

Locally Weighted Least Squares Regression for Image Denoising, Reconstruction and Up-sampling

Moritz Baecher

May 15, 2009

1 Introduction

Edge-preserving smoothing and super-resolution are classic and important problems in computational photography and related fields. The first addresses the problem of noise removal from images while preserving its sharp features such as strong edges. Image up-sampling, on the other hand, addresses the problem of spatial resolution enhancement of images. The goal is to obtain a high-resolution version of an image by sampling it on a denser lattice. This work investigates an image processing framework based on *locally weighted kernel regression*. A major strength of the presented tools is its generality. It allows to not only address both of the above-mentioned problems but also to reconstruct images from irregularly and sparse sampled pixels in a single framework.

There are two major kinds of techniques that use locally weighted least squares in the context of image processing: *Moving least squares* (MLS) and *kernel regression* (KR). Both formulations are very similar, but their origin is different. KR has its origins in statistics and is usually formulated in terms of a nonparametric estimation problem. MLS, on the other hand, is more seen as a powerful local scattered data approximation technique. Both techniques are well-established and their theoretical properties are well-studied, but practically speaking they both have the same goal:

Given a set of data points $(\mathbf{x}_i, f_i) \in \mathbb{R}^d \times \mathbb{R}$, both KR or MLS can be used to find a global function $f(\mathbf{x})$ that approximates or interpolates the (measured) values f_i at the locations \mathbf{x}_i .

An image in its continuous form can be interpreted as a two dimensional function $I(x, y)$ where the pair (x, y) denotes the spatial coordinates. A discrete image, on the other hand, is a set of intensity values I_i at discrete pixel positions (x_i, y_i) . KR and MLS seem therefore to be appropriate ways of deriving a continuous image from those discrete pixel intensities. This continuous image representation can then be resampled and therefore allows applications such as up-sampling and reconstruction.

In this work, we will take on the KR-glasses. The major motivation behind this choice is that we use a formulation based on a local Taylor expansion which is commonly used in the kernel regression theory. The use of a Taylor polynomial allows us to directly compute the resulting image and also – as an option – derivatives. It is therefore computationally more efficient and direct than a MLS-based formulation which usually involves two computation steps to get the same result: First, coefficients of a linear combination of a set of monomials is fitted and, in a second step, these coefficients are used to evaluate the resulting approximation at the current location. However, this choice comes with a cost. Formulations based on Taylor expansions are less intuitive and interpretable. It is also important to mention that the resulting approximation is the same, if the order of the Taylor expansion (KR) and the polynomial (MLS) is the same.¹

A classic linear KR method allows us already to up-sample, smooth and reconstruct images, but high-frequent features such as edges are not preserved and outliers (salt and pepper noise) are not handled very well. There are two major ways of solving those problems. Either we make the kernel function *data-dependent* or we use a *robust* local KR estimator. Data-dependent kernels lead to higher order bilateral filters and allow feature-preserving smoothing. However, the performance of these non-linear bilateral filters highly decreases when applied to images with salt and pepper noise. There, the robust non-linear KR estimator outperforms both, linear and bilateral filters, to a high extend.

1.1 Overview

We will start our discussion with a short overview over the previous work done in the area of MLS and KR-based techniques for image processing (Section 2). We then introduce the KR framework for images and discuss both non-linear extensions in detail (Section 3). In the results section (Section 4), we compare the performance of the linear KR method with its two non-linear versions. We refer to those three techniques as:

- *classic* kernel regression (linear)
- *bilateral* kernel regression (non-linear)
- *robust* kernel regression (non-linear)

¹Taylor polynomials and monomials span the same space – only the fitted coefficients are different. This only holds, of course, if we use the same local (kernel) weights.

2 Related Work

Image denoising, reconstruction and upsampling are difficult and fundamental image processing problems. As a result, many techniques have been suggested in recent years. Most of them focus on one specific problem and present a novel model that leads to more compelling results under certain assumptions. An example of a recent paper that addresses the problem of image up-sampling is given in [3]. As opposed to most other techniques, it relies on edge dependencies and not on some sort of smoothness assumption. Their up-sampling results outperform the ones presented in this work.

However, our goal here is a framework that allows to address not only super-resolution, but also the problems of denoising and reconstruction. Its major strength is its generality.

Among those general techniques, MLS and KR-based methods seem to be most widely used. We therefore restrict our discussion of related work to those two methods:

MLS-based methods. An early paper that uses MLS in the context of image denoising and reconstruction is given in [4]. They use two different kinds of robust estimators that minimize local differences in a weighted l_1 and l_0 sense. They present them as generalized bilateral filters and show rather preliminary results. A more recent paper [1] applies MLS to superresolution and noise filtering. It focuses more on investigating the effect of the two major parameters, scale (sigma in Gaussian weights) and approximation order, and presents a cross-validation technique to find the best parameters automatically. They also show that MLS and KR are closely related concepts. Their discussion, however, is restricted to the classic linear case.

KR-based methods. A paper that discusses multivariate locally weighted least squares regression and presents derivations for bias and variance of the underlying regression estimator is given in [6]. We use a similar notation to derive the bivariate formulation in this work. A paper that uses a KR-based method as an edge-preserving smoother is given in [2]. The later uses a robust non-linear estimator similar to the one we use in this work. Most related to the presented work here are the papers by Takeda et al. [8] and [7]. They apply KR in the context of image denoising, reconstruction and also up-sampling. They discuss higher order bilateral filters based on a data-dependent kernel and also present a novel kernel (steering kernel) that leads to better results. Robust estimators, however, are not used in their work.

A paper that uses KR in the context of surface reconstruction from point clouds is given in [5]. We also use a robust local KR estimator based on Welsch's objective function in this work and use *iteratively reweighted least squares* (IRLS) to solve the resulting regression problem.

3 A Kernel Regression Framework for Images

It follows an as-short-as-possible overview over bivariate kernel regression.

An ideal intensity image of a scene can be represented as a 2D function $I(\mathbf{x})$ that captures an intensity value at every continuous location $\mathbf{x} = (x, y)$.

If the same scene is captured by a photographic device, we end up with a set of P intensity values I_i at discrete pixel locations $\mathbf{x}_i = (x_i, y_i)$. This imaging process, however, is not perfect and the resulting intensity samples are noisy:

$$I_i = I(\mathbf{x}_i) + \epsilon_i, \quad i = 1, 2, \dots, P \quad (1)$$

where the ϵ_i 's are assumed to be identically distributed zero mean random variables.

If we assume that the original image I is locally smooth to an order N , the intensity value at a location \mathbf{x} near a pixel \mathbf{x}_i can be expressed as a local N -term Taylor expansion:

$$I(\mathbf{x}_i) \approx I(\mathbf{x}) + \nabla I(\mathbf{x})^T \Delta \mathbf{x}_i + \frac{1}{2} \Delta \mathbf{x}_i^T \nabla^2 I(\mathbf{x}) \Delta \mathbf{x}_i + \dots \quad (2)$$

where $\Delta \mathbf{x}_i = \mathbf{x}_i - \mathbf{x}$.

