is then denoised by the classic kernel regression[8] (34) with $N = 2$ and $h = 1.8$, result of which is shown in Fig. 12(c). The result of applying the bilateral filter (45) with $h_s = 1.5$ and $h_r = 7.4$ is shown in Fig. 12(d). For the sake of comparison, we have included the result of applying anisotropic diffusion[9] [26] and the recent wavelet-based denoising method[10] of [40] in Fig. 12(e) and (f), respectively. Finally, Fig. 12(g) shows the result of applying the iterative steering kernel regression of Fig. 10 with $N = 2$, $h = 2.5$, and 7 iterations. The RMSE values of the restored images of Fig. 12(c)–(g) are 8.94, 8.65, 8.46, 6.66 and 6.68, respectively. The RMSE results reported for the Fig. 12(f) and (g) are the results of 35 Monte Carlo simulations. We also noted that the wavelet method of [40], in general, is more computationally efficient than the steering kernel method, though it is applicable only to the removal of Gaussian noise.

We set up a second controlled simulated experiment by considering JPEG compression artifacts which result from compression of the image in Fig. 13(a). The JPEG image was constructed by using MATLAB JPEG compression routine with a quality parameter equal to 10. This compressed image with a RMSE value equal to 9.76 is illustrated in Fig. 13(b). We applied several denoising methods (similar to the ones used in the previous experiment). The results of applying classic kernel regression (34) ($N = 2$ and $h = 1.0$), bilateral filtering (45) ($h_s = 2.0$ and $h_r = 4.1$), Wavelet [40], and the iterative steering kernel regression ($N = 2$, $h = 2.0$, and 3 iterations) are given in Fig. 13(c)–(f), respectively. The RMSE values of the reconstructed images of (c)–(f) are 9.05, 8.52, 8.80, and 8.48, respectively.

In the third denoising experiment, we applied several denoising techniques on the color image shown in Fig. 14(a), which is corrupted by real film grain and scanning process noise. To produce better color estimates, following [41], first we transferred this RGB image to the YCrCb representation. Then we applied several denoising techniques (similar to the ones in the previous two experiments) on each channel (the luminance component Y, and the chrominance components Cr and Cb), separately. The results of applying Wavelet [40], and bilateral filtering (45) ($h_s = 2.0$ and $h_r = 3.5$ for all channels), and the iterative steering kernel regression ($N = 2$, $h = 2.0$, and 3 iterations) are given in Fig. 14(b)–(d), respectively. Fig. 14(e)–(g) shows the absolute values of the residuals on the Y channel. It can be seen that the proposed steering kernel

[7]Signal-to-noise ratio is defined as $10\log_{10}(\sigma^2/\sigma_n^2)$, where σ^2, σ_n^2 are variance of a clean image and noise, respectively.

[8]The criterion for parameter selection in this example (and other simulated examples discussed in this paper) was to choose parameters which give the best RMSE result. For the experiments on real data, we chose parameters which produced visually most appealing results.

[9]The software is available at http://www.cns.nyu.edu/david/aniso.html.

[10]In this experiment, we used the code (with parameters suggested for this experiment) provided by the authors of [40] available at http://decsai.ugr.es/~javier/denoise/index.html.

Fig. 12. Performance of different denoising methods are compared in this experiment. The RMSE of the images (c)–(g) are 8.94, 8.65, 8.46, 6.66, and 6.68, respectively. (a) Original image. (b) Noisy image, SNR $= 5.64$ [dB]. (c) Classic kernel regression. (d) Bilateral filter. (e) Anisotropic diffusion. (f) Wavelet [40]. (g) Iterative steering kernel regression. (h) Wavelet [40]. (i) Iterative steering kernel regression.

method produces the most noise-like residuals, which, in the absence of ground truth, is a reasonable indicator of superior performance.

B. Interpolation Experiments for Irregularly Sampled Data

The fourth experiment is a controlled simulated regression of an irregularly sampled image. We randomly deleted 85% of the pixels in the Lena image of Fig. 12(a), creating the sparse image of Fig. 15(a). To fill the missing values, first we implemented the Delaunay-spline smoother[11] with $\lambda = 0.087$ to fill the missing values, the result of which is shown in Fig. 15(b), with some clear artifacts on the edges. Fig. 15(c) shows the result of using the classic kernel regression (34) with $N = 2$ and $h = 2.25$. The result of the bilateral kernel regression with $N = 0, h_s = 2.25,$

[11]To implement the Delaunay-spline smoother we used MATLAB's "griddata" function with "cubic" parameter to transform the irregularly sampled data set to a dense regularly sampled one (Delaunay triangulation). The quality of the resulting image was further enhanced by applying MATLAB's spline smoother routine "csaps."

