

## Table of content

Part 1	Summary of ULM processing step.....	3
Part 2	: Point Spread Function <i>In Silico</i> Data .....	9
Part 3	: <i>In silico</i> canal simulation.....	15
Part 4	: <i>In vivo</i> ULM datasets.....	23
Part 5	: Global Scoring.....	33
Part 6	: Tables .....	35

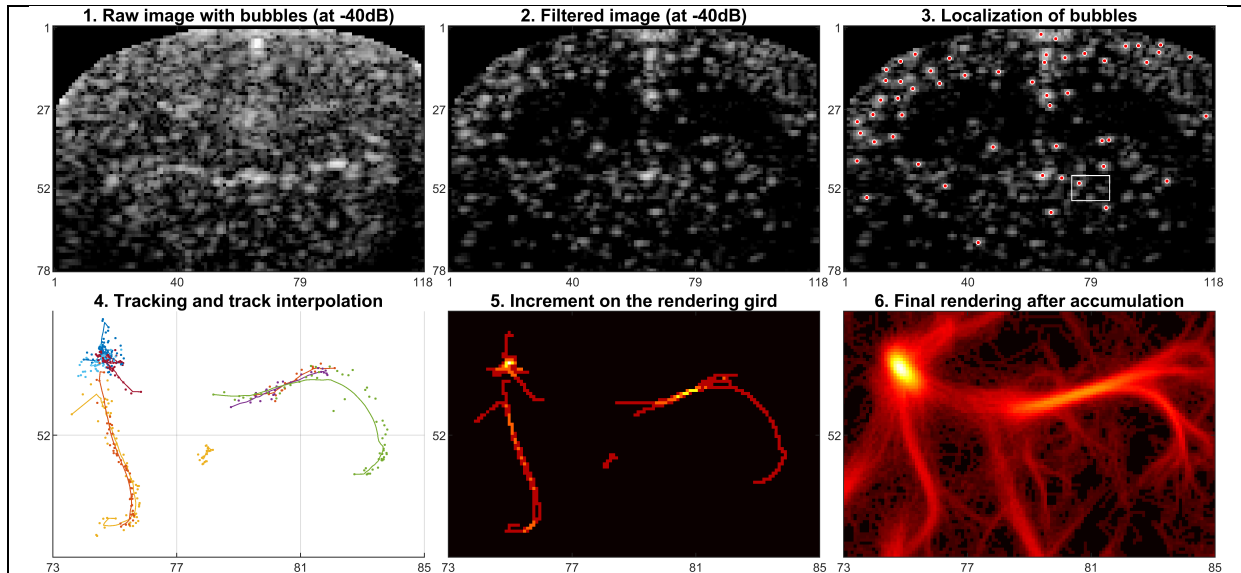
## Table of figures

Supplementary figure 1-1 ULM processing steps .....	3
Supplementary figure 1-2 Example of density rendering of the <i>in vivo</i> rat brain perfusion dataset .....	4
Supplementary figure 1-3 Example of density rendering with axial velocity encoding of the <i>in vivo</i> rat brain perfusion dataset.....	4
Supplementary figure 1-4 Example of velocity rendering of the <i>in vivo</i> rat brain perfusion dataset .....	4
Supplementary figure 1-5 Illustrations localisation algorithms on a centered and cropped image of a simulated microbubble of <i>in silico</i> dataset.....	5
Supplementary method 1-6 Analytical solution for the weighted average method.....	6
Supplementary method 1-7 Analytical solution for the radial symmetry algorithm.....	7
Supplementary figure 2-1 Simulated scatterers position over a $\lambda$ space .....	9
Supplementary figure 2-2 Beamformed image of scatterer moved in a $(\lambda \times \lambda)$ space .....	9
Supplementary figure 2-3 Simulated clutter noise explanation.....	10
Supplementary figure 2-4 Localisation errors for <i>in silico</i> PSF dataset .....	10
Supplementary figure 2-5 Distributions of directional errors for PSF at various SNR levels .....	10
Supplementary figure 2-6 Error maps for a scatterer moved in a $\lambda$ space at various SNR levels.....	13
Supplementary figure 3-1 Simulated canals <i>in silico</i> with Verasonics Vantage Research Simulator .....	15
Supplementary figure 3-2 Density based rendering of localisation algorithms and tracking algorithm .....	15
Supplementary figure 3-3 Distribution of errors for <i>in silico</i> canal simulation for localisation and tracking schemes for various SNR levels .....	18
Supplementary figure 3-4 Explanation of separation estimation .....	20
Supplementary figure 3-5 Measured gap for separation estimation .....	20
Supplementary figure 3-6 Definition of statistical classification and results for different localisation and tracking schemes .....	22
Supplementary figure 4-1 Power Doppler rendering of <i>in vivo</i> rat brain dataset.....	23
Supplementary figure 4-2 Positions localized by all algorithms.....	24
Supplementary figure 4-3 Density rendering localisation algorithms with track interpolation.....	26
Supplementary figure 4-4 Peak to peak Power Spectral Density analysis to devise a gridding index .....	28
Supplementary figure 4-5 Density renderings for dataset " <i>in vivo</i> rat brain bolus".....	29
Supplementary figure 4-6 Density renderings for dataset " <i>in vivo</i> rat kidney".....	31
Supplementary figure 4-7 Density renderings for dataset " <i>in vivo</i> mouse tumor" .....	32
Supplementary figure 5-1 PALA global score obtained for all the 7 algorithms tested .....	33
Supplementary table 5-2 Table with all the index values and score conversion .....	33
Supplementary table 5-3 Table of weights for different scoring scenarii .....	34
Supplementary table 5-4 Table weighted average scores for the 6 scenarii .....	34
Supplementary figure 6-1 state of the art of localisation methods for ULM .....	35
Supplementary figure 6-2 Summary of the media simulated.....	35
Supplementary figure 6-3 Summary of the metrics .....	35
Supplementary figure 6-4 Conversion of metrics to score.....	36

## Part 1 Summary of ULM processing step

This chapter details the ULM processing and localisation algorithms with graphical illustrations, and analytical formulas for weighted average and radial symmetry algorithms.

Supplementary figure 1-1 ULM processing steps

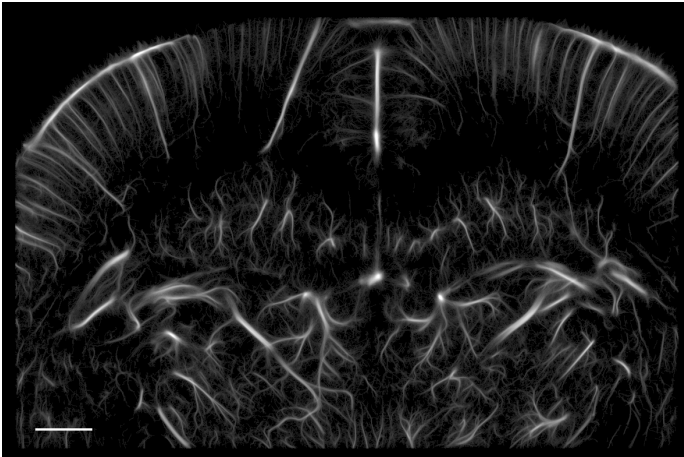


This illustrates the full processing of ULM algorithm.

1. Images of a perfused organ with microbubbles are acquired at high frame rate
2. Raw images are filtered to remove tissue signal
3. Microbubbles are detected and localized with a sub-pixel precision
4. Successive positions are paired together into tracks
5. After interpolation, tracks are projected on a rendering grid. Intensity pixel is incremented by 1 if a trajectory is present
6. By accumulating a large number of tracks, vascularisation can be reconstructed. (rendering with power law compression)

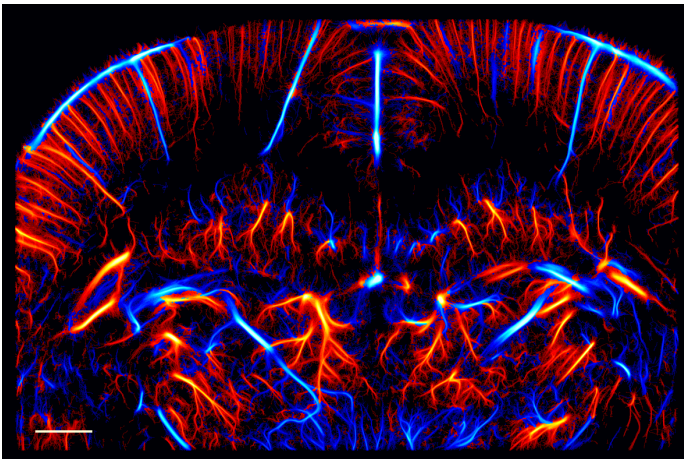
All this process has been implemented in this article and is provided in the supplementary online scripts.

### Supplementary figure 1-2 Example of density rendering of the *in vivo* rat brain perfusion dataset



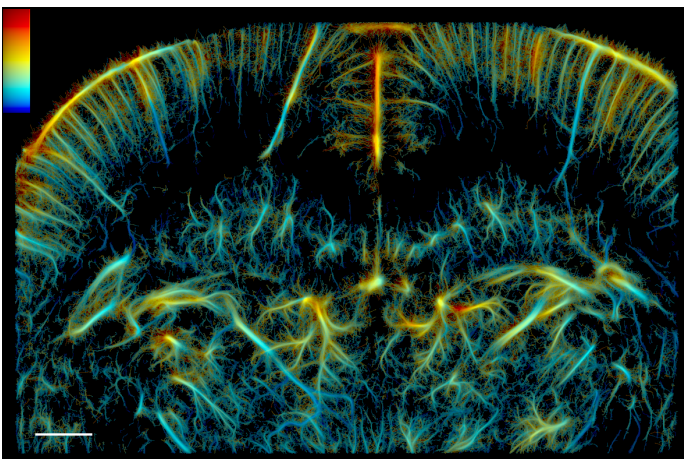
The density rendering represents the number of trajectories passing through a single pixel. Here, the square pixel's size has been defined to  $\frac{\lambda}{10} = 10\mu m$ . A power compression factor of 1/4 is applied to magnify the rendering. This example image has been processed with the radial symmetry algorithm. The associated code is provided in the online code package. Scale bar: 1 mm

### Supplementary figure 1-3 Example of density rendering with axial velocity encoding of the *in vivo* rat brain perfusion dataset



As ULM provides hemodynamic information, the axial direction of flow can be encoding in colors on the density image. Upward and backward flow can be distinguished. In this image, the blue color encodes the upward flows and the red color encodes the backward flows. This color encoding helps to differentiate arteries from veins in the neocortex. A power compression factor of 1/4 is applied on the density. Scale bar: 1 mm

### Supplementary figure 1-4 Example of velocity rendering of the *in vivo* rat brain perfusion dataset



In this image, the mean velocity is encoded with color. Slowest velocities appear blue, and high velocity regions are red with a max velocity of 70 mm/s. The velocity display range can be modified by users. A velocity power compression factor of 1/1.5 is applied to increase the dynamics of the rendering. Pixel's saturation is defined by the density image, with a power compression of 1/4, in order to shadow low density regions. Scale bar: 1 mm



**Supplementary figure 1-5 Illustrations localisation algorithms on a centered and cropped image of a simulated microbubble of *in silico* dataset**

	<p>Simulated beamformed image centered on a microbubble. The pixel size is <math>\lambda \times \lambda</math>. The real position of the simulated scatterer is <math>20.3\lambda</math> in the lateral position, and <math>61.3\lambda</math> in the axial position and is represented by the yellow solid dot. To apply the localisation kernel, the full image is cropped and centered on the pixel with the highest intensity. The localisation kernel returns the position shift of the estimated. The No-shift algorithm will always return a null shift. The image is rendered with a log compression.</p>
	<p><b>Interpolation based algorithm</b>                  Simulated beamformed image centered on a microbubble with interpolation to illustrate interpolation-based algorithms. Each pixel is divided into <math>10 \times 10</math> subpixels. Different localisation methods can be used: cubic, Lanczos, spline.</p>
	<p><b>Weighted average algorithm</b>                  Simulated beamformed image centered on a microbubble. Arrows illustrate the process of weighted average algorithm. For the axial shift, the image intensity (without log compression) is summed over the 5 lateral pixels, and weighted with <math>[-2; -1; 0; 1; 2]</math> coefficients and divided by the total intensity.</p> $z_{shift} = \frac{\sum_{i=-2}^2 \sum_{j=-2}^2 I(i, j) * j}{\sum_{i=-2}^2 \sum_{j=-2}^2 I(i, j)}$
	<p><b>Gaussian fitting algorithm</b>                  The top right-hand image is the input image of a centered microbubble with a log compression. The bottom right-hand image shows the Gaussian function used for the fitting.</p>
	<p><b>Radial symmetry algorithm</b>                  Simulated beamformed image centered on a microbubble with equipotential lines plotted in blue. The red arrows show the direction and amplitude of the gradient and point on the microbubble position.</p>

### Supplementary method 1-6 Analytical solution for the weighted average method

Let's assume you have an image of a microbubble composed of a grid of  $[N_z, N_x]$  pixels. To recover the position of the centroid, the interpolation schemes will upsample that image on a grid of  $[N_z * res, N_x * res]$  pixels, with  $res$  the upscaling resolution factor and then either find the maximum intensity or the centroid of the intensity distribution. The intensity  $I_{sr}$  coming out of the interpolation will be expressed as a linear combination of original intensities:

$$I_{sr}(i, j) = \sum_{(k_i, k_j) \in Z_q} I_k \phi_{int}(i - k_i, j - k_j)$$

where  $(i, j)$  belong to the interpolated space  $Z_{sr}$  and  $(k_i, k_j)$  belong to the original space  $Z_q$ .

The function  $\phi_{int}$  is called the synthesis function and can take many forms as long as it satisfies the interpolation property: it must vanish for all already known samples of the intensity except for the origin where it must take the value 1 - more simply put, if the intensity is known at the pixel in the departure grid, interpolation can not change its value in the arrival grid. This is the classical approach to interpolation, there is a more general approach where the synthesis function does not satisfy the interpolation property and is not necessarily finite support and where the coefficients are calculated based on the original intensity values but are not necessarily equal to them. The operation defined above is a discrete convolution equation.

Our assumption under the weighted average localisation scheme is that one does not need to calculate all of the intensities on the refined grid to perform localisation, but we can calculate the position of the maximum intensity based on centroids.

Mathematically speaking, our problem becomes:  $z_c = \left\{ z \in Z_{sr} / I_{sr}(z) = \max_{z \in Z_{sr}} I \right\}$  ;  $x_c = \left\{ x \in X_{sr} / I_{sr}(x) = \max_{x \in X_{sr}} I \right\}$

with  $(z_c, x_c)$  the coordinates of the centroid,  $(Z_{sr}, X_{sr})$  the subset of coordinates in the super-resolved basis, and  $I$  the intensity of the image.

We assume that we have centered our subset space  $Z_k$  on the maximum known value of  $I_k$  at  $z_{ck}$  and a Gaussian distribution of the intensity in our subset. The sub-pixel location of the peak can be estimated by calculating the centroid of the intensities<sup>29</sup>. In an image with intensities  $I(i, j)$ , we can define the image moments as:

$$M_{pq} = \sum_i \sum_j i^p j^q I(i, j),$$

and the centroid is defined by :

$$(z_c, x_c) = \left\{ \frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right\}$$

This centroid is equal to the location of the peak only if the intensity is Gaussian and in our discrete case, it can be written as:

$$z_c = z_{ck} + \frac{\sum_{i=-\lfloor \frac{f_z}{2} \rfloor}^{\lfloor \frac{f_z}{2} \rfloor} \sum_{j=-\lfloor \frac{f_x}{2} \rfloor}^{\lfloor \frac{f_x}{2} \rfloor} I(i, j) w_z(i, j)}{\sum_{i=-\lfloor \frac{f_z}{2} \rfloor}^{\lfloor \frac{f_z}{2} \rfloor} \sum_{j=-\lfloor \frac{f_x}{2} \rfloor}^{\lfloor \frac{f_x}{2} \rfloor} I(i, j)} ; x_c = x_{ck} + \frac{\sum_{i=-\lfloor \frac{f_z}{2} \rfloor}^{\lfloor \frac{f_z}{2} \rfloor} \sum_{j=-\lfloor \frac{f_x}{2} \rfloor}^{\lfloor \frac{f_x}{2} \rfloor} I(i, j) w_x(i, j)}{\sum_{i=-\lfloor \frac{f_z}{2} \rfloor}^{\lfloor \frac{f_z}{2} \rfloor} \sum_{j=-\lfloor \frac{f_x}{2} \rfloor}^{\lfloor \frac{f_x}{2} \rfloor} I(i, j)}$$

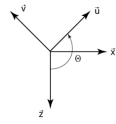
with the weights defined as:  $w_z(i, j) = i$  ;  $w_x(i, j) = j$  ; and  $(z_{ck}, x_{ck}) = \{(z, x) > I(i, j), \forall (i, j)\}$

And  $f_z = FWHM_z, f_x = FWHM_x$  the Full Width at Half Maximum of the intensity profile in the  $z, x$  direction respectively.