If we restrict our discussion to orders not higher than $N = 2$, we can simplify expression 2 as ²:

$$I(\mathbf{x}_i) = \beta_0 + [\beta_1, \beta_2]^T \Delta \mathbf{x}_i + [\beta_3, \beta_4, \beta_5]^T vech \{ \Delta \mathbf{x}_i \Delta \mathbf{x}_i^T \} \quad (3)$$

where the half-vectorization operator $vech(\cdot)$ is defined as

$$vech \left\{ \begin{bmatrix} a & b \\ c & d \end{bmatrix} \right\} = [a, b, d]^T \quad (4)$$

If we compare the expressions 3 and 2, we observe that β_0 is the *intensity image of interest* and the rest of the β_i ($i = 1, 2, 3, 4, 5$) are (scaled) first and second partial *derivatives* of it:

$$\beta_0 = I(\mathbf{x}) \quad (5)$$

$$\beta_1 = \frac{\partial I(\mathbf{x})}{\partial x} \quad (6)$$

$$\beta_2 = \frac{\partial I(\mathbf{x})}{\partial y} \quad (7)$$

$$\beta_3 = \frac{1}{2} \frac{\partial^2 I(\mathbf{x})}{\partial x^2} \quad (8)$$

$$\beta_4 = \frac{\partial^2 I(\mathbf{x})}{\partial x \partial y} \quad (9)$$

²we are making use of the symmetry property of the Hessian matrix $\nabla^2 I(\mathbf{x})$ here

$$\beta_5 = \frac{1}{2} \frac{\partial^2 I(\mathbf{x})}{\partial y^2} \quad (10)$$

The expression 3 can be further simplified and written as an inner product of two vectors

$$I(\mathbf{x}_i) = \mathbf{d}_{\mathbf{x}_i, \mathbf{x}}^T \mathbf{b} \quad (11)$$

where $\mathbf{d}_{\mathbf{x}_i, \mathbf{x}} = [1, \Delta \mathbf{x}_i^T, vech \{ \Delta \mathbf{x}_i \Delta \mathbf{x}_i^T \}^T]^T$ and $\mathbf{b} = [\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5]^T$.

We can compute the vector \mathbf{b} for an arbitrary location \mathbf{x} by minimizing the following locally weighted energy in a least squares sense:

$$\min_{\mathbf{b}} \sum_{i=1}^P (\mathbf{d}_{\mathbf{x}_i, \mathbf{x}}^T \mathbf{b} - I_i)^2 K_{h_s}(\|\Delta \mathbf{x}_i\|) \quad (12)$$

where $K_{h_s}(\|\Delta \mathbf{x}_i\|)$ denotes the kernel function with the spatial smoothness parameter h_s . We will specify the used kernels below.

The expression 12 can be recast in matrix form:

$$\hat{\mathbf{b}} = \arg \min_{\mathbf{b}} \|\sqrt{\mathbf{K}_{\mathbf{x}}} (\mathbf{D}_{\mathbf{x}} \mathbf{b} - \mathbf{I})\|^2 \quad (13)$$

where

$$\mathbf{I} = [I_1, I_2, \dots, I_P]^T \quad (14)$$

denotes the vector with the pixel intensities³,

$$\mathbf{K}_{\mathbf{x}} = diag \{ K_{h_s}(\|\Delta \mathbf{x}_1\|), K_{h_s}(\|\Delta \mathbf{x}_2\|), \dots, K_{h_s}(\|\Delta \mathbf{x}_P\|) \} \quad (15)$$

the vectors with the weights and

$$\mathbf{D}_{\mathbf{x}} = [\mathbf{d}_{\mathbf{x}_1, \mathbf{x}}^T, \mathbf{d}_{\mathbf{x}_2, \mathbf{x}}^T, \dots, \mathbf{d}_{\mathbf{x}_P, \mathbf{x}}^T]^T \quad (16)$$

the matrix dependent on all the deltas (pixel location differences).

The vector $\hat{\mathbf{b}}(\mathbf{x})$ can be found by solving the corresponding normal equations⁴

$$\hat{\mathbf{b}}(\mathbf{x}) = (\mathbf{D}_{\mathbf{x}}^T \mathbf{K}_{\mathbf{x}} \mathbf{D}_{\mathbf{x}})^{-1} \mathbf{D}_{\mathbf{x}}^T \mathbf{K}_{\mathbf{x}} \mathbf{I} \quad (17)$$

and the pixel intensity at the location \mathbf{x} is given by

$$\hat{I}(\mathbf{x}) = \mathbf{e}_1^T \hat{\mathbf{b}}(\mathbf{x}) \quad (18)$$

We use Gaussian-based kernels of the form

$$K_h(s) = \exp \left(-\frac{1}{2} \frac{s^2}{h^2} \right) \quad (19)$$

³ $\sqrt{\mathbf{A}}$ of a square, diagonal matrix \mathbf{A} of size n is defined as $diag \{ \sqrt{a_{11}}, \sqrt{a_{22}}, \dots, \sqrt{a_{nn}} \}$

⁴Note that the actual size of the system 17 is a lot smaller because of the local weighting (the influence of pixels far from \mathbf{x} is negligible).

in this work.

The framework that we discussed so far already allows us to reconstruct, upsample and also denoise images. We just have to evaluate equations 17 and 18 at every pixel location at the desired resolution. Note that the P input pixels do not have to lie on a regular grid (allows reconstruction). Edges, however, are smoothed and are not well-preserved. An other problem with the current technique is that it is not very robust in case of outliers (as we will see in the next Section 4). We will discuss non-linear extensions that address those problems in the next two subsections.

3.1 Higher Order Bilateral Filters

One way of improving the performance of the regression technique on edges is to make the kernels data-dependent. This leads to a so-called bilateral kernel:

$$K_{bilateral}(\|\mathbf{x}_i - \mathbf{x}\|, I_i - I(\mathbf{x})) = K_{h_s}(\|\mathbf{x}_i - \mathbf{x}\|)K_{h_d}(I_i - I(\mathbf{x})) \quad (20)$$

This types of kernels not only depends on pixel locations, but also on the intensity values at those locations. Bilateral kernels, therefore, adapt to local image features such as edges and are able to preserve them better.

However, such a kernel is not directly applicable if we try to upsample or reconstruct an image. The reason is that the intensity $I(\mathbf{x})$ is not known for all pixel locations \mathbf{x} . This limitation can be overcome by computing an initial estimate of I by using a simple interpolation scheme.

