

3D Image Restoration with the Curvelet Transform

A. Woiselle^{1,2,3}, J-L. Starck^{1,2}, J. Fadili⁴

(1) CEA, IRFU, SEDI-Service d'Astrophysique, F-91191 GIF-Sur-YVETTE, France.

(2) Laboratoire Astrophysique des Interactions Multi-échelles (UMR 7158),

CEA/DSM-CNRS-Universite Paris Diderot, F-91191 GIF-Sur-YVETTE, France.

(3) SAGEM Defense Securite, 95101 Argenteuil CEDEX, France.

(4) GREYC CNRS UMR 6072, Image Processing Group, ENSICAEN 14050 Caen Cedex, France.

June 27, 2008

Abstract

We present two new 3D curvelet transforms which are built as extensions of the 2D first generation curvelet transform. The first one, called the RidCurvelet, is especially well designed for representing 2D surfaces in a 3D space while the second one, the BeamCurvelet, is better adapted to represent 1D filaments in 3D space. We show that these 3D curvelet transforms can be built using existing 3D transforms, the 3D wavelet transform, the 3D ridgelet transform, and the 3D beamlet transform. We illustrate the applicability of these transforms on various examples such as the detection of linear structures and planar surfaces, as well as on denoising and inpainting.

1 Introduction

Data analysis has become a very important field of research in the past decades, one recent point of view being the multi-scale analysis. The data used to be 2D arrays but more and more, with the improving capabilities of computers, we tend to analyze 3D volumes as one block and not as slices.

Wavelets [13] have been used to study 2D or 3D isotropic elements, but they fail to represent the curvilinear edges in images. Thus the first generation curvelet transform has been built to analyse curves in two dimensional data. These curvelets are based on the wavelet transform and the ridgelet transform, the latter aimed at representing lines in an image. A wide range of applications such as denoising [14, 11], contrast enhancement [15], inpainting [8, 10], deconvolution [17, 9, 16] and source separation [1, 2] already use these transforms.

To tackle restoration problems in three dimensions, we extend these multi-scale geometric transforms, which leads us to two 3D Curvelet transforms. Section 2 describes the two transforms, section 3 shows the capability of each transform to detect different types of structures, and finally section 4 shows applications to denoising and inpainting.

2 The structure of the transforms

2.1 Two-dimensional background

2.1.1 The 2D Ridgelet transform

The two-dimensional ridgelet function with parameters $(s, k, \theta) \in \mathbb{R}_+^* \times \mathbb{R} \times [0, 2\pi[$ was defined by E. Candès and D. Donoho in [6, 3] as

$$\psi_{s,k,\theta}(x, y) = s^{-1/2} \psi((x \cos \theta + y \sin \theta - k)/s),$$

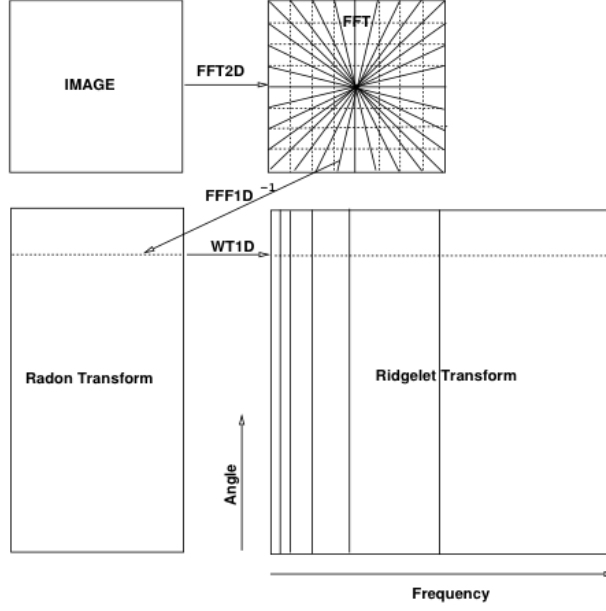


Figure 1: *A schematic view of the steps for a 2D ridgelet transform in the Fourier domain : Take the 2D FFT, extract lines passing through the origin, apply to each an inverse 1D FFT and a 1D Wavelet transform.*

with $\psi \in L^2(\mathbb{R}^2)$ a smooth function satisfying the admissibility condition (often a wavelet)

$$\int |\hat{\psi}(\xi)|^2 / \xi d\xi < +\infty.$$

We can notice that a ridgelet is a ϕ (wavelet) function along the direction θ , and is constant over the perpendicular direction.