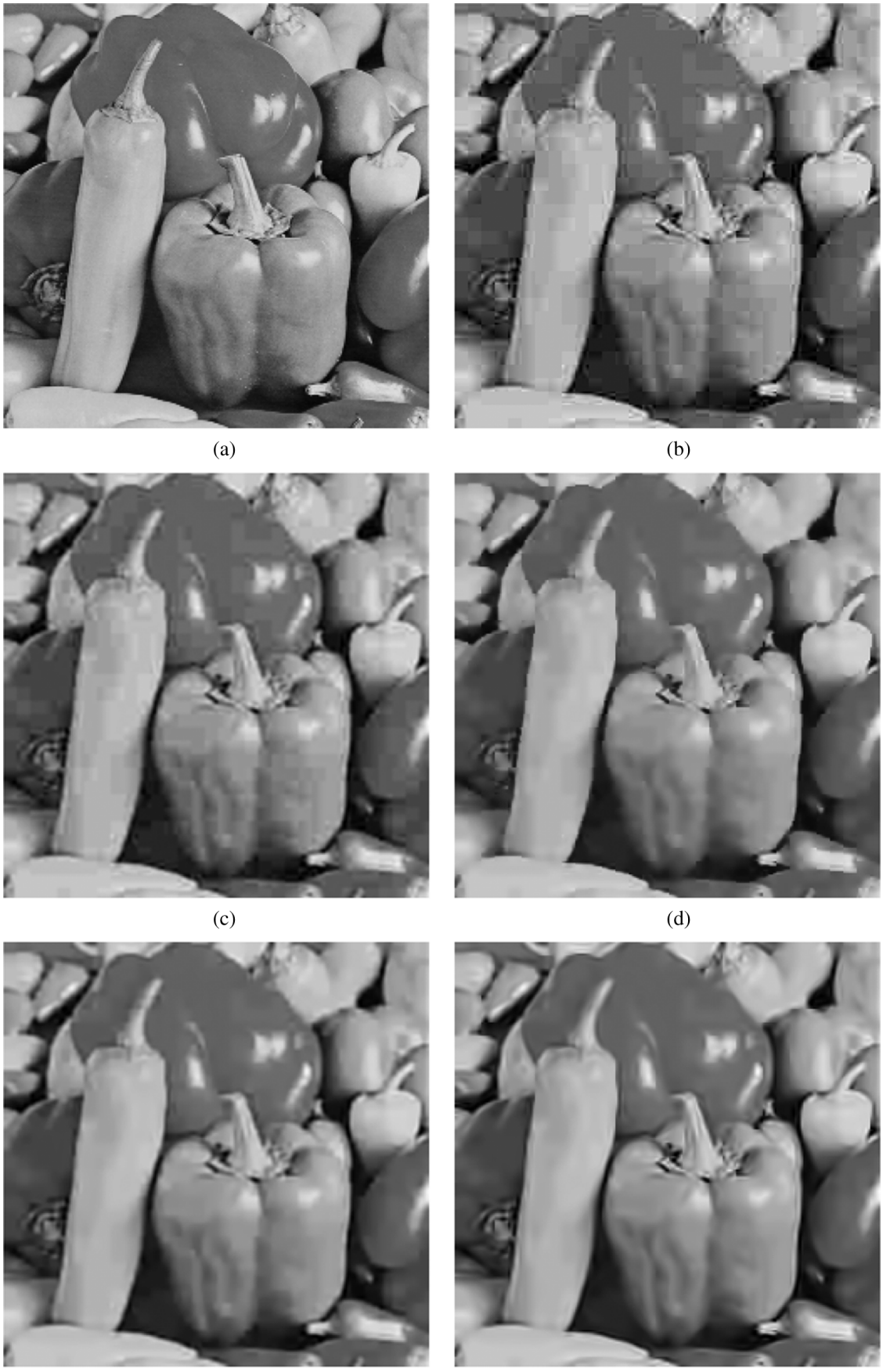


Fig. 13. Performance of different denoising methods are compared in this experiment on a compressed image by JPEG format with the quality of 10. The RMSE of the images (b)–(f) are 9.76, 9.03, 8.52, 8.80, and 8.48, respectively. (a) Original image. (b) Compressed image. (c) Classic kernel regression. (d) Bilateral filter. (e) Wavelet [40]. (f) Iterative steering kernel regression.