3.2 Linear vs. Robust Non-Linear Kernel Regression

Classic linear and also bilateral KR assumes the involved noise to be uniform. As a consequence, even a single outlier in the data can have a significant influence on the resulting image. To overcome this problem, a robust local estimator can be used. We use a so called ψ -type M-estimator here, similar to [5]. The idea is simple. Instead of minimizing the ordinary least squares criterion (see Figure 1 left column), a different objective function is used that gives outliers less weight (see Figure 1 right column). ψ -type estimators use differentiable objective functions and therefore lead to efficient minimization procedures. We employ Welsch's objective function (see Figure 1 right column) in

$$\min_{\mathbf{b}} \sum_{i=1}^P \rho(\mathbf{d}_{\mathbf{x}_i, \mathbf{x}}^T \mathbf{b} - I_i) K_{h_s}(\|\Delta \mathbf{x}_i\|) \quad (21)$$

and use iteratively reweighted least squares (IRLS)⁵ to minimize the resulting minimization problem:

$$\mathbf{b}^k = \min_{\mathbf{b}} \sum_{i=1}^P (\mathbf{d}_{\mathbf{x}_i, \mathbf{x}}^T \mathbf{b} - I_i)^2 w(r_i^{k-1}) K_{h_s}(|\Delta \mathbf{x}_i|) \quad (22)$$

where $r_i^{k-1} = \mathbf{d}_{\mathbf{x}_i, \mathbf{x}}^T \mathbf{b}^{k-1} - I_i$ denotes the i-th residual at the k-1-th iteration. We further initialize $w(r_i^0)$ to 1. There is a simple interpretation of the above equation: give the points with a high residual (potential outlier) a small weight. This iterative scheme seems to converge in a few steps.

⁵See <http://research.microsoft.com/en-us/um/people/zhang/inria/publications/tutorial-estim/node24.html> for more information on ψ -type M-estimators and IRLS

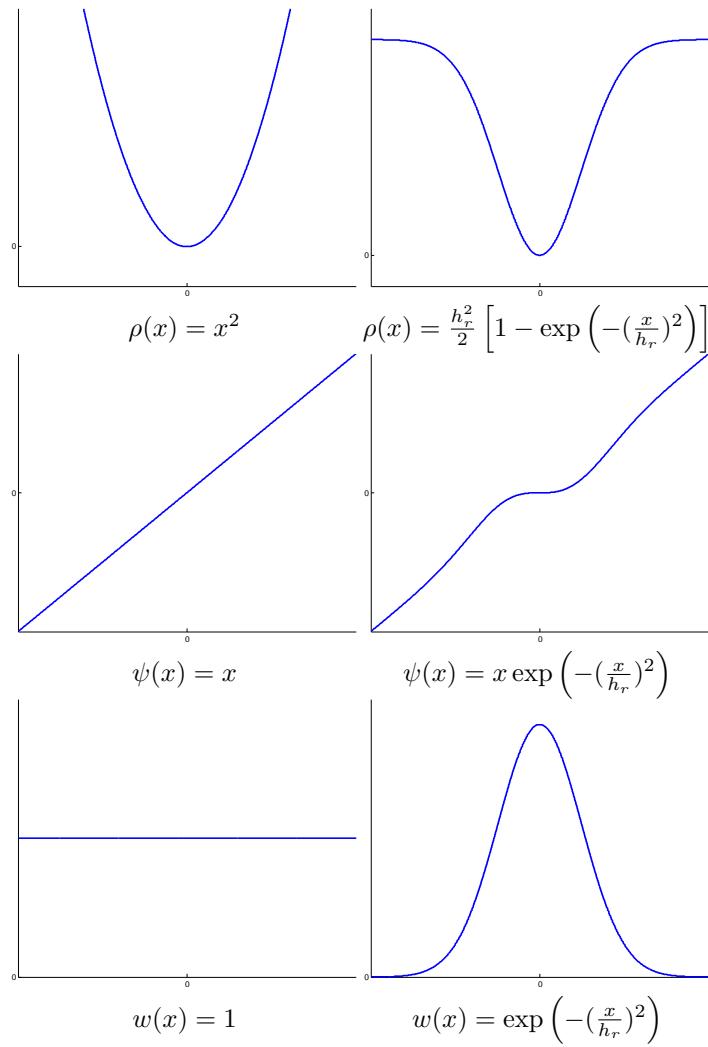


Figure 1: L2-norm vs. Welsch's error norm

4 Results

4.1 A First Example - 1D Step Function

In a first example, a 1D step function is sampled on a regular grid resulting in a 100 data points with $x_i \in [0, 1.0]$ and $y_i \in \{0.5, 1.0\}$. In a second step, noise with variance 0.001 is added to the y_i 's (see blue points in Figure 2 (a)). If we add an outlier (point with $y_i = 1.5$ to the data set, we get the plot in Figure 2 (b)).

We see that the bilateral KR (green) shows the best performance on the strong edge while smoothing the noise otherwise. However, if an outlier is added to the data set, robust KR (blue) outperforms both, classic KR and bilateral KR clearly. See table 1 for a summary of the resulting mean squared errors (MSE).

| | N | noisy data (MSE $\times 10^{-2}$) | noisy data with outlier (MSE $\times 10^{-2}$) |
|--------------|-----|---------------------------------------|--|
| data | | 0.1272 | 1.1267 |
| classic KR | 0 | 0.3068 | 0.3684 |
| bilateral KR | | 0.0083 | 1.0082 |
| robust KR | | 0.0532 | 0.0534 |
| classic KR | 1 | 0.3070 | 0.3606 |
| bilateral KR | | 0.0088 | 1.0087 |
| robust KR | | 0.2722 | 0.2727 |
| classic KR | 2 | 0.2101 | 0.3136 |
| bilateral KR | | 0.0172 | 1.0171 |
| robust KR | | 0.1502 | 0.1504 |

Table 1: Step function: Summary of the mean squared errors (MSE)

4.2 Image Denoising - Performance Comparison on Different Types of Noise

Figures 3 and 4 summarize the results of the comparison of the performance of classic, bilateral and robust KR in case of Gaussian noise. Bilateral KR seems to give the best results. Robust KR gives slightly better results on edges than the classic linear KR.

If salt and pepper noise is added to the image (see Figures 5 and 6), robust KR clearly outperforms the bilateral and the classic linear KR. The bilateral KR gives the worst result.

