

A VISUALIZATION ENVIRONMENT FOR NEMATIC LIQUID CRYSTAL MATERIALS

Paul A. Farrell Arden Ruttan

Department of Computer Science, Kent State University, Kent, OH 44242.

farrell@cs.kent.edu ruttan@cs.kent.edu

Hong Ong

Oak Ridge National Laboratory, Oak Ridge, TN

hong@cs.kent.edu

Yang-Ming Zhu

PET Engineering, Philips Medical Systems, 595 Miner Road, Cleveland, Ohio 44143

zhu@computer.org

ABSTRACT

We describe a visualization environment for examining the behavior of nematic liquid crystal materials. This permits examination of the local behavior of the solution, which is a tensor, and the identification of properties of physical importance, such as orientation, order, and axiality.

KEYWORDS

Visualization environment, liquid crystal physics.

1. INTRODUCTION

Liquid crystals are so called because they exhibit some of the properties of both the liquid and crystalline states. In fact they are substances which, over certain ranges of temperatures, can exist in one or more *mesophases* somewhere between the rigid lattices of crystalline solids, which exhibit both orientational and positional order, and the isotropic liquid phase, which exhibits neither. Liquid crystals resemble liquids in that their molecules are free to flow and thus can assume the shape of a containment vessel. On the other hand they exhibit orientational and possibly some positional order. This is due to the intermolecular forces which are stronger than those in liquids and which cause the molecules to have, on average, a preferred direction. Liquid crystals may exist in a number of mesophases, such as the *nematic*, *smectic*, *cholesteric* phases (see [4]). Here we confine ourselves to nematic liquid crystals, which exhibit orientational but no positional order.

In many liquid crystal devices, the liquid crystal molecules at the edge of the device are mechanically given a specific orientation which induces the molecules in the interior of the device to mimic that orientation to the extent possible. For a given device geometry and a given molecular edge orientation, it may not be possible for the liquid crystal molecules to duplicate that edge orientation throughout the interior of the device. For instance consider a thin slab containing liquid crystal material in which the molecules at the edge of the device are oriented radially. This is illustrated in Figure 1. Since the radial orientation cannot be maintained throughout the interior, pockets of liquid crystal with no fixed orientation form. These pockets, called defects, are illustrated by the empty spaces in Figure 1, and, depending on the temperature and magnetic or electric field strength, will form in different places. It is of significant importance in liquid crystal research to develop computer models which can accurately determine the orientation and the defects of liquid crystals in a confined medium, especially in the presence of changing temperature and field intensities.

In this paper, we describe a visualization environment, which enables us to examine the results of a computational cluster based simulation in order to view the molecular orientations and the existence of defects. The computational techniques and the visualization environments are implemented for three dimensional problems. However, in this paper, we will limit the illustrations to the case of two dimensional problems, in order to facilitate reproduction in grey scale. The paper is organized as follows: In section 2., we give a brief description of the mathematics of the liquid crystal model.

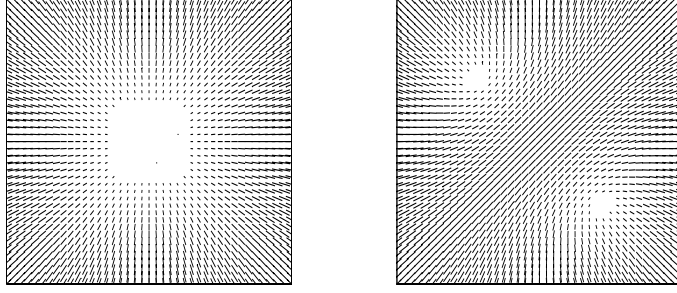


Figure 1: Liquid Crystal Orientation for Scaled Temperature $T = 0.3013$ and $T = 0.3174$

