USER INTERFACES TO INTERACT WITH TENSOR FIELDS. A STATE-OF-THE-ART ANALYSIS

Susana Merino Caviedes, David López Ibáñez, Juan García Avedillo and Marcos Martín Fernández
Laboratorio de Procesado de Imagen (LPI)
Dpto. de Teoría de la Señal y Comunicaciones e Ingeniería Telemática
Universidad de Valladolid, España

ABSTRACT

In this article we review the current situation of user interfaces for tensor fields. This is a very active research field nowadays, because the amount of information that tensors contain makes them difficult to design, and achieve an intuitive and easy to use interface. From a medical point of view, diffusion and strain tensors are the most widely used tensor images nowadays. The use of diffusion tensor magnetic resonance imaging (DTMRI) is being increasingly introduced in clinical practice. An important part of these interfaces is the visualization of the tensor field. Due to the high number of degrees of freedom, this is a difficult task, and many methods have been developed for it. We should also take into account that new hardware devices with a larger variety of functions could greatly help this kind of interfaces. However, most of these devices are being developed nowadays. As two examples, we study the DTMRI module of Slicer 3D and an immersive virtual environment for DTMRI visualization, as well as some of the feedback of their users.

1. INTRODUCTION

Tensor fields are attracting more and more interest, not only in physical and engineering applications, but also in medicine. A medical imaging modality, DTMRI, is able to obtain the diffusion tensor of the water molecules of organic tissue. In addition, the strain tensor of the heart can be computed from a series of tagged magnetic resonance images [1]. In cardiology, the mechanical deformation of the heart has been little studied due to technical difficulties in order to measure it [2]. Therefore, strain tensor images of the heart are a useful tool to obtain this information in a non-invasive way.

DTMRI images allow the measurement of water molecules diffusivity in organic tissue, which supplies information about its structure and what kind of tissue it is. Due to this, the use of tensor imaging in medicine has great interest, especially in those applications where the fabric of the tissue under study provides relevant information to the medical staff. A few medical applications where tensor imaging plays an important factor are: research on multiple sclerosis [3], image-guided neurosurgery [4], white matter abnormalities in schizophrenia [5], etc.

On the other hand, there is a human being that needs to employ these tensor fields to achieve these purposes. Therefore, a user interface must be developed that allows the interaction between the user and the applications that manage those tensor fields. However, this is not an easy task. A tensor codifies a great deal of information, and the way of interacting with it is not straightforward.

For example, there is a large number of ways a tensor field can be visualized. There are several types of glyphs, hyperstreamlines, volume techniques, etc. through which this task can be accomplished. Each of them have advantages and drawbacks, and each of them will be more appropriate for some goals than others.

The user should also be able to interact with these fields. There is a large variety of devices that may be used to accomplish th is purpose, from the familiar mouse and keyboard to some less known devices like trackers, wands or DataGloves.

All this must be blended together in order to make useful tools for the people that work with tensor fields. There are not many applications that work with tensor fields. One of them is Slicer 3D, which has a DTMRI module, in spite of being a more general software environment.

The rest of the document is structured as follows. In Section 2 a review of tensor concepts is given. Sections 3 to 6 explain most of the existing methods for tensor visualization. Hardware devices are exposed in Sections 7, 8 and 9. Lastly, in Section 10 some tensor interfaces are studied. In particular, 3D Slicer and an immersive virtual environment will be reviewed.

2. TENSOR CONCEPTS

2.1. Tensor Definition

A tensor is a certain class of geometric entity that generalizes the concepts of scalar, vector and linear operator, in a manner that is independent from any chosen reference frame. The comprehension of this basic idea is needed before being able to understand what a tensor is.

There are two equivalent ways of approaching the definition of tensors [6]:

- The usual physics way of defining tensors, in terms of objects whose components are transformed according to certain rules, introducing the ideas of covariant or contravariant transformations.

- The usual mathematics way, which involves defining certain vector spaces and not fixing any coordinate systems until bases are introduced out of necessity. Covariant vectors, for instance, can also be described as one-forms, or as the elements of the dual space to the contravariant vectors.

Physicists and engineers are among the first to recognize that vectors and tensors have a physical significance as entities, which goes beyond the (often arbitrary) coordinate system in which their components are enumerated. Similarly, mathematicians find that there are some tensor relationships which are more conveniently derived in a coordinate notation.

A rigorous definition of tensor is the following one:

Let:

1. E denote an n-dimensional vector space over a field $\mathbb{K}$, where $\mathbb{K}$ is either $\mathbb{R}$ (real numbers) or $\mathbb{C}$ (complex numbers).
2. $B = \{e_1, e_2, \ldots, e_n\}$ denote a base of the E field.
3. $\widetilde{B} = \{\widetilde{e}_1, \widetilde{e}_2, \ldots, \widetilde{e}_n\}$ denote another base of E.
4. $P = [p_i^j]$ denote the base change matrix or pass matrix between B and $\widetilde{B}$, namely:

$$\widetilde{e}_j = p_j^1 e_1 + ... + p_j^n e_n \tag{1}$$

Every magnitude (of undetermined nature) that, when measured in respect of the B base, remains determined unanimously by a collection of n^ρ numbers or functions, is denominated a ρ-order contravariant tensor over E:

$$X^{i_1,\ldots,i_\rho}, \qquad \text{with} \qquad 1 \leq i_1 \leq n, \ldots, 1 \leq i_\rho \leq n, \tag{2}$$

and when measured in respect of the $\widetilde{B}$ base, it also remains determined unanimously by another collection of n^ρ functions or numbers:

$$Y^{j_1,\ldots,j_\rho}, \qquad \text{with} \qquad 1 \leq j_1 \leq n, \ldots, 1 \leq j_\rho \leq n \tag{3}$$

in a manner that in-between both collections the following characteristic relationship is verified:

$$X^{i_1,\ldots,i_\rho} = p_{j_1}^{i_1} \cdots p_{j_\rho}^{i_\rho} Y^{j_1,\ldots,j_\rho} \tag{4}$$

Summarizing, and designating for our purpose the tensor order by ρ, and the dimension of the space by n, we can say that:

A ρ order tensor is a collection of n^ρ numbers or functions which characterize a certain property in a point of the n-dimensional space.

The notion of tensor is absolutely general. The property that distinguishes a scalar from a vector, and both of these from a more general tensor quantity, is the number of indexes in the representation matrix. This number is called the tensor rank. Thus, scalars are rank zero tensors, vectors are rank one tensors and matrices correspond to rank two (or second order) tensors.

A tensor field is defined as a function that assigns a tensor to every geometric space point. Occasionally, the definition is qualified by considering that the function which assigns a tensor to every point is only defined over a certain region or subset of the ordinary geometric space, for example over a curve or surface. The number of characteristic components varies depending on the dimension of the geometric space.

This work is focused in rank two tensor fields, due the importance they are acquiring for medical image with DTMRI. The three-dimensional geometric space implies $n = 3$, so a rank two tensor will have $3^2 = 9$ components susceptible to be represented as a 3×3 matrix.

There are projects concerning the possibility of working with higher tensor levels or even with pseudotensors specially indicated for certain problems like the intersection of fibers where we need much more information, though we are not going to consider them.

The increasing use of second-order tensors is the reason why we have considered the convenience of summarizing the most significant matrix characteristics and values, which are very useful for the representation of rank two tensor fields in three-dimensional environments. They are:

- Eigenvectors: vectors that are not affected by linear transformations or are multiplied by an scalar which does not produce any variation in its direction. Eigenvectors represent the matrix characteristic directions. Symmetric matrices present the property of having an orthogonal eigenvector base.

- Eigenvalues: An eigenvalue is the scale factor that its corresponding eigenvector has been multiplied by. They represent the basic measures of the matrix that are not altered by matrix linear transformations. For example, if we apply a coordinates change representing a rotation the eigenvalues will remain the same.

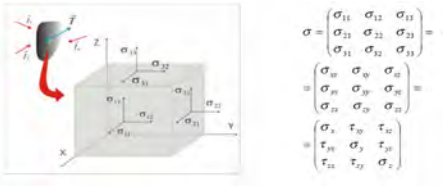


Fig. 1. Stress tensor.

- Determinant: it is a function depending on n, which associates a scalar $\det(A)$ to every $n \times n$ square matrix A. If the components of A are real numbers, then the determinant has the geometric meaning as the oriented volume of the parallelepiped spanned by its column vectors [7].
- Trace: is the sum of the main diagonal elements of a square matrix.

2.2. Tensor Fields in Physics and Other Engineering Branches

The use of tensors is fundamental in many fields of physics and engineering [8]. Next, some of the most commonly used rank two tensor fields are presented. However, we must point out that this is not an exhaustive list, and there are other types of tensors, like the Maxwell tensor or the spin tensor, that are not described here.

Gradient tensor of a vector field

The gradient of a vector field $\overrightarrow{F}$, is a tensor that provides the differential of the field when a displacement is made.

$$d\overrightarrow{F} = \overrightarrow{F}(\overrightarrow{r} + d\overrightarrow{r}) - \overrightarrow{F}(\overrightarrow{r}) = (\nabla \overrightarrow{F}) d\overrightarrow{r} \tag{5}$$

This tensor can be represented by a 3×3 matrix in a three-dimensional space. In Cartesian coordinates, it is composed of the three partial derivatives of the vector field components. The gradient tensor is very useful to describe the direction of maximum growth for each of the scalar fields that compose the vector field.

Inertia Tensor

The inertia tensor represents the mass distribution of a material system with respect to a coordinate frame. In conjunction with mass, the inertia tensor describes the inertia properties of rigid solids. Its components are given by:

$$I_{ij} = \int_V \rho(\overrightarrow{r}) \left[\delta_{ij} \sum_k x_k^2 - x_i x_j \right] dV \qquad \text{i,j,k=1,2,3} \tag{6}$$

Since inertia tensors are symmetric, $(I_{ij} = I_{ji})$, and only six of its components are independent. The inertia tensor is diagonal in respect with a Cartesian axis system (principal axis) at any point of the rigid solid. When this is the case, the diagonal elements (eigenvalues) I_1, I_2 and I_3 are denoted as principal moment of inertia. The eigenvectors, parallel to the principal axes, establish an orthonormal base (principal base), and the orthogonal transformation is obtained from them. This transformation allows to change from the given initial axes to the principal axes.

Stress tensor

In continuous medium mechanics, the stress tensor informs about the internal tensions and stresses. Given a reference frame, the stress tensor is defined by the equation:

$$dF_i = \sum_{j=1}^{3} \sigma_{ij} dA_j \tag{7}$$

The stress tensor represents the forces applied in each side of any differential space portion, as can be observed in Figure 1. There are three forces for each side, normal, vertical and horizontal. This tensor is allows an intuitive representation of the tensor idea.

Curvature Tensor

In differential geometry, the Riemann curvature tensor is the most standard way to express the curvature of Riemannian manifolds, or more generally, any manifold with an affine connection (torsionless or with torsion). The curvature tensor is given in terms of a Levi-Civita connection (more generally, an affine connection) ∇ (or covariant differentiation) by the following formula:

$$R(u,v)w = \nabla_u \nabla_v w - \nabla_v \nabla_u w - \nabla_{[u,v]} w \tag{8}$$

where R(u,v) is a linear transformation on the tangent space of the manifold and it is also linear in each argument.

If $u = \partial/\partial x_i$ and $v = \partial/\partial x_j$ are coordinate vector fields then $[u, v] = 0$ and therefore the previous formula simplifies to:

$$R(u,v)w = \nabla_u \nabla_v w - \nabla_v \nabla_u w \tag{9}$$

that is, the curvature tensor measures noncommutativity of the covariant derivative.

The linear transformation $w \to R(u,v)w$ is also called curvature transformation or endomorphism.

Deformation tensor

The deformation tensor is a symmetric tensor used in continuum medium mechanics and deformable solids mechanics to characterize body shape and volume variations. In a three-dimensional space, the deformation tensor has the following general form:

$$E = \begin{pmatrix} \varepsilon_{11} & \varepsilon_{12} & \varepsilon_{13} \\ \varepsilon_{21} & \varepsilon_{22} & \varepsilon_{23} \\ \varepsilon_{31} & \varepsilon_{32} & \varepsilon_{33} \end{pmatrix} \tag{10}$$

where each of the previous tensor components is a function whose domain is the set of points of the body whose deformation we pretend to characterize. For a small three-dimensional deformation U, their expression is:

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \tag{11}$$

The deformation tensor is directly related with the stress tensor by Hooke's equations, which are thermodynamical or constitutive relationships with the body material.

In mechanics, two tensor types can be distinguished:

- The finite deformation tensor, which allows to measure the real deformation and can be used for big deformations as well as small ones, and even takes into account geometrical non-linearities.
- The infinitesimal deformation tensor, which is obtained dropping some nonlinear terms from the finite one. This tensor is the most commonly used in engineering.

Diffusion tensor

Diffusion is the phenomenon by which substances traverse membranes (lipid bilayer), due to the continuous movement of molecules throughout liquids and gases. The movement of particles is generally the origin of heat, which implies that the more movement there is, the higher the temperature will be. In cells, passive transport does not need cell energy, as it is stimulated in an energetic way and it favors a concentration gradient (when a substance migrates from a high concentration region to another region with a lower concentration). There are two different types of cellular membrane diffusion:

1. Simple diffusion:

   It is the kinetic movement of molecules or ions through cell membranes, without the necessity of fastening using carrier proteins of the lipid bilayer. This kind of transport may be carried out by physiochemical mechanisms, like inverse osmosis, dialysis, and through channels or conduits capable of being ruled by the selective permeability and the floodgate mechanism of the different protein conduits.

2. Facilitated diffusion:

   The substance is not able to pass through the membrane without the help of a specific carrier protein. The main difference with simple diffusion through conduits is that, while the simple diffusion magnitude is increased in a proportional way to the concentration of the diffusing substance, in facilitated diffusion the magnitude reaches a maximum $V_{\max}$ while the substance concentration raises.

Diffusion is suitable to be represented by a second order tensor. In medical imaging, a tensor is obtained for each voxel (delimited portion of the space). It is also suitable to be divided in three different cases depending on the diffusion tensor eigenvalues, which are:

- Linear: one eigenvalue is clearly bigger than the other two.
- Planar: two eigenvalues have approximately the same value, and larger than the third.
- Spherical: all the eigenvalues are almost equal.

2.3. Diffusion Tensor Magnetic Resonance Imaging (DTMRI)

DTMRI is a technique that allows to measure water diffusion properties in human body by the use of MRI. Tensor diffusion is expected to become more and more relevant for diagnostic by medical image, especially in neurology and neurosurgery. DTMRI requires the use of special MRI pulse sequences, and it needs at least seven scalar diffusion-weighted images (DWI) taken in different spatial directions, in order to estimate the diffusion tensor image from them [9].