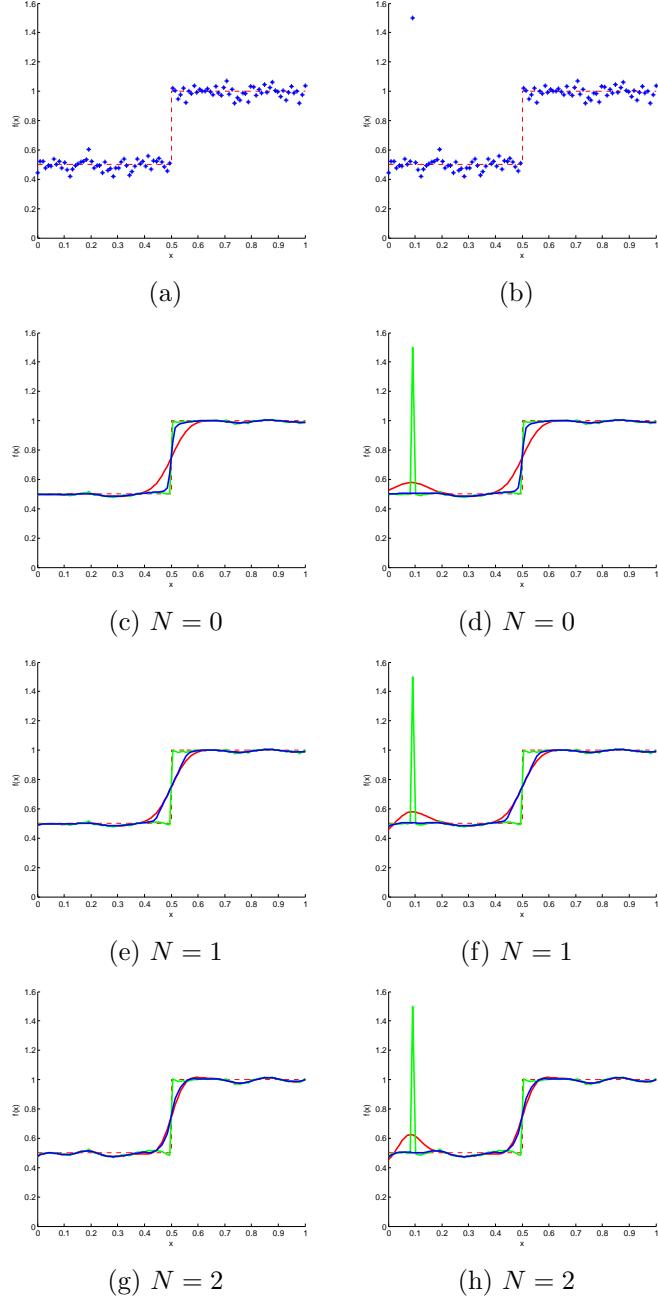


Figure 2: Step function: Classic KR in red ($N \in \{0, 1, 2\}, h_s = 0.05$), bilateral KR in green ($N \in \{0, 1, 2\}, h_s = 0.05, h_r = 0.1$) and robust KR in blue ($N \in \{0, 1, 2\}, h_s = 0.05, h_r = 0.1$, number of iterations = 3).

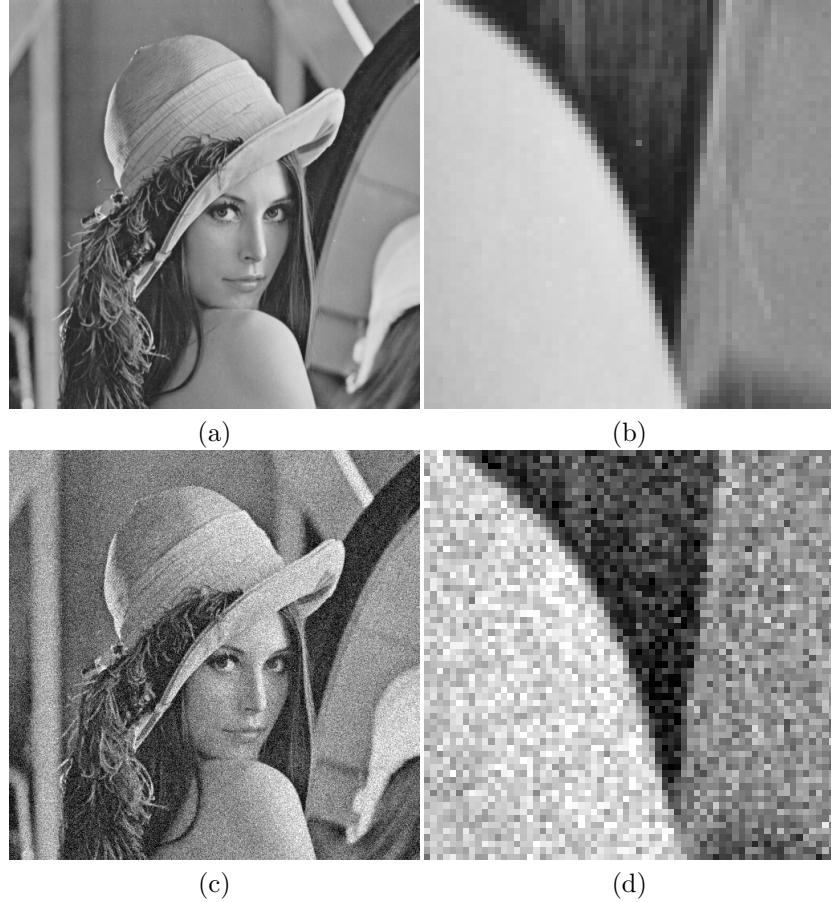


Figure 3: Denoising (Gaussian Noise): (a) and (b): Original 512×512 image with pixel intensities in $[0, 1.0]$, (c) and (d): Original image with additive noise (mean: 0.0, variance: 0.01). Noise: 0.0992 (RMSE) and 20.0701 dB (SNR).

