Super-Drizzle: Applications of Adaptive Kernel Regression in Astronomical Imaging

Hiroyuki Takeda
Ph.D. Student, Electrical Engineering Department, University of California, Santa Cruz,
htakeda@ee.ucsc.edu

Sina Farsiu
Postdoctoral Scholar, Electrical Engineering Department, University of California, Santa Cruz,
farsiu@ee.ucsc.edu

Julian Christou*
Senior Specialist, University of California, Santa Cruz,
christou@ucolick.org

Peyman Milanfar
Professor, Electrical Engineering Department, University of California, Santa Cruz,
milanfar@ee.ucsc.edu

ABSTRACT

The drizzle algorithm is a widely used tool for image enhancement in the astronomical literature. For example, a very popular implementation of this method, as studied by Frutcher and Hook [1], has been used to fuse, denoise, and increase the spatial resolution of the images captured by the Hubble Space Telescope (HST). However, the drizzle algorithm is an ad-hoc method, equivalent to a spatially adaptive "linear" filter, which limits its range of performance.

To improve the performance of the drizzle algorithm, we make contact with the field of non-parametric statistics and generalize the tools and results for use in image processing and reconstruction. In contrast to the parametric methods, which rely on a specific model of the signal of interest, non-parametric 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. We promote the use and improve upon a class of non-parametric methods called *kernel regression* [2, 3].

1. INTRODUCTION

Recovering high quality images is a fundamental issue in image processing. There are many methods for image denoising and interpolation of given regularly sampled data sets, as illustrated in Fig.1(a) and Fig.1(b), respectively. In a more general case, a given data set is not regularly sampled, as illustrated in Fig.1(c). Fig.2 schematically represents a typical example where we have such a irregularly sampled data set. In a process often called *image fusion* we combine several under-sampled and noisy images (frames) in order to reconstruct a high quality image. In this section, we review a well-known image reconstruction method for irregularly sampled data sets, called the *shift-and-add* algorithm, and introduce two alternative methods in general form, which are categorized as the *spatially adaptive* filter and the *data-adaptive* filter.

1.1. Shift-and-Add Algorithm

The shift-and-add algorithm, developed through the years in different guises such as [4, 5, 6, 7, 8, 9, 10, 11], is a simple method to reconstruct an image from a data set composed of multiple images. Fig.3 schematically explains the algorithm. Suppose we have several images and know the relative motions between every pair of frames. We proceed by selecting a frame, upsampling it, and shifting all the samples (pixels) of the frame to their appropriate positions on a finer resolution grid. If several samples are located at (or sufficiently near) the same pixel lattice position, taking either the average value or the median value is an appropriate way to combine them and reduce noise.

The shift-and-add algorithm, however, has two serious drawbacks. One is that the algorithm places every sample on its nearest pixel lattice. That is to say, the pixel values from the low-resolution images are interpolated or directly snapped to the nearest high resolution grid points. This generates some positional (or motion) errors, which in effect add another kind of noise to the reconstructed images. Such errors cause the emergence of undesirable and false

*Currently a Program Manager with the Division of Astronomical Sciences at the National Science Foundation.

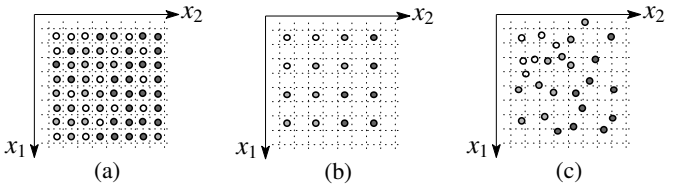


Fig. 1. (a) Denoising. (b) Interpolation of regularly sampled data. (c) Reconstruction from irregularly sampled data.

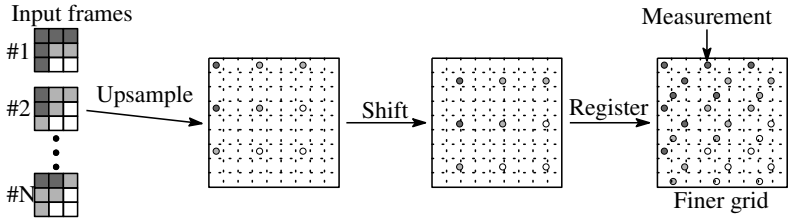


Fig. 2. Irregularly sampled data given by image fusion.

jaggies (or zipper) artifacts around edges of objects in the images. Although we can reduce the positional errors by choosing large resolution enhancement factor (i.e. a finer high resolution grid), this choice comes with the distinct disadvantage that we have an insufficient number of (non-redundant) low-resolution frames, we may be forced to leave many high-resolution pixels unfilled. Meanwhile, if the motions are very small, then all samples are registered at same locations, and consequently, a reconstructed image will have missing pixels. Fig.3 is an illustration of the missing pixels problem presented by the shift-and-add approach. A fast and memory efficient way to ameliorate the drawbacks this method is proposed in [11].

There are two other approaches to overcoming the same drawback, called the *drizzle* algorithm and the *Nadaraya-Watson Estimator* (NWE) which were proposed in [1] and [12], respectively. These two methods can be categorized as spatially adaptive filters which make use of linear combinations of pixels in a local neighborhood to denoise or reconstruct a pixel at a desired position. They can be referred to as *spatially adaptive* because the coefficients of these localized filters are dependent on the relative positions of the nearby samples being used, with higher weight given to pixels closer to the position of interest. These methods, however, do not take the actual pixel values of these nearby pixel values into account, and therein lies their weakness. In the following section, we briefly review drizzle and NWE, and more generally the class of spatially adaptive filters, and claim that they can be generalized to a much more effective class of filters which takes both the relative spatial *and* "radiometric" distances of nearby pixels into account. In this spirit, we call these more general filters "data-adaptive", and describe them in general form.

1.2. Introduction to Spatially Adaptive and Data-Adaptive Filters

Suppose we have two samples, marked at points 1 and 2 as illustrated in Fig.4(a), and wish to compute an appropriate pixel value at an arbitrary (pixel) position $\mathbf{x} = [x_1, x_2]^T$, which is marked in gray. The drizzle algorithm draws a square window (an effective area) around each sample, and computes a weighted average of all the available samples. The weights are essentially given by the size of the overlapped regions between the effective areas and a pixel lattice of interest. The size of effective areas is controllable by a parameter called *pixfrac* [1]. When the size of effective areas is set sufficiently small that no effective areas overlap on multiple pixel lattices, the drizzle algorithm becomes equivalent to the shift-and-add algorithm described earlier. Therefore, we can regard the shift-and-add algorithm as a special case of the drizzle algorithm. As can be expected, and is shown in Fig.4(a), it takes some computational load to calculate weights at a pixel lattice where an effective area is partially overlapping, particularly if the images have undergone somewhat complex geometric distortions.

On the other hand, unlike the drizzle algorithm, the effective areas of each sample in the NWE spread on the entire region of a pixel grid, and weights are given by a function of spatial distances between each sample's position and the center point of the pixel lattice of interest. To say this differently, a range of influence is defined for each given pixel by imposing a weight (*kernel*) function centered at that pixel's location. This weight is typically a decaying function away from pixel position, and the rate of decay can be controlled (even estimated from the local neighborhood). As such, the implementation of NWE is easier, and somewhat more general. So as we can see, both the drizzle algorithm and NWE create an output pixel value using its nearby samples and as illustrated in Fig.4, unlike the shift-and-add algorithm, they deal with all the samples at the exact positions where the samples are measured.

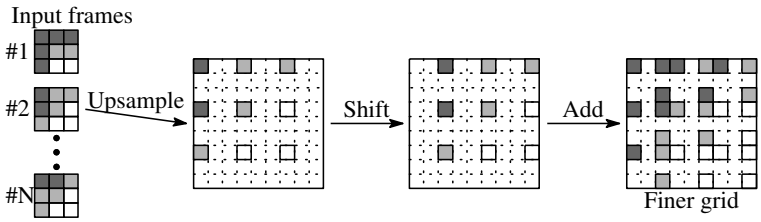


Fig. 3. Schematic representation of the shift-and-add algorithm.

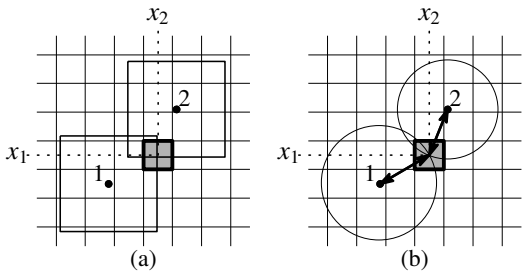


Fig. 4. The schematic representations of a pixel value computation by (a) drizzle algorithm and (b) Nadaraya-Watson estimator.