Data from a DTMRI image are representations of the diffusion tensor corresponding with infinitesimal volumes (voxels) of a human body. The molecule mobility in different directions is characterized by the degree of anisotropy, which allows to determine not only the type of tissue but also fiber orientation. There are several parameters used to facilitate the interpretation of the data. These parameters can be classified as diffusivity measurements, anisotropy measurements and orientation related parameters:

- Mean diffusivity. The goal of this parameter is to obtain a measure of the diffusion in a voxel, free of anisotropy effects, and to provide a value that is invariant with tissue direction or orientation. Using this parameter, tissues with the same mean diffusivity characteristics will ideally have the same value, independently of its direction or orientation.

$$\langle D \rangle = \frac{trace(D)}{3} = \frac{D_{xx} + D_{yy} + D_{zz}}{3} = \frac{\lambda_1 + \lambda_2 + \lambda_3}{3} \tag{12}$$

- Anisotropy is the property of presenting different characteristics depending on the orientation. It is the opposite property to isotropy, which is characterized for presenting equal properties in all directions. If we define $L_i = \lambda_i - \langle D \rangle$, the expressions of some anisotropy indexes are:

  - Relative anisotropy (RA), a coefficient that measures the variation of the eigenvalues:

$$RA = \sqrt{\frac{L_1^2 + L_2^2 + L_3^2}{\sqrt{3}\langle D \rangle}} \tag{13}$$

  - Fractional anisotropy (FA), that characterizes the ellipsoid eccentricity:

$$FA = \sqrt{\frac{3[L_1^2 + L_2^2 + L_3^2]}{\sqrt{2(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)}}} \tag{14}$$

  - Volume Ratio (VR), which is the quotient between the ellipsoid volume and the volume of a sphere with the same mean diffusivity [10].

$$VR = \frac{\lambda_1 \lambda_2 \lambda_3}{\langle D \rangle^3} \tag{15}$$

- Fiber orientation: DTMRI facilitates information regarding the direction followed by the fibers present in the white matter of the nervous system.

There are other types of indexes, commonly employed to characterize the type of diffusion presented. By using the largest eigenvalues ($\lambda_1 > \lambda_2 > \lambda_3$) of the tensor, the following normalized quantitative shape measures are obtained:

- Linear coefficient:

$$c_l = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3} \tag{16}$$

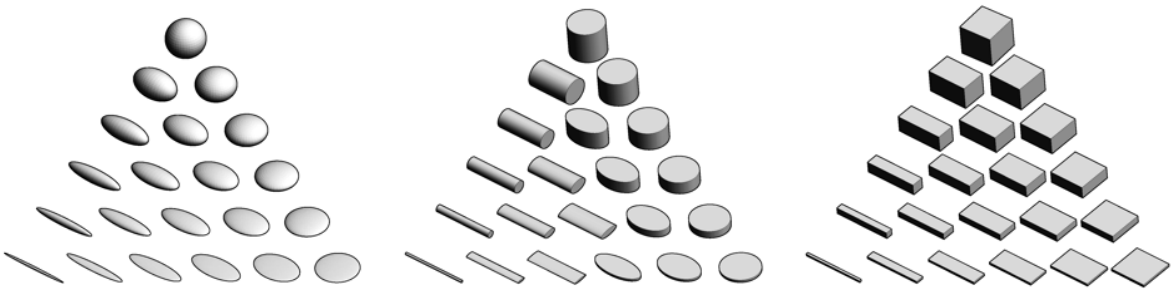
- Planar coefficient:

$$c_p = \frac{2(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3} \tag{17}$$

- Spherical coefficient:

$$c_s = \frac{3(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3} \tag{18}$$

The sum of these three coefficients must always be the unity, when they are defined this way.



(a) Ellipsoids (b) Cylinders (c) Cuboids

Fig. 2. Comparison between glyphs using different geometric figures ([12]).

3. GLYPHS

A popular way of visualizing tensor fields is to represent the tensor at each point of the field with a glyph [9], which is a geometric figure whose form depends on the characteristics of the tensor it represents. There is a wide variety of these forms, and we will review the most important types, which we will classify taking into account which tensor characteristics are used in the definition of the glyph shape. Therefore, the categories are: glyphs that codify eigenvectors and eigenvalues, glyphs that use anisotropy coefficients, and glyphs that represent other measures, like curvature, shear, rotation, etc. produced by the tensor.

3.1. Glyphs coding Eigenvectors and Eigenvalues

The most popular shape for a glyph is an *ellipsoid*, by which tensor eigenvectors and their corresponding eigenvalues are represented. The ellipsoid's major axis is orientated along the major eigenvector[1], and the medium and minor axes in the direction of the medium and minor eigenvectors, respectively. We must point out that this representation is valid only for symmetric tensors, in order to make sure that the eigenvalues are nonnegative real and the eigenvectors are real and orthogonal. The size of the ellipsoid axes will be proportional to the corresponding eigenvalues.

Depending on the relative values of the eigenvalues, the ellipsoid will take one form or another:

- If λ_1 is visibly greater than the other eigenvalues, the ellipsoid's shape will be elongated in the mayor eigenvector's direction. This happens where diffusion is very linear as for example in some brain regions where neural fiber tracks, also called tracts, form coherent bundles along the same direction.
- If $\lambda_1 \simeq \lambda_2$, the ellipsoid's shape resembles a disk, and the smaller λ_3 is, the flatter it will be. This type of tensors are often seen in regions where a fiber bundle bifurcates or where two bundles cross each other, and bring incertitude in the computation of the direction of tracts, as seen in Section 4.
- If $\lambda_1 \simeq \lambda_2 \simeq \lambda_3$, the resulting ellipsoid looks like a sphere. When this happens, the region is isotropic, and there are no preferred diffusion directions.

We must take into account that the factor of anisotropy FA suffers great variations from a region to another, which translates to big differences in ellipsoid sizes,which is not good for the visualization. Laidlaw *et al.* propose a normalization for ellipsoids [11], by which the major axis is given unitary value, and the size of the other axes is computed accordingly, in order to preserve the proportion between them. The ellipsoids are represented on a 2D slice, where the background color at each pixel supplies the information lost in the normalization.

On the other hand, ellipsoids are not the only shapes one can use for representing tensor eigenvectors and eigenvalues. In [12] we can see that cuboids and cylinders can be used to represent eigenvectors and eigenvalues instead of ellipsoids. In Figure 2 glyphs using these three geometrical figures can be observed: ellipsoids in Figure 2(a), cylinders in Figure 2(b) and cuboids in Figure 2(c).

3.2. Glyphs that Include Anisotropy Coefficients

3.2.1. Composite Shapes

A method for representing tensors so that its different anisotropy coefficients can be distinguished is presented in [9]. Instead of ellipses or other individual shapes, a composite of a headless arrow, a circle and a sphere, each of them proportional to the

[1]We name major eigenvector to the eigenvector corresponding to the largest eigenvalue λ_1, and the medium and minor eigenvectors to the ones corresponding respectively to λ_2 and λ_3 ($\lambda_2 \geq \lambda_3$).

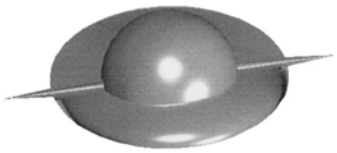


Fig. 3. Composite shape representing a tensor with eigenvalues $\lambda_1 = 1$, $\lambda_2 = 0.7$ and $\lambda_3 = 0.4$ ([9]).

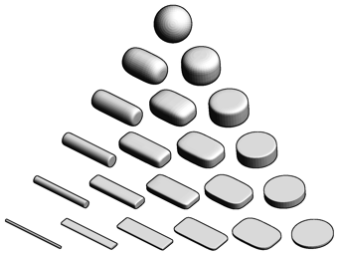


Fig. 4. Superquadric tensor glyphs ([12]).

coefficients c_l, c_p and c_s respectively is used. This composition is named composite shape, and it helps to appreciate better the exact value of the coefficients, as can be seen in Figure 3.

Unfortunately this glyph also presents ambiguity, so these glyphs must be given a coloring depending on the tensor anisotropy to solve this problem. The color is interpolated between the blue linear case, the yellow planar case and the red spherical case.

3.2.2. Superquadric Tensor Glyphs

In order to overcome the problems presented by the glyphs seen in Subsection 3.1, Kindlmann proposed a new kind of glyphs called *superquadric glyphs* [12]. The basic idea is to use superquadrics to represent a tensor, depending on the anisotropy coefficients c_l and c_p, and using the eigenvectors to orientate the glyph.

The shape of a superquadric is controlled by two parameters, and spheres, cubes and cylinders can be expressed as particular cases of a superquadric. The expression of the base geometry of a superquadric glyph varies depending on the anisotropy:

Case $c_l \geq c_p$:

$$q(x,y,z) \equiv \left(y^{2/\alpha_l} + z^{2/\alpha_l}\right)^{\alpha_l/\beta_l} + x^{2/\beta_l} - 1 = 0 \tag{19}$$

Case $c_l < c_p$:

$$q(x,y,z) \equiv \left(x^{2/\alpha_p} + y^{2/\alpha_p}\right)^{\alpha_p/\beta_p} + z^{2/\beta_p} - 1 = 0 \tag{20}$$

where $\alpha_l = (1-c_p)^\gamma$, $\beta_l = (1-c_l)^\gamma$, $\alpha_p = (1-c_l)^\gamma$ and $\beta_p = (1-c_p)^\gamma$, and γ controls the edge sharpness.

The resulting superquadric tensor glyphs for $\gamma = 3$ can be seen in Figure 4. We must point out that their axes are scaled by the tensor eigenvalues.

3.3. Glyphs that codify other tensor characteristics

There are visualization methods that codify other tensor information, like the curvature or shear produced when the tensor is applied as a linear transformation. In particular, in [13] a *flow probe* is proposed that shows information about curvature, shear, rotation, velocity, acceleration and convergence/divergence. This probe can be seen in Figure 5.

3.4. Comments about Glyphs

Glyphs allow the visualization of all the information associated to a tensor. Color is often used to disambiguate glyphs, giving them a hue computed as the interpolation of the tensor anisotropy coefficients c_l, c_p and c_s.

Glyphs, however, have also drawbacks. First, a glyph corresponds to only one tensor, so the continuity of the tensor field is lost in the visualization. Another essential point is the number and position of the represented glyphs, in order to avoid a sensation of cluttering, which might happen if too many glyphs are represented. Furthermore, overlapping of glyphs must be avoided.

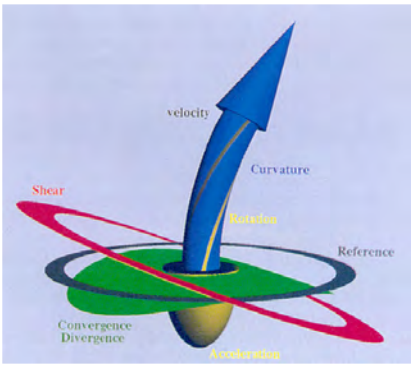


Fig. 5. Flow Probe and its components ([13]).

4. TRACT VISUALIZATION

One of the main applications of DTMR Imaging is the computation of tractographies of the brain white matter, where the structure of the tracts[2] appearing in the image is obtained from the diffusivity measure at each voxel.

Finding the tracts passing by a voxel is not a trivial problem, mainly due to the following reasons:

- DTMRI images have a poor resolution compared to the size of nerve fibers, whose section can be measured in microns, whereas the volume of a DTMRI voxel is of the order of mm^3. However, this can be partially solved, because nerve fibers often form large bundles going in the same direction.
- Tractographies are often computed following the direction given by the major eigenvector (corresponding to λ_1). When the tensor anisotropy is clearly linear, this method yields adequate results. Nevertheless, where planar or spherical anisotropy prevails, there is ambiguity about the direction of the tract at that point [9]. In [14] a new technique called *tensorlines* is introduced which tries to solve this problem incorporating nearby orientation and anisotropic information to the computation.

Usually a large number of seed points are selected as starting points for tract computation, and afterwards a culling algorithm is applied to select the most representative ones [15].

Tractography visualization is an important task, which overlaps with some tensor field visualization techniques as hyperstreamlines [16] or streamtubes [15]. In the following subsections we will review hyperstreamlines, streamtubes and streamsurfaces as tensor visualization techniques.

4.1. Hyperstreamlines

Following the philosophy of streamlines, Delmarcelle and Hesselink propose *hyperstreamlines* as a model for the visualization of symmetric tensor fields [16]. This concept was inspired by streamlines as a representation technique for vector fields, where each vector in the field is tangent to a streamline.

The shape of streamlines is similar to a tube with an ellipsoidal crossection. The tube direction at each point is computed from the vector field formed by the major eigenvector of each tensor in the field, similarly to streamlines. In addition, the axes of the crossection ellipsoid are proportional to the medium and minor eigenvalues, and oriented in the directions of their corresponding eigenvectors.

Therefore this method can be used to represent all the tensor information at each point of a hyperstreamline, and a continuity sensation is achieved, too. It is not possible, however, to represent this information in the whole volume, becase hyperstreamlines still are discrete elements and may suffer from cluttering, as happens with glyphs.

An example of hyperstreamlines can be seen in Figure 6. In this picture hyperstreamlines are given color at each point depending on their linear anisotropy. There are, though, other ways of coloring the hyperstreamlines.

A technique to provide information to the user about the probability that a tract has been correctly computed is presented in [17], where Ehricke *et al* compute at each point i of the streamline the probability $P_{loc,i}$ that this point is crossed by a tract. To achieve this, the anisotropy coefficients and the conformity of the major eigenvectors in a neighborhood of the point are employed. Using these probabilities and the streamlines of the vector field defined by the major eigenvector, the authors compute the probability $P_{path,i}$ that a sample belongs to the tract taking into account the path followed before by the streamline. This probability is given by the expression:

$$P_{path,i} = P_{cond,i} = P_{loc,i} P_{path,i-1} \tag{21}$$

[2]Tract: a bundle of myelinated nerve fibers following a path through the brain.

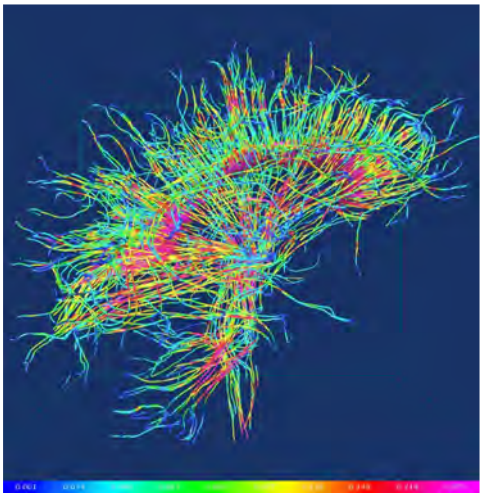


Fig. 6. Visualization of DTMRI using hyperstreamlines, whose color is proportional to the tensor linear anisotropy at each voxel.

