THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Resampling Diffusion Tensor Images

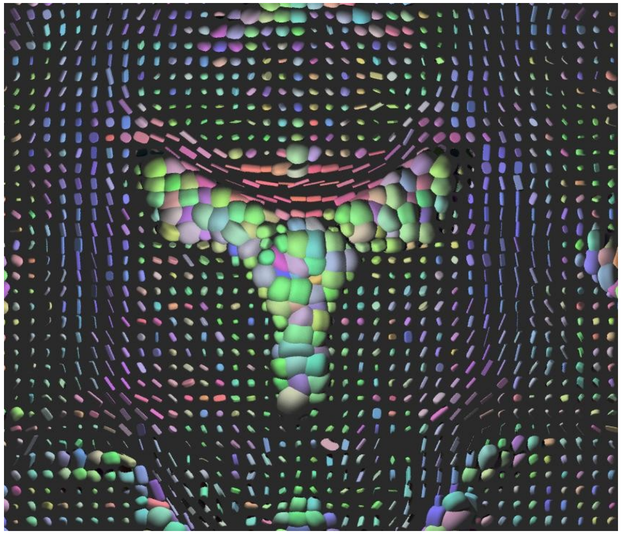
François Budin

Neuro Image Research and Analysis Laboratories

Overview

- Diffusion Tensor Images
- Resampling
  - Transformation
  - Interpolation
  - Structure preservation
- Results
- Conclusion

Diffusion Tensor Images



- Computed from a Diffusion Weighted Image (DWI)
- Represents the diffusion of water molecules in the brain
- Allows to visualize the structure of the tissues
- Each pixel contains a positive semi-definite symmetric matrix.
- Can be represented as an ellipsoid or a superquadric
  - The eigenvectors represent the orientation of the shape
  - The eigenvalues represent the size along each axis

Resampling: Why?

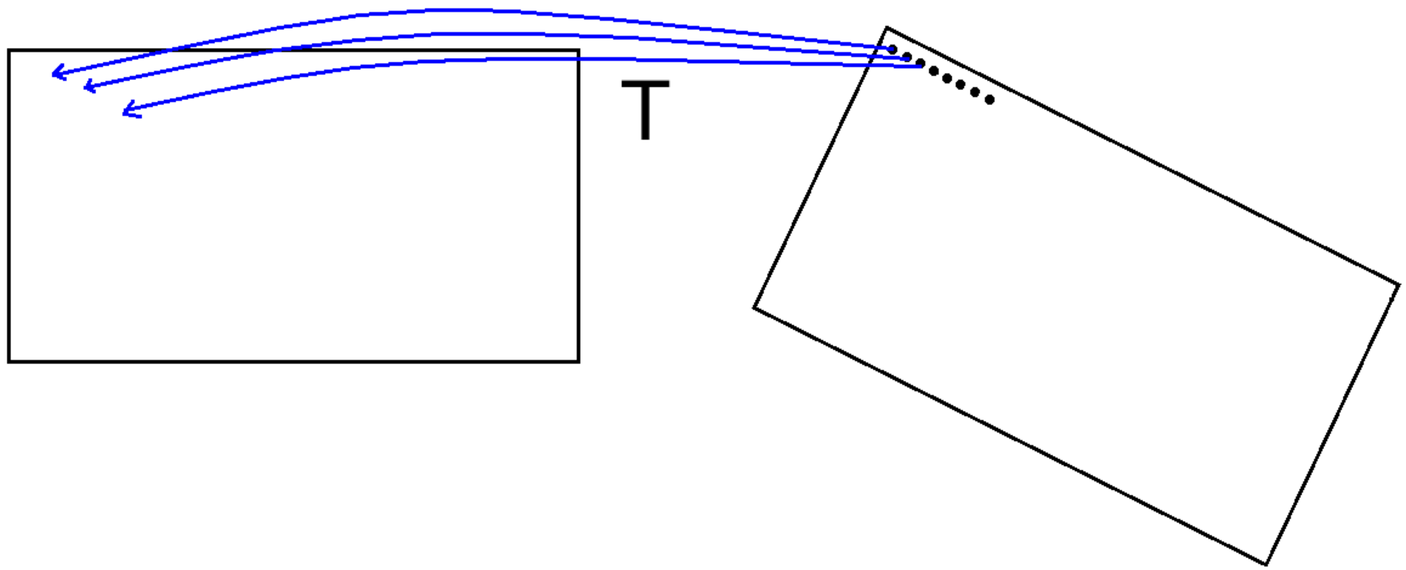
- Modify the property of the image:
  - Size
  - Orientation
  - Spacing
- Allows for the comparison between 2 images which are not initially aligned

Resampling: How ?