The coefficients of a transformed image are obtained by projecting the image on these functions, which gives us

$$\mathcal{R}f(s, k, \theta) = \int \psi(x, y) f(x, y) dx dy.$$

On a numerical point of view, the ridgelet transform is done in Fourier domain, which means extracting lines passing through the center at different angles, and applying a 1D wavelet transform on each. Figure 1 shows the process in Fourier domain.

The ridgelet functions obtained are well adapted to represent lines that run along the entire size of an image. To represent a curve, which is limited in space, the idea was to apply the ridgelet transform on a time-space partition of the image, with an appropriate scaling, the parabolic scaling law.

2.1.2 The 2D Curvelet transform

The 2D first generation curvelet is then defined as follows (see [5] and [7]): The image is partitioned in spectral bands using a filter-bank $\forall s \in \mathbb{N}, \Psi_{2^s} = 2^{4s} \Psi(2^{2s} \cdot)$ which extracts the frequencies $|\xi| \in [2^{2s}, 2^{2s+2}]$, and a low-pass scaling wavelet ψ_0 for $|\xi| \leq 1$.

Let $P_0 f = \psi_0 * f$ and $\Delta_s f = \Psi_{2^s} * f$. Then, we tile the direct space with subsets Q of size 2^s .

$$Q = Q(s, k_1, k_2) = \left[\frac{k_1}{2^s}, \frac{k_1 + 1}{2^s} \right] \times \left[\frac{k_2}{2^s}, \frac{k_2 + 1}{2^s} \right]$$

with smooth windows w_Q localised near Q , such that $\sum_{Q \in \mathcal{Q}_s} w_Q^2 = 1$.
We use a normalisation over the space tiles using a transport operator T_Q :

$$T_Q f(x_1, x_2) = 2^s f(2^s x_1 - k_1, 2^s x_2 - k_2)$$

and we run a ridgelet transform on each tile to obtain the curvelet coefficients.
Thus the curvelet transform of an image f is

$$\mathcal{C}f = \{\mathcal{R}((T_Q)^{-1} w_Q \Delta_s f) : s \in \mathbb{N}, Q \in \mathcal{Q}_s\}.$$

The algorithm is :

1. Apply a 2D wavelet transform with J scales from the finest to coarsest one,
2. Set an initial block size B_1 for the finest scale,
3. **for** $j = 1$ **to** $J - 1$ **do**
4. Partition the scale j with blocks of size B_j ,
5. **if** j is even **then** $B_{j+1} = B_j$ **else** $B_{j+1} = 2B_j$
6. **end for**

2.2 The RidCurvelet transform

The first extension of the curvelet transform in 3D is accomplished by using the 3D ridgelet transform. A three-dimensional ridge function is given by :

$$\begin{aligned} \psi_{s,k,\theta_1,\theta_2}(x_1, x_2, x_3) &= s^{-1/2} \psi((x_1 \cos \theta_1 \cos \theta_2 + x_2 \sin \theta_1 \cos \theta_2 + x_3 \sin \theta_2 - k)/s), \\ (s, k, \theta_1, \theta_2) &\in \mathbb{R}_+^* \times \mathbb{R} \times [0, 2\pi) \times [0, \pi). \end{aligned}$$

Therefore, the ridgelet transform of a function $f \in L^2(\mathbb{R}^3)$ is

$$\mathcal{R}f(s, k, \theta_1, \theta_2) = \int \psi_{s,k,\theta_1,\theta_2}(x_1, x_2, x_3) f(x_1, x_2, x_3) dx_1 dx_2 dx_3.$$

The transform consists in summing the cube over planes at every direction and position. For a fixed direction (θ_1, θ_2) , the summation gives us a line. Each point on this line represents a plane in the original cube. We make a mono-dimensional wavelet transform on each 3D line to obtain the ridgelet transform.

The 3D ridge function aims at representing planes in a 3D space. It is constant over a plane and oscillates like ψ in the normal direction.