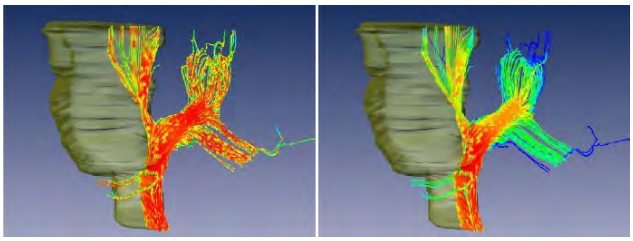


Fig. 7. Color mapping of streamlines using the probability that the point belongs to the tract ([17]). The local probability is used on the left, and the probability conditioned on the followed path is shown on the right. Color vary from blue (lowest value) to red (highest value).

This probability can be used to color the streamlines or hyperstreamlines, as can be observed in Figure 7, or perhaps could be used to map their transparency and opacity in the rendering of the visual scene.

4.2. *Streamtubes* and *streamsurfaces*

A very similar way to hyperstreamlines of tract visualization are *streamtubes*, proposed by Zhang *et al.* [15], which try to overcome certain limitations of *hyperstreamlines*. Firstly, its crossection can get quite large, which makes smaller the number of them that can be visualized in a volume; this can become problematic in the analysis of biological tissue. Secondly, in regions where the anisotropy is mainly planar, hyperstreamlines are not very effective [15].

For all these reasons, Zhang *et al.* propose to split the volume in regions of linear anisotropy and planar anisotropy, so that the former are visualized by *streamtubes*, and the latter by *streamsurfaces*. Herein lies the problem of setting adequate thresholds for the regions. In [15] these thresholds are manually defined.

Streamtubes are in essence very similar to hyperstreamlines. Their main difference lies in how the cross-section is computed. Although its shape is an ellipse in both techniques, the major axis, corresponding to the medium eigenvector and oriented along this direction, is forced to be constant, and the length of the minor axis, oriented along the minor eigenvector, will be computed so that the proportion between axes is $\frac{\lambda_2}{\lambda_3}$. This way more elements may be represented than with hyperstreamlines, although the information in the whole volume still cannot be visualized in one representation.

In order to avoid ambiguity between different tensors with the same quotient $\frac{\lambda_2}{\lambda_3}$, the streamtube is given a red hue proportional to the linear anisotropy coefficient c_l. That means that the more linear the diffusion is, the redder the streamtube will be.

Streamsurfaces are represented in regions where c_p has a high value, and they try to be an approximation at each point to the surface defined by the major and medium eigenvectors, which will lie on the tangent orthogonal plane to the streamsurface at each point. As happened with streamtubes, a streamsurface is given a hue proportional to the planar anisotropy coefficient c_p, but in this case the chosen color is green. We must note that the same colors, red and green, are chosen in [18] to represent linear and planar anisotropy regions respectively.

In Figure 8 an example of streamtubes and streamsurface can be observed. This technique received feedback from the doctors and medical students that used it [15]. They said that more anatomical detail and streamtubes in the region of interest would enhance their understanding, and some of them would also like to be able to display other scalar-valued images.

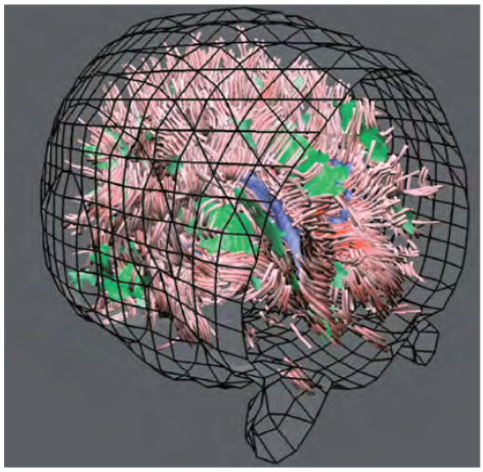


Fig. 8. Brain DTMRI visualization with streamtubes and streamsurfaces ([15]).

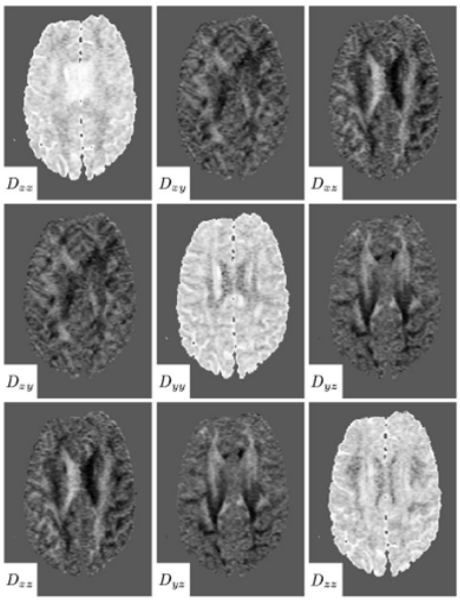


Fig. 9. Visualization of each tensor component of a DTMRI slice.

5. VOLUME VISUALIZATION TECHNIQUES

5.1. Use of Pseudocoloring

The simplest way to visualize a tensor field, but also the least intuitive to the viewer, is to represent each tensor component of a slice with a pseudocolor. In three dimensions, the volume is divided into 2D slices, and a pseudocolor is given to each component. Then nine images of that slice are presented in a 3 by 3 collage, one for each component of the tensor [19], as can be seen in Figure 9.

This technique, in spite of being the easiest to implement, has several drawbacks. First, it forces the user to mentally integrate the interactions of the components of the vector, which is complicated and would need a great deal of training to do so. Second, it must be done slice by slice, which makes the user to lose sight of the tridimensional perspective of the tensor field. Because of these reasons, this method is barely used.

On the other hand, the use of color in addition to other techniques can be very useful to improve a tensor field visualization. As a result, more information about the tensor field can be coded in the visualization. An example of this is the use of color to help disambiguate tensor glyphs, as described in Section 3.

Coloring can also help in the visualization of a tensor field by making clearer a representation, and thus reducing the cluttering effect. Brun et al. [20] use Laplacian eigenmaps to assign color to tracts computed from a DTMRI volume, so that similar fibers are similarly colored. The application of this technique to hyperstreamline visualization is shown in Figure 10. If we compare Figures 10 and 6, we can observe that the Laplacian eigenmap coloring greatly reduces the cluttering effect.

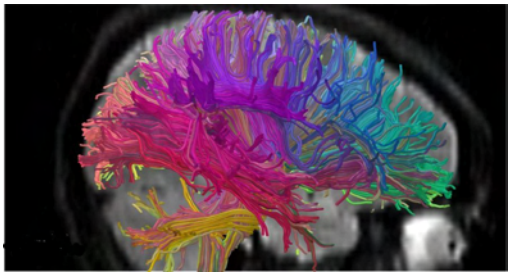


Fig. 10. DTMRI visualization using hyperstreamlines with Laplacian eigenmap coloring ([20]).

5.2. Volume Deformation

An alternative method for tensor field representation is to observe the transformation that an object suffers because of the tensor influence. This method is known as *volume deformation* [19, 21].

Boring and Pang [21] put a geometric object, called interrogation object, under the action of a certain tensor field, obtaining a deformed object as a result.To achieve this they use two strategies. In the first one:

$$O(\mathbf{x}) = I(\mathbf{x}) + s\left[\mathbf{T}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x})\right] \tag{22}$$

where $O(\mathbf{x})$ is the deformed object, $I(\mathbf{x})$ is the interrogation object, $\mathbf{T}(\mathbf{x})$ is the tensor at point $\mathbf{x}$, $\hat{\mathbf{n}}(\mathbf{x})$ is a vector field independent from the object, which is usually defined as the surface normal of the interrogation object, and s is a scale factor.

In the second strategy, if we define $\mathbf{r}(\mathbf{x}) = \mathbf{T}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x})$, the equation (22) can be expressed:

$$O(\mathbf{x}) = I(\mathbf{x}) + s\left[\mathbf{r}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x})\right] \hat{\mathbf{n}}(\mathbf{x}) \tag{23}$$

which is the deformation by the projection of $\mathbf{r}(\mathbf{x})$ onto $\hat{\mathbf{n}}(\mathbf{x})$ in the direction of $\hat{\mathbf{n}}(\mathbf{x})$, and consequently helps identify the deformation in the normal direction of the interrogation object surface. If we want to analyse the deformation corresponding to torsion, the following equation should be employed:

$$O(\mathbf{x}) = I(\mathbf{x}) + s\left[\mathbf{r}(\mathbf{x}) - (\mathbf{r}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x}))\, \hat{\mathbf{n}}(\mathbf{x})\right] \tag{24}$$

The equations (23) and (24) allow the study of the tensor field at a certain direction, which is convenient in respect to user interaction. In addition, with this method it is not necessary to impose symmetry restrictions to the tensors represented by it. However, the fact of taking the normal vectors of the interrogation object surface forces it to be well defined at every point.

Zheng and Pang [19] indicate that the major drawback of this method is that the vector variation is only visualized in a direction defined by the user on each tensor field representation, and they propose two techniques to solve this.

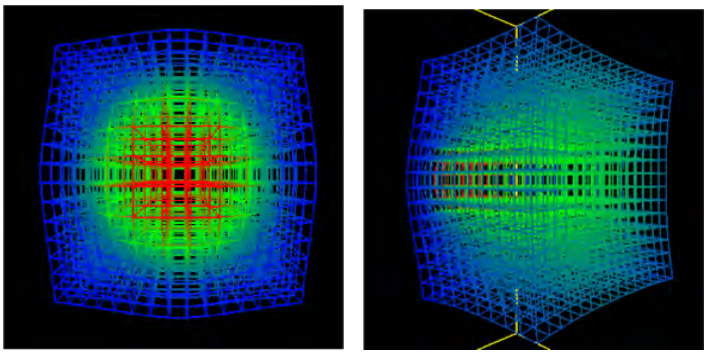
1. **Normal Vector Deformation**. Here, the interrogation may be different at various parts of the tensor field, and may change over time. The deforming forces are obtained multiplying tensors and the normal vectors of the interrogation object. Space is modeled as a lattice of springs connecting node neighbors, and the deformation proceeds until the spring system reaches equilibrium.

2. **Anisotropic Deformation**. If the transformation corresponding to a constant tensor field is applied to a spheric object, the result would be an ellipsoid. The aim of this method is to unify these objects in one in order to represent the tensor field information. Here a user-defined normal vector field is not needed, and the position of point $\mathbf{x}$ after deformation will be given by function $D(\mathbf{x})$, which should satisfy the condition [19]:

$$\left.\frac{\partial D}{\partial x}\right|_{\mathbf{x}=\mathbf{x_0}} = R(\mathbf{x_0}) \cdot T(\mathbf{x_0}) \tag{25}$$

   where $R(\mathbf{x})$ is a rotation tensor field and $T(\mathbf{x})$ is the tensor field. However, no $D(\mathbf{x})$ satisfies this critical condition for a general tensor field [19]. This is solved using again the spring model to approximate $D(\mathbf{x})$.

In Figure 11 a visualization using this technique can be observed. We must point out that the represented tensor field, in spite of not having symmetry conditions, must be continuous.



(a) (b)

Fig. 11. Example of volume deformation for the single point tensor field data set ([19]).

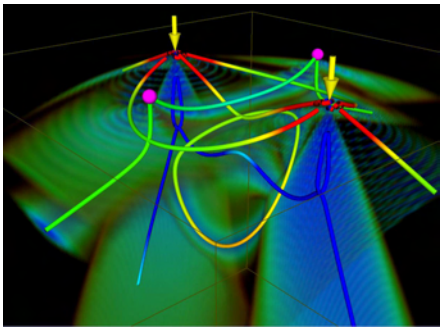


Fig. 12. Representation of a double point load data using topological features ([23]).

5.3. Use of Topological Features

In [22, 23], a new method for tensor field visualization is proposed. This method consists of extracting topological features of the tensor field, mainly degenerate points where two or more eigenvalues are identical. This provides an uncluttered view of the tensor field, but a lot of information about the tensor field is hidden from the user.

In Figure 12, a representation of a double point load data taken from [24] can be observed. The magenta circles are triple degenerate points, and the yellow arrows indicate the two point load directions.

5.4. Use of Textures

Textures have been sometimes used in combination with other tensor field visualization techniques like glyphs [11, 25], but they can also be employed as standalone methods for tensor field visualization, like HyperLIC and reaction-diffusion textures, which will be described below.

5.4.1. Line Integral Convolution and HyperLIC

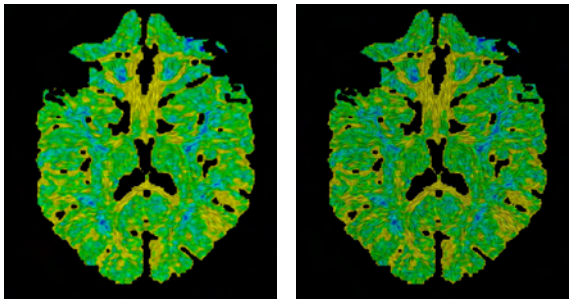
One of the existing methods to represent vector fields in a volume is *Line Integral Convolution* or LIC, proposed at first by Cabral and Leedom in [26]. This technique takes a vector field and a white noise image of the same size, computes streamlines at each pixel and promediates the noise values of the pixels that fall in the streamline. Later in [27], the visualization of time-dependent flux, the addition of the vector magnitude to the visualization, and the ability of employing other coordinate systems were incorporated to LIC. In Figure 13 an example of vedtor field visualization using LIC is given.

In [28] LIC is adapted for its application to flux visualization in volumes, and additionaly introduced some visualization enhancements that are explained in Section 6.2. For tensor field visualization, however, HyperLic [29] is the related technique that fits better for this purpose.

HyperLIC uses the vector fields defined by the major, medium and minor eigenvectors of the tensors at each point. Streamlines are computed at N contiguous points and in their vicinity, resulting in a volume that can be compared to a hyperstreamline. The value at each pixel is computed as the average of the noise inside this partial volume in the white noise image. Additionally color can be given to the visualization mapping the tensor magnitude at each point [29].

In Figure 14 an example of a 2D visualization using HyperLIC can be observed. Two values for N are used. we can see that regions corresponding to linear anisotropy are sharper, whereas regions with planar anisotropy are blurred, especially when N takes a high value. The reason of this is that the average computed in nearby points of the same streamline have small variations,

Fig. 13. Use of LIC to visualize a circular vector field.



(a) N=2 (b) N=8

Fig. 14. Use of HyperLIC for the visualization of a 2D slice of a brain DTMRI image ([29]).

and this is often the case where c_l is high. On the other hand, in regions of planar anisotropy, the average in the white noise image acts as a smoothing kernel.