2. THE LIQUID CRYSTAL MODEL

We begin with a description of the liquid crystal problem. The problem under consideration is to determine the minimum energy equilibrium configuration of liquid crystals in a slab

$$\Omega = \{(x_1, x_2, x_3) : 0 \leq x_1 \leq a, 0 \leq x_2 \leq b, 0 \leq x_3 \leq c\} \quad (1)$$

with surface $\partial\Omega$. Using the Landau-de Gennes formulation, the free energy can be expressed in terms of a tensor order parameter field Q ; see Priestly et al. [6]. The free energy is given by

$$F(Q, T) = F_{\text{vol}}(Q, T) + F_{\text{surf}}(Q) = \int_{\Omega} f_{\text{vol}}(Q, T) dV + \int_{\partial\Omega} f_{\text{surf}}(Q) dS, \quad (2)$$

where $Q = Q(p)$, $p \in \Omega$, is a 3×3 symmetric traceless tensor, which is represented by

$$Q(p) = \begin{pmatrix} q_1 & q_2 & q_3 \\ q_2 & q_4 & q_5 \\ q_3 & q_4 & 1 - q_1 - q_4 \end{pmatrix} \quad (3)$$

and the q_i are real-valued functions on Ω . The q_i are to be determined so that the free energy (2) is minimal. The representation

$$\begin{aligned} f_{\text{vol}}(Q, T) = & \frac{1}{2} \mathcal{L}_1 Q_{\alpha\beta,\gamma} Q_{\alpha\beta,\gamma} + \frac{1}{2} \mathcal{L}_2 Q_{\alpha\beta,\beta} Q_{\alpha\gamma,\gamma} + \frac{1}{2} \mathcal{L}_3 Q_{\alpha\beta,\gamma} Q_{\alpha\gamma,\beta} \\ & + \frac{1}{2} \mathcal{A} \text{trace}(Q^2) - \frac{1}{3} \mathcal{B} \text{trace}(Q^3) + \frac{1}{4} \mathcal{C} \text{trace}(Q^2)^2 \end{aligned} \quad (4)$$

uses the convention that summation over repeated indices is implied and indices separated by commas represent partial derivatives. For example,

$$Q_{\alpha\beta,\gamma} Q_{\alpha\gamma,\beta} = \sum_{\alpha=1}^3 \sum_{\beta=1}^3 \sum_{\gamma=1}^3 \frac{\partial Q[\alpha, \beta]}{\partial x_\gamma} \cdot \frac{\partial Q[\alpha, \gamma]}{\partial x_\beta}.$$

The parameters $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{L}_1, \mathcal{L}_2$ and \mathcal{L}_3 depend on the liquid crystal properties. The parameter \mathcal{A} also depends on the temperature of the liquid crystal material. In this paper we assume strong anchoring, i.e.

$$F_{\text{surf}}(Q) = \int_{\partial\Omega} f_{\text{surf}}(Q) dS = 0. \quad (5)$$

Details can be found in [5, 6, 8].

The minimum energy equilibrium configuration of the liquid crystals is determined by solving the Euler-Lagrange equations associated with (2). These equations yield a boundary value problem for a system of nonlinear partial differential equations for the q_i .

For the current paper, we will assume that the liquid crystal material is contained in a small cube. By changing the length scale, the domain of the problem can be changed from a small cube to a standard $1 \times 1 \times 1$ cube. Discretizing the Euler-Lagrange equations on the unit cube using finite differences produces a large three dimensional grid of points, typically with at least 100 points along each edge, that is about 100^3 points.

For each point $p_{i,j,k} = (x_1^i, x_2^j, x_3^k)$ in this grid, there are 5 variables representing the values of

$$q_1(p_{i,j,k}), \dots, q_5(p_{i,j,k})$$

in the expansion of the tensor at that point, and 5 discretized Euler Lagrange equations. This produces a system on the order of 5×10^6 nonlinear equations in 5×10^6 unknowns. We represent this by

$$G(Q, T) = 0. \quad (6)$$

We solve this system by Newton's method. Each iteration of Newton's method requires the solution of a linear system of equations with the matrix of partial derivatives $G_Q(Q, T)$ obtained from the discretized Euler-Lagrange equations. Since the matrix $G_Q(Q, T)$ is extremely large, on the order of $10^6 \times 10^6$ and sparse, each row containing no more than 5×27 entries, iterative methods must be used to solve the linear system arising from Newton's method. Since $G_Q(Q, T)$ can be indefinite or singular for some values of Q and T we use the Krylov subspace method MINRES [3] to solve the those linear systems.