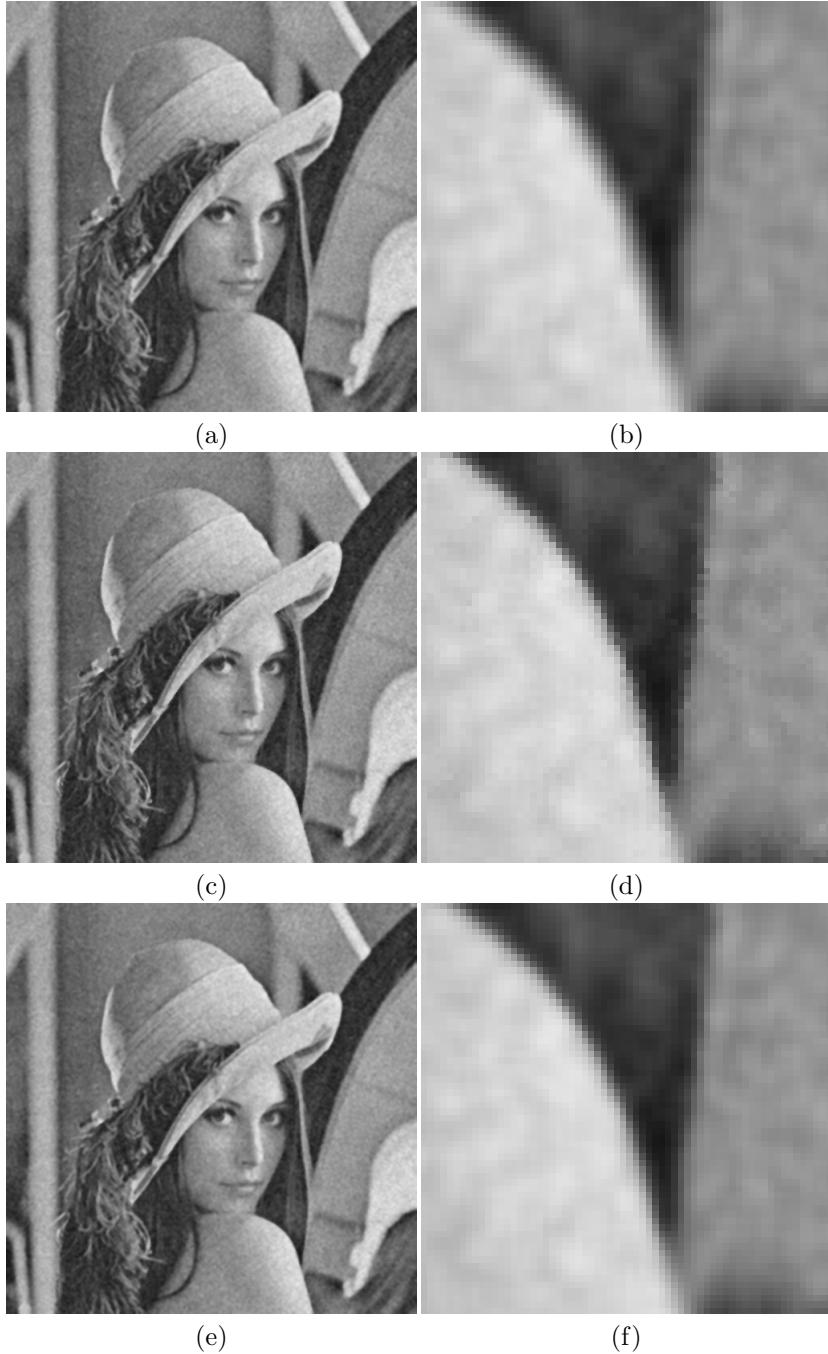


Figure 4: Denoising (Gaussian Noise): (a) and (b): Linear KR ($N = 0$, window size = 5, $h_s = 2.0$). Error: 0.0364 (RMSE) and 28.779 dB (SNR), (c) and (d): Bilateral KR ($N = 0$, window size = 5, $h_s = 2.0$, $h_d = 0.4$). Error: 0.03404 (RMSE) and 29.3603 dB (SNR), (e) and (f): Robust KR ($N = 0$, window size = 5, $h_s = 2.0$, $h_r = 0.4$, number of iterations = 3). Error: 0.03516 (RMSE) and 29.079 dB (SNR).

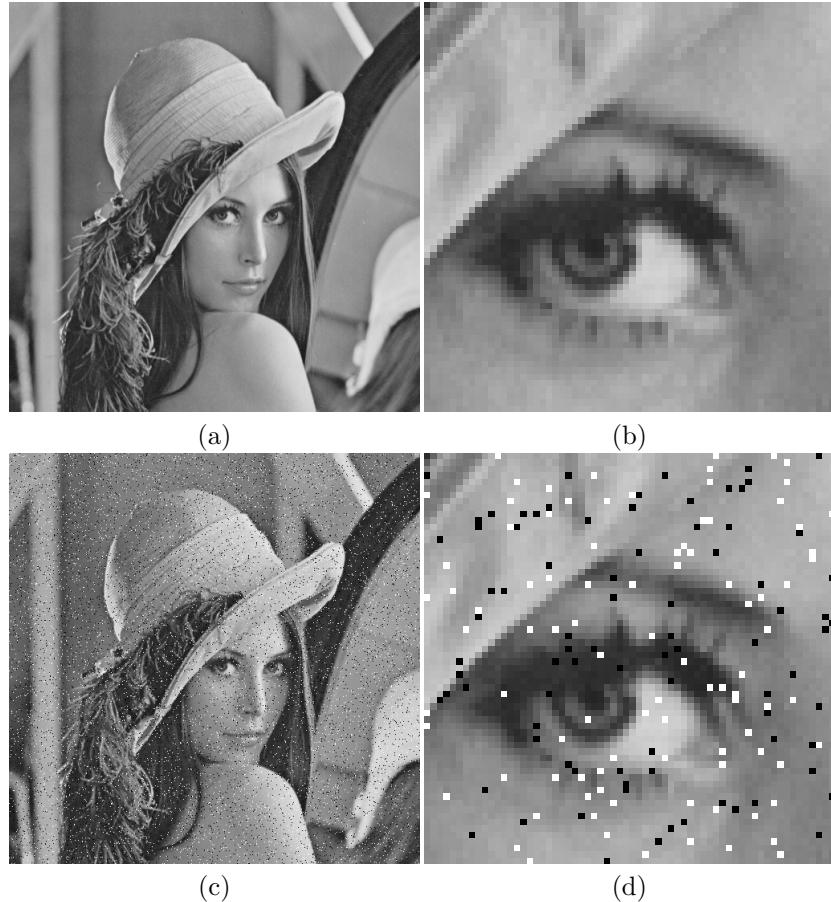


Figure 5: Denoising (Salt and Pepper): (a) and (b): Original 512×512 image with pixel intensities in $[0, 1.0]$, (c) and (d): Original image with salt and pepper noise (approximately 5 % of the pixels set to black or white). Noise: 0.11958 (RMSE) and 18.4465 dB (SNR).