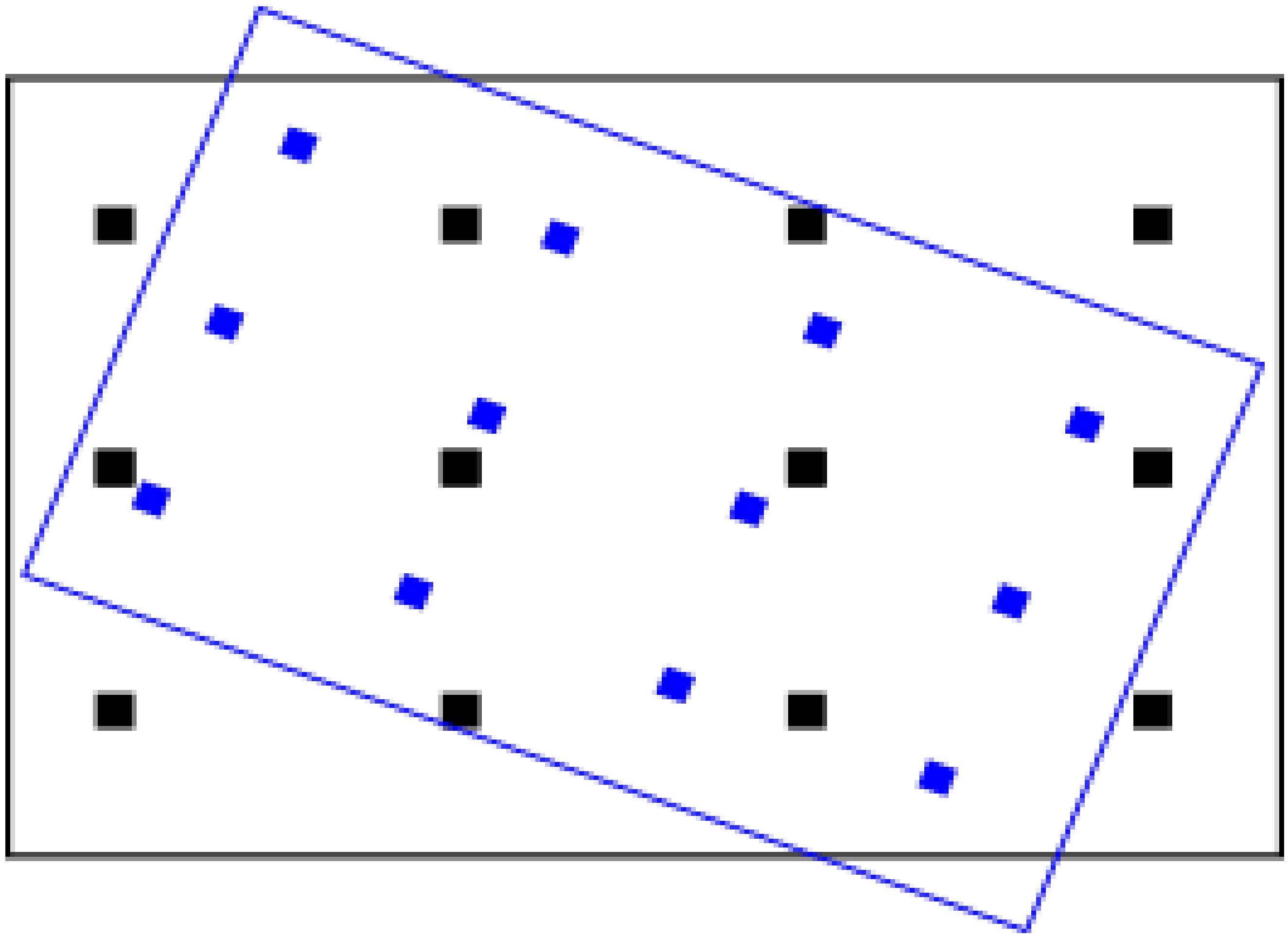
Can be separated in 2 steps:

- Transformation
  - Translation
  - Rotation
  - Affine transformation
  - Non-rigid transformation
- Interpolation
  - Nearest Neighbor
  - Linear