As explained above, the drizzle algorithm and NWE have some similarities, which is worth formalizing mathematically. By defining K as the kernel function, $z(\mathbf{x})$ as the true pixel value we wish to reconstruct at $\mathbf{x}$, and y_i as the i-th measured sample at $\mathbf{x}_i$, we can can express NWE as,

$$\hat{z}_{\text{NWE}}(\mathbf{x}) = \frac{\sum_i K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x})\, y_i}{\sum_i K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x})}, \quad \mathbf{x} = [x_1, x_2]^T. \tag{1}$$

Where $\mathbf{H}$ controls the spread of the kernel function, as we will explain later in the paper. In contrast, the weights in the case of drizzle are not simply a function of the distances $(\mathbf{x}_i - \mathbf{x})$. However, since the weights in both cases depend only on the position of a pixel of interest $\mathbf{x}$ and the samples' positions $\mathbf{x}_i$, we can write both by a general formulation using the general kernel function K as:

$$\hat{z}(\mathbf{x}) = \frac{\sum_i \mathsf{K}(\mathbf{x}_i, \mathbf{x})\, y_i}{\sum_i \mathsf{K}(\mathbf{x}_i, \mathbf{x})}. \tag{2}$$

Therefore, we categorize both the drizzle algorithm and NWE as spatially adaptive filters. The intellectually interesting next step to take here in order to improve performance is to explicitly take radiometric information into account as well, and formulate a more general class of filters which we call "data-adaptive" filters. To be more mathematically precise, we can write these as

$$\hat{z}(\mathbf{x}) = \frac{\sum_i \mathsf{K}(\mathbf{x}_i, \mathbf{x}, y_i, y)\, y_i}{\sum_i \mathsf{K}(\mathbf{x}_i, \mathbf{x}, y_i, y)}. \tag{3}$$

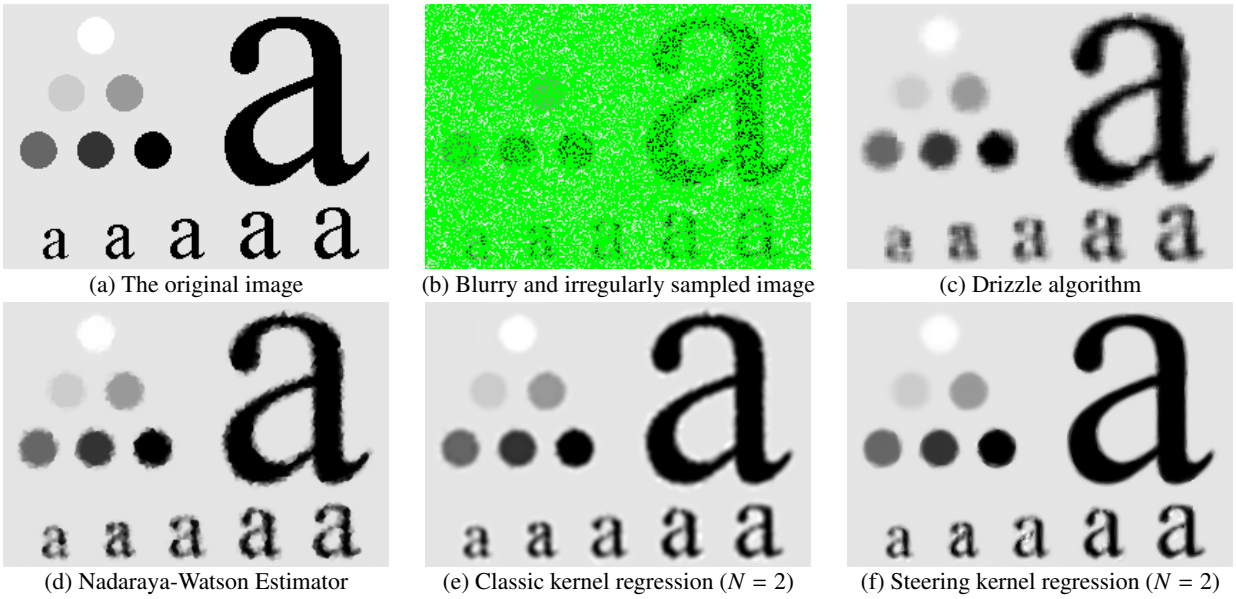
The formulations of (2) and (3) can be written even more generally still. Namely, the spatially adaptive filter and the data-adaptive filter can take the form of the weighted linear combination of all the samples as follows:

$$\text{Spatially adaptive filter} \quad : \quad \hat{z}(\mathbf{x}) = \textstyle\sum_i W_i(\mathbf{x}_i, \mathbf{x})\, y_i, \tag{4}$$

$$\text{Data-adaptive filter} \quad : \quad \hat{z}(\mathbf{x}) = \textstyle\sum_i W_i(\mathbf{x}_i, \mathbf{x}, y_i, y)\, y_i, \tag{5}$$

We shall discuss this general form in section 2.1, but for now, the weight W_i can be thought of as an *equivalent kernel*[2, 3], in the sense that it is a weight function derived from the original choice of kernels. A number of image processing methods such as the SUSAN filter[13], Bilateral filter[14], Mean-Shift[15], Non-Local Mean[16], etc., which have recently shown great promise, have been motivated by various ways of choosing and improving the data-adaptive weights $W_i(\mathbf{x}_i, \mathbf{x}, y_i, y)$. Most, if not all, of the aforementioned methods are suitable for one application; namely denoising. What we will illustrate here is that the equivalent kernel framework can be made available for not only denoising, but for interpolation as well. With this in mind, we focus on the kernel regression framework [3] in the rest of this paper, which encompasses the general formulation discussed above.

But first, we present a simulation which illustrates an example of the data-adaptive filter (5) comparing to the spatially adaptive filter (4) in Fig.5. In the simulation, we blurred the original image (Fig.5(a)) with a 5×5 Gaussian



(a) The original image (b) Blurry and irregularly sampled image (c) Drizzle algorithm

(d) Nadaraya-Watson Estimator (e) Classic kernel regression ($N = 2$) (f) Steering kernel regression ($N = 2$)

Fig. 5. An image reconstruction simulation on irregularly sampled data: (a)The original image, (b)Blurry and irregularly sampled image (the green pixels are representing missing pixels.), (c)-(e)The reconstructed images by spatially adaptive methods, and (f)The reconstructed image by a data adaptive method. The corresponding PSNR's with respect to the blurred image for the reconstructed images are (c)21.46, (d)24.83, (e)28.48, and (f)30.15.

PSF with the standard deviation 1.0, and irregularly downsampled the blurred image (we randomly discarded 75% of the original pixels), which is in Fig.5(b) (the green pixels are representing the missing pixels). This situation can occurr when the quality of a data transmission channel is so poor that the channel drops many data points, or a data set is so noisy that many pixels are eliminated by a pre-processing mechanism. The estimated images by spatially adaptive methods: the drizzle algorithm ($pixfrac = 0.14$), NWE ($h = 1.0$), the second order classic kernel regression ($h = 1.4$), are shown in Fig.5(c) through Fig.5(e), respectively. The reconstructed image by a data-adaptive method (the second order iterative steering kernel regression with $h = 4.5$ and one iteration) which we will discuss in Section 2.2.2, is shown in Fig.5(f). The corresponding PSNR's[2] are indicated in the caption of the figure. Note that we chose the parameters for each method in the above simulation to produce the best PSNR. Furthermore, and a final deblurring of the interpolated images was necessary to recover the high frequency components. To summarize, in this paper, we will present the theory behind the spatially adaptive filter (4) including both the drizzle algorithm and NWE by using the kernel regression framework to understand more clearly about the spatially adaptive filter, and expand our discussion to a data-adaptive filter (5) in order to improve their performances.

In this paper, we introduce new aspects of the kernel regression framework to the astronomy science community. The novelties of this paper include

(i) We exploit the kernel regression framework to justify a powerful variation of the drizzle algorithm with superior performance, applicable to both regularly and irregularly sampled data.

(ii) Unlike the drizzle algorithm, the effective size and shape of the kernel (effective areas) in the proposed method are adaptive locally not only to the spatial sampling density of the data, but also to the actual (measured brightness) values of those samples.

(iii) The proposed method is applicable to both single frame and multi-frame processing scenarios, and is equally effective for both oversampled and undersampled images.