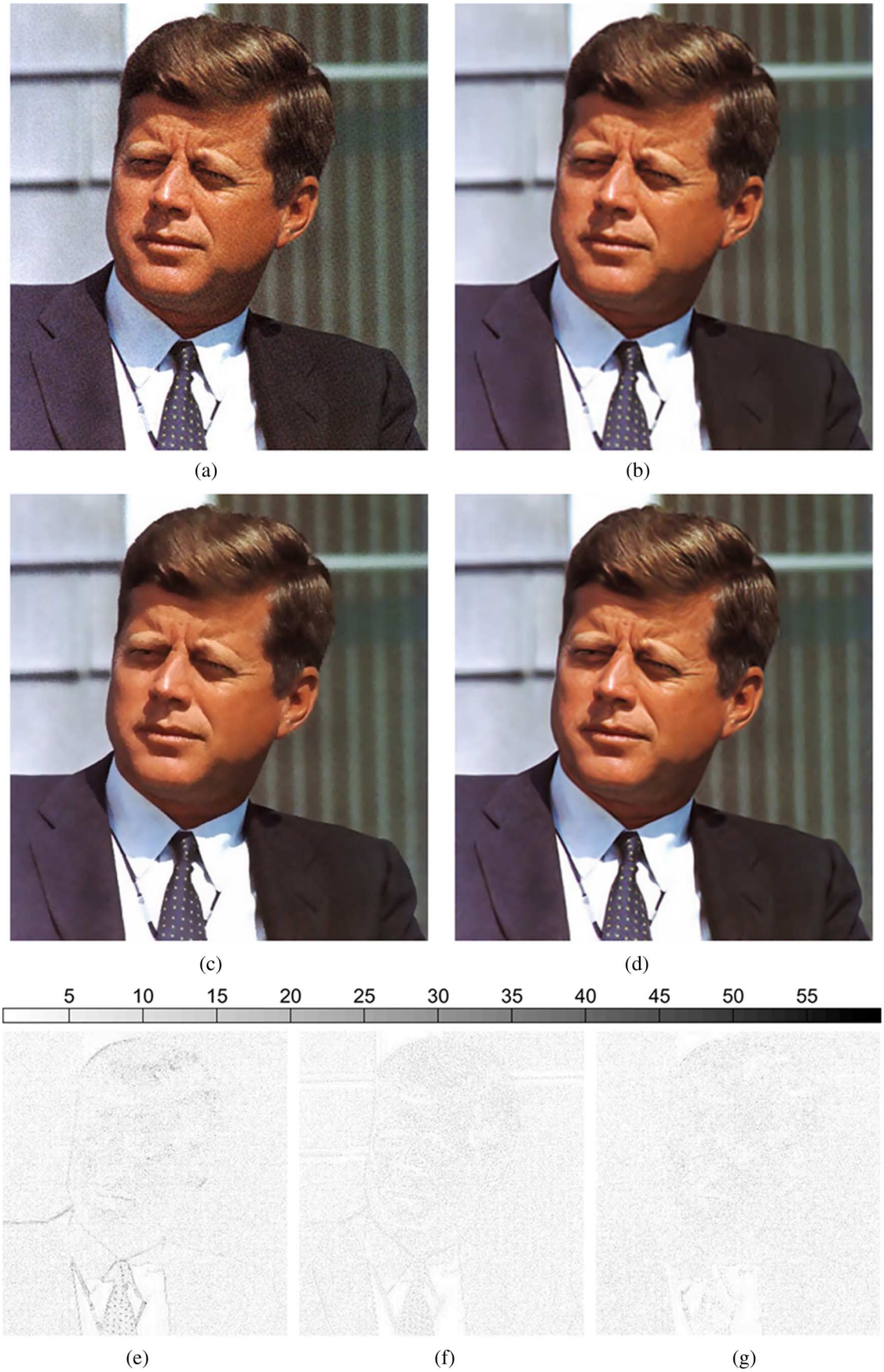


Fig. 14. Performance of different denoising methods are compared in this experiment on a color image with real noise; (e)–(g) show the residual between given noisy image and the respective estimates. (a) Real noisy image. (b) Wavelet [40]. (c) Bilateral filter [7]. (d) Iterative steering kernel regression. (e) Wavelet [40]. (f) Bilateral filter [7]. (g) Iterative steering kernel regression.