### Supplementary method 1-7 Analytical solution for the radial symmetry algorithm

Let  $I$  be a  $5 \times 5$  matrix containing a microbubble:  $I = \begin{bmatrix} I_{11} & I_{12} & I_{13} & I_{14} & I_{15} \\ I_{21} & I_{22} & I_{23} & I_{24} & I_{25} \\ I_{31} & I_{32} & I_{33} & I_{34} & I_{35} \\ I_{41} & I_{42} & I_{43} & I_{44} & I_{45} \\ I_{51} & I_{52} & I_{53} & I_{54} & I_{55} \end{bmatrix}$

We define  $(\vec{u}, \vec{v})$  the basis as the rotation of the original basis  $(\vec{z}, \vec{x})$ , rotated by  $\theta = +\frac{3\pi}{4}$ , and the rotation matrix  $R_{(\vec{u}, \vec{v})}$ :



$$R_{(\vec{x}, \vec{z}) \rightarrow (\vec{u}, \vec{v})} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} = -\frac{\sqrt{2}}{2} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$$

and

$$R^{-1} = R_{(\vec{u}, \vec{v}) \rightarrow (\vec{x}, \vec{z})} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} = -\frac{\sqrt{2}}{2} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

We set:  $dIdv = I(1: N_z - 1, 1: N_x - 1) - I(2: N_z, 2: N_x)$ , and  $dIdu = I(1: N_z - 1, 2: N_x) - I(2: N_z, 1: N_x - 1)$

The line given by the equation  $dIdu \cdot \vec{u} + dIdv \cdot \vec{v}$  is the line that defines the gradient of intensity according to the basis  $(\vec{u}, \vec{v})$ . The line given by the equation:

$$(u \cdot \vec{u} + v \cdot \vec{v})(dIdu \cdot \vec{u} + dIdv \cdot \vec{v}) = 0$$

will define the subset  $(u, v)$  of coordinates defining an orthogonal to the intensity gradient and we can write it as:

$$\begin{aligned} \begin{bmatrix} u \\ v \end{bmatrix}^T \begin{bmatrix} dIdu \\ dIdv \end{bmatrix} = 0 &\Leftrightarrow \begin{bmatrix} u \\ v \end{bmatrix}^T \cdot R_{(\vec{u}, \vec{v}) \rightarrow (\vec{x}, \vec{z})} \begin{bmatrix} dIdu \\ dIdv \end{bmatrix} = 0 \\ &\Leftrightarrow \begin{bmatrix} u \\ v \end{bmatrix}^T \cdot -\frac{\sqrt{2}}{2} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} dIdu \\ dIdv \end{bmatrix} = 0 \\ &\Leftrightarrow [u \ v] \cdot \begin{bmatrix} dIdu - dIdv \\ dIdu + dIdv \end{bmatrix} = 0 \\ &\Leftrightarrow u \cdot (dIdu - dIdv) + v \cdot (dIdu + dIdv) = 0 \end{aligned}$$

Let  $(u_k, v_k) = (z_k, x_k)$  be the coordinates belonging to the orthogonal to the gradient, we need to find the center  $(z_c, x_c)$  minimizing the distance to the orthogonal of the gradient. If we want our basis to be centered around  $(z_c, x_c)$ , the equation above becomes:  $(z_k - z_c) - m(x_k - x_c) = 0$

with  $m = \frac{dIdu + dIdv}{dIdv - dIdu}$ .

The distance from the point  $C(z_c, x_c)$  to the point  $K(z_k, x_k)$  belonging to the line directed by the vector  $\vec{u}_d$  orthogonal to the gradient in  $K$  is

$$d(c, K \in \vec{u}_d) = \frac{\|\vec{KC} \wedge \vec{u}_d\|}{\|\vec{u}_d\|}$$

thus :  $d(c, K \in \vec{u}_d) = d_k = \frac{|(z_k - z_c) - m(x_k - x_c)|}{1 + m^2}$

Let  $\chi^2 = \sum_k d_k^2 w_k$  with  $w_k$  a weight given to each pixel. The goal of this is to compensate for low SNR in areas where the gradient is low, and:

$$w_k = \frac{dImag^2}{\sqrt{(z_k - z_c)^2 + (x_k - x_c)^2}}$$

and  $dImag^2 = dIdu^2 + dIdv^2$

As  $w_k$ , is defined with respect to  $(z_c, x_c)$ , we will use the classical weighted average method as an initial guess of  $(z_c, x_c)$  to calculate each  $w_k$ . To find the maximum intensity point, one must

minimize all of the distance  $d_k$ , for example, minimize the sum of all distances  $d_k$ . We will thus differentiate  $\chi^2$  with respect to  $(z_c, x_c)$  and solve for  $(z_c, x_c)$ .

$$\frac{\partial \chi^2}{\partial x_c} = \frac{\partial \sum_k d_k^2 w_k}{\partial x_c} = \sum_k \frac{\partial \left( \frac{((z_k - z_c) - m_k(x_k - x_c))^2 w_k}{m_k^2 + 1} \right)}{\partial x_c} = \sum_k \frac{2((z_k - z_c) - m_k(x_k - x_c)) m_k w_k}{m_k^2 + 1}$$

solving for  $\frac{\partial \chi^2}{\partial x_c} = 0$  leads to:

$$\sum_k \frac{m_k w_k}{m_k^2 + 1} ((z_k - z_c) - m_k(x_k - x_c)) = 0$$

$$\sum_k \frac{m_k w_k}{m_k^2 + 1} * z_c - \sum_k \frac{m_k^2 w_k}{m_k^2 + 1} * x_c = \sum_k \frac{m_k w_k}{m_k^2 + 1} * (z_k - m_k x_k)$$

and for  $\frac{\partial \chi^2}{\partial z_c} = 0$

$$\sum_k \frac{w_k}{m_k^2 + 1} * z_c - \sum_k \frac{m_k w_k}{m_k^2 + 1} * x_c = \sum_k \frac{w_k}{m_k^2 + 1} * (z_k - m_k x_k)$$

If we write this with matrices, it becomes:

$$\begin{bmatrix} \sum_k \frac{m_k w_k}{m_k^2 + 1} & \sum_k \frac{m_k^2 w_k}{m_k^2 + 1} \\ \sum_k \frac{w_k}{m_k^2 + 1} & \sum_k \frac{m_k w_k}{m_k^2 + 1} \end{bmatrix} = \begin{bmatrix} \sum_k \frac{m_k w_k}{m_k^2 + 1} * (z_k - m_k x_k) \\ \sum_k \frac{w_k}{m_k^2 + 1} * (z_k - m_k x_k) \end{bmatrix}$$

Calculating the values of  $(z_c, x_c)$ , is straightforward from the expression above:

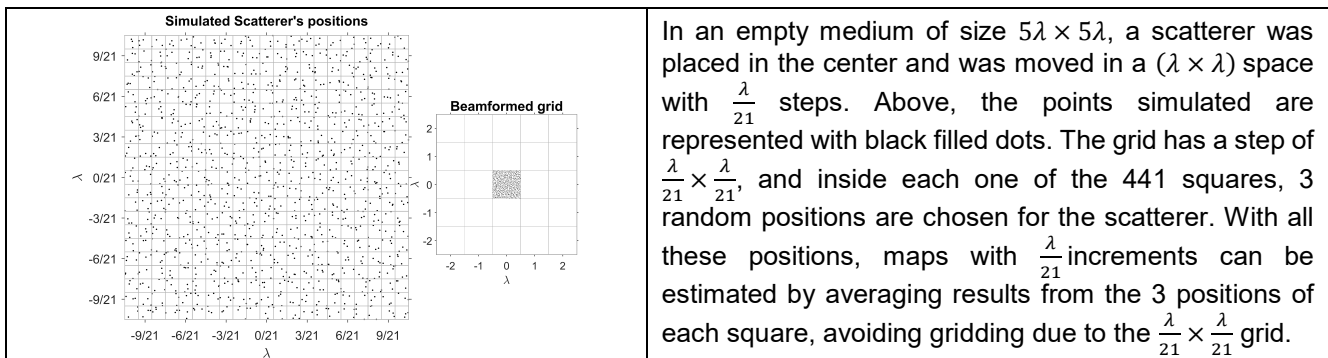
$$\text{Let } \Omega = \begin{bmatrix} \sum_k \frac{m_k w_k}{m_k^2 + 1} & \sum_k \frac{m_k^2 w_k}{m_k^2 + 1} \\ \sum_k \frac{w_k}{m_k^2 + 1} & \sum_k \frac{m_k w_k}{m_k^2 + 1} \end{bmatrix}$$

$$\begin{bmatrix} z_c \\ x_c \end{bmatrix} = \Omega^{-1} \begin{bmatrix} \sum_k \frac{m_k w_k}{m_k^2 + 1} * (z_k - m_k x_k) \\ \sum_k \frac{w_k}{m_k^2 + 1} * (z_k - m_k x_k) \end{bmatrix}$$

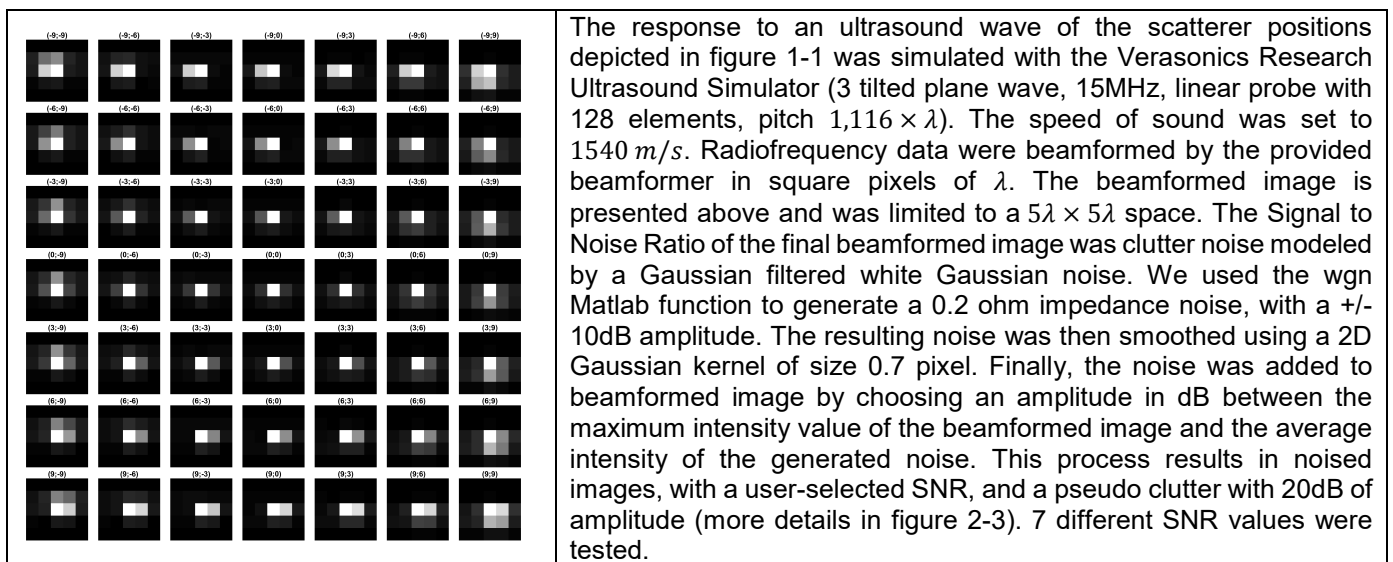
## Part 2 : Point Spread Function *In Silico* Data

We will present supplementary figures related to the dataset containing simulated scatterers designed to investigate the PSF inhomogeneity in space and its effect on localisation. The goal of these few figures is to help us understanding how the algorithms are affected by sub-wavelength displacements and how well they can recover one scatterers position with different SNR but without any motion and without trying to reconstruct canals. No tracking is performed in this chapter.

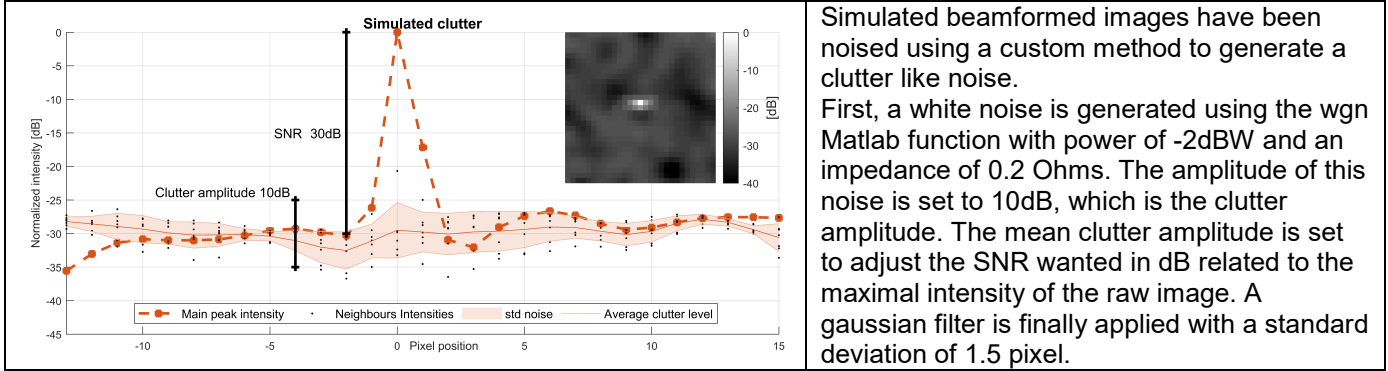
**Supplementary figure 2-1 Simulated scatterers position over a  $\lambda$  space**



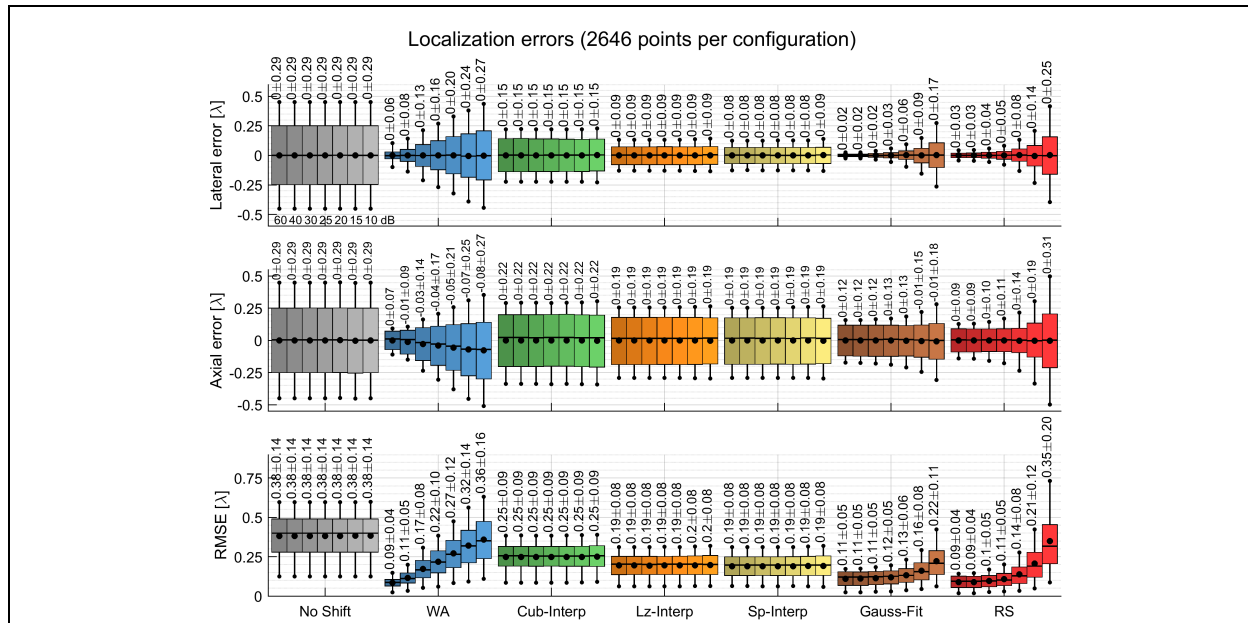
**Supplementary figure 2-2 Beamformed image of scatterer moved in a  $(\lambda \times \lambda)$  space**