As the 2D version, the 3D RidCurvelet transform is implemented in Fourier space : the sum over the planes becomes a line extraction in Fourier. The main steps are summarized on figure 2.

2.3 The BeamCurvelet transform

The other extension of the curvelet in 3D is done by using the 3D Beamlet Transform instead of the ridgelets. A three-dimensional beam function is given by :

$$\begin{aligned} \psi_{s,k_1,k_2,\theta_1,\theta_2}(x_1, x_2, x_3) &= s^{-1/2} \psi((-x_1 \sin \theta_1 + x_2 \cos \theta_1 + k_1)/s, \\ &\quad (x_1 \cos \theta_1 \cos \theta_2 + x_2 \sin \theta_1 \cos \theta_2 - x_3 \sin \theta_2 + k_2)/s). \end{aligned}$$

Compared to 3D ridgelets, which sum over planes, the beamlet sums over the lines (θ_1, θ_2) , which gives us a plane for each direction. Therefore, the beamlet transform of a function $f \in L^2(\mathbb{R}^3)$ is

$$\mathcal{B}f(s, k, \theta_1, \theta_2) = \int \psi_{s,k,\theta_1,\theta_2}(x_1, x_2, x_3) f(x_1, x_2, x_3) dx_1 dx_2 dx_3.$$

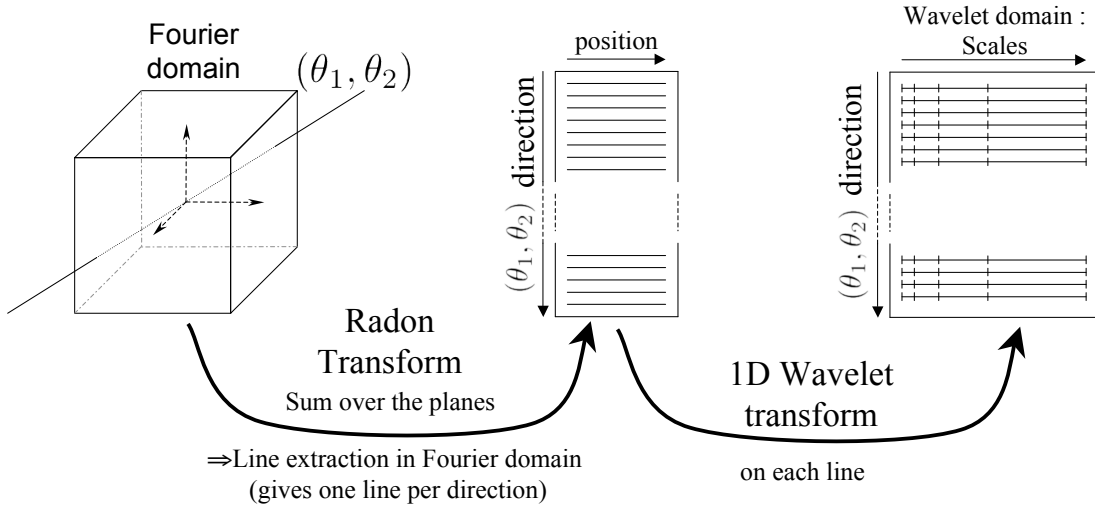


Figure 2: *A schematic view of the steps of a 3D ridgelet transform in the Fourier domain : Take the 3D FFT, extract lines passing through the origin at every direction (θ_1, θ_2) , apply to each obtained line an inverse 1D FFT and a 1D Wavelet transform (for speed, the wavelet transform can be performed directly in Fourier domain).*

The transform consists in summing the cube over lines at every direction and position. For a fixed direction (θ_1, θ_2) , the summation gives us a plane. Each point on this plane represents a line in the original cube. We then apply a two dimensional wavelet transform on each plane to obtain the beamlet transform.

The 3D beam function aims at representing filaments in a 3D space. It is constant over a line and oscillates like ψ in the radial direction.

As the RidCurvelet, the BeamCurvelet transform is implemented in Fourier space : the sum over the lines becomes a plane extraction in Fourier. The main steps are depicted on figure 3.