Fig. 15. Irregularly sampled data interpolation experiment, where 85% of the pixels in the Lena image are omitted in (a). The interpolated images using different methods are shown in (b)–(f). RMSE values for (b)–(f) are 9.15, 9.69, 9.72, 8.91, and 8.21, respectively. (a) Irregularly downsampled. (b) Delaunay-spline smoother. (c) Classic kernel regression, $N = 2$. (d) Bilateral kernel regression. (e) Steering kernel regression, $N = 0$. (f) Steering kernel regression, $N = 2$.

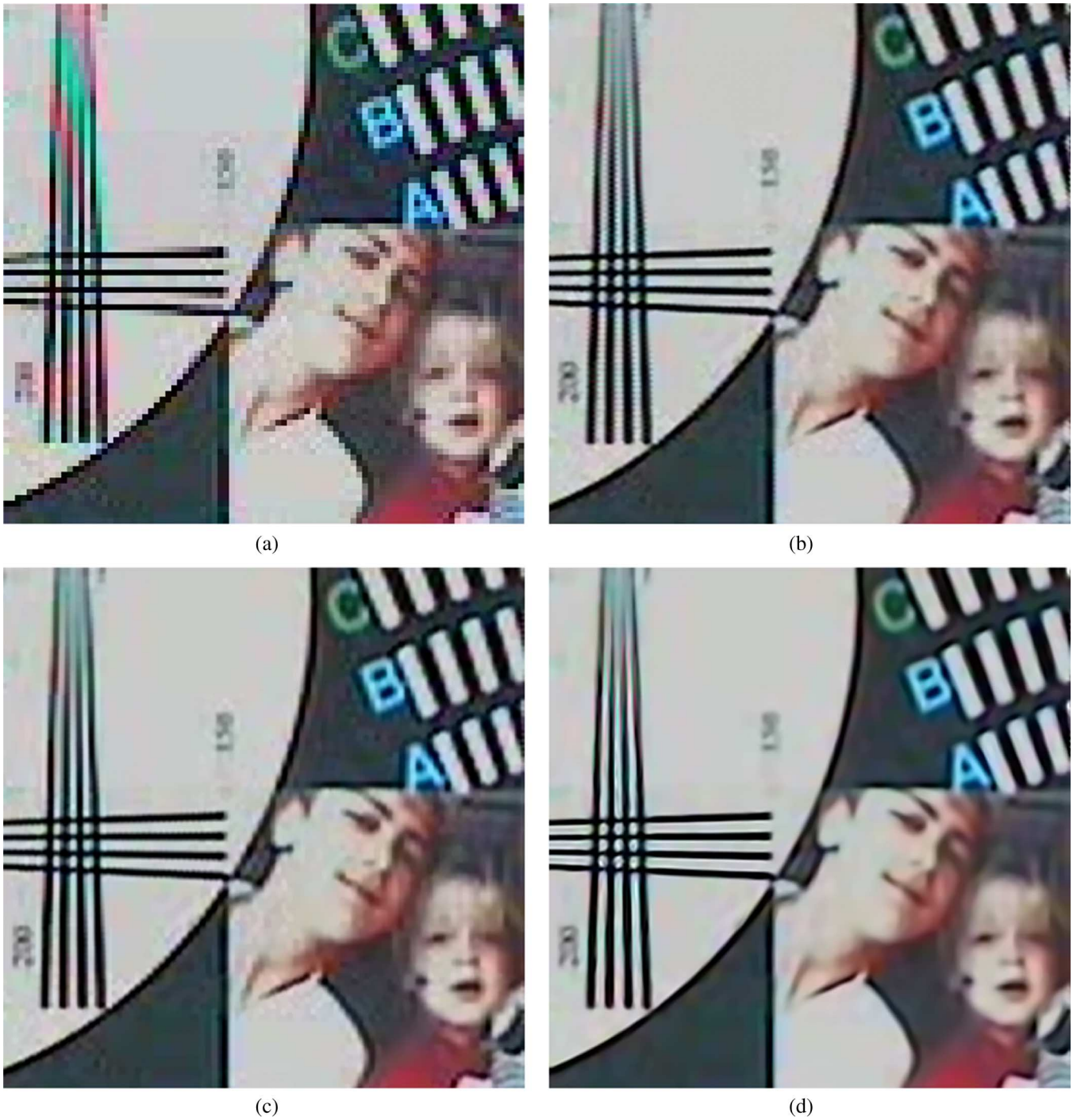


Fig. 16. Image fusion (super-resolution) experiment of a real data set consisting of ten compressed color frames. One input image is shown in (a); (b)–(d) show the multiframe shift-and-add images after interpolating by the Delaunay-spline smoother, classical kernel regression, and steering kernel regression methods, respectively. The resolution enhancement factor in this experiment was 5 in each direction. (a) The first input frame. (b) Multiframe Delaunay-spline smoother. (c) Multiframe classic kernel regression. (d) Multiframe steering kernel regression.

and $h_r = 3.0$ is shown in Fig. 15(d). Fig. 15(e) and (f) shows the results of implementing steering kernel regression ($N = 0$, $h = 0.8$, and no iterations), and ($N = 2$, $h = 1.6$, and 1 iteration), respectively. The RMSE values for images Fig. 15(b)–(f) are 9.15, 9.69, 9.72, 8.91, and 8.21, respectively.

Our final experiment is a multiframe super-resolution of a real compressed color image sequence captured with a commercial video surveillance camera; courtesy of Adyoron Intelligent Systems, Ltd., Tel Aviv, Israel. A total number of ten frames were used for this experiment, where the underlying motion was assumed to follow the translational model. One of these frames is shown in Fig. 16(a). To produce better color estimates, following [41], first we transferred the RGB frames to the YCrCb representation, and treated each channel separately. We used the method described in [42] to estimate the motion vectors. Then, we fused each channel of these frames on a high-resolution grid