### Supplementary figure 2-3 Simulated clutter noise explanation



### Supplementary figure 2-4 Localisation errors for *in silico* PSF dataset

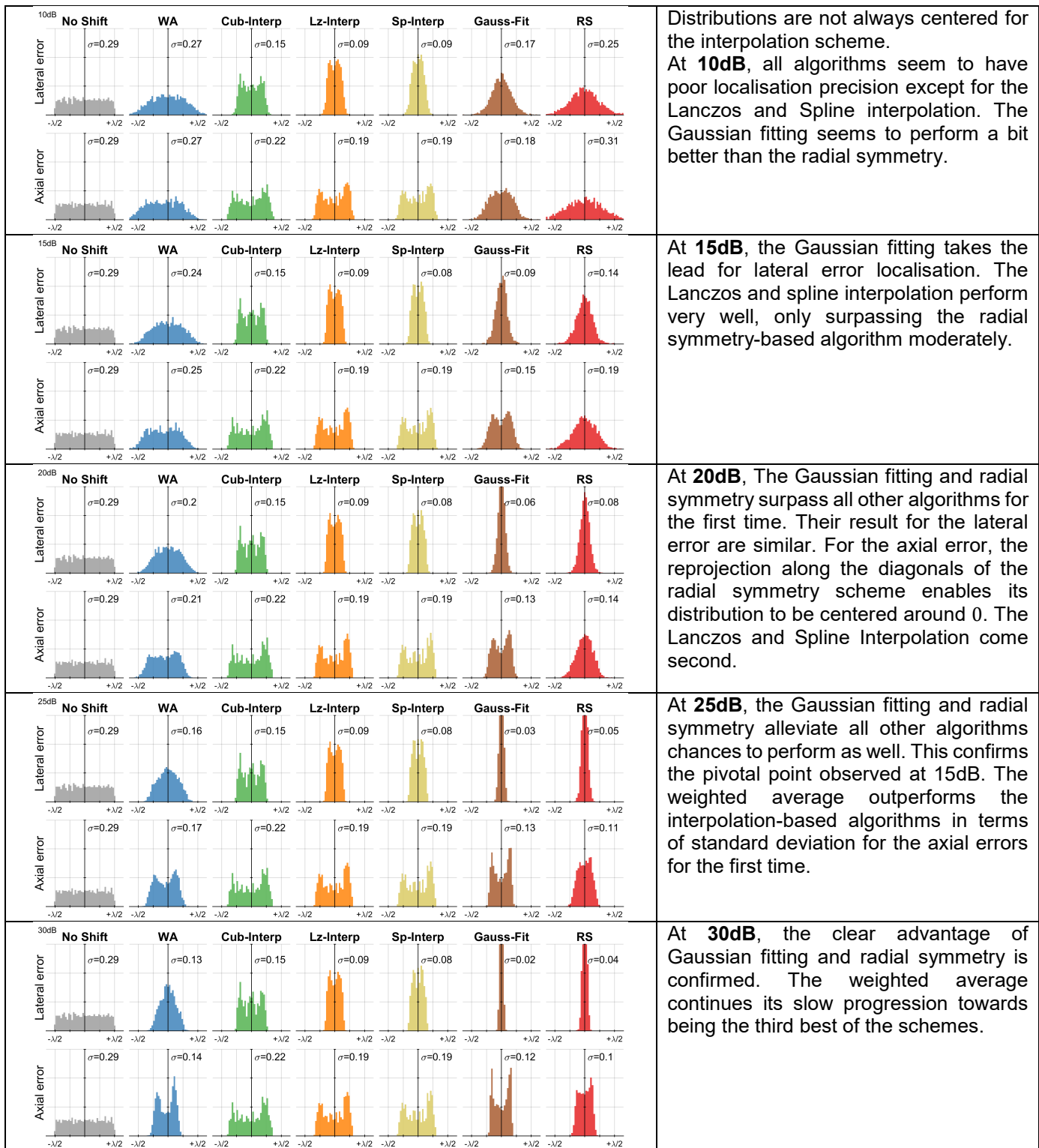


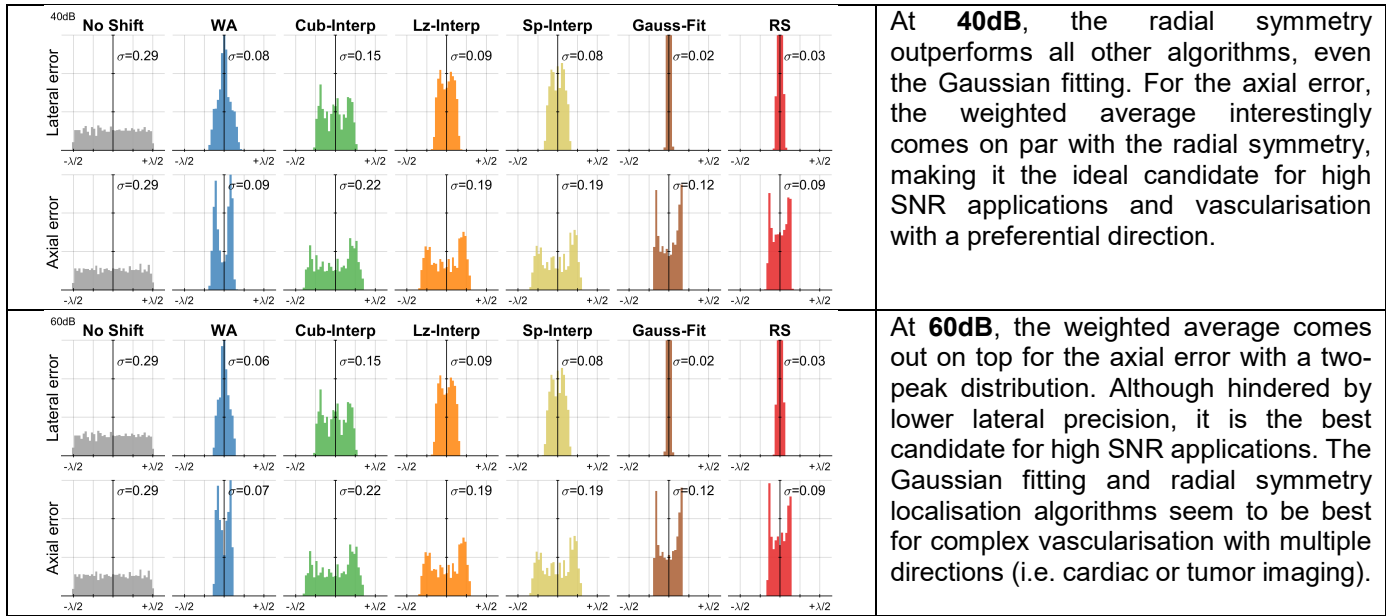
For each position simulated, localisation with the 7 different algorithms was calculated and compared to the true position. The axial, lateral and Root Mean Square error (RMSE) are plotted here. We can see that the standard deviations of the error for the interpolation schemes (Cub-Interp for cubic, Lz-Interp for Lanczos, Sp-Inter for spline) are almost constant as these schemes have a high average error and standard deviation. The No-Shift has the maximum error in all cases as it is reporting the scatterer to be exactly centered regardless of its intensity. The Gaussian fitting (Gauss-Fit) and radial symmetry (RS) scheme are more affected by SNR than the other schemes but have considerably lower lateral errors and standard deviations. Finally, it is important to note that the weighted average (WA) scheme has similar standard deviations to the best localisation algorithms until SNR drops below 40dB making it a very good candidate for high SNR applications such as in harmonic imaging in both directions. For the axial direction, it continues being a good candidate up until SNR drops below 25dB. This is very useful for vascularisation aligned in a preferential axis (i.e. the brain or the kidney).

### Supplementary figure 2-5 Distributions of directional errors for PSF at various SNR levels

For each of the position simulated, the lateral and axial error distributions are plotted.

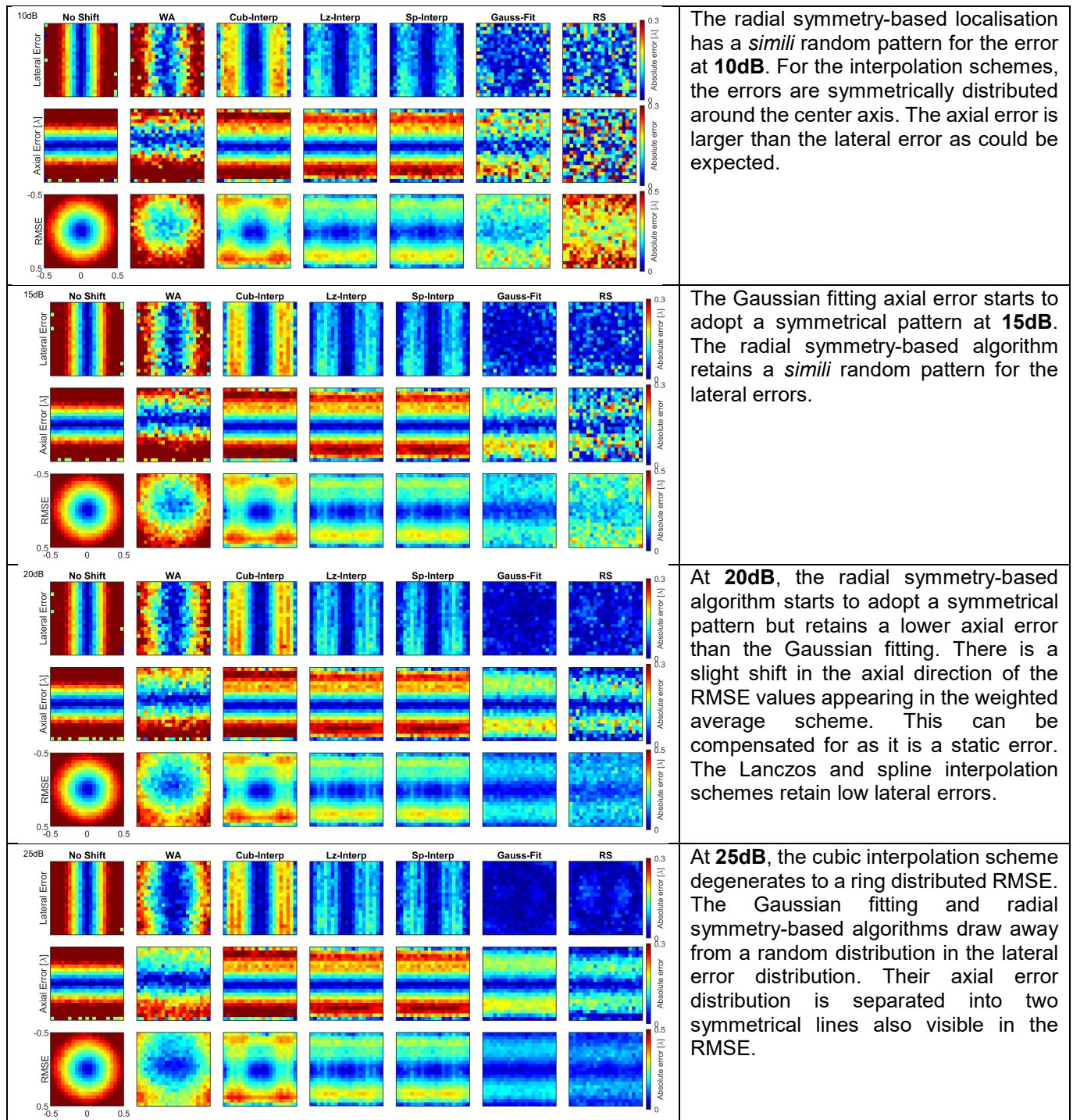


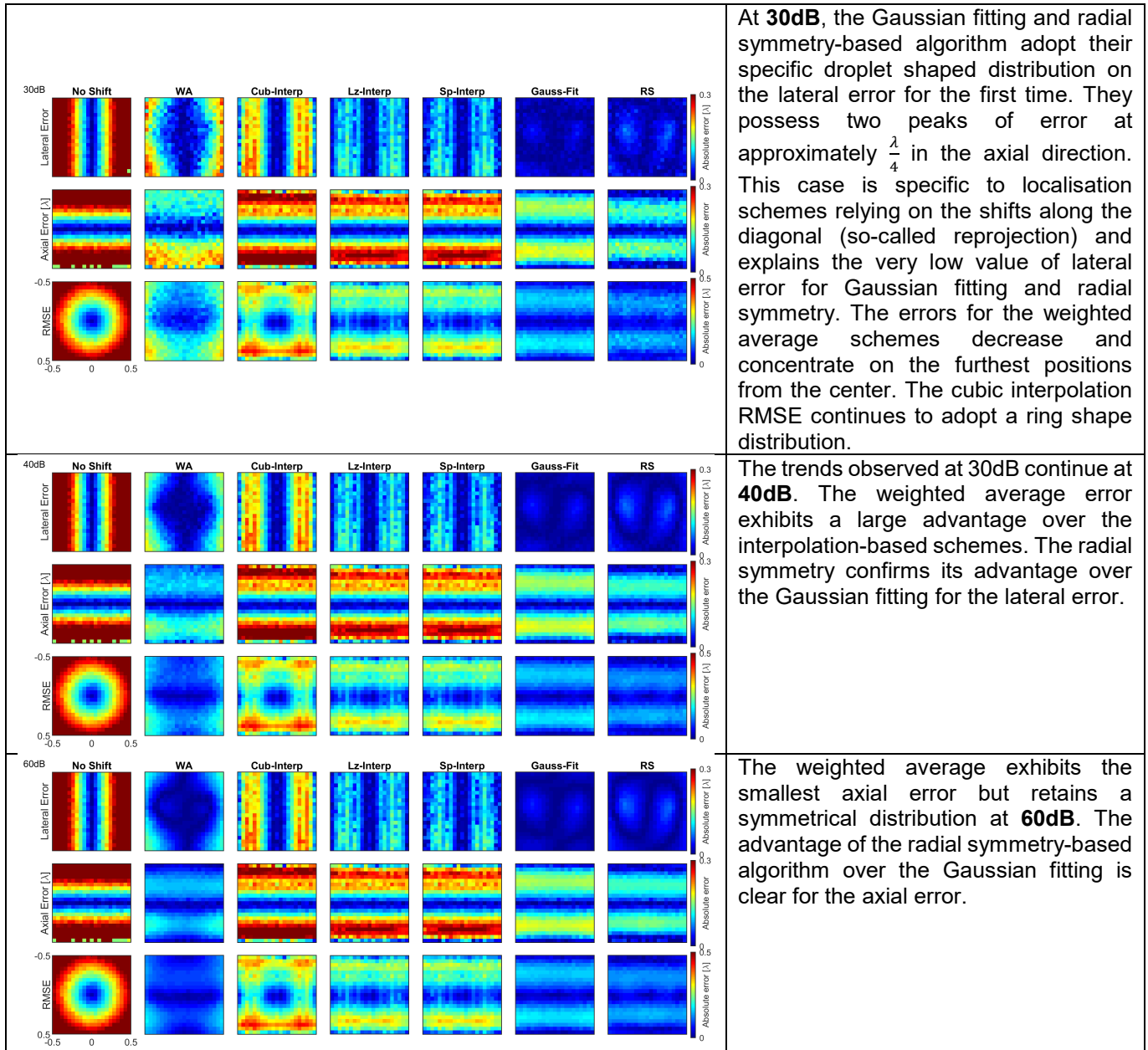




### Supplementary figure 2-6 Error maps for a scatterer moved in a $\lambda$ space at various SNR levels

For each of the position simulated, the lateral, axial error and RMSE are rendered with respect to their location.