(iv) The proposed method takes advantage of a generic model that is appropriate for reconstructing images contaminated with different noise models, including additive Gaussian, Laplacian, and Salt & Pepper, and even for removal of compression artifacts.

[2]Peak Signal to Noise Ratio $= 20 \log_{10} \left(\frac{\text{Peak Signal Value}}{\text{Root Mean Square Error}} \right)$ [dB]

Experiments on simulated and real data show diverse applications and the superiority of the proposed adaptive technique with respect to the state of the art methods in the literature. We show that the proposed algorithm not only visually and numerically (PSNR comparison) improves the quality of reconstruction, but also due to its non-parametric structure, will result in images that are faithful with respect to the photometric properties of the original (ideal) image.

2. IMAGE PROCESSING AND RECONSTRUCTION

In this section, we present the derivation of the spatially adaptive filter by using the kernel regression framework, and we will see that the filter class introduced in (2) which includes the drizzle algorithm and NWE, is neatly explained as a special case of the general formulation (4). Following this, our discussion will progress to the data-adaptive filter.

2.1. Spatially Adaptive Filters

Classical parametric denoising 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. A generative model based upon the estimated parameters is then produced as the best estimate of the underlying signal. In contrast, non-parametric methods rely on the data itself to dictate the structure of the model, in which case the implicit model is referred to as *a regression function* [2]. In particular, consider the estimation problem in two dimensions where the each measured sample in Fig.1(c) is given by

$$y_i = z(\mathbf{x}_i) + \varepsilon_i, \quad i = 1, 2, \cdots, P, \quad \mathbf{x}_i = \left[x_{1i},\ x_{2i}\right]^T, \tag{6}$$

where y_i's are measurements, $z(\cdot)$ is the (hitherto unspecified) regression function (i.e. an unknown image) to be estimated, and ε_i's are independent and identically distributed zero mean noise values (with otherwise no particular statistical distribution assumed).

As the specific form of $z(\cdot)$ is unspecified, in order to estimate the value of the function at any point $\mathbf{x}$ given by the data, one can trust in a generic, local expansion of the function about this point. Specifically, if $\mathbf{x}$ is near the samples at $\mathbf{x}_i$, we have the N-term Taylor series[3]

$$z(\mathbf{x}_i) \approx z(\mathbf{x}) + \{\nabla z(\mathbf{x})\}^T (\mathbf{x}_i - \mathbf{x}) + \frac{1}{2!} (\mathbf{x}_i - \mathbf{x})^T \{\mathcal{H} z(\mathbf{x})\} (\mathbf{x}_i - \mathbf{x}) + \cdots \tag{7}$$

$$= \beta_0 + \boldsymbol{\beta}_1^T (\mathbf{x}_i - \mathbf{x}) + \boldsymbol{\beta}_2^T \mathrm{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} + \cdots, \tag{8}$$

where ∇ and $\mathcal{H}$ are the gradient (2×1) and Hessian (2×2) matrices respectively and vech$(\cdot)$ is the *half-vectorization operator* [17], which lexicographically orders the lower-triangular portion of a matrix into a column vector.

The above suggests that if we 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. Indeed, the coefficients $\{\boldsymbol{\beta}_n\}_{n=1}^N$ will provide localized information on the *derivatives* of the regression function. Naturally, since the approach is based on local approximations, a reasonable step might be to estimate the coefficients $\{\boldsymbol{\beta}_n\}_{n=0}^N$ from the data, giving the nearby samples higher weight than samples farther away. A general fitting formulation capturing this idea solves the following optimization problem:

$$\min_{\{\boldsymbol{\beta}_n\}_{n=0}^N} \sum_{i=1}^{P} \left| y_i - \beta_0 - \boldsymbol{\beta}_1^T (\mathbf{x}_i - \mathbf{x}) - \boldsymbol{\beta}_2^T \mathrm{vech}\left\{(\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^T\right\} - \cdots \right|^m \mathsf{K}(\mathbf{x}_i, \mathbf{x}), \tag{9}$$

where m is a parameter, which when set to $m = 2$ would yield a local weighted least squares formulation; but can also be set to $m = 1$ for instance, for a more robust framework. In the kernel regression framework, $\mathsf{K}(\mathbf{x}_i, \mathbf{x})$ is defined as

$$\mathsf{K}(\mathbf{x}_i, \mathbf{x}) \Longrightarrow K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) = \frac{1}{\det(\mathbf{H})} K\left(\mathbf{H}^{-1}(\mathbf{x}_i - \mathbf{x})\right). \tag{10}$$

The kernel function K penalizes distance away from the local position where the approximation is centered, and where $\mathbf{H}$ is a 2×2 "smoothing" matrix which controls the width of this weight function in terms of assigning relative weights to pixels near and far. The standard choice of the matrix is

$$\mathbf{H}_i = h\mu_i \mathbf{I}_2, \tag{11}$$

[3]Other expansions are also possible, e.g. orthogonal series.

where h is the so-called *global smoothing parameter*, and μ_i is a scalar called *local density parameter* that captures the local density of data, nominally set to $\mu_i = 1$, though adaptive ways of computing μ_i are proposed in [18]. In general, the function $K(\cdot)$ is a symmetric function, which attains its maximum at zero, and which decays away from zero at a rate controlled by the smoothing matrix. More specifically, the standard definition of the kernel function for two dimensional data has

$$\int_{R^2} \mathbf{t}K(\mathbf{t})d\mathbf{t} = 0, \quad \int_{R^2} \mathbf{t}\mathbf{t}^T K(\mathbf{t})d\mathbf{t} = c\mathbf{I}_2. \tag{12}$$

The choice of the particular form of the function $K(\cdot)$ is open, and may be selected as a Gaussian, exponential, or other valid form that complies with the above constraints.

As we alluded to earlier, reasonable choices of the penalizing parameter (m) are 2 or 1. When we choose $m = 2$, the optimization problem (9) can be posed as weighted least-squares, which takes the following matrix form:

$$\hat{\mathbf{b}} = \arg\min_{\mathbf{b}} \left\| \mathbf{y} - \mathbf{X}_{\mathbf{x}}\mathbf{b} \right\|^2_{\mathbf{W}_{\mathbf{x}}}, \tag{13}$$

where

$$\mathbf{y} = [y_1, y_2, \cdots, y_P]^T, \quad \mathbf{b} = \left[\beta_0, \boldsymbol{\beta}_1^T, \cdots, \boldsymbol{\beta}_N^T\right]^T, \quad \mathbf{W}_{\mathbf{x}} = \text{diag}\left[\mathsf{K}(\mathbf{x}_1, \mathbf{x}), \mathsf{K}(\mathbf{x}_2, \mathbf{x}), \cdots, \mathsf{K}(\mathbf{x}_P, \mathbf{x})\right], \tag{14}$$

$$\mathbf{X}_{\mathbf{x}} = \begin{bmatrix} 1 & (\mathbf{x}_1 - \mathbf{x})^T & \text{vech}^T\left\{(\mathbf{x}_1 - \mathbf{x})(\mathbf{x}_1 - \mathbf{x})^T\right\} & \cdots \\ 1 & (\mathbf{x}_2 - \mathbf{x})^T & \text{vech}^T\left\{(\mathbf{x}_2 - \mathbf{x})(\mathbf{x}_2 - \mathbf{x})^T\right\} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 1 & (\mathbf{x}_P - \mathbf{x})^T & \text{vech}^T\left\{(\mathbf{x}_P - \mathbf{x})(\mathbf{x}_P - \mathbf{x})^T\right\} & \cdots \end{bmatrix}, \tag{15}$$

with "diag" defining the diagonal elements of a diagonal matrix. Regardless of the order N, our primary interest is to compute an estimate of the image (pixel values), and the necessary computations are limited to those that estimate the parameter β_0. Therefore, the solution for the optimization problem is simplified to

$$\hat{z}(\mathbf{x}) = \hat{\beta}_0 = \mathbf{e}_1^T \left(\mathbf{X}_{\mathbf{x}}^T \mathbf{W}_{\mathbf{x}} \mathbf{X}_{\mathbf{x}}\right)^{-1} \mathbf{X}_{\mathbf{x}}^T \mathbf{W}_{\mathbf{x}}\, \mathbf{y}, \tag{16}$$

where $\mathbf{e}_1$ is a column vector with the first element equal to one, and the rest equal to zero. This estimator can be summarized using the spatially adaptive "equivalent" weight function W_i. That is

$$\hat{z}(\mathbf{x}) = \hat{\beta}_0 = \textstyle\sum_i W_i(\mathbf{x}_i, \mathbf{x}, N)\, y_i. \tag{17}$$

where in this case W_i depends upon the regression order N as well as the sample positions $\mathbf{x_i}$, and the position of interest $\mathbf{x}$.