Due to the complexity of the liquid crystal equations, the C code for Newton's method is generated using a Maple program. It calculates the Euler-Lagrange equations symbolically and discretizes them using centered difference approximations for the derivatives. The resulting equations for a given point depend on the data at that point and the data at the closest 26 points, that is the points in the $3 \times 3 \times 3$ sub-cube centered at that point. To conserve memory we use a matrix-free formulation, that is the entries of $G_Q(Q, T)$ are not stored, but are calculated as needed.

Our visualization system can work with local data files which could be generated by one of these solvers or work with one of these solvers directly. If it works with a solver, then it serves as a front end for the computation using a canonical interoperative client-server communication. In fact, the full computation is a path following problem since there may exist multiple solutions at each temperature given a set of parameters. Our visualization system can potentially steer this path following process as well.

3. VISUALIZING LIQUID CRYSTALS

Bailey developed an initial version of such a system in [2] for liquid crystal order parameter tensor field visualization, which renders the tensor field using the proprietary Hewlett-Packard Starbase graphics library. The system described here is implemented in OpenGL in order to enhance portability, and incorporates extended visualization options and functionality, which include two more rendering modes, interactive zooming and picking to identify data values.

3.1 Director, order parameters, and defects

The orientation of liquid crystals can be abstracted by a director field, $\mathbf{n} = (n_x, n_y, n_z)$, which is a unit vector defined in space. The direction of the molecule's long axis changes constantly due to thermal fluctuation and

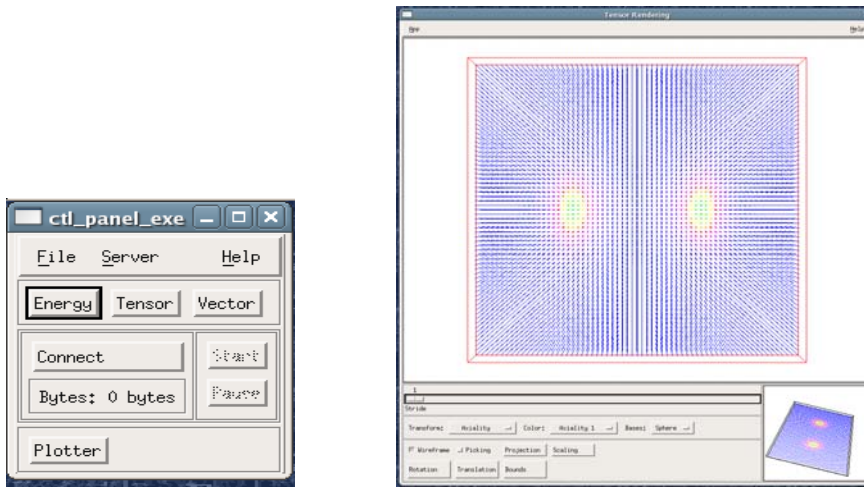


Figure 2: Visualization Environment

the thermal average of all possible directions gives the director. Since experimentally the nematic phase is not ferroelectric, physically \mathbf{n} and $-\mathbf{n}$ are equivalent. One way to characterize the degree of alignment of the axis of rodlike molecules, is to use the order parameter defined as $S = \frac{1}{2} \langle 3\cos^2\theta - 1 \rangle$, where $\langle \rangle$ represents the thermal average, and θ is the angle between the molecule and the director \mathbf{n} . The larger S , the more ordered the molecules.

The above definition of order parameter can be naturally generalized to the ordering matrix (tensor) $Q = \frac{1}{2} \langle 3\mathbf{n}\mathbf{n}^T - I \rangle$, where \mathbf{n}^T is the transpose of the director \mathbf{n} and I is the identity matrix. The tensor Q is traceless and symmetric. Since it is symmetric, all its eigenvalues are real. If all its eigenvalues are equal then they must be zero due to the traceless property. In this case it is an isotropic liquid; if two of them are equal, then it is a uniaxial nematic; if none of them are equal, it is biaxial. Distinguishing these cases is considered important for a visualization environment.