5.4.2. Reaction-Diffusion Textures in Tensor Field Visualization

In HyperLIC, a white noise image was processed according to some characteristics of the tensor field in order to generate a visualization of said tensor field. In reaction-diffusion textures, instead of noise a well organized pattern of ellipsoids is tuned in order to obtain the visualization of the tensor field [30, 31].

A reaction-diffusion system models the reactions between two chemicals that diffuse at different rates and interact according to certain rules of activation and inhibition [31], and may be employed for texture generation in computer graphics.

In order to use this technique to tensor field visualization, Kindlmann et al. [31] adapt this model to the case of a single chemical c to an anisotropic medium where $\mathbf{D}$ is the diffusion tensor matrix. The behaviour of the model is given by the equation [30, 31]:

$$\frac{\partial c}{\partial t} = \nabla \cdot (\mathbf{D}\nabla c) = \frac{\partial}{\partial x_i}\left(D_{ij}\frac{\partial c}{\partial x_j}\right) = D_{ij}\frac{\partial^2 c}{\partial x_i \partial x_j} + \frac{\partial D_{ij}}{\partial x_i}\frac{\partial c}{\partial x_j} \tag{26}$$

where $\mathbf{D} = \{D_{ij}\}_{1\leq i,j\leq 3}$ and c is the concentration of the chemical.

In Figure 15 a 2D slice of a 2D DTMRI is visualized using ellipse glyphs and reaction-diffusion texture. Reaction-diffusion textures have some advantages over glyphs [30]. Firstly, the placement of spots in the field follows better the underlying features of features of the data, whereas glyphs have a predetermined position in a grid. Secondly, the distribution of the texture spots is in accordance to their size and shape, so they can follow curves more closely. In addition, glyphs tend to overlap in isotropic regions, which does not happen with reaction-diffusion textures.

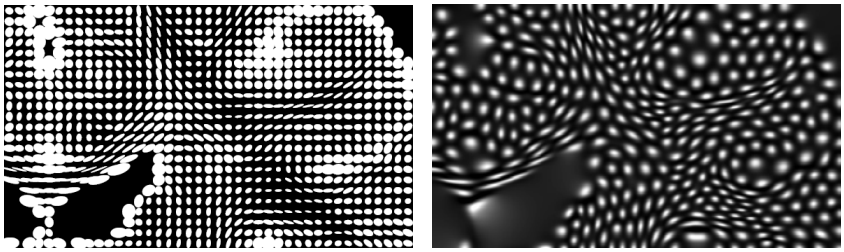
As shortcomings of this method we can list that an initial population of spots must be computed, it is computationally expensive, and additional processing must be carried out in order to use the spots as glyphs.

Although reaction-diffusion is a standalone visualization technique for tensor fields, it may also be used combined with other techniques, like volume rendering [31]. In Figure 16, an example of this is shown.

6. RENDERIZATION TECHNIQUES FOR TENSOR VISUALIZATION

6.1. Use of Barycentric Mapping

An important question in visualization is the ability to show the information relevant to the viewer, and to hide the one that is not. If the data we want to visualize is behind other uninteresting data, the purpose of the visualization is not fulfilled. A solution to



(a) Ellipse glyphs (b) Reaction-diffusion texture

Fig. 15. Comparison between ellipse glyphs and reaction-diffusion texture, when they are used to visualize a portion of a 2D DTMRI slice ([30]).

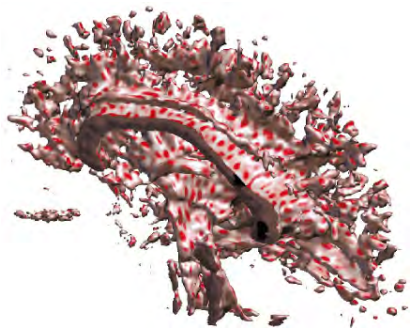


Fig. 16. Reaction-diffusion mapped onto surface rendering ([31]).

this problem is to assign an opacity to the data, so that those having a greater interest are more opaque, and those data we do not want to visualize are transparent. This can be achieved with an *opacity function*.

Kindlmann et al. propose a function based on a barycentric opacity map [31], in which the opacity depends on the anisotropy coefficients c_l, c_p y c_s defined in [9]. We must specify that c_s measures in fact the isotropy, so a tensor should be more transparent the higher this coefficient is.

In order to construct a barycentric map, we take an equilateral triangle, where each vortex and its corresponding opposite side are the extreme values the anisotropy coefficients can take, as can be seen in Figure 17. We will only set the values for c_l and c_p, obtaining c_s from the equality $c_l + c_p + c_s = 1$. This way, each point of the triangle will correspond to a fixed value of c_l, c_p and c_s. Once we do this, we must assign an opacity for each point in the triangle; this way, a certain value will correspond to each tensor, depending on its type of anisotropy. The relative weight of linear and planar anisotropy can vary by means of this mapping, so that we can see the regions in which one or the other prevails.

For example, we can observe in Figure 18 the result of using different barycentric maps to control the transparency of a cerebral DTMRI volume.

Barycentric mapping can also be used to assign colors to tensors with different anisotropy coefficients. In Figure 19 a DTMRI is rendered using barycentric maps to determine the color and opacity given to a tensor. As can be observed, the use of color helps distinguish the regions of linear anisotropy from the ones of planar anisotropy.

6.2. Halos

When there is a high density of hyperstreamlines or streamtubes in a volume, it can be difficult to tell each one apart from the others, even with the illumination technique proposed by Zöcker *et al.* [32]. In order to solve this problem, Wenger *et al.* [33] propose the addition of an halo to each streamtube, so that it can be better distinguished from the rest of them. The model for

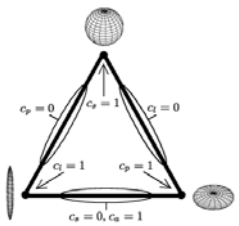


Fig. 17. Barycentric map using the tensor anisotropy coefficients.

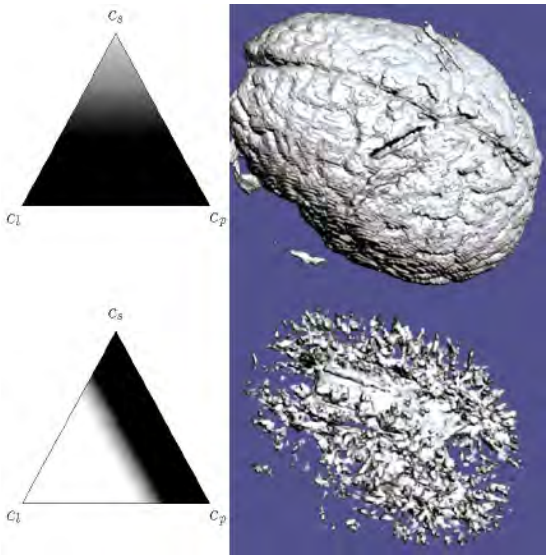


Fig. 18. Examples of barycentric maps for opacity computation and the resulting volumes ([31]).

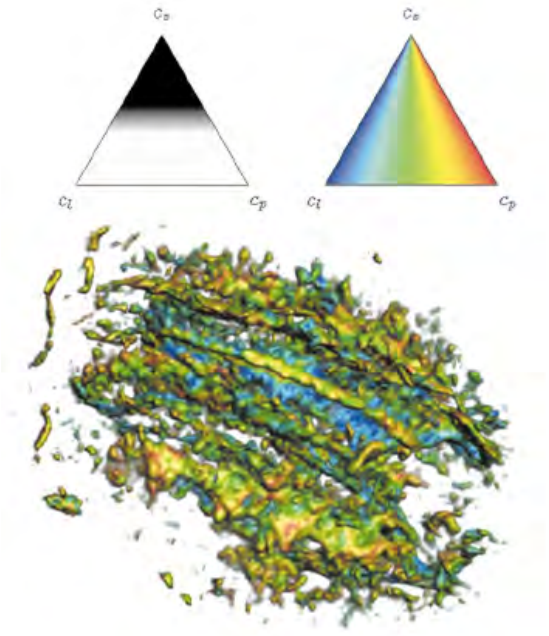


Fig. 19. Volume rendering using barycentric maps for tensor opacity and color ([31]).

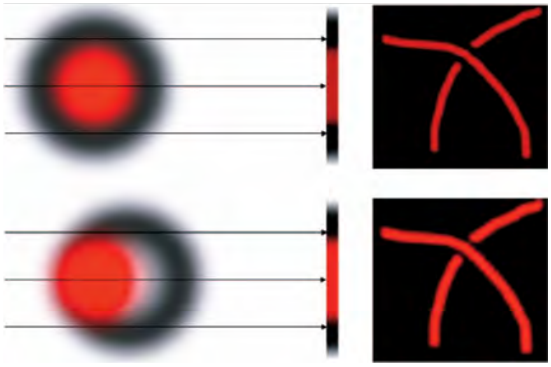
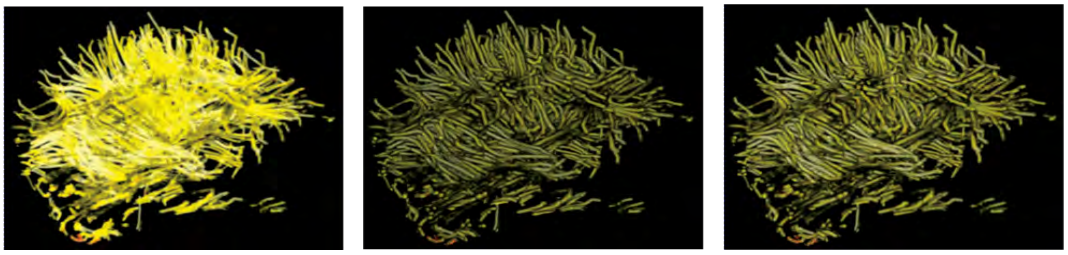


Fig. 20. Result of adding halos to a thin thread ([33]).



(a) Without halos (b) With concentric halos (c) With displaced halos

Fig. 21. Different visualizations of a volume showing brain connectivity information ([33]).

these halos is taken from [28], where, following pictoric techniques, a shadow is cast in order to improve the visual sensation of depth in Volume LIC, introducing discontinuities between different objects.

In Figure 20, the way halos work can be observed. They make discontinuities to appear so that the individual threads can be better distinguished. In Figure 21 three thread volumes are shown. In the volume without halos, it is difficult to tell a thread apart from its neighbors. In the volume with halos this problem is solved. However, the concentric halos make threads look darker than the displaced halos.

6.3. Hueballs and Lit-tensors

There are methods to represent tensor fields in the whole volume using rendering techniques [31, 34]. As opposed to glyphs, these methods do not show all the information contained by tensors, but they display it at every point of the volume, which gives a sense of continuity to the tensor field.

In the following subsections we describe the techniques that Kindlmann and Weinstein [34] have named *hueballs* and *lit-tensors*.

6.3.1. Hueballs

Hueballs give information about the orientation of vectors by defining a colormap on a sphere, so that we have a continuous mapping from direction to color [34]. In order to use this technique with second-order tensor fields, we must consider tensors as a linear transform that we apply to a user-defined input vector so as to obtain an output vector field, as the following equation states:

$$\begin{aligned} M\mathbf{v} &= M(\mathbf{v}\cdot\mathbf{e_1})\mathbf{e_1} + M(\mathbf{v}\cdot\mathbf{e_2})\mathbf{e_2} + M(\mathbf{v}\cdot\mathbf{e_3})\mathbf{e_3} \\ &= \lambda_1(\mathbf{v}\cdot\mathbf{e_1})\mathbf{e_1} + \lambda_2(\mathbf{v}\cdot\mathbf{e_2})\mathbf{e_2} + \lambda_3(\mathbf{v}\cdot\mathbf{e_3})\mathbf{e_3} \\ &= (\lambda_1\mathbf{v}\cdot\mathbf{e_1}, \lambda_2\mathbf{v}\cdot\mathbf{e_2}, \lambda_3\mathbf{v}\cdot\mathbf{e_3}) \end{aligned} \tag{27}$$

where M is the tensor matrix at a pixel, $\mathbf{v}$ is the user-defined input vector, and e_i is the unit-length eigenvector corresponding to the eigenvalue λ_i.

This way, hueballs may be used to assign a certain color to each tensor depending on the deviation suffered by the input vector, which at the same time is related to the anisotropy coded by the tensor [31]. This deviation is termed *deflection*.

We must note that because all the eigenvalues are non-negative, the resulting output vectors will remain in the same octant as the input vector. Therefore, the angle defined by $\mathbf{v}$ and $M\mathbf{v}$ will belong to the interval $\left[0, \frac{\pi}{2}\right]$. The higher the anisotropy is,

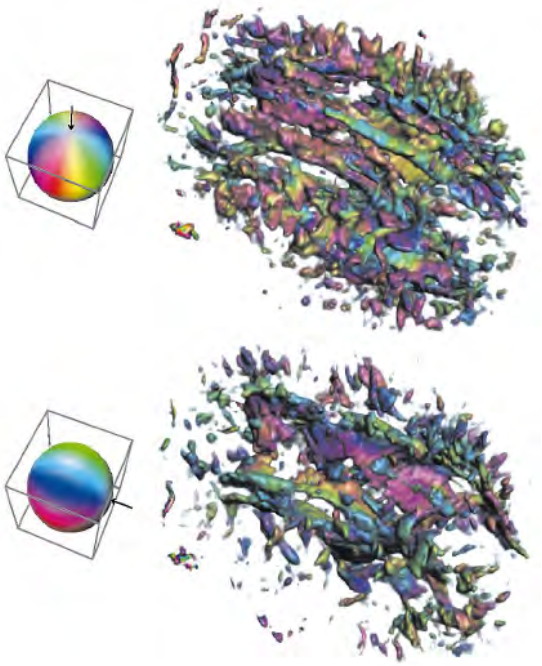


Fig. 22. DTMRI volume visualized with hueball coloring and opacity mapped by deflection ([31]).

the more accentuated deflection it will provoque. On the other hand, if $\mathbf{v}$ is orthogonal to any of the tensor eigenvectors, the contribution of that eigenvector to the tensor field will not be visualized. This is the reason why we should take different values for $\mathbf{v}$; if this is not done, important information about the tensor field could be lost.

In this method, deflection can be used to assign opacity to each tensor. In isotropic regions, the input vector suffers no deflection. Therefore, the deflection dependent function defined should give higher opacity values to points where the deflection angle is large. In Figure 22, we can observe diferent rendering of a DTMRI volume where hueball color and opacity mapping is employed. It is shown that different input vectors lead to changes in the visualization.