If we seek a more robust formulation as in the case where the data contain outliers, the choice $m = 1$ may be more appropriate (9). Regardles, for any m, we can rewrite (9) in matrix form as

$$\hat{\mathbf{b}} = \arg\min_{\mathbf{b}} \left\| \mathbf{y} - \mathbf{X}_{\mathbf{x}}\mathbf{b} \right\|^m_{\mathbf{W}_{\mathbf{x}}}. \tag{18}$$

For a general m, a closed-form solution as in the one above for $m = 2$ is not practical and we must resort to an iterative estimator such as the steepest descent method. The steepest descent iterations for this case amount to the following:

$$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} + \mu \mathbf{X}_{\mathbf{x}}^T \mathbf{W}_{\mathbf{x}}\, \text{sign}\left(\mathbf{y} - \mathbf{X}_{\mathbf{x}}\hat{\mathbf{b}}^{(n)}\right) \odot \left|\mathbf{y} - \mathbf{X}_{\mathbf{x}}\hat{\mathbf{b}}^{(n)}\right|^{m-1}, \tag{19}$$

where μ is the step size, and $\odot$ is the element-by-element multiplication operator.

Returning to the estimation problem based upon (9), one can choose the order N to effect an increasingly more complex local approximation of the signal. In the statistics literature, locally constant, linear and quadratic approximations (corresponding to $N = 0, 1, 2$ respectively) have been considered most widely. In particular, choosing $N = 0$ (corresponding to local constant estimation), a spatially adaptive filter (2) is obtained from (17). Of course, higher order approximations ($N > 0$) are also possible, and the estimator (16) can be written in the generalized spatially adaptive filter (4) (q.v. [3]). What we concentrate on in the rest of this paper is the modification of the kernels. More specifically, we propose novel ways to adapt the kernels to local data. The result is a class of locally adaptive image

filters which are able to perform denoising with high quality, even at very low SNR[4], while being simultaneously applicable to interpolation of missing data.

In the interest of completeness, we mention that the normalized convolution method presented in [19] is very similar to kernel regression with the classic kernel function (10), which is a spatially adaptive filter introduced in this section. An edge adaptive version of this work is also very recently proposed in [20].

2.2. Data-Adaptive Filter

A strong denoising effect can be realized by making the global smoothing parameter h larger (or making the effective areas larger). However, with a larger h, the estimated image will be more blurred so that we have sacrificed details to the effect of denoising. In order to have both a strong denoising effect and a sharper image, one can consider an alternative approach that will adapt the local effect of the filter using not only the position of the nearby samples, but also their gray values, which yields us the data-adaptive version of kernel regression,

$$\min_{\{\boldsymbol{\beta}_n\}_{n=0}^{N}} \sum_{i=1}^{P} \left| y_i - \beta_0 - \boldsymbol{\beta}_1^T (\mathbf{x}_i - \mathbf{x}) - \boldsymbol{\beta}_2^T \operatorname{vech}\left\{ (\mathbf{x}_i - \mathbf{x}) (\mathbf{x}_i - \mathbf{x})^T \right\} - \cdots \right|^m \mathsf{K}(\mathbf{x}_i, \mathbf{x}, y_i, y) \,. \tag{20}$$

Correspondingly, for $m = 2$, we have

$$\hat{z}(\mathbf{x}) = \hat{\beta}_0 = \textstyle\sum_i W_i(\mathbf{x}_i, \mathbf{x}, y_i, y, N)\, y_i. \tag{21}$$

That is to say, the proposed kernels will take into account two factors: spatial distances and radiometric (gray value) distances. If the kernel choice is made such that it depends only on the *difference* between the positions and gray values of available pixels and the position/value of the pixel of interest, then the kernel function can be denoted as

$$\mathsf{K}(\mathbf{x}_i, \mathbf{x}, y_i, y) \Longrightarrow K(\mathbf{x}_i - \mathbf{x}, y_i - y). \tag{22}$$

We name this the *data-adaptive kernel* function, and discuss the selections of the function in this section.

2.2.1. Bilateral Kernel

A simple and intuitive choice of the data-adaptive kernel K is to use "separable" kernels for penalizing the spatial and radiometric distances. Indeed this is precisely the thinking behind the *bilateral* filter, introduced in [14], and carefully analyzed in [21]. Mathematically, the bilateral choice is given by

$$K(\mathbf{x}_i - \mathbf{x}_j, y_i - y_j) \equiv K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}_j) K_{h_r}(y_i - y_j), \tag{23}$$

where h_r is the radiometric smoothing parameter (a scalar value) that controls the rate of decay, and $K_{\mathbf{H}}(\cdot)$ and $K_{h_r}(\cdot)$ are the spatial and radiometric kernel functions, respectively. With this kernel, for the special case $N = 0$, the estimator (16) can be summarized as

$$\hat{z}(\mathbf{x}_j) = \frac{\sum_{i=1}^{P} K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}_j) K_{h_r}(y_i - y_j) y_i}{\sum_{i=1}^{P} K_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}_j) K_{h_r}(y_i - y_j)}. \tag{24}$$

In general, the values of h and h_r are fixed. While the bilateral kernel choice is attractive and simple, breaking K into the spatial and radiometric kernels separately can weaken the estimator performance if the SNR is very low. A simple justification for this claim comes from studying very noisy data sets, where radiometric distance $(y_i - y_j)$'s tend to be large and therefore all radiometric weights are very close to zero, and effectively useless. In the following, we present a better selection of kernels, which overcome this difficulty.

2.2.2. Steering Kernel

To pose the framework which takes us beyond the bilateral kernel choice, we observe that the effect of computing $K_{h_r}(y_i - y_j)$ in (23) 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

[4]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.

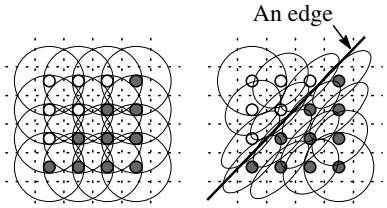


Fig. 6. (a) Standard kernels. (b) Steering kernels along a local edge.

estimator (say standard –spatially adaptive– kernel regression). Next this estimate is used to measure the dominant orientation of the local gradients in the image (e.g. [22]). 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 adaptive 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 adaptive kernel takes the form

$$K(\mathbf{x}_i - \mathbf{x}, y_i - y) \Longrightarrow K_{\mathbf{H}_i}(\mathbf{x}_i - \mathbf{x}), \tag{25}$$

where $\mathbf{H}_i$'s are the data-dependent symmetric, positive definite, matrices which we call *steering* matrices. They are defined as

$$\mathbf{H}_i = h\mu_i \mathbf{C}_i^{-\frac{1}{2}}, \tag{26}$$

where $\mathbf{C}_i$'s are covariance matrices based on the gradients of the local gray-values. The principal directions indicated by a good choice for $\mathbf{C}_i$ will effectively spread the kernel function along the local edges as shown in Fig. 6. 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.

More specifically, the local edge structure is related to the gradient covariance (or equivalently, the locally dominant orientation) in an analysis window around the position of interest. The dominant local orientation of the gradients is then related to the eigenvectors of this estimated matrix. While this approach (which is essentially a local principal components method) is simple and has nice tolerance to noise, the resulting estimate of the covariance may be rank deficient, and therefore care must be taken not to take the inverse of the estimate directly in this case. In [22], we proposed an effective *multiscale* technique for estimating local orientations, which fits the requirements of this problem nicely.

In order to have a convenient and intuitive form of the covariance matrix, we decompose it into three components as follows:

$$\mathbf{C}_i = \gamma_i \mathbf{U}_{\theta_i} \mathbf{\Lambda}_i \mathbf{U}_{\theta_i}^T, \tag{27}$$

$$\mathbf{U}_{\theta_i} = \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{bmatrix}, \quad \mathbf{\Lambda}_i = \begin{bmatrix} \sigma_i & 0 \\ 0 & \sigma_i^{-1} \end{bmatrix}. \tag{28}$$

where $\mathbf{U}_{\theta_i}$ is the rotation matrix and $\mathbf{\Lambda}_i$ is the elongation matrix. 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.7 explains schematically how these parameters affect the spreading of kernels. A simple choice of γ_i, for instance is a geometric mean of the eigenvalues of $\mathbf{C}_i$. Such a γ_i makes the steering kernel area large in low frequency areas and small in high frequency areas [3]. To complete the description of our framework, we note that we implement the steering kernel regression in an iterative setting as follows: (i) initial estimation of gradient images, (ii) compute steering matrices, (iii) apply steering kernel regression, which yield an estimated image and gradient images, (iv) repeat (ii) and (iii) several more times. Due to the page limitation, we direct the interested readers to a more detailed description about the iteration method in [3].