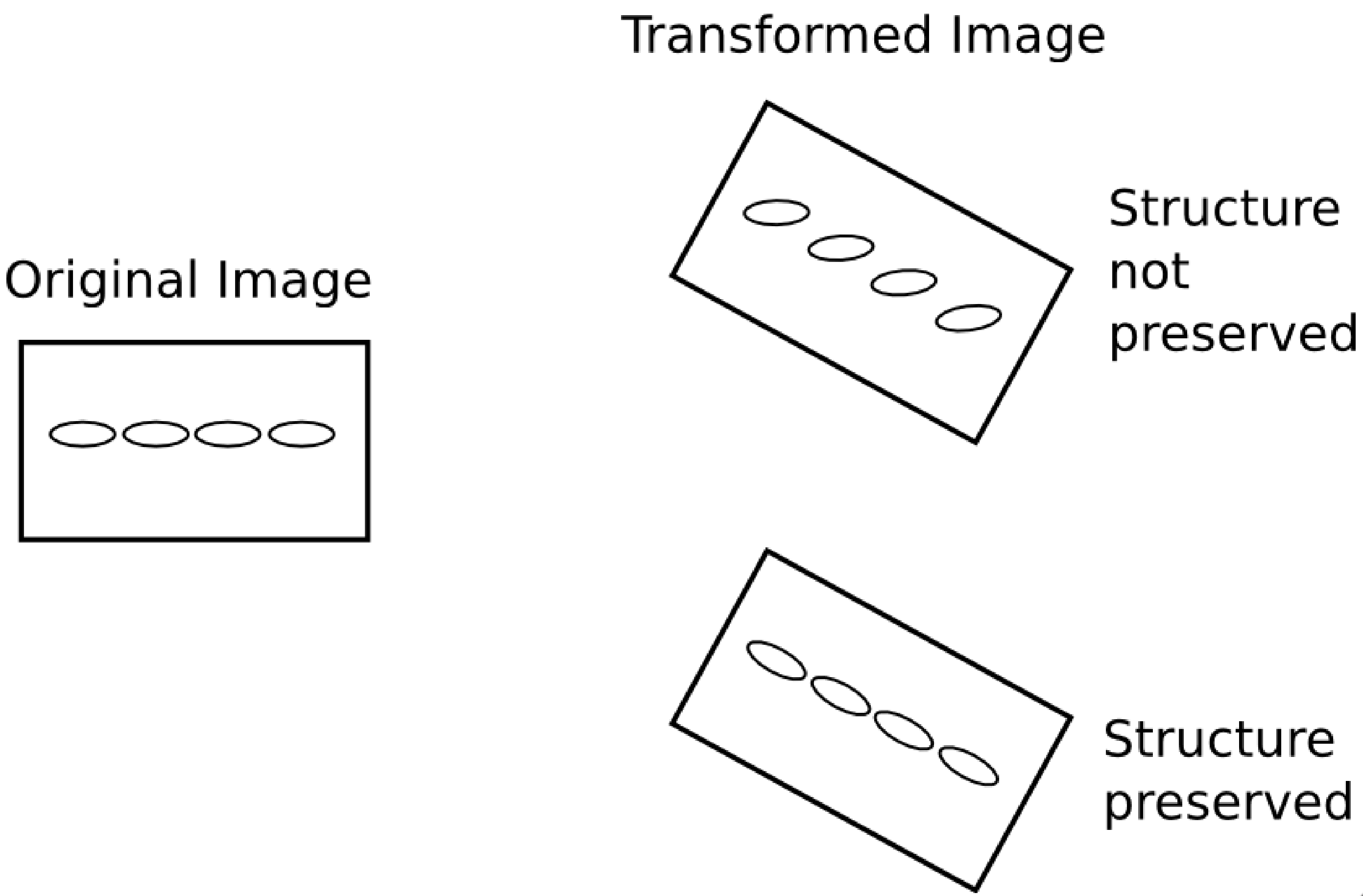
Transforming an image



Interpolation



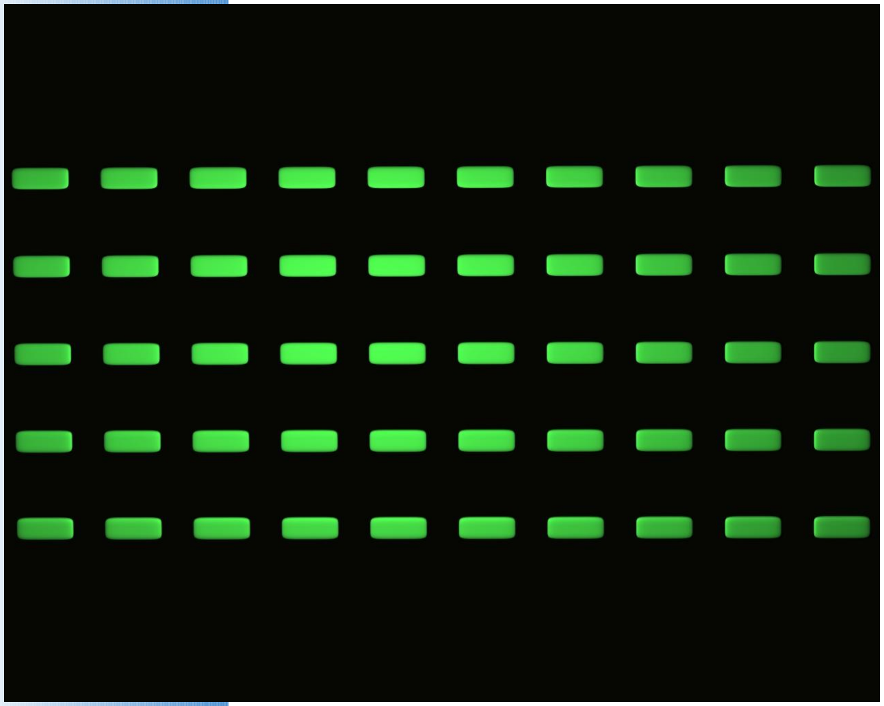
Structure preservation (1)

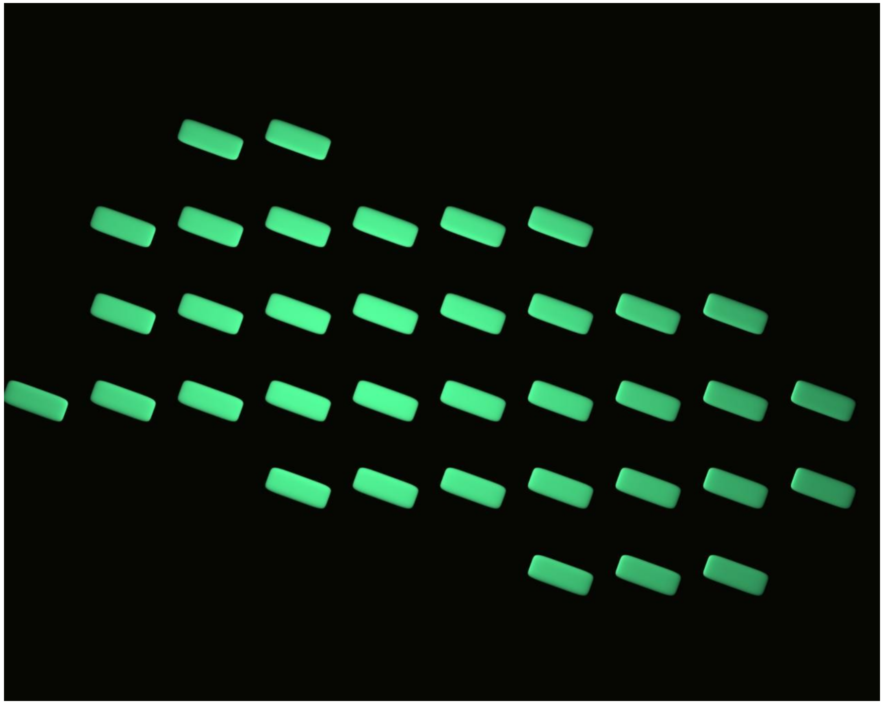


Structure preservation (2)

- A transformation has to be applied to the tensors.
- The transformation should not change the size of the tensors (the tissue's properties are not modified).
- In the case of a rotation, the same transformation is applied to transform the points and to transform the tensors.
- In the case of a more complex transformation, the rotation to apply has to be found.

Rotation





Affine Transformation

2 methods[1]:

- Finite Strain
- Preservation of Principal Direction

1 - D.C. Alexander, C. Pierpaoli, P.J. Basser, J.C. Gee. Spatial Transformations of Diffusion Tensor Magnetic Resonance Images, IEEE Transactions on medical imaging, vol. 20, No. 11, November 2001

Finite Strain (1)

- The rotation matrix is extracted from the transformation matrix:

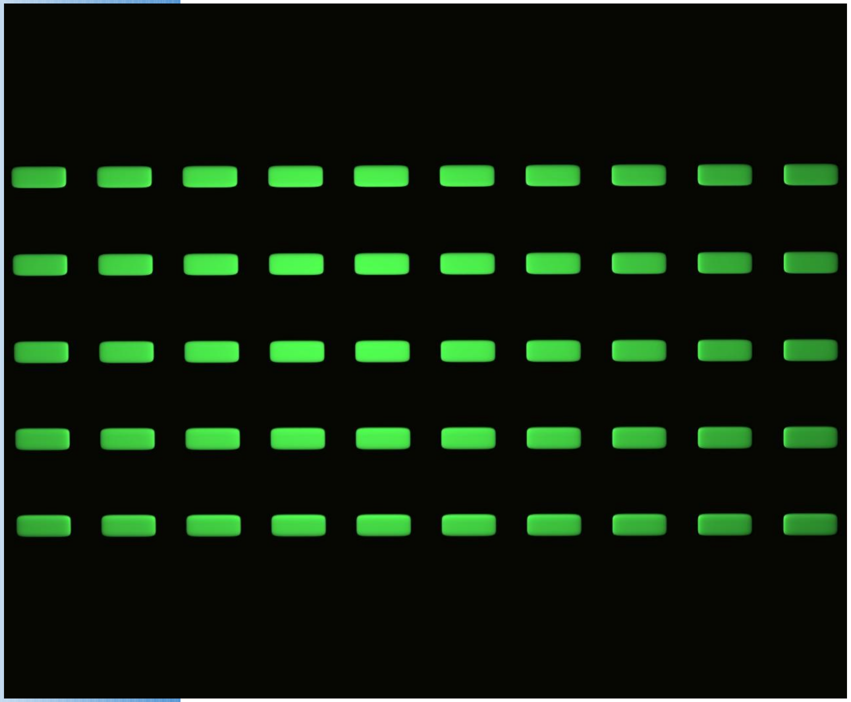
$$F=UR$$

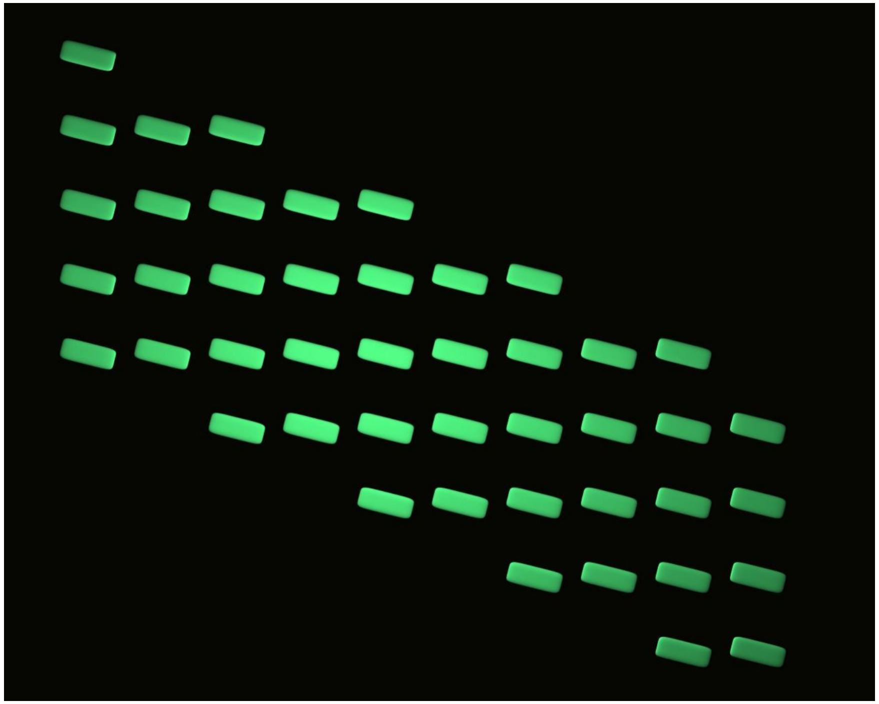
$$R=(FF^{T})^{(-1/2)}F$$

- The matrix is computed only once for the while image

Finite Strain (2)