3 Detection characterisation

In order to validate the efficiency of the transforms to adapt itself to planes and filaments, we have plotted detection level curves using Monte-Carlo simulations. Toward this, we have simulated two cubes containing either a plane or a filament with an unitary amplitude, and added a Gaussian noise with different powers, from $3 \cdot 10^{-2}$ to 9. The two novel transforms (RidCurvelet and BeamCurvelet) and the 3D Wavelet transform have been applied to each cube. Fifty Monte-Carlo realizations of noise have been generated for each noise level and data. Figure 4 shows the evolution of the largest normalized coefficient (i.e. the atoms are normalized to a unit 2 norm), as a fraction of the noise level, for each transform and each data. Error bars have been also plotted.

For the cube containing the plane, the transform getting the strongest detection level is the RidCurvelet, and for the filament, the best one is the BeamCurvelet, which corresponds to our expectations. We can notice that the detection level of the line by the BeamCurvelet is lower than that for the plane, which could be surprising, but it is due to the amount of information (number of voxels) contained in a plane, that is much bigger than the one in a line.

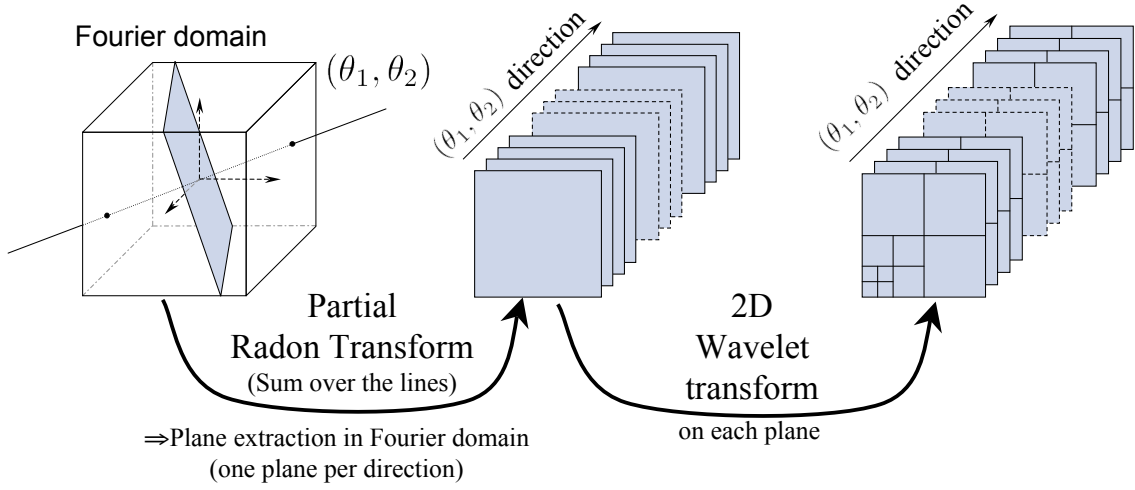


Figure 3: *A schematic view of the steps of a 3D beamlet transform : Take the 3D FFT, extract planes passing through the origin at every direction (θ_1, θ_2) , apply to each obtained line an inverse 2D FFT and a 2D Wavelet transform (for speed, the wavelet transform can be performed directly in Fourier domain).*