Fig.8 illustrates another simulation and shows the power of data-adaptive methods. This simulation uses the blurry image shown which is shown in Fig.8(a) and capturing two stars; a brighter one and a darker one. We added white Gaussian noise (SNR = 20[dB]) to the original image. The resulting noisy image is shown in Fig.8(b). Next, we irregularly downsampled the noisy image (discarded 80% of the pixels of the noisy image where these missing pixels are in green) as shown in Fig.8(d). The reconstructed images by the classic drizzle algorithm ($pixfrac = 0.3$) and iterative steering kernel regression ($N = 2$, $h = 5.3$, and one iteration) are in Fig.8(d) and Fig.8(e), respectively, where the corresponding PSNR's are 28.91[dB] and 31.74[dB]. Furthermore, Fig.8(f) illustrates the comparison between the

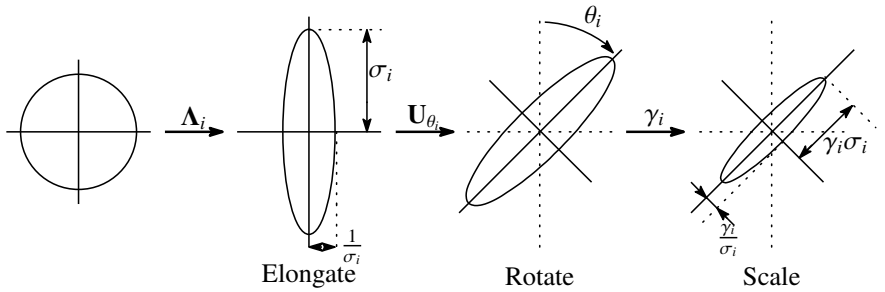
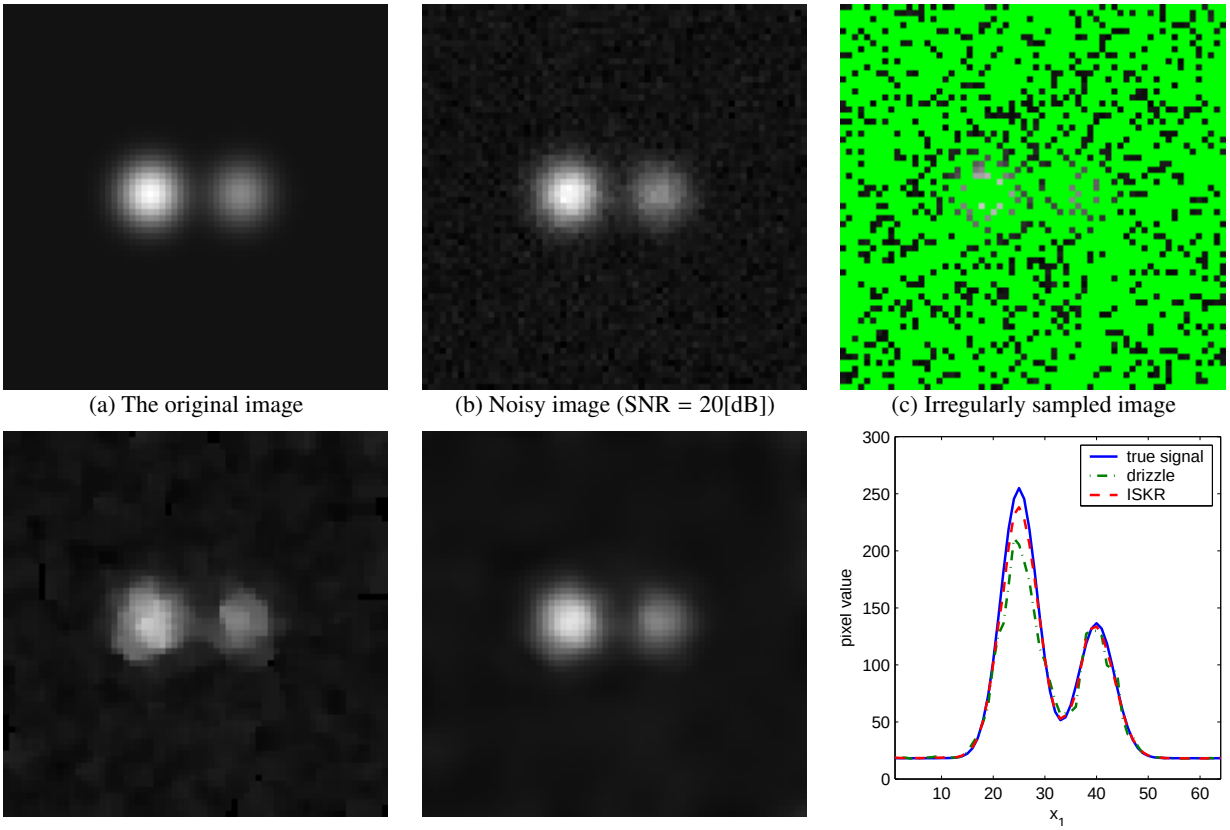


Fig. 7. The schematic representation of the steering kernel by the parameterized $\mathbf{C}_i = \gamma_i \mathbf{U}_{\theta_i} \Lambda_i \mathbf{U}_{\theta_i}^T$.



(a) The original image (b) Noisy image (SNR = 20[dB]) (c) Irregularly sampled image

(d) Drizzle algorithm (e) Steering kernel reg. ($N = 2$) (f) Comparison

Fig. 8. Another simulation of image reconstruction from noisy and irregularly sampled data: (a)The original (blurry) image, (b)Noisy image, (c)Irregularly sampled image (the green pixels are representing missing pixels.), (d)The reconstructed image by the drizzle algorithm, (e)The reconstructed image by the iterative steering kernel regression, and (f)Comparison between the drizzle algorithm and steering kernel regression by plotting the middle cross-section of the respective images: the solid (blue) line is the original image, the dash-dot (green) line is the drizzle algorithm, and the dash (red) line is the iterative steering kernel regression. The bias and variance of the drizzle algorithm by a Monte-Carlo simulation (by using different realizations of measurement noise with the same sampling positions shown in (c)) are Bias = 0.298 and Var = 17.99, and the ones of iterative steering kernel regression are Bias = 0.00065 and Var = 7.43.

drizzle algorithm and steering kernel regression by plotting one horizontal cross-section in the middle of the each reconstructed image. As seen in the reconstructed images and the comparison, steering kernel regression did a better job visually and photometric-wise.

3. EXPERIMENTS

The first experiment is an example of image reconstruction from a real astronomical data set, which is shown in Fig.9(a). The real data have some outliers due to the bad image sensors. The black dots of Fig.9(b) are representing

those bad ones. Moreover, because illumination of each image sensors (detectors) is not uniform, we correct the raw pixel value by using the flat field data shown in Fig.9(c). Fig.9(d) shows the compensated image by both Fig.9(b) and Fig.9(c). The green pixels are representing missing pixels. As seen in Fig.9(d), the outliers are eliminated, but the data set has some missing pixels and noise. We reconstructed the image using both the drizzle algorithm (with $pixfrac = 0.99$) and using the iterative steering kernel regression (with $N = 2$, $h = 15.0$, and one iteration), which are shown in Fig.9(e) and Fig.9(f), respectively. To compare the performance between the drizzle algorithm and steering kernel regression, Fig.10 illustrates the absolute residual images (absolute difference between the estimated images and the noisy image). The residual images of steering kernel regression (Fig.10(a)) is noise-like, whereas on the other hand, we see structure (namely) stars in the residual image corresponding to the reconstruction using the drizzle algorithm (Fig.10(b)). That is to say, the drizzle algorithm unexpectedly remove some image contents from the original data set. In addition, Fig.10(c) and Fig.10(d) illustrate details from these residual images which make the point more clearly.