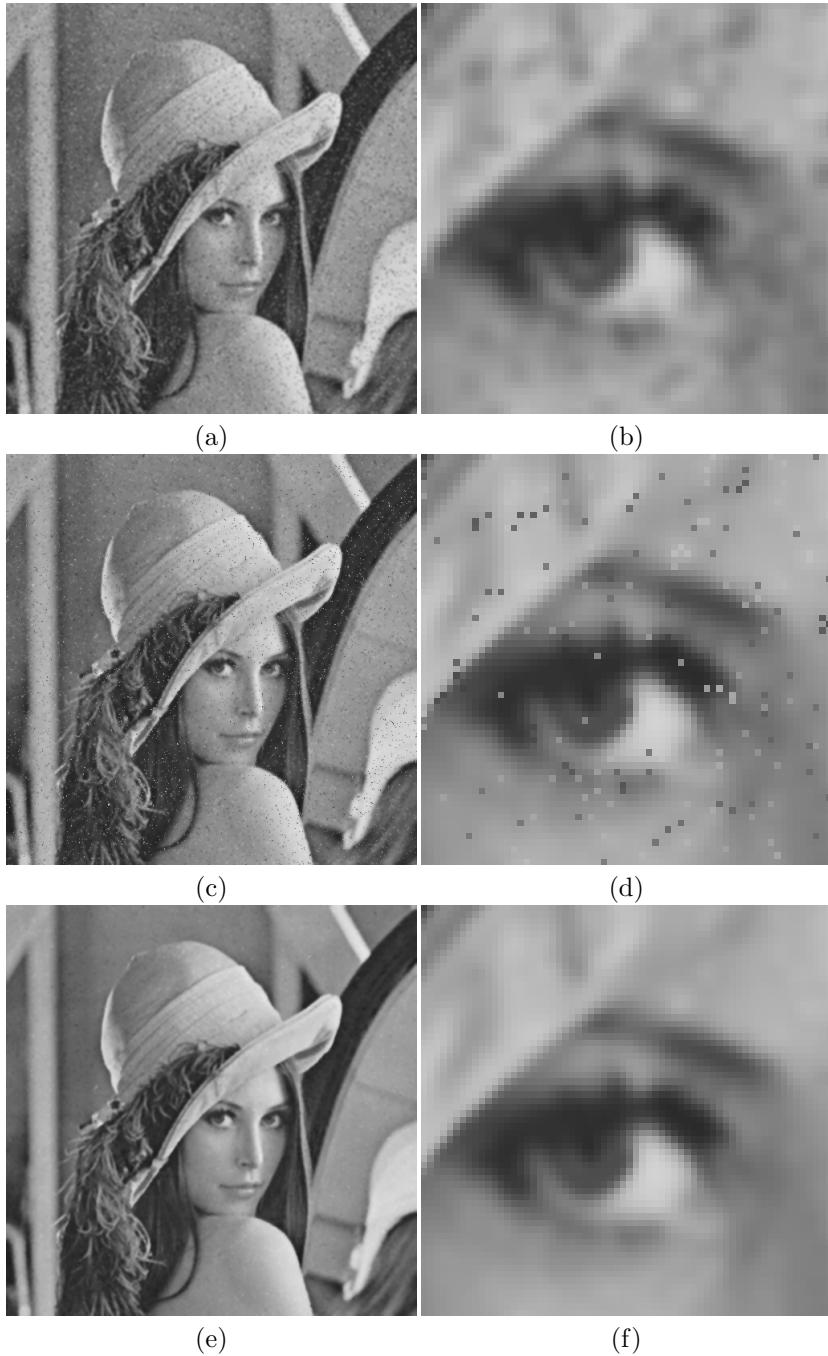


Figure 6: Denoising (Salt and Pepper): (a) and (b): Linear KR ($N = 0$, window size = 5, $h_s = 1.5$). Error: 0.04 (RMSE) and 27.9588 dB (SNR), (c) and (d): Bilateral KR ($N = 0$, window size = 5, $h_s = 1.5$, $h_d = 0.4$). Error: 0.048661 (RMSE) and 26.2564 dB (SNR), (e) and (f): Robust KR ($N = 0$, window size = 5, $h_s = 1.5$, $h_r = 0.4$, number of iterations = 3). Error: 0.026678 (RMSE) and 31.4769 dB (SNR).

4.3 Image Reconstruction - From Irregular Samples to Full Pictures

The KR framework can also be used to reconstruct images. If we remove 85 % of the pixels (randomly) and apply classic, linear and robust KR we get the results summarized in Figures 7 and 8. All the KR variants do a decent job in uniform areas. The bilateral KR seems to reconstruct edges better than the other two. The edges, in case of the robust KR, are slightly sharper than the ones we get with the classic linear KR.

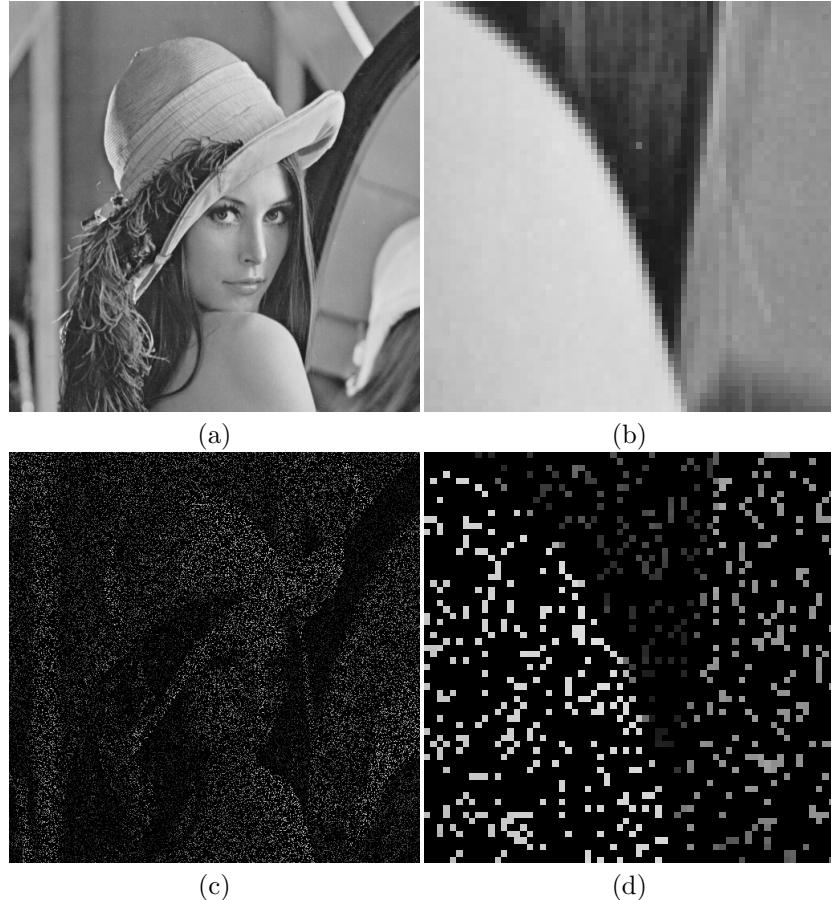


Figure 7: Reconstruction: (a) and (b): Original 512×512 image with pixel intensities in $[0, 1.0]$, (c) and (d): Original image with 85 % of the pixels removed.



Figure 8: Reconstruction: (a) and (b): Linear KR ($N = 0$, window size = 11, $h_s = 2.0$). RMSE (comparison with ground truth): 0.043029 , (c) and (d): Bilateral KR ($N = 0$, window size = 11, $h_s = 2.0$, $h_d = 0.4$). RMSE (comparison with ground truth): 0.037698, (e) and (f): Robust KR ($N = 0$, window size = 11, $h_s = 2.0$, $h_r = \frac{1}{16}$, number of iterations = 5). RMSE (comparison with ground truth): 0.041621.

4.4 Image Up-Sampling - Enhance the Spatial Resolution of an Image

Another application of the KR framework is image upsampling. A superresolution example is summarized in Figures 9 and 10. An image is first down-sampled by a factor 4 and than up-sampled again using the different KR variants. The bilateral KR gives the best results. The results we get for the classic and the robust KR look the same and their RMSEs are almost the same.