6.3.2. Lit-tensors

It is difficult to appreciate the topology of an object when it is rendered without light sources, because it lacks the lighting and shading that real three-dimensional surfaces have. Lit-tensors are a shading technique for tensors that also allows a better appreciation of the type and orientation of their anisotropy [31, 34]. To achieve this, the following are contemplated:

1. If the anisotropy is completely linear ($c_l = 1$), the lighting model will be identical to that of illuminated streamlines, which can be found in [32].

2. If the anisotropy is completely planar ($c_l = 1$), the lighting model will be the one used with traditional surface rendering, for example the Blinn-Phong model [35].

3. The transition between these two models must be regular and smooth: small variations in anisotropy must lead to small changes in shading. Said transition is controlled by a parameter c_θ, defined as:

$$c_\theta = \frac{\pi}{2}\frac{c_p}{c_l + c_p} = \frac{\pi(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 - 2\lambda_3} \tag{28}$$

In Figure 23 a volume is show with and without lit-tensor shading.

7. HARDWARE DEVICES

Hardware devices have the intention to facilitate and improve the possibilities for system interaction and/or visualization. The expectations created around this non-conventional system come from the information generated by virtual environments. In this document we have chosen some classification methods knowing that in a short future they will have to be reviewed because the continuous developments in this field.

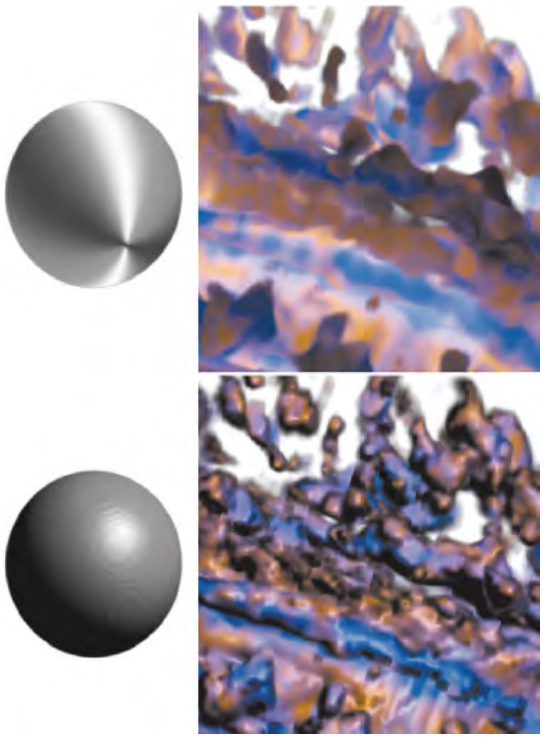


Fig. 23. Comparison between using littensors (up) and not using them (down) ([31]).

7.1. Terminology

The term *Tangible User Interfaces* (TUIs) [36] is defined by their authors as those user interfaces that increase the real physic world adding digital information to each object and space.

The term *Tangible* is applied to the object susceptible of being perceived through the sense of tact.

The concept *Embodiment* refers to the measured distance between the used input device and the output generation device. The distance may be zero in those cases where input happens over the output object. For example, in the making of a sculpture in the real world where a chinsel beat over the material generates a result.

According to how big this distance is, four different states can be defined:

- Input and output taking place at the same device, as in the previous example. This kind of interaction is the most common one in the real world (objects endure physical manipulation and change accordingly).
- The output happens near the input device. For example, an optic pencil acting over a screen alters the image shown in the latter.
- Output near user. This state was named *nongraspable* by Ullmer and Ishii [37], meaning that there is a very tenuous connection between the input and output devices. In conventional HCI[3] this state corresponds for example with sound edition tools.
- Output happening in some other place.

An application can belong simultaneously to more than one of these states.

Metaphor is one of the fundamental elements in the development of interaction devices. Device design allows the adoption of shape, form, color, weight and texture in spite of evoking metaphoric links to ideas or objects.

One of the most powerful metaphors is the one that allows vitality assignation to an inert object and that is exactly what happens with tangible interfaces.

As with the embodiment property, devices can also be classified in several metaphor levels simultaneously.

8. INPUT DEVICES

There is a distinction when talking about input devices and interaction techniques. Input devices are just physical tools employed to implement different interaction techniques.

[3] Human Computer Interaction.

Generally we could say that any interaction technique may be developed with almost any input device. The question is which device is more adequate to work with or to be used with a particular technique.

When talking about input devices it is convenient to talk about their degrees of freedom (DOF). For example devices known as trackers provide three values about the position and other three regarding orientation, which amounts to a total of 6 DOF. For the most part of the devices with fewer DOF, it is possible to emulate others with a larger number of them by adding buttons or keyboard keys [38].

8.1. Interaction

As previously mentioned, the embodiment concept tends to grow when the cognitive distance between input mechanism and the mechanism that reproduces the result decreases [36]. We will consider in a general manner that the reduction between user action and system response as a positive property because it contributes to a instinctive use of the devices.

8.1.1. Intuitive Use

Intuitive use refers to the necessity that hardware device handling becomes natural. One of the factors that makes the suitability of a device apparent may be measured by its transparency degree. The user must not realize that he/she is using the device nor learn codification or manipulation rules. A descent in the learning curve allows a notable increment in productivity.

The spontaneity in the behavior of the user with the system opens new possibilities regarding human-machine communications. As a result, the use and development of transparent interfaces has been adopted, trying that natural user movements have a response based in quotidian logic [39].

8.1.2. Functionality

Its goal is to establish a ranking of actual deficiencies of each system. We have to consider that nowadays user interaction systems are in their research phase. That is the reason why most of these systems are susceptible of being improved or even drastically changed in the technology used. Functionality is a basic exigence for systems that have to make sure that no technical or conceptual problems arise for the task it has being designed for. The problems coming from an ineffective design, software dysfuntions, sensor noise problems, etc. affect negatively the assessment of this property.

8.1.3. Complementary Contributions

It is necessary for the adoption of a new device that it improves to a higher degree the perception or communication with the system. The increase in perceptive quality and/or communicative quality can be undestood from the point of view of realism or the similarity with the physical world. This way, behaviour, intuitive gestures, and to mimic natural stimuli are approved.

8.1.4. Ergonomy

The device ergonomy and the work environment are important parameters to take into account as they affect directly the user's subjective valuations.

8.1.5. Intrusion

Intrusion as a parameter refers to user interfaces that have to be placed over the user's body or might obstruct certain degrees of gesticulation freedom. The user is generally aware of the devices adhered to his body. This perception implies a decrease of the user's immersion into the system capacities, at the same time that it could limit motor capacities (data gloves with feedback capacities, 3D positioning sensors by radiofrequency, Head mounted Display (HMD), etc. are some examples of it).

8.1.6. Noise

Noise in the information flux may endanger system functionality. These interferences are frequent in experimental use of sensors for new hardware interfaces. Filtering these perturbations constitutes a very important task.

On the other hand, it is often necessary to make modifications and restrictions for the workspace in which the user's interaction happens. The most reliable systems, and consequently those susceptible of a widespread use, are those that can adapt to any environment: this means avoiding or adapting to the noisy conditions of the environment, not being sensitive to perturbations, etc.

8.1.7. Standardization

The majority of the devices presently at use are on an experimentation phase and are susceptible of improvements or paradigm changes. The biggest problem at this point is that ideally these devices should be custom built and within the limits of their intended use.

8.1.8. Cost

Cost has its *raison d'être* in the fact that almost every hardware device has an exaggerate price. We have come to this conclusion examining electronic components prices in which the devices are integrated and taking into account the intrinsic difficulties of the software development for its use.

8.2. Hardware Input Devices

In this class we can include all the features or systems whose goal is to capture or to transmit the user's sensorial stimulus.

Input devices has the ability of detecting in real time user's gestures by means of different sensor technologies. Then, information is transmitted to the host.

There is a distinction that must be made when we talk about input devices and interaction techniques. Input devices are just the physical tools that are used to implement various interaction techniques. In general, may different interface techniques can be mapped onto any given input device. The question is how natural, efficient, and appropriate a certain input device will work with a given technique.

When talking about input devices it is convenient to talk about the degrees of freedom (DOF) that an input device has. For example, a device such as a tracker generally produces 3 position values and 3 orientation values for a total of 6 DOF. For the most part, a device with a smaller number of DOF can be used to emulate a device with a higher DOF with the addition of buttons or modifier keys.

A rough classification of input devices could be [38, 40]:

- Discrete.
- Continuous.
- Combos/Hybrid.
- Speech Input.

8.2.1. Discrete Input Devices

Discrete input devices simply generate one event at a time based on the user. In other words, when the user presses a button an event is generated which is usually a boolean value stating whether the button was pressed down or released. The keyboard is an obvious example of a discrete input device.

The Pinch Glove system developed by Fakespace [41] is another example of a discrete input device. These gloves had a conductive material at each of the fingertips so that when the user pinches two fingers together a electrical contact is made which generates a boolean value. There are many different pinching combinations that can be made which allows for a significant amount of input device to task mappings, which are listed below.

1. Keyboard

   Keyboard is the device which has suffered the most important standardization in their long existence so far.

   It is possible, in the future, that keyboards will not be so important due to the advances in speech recognition input algorithms.

2. DataGloves

   A *DataGlove* is a device used as an option to activate different events in Virtual Environments [42, 43]. These gloves had a conductive material at each of the fingertips so that when the user pinches two fingers together a electrical contact is made which generates a boolean value. There are many different pinching combinations that can be made which allows for a significant amount of input device to task mappings.

   The virtual tool belt is a custom built device which places cloth contacts along a belt that the user wears. The user can press the buttons on the belt to activate tools that he/she might use in the virtual environment application.

   This is a widely used device in virtual environments, to which sensors can be associated to measure each hand's position in real time, making a continuous or hybrid device out of this one, depending on what application it is being used for.

Fig. 24. Pinch Glove

The device can be more or less intuitive depending on which functions are implemented for which gesture. For example, if we join our thumb with our index finger, we could imitate a pincer for moving or rotating virtual objects. However, in other cases the relationship between action and gesture is not as straightforward and remembering each implemented function becomes necessary. Due to its high cost and in spite of being easy to manufacture, this is not a device positively rated by its users. Its use in specific applications must be considered, however, because it is possible to manufacture it at a very low cost.

8.2.2. Continuous Input Devices

Continuous input devices generate a continual stream of events in isolation (no user manipulation) or in response to user action. For example, a tracker is a device which will continually output position and orientation records even if the device is not moving. These types of devices are important when we want to know where something is in the virtual space and we do not want to have to keep asking for it.

A perfect example of this is head tracking. Two of the most common continuous devices are trackers and datagloves. Data gloves transmit records on the bend angles of the fingers.

Trackers

One of the most important aspects of 3D interaction in virtual worlds is providing a correspondence between the physical and virtual environments. As a result, having accurate tracking is extremely important to making the VE usable. Currently there are a number of different tracking technologies in the marketplace. The different types of trackers are: magnetic, mechanical, acoustic, inertial, vision/camera and hybrids.

Next, we will go over each type of technology [38].

1. Magnetic Trackers

   Magnetic tracking uses a transmitting device that emits a low frequency magnetic field that a small sensor, the receiver, uses to determine its position and orientation relative to a magnetic source. These trackers can use extended range transmitters which increase the range of the device from around an 8 foot radius to anywhere from a 15 to 30 foot radius. The tracker shown in Figure 25 is called the Ascension MiniBird. It uses a smaller emitter and receivers and has better accuracy than the regular system. However, its range is limited to about a 4 foot radius. It is primarily used in medical applications where range of the device is not a factor.

Fig. 25. Ascension MiniBird

2. Mechanical Trackers

   Mechanical trackers have a rigid structure with a number of joints. One end is fixed in place while the other is attached to the object to be tracked (usually the user's head). The joint angles are used to obtain position and orientation records.

3. Acoustic Trackers

Acoustic tracking devices use high frequency sound emitted from a source component that is placed on the hand or object to be tracked. Microphones placed in the environment receive ultrasonic pings from the source components to determine their location and orientation.

In most cases, the microphones are placed in a triangular fashion and this region determines the area of tracked space. One of the most interesting problems with this type of tracking is that certain noises such as jingling keys or a ringing phone will interfere with the device.

Fig. 26. Logytech Fly Mouse

One of the main issues of these devices is that there are some noises that can produce interfernces, like the jingle of keys or a telephone ringing.

4. Inertial Trackers

   Inertial tracking systems use a variety of inertial measurement devices such as gyroscopes, servo accelerometers, and micro-machined quartz tuning forks. Since the tracking system is in the sensor, range is limited to the length of the cord which attaches the sensor to the electronics box as can be seen in Figure 27.

   Two of the big limitations of these devices is that they only track orientation and are subject to error accumulation. The InterSense IS300 handles error accumulation by using a gravitometer and compass measurements to prevent accumulation of gyroscopic drift and also uses motion prediction to predict motion up to 50 milliseconds into the future [44].

Fig. 27. Intersense IS300

5. Camera or Vision Trackers

   Camera/vision based tracking takes one or more cameras and places them in the physical environment. The cameras then grab video of the user or object to be tracked. Usually image processing techniques, such as edge detection algorithms, are used to identify the position and/or orientation of various body parts such as the head and hands. Setting up vision-based tracking systems can be difficult since there are many parameters that must be fixed in order to track the user properly. These parameters include the number of cameras, the placement of the camera, what background (what is in back of the user) is put up, and if the user will be wearing special optical tools such as LEDs or colored gloves to aid in tracking.

6. Hybrid Trackers

   Hybrid trackers attempt to put more than one tracking technology together to help increase accuracy, reduce latency, and, in general, provide a better virtual environment experience. The device shown in Figure 28 is the InterSense IS600. It combines inertial and ultrasonic tracking technologies which enables the device to attain 6 DOF. The major difficulty with hybrid trackers is that the more components added to the system, the more complex the device becomes.

Fig. 28. Intersense IS600

7. DataGloves

   Data gloves measure finger movement of the hand by using various kinds of sensor technology. These sensors are embedded in the glove or placed on top of the glove, usually on the back of the hand. The number of sensors in the glove depends on the manufacturer.

   Virtual Technologies' CyberGlove has either 18 or 22 sensors which can measure at least 2 joints in each finger, wrist roll and yaw, and others. These types of gloves are commonly used for hand gesture and posture recognition which can be applied to a variety of different interface techniques in virtual environments.

   Fifth Dimension Technologies (5DT) offers gloves that have either 5 sensors, one for each fingertip or 16 sensors, 2 for each finger and abduction between fingers. 5DT also has wireless versions of each glove [39, 43].

Fig. 29. VPL DataGlove

8.2.3. Combination or Hybrid Devices

A combination/hybrid input device combines both discrete and continuous event generating devices to form a single device that is more flexible. Two of the most common hybrid devices are the joystick and mouse. Another device in this category is the pen-based tablet [45]. Pen-based tablets are becoming more and more popular in virtual environment applications because they give the user the ability to interact in 2D which provides a useful combination in certain interfaces.