with five times more pixels in each direction (i.e., 25 times as many overall pixels) as illustrated in Fig. 2, interpolated the missing values, and then deblurred the interpolated image using Bilateral total variation regularization [2].[12] The result of interpolating the irregularly sampled image by the Delaunay-spline smoother (implementation similar to the previous experiment with $\lambda = 0.28$ for the luminance and $\lambda = 0.43$ for the chrominance channels) followed by deblurring is shown in Fig. 16(b). The results of applying the classic kernel regression ($N = 2$ and $h = 2.0$ for the luminance channel and $h = 3.5$ for the chrominance channels) followed by deblurring and the steering kernel regression ($N = 2$, $h = 4.0$ for the luminance channel and $h = 8.0$ for the chrominance channels, and 1 iteration) followed by deblurring are shown in Fig. 16(c) and (d), respectively.

A comparison of these diverse experiments shows that, in general, the robust nonparametric framework we propose results in reliable reconstruction of images with comparable (if not always better) performance with respect to some of the most advanced methods designed specifically for particular applications, data and noise models. We note that while the proposed method might not be the best method for every application, imaging scenario, data or noise model, it works remarkably well for a wide set of disparate problems.

V. Conclusion and Future Works

In this paper, we studied a nonparametric class of regression methods. We promoted, extended, and demonstrated kernel regression, as a general framework, yielding several very effective denoising and interpolation algorithms. We compared and contrasted the classic kernel regression with several other competing methods. We showed that classic kernel regression is in essence a form of locally adaptive linear filtering process, the properties of which we studied under the equivalent kernel topic.

To overcome the inherent limitations dictated by the linear filtering properties of the classic kernel regression methods, we developed the nonlinear data adapted class of kernel regressors. We showed that the popular bilateral filtering technique is a special case of adaptive kernel regression and how the bilateral filter can be generalized in this context. Later, we introduced and justified a novel adaptive kernel regression method, called steering kernel with with comparable (if not always better) performance with respect to all other regression methods studied in this paper. Experiments on real and simulated data attested to our claim.