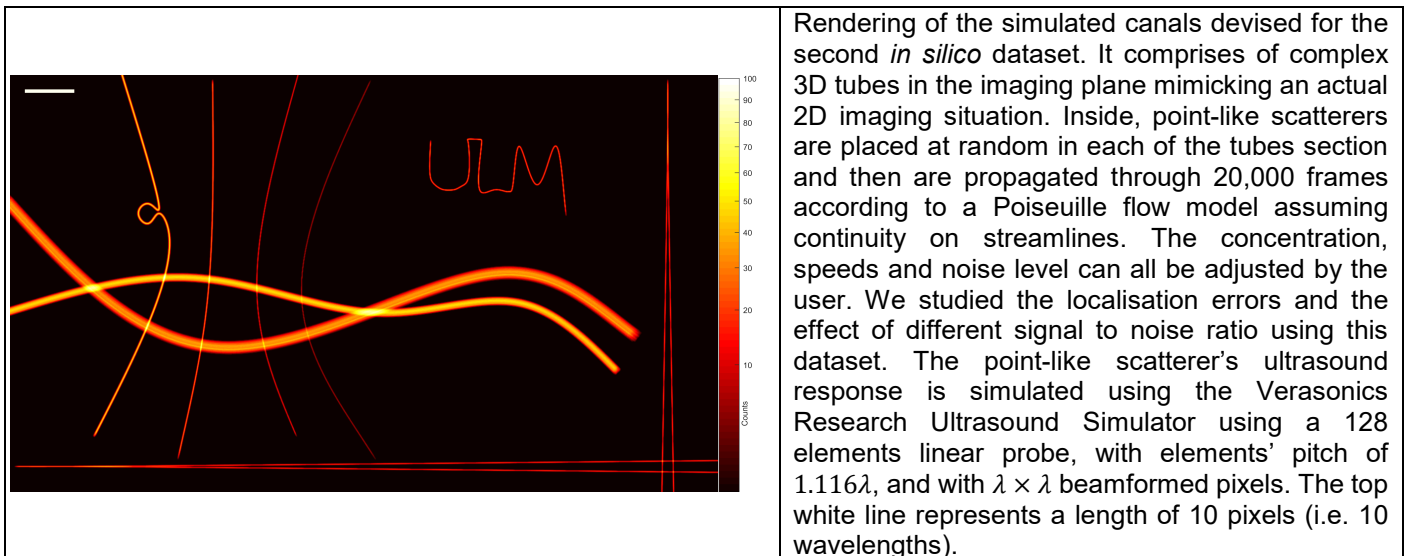


## Part 3 : *In silico* canal simulation

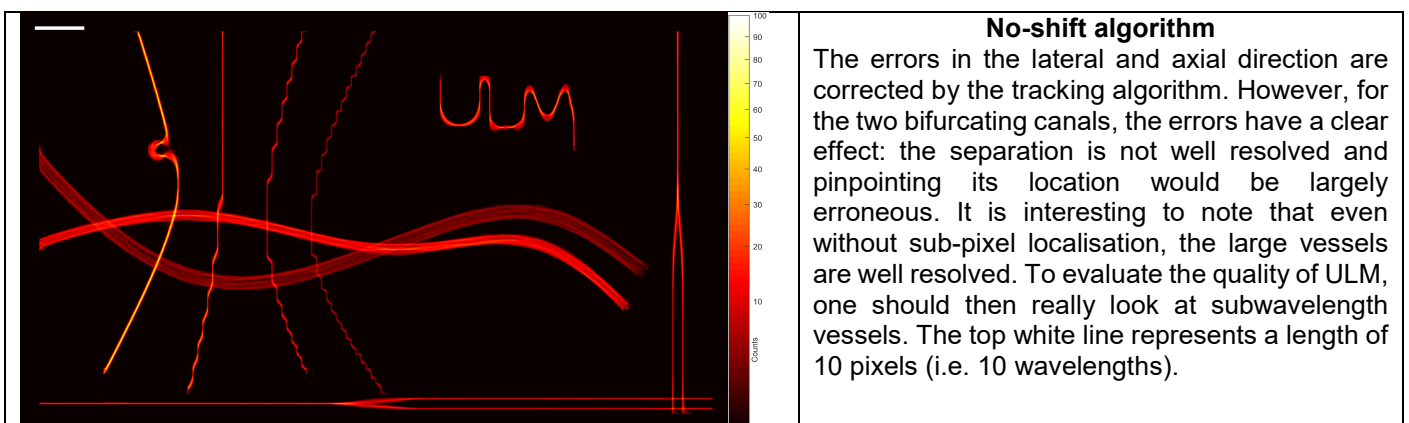
We will present here the dataset comprised of simulated canals. We have produced additional figures to explain how the dataset was devised, how well the different algorithms perform on this dataset with the addition of tracking. The idea behind this set is to be able to perform the complete ULM process and to have access to the ground truth to calculate statistical indices and look at complicated structures designed by our team. The errors of each algorithm are presented, as well as our separation index which was used to calculate the maximum attainable resolution.

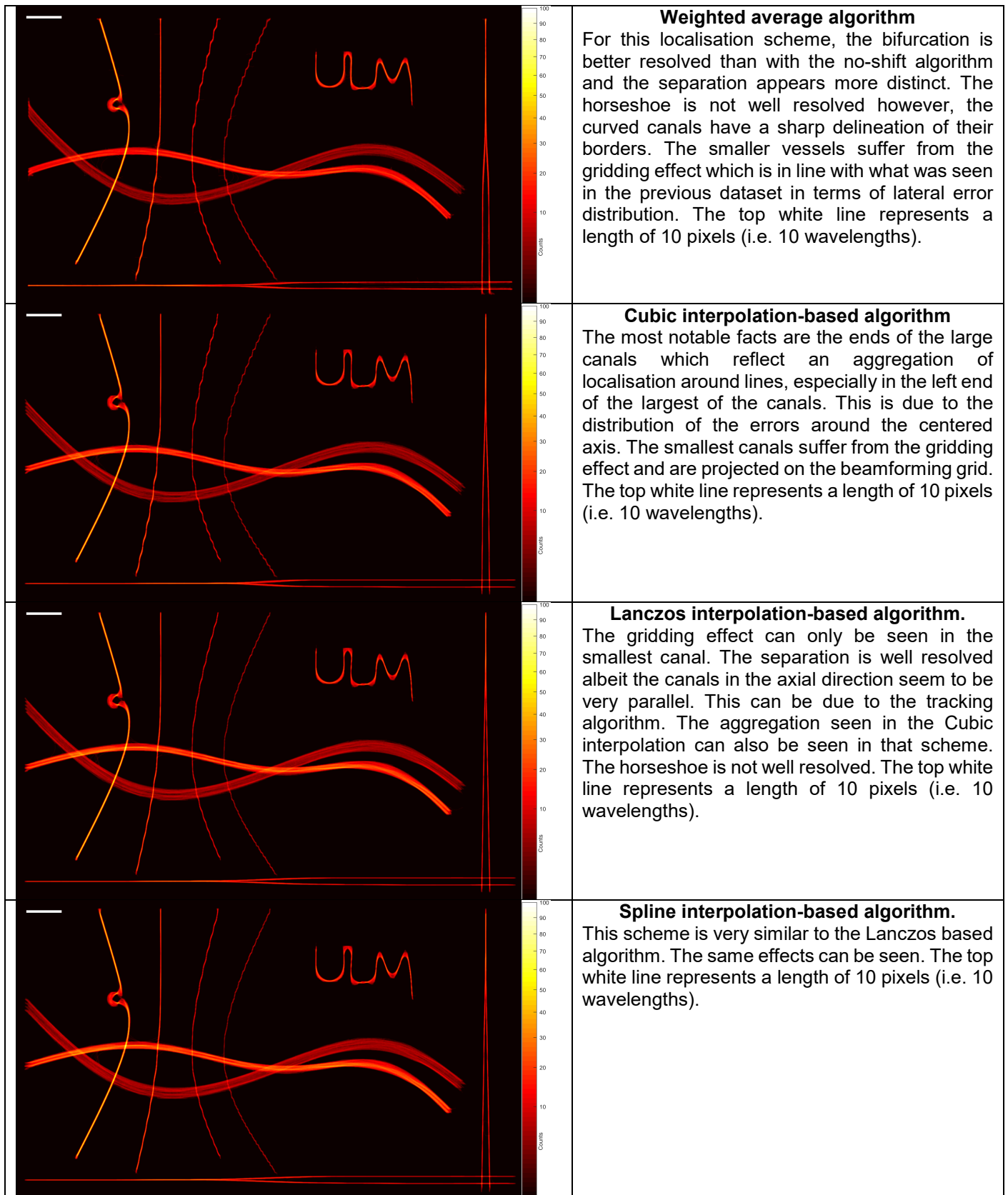
In the next 7 figures, the density renderings are obtained by counting the number of trajectories passing through each pixel of size  $\frac{\lambda}{10} \times \frac{\lambda}{10}$ . These trajectories are obtained by applying the 7 different localisation algorithms and then Kuhn-Munkres assignment-based tracking with a custom defined interpolation of microbubbles' trajectories. This will smooth the trajectories and restore a more natural curvature.