The second experiment is concerned with fusion of frames and simultaneous cosmic ray noise reduction. We downloaded two images of NGC5195, known as the companion of the spiral galaxy Messier 51, from the astronomical data archives[5], which are taken by HST and illustrated in Fig.11(a) and Fig.11(b). They are corrupted by noise caused by cosmic rays, and each has a gap in the middle, which is indicated in green color. Unlike the previous experiment, the exact positions of the outliers are unknown; in other words, we do not have a mask for bad pixels. If we apply the drizzle algorithm directly to the data set, we have the reconstructed image illustrated in Fig.11(c) (we used $pixfrac = 1.0$). Although Fruchter and Hook presented a method to create a mask of bad pixels for this kind of data sets in [1], the method includes several steps. However, in this situation, the kernel regression with $m = 1$ (robust kernel regression) helps us to reduce cosmic ray noise without the use of the bad pixel mask. After estimating the relative position between these two frames, we processed the two images and reconstructed the image shown in Fig.11(d) which employed classic kernel regression ($N = 2$, $m = 1$, and $h = 0.8$).

The third experiment is concerned with cosmic ray noise reduction and simultaneous resolution enhancement, or super-resolution, on a real data set. We have 5 frames of Cotton candy nebula[6]. All the frames are shown in Fig.12(a) through Fig.12(e). Directly applying the drizzle algorithm with $pixfrac = 1.0$ and classic kernel regression ($N = 2$, $m = 1$, and $h = 1.0$), we reconstruct an image having twice as many pixels in both vertical and horizontal directions as the original ones. Fig.12(f) and Fig.12(g) show the estimated images. Both of the methods produced good interpolated image. However, without the pre-processing (creating the mask), the drizzle algorithm cannot suppress the cosmic ray noise.

4. CONCLUSION

In this paper, we defined and discussed spatially adaptive filters and contrasted them to the more general and superior data-adaptive filters, which can be used for both denoising and interpolation. Simulations and experiments confirmed that data-adaptive filters have excellent performance not only on regularly sampled date, but also on irregularly sampled data sets. Moreover, unlike the drizzle algorithm, we also showed that applying data-adaptive kernel regression can suppress cosmic ray noise effectively, and without the numerous steps involved in doing the same using the drizzle algorithm.

There is another important aspect of the drizzle algorithm presented in [1], which concerns correcting geometric distortion effects. The algorithm corrects these by transforming the effective areas in the same way as the estimated distortions undergone by the frames. It is worth noting that the same can be done in the general kernel regression framework without much additional complication in implementation.

Since the kernel regression framework is built on a very general data model, its applicability is extremely wide. Applying it to astronomical images is only an example. We have demonstrated its applicability to many more data sources for example in [3]. Furthermore, although we discussed the framework in two dimensions, kernel regression is, of course, able to process (denoise and interpolate) higher-dimensional data sets as well.

[5]http://archive.stsci.edu/

[6]The data is available at www.stsci.edu/instruments/wfpc2/dither/examples.html

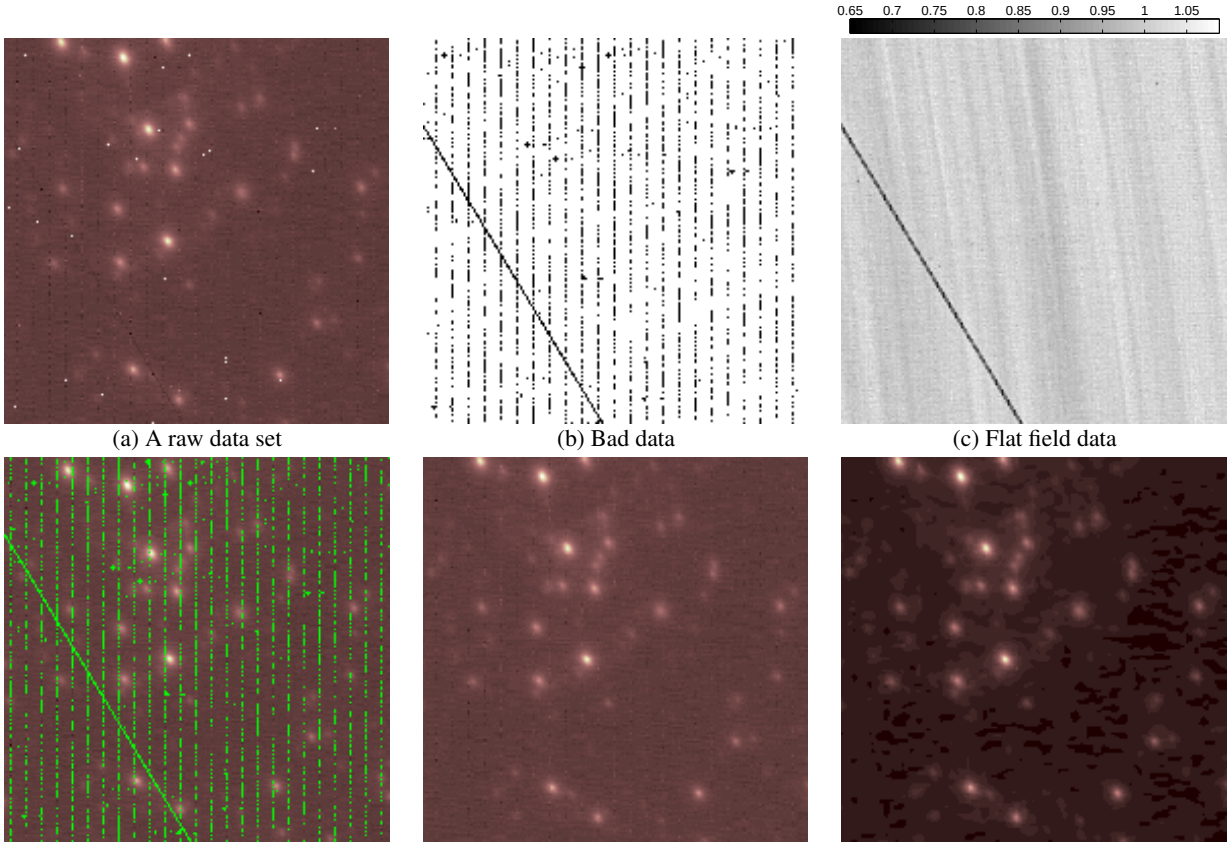
ACKNOWLEDGEMENT

We would like to thank Dr. Marcos Van Dam, MAST (Multimission Archive at STScI), and Dr. Anton M. Koekemoer for providing astronomical image data sets shown in Fig.9, Fig.11, and Fig.12, respectively.

This work was supported in part by the US Air Force Grant F49620-03-1-0387, and by the National Science Foundation Science and Technology Center for Adaptive Optics, managed by the University of California at Santa Cruz under Cooperative Agreement No. AST-9876783.

REFERENCES

1. A. S. Fruchter and R. N. Hook, "Drizzle: A method for the linear reconstruction of undersampled images," *Publications of the Astronomical Society of the Pacific*, vol. 114, pp. 144–152, February 2002.
2. M. P. Wand and M. C. Jones, *Kernel Smoothing*, ser. Monographs on Statistics and Applied Probability. London; New York: Chapman and Hall, 1995.
3. H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," accepted to IEEE Transactions on Image Processing.
4. R. H. T. Bates and F. M. Cady, "Towards true imaging by wideband speckle interferometry," *Optics Communications*, vol. 32, no. 3, pp. 365–369, March 1980.
5. J. C. Christou, E. K. Hege, D. Freeman, and E. Ribak, "Self-calibrating shift-and-add technique for speckle imaging," *Journal of the Optical Society of America A-Optics Image Science and Vision*, vol. 3, no. 2, pp. 204–209, February 1986.
6. A. M. Sinton, B. L. K. Davey, and R. H. T. Bates, "Augmenting shift-and-add with zero-and-add," *Journal of the Optical Society of America A-Optics Image Science and Vision*, vol. 3, no. 7, pp. 1010–1017, July 1986.
7. E. Ribak, "Astronomical imaging by filtered weighted-shift-and-add technique," *Journal of the Optical Society of America A-Optics Image Science and Vision*, vol. 3, no. 12, pp. 2096–2076, December 1986.
8. M. C. Chiang and T. E. Boulte, "Efficient super-resolution via image warping," *Image and Vision Computing*, vol. 18, no. 10, pp. 761–771, July 2000.
9. M. Elad and Y. Hel-Or, "A fast super-resolution reconstruction algorithm for pure translational motion and common space-invariant blur," *IEEE Transactions on Image Processing*, vol. 10, no. 8, pp. 1187–1193, August 2001.
10. S. Lertrattanapanich and N. K. Bose, "High resolution image formation from low resolution frames using delaunay triangulation," *IEEE Transactions on Image Processing*, vol. 11, no. 12, pp. 1427–1441, December 2002.
11. S. Farsiu, D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multi-frame super-resolution," *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1327–1344, October 2004.
12. E. A. Nadaraya, "On estimating regression," *Theory of Probability and its Applications*, pp. 141–142, September 1964.
13. S. M. Smith and J. M. Brady, "Susan – a new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997.
14. C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *Proceeding of the 1998 IEEE International Conference of Compute Vision, Bombay, India*, pp. 836–846, January 1998.
15. D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 5, pp. 603–619, May 2002.
16. A. Buades, B. Coll, and J. M. Morel, "A review of image denosing algorithms, with a new one," vol. 4, no. 2, pp. 490–530, 2005.
17. D. Ruppert and M. P. Wand, "Multivariate locally weighted least squares regression," *The annals of statistics*, vol. 22, no. 3, pp. 1346–1370, September 1994.
18. B. W. Silverman, *Density Estimation for Statistics and Data Analysis*, ser. Monographs on Statistics and Applied Probability. London; New York: Chapman and Hall, 1986.
19. H. Knutsson and C. F. Westin, "Normalized and differential convolution - methods for interpolation and filtering of incomplete and uncertain data," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Regocnition (CVPR)*, pp. 515–523, June 1993.
20. T. Q. Pham, L. J. van Vliet, and K. Schutte, "Robust fusion of irregularly sampled data using adaptive normalized convolution," *EURASIP Journal on Applied Signal Processing, Article ID 83268*, 2006.
21. M. Elad, "On the origin of the bilateral filter and ways to improve it," *IEEE Transactions on Image Processing*, vol. 11, no. 10, pp. 1141–1150, October 2002.