In a free space, that is one without boundaries and external fields, the director field is uniform. However, the director can be easily altered by surface alignment and/or external field. Thus, it is possible that the director field undergoes some distortions: splay, twist, and bend. If the director field is not a smooth function of r at all points under some circumstances, the singular points and lines are called defects.

One can compute the energy of a defect according to the elastic theory sketched in the next section. Since in a defect core the director is not a smooth function, one has to take some approximation. One argument is that the energy is so high that the defect core is in an isotropic state. On the contrary, Schopohl and Sluckin [7] found that the defect core is not isotropic, instead, it is biaxial.

The objective of the environment is to allow us to examine the orientation of the molecules in the most intuitive way possible, together with some related information such as energy. To enable this, three panels can display a tensor representation of the molecules, a representation of the eigenvectors and eigenvalues of the tensor, and a representation of the energy. We describe each of these in the next sections.

3.2 Tensor rendering mode

The purpose of the tensor rendering mode is to use representing geometries to visualize the tensor field. These are oriented in space according to the orientation of the eigenvectors of the Q tensor, and modified by transformations related to the eigenvalues. Color coding is used to convey further information related to the eigenvalues. The choice of bases, shape transforms, and color coding can be changed interactively using pulldown menus.

Three different geometrical shapes, sphere, circle, and axes are used as bases to represent the liquid crystal molecules. These do not convey any information about the local material properties by themselves. To do this

we apply a local shape transform and also color code the basis element.

3.2.1 Shape Transforms

The shape transforms available consist of three transforms, called **Axiality**, **Average Motion**, and **Energy** in addition to an identity transform. In all cases the basis element is first rotated to align its x,y,z axes with the eigenvectors at the point, and translated to the correct position in space. In all cases, if the scaling factor in any direction is too small, it is clamped at a prespecified small positive value. This not only not permits a visible representation in three dimensions but also eliminates a singular value problem, which might also arise later.

The **Axiality** transformation scheme is as follows. Let the three eigenvalues be λ_1, λ_2 , and λ_3 . The corresponding eigenvectors u, v , and w are $(u_1, u_2, u_3), (v_1, v_2, v_3)$, and (w_1, w_2, w_3) , respectively. Then $\lambda'_i = \lambda_i - \min(\lambda_1, \lambda_2, \lambda_3), i = 1, 2, 3$ are the scaling factors in the three eigenvector directions u, v , and w . In a uniaxial region, the two smallest eigenvalues are equal and thus are clamped giving rise to a long cigar shaped ellipsoid. To increase the dynamic range, these transformed values can be normalized with respect to, for example, the largest possible difference of eigenvalues over the whole tensor field.

In the **Average Motion** transformation the scaling factor in each direction reflects the average tendency for the molecule to orient in that direction in the sense of statistical physics. Loosely, this is analogous to the concept of an electron cloud. In this scheme the scaling factors are computed as

$$\lambda'_i = \frac{2(\lambda_i + \frac{1}{2}S_{max})}{3S_{max}},$$

where S_{max} is the maximum eigenvalue across the whole tensor field. If the boundary conditions specify a uniaxial field at the boundary, the eigenvalues achieve the global maximum value, that is, at the boundary the eigenvalues are $S, -\frac{1}{2}S, -\frac{1}{2}S$, where $S = S_{max}$. In that case, the scaling factors become 1, 0, 0. A less convenient property of this scheme is that, when the anisotropy is weak, that is, the maximum eigenvalue is much less than S_{max} , all three scaling factors are close to $\frac{1}{3}$. As a consequence it is hard to tell whether the region is uniaxial, biaxial, or negative uniaxial.

The **Eigenvalue** transform uses the absolute values of the eigenvalues as scaling factors directly. A disadvantage is that this scheme cannot distinguish between positive and negative eigenvalues.