**Supplementary figure 3-1 Simulated canals *in silico* with Verasonics Vantage Research Simulator**

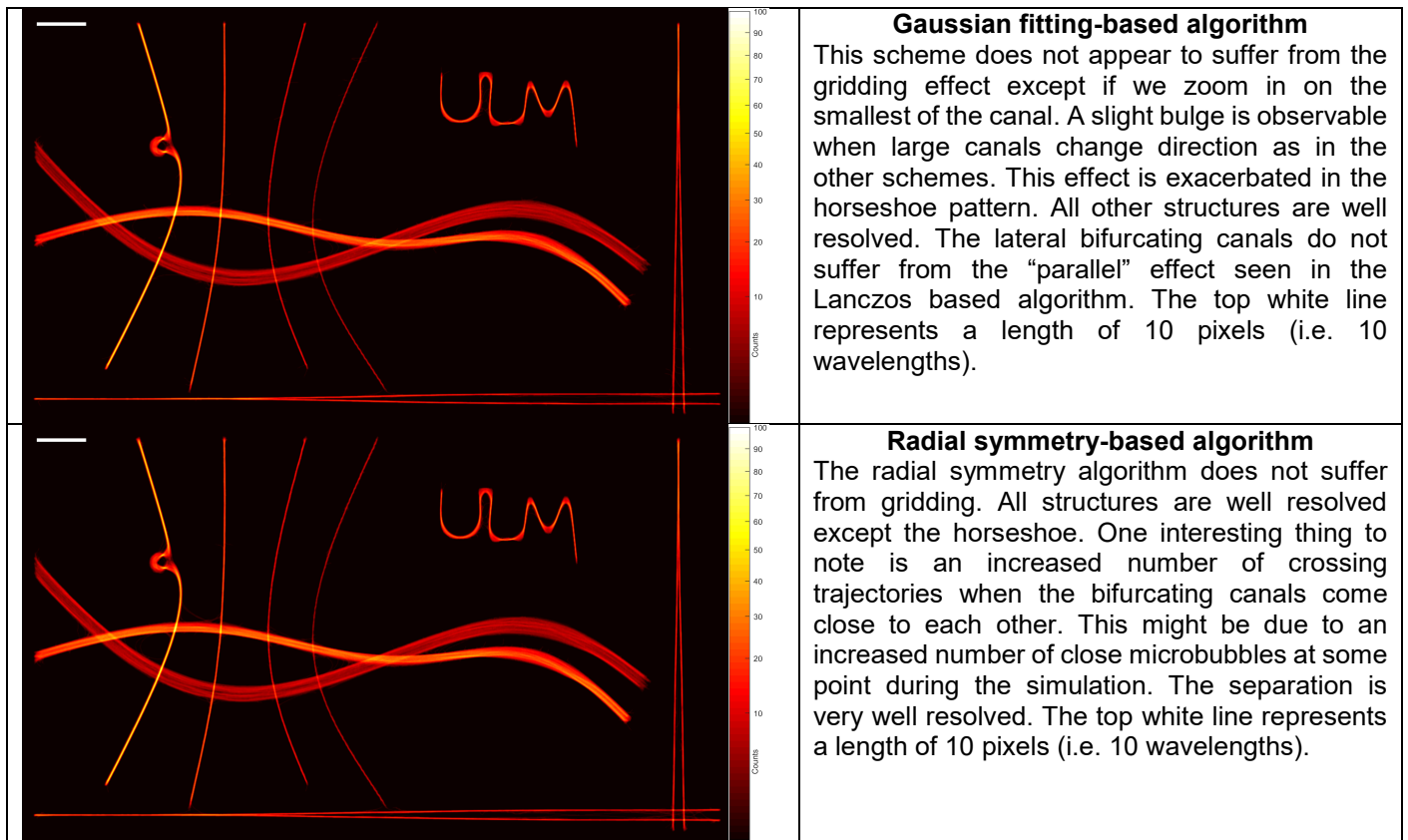


**Supplementary figure 3-2 Density based rendering of localisation algorithms and tracking algorithm**

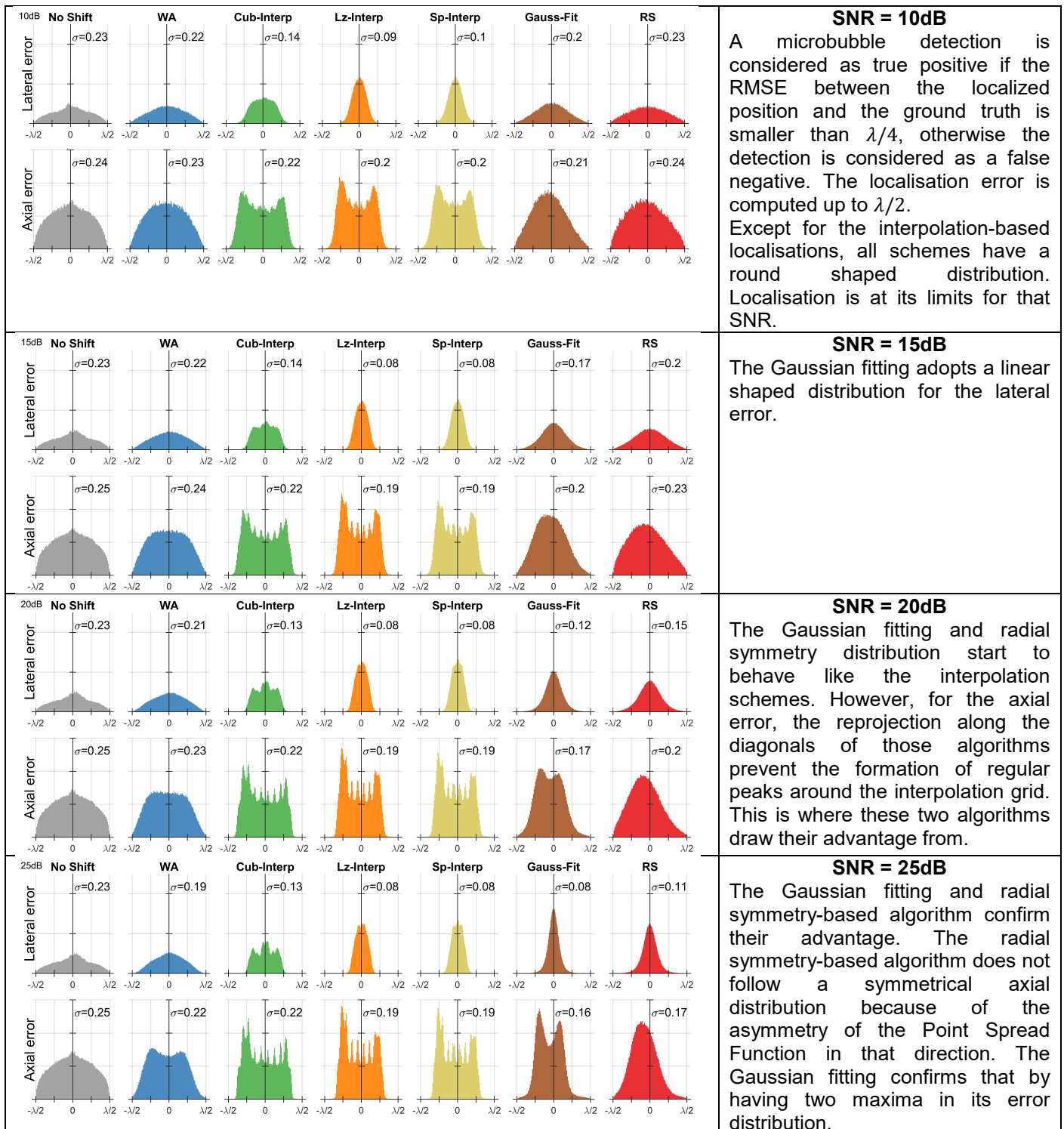


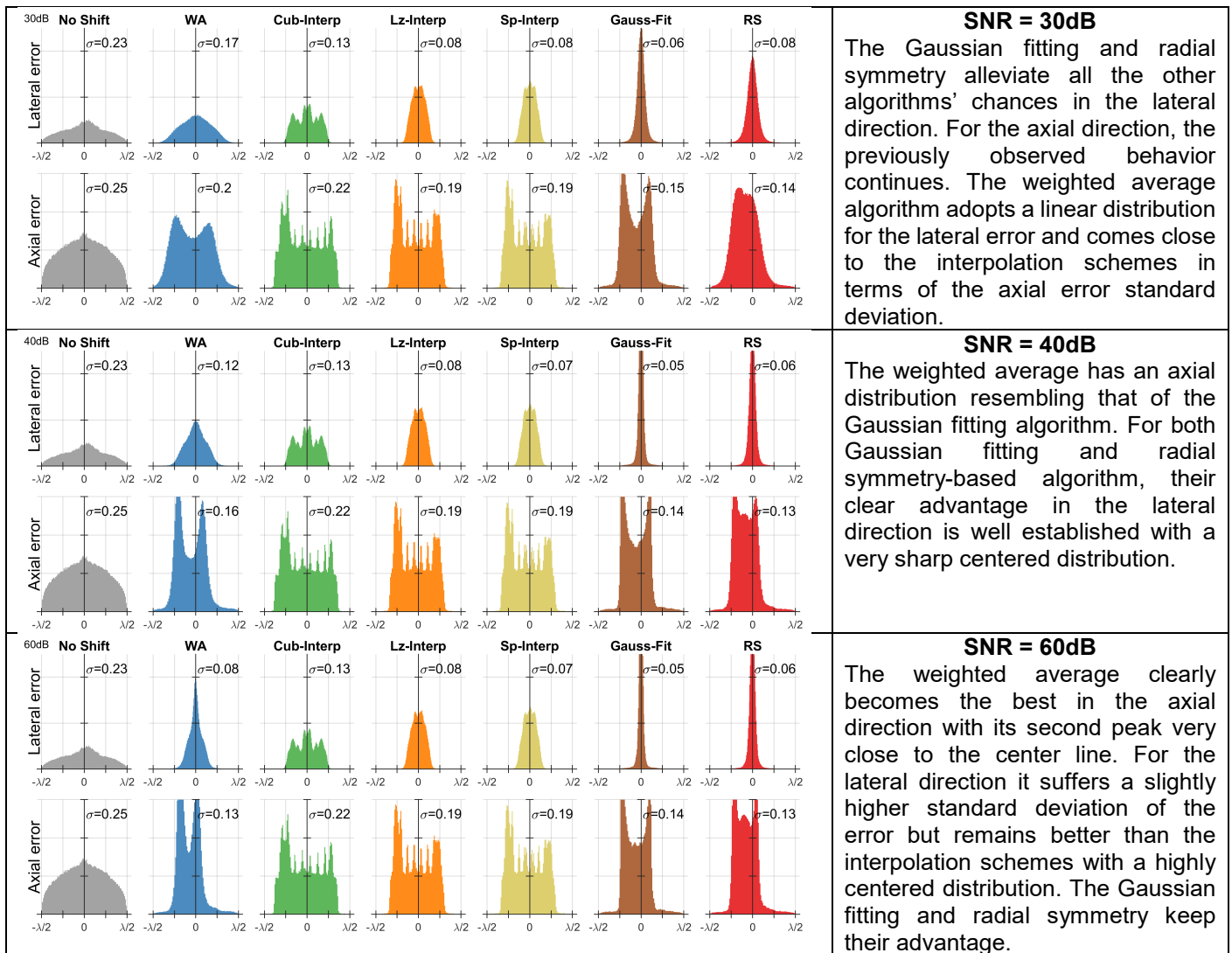




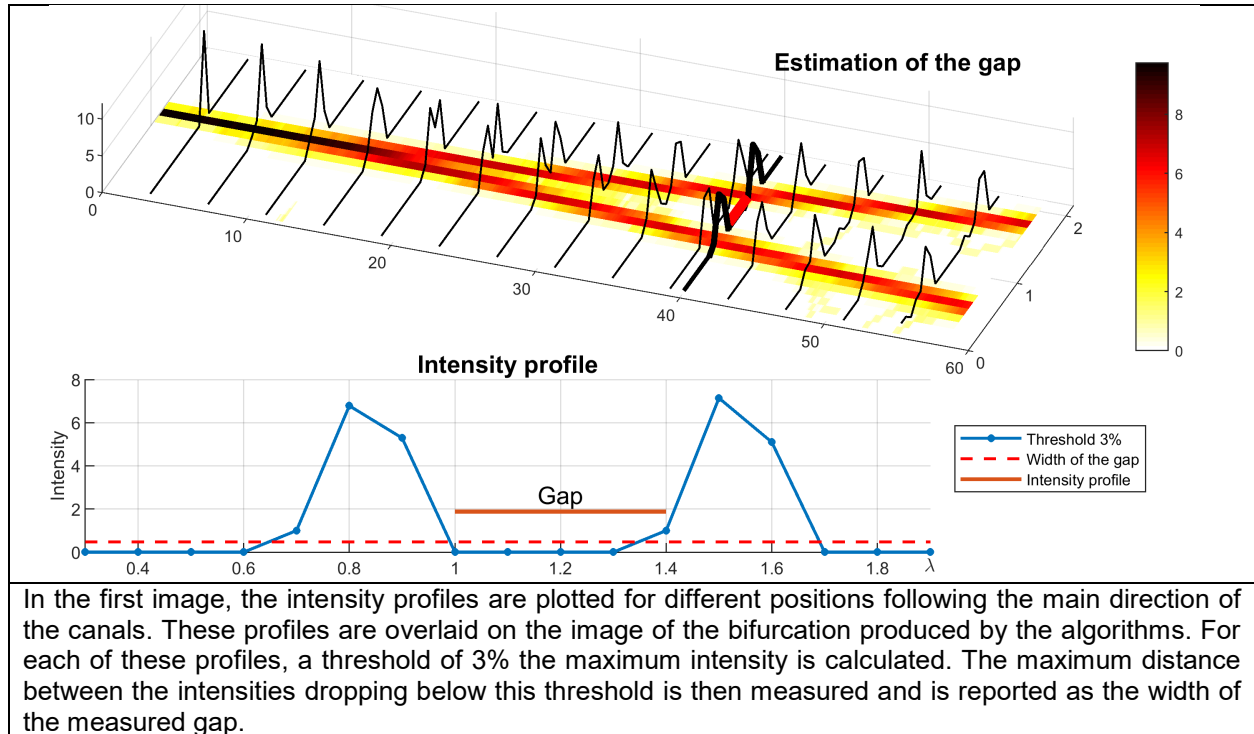


**Supplementary figure 3-3 Distribution of errors for *in silico* canal simulation for localisation and tracking schemes for various SNR levels**

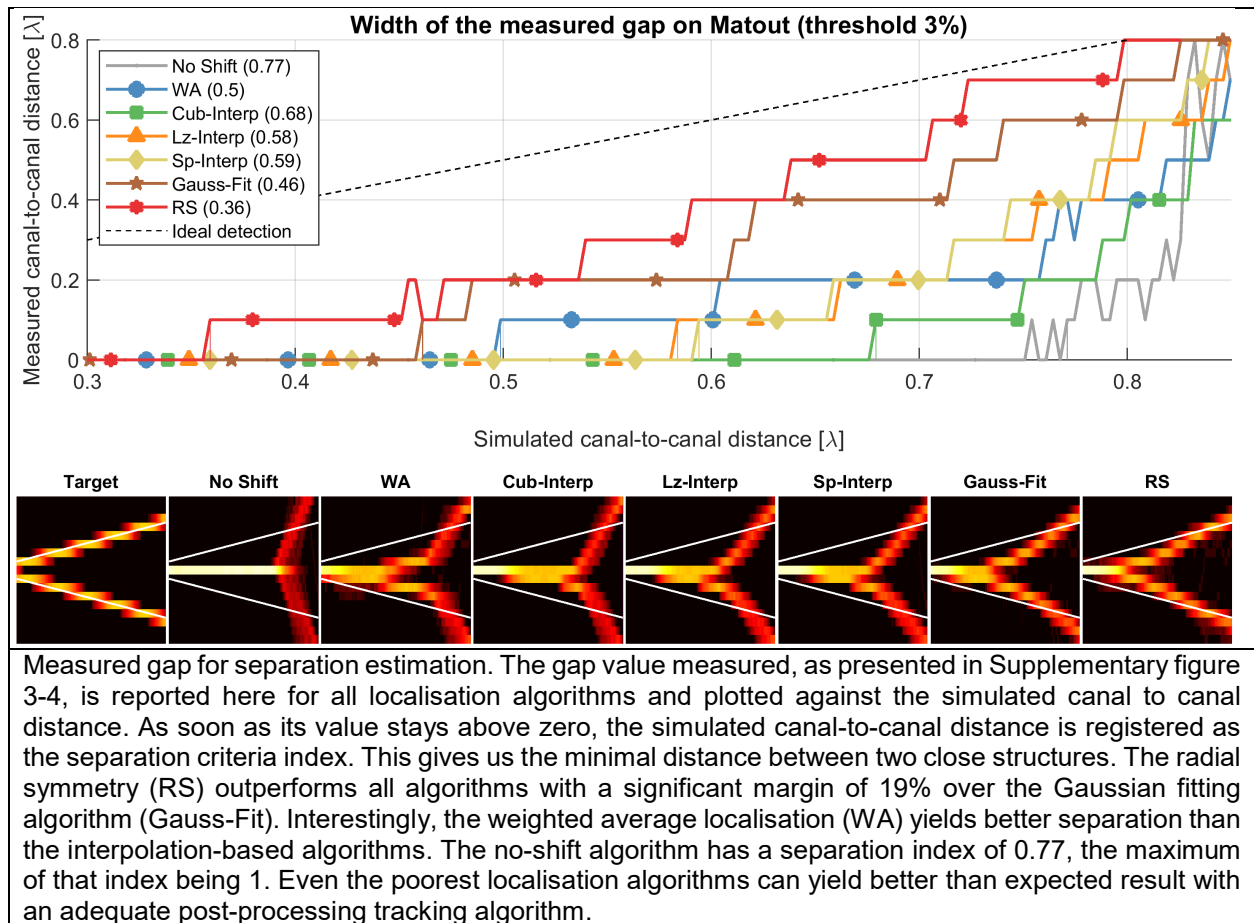




### Supplementary figure 3-4 Explanation of separation estimation

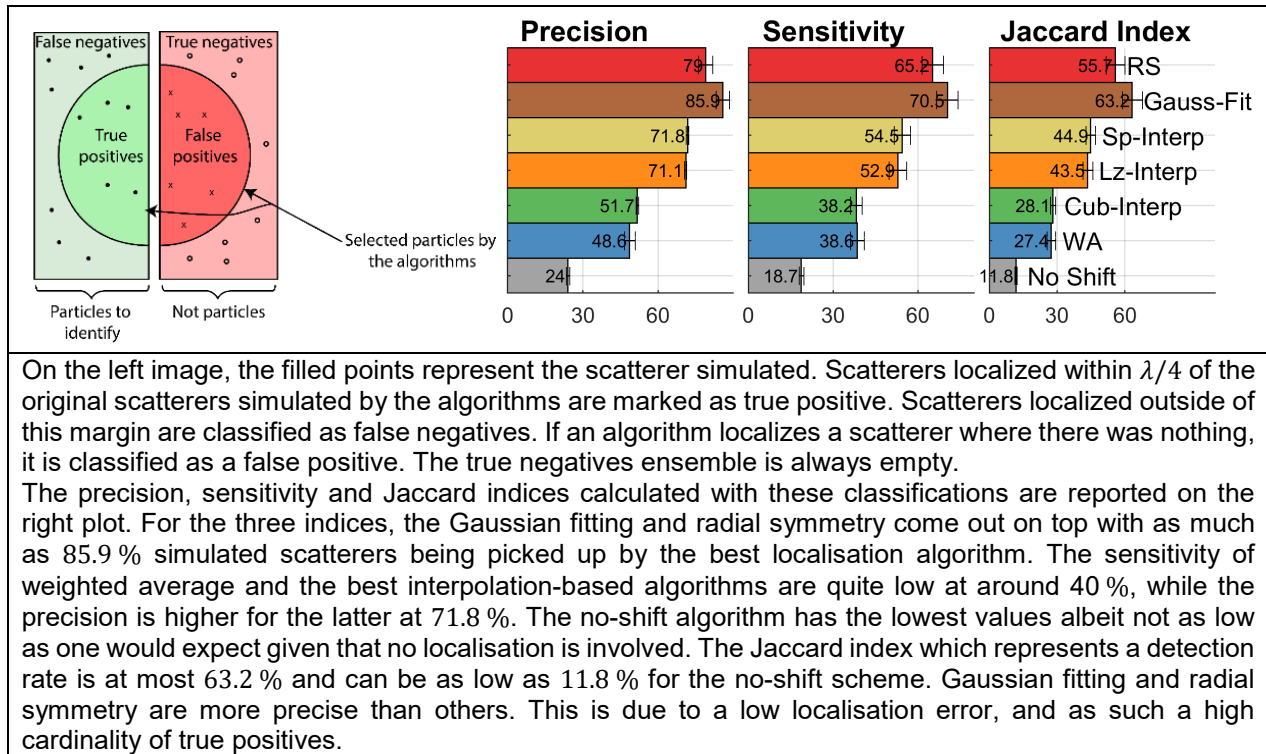


### Supplementary figure 3-5 Measured gap for separation estimation





### Supplementary figure 3-6 Definition of statistical classification and results for different localisation and tracking schemes



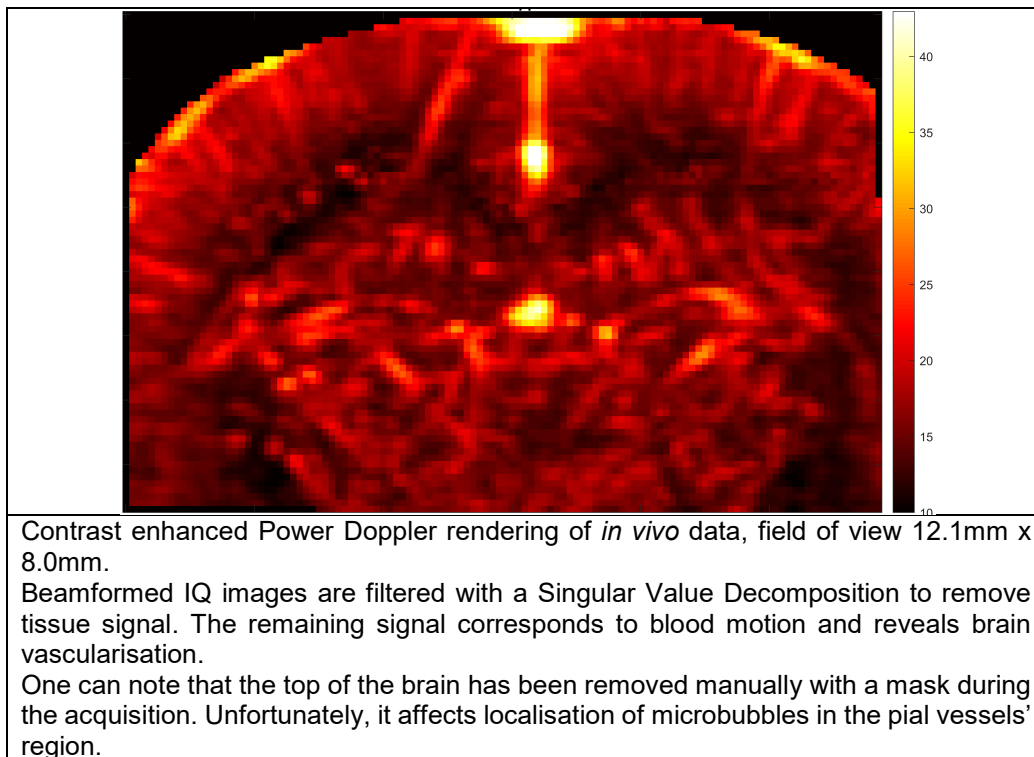


## Part 4 : *In vivo* ULM datasets

In this chapter, we will present the *in vivo* rat brain perfusion dataset. The ULM rendering was applied with and without track interpolation. Non-interpolated tracks reveal the behavior of localisation methods. The aim of this chapter is to present real-life images and to better understand the different steps of the ULM process. It also helps us to illustrate the gridding effect on *in vivo* data and calculate a gridding index.