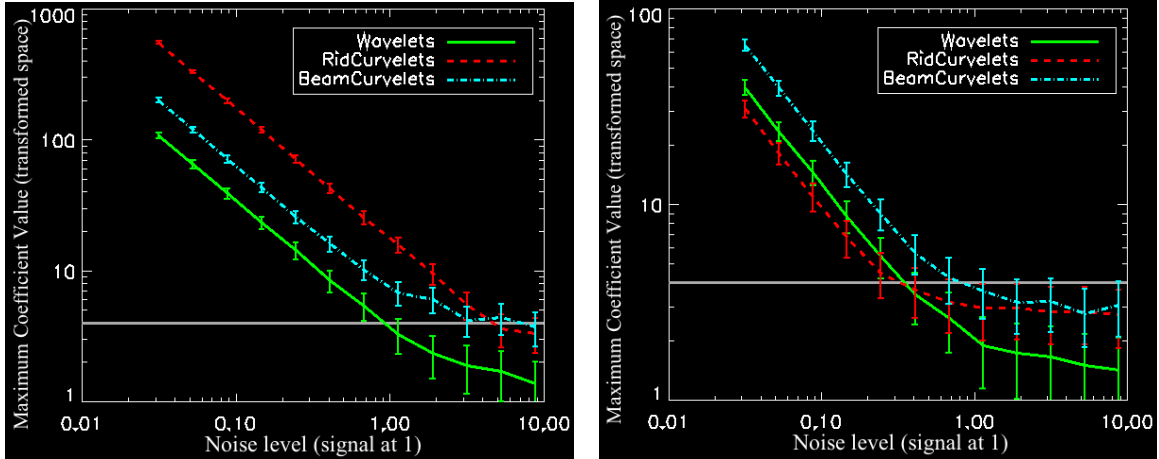


Figure 4: *Left part (resp. right) : Given an image containing a plane (resp. a line), the curves represent the values of the greatest coefficient of the transformed image over the noise level. The grey horizontal line represents the noise maximum values (i.e. four times the noise standard deviation). The RidCurvelets (resp. BeamCurvelets) detect planes (resp. lines) with a better SNR than the two other transforms.*

4 Applications

4.1 Denoising

In order to apply the results shown by the detection curves (see previous section and figure 4), we tried denoising a simulated data-cube containing one filament and one plane. The results are shown on figure 6. Let C be the original data-cube, C^n the cube with additive Gaussian noise, and C_f^n the filtered cube with our denoising methods. Table 1 shows the denoising performed with both transforms for two levels of noise. The Mean Integrated Squared Error is calculated for the whole cube, and also restricted to the left or right part to separate the filament and the plane.

Figures 5 and 6 show respectively the central slices of the results of the denoising, and a density representation in the case of high noise. We can see that the plane is well recovered by the two transforms, but the line has almost disappeared with the RidCurvelet when the noise is high. With the BeamCurvelet, the line is recovered, but not perfectly, because of the low energy of 1D elements. The plane is recovered too, which is consistent with the detection level plots.

	Noise Level σ	$PSNR(dB)$	$\frac{\text{var}(C_f^n - C)}{\text{var}(C)}$	$\frac{\text{var}(C_f^n - C)}{\text{var}(C)} _{plane}$	$\frac{\text{var}(C_f^n - C)}{\text{var}(C)} _{filament}$
RidCurvelet	0.1	84.3	0.037	0.028	0.185
BeamCurvelet	0.1	83.2	0.051	0.040	0.198
RidCurvelet	0.4	75.8	0.281	0.222	1.009
BeamCurvelet	0.4	76.2	0.281	0.225	0.683

Table 1: Power of the reconstruction error $(C_f^n - C)$ over the original image (C) on the entire image or restricted to objects.

4.2 Inpainting

Inpainting is the process of recovering missing parts in altered data. Let x be our three-dimensional data cube with missing data indicated by the mask M . The available data is $y = Mx$. Given a dictionary Φ , we are trying to recover x from the observed y and the mask M . This is an ill posed idealized problem. To get a consistent solution, one must seek regularized solutions. One such regularization is to suppose that x is sparse in one dictionary of atoms Φ , which means that x can be represented by a few atoms from Φ .

Therefore, we want to solve the following inpainting problem :

$$\arg \min_x \|\Phi^T x\|_0 \text{ s.t. } y = Mx$$

The algorithm applied is [8] :

Initialisation : $x^{(0)} = y$

Repeat :

$$\begin{cases} \lambda^{(n)} &= k^{(n)} * MAD(\Phi^T x^{(n)}) \\ x^{(n+1)} &= \Phi \mathcal{HT}_{\lambda^{(n)}} \left(\Phi^T \left[x^{(n)} + M(y - Mx^{(n)}) \right] \right) \\ &= \Phi \mathcal{HT}_{\lambda^{(n)}} \left(\Phi^T \left[(I - M)x^{(n)} + y \right] \right) \end{cases}$$

Where $\mathcal{HT}_{\lambda^{(n)}}$ is the hard-thresholding operator with threshold $\lambda^{(n)}$, and $k^{(n)}$ decreases linearly with k and stops when it reaches 0.. MAD stands for *Median Absolute Deviation*.