- With some transformations (ie: shearing), the structure is not well preserved

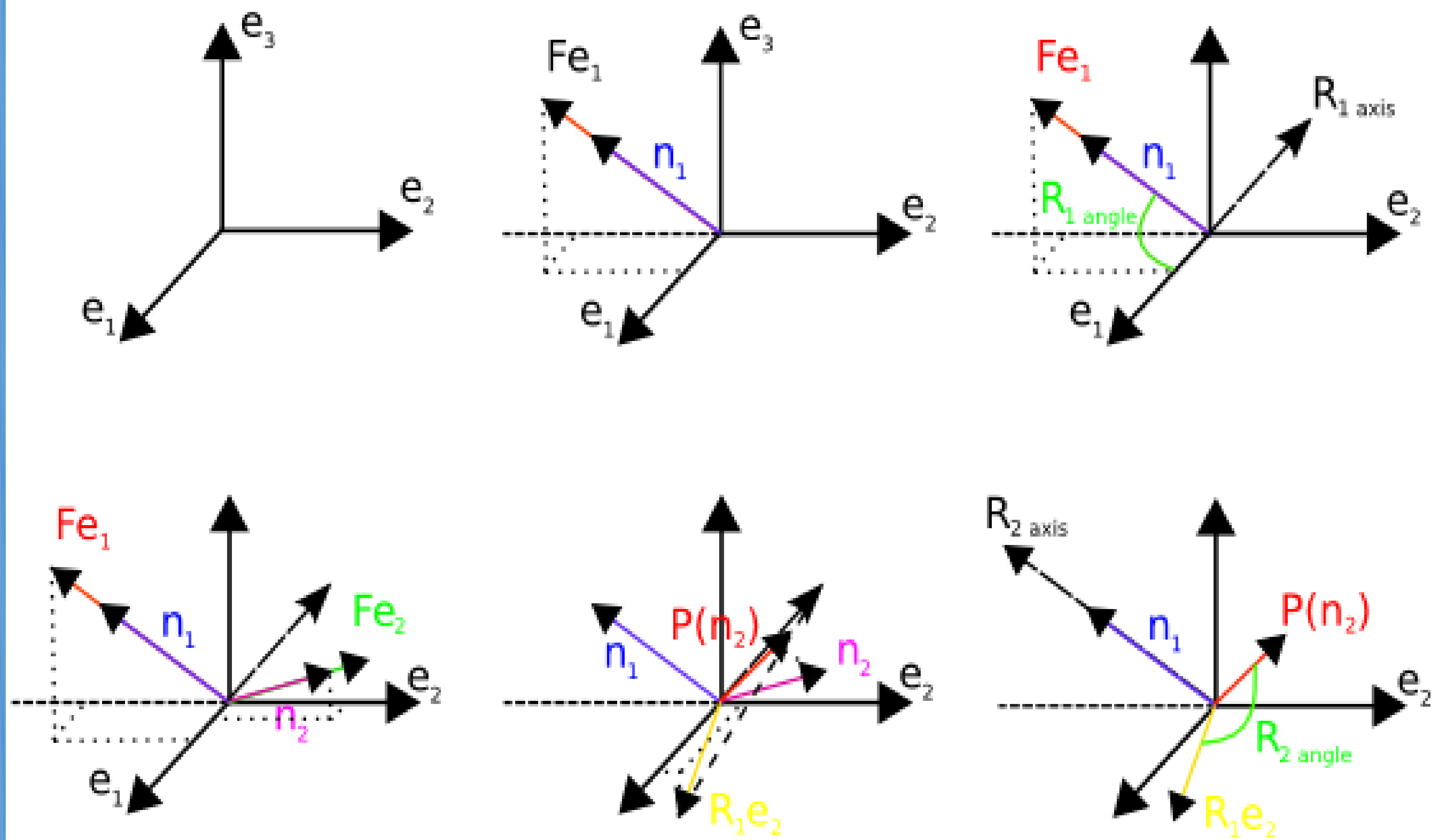




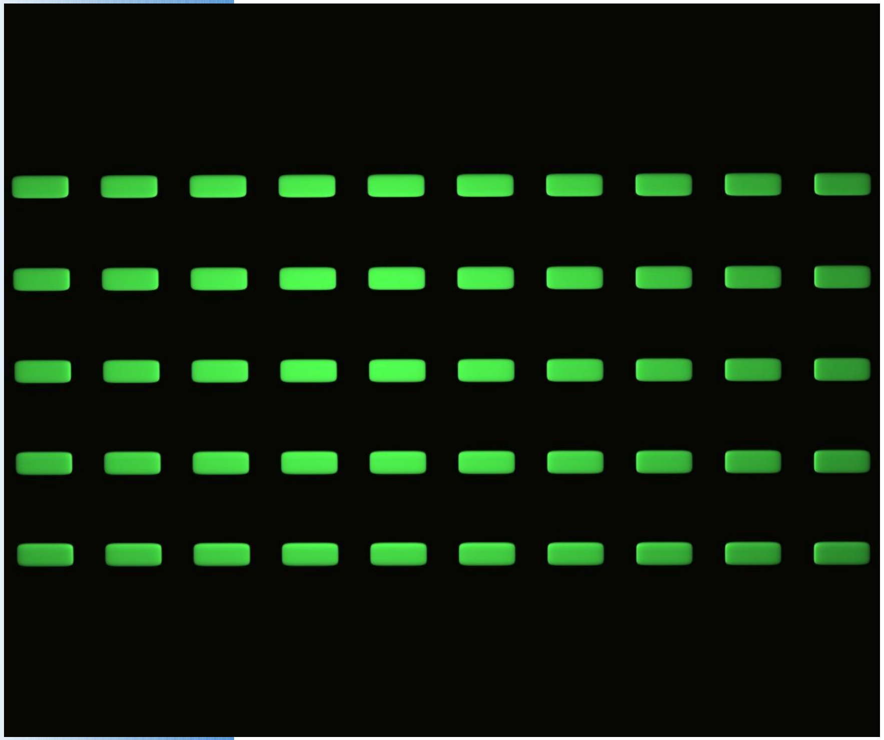
Preservation of Principal Direction (1)

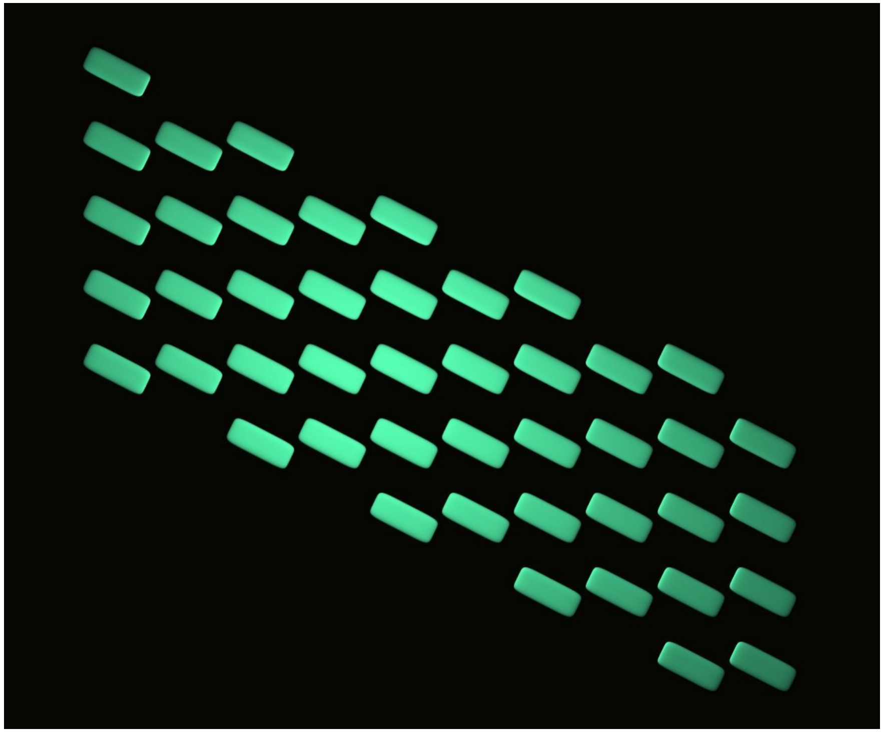
- The transformation applied to a tensor depends on its original orientation.
- The principal direction of a tensor is given by the eigenvector corresponding to its largest eigenvalue.

Preservation of Principal Direction (2)



Preservation of Principal Direction (3)