Another type of input devices are isometric which have a large spring constant so they cannot be perceptibly moved. Their output varies with the force the user puts on the device. A translation isometric device is pushed while a rotation isometric device is twisted. A problem with these devices is that users may tire quickly from the pressure they must apply in order to use them.

The BAT is a device that was developed by Colin Ware in the late 1980's. It is essentially just a tracking device with three buttons attached to it. It is similar to the other 3D mice mentioned in the previous slide except it is rather easy to build one with a few electrical components (provided you have the tracking device).

The Wand is a device that is commonly seen in surround-screen virtual (SSVR) environments. It is simply a more elegant version of the BAT that is commercially developed.

Fig. 30. Ring Mouse

The Flex and Pinch input system is a custom built device which takes the functionality of the Pinch Glove system and combines it with the bend sensing technology of a data glove. The pinch buttons are made from conductive cloth and can be placed anywhere on the bend sensing glove, as can be seen in Figure 24.

ShapeTape is a continuous bend and twist sensitive strip which encourages two-handed manipulation. A BAT is attached and the tool is used for creating and editing curves and surfaces along with camera control and command access. ShapeTape [46] senses bend and twist with two fiber optic sensors at 6cm intervals.

The Cubic Mouse [47, 48] is an input device developed at GMD that allows users to intuitively specify three-dimensional coordinates in graphics applications. The device consists of a box with three perpendicular rods passing through the center and buttons for additional input.

Figure 31 shows a picture of the SpaceOrb [49], an isometric device from Labtec priced at approximately forty dollars.

Fig. 31. SpaceOrb

8.2.4. Speech Input

Speech input provides a nice complement to other input devices. As a result, it is a natural way to combine different modes of input (e.g. multimodal interaction) to form a more cohesive and natural interface. However, there are many issues to consider when dealing with speech input besides what speech recognition engine to use.

There are tradeoffs that must be made when dealing with speech input. An important issue is where the microphone is to be placed. Ideally, a wide area mike would be best so that the user does not have to wear a headset. Placing such a microphone in the physical environment could be problematic since it might pick up noise from other people or machines in the room.

Other important issues that must be addressed are whether or not the user must signal the speech recognizer when he/she is about to speak and whether to have a speaker independent or dependent recognizer.

In general, when functioning properly speech input can be a valuable tool in virtual environments especially when both of the user's hands are occupied.

There are many other issues to consider:

- continuous vs. one-time recognition.
- choice and placement of microphone.
- training vs. no training.
- handling of false positive recognition.

- surrounding noise interference.

8.2.5. Device Parameters

A parameter is any characteristic of a device or its interface which can be tuned or measured along a continuum of values. Input parameters are the sorts of features controlled or determined one way or another by designers or by system characteristics [39].

Some parameters, such as mass, resolution, or friction, are "in the device" or its electronics, and can be designed in, but cannot be tuned thereafter. Others exist in the interface software or system software. Examples are sampling rate or the control-display relationship. Still others exist through a complex weave of transducer characteristics, interface electronics, communications channels, and software. Lag or feedback delay is one such example.

Although some parameters can be adjusted to improve performance, others are simply constraints. Resolution, sampling rate, and lag are parameters with known optimal settings. Resolution and sampling rate should be as high as possible, lag as low as possible. Obviously, these parameters are constrained or fixed at some reasonable level during system design. Although typical users are quite unconcerned about these, for certain applications or when operating within a real-time environments, limitations begin to take hold.

8.2.6. Resolution

Resolution is the spatial resolving power of the device/interface subsystem. It is usually quoted as the smallest incremental change in device position that can be detected (e.g., 0.5 cm or 1); however, alone the specification can be misleading.

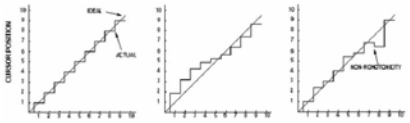


Fig. 32. Resolution. a) The ideal case for 1 unit of resolution over 10 units of movement. b) non-linearity is introduced showing degraded resolution at the extremes of movement than in the center. c) Non-monotonicity occurs when the output fails to increase uniformly with the input.

Touch screens, tablets, and other devices using finger or stylus input have apparent resolution problems since it is difficult to resolve finger or stylus position to the same precision as the output display (a single pixel). In fact the resolving power of the input device often exceeds that of the output device; however the input/outut mapping is limited by the contact footprint of the input device (e.g., the width of the finger tip).

3D trackers have a resolution better than 1 inch and 1 degree, but this varies with the proximity of the sensor to the source and other factors. Resolution often sounds impressive in specification sheets; but when application demands increase, such as widening the field of use or combining two or three trackers, limitations become apparent. The specification sheets of 3D trackers are surprisingly sparse in their performance details. Is the resolution cited a worst-case value over the specified field of use, or is it "typical"? How will resolution degrade if two or more trackers are used? What is the effect of a large metal object five meters distant? These questions persist.

Resolution will constrain a variety of applications for 3D virtual worlds.

8.2.7. Sampling Rate

Sampling rate is the number of measurements per unit time (e.g., samples per second) in which the position of the input device is recorded. It is the input analog of the refresh rate for updating the output display. The sampling rate begins to constrain performance when input or output motion is quick.

Typical rates are from 10 to 100 samples per second. This is fine for many applications, but may be inadequate for real-time 3D environments which must sense and respond to the natural, sometimes rapid, motions of humans. A high sampling rate is essential in 3D virtual worlds because they are dynamic environments. In the mouse for example, the back-and-forth motion of the the hand and cursor will not exceed about 10 cm. If the controlling device is a 3D tracker attached to the head, similar back-and-forth motion (a few cm at the nose) will translate into very large movements in the visual space in very small time intervals. Smooth viewing motion necessitates a high sampling rate with immediate display refreshes.

8.2.8. Lag

Lag is the phenomenon of not having immediate updates of the display in response to input actions. High sampling and refresh rates are wasted if the re-drawn viewpoint does not reflect the immediately preceding sample, or if the sample does not capture the immediate habit of the user. One reason this may occur is the drawing time for the scene. If the image is complex (e.g., thousands of texture-mapped polygons with 24-bit color), then drawing time may take two or three (or more) sampling periods.

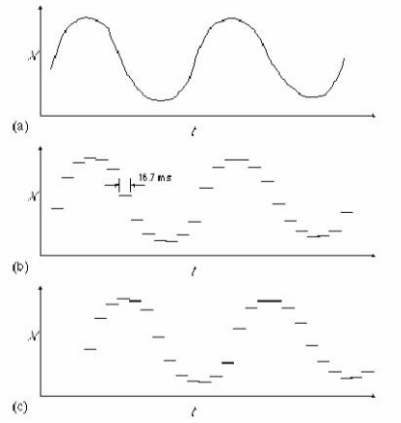


Fig. 33. Sampling rate and lag. a) Sinusoidal back-and-forth motion of a mouse at 6 Hz is shown. b) At 60 Hz. sampling, the cursor appears. c) With three samples of lag, cursor motion is delayed as in c

3D trackers have significant lag, in the range of 30 ms to 250 ms (depending on how and where measurements are made). Furthermore, the source of the lag is not always obvious and is difficult to measure. The sampling rate for input devices and the update rate for output devices are major contributors; but lag is increased further due to "software overhead", a loose expression for a variety of system-related factors. Communication modes, network configurations, number runching, and application software all contribute. Since lag is on the order of a few hundred milliseconds in virtual environments, its effect on user performance is less apparent.

The communication link between the device electronics and the host computer may prove a bottleneck and contribute to lag as the number of sensors and their resolution increases. The CyberGlove by Virtual Technologies provides greater resolution of finger position than many gloves by including more sensors–up to 22 per glove. However, correspondingly more data are required. At the maximum data rate of 38.4 kilobaud, it takes about 5 ms just to relay the data to the host. Alone this is trivial, however a trade-off is evident between the desire to resolve intricate hand formations and the requisite volume of "immediate" data. If we speculate on future interaction scenarios with full body suits delivering the nuances of complex motions (a common vision in VR) then it is apparent that lag will increase simple due to serial bias in the communications link. Since technological improvements can be expected on all fronts, lag may become less significant in future systems.

9. OUTPUT DEVICES

9.1. Visual Displays in Virtual Environments

Two of the main properties that are often used to describe virtual environments are the *suspension of disbelief* and whether there is a *viewer-centered perspective* [50]. Suspension of disbelief is defined in [50] as the ability to give in to a simulation, ignoring its medium. A viewer-centered perspective makes the perspective of the visualization system be the same as that of the user's. This is important in order to achieve a full suspension of disbelief, and it is usually implemented thanks to a sensor that tracks the user's position [50].

Kjeldskov suggests the notion of available field of view, defined as the field of view available to the user in any given viewing direction, as a means to measure immersion [51]. Taking this into account, virtual displays can be classified into fully and partial immersive displays. The former always provide an available field of view, whereas the latter do not.

9.1.1. Fully Immersive Displays

Head mounted displays (HMD) consist of a stereo pair of small displays that cover the eyes. It also has a head-tracking device whose purpose is to inform of the user's perspective, so that the view it provides is synchronized to the latter. This device prevents the user from seeing the real world, which helps increase his/her feeling of immersion in the virtual world. It can be seen in Figure 34(a). Some disadvantages are that its field of vision is poor, and therefore, it does not take advantage from peripheral vision. Also, its weight is an impediment in order to achieve a fully suspension of disbelief. Another technology on fully immersive displays are Virtual Retina Displays (VRD) or Retinal Scan Displays, which scan images directly into the user's retina. This is a field being currently investigated. The aim is to be able to build devices similar to the model shown in Figure 34(b).

Another example of fully immersive devices are Binocular Omni-Oriented Monitors (BOOM). These devices are suspended from an articulated arm, and the user must position the BOOM manually [50]. Like HMDs, the peripheral vision is not employed, and the user has limited movement. However, it usually provides a better field of view, and the user does not need to hold the BOOM weight.

(a) (b)

Fig. 34. Fully immersive displays. (a) Head Mounted Display. (b) Virtual Retina Display.

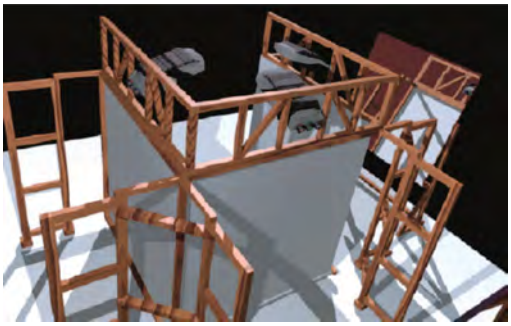


Fig. 35. A schematic of the CAVE.

9.1.2. Partially Immersive Displays

While fully immersive displays support the feeling of being a part of the virtual environment, in partially immersive displays the user has the sensation of looking at it from the outside [51]. Some devices that belong to this group are panoramic displays, stereo monitors, etc.

9.1.3. The CAVE

The CAVE is an acronym for Audio-Visual Experience Automatic Virtual Environment. It is composed of three to six cube faces surrounding the viewer. These faces are translucent display screens, where stereo images are projected [50, 52]. This device can be seen in Figure 35. A head-tracking device must be included, in order to adapt the projections to the user's perspective.

Depending on how many faces there are, it can be classified as either full immersive (six faces) or partially immersive (three to six faces). This device also has the advantage that more than one person can use it at the same time. However, the perspective will only be modified according to the user that is wearing the tracker. It also takes advantage of peripheral vision. However, its dimensions are very large, and it is very expensive.

9.2. Haptic and Tactile Feedback

Haptics represents a critical component in virtual environment interaction. Allowing a user to touch and feel in the virtual world in the same way that they do in the physical world is extremely powerful. Unfortunately, haptic and tactile output device research is still in its early stages.

There are essentially four different methods in which haptic and tactile feedback is generated. The first method is ground-referenced feedback, which creates a physical link between the user and ground with the feedback relative to a single contact point. The second method is body-referenced feedback which places a device on some part of the user's body. An example of a body-referenced haptic device is the Virtual Technologies CyberGrasp which is shown in Figure 36. The third method for generating feedback is tactile which uses some type of oscillatory or vibrating device to stimulate the user's tactile sense. Finally, the last method of generating feedback is via dermal tactile which stimulates the user's nerves in the fingertips [42, 53].

Haptic feedback from devices like the CyberGrasp are very good for grabbing objects and moving them around and they can provide a limited form of real world interaction. The main problem with these devices is that they are somewhat intimidating to the user. People are commonly afraid to put these devices on and once they do they're afraid they'll break them or get injured [54].

Fig. 36. TactileFeedback

For example, some of the most exciting work explores tactile feedback in 3D interfaces [55] modified the DataGlove by mounting piezoceramic benders under each finger. When the virtual fingertips touched the surface of a virtual object, contact was cued by a "tingling" feeling created by transmitting a 20-40 Hz sine wave through the piezoceramic transducers. This is a potential solution to the blind touch problem cited above; however providing appropriate feedback when a virtual hand contacts a virtual hard surface is extremely difficulty.

Others authors [56] confronted the same problem: Even in a linear analog system, there is no force applied until the probe has overshot [and] penetrated the virtual surface. The system has inertia and velocity. Unless it is critically damped, there will be an unstable chatter instead of a solid virtual barrier. They added a brake (a variable damping system) and were able to provide reasonable but slightly "mushy" feedback for hard surface collision.

It is interesting to speculate on the force equivalent of control-display (C-D) gain. Indeed, such a mapping is essential if, for example, input controls with force feedback are implemented to remotely position heavy objects. The force sensed by the human operator cannot match that acting on the remote manipulator, however. Issues such as the appropriate mapping (e.g., linear vs. logarithmic), thresholds for sensing very light objects, and learning times need further exploration.

10. SOFTWARE INTERFACES FOR TENSOR FIELDS

To our knowledge, there is little in the way of software oriented to user interfaces for second-order tensor fields. Wünsche developed a toolkit for visualizing tensor fields [57], and VTK [58] is a popular visualization toolkit that offers methods for tensor visualization.