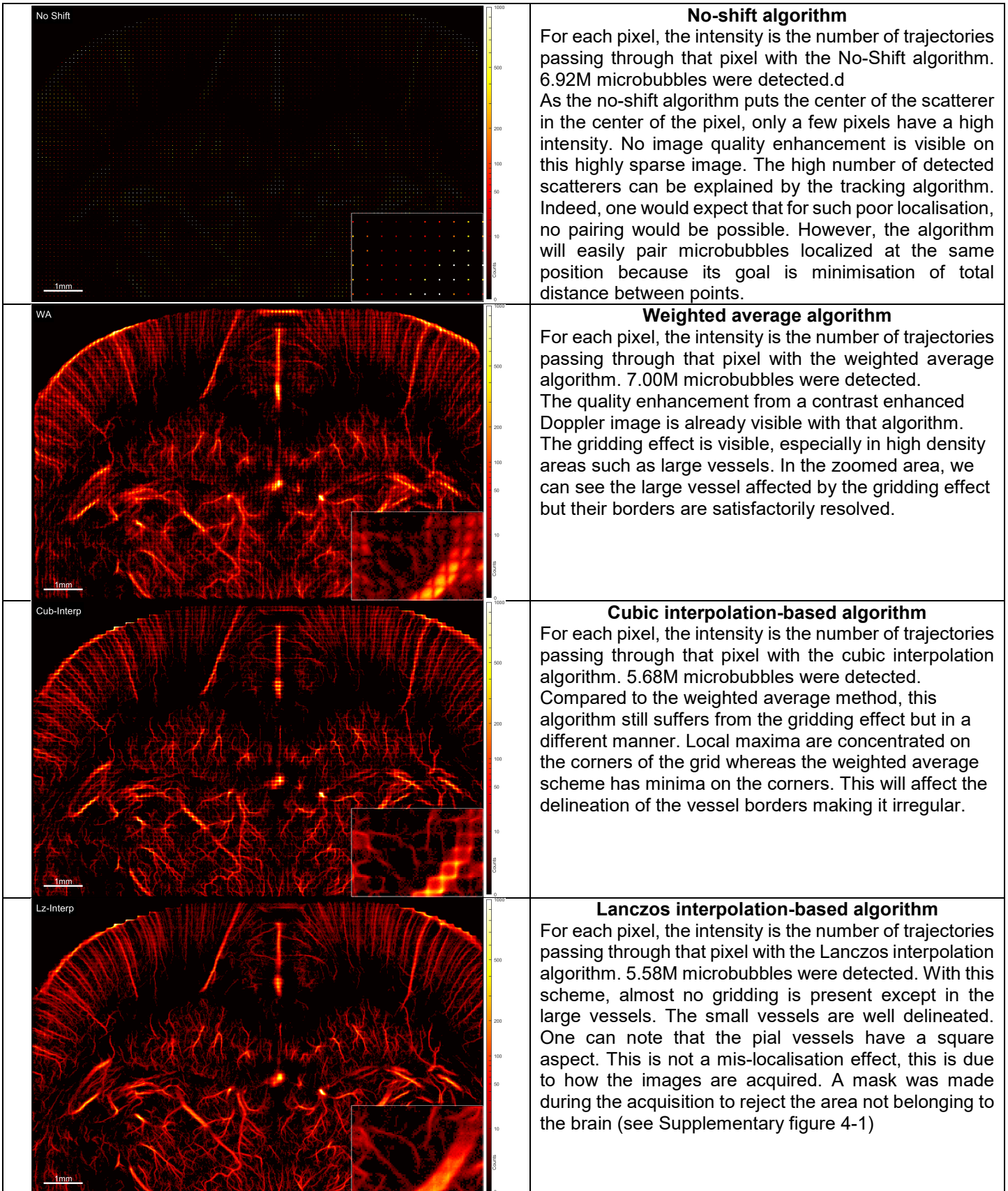
At the end of this chapter, we present the ULM rendering for the three *in vivo* additional datasets: “*in vivo* rat brain bolus”, “*in vivo* rat kidney”, and “*in vivo* mouse tumor”.

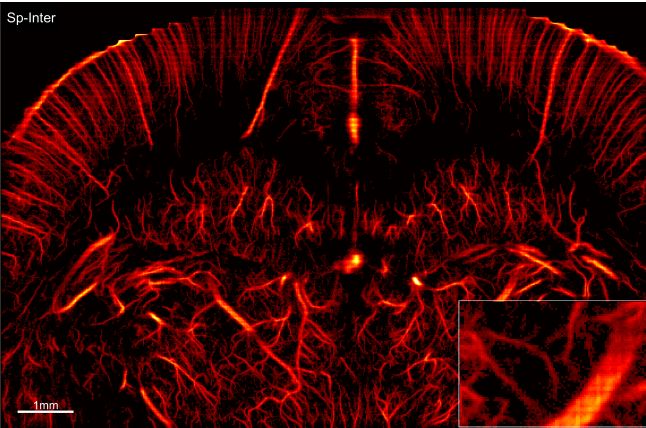
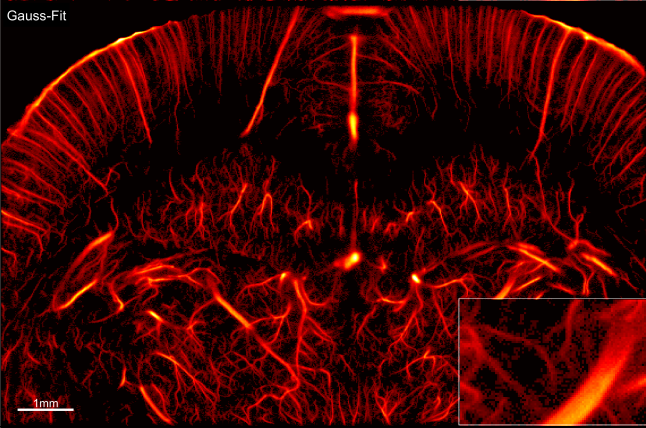
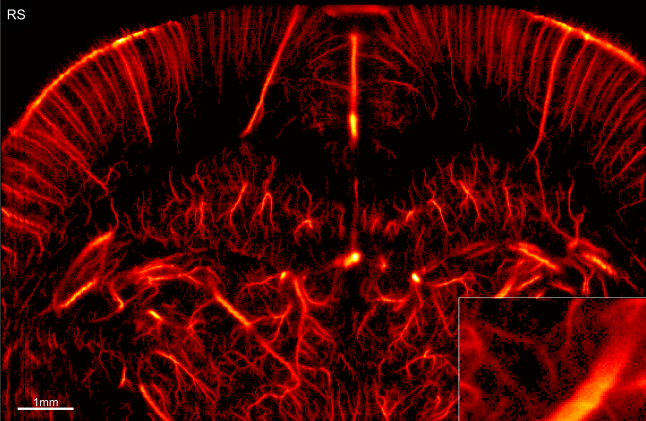
**Supplementary figure 4-1 Power Doppler rendering of *in vivo* rat brain dataset**



In the next 7 figures, the density renderings are obtained by counting the number of trajectories passing through each pixel of size  $\frac{\lambda}{10} \times \frac{\lambda}{10}$ . These trajectories are obtained by applying the 7 different localisation algorithms and then Kuhn-Munkres assignment-based tracking without interpolation of microbubbles' trajectories.

**Supplementary figure 4-2 Positions localized by all algorithms**

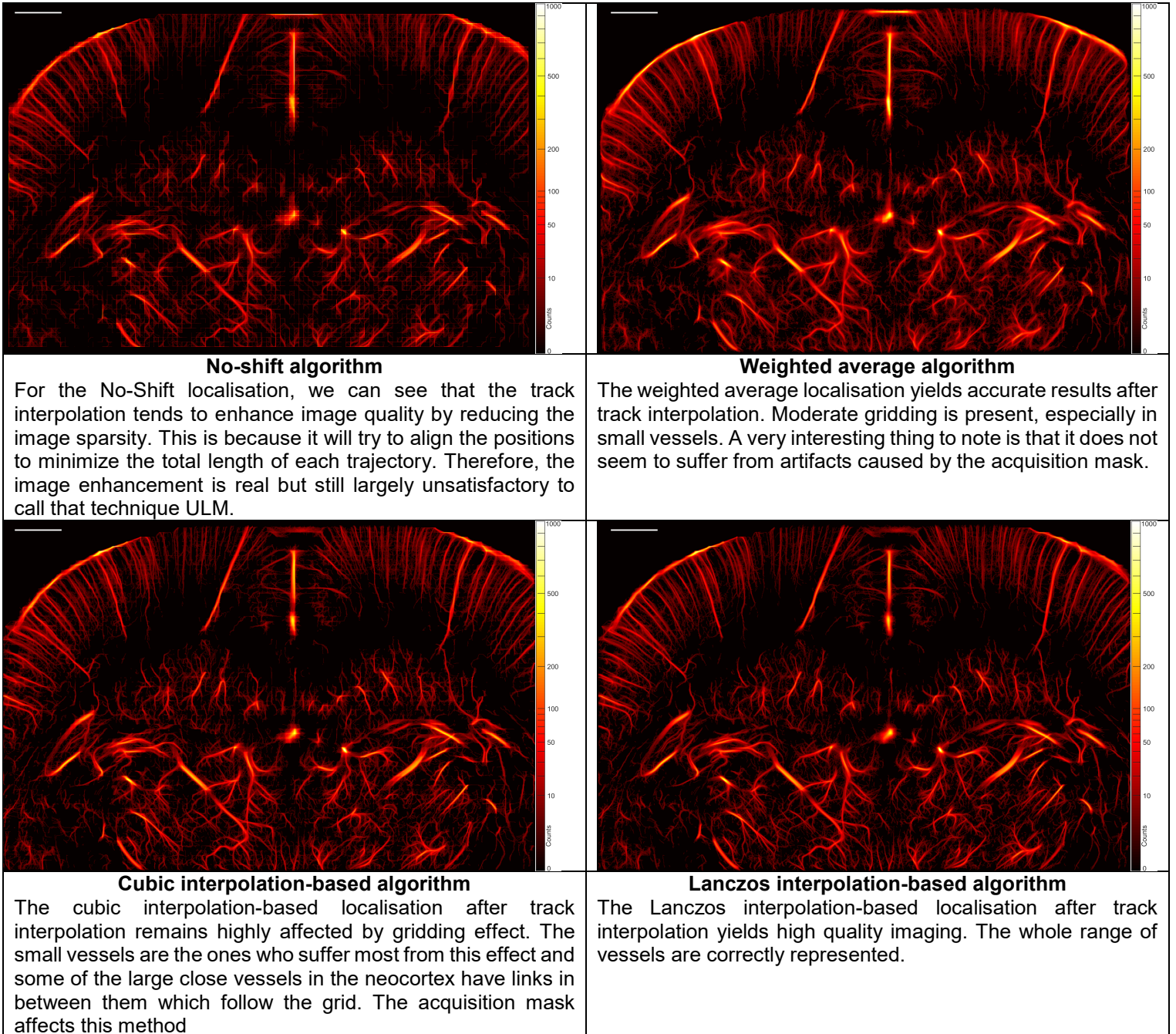


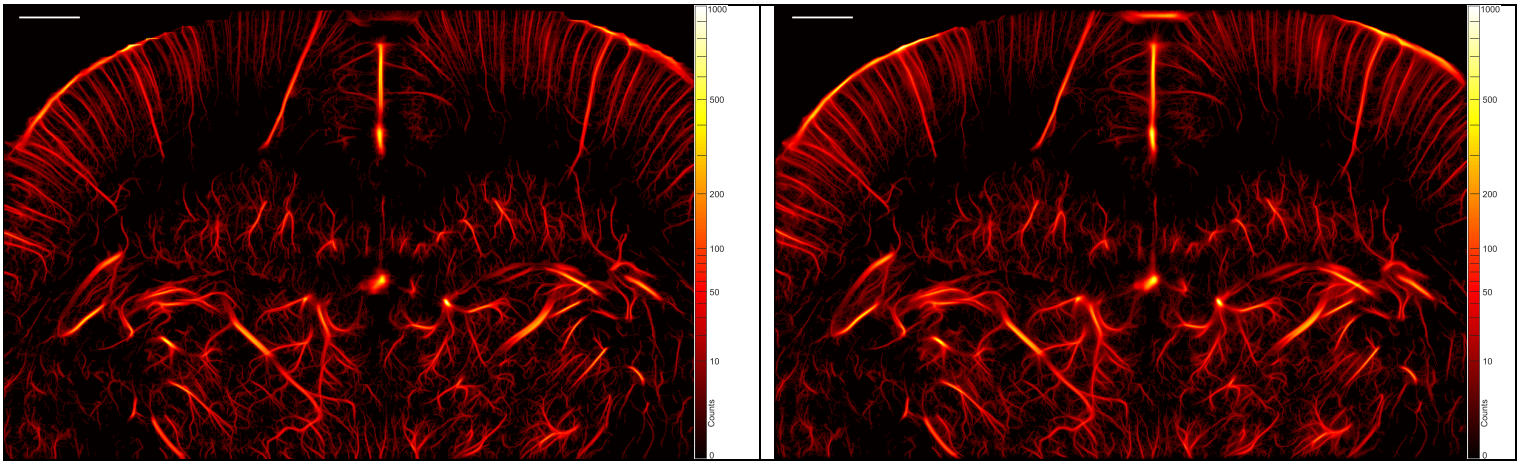
<p>Sp-Inter</p> 	<p><b>Spline interpolation-based algorithm</b></p> <p>For each pixel, the intensity is the number of trajectories passing through that pixel with the spline interpolation algorithm. 5.72M microbubbles were detected. The gridding effect is a bit more visible than in the Lanczos interpolation. The same effects on the pial vessels are visible.</p>
<p>Gauss-Fit</p> 	<p><b>Gaussian fitting algorithm</b></p> <p>For each pixel, the intensity is the number of trajectories passing through that pixel with the Gaussian fitting algorithm. 6.78M microbubbles were detected. Almost no gridding effect for this localisation implementation. We see that the delineation of vessels is not too sharp and the areas around vessel often contain many isolated pixels. Track interpolation will fully reconstruct the shape of these vessels as they are described by a very few numbers of microbubbles. This can be related to the high sensitivity of this algorithm. The pial vessels are also affected by the mask applied during the acquisition.</p>
<p>RS</p> 	<p><b>Radial symmetry-based algorithm</b></p> <p>For each pixel, the intensity is the number of trajectories passing through that pixel with the radial symmetry-based algorithm. 6.61M microbubbles were detected. Similar observations can be made than for the Gaussian fitting localisation algorithm. In this implementation, we seem to see a bit more isolated localisations than on the Gaussian fitting. The pial vessels are also affected by the acquisition mask and this tends to cause spikes towards the top of the brain.</p>



### Supplementary figure 4-3 Density rendering localisation algorithms with track interpolation

In the next 7 figures, the density renderings are obtained by counting the number of trajectories passing through each pixel of size  $\frac{\lambda}{10} \times \frac{\lambda}{10}$ . These trajectories are obtained by applying the 7 different localisation algorithms and then Kuhn-Munkres assignment-based tracking with a custom defined interpolation of microbubbles' trajectories. This will smooth the trajectories and restore a more natural curvature. A compression factor of 1/3 is applied.