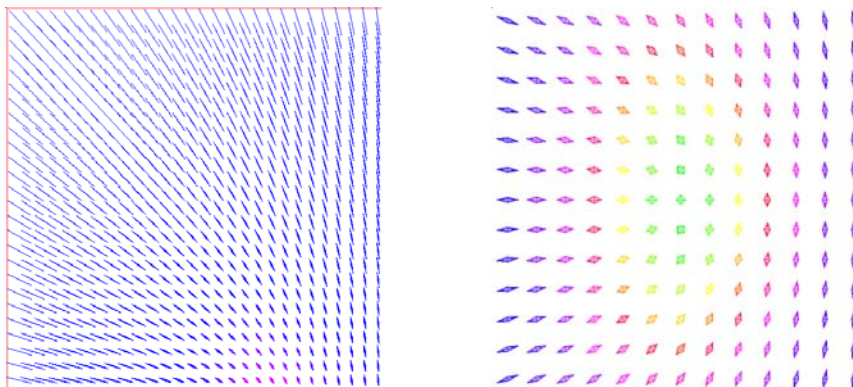


Figure 3: Uniaxial Tensor

Biaxial and Negative Uniaxial Tensor

3.2.2 Color Coding

As indicated above each of the transformation schemes fail to convey all the properties desired. To remedy this, the environment also supports four coloring schemes for the transformed basis elements, corresponding to **Axiality**, **Order**, **Ratio**, and **Energy Density**. Let l_{max} and l_{min} be the eigenvalues with maximum and

minimum absolute values respectively. While the above symbols are defined locally, we also define a global value S_{max} which is the largest eigenvalue across the full tensor field. As we noted earlier, the largest eigenvalue may be achieved on the boundaries where no defects are involved or if there is a defect at a point far away from the boundary.

There are actually four variations on the **Axiality coding** available. In all cases, a function

$$f(p) = \frac{2 \sum_{i=1}^3 \lambda_i^2}{l_{max}^2} - 3$$

where p is a data point at (x, y, z) , is used to define the color. The only difference among the schemes is the method of generating the color mapping.

In the first, if $|l_{max}| \leq S_{max}/12$, then the color is set to a constant color such as grey, otherwise the above function is used. To distinguish between positive l_{max} and negative l_{max} , two different color lists can be used. Thus $\lfloor 12f(p) \rfloor$ can be used as an index to the color list, where $\lfloor \cdot \rfloor$ is the floor function. The color lists were chosen so that for positive l_{max} , the colors range from blue to red, and for negative l_{max} , the colors range from red to green. In the second variation, if $l_{max} = 0$ (isotropic), a default color (dark grey for example) is assigned to the data object.

In the third variation there is no predefined color list. The RGB values are generated instead. If $|l_{max}| \leq S_{max}/12$, the RGB tuple is $(1/3, 1/3, 1/3)$ which yields a shade of grey, otherwise, the $f(p)$ is clamped between 0 and 1. If $l_{max} < 0$, the tuple is $(1 - f(p), 0, f(p))$ which gives colors ranging from blue to red, and if $l_{max} > 0$, the tuple is $(0, 1 - f(p), f(p))$ which gives colors ranging from blue to green.

In the fourth variation, the $f(p)$ is clamped between 0 and 1. If $l_{max} < 0$, the RGB tuple is $((1 - f(p))|l_{max}|/S_{max}, 0, f(p)|l_{max}|/S_{max})$, and if $l_{max} \geq 0$, the tuple is $(0, (1 - f(p))|l_{max}|/S_{max}, f(p)|l_{max}|/S_{max})$. The range of colors given by this scheme depends on the value of $|l_{max}|$. If $|l_{max}|$ is comparable to S_{max} , the colors range from blue to red for negative l_{max} and from blue to green for positive l_{max} . If $|l_{max}|$ is much less than S_{max} , the colors are close to dark grey.

The colors given by the first three variations do not depend on the ratio $|l_{max}|/S_{max}$ and these schemes are regarded as local schemes. The colors generated by the fourth variation does depend on $|l_{max}|/S_{max}$ and it is thus a global scheme.

In the **Order coding** there are two variations, **Order** and **Log Order**. In the first, the function $f(p)$ is defined as

$$f(p) = \frac{\sqrt{\sum_{i=1}^3 \lambda_i^2}}{\sqrt{\frac{3}{2} S_{max}}}$$

If $f(p) \leq 1$, $\lfloor 12f(p) \rfloor$ is used to index a color list which changes from blue to red; otherwise, $\lfloor 12/f(p) \rfloor$ is used to index another possible color list which changes from blue to green, for example.