As for medical applications that deal with tensor images, we must mention the following ones:

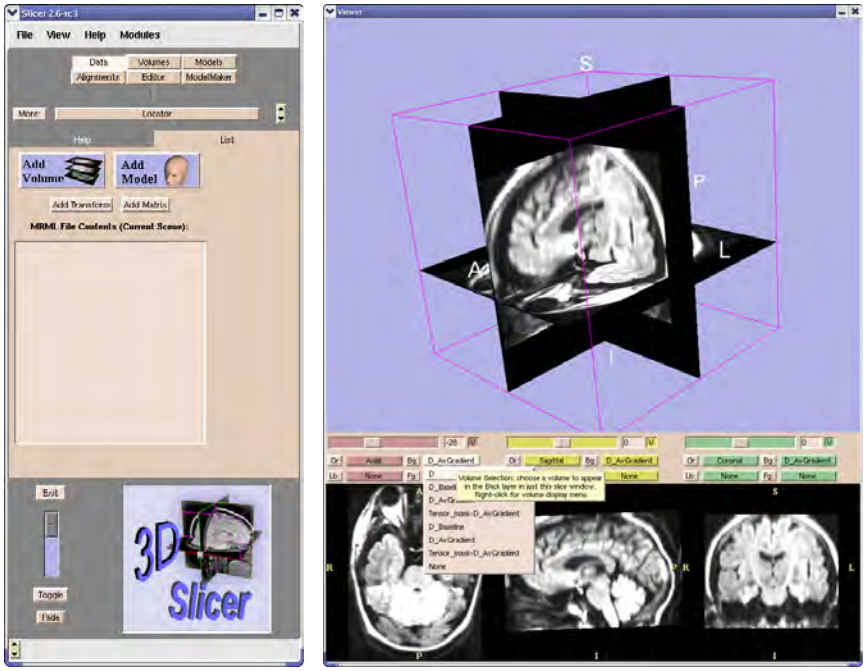
- 3D Slicer [59–61].
- Bioimage Suite [62], developed at the University of Yale.
- BioTensor [63].
- MedINRIA [64], developed by Pierre Fillard and Nicolas Toussaint, members of the INRIA research team Asclepios at Sophia Antipolis.
- Teem [65], by Gordon Kindlmann.
- A DTMRI add-on for Analyze 7.0 [66].
- A diffusion plugin written for the Neurolens application [67].
- An immersive virtual environment for DTMRI visualization, described in [68].

10.1. 3D Slicer

Now we will evaluate an application able to work with DTMRI images: *3D Slicer* [59–61]. This application was developed by the MIT AI Laboratory [69] and Brigham and Women's Hospital at Harvard [70], and its current development is mainly supported by the National Alliance for Medical Image Computing (NA-MIC) [71]. Version 3 is expected to be available in January 2007 [72].

3D Slicer is a software environment oriented to image-guided medicine, in particular for pre-operative planning, surgical guidance, and diagnostic visualization [60]. It also allows volume data visualization of different kinds of medical image modalities, and their segmentation and registration. This package has a modular and extensible architecture, and is portable to several operative systems.

One of the available modules for 3D Slicer is a module for diffusion tensor image visualization, that allows the user to load a DTMRI volume and process it, in order to get some image characteristics, like eigenvalues or anisotropy coefficients, and a number of methods for tensor visualization already seen in the present document, like glyphs or hyperstreamlines.



(a) (b)

Fig. 37. (a) Main window and (b) viewer window of 3D Slicer.

10.1.1. 3D Slicer and DTMRI Module Interfaces

The application interface is divided in three windows: a main window, a viewer window, and a command console. The first two can be observed in Figures 37(a) and 37(b), respectively. The preferred input device is a conventional mouse, although sometimes the keyboard is needed. It is nonetheless possible to use it in conjunction with other devices.

Looking at Figure 37(a), we can see how the main window is divided. It has a menu bar at the top of the window, some buttons, each for a Slicer 3D module, a frontal panel divided into tabs, and at the bottom there is a small window with controls for the viewer window, although it is not always active. A study on website design [73] for placement and presentation of navigation menus showed that top and left placements for navigation elements, and the tabbed navigation style are the preferred ones of the users. Therefore, the layout of 3D Slicer follows these directives.

The viewer window showed in Figure 37(b) is divided into a window for 3D volume visualization and and three small windows, each for an axial, sagittal or coronal slice. Nonetheless this layout can be changed in the main window. It can be seen that there are some user controls in the viewer window that help the navigation throughout the volume, changing the point where the slices intersect, allowing to show or hide slices in the visualization, or choosing what volume to represent. A good point to help users is that a tooltip box is showed when the pointer is over an element a certain amount of time, as can be seen in Figure 37(b). The user can zoom the volume in and out and also rotate it using the mouse.

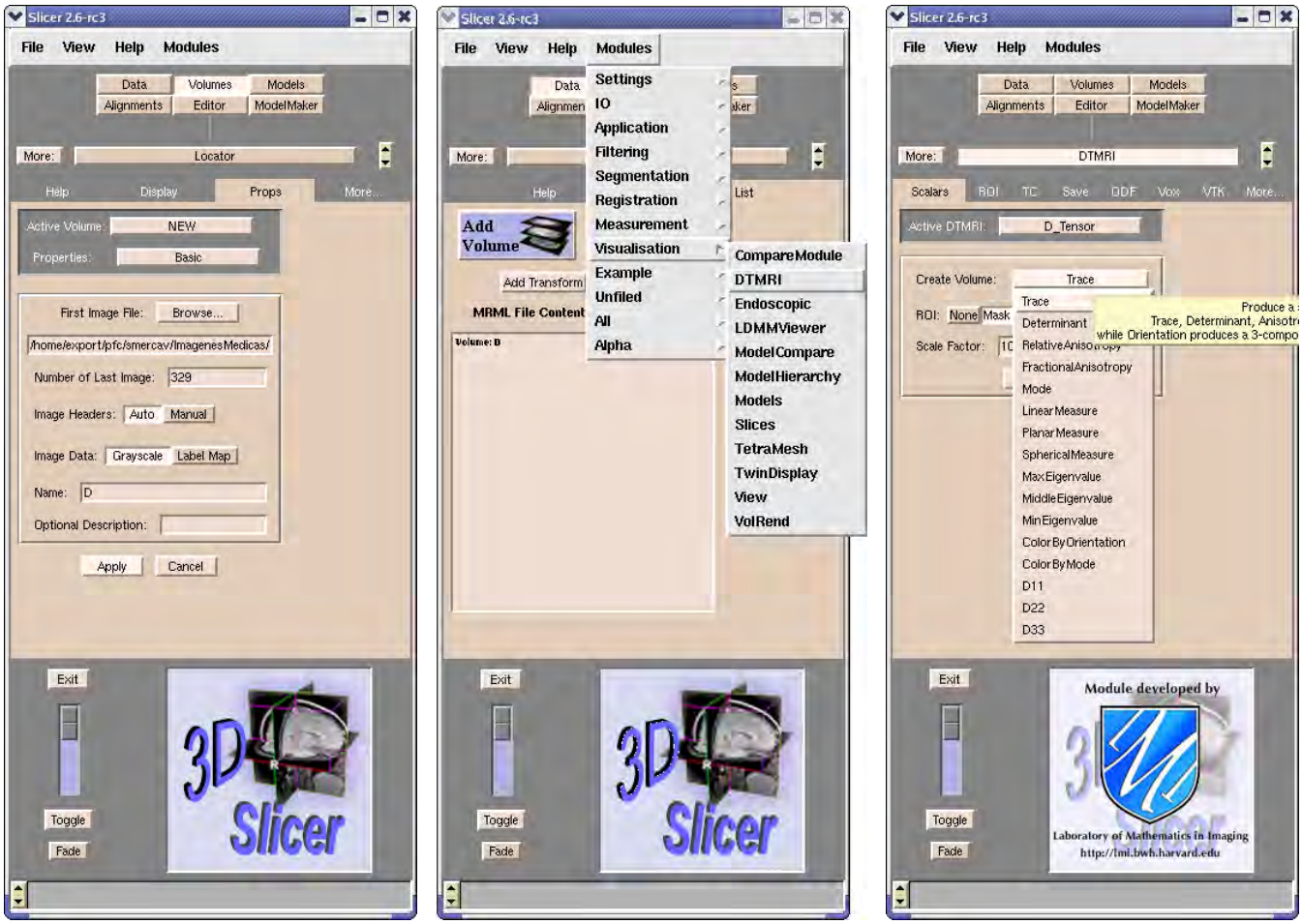
In Figure 38(a) the interface for loading volumes is shown. In Figure 38(b) we can see how to reach th DTMRI module through the menu, and in Figure 38(c), the diferent options of the DTMRI module for computation of some of the tensor characteristics are shown.

If we want to select a region of interest, the actions we must follow are complicated. After we choose to start editing we must go to the Editor panel showed in Figure 39 and click on the Draw button. Then, we must select a color for the contour and then draw it on one of the 2D slices. A selected contour as ROI can be seen in Figure 40. The conclusion is that the method for selecting a region of interest is quite complicated, especially for new users, and not very intuitive. Slicer offers ways to make calculations on these regions using another module called VolumeMath.

Finally we try one of the visualization methods offered by the DTMRI module: hyperstreamlines, for which we will use the Tract panel. We will make a tractography employing user-selected seed points, which are taken from the position of the mouse cursor while the key “S” is pressed. Once a seed point is selected, the corresponding tract/hyperstreamline is computed. In Figure 41 the visualization of the obtained results can be observed.

10.1.2. 3D Slicer: User Feedback

A survey about 3D Slicer was conducted from March 2002 to November 2004, whose results can be accessed at [61]. Looking at the answers given by the users can be helpful to determine how effective is the user interface of this application. Nevertheless, we must point out that there are factors in interface design that users only notice when they are absent [73]; for example, that



(a) (b) (c)

Fig. 38. Slicer main window layout: (a) Loading a volume. (b) Selecting the DTMRI module. (c) Choosing which tensor scalar magnitude to compute.

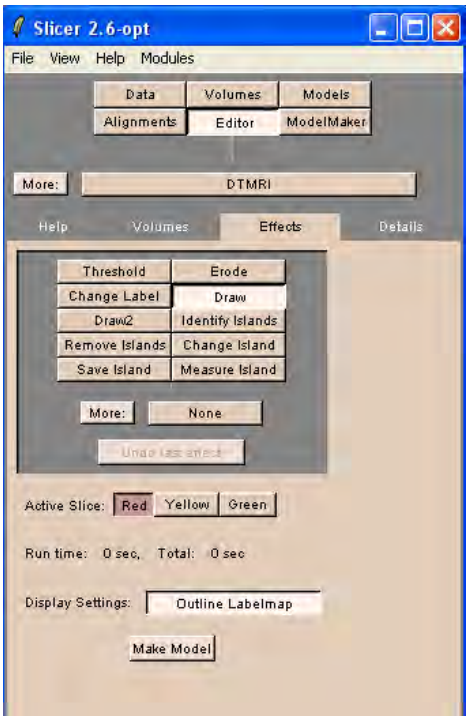


Fig. 39. Detail of the Editor panel.

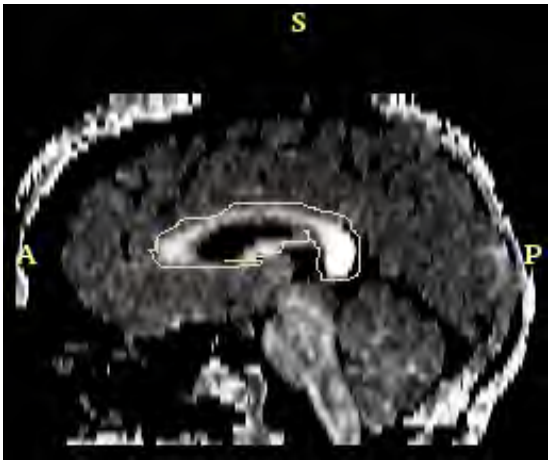


Fig. 40. Contour selected as region of interest.

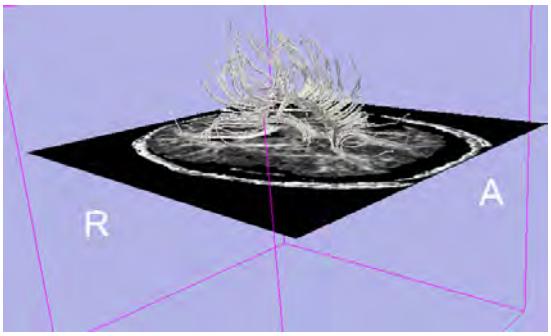


Fig. 41. Tracts computed with the DTMRI module.

every function in the interface works properly. Therefore, this kind of features is unlikely to appear on the survey unless they are unsatisfactory to the participants.

One of the most repeated comments is that 3D Slicer, at least for lower versions than 2.0, needed a better graphic user interface (GUI), with a cleaner appearance. Also, the addition of pictures and tooltips to the buttons was desired, which as we have seen is solved in version 2.6. A user found the menus too long. Also, pop-up windows were disliked for entering data and should only be used for errors and warnings. In the current version these windows are also employed for showing measurement results. It was noted, however that sometimes the GUI was unresponsive. Furthermore, a wide variety of bugs were reported.

The user interface to the data visualization (viewer window) was one of the most liked features, as it was displaying the pixel map and the model on a single screen. Interaction between the three 2D dataset views would be appreciated, though. It was also found useful the ability to overlay labelmaps onto gray scales and to rotate 3D volumes. A user wished for a memo function that allowed to put notes on 2D slices. Interactive volume rendering was a desired new feature.

One of the most often mentioned themes was the drawing features. In general, many users wanted more and better tools for this purpose, like for example the ability to draw on the surface anatomy. However, the fact that editing needed multiple layers was disliked. On the other hand, people complained about the measurement tools of Slicer. In particular, measuring distances was difficult and time consuming.

A certain comment attracted our attention. Someone admitted not using most of the available features because he was relatively new to Slicer 3D. However, he/she was using 3D Slicer since almost a year ago. In fact, most of the participants said that they only worked with a few modules. This shows that 3D Slicer has a large range of functionalities, but many of them are unused. Users also wished for a full tutorial and a user-friendly manual.

10.2. Immersive Virtual Environment for DT-MRI Volume Visualization

There are reports of using virtual immersive environments with tensor field representations. In [15, 68] the use of the CAVE for tract visualization is reported. Thanks to the CAVE, more than one person can use the virtual environment concurrently. In Figure 42 a DT-MRI volume is visualized in the CAVE using streamtubes and streamsurfaces.

A wand was employed as input device to interact with the virtual world. A ray was simulated as if it came out from the wand, with the same orientation and direction as the latter. This ray stopped when it hit an object [68]. The user could also interact with the system by leaning toward the represented DTMRI volume, which made the visualization grow larger [68]. This method was

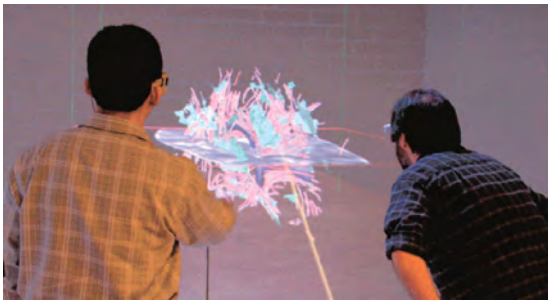


Fig. 42. DTMRI visualization in the CAVE ([68]).

chosen to emulate the natural human behavior when a closer look from an object is wanted.