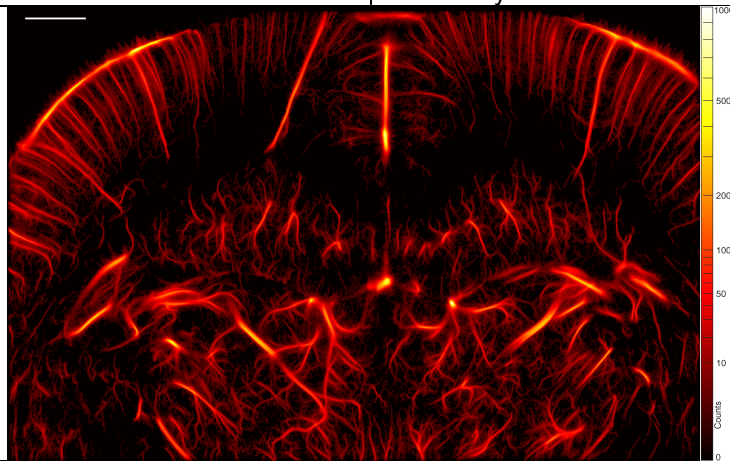


**Spline interpolation-based algorithm**

The spline interpolation-based localisation yields imaging close to the Lanczos based algorithm. It is also affected by the acquisition mask.

**Gaussian fitting algorithm**

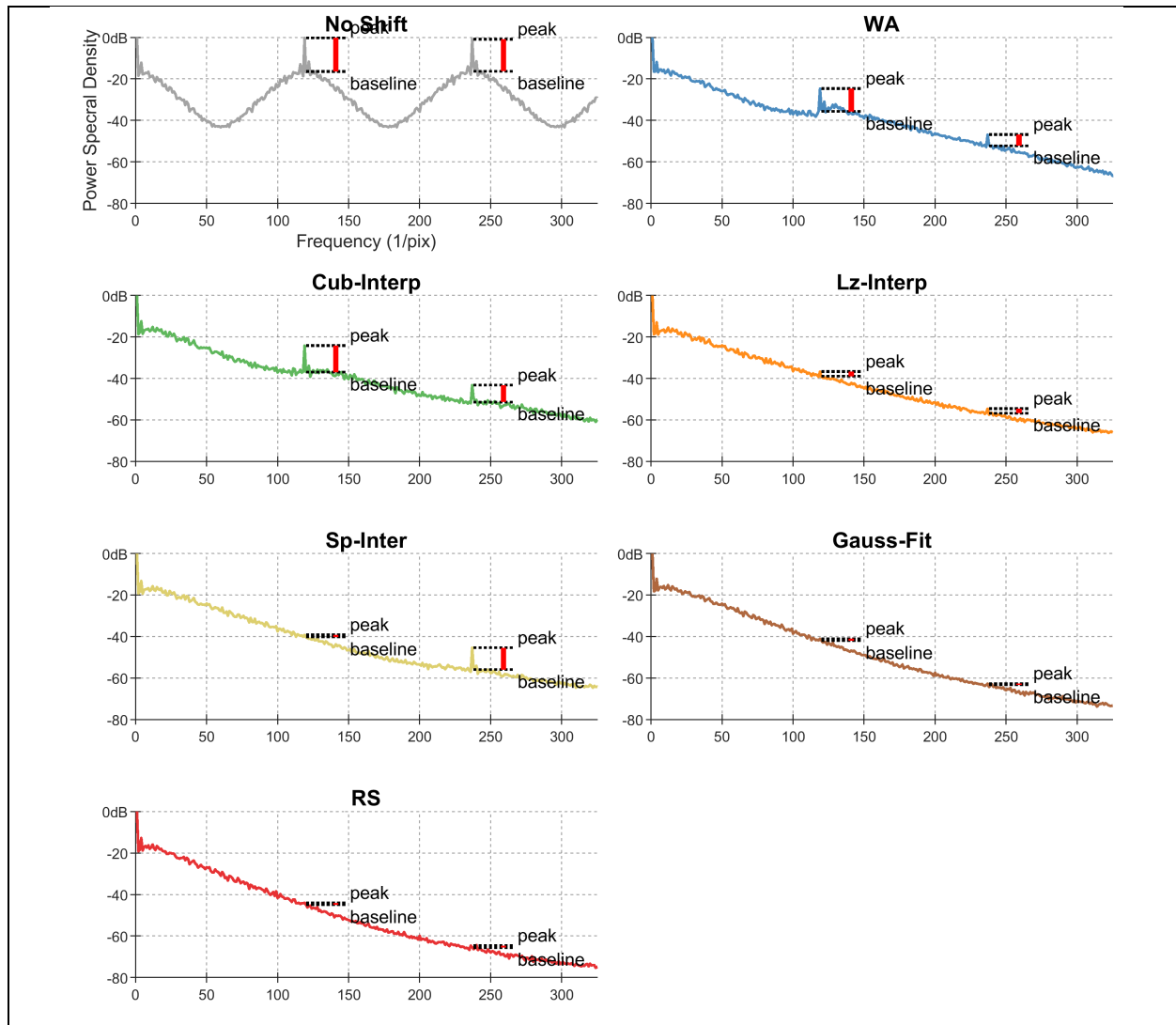
The Gaussian fitting-based localisation yields one of the best images in the algorithms analyzed. Almost no gridding can be perceived and the tracking algorithm seems to perform accurately



**Radial symmetry-based algorithm**

The radial symmetry yields the best image of the set if we don't consider the artifacts caused by the acquisition mask. It seems to have more of the smaller vessels compared to the Gaussian fitting-based algorithm, which we hypothesize to be vessels because of its high sensitivity index.

**Supplementary figure 4-4 Peak to peak Power Spectral Density analysis to devise a gridding index**

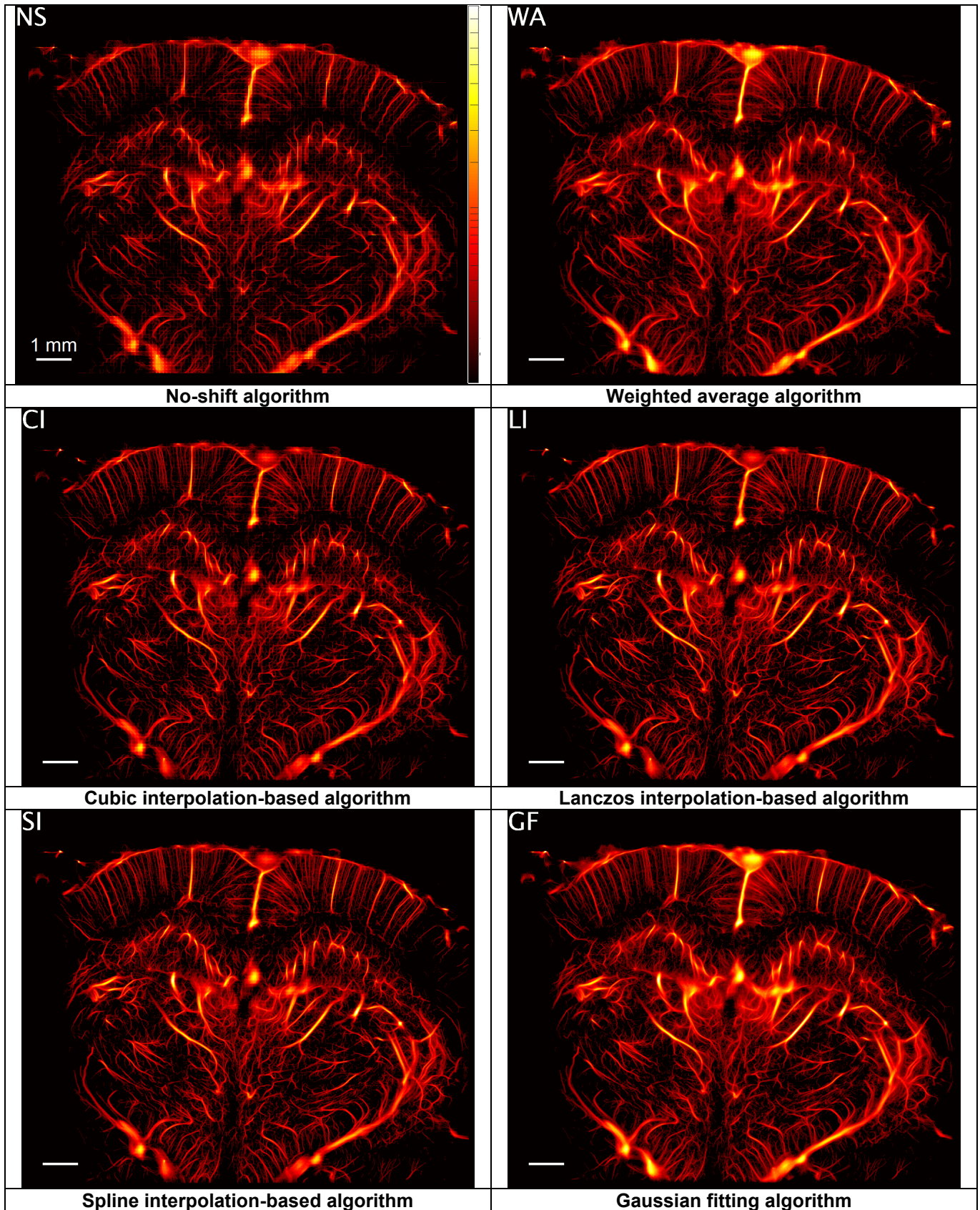


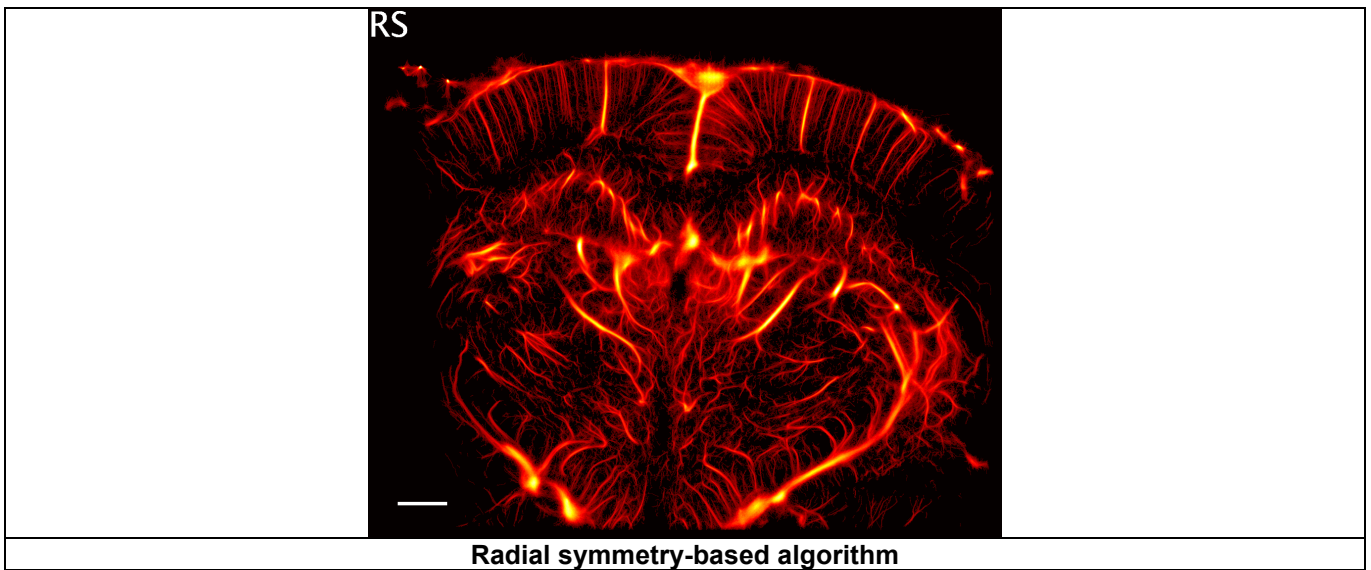
These graphs represent the Power Spectral Density of the signal present in the *in vivo* images post-localisation and tracking, but before track interpolation. The PSD is averaged along the z direction, leading to a lateral frequency dependency.

Two of the main peaks are chosen to calculate the value of the peak to baseline (the fundamental frequency and its first harmonic). By summing these two values, we obtain an index that will characterize the amount of gridding present in the images. We can see that both the Gaussian fitting (Gauss-Fit) and radial symmetry (RS) based algorithm do not present large peaks. The No-Shift on the other hand has a periodic Power Spectral density at frequency corresponding to the reconstruction grid scale. These values are calculated and presented in the main article.