In the Log Order scheme, the function $f(p)$ is defined as

$$f(p) = -\log_2 \frac{\sqrt{\sum_{i=1}^3 \lambda_i^2}}{\sqrt{\frac{3}{2} S_{max}}}$$

Two color lists are employed. $f(p)$ is clamped between -12 and 12. The negative value and nonnegative of $\lfloor f(p) \rfloor$ are used to index two color list respectively. Obviously, 25 different colors are used here. The two color lists range from blue to red and blue to green respectively.

The two remaining color codings are a **Ratio coding** and an **Energy Density coding**. The **Ratio coding** calculates the ratio between l_{min} and l_{max} . If $l_{max} = 0$, a default color (dark grey) is assigned, otherwise $\lfloor 24|l_{min}/l_{max}| \rfloor$ is used to index a color list ranging from blue to red. In the **Energy Density coding** the energy density at each point is calculated. Assume the energy density at the data point is e and the global maximum and minimum energy densities are e_{min} and e_{max} respectively. The value of $\lfloor 12(e - e_{min})/(e_{max} - e_{min}) \rfloor$ can be used to index a color list (blue to red).

3.3 Vector rendering mode

In the vector mode, three eigenvectors are rendered to different viewports according to some criterion. The eigenvalues are color coded. One scheme (called by clicking the Sorted button) is to sort the vectors based on their corresponding eigenvalues (sorted vectors), which proved to be problematic. We thus developed an alternative: grouped vector (called by clicking the Grouped button), where eigenvectors are separated using an algorithm, which is described in detail in [10].

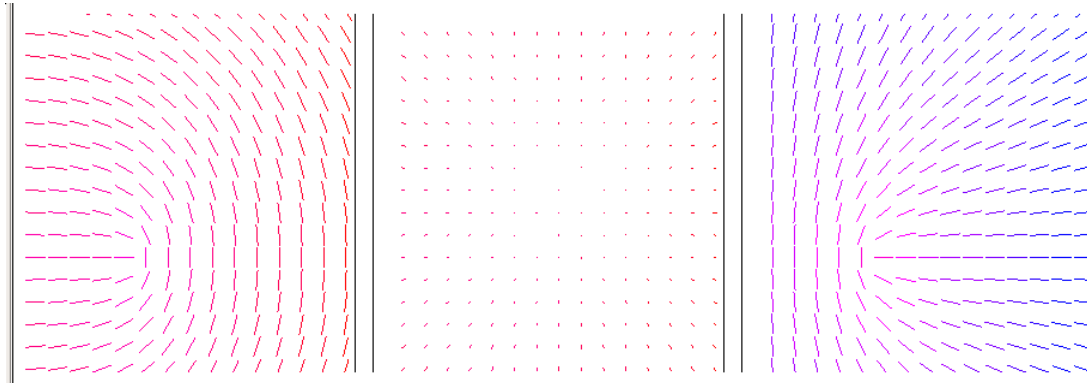


Figure 4: Zoomed Vector Representation

4. ADDITIONAL CONTROLS

The visualization system initializes with a control widget, which permits selecting the input from a data file or from a connection to a remote computational engine. A standard file browser allows selection of a data file. For the computational engine case a server address can be set. In this case, the control widget will open a socket by using the given address, connect to that server, and get a set of default parameters from that server. For the latter case, the program does not start the calculation immediately since the user may wish to modify the parameters.

In addition to the application specific visualization widgets described above, popup menus provide for more standard functionality such as rotation, translation, selecting a stride into the data set to reduce the density, changing the representation from wire-frame to filled polygon, and changing the projection from perspective to parallel. The parallel option was included, since in the perspective projection, it is hard to distinguish an escaped liquid crystal director field, that is one where the principal direction is out of the plane, from one where the directors are parallel to the direction from the eye to the center of viewport. In the implementation, if any scaling factor is zero, we set its value to a very small value, instead of setting it to zero exactly, otherwise the scaling matrix becomes singular and does not have an inverse matrix. It was also noted that although the filled polygon representation is more visually attractive, the wireframe mode proved more popular in practice.