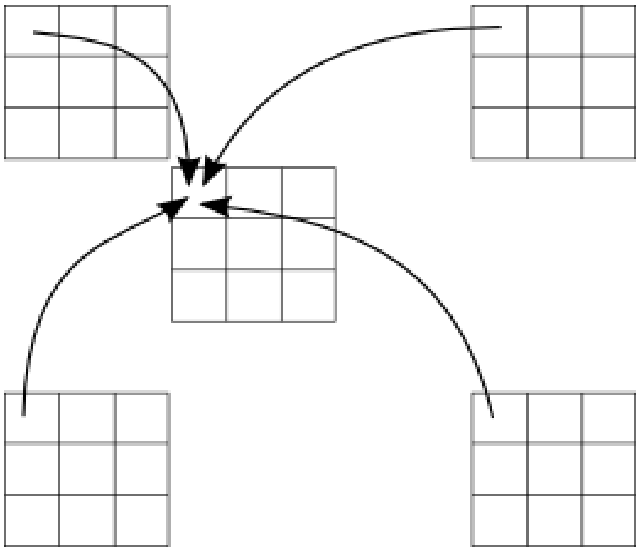


Transformations of higher order

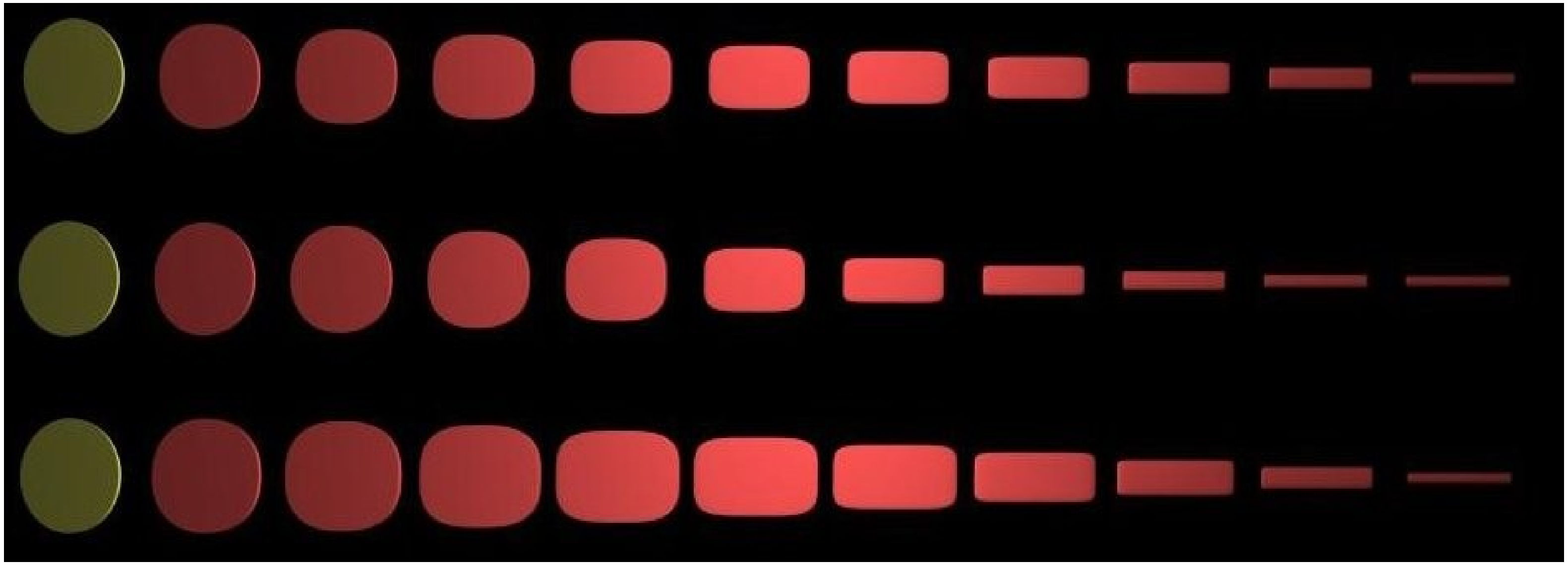
- Any transformation can be expressed as a deformation field $u(x)$ such that at each point of the image $T(x)=x+u(x)$
- If T is an affine transform: $T(x)=Fx+t$
- Deriving both expressions with respect to x gives $F=I+J_u$ where J_u is the Jacobian of u
- This gives a local affine model $F(x)=I+J_u$

Interpolation

- 2 methods have been implemented:
  - Nearest neighbor
  - Component-wise: each component of the matrix is individually interpolated
- Problem: The resulting matrix does not always belong to the tensor space



Component-wise interpolation



From top to bottom: linear interpolation, Bspline interpolation and Windowed sinc interpolation (Welch)

Tensor Space

- The tensor space is not a vector space but a convex half-cone in the vector space of matrices.
- Some operations are not stable in that space.
- 2 possibilities:
  - To bring back an interpolated matrix into this space, some corrections can be applied.
  - Log-Euclidean Framework

Corrections

- Set negative eigenvalues to zero.
- Set negative eigenvalues to their absolute value.
- Set the matrix representing the tensor to the nearest symmetric positive semi-definite matrix.
  X is the nearest symmetric positive semi-definite matrix to A in the Frobenius norm, B is the symmetric part of A and H is the symmetric polar factor of B.

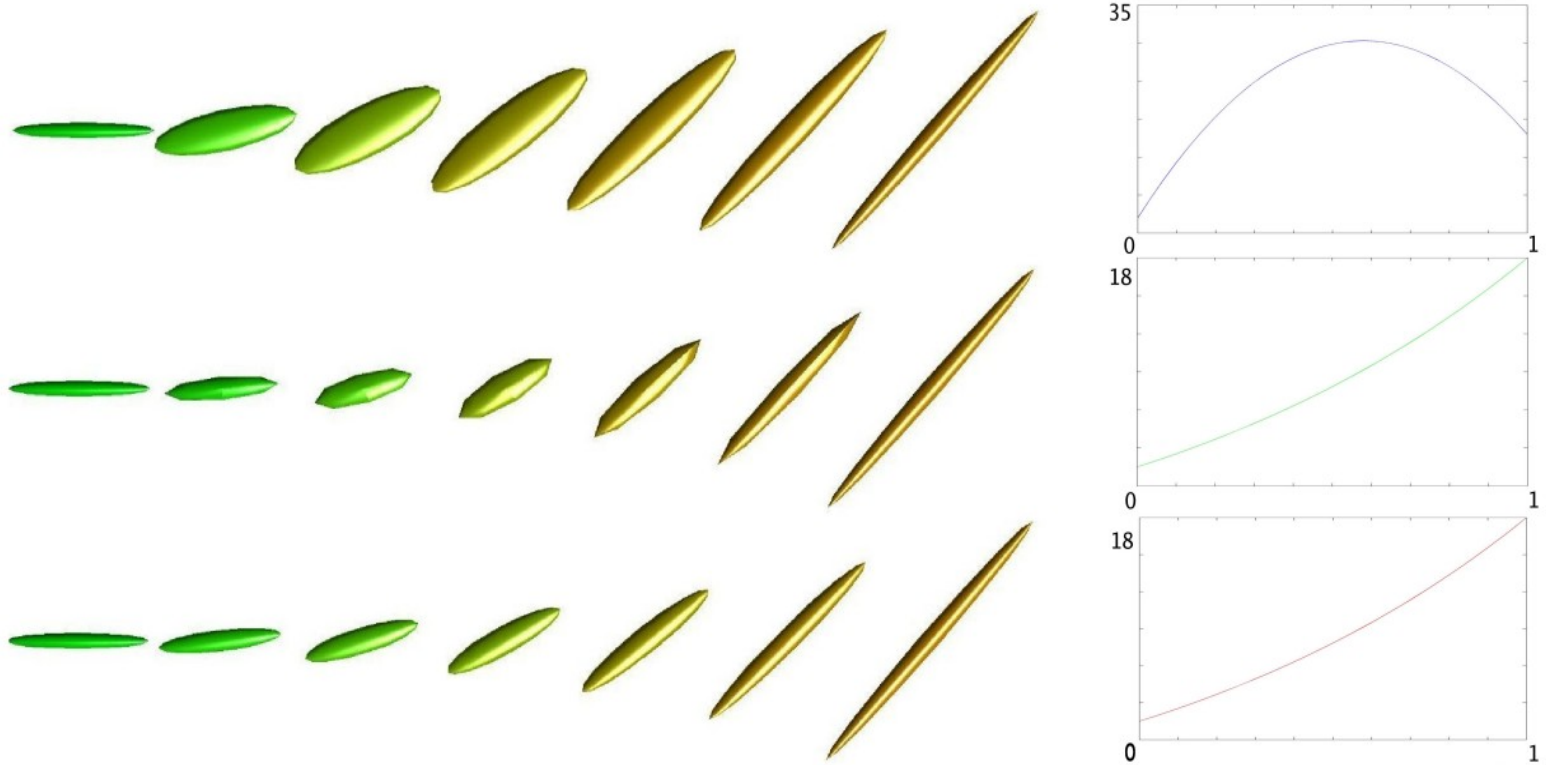
$$X = (B + H)/2$$
$$B = (A + A^T)/2$$
$$H = \sqrt{(B^T \cdot B)}$$