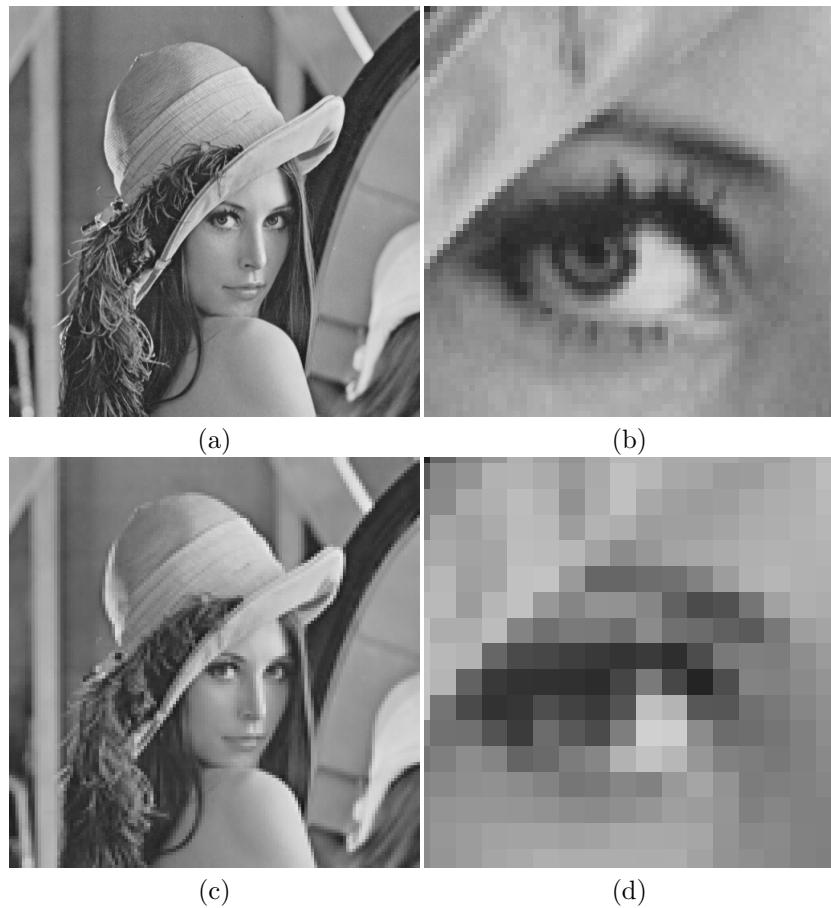


Figure 9: Upsampling: (a) and (b): Original 512×512 image with pixel intensities in $[0, 1.0]$, (c) and (d): Down-sampled image (factor 4).

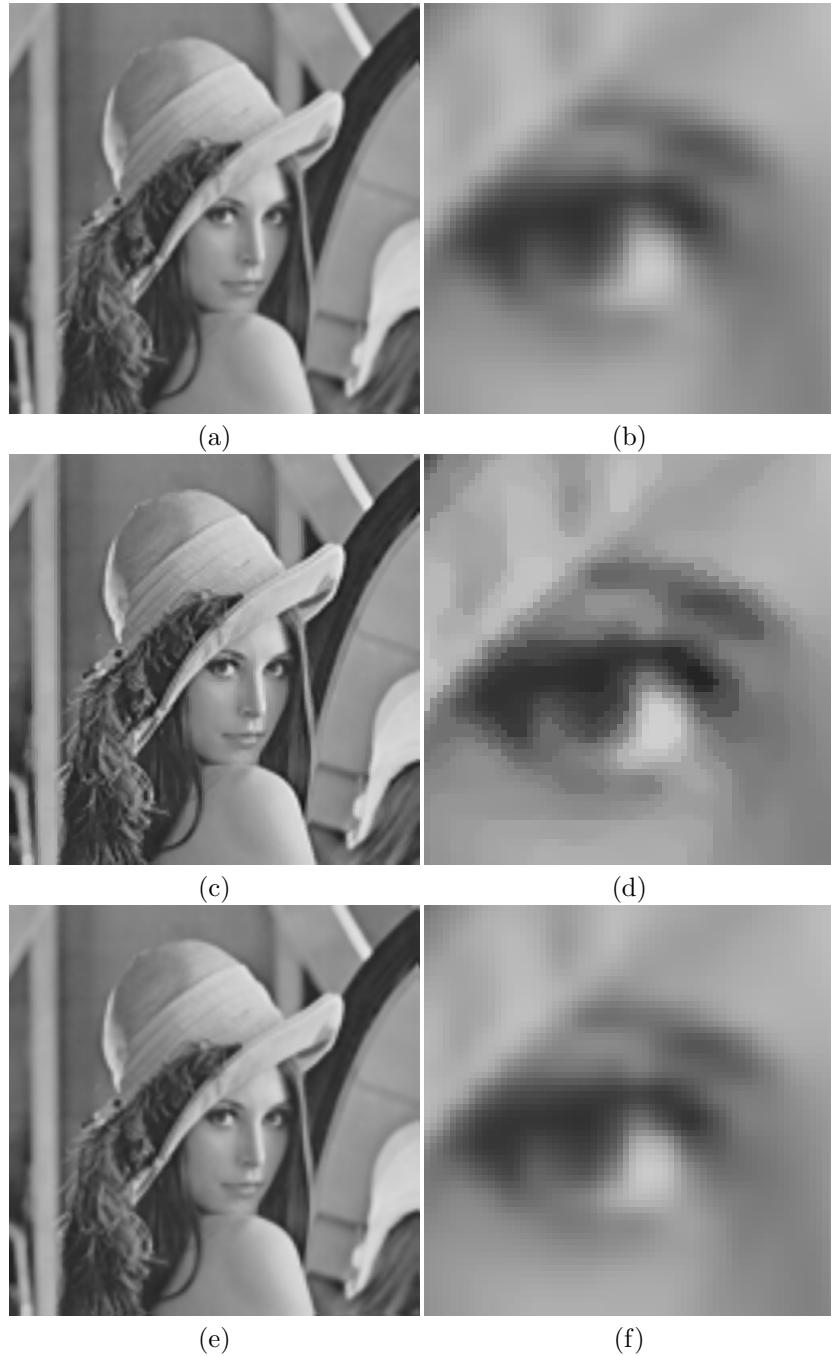


Figure 10: Upsampling: (a) and (b): Linear KR ($N = 0$, window size = 11, $h_s = 0.5$). RMSE (comparison with ground truth): 0.05663 , (c) and (d): Bilateral KR ($N = 0$, window size = 11, $h_s = 0.5$, $h_d = 0.01$). RMSE (comparison with ground truth): 0.037072, (e) and (f): Robust KR ($N = 0$, window size = 11, $h_s = 0.5$, $h_r = 0.6$, number of iterations_{Pg}= 3). RMSE (comparison with ground truth): 0.056606.

4.5 Higher Order Bilateral Filters - A More Colorful Example

The KR framework also allows higher order filtering. An example of higher order bilateral filters applied to a color example is summarized in Figures 11 and 12. The image is first converted to the YCbCr space and the bilateral KR with the same parameters is applied to each channel independently. If we compare the overall results visually, than the oder zeros bilateral KR seems to give a better result. However, if we zoom in the result we get with the second order bilateral filter looks smoother than the one we get with the zero order filter. The RMSE and the SNR for the second order filter result are also slightly better than the ones we get for the zero order-filtered image.

4.6 Derivatives - A By-Product

The framework also allows to directly compute partial derivatives of an image. In case of a degree 1 approximation ($N = 1$), we can optionally compute the first order derivatives. The degree 2 approximation ($N = 2$) allows us to directly compute first and second order partial derivatives as can be seen in Figures 13 and 14.

5 Conclusion

We saw a framework based on kernel regression that allows image denoising, upsampling and reconstruction. Bilateral KR seems to give the best results if no outliers (salt and pepper noise) is involved. In case of outliers, robust non-linear KR seems to clearly outperform bilateral KR. This investigation suggests that a combination of a bilateral kernel and a robust estimator is used.

References

- [1] N. K. Bose and Nilesh A. Ahuja. Superresolution and noise filtering using moving least squares. In *IEEE Transactions on Image Processing 2006*. IEEE, 2006.
- [2] C. K. Chu, I. K. Glad, F. Godtliebsen, and J. S. Marron. Edge-preserving smoothers for image processing. In *Journal of the American Statistical Association*. American Statistical Association, 1998.
- [3] Raanan Fattal. Image upsampling via imposed edge statistics. In *Proceedings of SIGGRAPH 2007*. ACM SIGGRAPH, 2009.