To allow examination of only part of the data set, a bounds popup button allows the user to change the bounds in the three coordinate directions. Each coordinate has two sliders to control the lower and upper bounds. Error checking is included to prevent the user from setting incorrect bounds.

The user may also select from two functionalities for the mouse buttons. In one the left and right buttons zoom in and out at the point pointed to by the mouse. The other uses the OpenGL select functionality to permit picking of individual data elements. When an element is picked, the corresponding eigenvalues and eigenvectors are displayed.

A subsidiary pane, which always includes the full area is included. When bounds are set the subsection selected is indicated on it, and when zooming is used the point at the center of the zoom area is indicated.

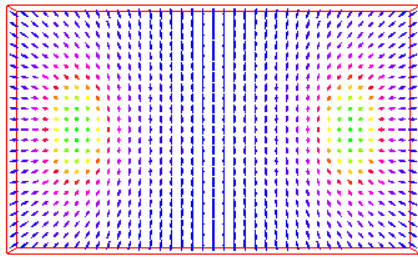
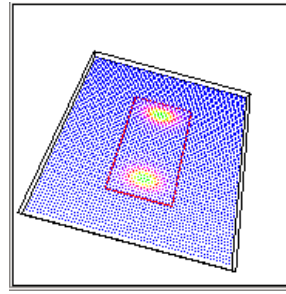


Figure 5: Effect of Setting Bounds



Display of Region

5. CONCLUSION

We have described a visualization environment for nematic liquid crystals, which enables us to detect not only the local average orientation of the molecules, but also the strength of the alignments with that direction. We can also easily see, where the nature of the solution varies from uniaxial, to biaxial, to negative uniaxial, and the local average motion and energy density. The environment can also be connected to a computational server actively running a solution of the Landau-de-Gennes equations to monitor the evolution of the solution with temperature.

6. ACKNOWLEDGEMENT

The authors were supported in part by NSF grants CDA-9617541, ASC-9720221, ITR-0081324, and DMR-8920147 ALCOM and by the OBR Investment Fund OCARnet.

REFERENCES

- [1] J. Baglama, et al., 1998, Computation of a few small eigenvalues of a large matrix with application to liquid crystal modeling, *J. Comp. Phys.*, 146, pp. 203–226.
- [2] J.R. Bailey, 1996, A Distributed Scientific Visualization Environment, Master Thesis of Computer Science, Kent State University.
- [3] D. Calvetti, L. Reichel, and Q. Zhang, 1994, Conjugate gradient algorithms for inconsistent linear system, *Proceedings of the Cornelius Lanczos International Centenary Conference*, eds. J. D. Brown, et al. , SIAM, Philadelphia, pp.267-272.
- [4] P. J. Collings, 1990, *Liquid Crystals, Nature's Delicate Phase of Matter*, Princeton Science Library, Princeton University Press, New Jersey.
- [5] P. A. Farrell, A. Ruttan and R. R. Zeller, 1992, Finite difference minimization of the Landau-de Gennes free energy for liquid crystals in rectangular regions, *Comput. Appl. Math.*, 1992, pp. 137–146.
- [6] E. B. Priestly, P. J. Wojtowicz and P. Sheng, eds., 1975, *Introduction to Liquid Crystals*, Plenum Press, New York, 1975.
- [7] N. Schopohl and T.J. Sluckin, 1987, Defect core structure in nematic liquid crystals, *Phys. Rev. Lett.* 59, 2582.
- [8] E. G. Virga, 1994, *Variational Theories for Liquid Crystals*, Chapman and Hall, London.
- [9] M. Woo, J. Neider, and T. Davis, 1997, *OpenGL Programming Guide*, Addison-Wesley Developers Press, 2nd ed.
- [10] Yang-Ming Zhu, P.A. Farrell, 2002 Vector Grouping Algorithm for Liquid Crystal Tensor Field Visualization, *Liquid Crystals*, 29(10), pp. 1259 - 1264.