Correct computation

- The Riemannian framework
  - Complex
  - Slow
- The Log-Euclidean framework
  - Compute the logarithm of the tensors
  - Interpolate
  - Compute the exponential of the tensors

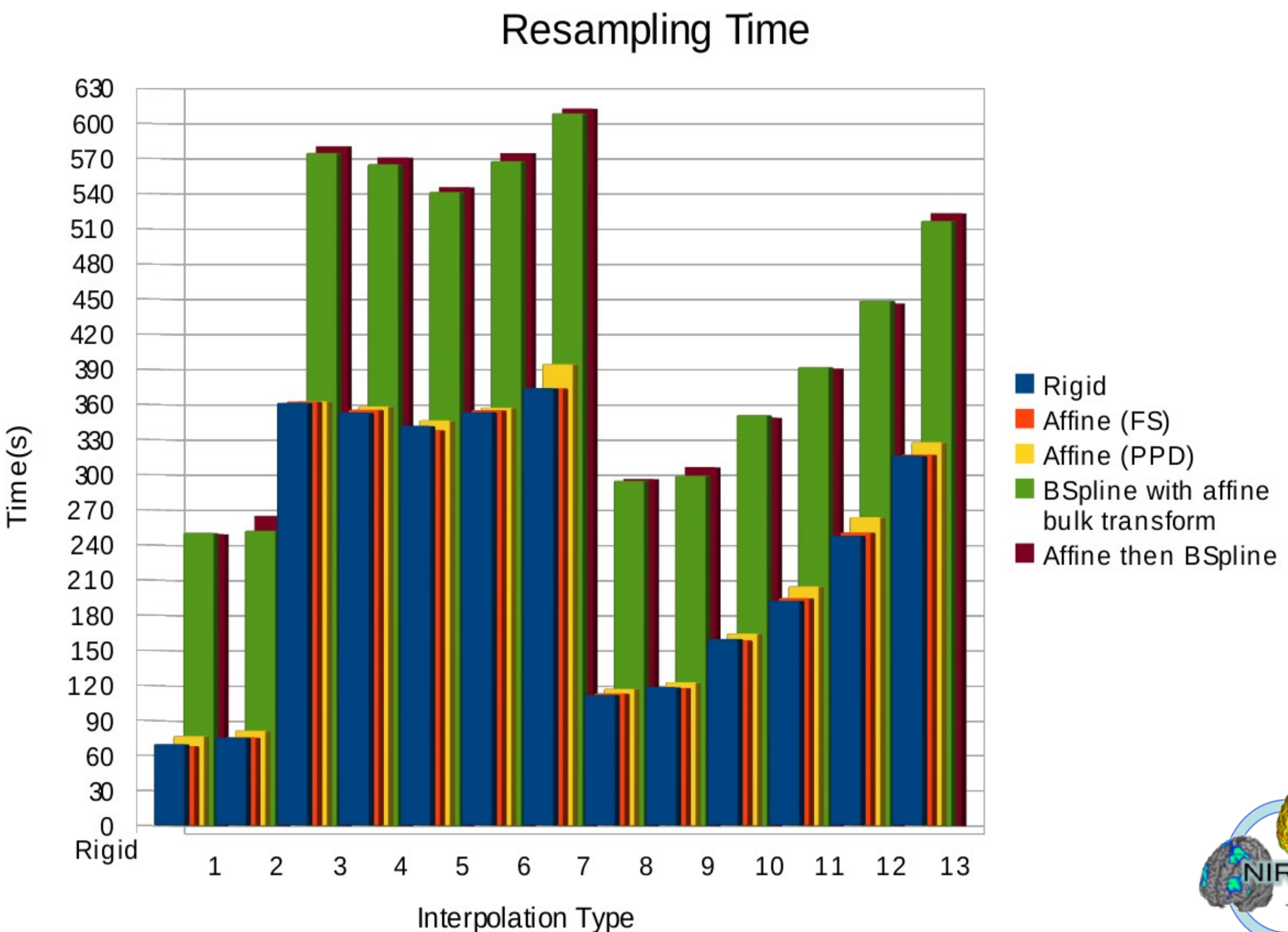
Log-Euclidean Framework

Top to bottom: Linear component-wise interpolation, affine-invariant interpolation, log-Euclidean interpolation



Log-Euclidean Metrics for Fast and Simple Calculus on Diffusion Tensors. Vincent Arsigny, Pierre Fillard, Xavier Pennec, Nicholas Ayache. Magnetic Resonance in Medicine, Vol. 56, No. 2. (2006), pp. 411-421

