Projects for
Chapter 7: Tensor Visualization

1 PROJECT 1

Extend the *tensor metrics* code sample provided for Chapter 7 to compute and visualize additional tensor-related metrics, such as

- the linear, spherical, and planar anisotropies of Westin *et al.* (Equations 7.10)
- relative anisotropy (Equation 7.12)

Visualize these metrics using the confidence-based masking already implemented in the above-mentioned tensor metrics code sample. Discuss which metric (fractional anisotropy, diffusivity, linear/spherical/planar anisotropy, or relative anisotropy) appears to be able to separate fiber structures best from surrounding tissue.

2 PROJECT 2

Extend the *tensor metrics* code sample provided for Chapter 7 to visualize any of the tensor-related metrics implemented in that sample and/or in the previous project using a 3D isosurface. For this, you can reuse and/or extend the isosurface sample code provided in Chapter 5. To decrease the amount of (visual) noise, modulate the opacity of the isosurface vertices with the tensor field's confidence values.

3 PROJECT 3

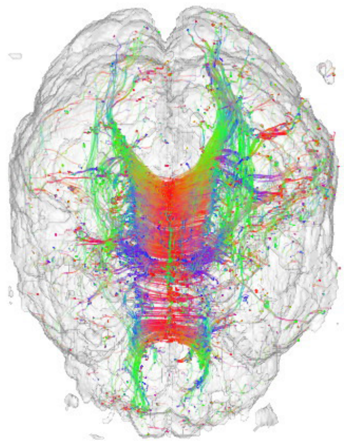
Extend the *tensor metrics* code sample provided for Chapter 7 to visualize any of the tensor-related metrics implemented in that sample and/or in the previous project using *volume rendering*. For this, you can reuse and/or extend the volume rendering sample code provided in Chapter 10. To decrease the amount of visual noise, use an opacity transfer function that reflects the tensor field's confidence values.

For the particular case of visualizing the direction of the major eigenvector of the tensor field, modify both color and opacity transfer functions as follows:

- color transfer function: This function should encode the direction of the major eigenvector, as described in the *tensor metrics* code sample provided for Chapter 7.
- opacity transfer function: This function should encode *both* the tensor field's confidence value and the value of the major eigenvalue. This way, shorter eigenvectors or eigenvectors computed at uncertain locations should look more transparent.

4 PROJECT 4

The *tensor streamlines* sample code provided for Chapter 7 shows how to visualize a 3D DTI tensor field by tracing streamlines along its major eigenvector field. The figure below shows an example visualization created with this method. Here, streamlines are densely seeded in areas of high fractional anisotropy, and are colored to show their local tangent direction (see also Figure 7.7 in Chapter 7).



Streamlines of the major eigenvector field of a 3D DTI tensor field.

However, this sample code uses a suboptimal streamline *seeding*. That is, streamlines

1. are not guaranteed to cover *all* volume areas where the anisotropy exceeds the user-prescribed minimal value;

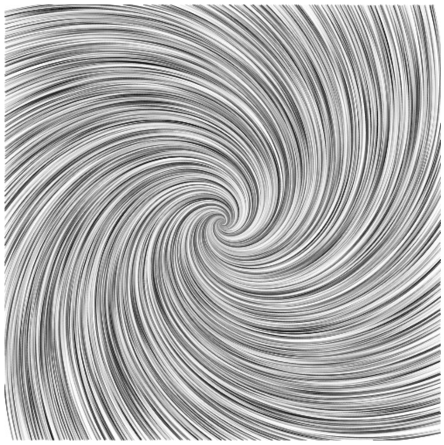
2. are not guaranteed to *evenly* cover the above-mentioned areas.

Design an alternative streamline-seeding and/or streamline-tracing algorithm that fulfills the above two criteria better than the current sample code. Compare your algorithm by showing side-by-side visualizations produced by your method and the current sample code.

Hints: Two methods for constructing evenly-distributed streamlines in 2D vector fields are described in Chapter 6, Section 6.5.1 (Jobard and Lefer, Mebarki *et al.*). Consider adapting these methods to the current problem, where we have the additional constraint of seeding only high-anisotropy areas.

5 PROJECT 5

Consider the *tensor streamlines* sample code provided for Chapter 7, also mentioned in Project 4. Adapt the provided sample code to generate a dense streamline visualization similar to the LIC-like effect presented in the figure below for a 2D vector field (the sample code for this technique is given in the *dense streamlines* demo in Chapter 6).



LIC-like visualization created by dense streamlines for a 2D vector field.

For this, consider first modulating the transparency and thickness of the densely-seeded 3D streamlines along their paths. Next, use a random luminance for different streamlines. Achieving the same effect is harder for a 3D streamline set than for its 2D counterpart, so careful parameter tuning and/or additional parameter settings may be necessary.

6 PROJECT 6

Implement the *hyperstreamlines* tracing technique described in Chapter 7.8, for the DTI dataset provided in the *tensor representation* sample code for Chapter 7. For this, proceed as follows:

- Compute the major, medium, and minor eigenvector and eigenvalue fields of the DTI field, as described in the *tensor metrics* sample in Chapter 7.
- Densely seed streamlines in areas of the DTI field where *both* the confidence and fractional anisotropy fields have high values.
- Trace streamlines until either the confidence or fractional anisotropy values decrease below an user-specified limit.
- For each streamline point $\mathbf{s}_i$, construct an ellipse in the plane perpendicular to the major eigenvector at $\mathbf{s}_i$ (which is tangent to the streamline). Represent the ellipse by a (closed) polyline having a fixed number of N sample points. Scale the ellipse using the values of the medium and minor eigenvalues, along the directions of the medium and minor eigenvectors respectively.

- Construct a stream tube surface by creating quadrilaterals that connect corresponding sample points of ellipses for consecutive sample points $\mathbf{s}_i$ and $\mathbf{s}_{i+1}$. Since all ellipses use N sample points, the connection algorithm should be simple.

- Color the stream tube by color-mapping the value of the fractional anisotropy using any of the color maps presented in Chapter 5. Additionally, you can modulate the stream tube opacity by the same scalar value.

Compare your results with the hyperstreamline visualization presented in Chapter 7, Figure 7.16.

Hints: Start from the sample code *tensor streamlines* provided for Chapter 7, that shows how to construct 3D streamlines that show the major eigenvector of a tensor field.

End of Projects for
Chapter 7: Tensor Visualization