The outstanding performance of the adaptive methods can be explained by noting that the spline smoother [formulated in (12)] in effect exploits the Tikhonov (L_2) regularizers. However, the data-adapted kernel regression in its simplest form (bilateral filter) exploits the (PDE-based) total variation (TV) regularization [28], [43]. The relation between the bilateral filtering and TV regularization is established in [2]. The study in [2] also shows the superior performance of the TV-based regularization compared to the Tikhonov-based regularization.

Finally, in Section III-C, we proposed an iterative scheme to further improve the performance of the steering kernel regression method. An automatic method for picking the optimal number of iterations as well as the optimal regression order is a part is an open problem.

[12]For this experiment, the camera point spread function (PSF) was assumed to be a 5×5 disk kernel (obtained by the MATLAB command "fspecial('disk', 2)"). The deblurring regularization coefficient for the luminance channel was chosen to be 0.2 and for the chrominance channels was chosen to be 0.5.

Appendix I
Local Gradient Estimation

In this Appendix, we formulate the estimation of the direct gradient $\boldsymbol{\beta}_1$ of the second-order $(N = 2)$ kernel regressors. Note that direct gradient estimation is useful not only for the iterative steering kernel regression, but also for many diverse applications such as estimating motion via gradient-based methods (e.g., optical flow) and dominant orientation estimation.

$$\boldsymbol{\nabla}\hat{z}(\mathbf{x}) = \hat{\boldsymbol{\beta}}_1 = \begin{bmatrix} \mathbf{e}_2^T \\ \mathbf{e}_3^T \end{bmatrix} \left(\mathbf{X}_{\mathbf{x}}^T \mathbf{W}_{\mathbf{x}} \mathbf{X}_{\mathbf{x}}\right)^{-1} \mathbf{X}_{\mathbf{x}}^T \mathbf{W}_{\mathbf{x}} \mathbf{y}. \tag{55}$$

Following the notation of (27), the local quadratic derivative estimator is given by

$$\begin{aligned} \boldsymbol{\nabla}\hat{z}(\mathbf{x}) = \sum_{i=1}^{P} \mathbb{S}^{-1} \big[& -\mathbf{S}_{21}\mathsf{S}_{11}^{-1} + (\mathbf{x}_i - \mathbf{x}) \\ & -\mathbf{S}_{23}\mathbf{S}_{33}^{-1} \mathrm{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} \big] K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) y_i \end{aligned} \tag{56}$$

where

$$\begin{aligned} \mathsf{S}_{11} &= s_{11} - \mathbf{s}_{13}\mathbf{s}_{33}^{-1}\mathbf{s}_{31}, \quad \mathbf{S}_{21} = \mathbf{s}_{21} - \mathbf{s}_{23}\mathbf{s}_{33}^{-1}\mathbf{s}_{31} \\ \mathbf{S}_{23} &= \mathbf{s}_{23} - \mathbf{s}_{21}s_{11}^{-1}\mathbf{s}_{13}, \quad \mathbf{S}_{33} = \mathbf{s}_{33} - \mathbf{s}_{31}s_{11}^{-1}\mathbf{s}_{13} \\ \mathbb{S} &= \mathbf{s}_{22} - \mathbf{S}_{21}\mathsf{S}_{11}^{-1}\mathbf{s}_{12} - \mathbf{S}_{23}\mathbf{S}_{33}^{-1}\mathbf{s}_{32}. \end{aligned} \tag{57}$$

References

[1] T. F. Chan, "Nontexture inpainting by curvature-driven diffusions," *J. Vis. Commun. Image Represent.*, vol. 12, no. 10, pp. 436–449, May 2001.

[2] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multi-frame super-resolution," *IEEE Trans. Image Process.*, vol. 13, no. 10, pp. 1327–1344, Oct. 2004.

[3] M. P. Wand and M. C. Jones, *Kernel Smoothing*, ser. Monographs on Statistics and Applied Probability. New York: Chapman & Hall, 1995.

[4] P. Yee and S. Haykin, "Pattern classification as an ill-posed, inverse problem: a reglarization approach," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Apr. 1993, vol. 1, pp. 597–600.