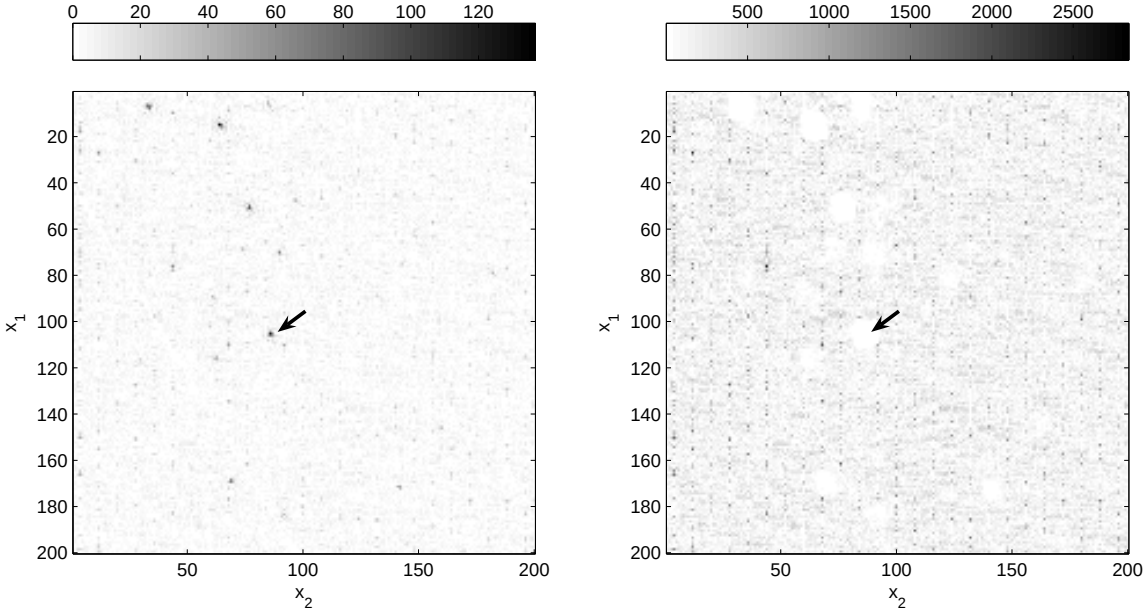


(a) A raw data set (b) Bad data (c) Flat field data

(d) The compensated image (e) Drizzle algorithm (f) Steering kernel reg. (N=2)

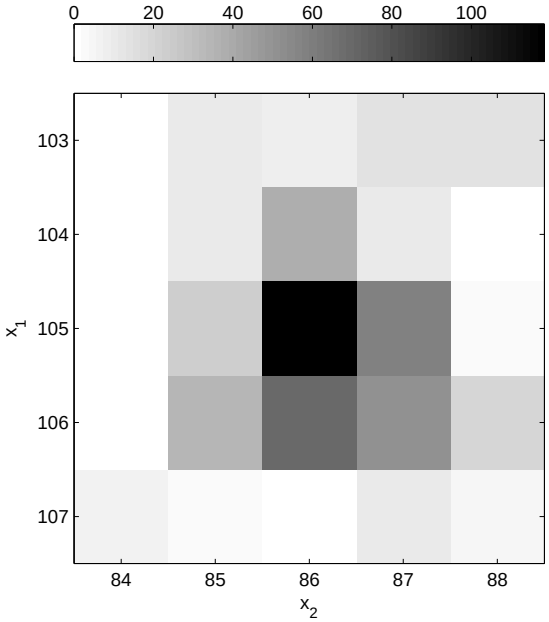
Fig. 9. Image reconstruction on a real data set: (a)A raw data set, (b)Bad data (the black dots are representing the bad pixels.), (c)Flat field data, (d)The compensated image (the green pixels are representing missing pixels.) by the bad data mask and the flat field data, and (e)-(f)The reconstructed images by the drizzle algorithm and iterative steering kernel regression, respectively.

22. X. Feng and P. Milanfar, "Multiscale principal components analysis for image local orientation estimation," *Proceedings of the 36th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA*, November 2002.

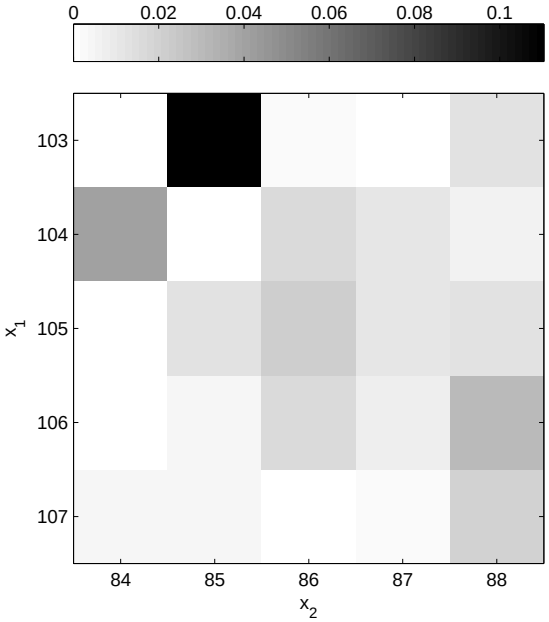


(a) Drizzle algorithm

(b) Steering kernel reg. (N=2)