There is a compromise between visual fidelity and interactivity. When the level of detail is high, the rate of frames per second that can be displayed decreases, diminishing the interactive capacity of the system. This is solved using different degrees of resolution in the different actors in the visualization, prioritizing the objects of more interest to the user [68]. The neural structures (represented in this case study by streamtubes and streamsurfaces) are given high priority, whereas the anatomical landmarks resolution vary depending on the goal pursued by the visualization. For example, if the visualization goal is brain tumor surgery, the blood vessels surrounding the tumor will be given higher priority [68].

It was reported that the doctors and medical students that worked with the system found it easy to use, that the learning time was just a few minutes, and that the system provided a better understanding of the 3D structures than a desktop 2D representation [15]. These systems are, however, very expensive and can be quite seldomly accessed.

11. CONCLUSIONS

Nowadays with the calculus power provided by computer, tensor field manipulation has become a reality. The equations and the theory were developed a long time ago, but so far a common use has not been possible. A tensor provides a great deal of information, but for the same reason, making good user interfaces to interact with these fields is a difficult task, which is being strongly researched and developed.

One of the most active fields related to this purpose is tensor visualization. We have seen a large number of methods for representing tensor fields. All of them have advantages and drawbacks, so each are more adequate to certain tasks than the others. For example, the natural way for viewing tractographies are hyperstreamlines and streamtubes. On the other hand, if some characteristic, for example the linear anisotropy, must be represented in the whole volume, texture techniques like HyperLIC or a renderization with hueballs or barycentric mapping should be employed. Not every tensor field visualization technique is able to convey all the information available in the field.

One of the issues in glyph visualization and tractographies is to choose the number and position of the elements to be represented. If this is not done, we might have cluttered visualizations, which would be confusing to the user. However, if there is a small number of them, the understanding of the tensor field is diminished.

There are not too many implemented user interfaces for tensor visualization. One of them is Slicer 3D, which has a module for DTMRI visualization. Although users are satisfied overall with it, they have reported that a better graphic user interface should be provided, and that there is a large part of the available functions that they do not use. They would also like to interact more with the data.

The input devices needed in Slicer 3D are a standard mouse and keyboard. There are other input devices that would be very useful to interact with tensor fields given that they offer more degrees of freedom and its use is more intuitive to the user. However, their cost is high and many of them are in an experimentation phase.

Tensor user interfaces are being strongly researched and developed. Future evolution for this field is quite optimistic since previsions augur an exponential increase on tensors fields use.

Acknoledgments

Thanks to Anders Brun for his input on using color to unclutter tract visualization. The authors acknowledge the Comisión Interministerial de Ciencia y Tecnología for research grant TEC2004-06647-C03-01, the Fondo de Investigaciones Sanitarias for grant PI-041483, the Junta de Castilla y León for grant VA075A05 and the European Commission for the funds associated to the Network of Excellence SIMILAR (FP6-507609).

12. REFERENCES

[1] Zhenhua Hu, Dimitris Metaxas, and Leon Axel. In vivo strain and stress estimation of the heart left and right ventricles from mri images. *Medical Image Analysis*, 7(4):435–444, December 2003.

[2] E. McVeigh and C. Ozturk. Imaging myocardial strain. *Signal Processing Magazine, IEEE*, 18(6):44–56, 2001.

[3] Daniel Goldberg-Zimring, Andrea U. J. Mewes, Mahnaz Maddah, and Simon K. Warfield. Diffusion Tensor Magnetic Resonance Imaging in Multiple Sclerosis. *J Neuroimaging*, 15(4_suppl):68S–81, 2005.

[4] I.-F. Talos, L. O'Donell, C.-F. Westin, S. K. Warfield, and W. Wells III anad S.-S. Yoo et al. Diffusion Tensor and Functional MRI Fusion with Anatomical MRI for Image Guided Neurosurgery. In *Proceedings of the 6th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2003)*, 2003.

[5] M. Kubicki, C.-F. Westin, R. W. McCarley, and M. E. Shenton. The Application of DTI to Investigate White Matter Abnormalities in Schizophrenia. *Annals of New York Academy of Sciences*, 1064:134–148, 2005.

[6] André Lichnerowicz. *Elements de Calcul Tensoriel*. Casa Librairie Armand Colin, 1962.

[7] Eric W Weisstein. Determinant. From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/Determinant.html`. Visited the 24 november 2006.

[8] David C. Kay. *Cálculo Tensorial*. MacGraw-Hill, 1989.

[9] C.-F. Westin, S.E. Maier, H. Mamata, A. Nabavi, F.A. Jolesz, and R. Kikinis. Processing and visualization for diffusion tensor MRI. *Medical Image Analysis*, 6:93–108, 2002.

[10] Denis Le Bihan, Jean-François Mangin, Cyril Poupon, Chris A. Clark, Sabina Pappata, Nicolas Molko, and Hughes Chabriat. Diffusion tensor imaging: Concepts and applications. *Journal of Magnetic Resonance Imaging*, 13(4):534–546, 2001.

[11] D.H. Laidlaw, E.T. Ahrens, D. Kremers, M.J. Avalos, R.E. Jacobs, and C. Readhead. Visualizing diffusion tensor images of the mouse spinal cord. In *Visualization '98. Proceedings*, pages 127–134, 527, 1998.

[12] G. Kindlmann. Superquadric Tensor Glyphs. In *Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization*, 2004.

[13] W.C. de Leeuw and J.J. van Wijk. A probe for local flow field visualization. In *Visualization, 1993. Visualization '93, Proceedings., IEEE Conference on*, pages 39–45, 1993.

[14] David Weinstein, Gordon Kindlmann, and Eric Lundberg. Tensorlines: Advection-diffusion based propagation through diffusion tensor fields. *IEEE Visualization '99*, pages 249–253, 1999.

[15] Song Zhang, Çagatay Demiralp, and David H. Laidlaw. Visualizing Diffusion Tensor MR Images Using Streamtubes and Streamsurfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):454–462, 2003.

[16] Thierry Delmarcelle and Lambertus Hesselink. Visualizing second-order tensor fields with hyperstreamlines. *IEEE Computer Graphics and Applications*, 13(4):25–33, 1993.

[17] Hans-H. Ehricke, Uwe Klose, and Wolfgang Grodd. Visualizing mr diffusion tensor fields by dynamic fiber tracking and uncertainty mapping. *Computers & Graphics*, 30:255–264, 2006.

[18] W. Bengar and H.-C. Hege. Tensor splats. *Conference on Visualization and Data Analysis 2004*, 2004.

[19] Xiaoqiang Zheng and Alex Pang. Volume deformation for tensor visualization. *IEEE Visualization*, pages 379–386, 2002.

[20] A. Brun, H.-J. Park, H. Knutsson, and C.-F. Westin. Coloring of DT-MRI fiber traces using Laplacian eigenmaps. In *Proceedings of the Ninth International Conference on Computer Aided Systems Theory (EUROCAST)*, 2003.

[21] E. Boring and A. Pang. Interactive deformations from tensor fields. In *Visualization '98. Proceedings*, pages 297–304, 545, 1998.

[22] Xiaoqiang Zheng and Alex Pang. Topological Lines in 3D Tensor Fields. *IEEE Visualization 2004*, pages 313–320, 2004.

[23] Xiaoqiang Zheng, Beresford N. Parlett, and Alex Pang. Topological Lines in 3D Tensor Fields and Discriminant Hessian Factorization. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):395–407, 2005.

[24] Topological lines in 3D Tensor Fields. `http://www.cse.ucsc.edu/research/avis/tensortopo.html`. Visited the 30th october 2006.

[25] Andreas Sigfridsson, Tino Ebbers, Einar Heiberg, and Lars Wigström. Tensor field visualization using adaptive filtering of noise fields combined with glyph rendering. *IEEE Visualization 2002*, pages 371–377, 2002.

[26] Brian Cabral and Leith Casey Leedom. Imaging vector fields using line integral convolution. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 263–270, New York, NY, USA, 1993. ACM Press.

[27] L.K. Forssell and S.D. Cohen. Using line integral convolution for flow visualization: curvilinear grids, variable-speed animation, and unsteady flows. *Visualization and Computer Graphics, IEEE Transactions on*, 1(2):133–141, 1995.

[28] V. Interrante and C. Grosch. Strategies for effectively visualizing 3D flow with volume LIC. In *Visualization '97., Proceedings*, pages 421–424, 568, 1997.

[29] Xiaoqiang Zheng and Alex Pang. Hyperlic. *IEEE Visualization 2003*, pages 249–256, 2003.

[30] G. Kindlmann. *Visualization and Analysis of Diffusion Tensor Fields*. PhD thesis, School of Computing. University of Utah, 2004.

[31] Gordon Kindlmann, David Weinstein, and David Hart. Strategies for direct volume rendering of diffusion tensor fields. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):124–138, 2000.

[32] M. Zöckler, D. Stalling, and H.-C. Hege. Interactive visualization of 3D-vector fields using illuminated streamlines. In *Visualization '96. Proceedings.*, pages 107–113, 474, 1996.

[33] Andreas Wenger, Daniel F. Keefe, Song Zhang, and David H. Laidlaw. Interactive volume rendering of thin thread structures within multivalued scientific data sets. *IEEE Transactions on Visualization and Computer Graphics*, 10(6):664–672, 2004.

[34] Gordon Kindlmann and David Weinstein. Hue-balls and lit-tensors for direct volume rendering of diffusion tensor fields. *Proceedings Visualization '99*, pages 183–189, 1999.

[35] James F. Blinn. Models of light reflection for computer synthesized pictures. *SIGGRAPH Comput. Graph.*, 11(2):192–198, 1977.

[36] K. P. Fishkin. *A taxonomy for and analysis of tangible interfaces*. Personal Ubiquit Computing, 2004.

[37] H. Ishii and B. Ulmer. Tangible bits: towards seamless interfaces between people, bits, and atoms. In *Proceedings of the CHI'97 conference on human factors in computer systems*, 1997.

[38] Joseph J. LaViola and Jr. Brown. Input and output devices for 3d interaction in virtual reality. In *Input and Output Devices for 3D Interaction in Virtual Reality*, 2000.

[39] I. Scott MacKenzie. Input devices and interaction techniques for advanced computing. *Dept. of Computing and Information Science University of Guelph Guelph, Ontario, Canada*, 2000.

[40] Youngblut, C. R.E. Johnson, S.H. Nash, R.A. Wienclaw, and C.A. Will. Review of virtual environment interface technology. *Review of Virtual Environment Interface Technology. Institute for Defense Analysis*, page 3186, 1996.

[41] fakespace. `http://www.fakespace.com`.

[42] Virtex. `http://www.virtex.com`.

[43] 5dt. `http://www.5dt.com`.

[44] isense. `http://www.isense.com`.

[45] wacom. `http://www.wacom.com`.

[46] pegatech. `http://www.pegatech.com`.

[47] spacemouse. `http://www.spacemouse.com`.

[48] qualixdirect. `http://www.qualixdirect.com/html/3d_mouse_and_head_tracker.html`.

[49] Spacetec. `http://www.spacetec.com`.

[50] Carolina Cruz-Neira, Daniel J. Sandin, Thomas A. DeFanti, Robert V. Kenyon, and John C. Hart. The cave: audio visual experience automatic virtual environment. *Commun. ACM*, 35(6):64–72, 1992.

[51] J. Kjeldskov. Interaction: Full and partial immersive virtual reality displays. In *Proceedings of IRIS24*, pages 587–600, 2001.

[52] Carolina Cruz-Neira. *Applied Virtual Reality*. Siggraph, 1998.

[53] Sensable. `http://www.sensable.com`.

[54] Grigore C. Burdea. *Force and Touch Feedback for Virtual Reality*. Wiley Interscience, 1996.

[55] T. G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvill. A hand gesture interface device. In *Proceedings of the CHI+GI '87 conference on Human Factors in Computer Systems*, 1987.

[56] Jr. Brooks, F. P., Ouh-Young M., J. J. Batter, and P. J. Kilpatrick. Project grope: Haptic displays for scientific visualization. *Computer Graphics*, 24(4):177–185, 1990.

[57] Burkhard Claus Wünsche. *A Toolkit for the Visualization of Tensor Fields in Biomedical Finite Element Models*. PhD thesis, Department of Computer Science. University of Auckland, 2004.

[58] VTK. The Visualization Toolkit. Visited the 13th july 2006.

[59] David T. Gering. A System for Surgical Planning and Guidance using Image Fusion and Interventional MR. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1999.

[60] D. Gering, A. Nabavi, R. Kikinis, W. Eric, L. Grimson, N. Hata, P. Everett, F. Jolesz, and W. Wells III. An Integrated Visualization System for Surgical Planning and Guidance using Image Fusion and Interventional Imaging. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 1999.

[61] 3D Slicer. Medical Visualization and Processing Environment for Research. `www.slicer.org`. Visited the 23rd june 2006.

[62] BioImage Suite. `http://www.bioimagesuite.org/`. Visited the 21st november 2006.

[63] BioTensor Tutorial. Visited the 13th july 2006.

[64] MedINRIA. Asclepios Research Project — INRIA Sophia Antipolis. `http://www-sop.inria.fr/asclepios/software/MedINRIA/`. Visited the 21st november 2006.

[65] Teem summary. `http://www.na-mic.org/Wiki/index.php/TeemSummary`. Visited the 21st november 2006.

[66] Analyze 7.0. New Features/Enhancements. `http://www.analyzedirect.com/Analyze/enhancements.asp`. Visited the 21st november 2006.

[67] Software. `http://arnaud.guidon.free.fr/blog/?page_id=4`. Visited the 21st november 2006.

[68] S. Zhang, Ç. Demiralp, D. F. Keefe, M. DaSilva, David H. Laidlaw, B. D. Greenberg, P. J. Basser, C. Pierpaoli, E. A. Chiocca, and T. S. Deisboeck. An immersive virtual environment for DT-MRI volume visualization applications: a case study. In *VIS '01: Proceedings of the conference on Visualization '01*, pages 437–440, Washington, DC, USA, 2001. IEEE Computer Society.

[69] Computer Science and Artificial Intelligence Laboratory. `http://www.csail.mit.edu/index.php`. Visited the 23rd june 2006.

[70] Brigham and Women's Hospital website. `http://www.brighamandwomens.org/`. Visited the 23rd june 2006.

[71] National alliance for medical image computing. `http://www.na-mic.org`. Visited the 23rd june 2006.

[72] Slicer 3 NAMIC Wiki webpage. `http://www.na-mic.org/Wiki/index.php/Slicer3`. Visited the 23rd june 2006.

[73] A. Burrell and A.C. Sodan. Web interface navigation design: Which style of navigation-link menus do users prefer? In *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*, pages 1–10, 2006.