[5] H. Knutsson and C.-F. Westin, "Normalized and differential convolution: methods for interpolation and filtering of incomplete and uncertain data," in *Proc. Computer Vision and Pattern Recognition*, Jun. 16–19, 1993, pp. 515–523.

[6] T. Q. Pham, L. J. van Vliet, and K. Schutte, "Robust fusion of irregularly sampled data using adaptive normalized convolution," *EURASIP J. Appl. Signal Process.*, article ID 83268, 2006.

[7] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. IEEE Int. Conf. Compute Vision*, Bombay, India, Jan. 1998, pp. 836–846.

[8] M. Elad, "On the origin of the bilateral filter and ways to improve it," *IEEE Trans. Image Process.*, vol. 11, no. 10, pp. 1141–1150, Oct. 2002.

[9] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Trans. Image Process.*, vol. 10, no. 10, pp. 1521–1527, Oct. 2001.

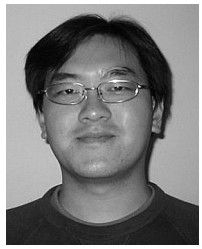
[10] N. K. Bose and N. A. Ahuja, "Superresolution and noise filtering using moving least squares," *IEEE Trans. Image Process.*, vol. 15, no. 8, pp. 2239–2248, Aug. 2006.

[11] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*, ser. Monographs on Statistics and Applied Probability. New York: Chapman & Hall, 1986.

[12] W. Hardle, *Applied Nonparametric Regression*. Cambridge, U.K.: Cambride Univ. Press, 1990.

[13] W. Hardle, M. Muller, S. Sperlich, and A. Werwatz, *Nonparametric and Semiparametric Models*, ser. Springer Series in Statistics. New York: Springer, 2004.