Processing time

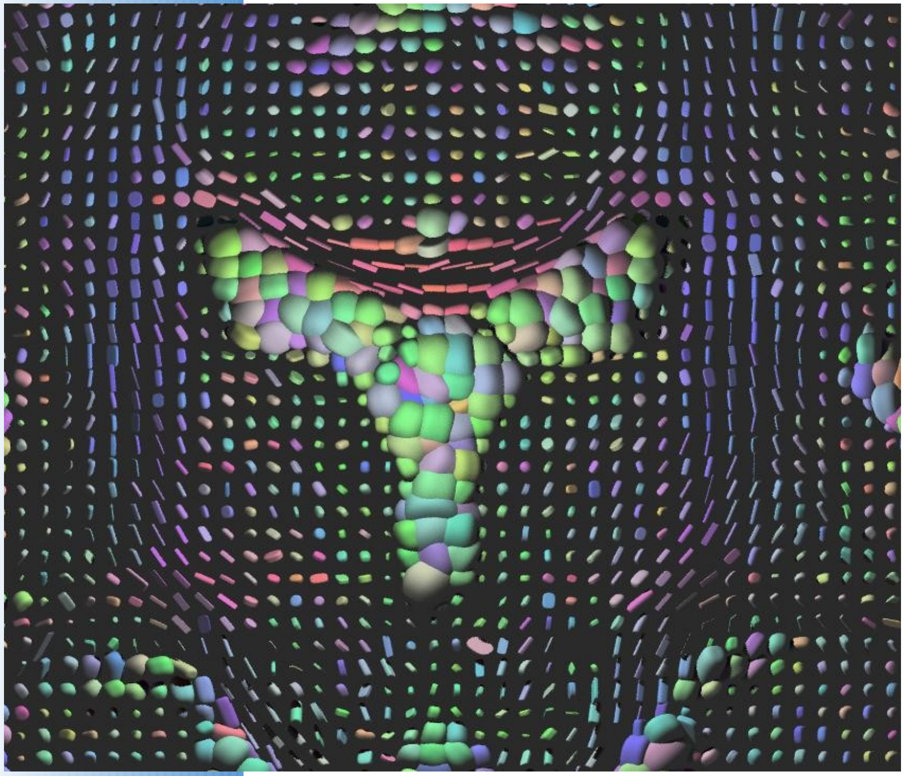


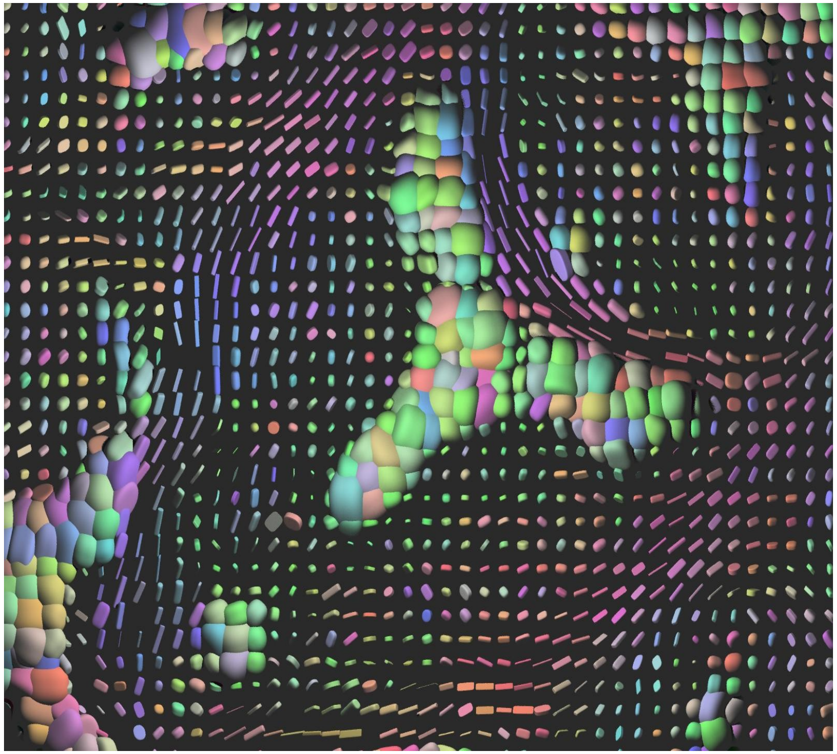
Memory usage

Images	
Input image	768MB
Output image	768MB
Interpolators	
Nearest Neighborhood	No additional memory used
Linear, BSpline and Windowed Sinc	768MB (input image size) to separate the components of the tensors into different images
BSpline	Additional 768MB (input image size) for the `itk::BSplineDecompositionImageFilter` used in the BSpline interpolator.
Deformations	
Deformation field	768MB
Other transformations	Small amount of memory used

Input image: 512x256x256, float values

Some results: after a 45 degrees rotation





Conclusion

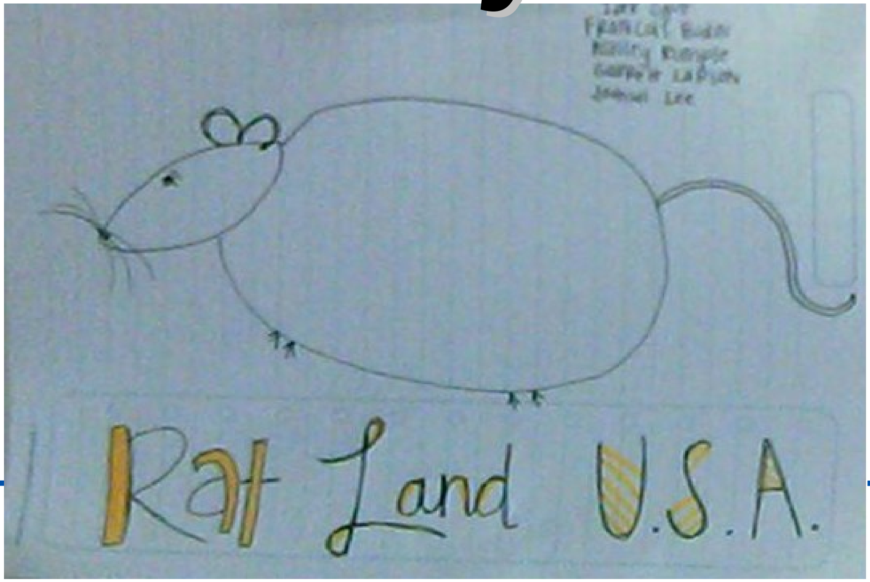
- New interpolation algorithm could be created
- Implemented as a module in Slicer3
- Submission of an Insight Journal paper

Acknowledgements

- Martin Styner, Ph.D, UNC
- Ipek Oguz, Ph.D., UNC
- Sylvain Bouix, Ph.D., BWH
- Martha Shenton, Ph.D., BWH

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Thank you!



Lab logo – courtesy Ashley Rumple