Figure 11: Denoising (Gaussian Noise) on Color Image: (a) and (b): Original 512×512 image with pixel intensities in $[0, 1.0]$, (c) and (d): Original image with additive noise (mean: 0.0, variance: 0.01). Noise: 0.097639 (RMSE) and 20.2076 dB (SNR).



Figure 12: Denoising (Gaussian Noise) on Color Image: (a) and (b): Bilateral KR ($N = 0$, window size = 11, $h_s = 100.0$, $h_d = 0.1$). Error: 0.039308 (RMSE) and 28.1104 dB (SNR), (c) and (d): Bilateral KR ($N = 2$, window size = 11, $h_s = 100.0$, $h_d = 0.1$). Error: 0.038411 (RMSE) and 28.311 dB (SNR).

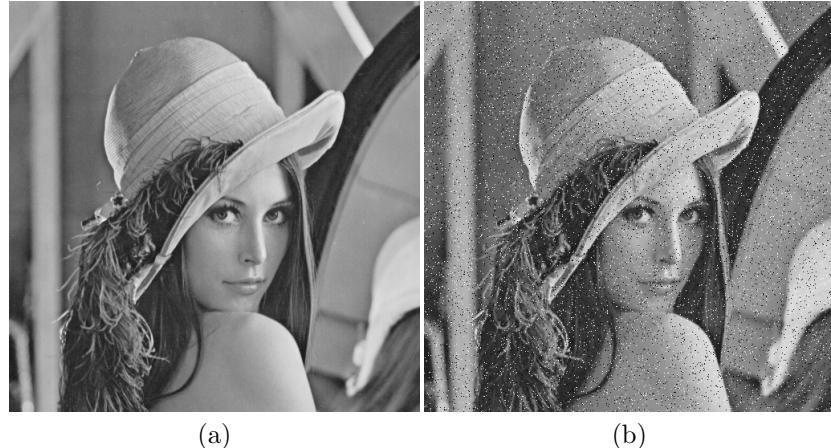


Figure 13: Derivatives (Salt and Pepper): (a) and (b): Original 512×512 image with pixel intensities in $[0, 1.0]$, (c) and (d): Original image with salt and pepper noise (approximately 5 % of the pixels set to black or white).

- [4] M. Fenn and G. Steidl. Robust local approximation of scattered data. In *Geometric Properties from Incomplete Data, Computational Imaging and Vision*. Springer, 2005.
- [5] A. C. Oeztireli, G. Guennebaud, and M. Gross. Feature preserving point set surfaces based on non-linear kernel regression. In *Proceedings of EUROGRAPHICS 2009*. Eurographics, 2009.
- [6] D. Ruppert and M. P. Wand. Multivariate locally weighted least squares regression. In *The Annals of Statistics*. Institute of Mathematical Statistics, 1994.
- [7] H. Takeda, S. Farsiu, and P. Milanfar. Higher order bilateral filters and their properties. In *Proceedings of the SPIE 2007*. SPIE, 2007.
- [8] H. Takeda, S. Farsiu, and P. Milanfar. Kernel regression for image processing and reconstruction. In *IEEE Transactions on Image Processing 2006*. IEEE, 2007.

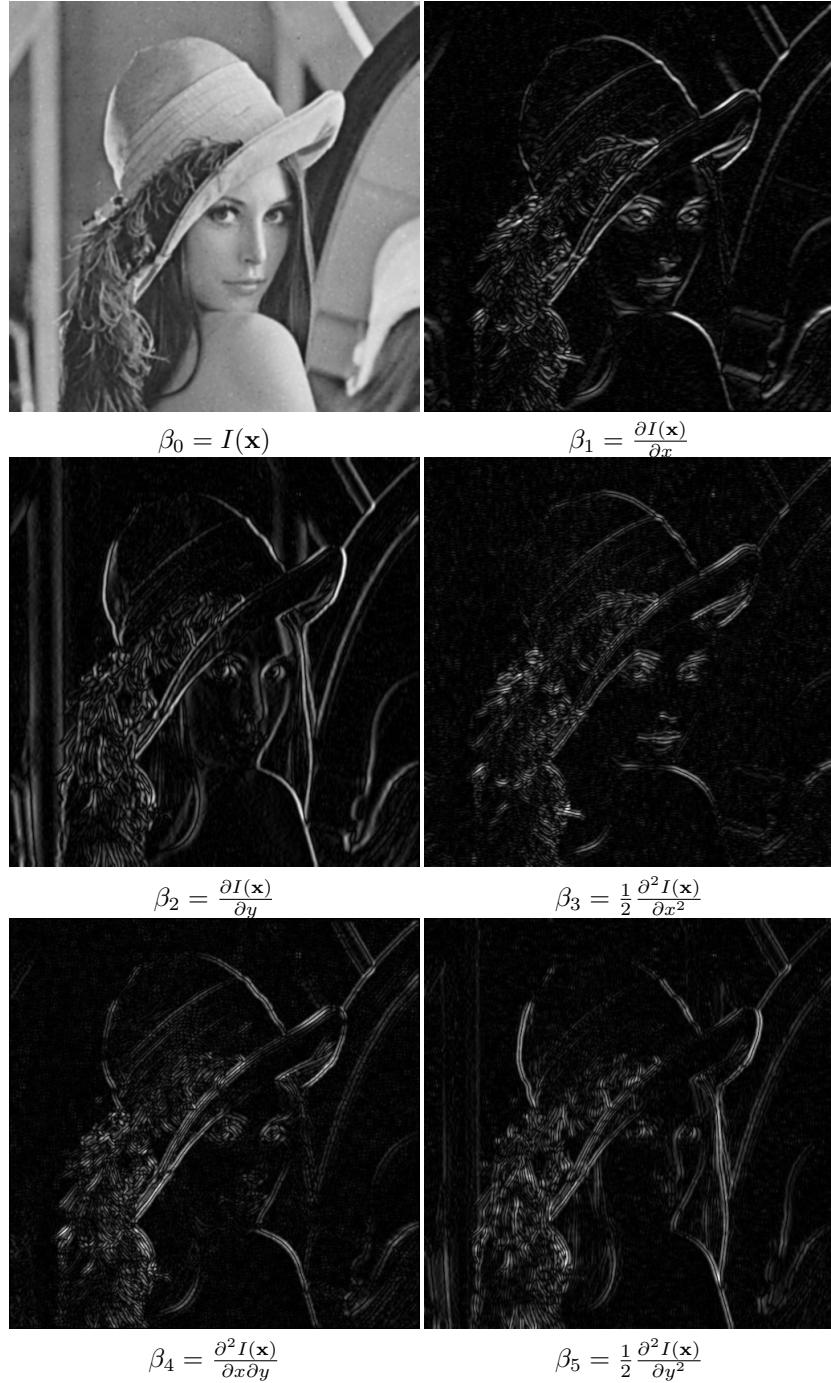


Figure 14: Derivatives: Robust KR with $N = 2$. We get partial derivatives of an image as an optional by-product.