Figure 7 shows the inpainting of the $\Lambda - CDM$ simulation based on the RAMSES Code [18], on which we applied a random mask of 20% (respectively 80%) missing voxels. The L^2 and L^1 reconstruction error compared to the power of the original cube is 1.3% and 0.1% (resp. 25.7% and 0.77%). The error is lower from far with the L^1 norm, because when there is

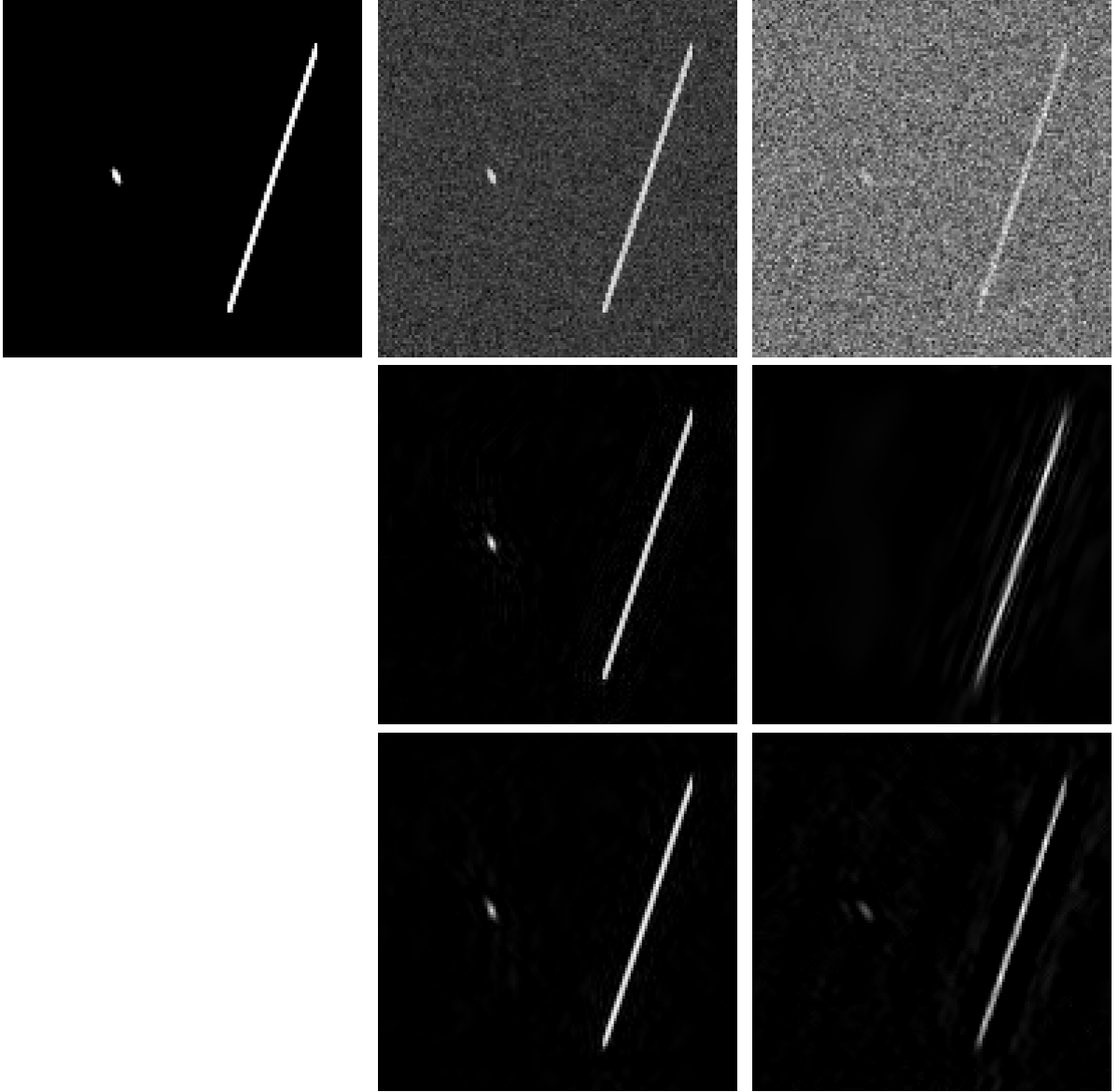


Figure 5: *Central slices of the noisy and denoised cubes. First row : original image and noisy versions. Second row : the images denoised with the RidCurvelet transform. Third row : the images denoised with the BeamCurvelet transform.*