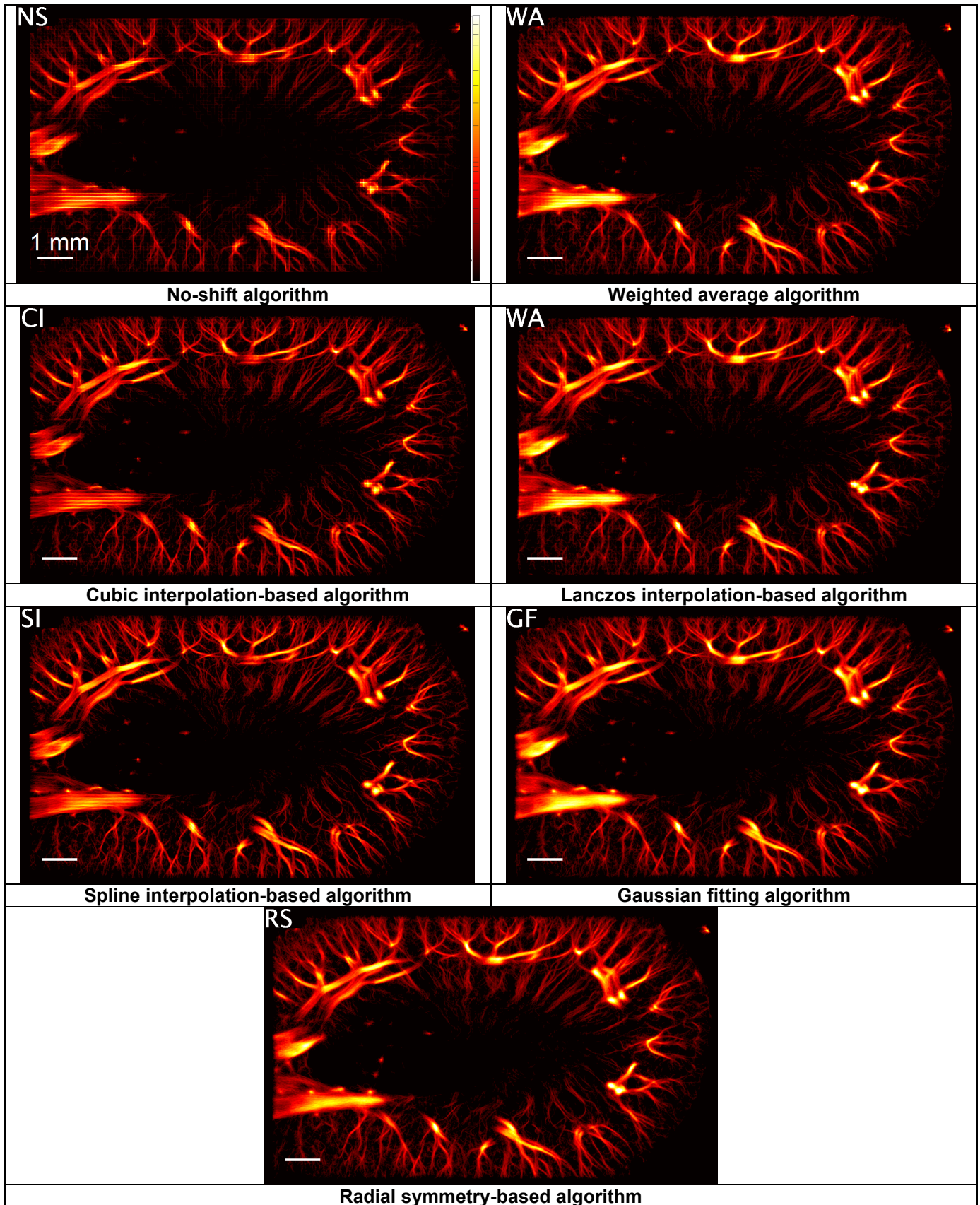


Supplementary figure 4-5 Density renderings for dataset “*in vivo* rat brain bolus”



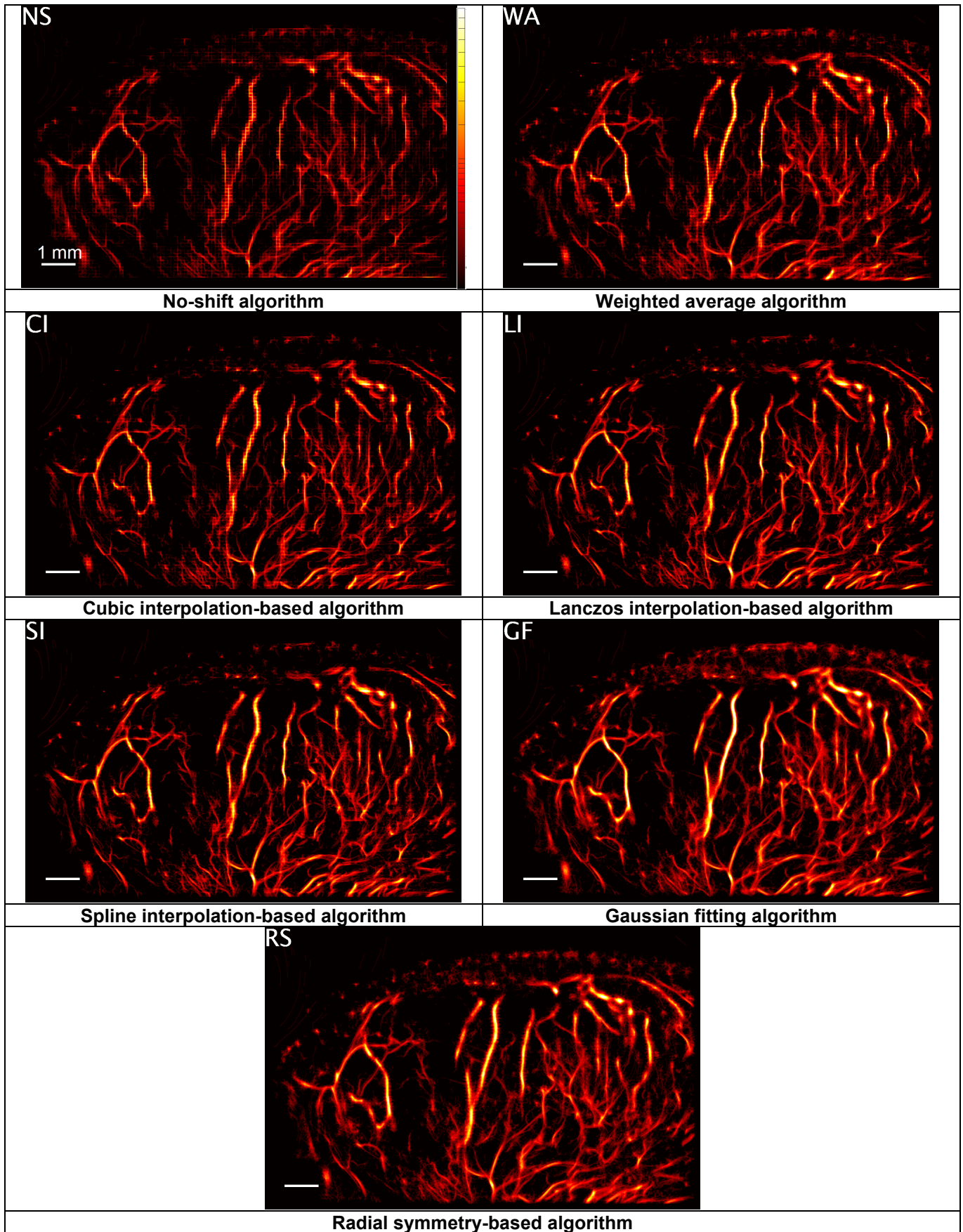


Supplementary figure 4-6 Density renderings for dataset “*in vivo* rat kidney”





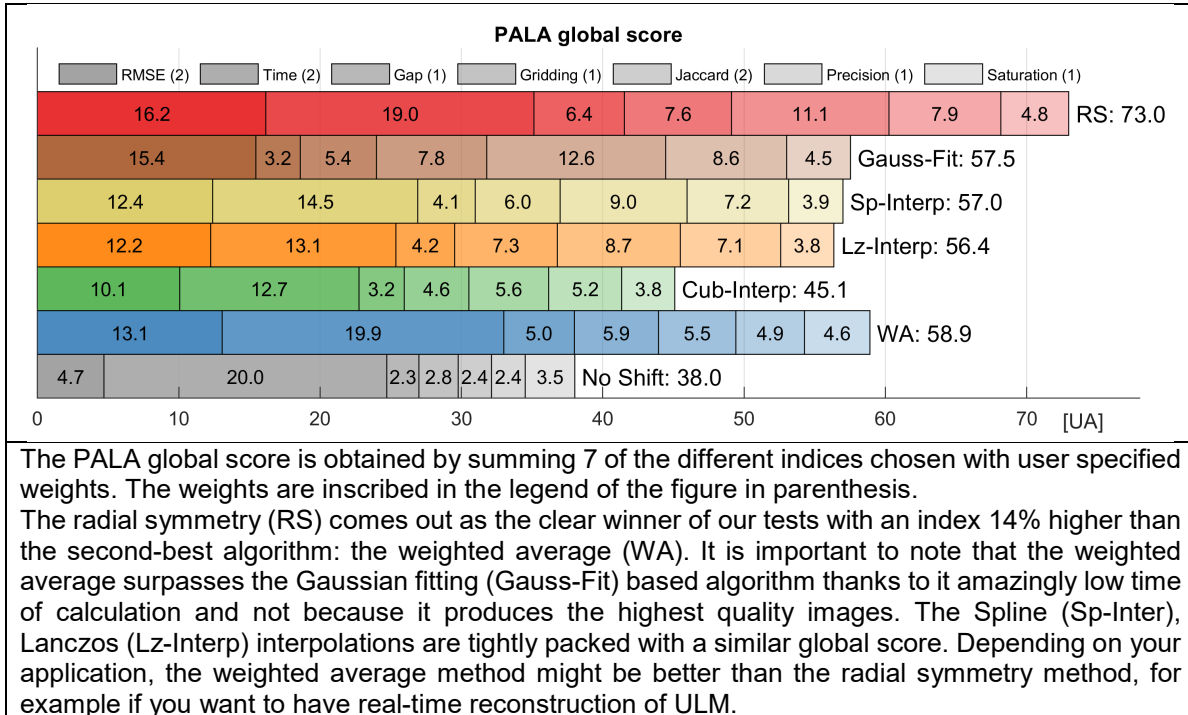
Supplementary figure 4-7 Density renderings for dataset “*in vivo* mouse tumor”



## Part 5 : Global Scoring

In this chapter, we present the **PALA global score** devised with our study and explain additional information such as how we simulated noise in the two first datasets, classical ULM rendering.

**Supplementary figure 5-1 PALA global score obtained for all the 7 algorithms tested**



**Supplementary table 5-2 Table with all the index values and score conversion**

		No-Shift	WA	Cub-Interp	Lz-Interp	Sp-Interp	Gauss-Fit	RS
<b>RMSE</b>	$\lambda$	0.38	0.17	0.25	0.19	0.19	0.11	0.10
	Score	23.5	65.4	50.3	61.2	62.0	77.2	80.8
<b>Processing Time</b>	S	1563	1598	8429	7627	5531	75640	1984
	Score	100.0	99.5	63.4	65.6	72.6	15.8	94.8
<b>Gap</b>	$\lambda$	0.77	0.50	0.68	0.58	0.59	0.46	0.36
	Score	22.9	50.1	32.1	41.6	40.6	53.9	64.1
<b>Gridding</b>	[AU]	21.7	12.2	16.2	8.2	12.0	6.6	7.3
	Score	27.7	59.4	45.9	72.5	60.1	78.1	75.8
<b>Jaccard</b>	Score	11.8	27.4	28.1	43.5	44.9	63.2	55.7
<b>Precision</b>	Score	24.0	48.6	51.7	71.1	71.8	85.9	79.0
<b>Saturation</b>	%	35.1%	46.3%	37.6%	37.8%	38.6%	45.1%	48.2%
	Score	35.1	46.3	37.6	37.8	38.6	45.1	48.2

This table presents the highest values in green and the lowest in red. One can note that the radial symmetry (RS) based algorithm comes first or close second for all indices.

**Supplementary table 5-3 Table of weights for different scoring scenarii**

	RMSE	Pro Time	Gap	Gridding	Jaccard	Precision	Saturation
<b>Non-Weighted</b>	1	1	1	1	1	1	1
<b>3D ULM</b>	2	3	1	2	1	1	2
<b>2D Scanning</b>	1	2	1	1	1	1	2
<b>Real-time</b>	1	3	1	1	1	1	1
<b>Low SNR</b>	2	0.5	2	2	2	2	1
<b>PALA global Score</b>	2	2	1	1	2	1	1

This table presents weighting scenarii to reflect experimental requirements: e.g. for real-time imaging, the processing time is primordial.  
The final scores are then computed by a weighted average of indices presented in the next table.

**Supplementary table 5-4 Table weighted average scores for the 6 scenarii**

	No-Shift	WA	Cub-Interp	Lz-Interp	Sp-Interp	Gauss-Fit	RS
<b>Non-Weighted</b>	35.0	56.7	44.2	56.2	55.8	59.9	71.2
<b>3D ULM</b>	44.3	63.9	47.5	58.0	58.0	54.2	74.4
<b>2D Scanning</b>	42.2	60.3	45.6	55.2	55.7	53.3	71.3
<b>Real-time</b>	49.4	66.2	48.4	58.3	59.5	50.1	76.5
<b>Low SNR</b>	26.5	52.0	42.2	56.6	55.1	66.9	70.1
<b>PALA global Score</b>	<b>38.0</b>	<b>58.9</b>	<b>45.1</b>	<b>56.4</b>	<b>57.0</b>	<b>57.5</b>	<b>73.0</b>

This table presents the weighted average scores for each scenarii over 100.

## Part 6 : Tables

Supplementary figure 6-1 state of the art of localisation methods for ULM

Localisation method	Related articles	Comments
Gaussian fitting	(Ackermann and Schmitz, 2016; Luke et al., 2016; O'Reilly and Hynynen, 2013; Song et al., 2018b) <sup>19-22</sup>	Usually, these works use a Gaussian convolved PSF rather than a Gaussian fitting with an optimizer.
Weighted average based	(Christensen-Jeffries et al., 2015; Hansen et al., 2016; Heiles et al., 2019; Lin et al., 2017; Song et al., 2018a; Soulioti et al., 2018; Viessmann et al., 2013; Zhang et al., 2018; Zhu et al., 2019) <sup>10,13,15,23-28</sup>	Except for Heiles et al 2019, these work on data beamformed with pixels of sizes below the wavelength or data beamformed with commercial scanners which might affect PSF shape and full width at half maximum.
Lanczos based interpolation and Gaussian fitting	(Errico et al, 2015) <sup>9</sup>	
Spline based interpolation	(Huang et al., 2020; Song et al., 2018b) <sup>22,29</sup>	
Linear based interpolation	(Song et al., 2018a) <sup>15</sup>	On top of the linear-based interpolation, this paper convolves with a Gaussian profile.
Cubic based interpolation	(Song et al., 2018b) <sup>22</sup>	This paper is a comparison of algorithms
Radial symmetry	(Parthasarathy, 2012) <sup>30</sup> (optic super-resolution only)	
RF-based	(Brown et al., 2019; Christensen-Jeffries et al., 2017a, 2017b; Desailly et al., 2013, 2015) <sup>16,17,31-33</sup>	These papers are based on radiofrequency data before beamforming. In particular, the papers from the team at Imperial College/Kings College London use the onset of the Hilbert transform of the RF signals.

Supplementary figure 6-2 Summary of the media simulated

Structure	Diameters	Maximal velocities
A pseudo double helix	$3\lambda$ $2\lambda$	$v_{max} = 3\lambda/frame$ $v_{max} = 2\lambda/frame$
A curved tube with a constant diameter and a horseshoe pattern	$0.5\lambda$	$v_{max} = 0.5\lambda/frame$
3 curved tubes with 3 different diameters	$0.2\lambda$	$v_{max} = 0.2\lambda/frame$
	$0.1\lambda$	$v_{max} = 0.1\lambda/frame$
	$0.05\lambda$	$v_{max} = 0.05\lambda/frame$
4 spreading tubes with a constant diameter	$0.1\lambda$	$v_{max} = 0.9\lambda/frame$
A watermark comprised of the word ULM that does not serve other purposes but identification	$0.1\lambda$	$v_{max} = 0.4\lambda/frame$

Supplementary figure 6-3 Summary of the metrics

Number	Measurement	Dataset		
		In silico PSF	In silico flow	In vivo
1	Lateral error	X	X	

2	Axial error	X	X	
3	RMSE	X	X	
4	True Positive		X	
5	False Negative		X	
6	False Positive		X	
7	Gap		X	
8	Number of detections			X
9	Saturation			X
10	Gridding Index			X
11	Processing time			X

Supplementary figure 6-4 Conversion of metrics to score

Metric	Unit	Conversion	Score Range
RMSE	Wavelengths	$Score(x_i) = 100 * (1 - \frac{x_i}{0.5\lambda})$	0 $\lambda$ : 100/100 0.5 $\lambda$ : 0/100
Processing Time	Seconds	$Score(x_i) = 100 * (1 - 0.5 * \log_{10}(\frac{x_i}{\min(x)}))$	$\min(x)$ : 100/100 100 * $\min(x)$ : 0/100
Separation Index	Wavelengths	$Score(x_i) = 100 * (1 - \frac{x_i}{1\lambda})$	0 $\lambda$ : 100/100 1 $\lambda$ : 0/100
Gridding Index	dB	$Score(x_i) = 100 * (1 - \frac{x_i}{30})$	0 dB : 100/100 30 dB : 0/100
Jaccard	[%]	$Score(x_i) = x_i$	100 % : 100/100 0 % : 0/100
Precision	[%]	$Score(x_i) = x_i$	100 % : 100/100 0 % : 0/100
Saturation	[%]	$Score(x_i) = x_i$	100 % : 100/100 0 % : 0/100