[14] W. Hardle, *Smooting Technique: With Implementation in S*, ser. Springer Series in Statistics. New York: Springer-Verlag, 1991.
[15] E. A. Nadaraya, "On estimating regression," *Theory Probabil. Appl.*, pp. 141–142, Sep. 1964.
[16] D. Ruppert and M. P. Wand, "Multivariate locally weighted least squares regression," *Ann. Statist.*, vol. 22, no. 3, pp. 1346–1370, Sept. 1994.
[17] M. G. Schimek, *Smooting and Regression-Approaches, Computation, and Application*, ser. Wiley Series in Probability and Statistics. New York: Wiley, 2000.
[18] M. Unser, "Splines: a perfect fit for signal and image processing," *IEEE Signal Process. Mag.*, vol. 16, no. 6, pp. 22–38, Nov. 1999.
[19] J. E. Gentle, W. Hadle, and Y. Mori, *Handbook of Computational Statistics: Concepts and Methods*. New York: Springer, 2004, pp. 539–564.
[20] C. de Boor, *A Practical Guide to Splines*. New York: Springer-Verlag, 1978.
[21] R. L. Eubank, *Nonparametric Regression and Spline Smoothing*, ser. Statistics, Textbooks and Monographs, 2nd ed. New York: Marcel-Dekker, 1999.
[22] L. Piegl and W. Tiller, *The NURBS Book*. New York: Springer, 1995.
[23] M. Arigovindan, M. Suhling, P. Hunziker, and M. Unser, "Variational image reconstruction from arbitrarily spaced samples: a fast multiresolution spline solution," *IEEE Trans. Image Process.*, vol. 14, no. 4, pp. 450–460, Apr. 2005.
[24] B. W. Silverman, "Spline smoothing: the equivalent variable kernel method," *Ann. Statist.*, vol. 12, no. 3, pp. 898–916, Sep. 1984.
[25] N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video*, ser. Pretice Hall signal processing. Englewood Cliffs, NJ: Prentice-Hall, 1984.
[26] M. J. Black, G. Sapiro, D. H. Marimont, and D. Heeger, "Robust anisotropic diffusion," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 421–432, Mar. 1998.
[27] Y. You and M. Kaveh, "Fourth-order partial differential equations for noise removal," *IEEE Trans. Image Process.*, vol. 9, no. 10, pp. 1723–1730, Oct. 2000.
[28] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D*, vol. 60, pp. 259–268, Nov. 1992.
[29] S. M. Kay, *Fundamentals of Statistical Signal Processing-Estimation Theory*, ser. Signal Processing. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[30] J. Fan and I. Gijbels, *Local Polynomial Modelling and Its Applications*, ser. Monographs on Statistics and Applied Probability. New York: Chapman & Hall, 1996.
[31] C. K. Chu and J. S. Marron, "Choosing a kernel regression estimator (with discussion)," *Statist. Sci.*, vol. 6, pp. 404–436, 1991.
[32] ——, "Comparison of two bandwidth selectors with dependent errors," *Ann. Statist.*, vol. 4, pp. 1906–1918, 1991.
[33] W. Hardle and P. Vieu, "Kernel regression smoothing of time series," *J. Time Ser. Anal.*, vol. 13, pp. 209–232, 1992.
[34] I. S. Abramson, "On bandwidth variation in kernel estimates—a square root law," *Ann. Statist.*, vol. 10, no. 4, pp. 1217–1223, Dec. 1982.
[35] X. Feng and P. Milanfar, "Multiscale principal components analysis for image local orientation estimation," presented at the 36th Asilomar Conf. Signals, Systems and Computers, Pacific Grove, CA, Nov. 2002.
[36] R. Mester and M. Hotter, "Robust displacement vector estimation including a statistical error analysis," in *Proc. 5th Int. Conf. Image Processing and its Applications*, 1995, pp. 168–172, R. B. GmbH.
[37] R. Mester and M. Muhlich, "Improving motion and orientation estimation using an equilibrated total least squares approach," in *Proc. IEEE Int. Conf. Image Processing*, 2001, pp. 929–932.
[38] J. M. B. K. V. Mardia and J. T. Kent, *Multivariate Analysis*. New York: Academic, 1979.
[39] A. Edelman, "Eigenvalues and condition numbers of random matrices," *SIAM J. Matrix Analysis and Aplications*, vol. 9, pp. 543–560, 1988.
[40] J. Portilla, V. Strela, M. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of Gaussians in the wavelet domain," *IEEE Trans. Image Process.*, vol. 12, no. 11, pp. 1338–1351, Nov. 2003.
[41] S. Farsiu, M. Elad, and P. Milanfar, "Multiframe demosaicing and super-resolution of color images," *IEEE Trans. Image Proces.*, vol. 15, no. 1, pp. 141–159, Jan. 2006.
[42] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Proc. Eur. Conf. Computer Vision*, May 1992, pp. 237–252.
[43] T. F. Chan, S. Osher, and J. Shen, "The digital TV filter and nonlinear denoising," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 231–241, Feb. 2001.
[44] H. Takeda, S. Farsiu, and P. Milanfar, "Higher order bilateral filters and their properties," presented at the SPIE Computational Imaging Conf., San Jose, CA, Jan. 2006, vol. 6498.



Hiroyuki Takeda (S'05) received the B.S. degree in electronics from Kinki University, Japan, and the M.S. degree in electrical engineering from the University of California, Santa Cruz (UCSC), in 2001 and 2006, respectively, where he is currently pursuing the Ph.D. degree in electrical engineering.

His research interests are in image processing (motion estimation, interpolation, and super-resolution) and inverse problems.

Sina Farsiu (M'05) received the B.Sc. degree in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 1999, the M.Sc. degree in biomedical engineering from the University of Tehran, Tehran, in 2001, and the Ph.D. degree in electrical engineering from the University of California, Santa Cruz (UCSC), in 2005

He is currently a Postdoctoral Scholar at UCSC. His technical interests include signal and image processing, optical imaging through turbid media, adaptive optics, and artificial intelligence.

Peyman Milanfar (SM'98) received the B.S. degree in electrical engineering and mathematics from the University of California, Berkeley, and the M.S., E.E., and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1988, 1990, 1992, and 1993, respectively.

Until 1999, he was a Senior Research Engineer at SRI International, Menlo Park, CA. He is currently an Associate Professor of electrical engineering at the University of California, Santa Cruz. He was a Consulting Assistant Professor of computer science at Stanford University, Stanford, CA, from 1998 to 2000, where he was also a Visiting Associate Professor in 2002. His technical interests are in statistical signal and image processing and inverse problems.

Dr. Milanfar won a National Science Foundation CAREER award in 2000 and he was Associate Editor for the IEEE SIGNAL PROCESSING LETTERS from 1998 to 2001.