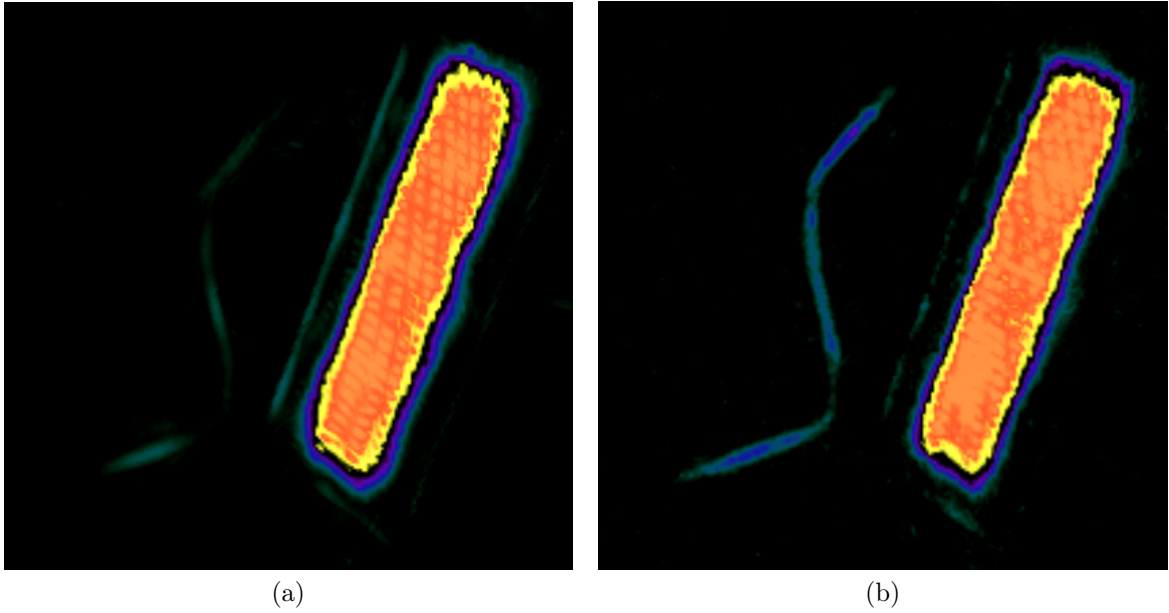


Figure 6: *Denoising results of a synthetic cube containing a plane and a filament. (a) denoised with the RidCurvelet, (b) with the BeamCurvelet.*

much of the data missing, like eighty percent, we lose the hot spots, and the inpainting being a method building smooth reconstructions, we have important localized errors, even though it looks very similar to the original.

5 Conclusion

In this paper, we described two 3D curvelet transforms. These two transforms are well adapted to represent filaments and surfaces in 3D space. The RidCurvelet a few fast equivalents, of which we can cite the second generation fast 3D curvelet transform [4] and the surfacelets [12] by Lu and Do, but there is currently no transform comparable to the BeamCurvelet, which may be the seed of further research. Finding a fast equivalent is very interesting, since the as-it transform has a computationally cost relatively high ($\mathcal{O}(n^4)$ for a n^3 samples data cube). Yet they are efficient in their domain and can be used in any inverse problem application, like deconvolution, super-resolution or source separation.

References

- [1] J. Bobin, Y. Moudden, J.-L. Starck, and M. Elad. Morphological diversity and source separation. *IEEE Signal Processing Letters*, 13(7):409–412, 2006.
- [2] J. Bobin, J.L. Starck, J. Fadili, and Y. Moudden. Sparsity and morphological diversity in blind source separation. *IEEE Transactions on Image Processing*, 16(11):2662–2674, November 2007.
- [3] E.J. Candès. *Ridgelets, theory and applications*. PhD thesis, Stanford University, 1998.
- [4] E.J. Candès, L. Demanet, D.L. Donoho, and L. Ying. Fast discrete curvelet transforms. *SIAM Multiscale Model. Simul.*, 5(3):861–899, 2006.