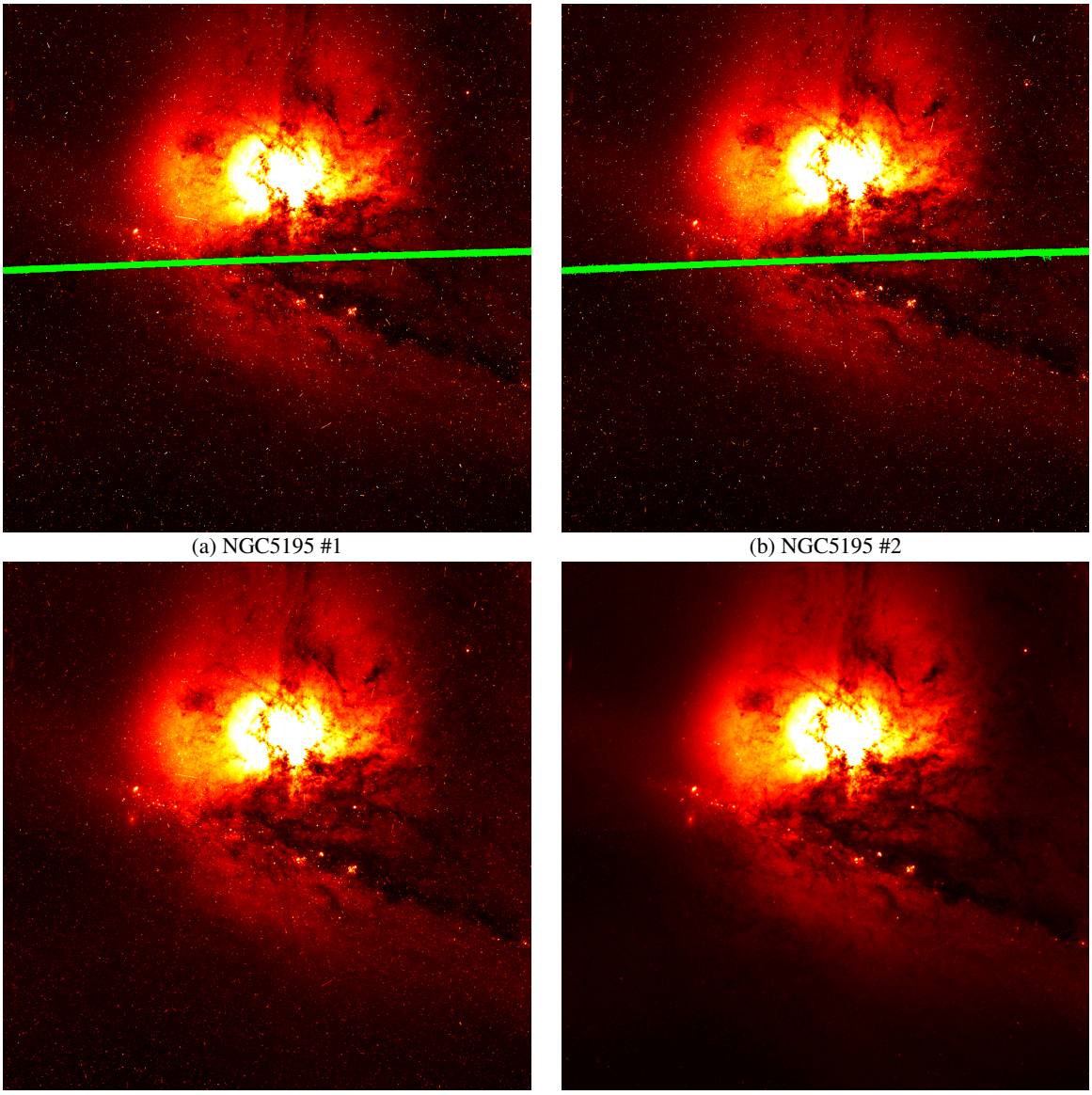


(c) Drizzle algorithm



(d) Steering kernel reg. (N=2)

Fig. 10. The absolute residual images (absolute difference between the estimated images and the noisy image): (a)The drizzle algorithm, (b)Iterative steering kernel regression, and (c)-(d)The selected regions of the each residual image around the sections where the arrows point. While the image (d) illustrate the residual in the very narrow scale as the scale bar shows, it is still noisy-like, which indicates that the iterative steering kernel regression removed noise only, unlike the drizzle algorithm.



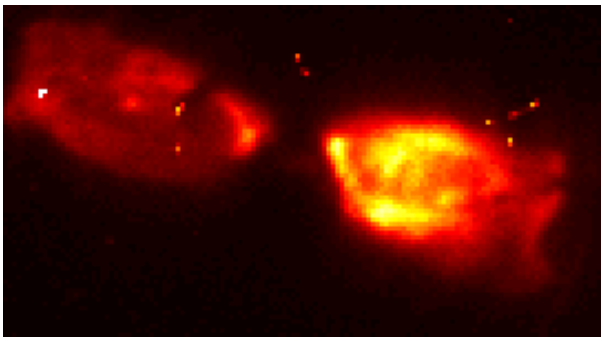
(a) NGC5195 #1

(b) NGC5195 #2

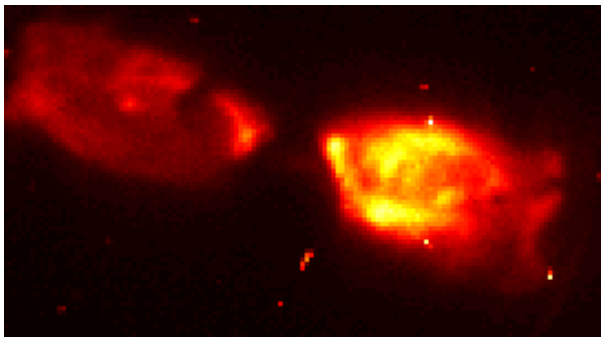
(c) Drizzle algorithm

(d) L_1 classic kernel reg. (N=2)

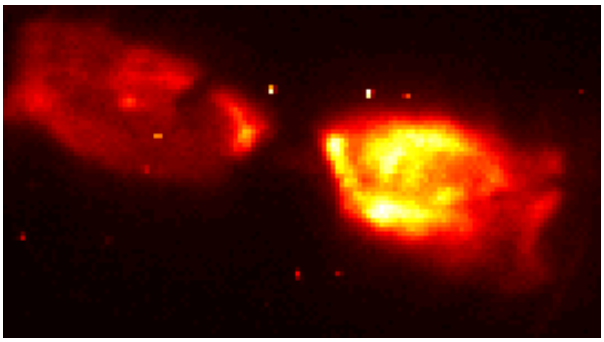
Fig. 11. An example of the cosmic ray reduction on NGC5195: (a)-(b)The first and second input frames (the green pixels are representing missing pixels.), (c)The reconstructed image by applying the drizzle algorithm directly (without creating the mask for the outliers), (d)The reconstructed image by L_1 classic kernel regression.



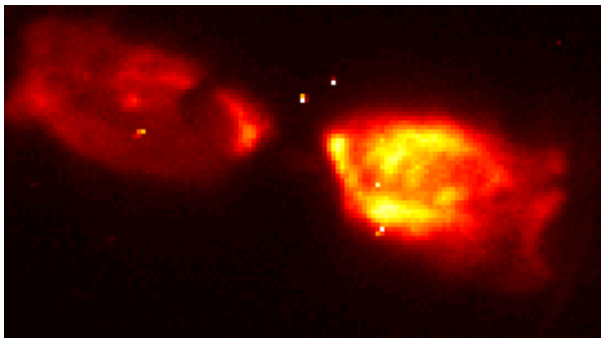
(a) The first frame



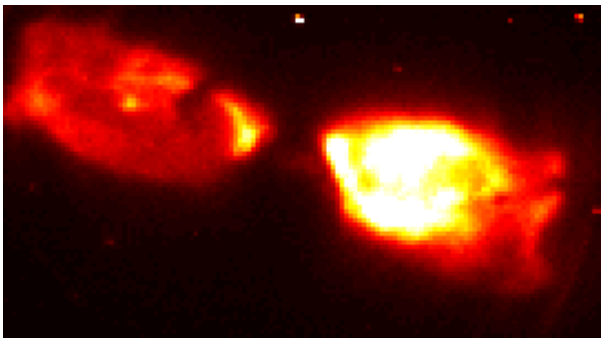
(b) The second frame



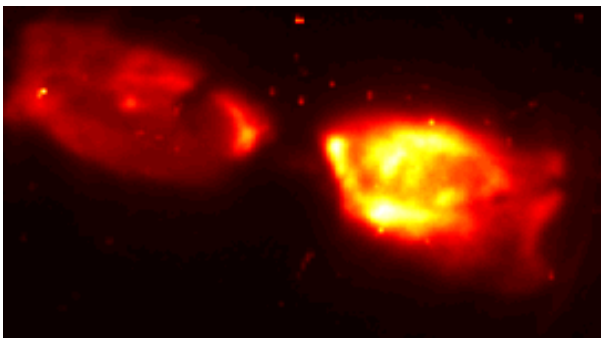
(c) The third frame



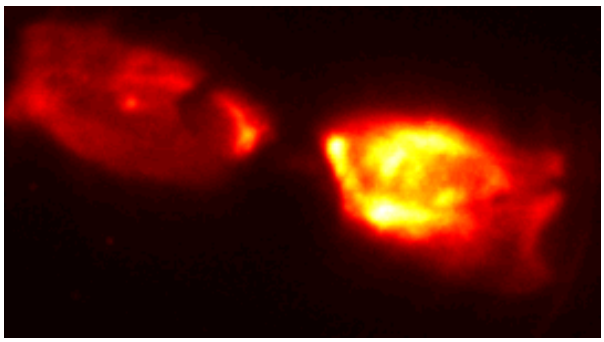
(d) The fourth frame



(e) The fifth frame



(f) Drizzle algorithm



(g) L_1 Classic kernel reg. (N=2)

Fig. 12. A super-resolution example on the cotton candy nebula sequence: (a)-(e)The first through fifth frames, (f)The reconstructed image by applying the drizzle algorithm directly (without creating the bad pixel mask), and (g)The image reconstructed by L_1 classic kernel regression.