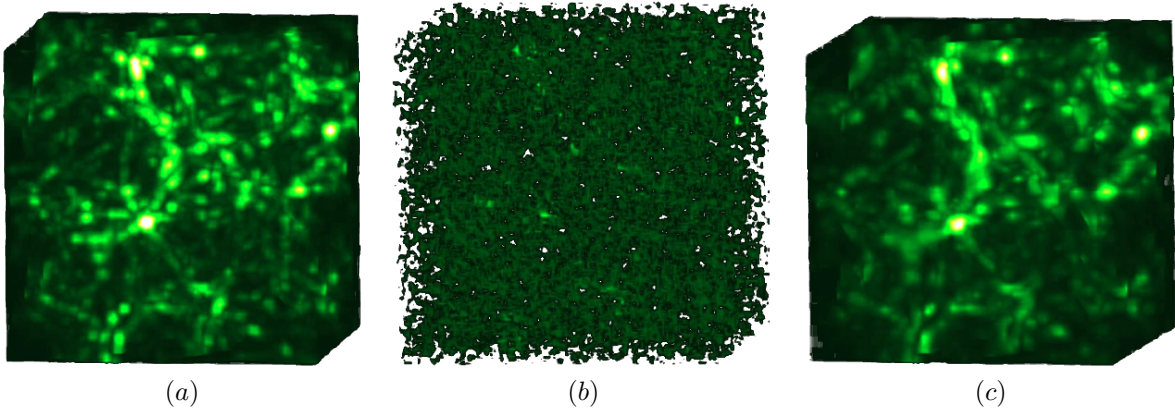


Figure 7: *Inpainting of a cube with 80% randomly missing voxels. (a) original, (b) masked and (c) inpainted.*

- [5] E.J. Candès and D.L. Donoho. Curvelets. *Stanford University, Dept. of Statistics, Technical Report*, 1999.
- [6] E.J. Candès and D.L. Donoho. Ridgelets: the key to high dimensional intermittency. *Philosophical Transactions of the Royal Society of London A*, 357:2495–2509, 1999.
- [7] D.L. Donoho and M.R. Duncan. Digital curvelet transform: strategy, implementation and experiments. 1999.
- [8] M. Elad, J.-L. Starck, P. Querre, and D.L. Donoho. Simultaneous cartoon and texture image inpainting using morphological component analysis. *Computational Harmonic Analysis*, 19:340–358, 2005.
- [9] M. J. Fadili and J.-L. Starck. Sparse representation-based image deconvolution by iterative thresholding. In F. Murtagh and J.-L. Starck editors, editors, *Astronomical Data Analysis IV, Marseille, France*, 2006.
- [10] M. J. Fadili, J.-L. Starck, and F. Murtagh. Inpainting and zooming using sparse representations. *The Computer Journal*, 2006.
- [11] G. Hennenfent and F.J. Herrmann. Seismic denoising with nonuniformly sampled curvelets. *IEEE Computing in Science and Engineering*, 8(3):16–25, May 2006.
- [12] Y. Lu and M. N. Do. 3-d directional filter banks and surfacelets. In *Proc. of SPIE Conference on Wavelet Applications in Signal and Image Processing XI, San Diego, USA*, 2005.
- [13] S. Mallat. *A wavelet tour of signal processing*. Academic Press, 1998.
- [14] J.-L. Starck, E.J. Candès, and D.L. Donoho. The curvelet transform for image denoising. *IEEE Transactions on Image Processing*, 11(6):670–684, June 2002.
- [15] J.-L. Starck, F. Murtagh, E. Candès, and D.L. Donoho. Gray and color image contrast enhancement by the curvelet transform. *IEEE Transactions on Image Processing*, 12(6):706–717, 2003.
- [16] J.L. Starck, M.K. Nguyen, and F. Murtagh. Deconvolution based on the curvelet transform. In *International Conference on Image Processing*, pages II: 993–996, 2003.
- [17] J.L. Starck, M.K. Nguyen, and F. Murtagh. Wavelets and curvelets for image deconvolution: a combined approach. *Signal Processing*, 83:2279–2283, 2003.
- [18] R. Teyssier. Cosmological hydrodynamics with adaptive mesh refinement. a new high resolution code called ramses. *Astronomy and Astrophysics*, 385:337–364, April 2002.