Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustr. 7, D-14195 Berlin-Dahlem

Detlev Stalling   Thomas Steinke

Visualization of Vector Fields in Quantum Chemistry

Preprint SC-96-1 (December 1996)

Visualization of Vector Fields in Quantum Chemistry

Detlev Stalling Thomas Steinke

Abstract

Many interesting phenomena in molecular systems like interactions between macromolecules, protein-substrate docking, or channeling processes in membranes are gouverned to a high degree by classical Coulomb or van-der-Waals forces. The visualization of these force fields is important for verifying numerical simulations. Moreover, by inspecting the forces visually we can gain deeper insight into the molecular processes. Up to now the visualization of vector fields is quite unusual in computational chemistry. In fact many commercial software packages do not support this topic at all. The reason is not that vector fields are considered unimportant, but mainly because of the lack of adequate visualization methods. In this paper we survey a number of methods for vector field visualization, ranging from well-known concepts like arrow or streamline plots to more advanced techniques like line integral convolution, and show how these can be applied to computational chemistry. A combination of the most meaningful methods in an interactive 3D visualization environment can provide a powerful tool box for analysing simulations in molecular dynamics.

Preprint SC-96-1 (December 1996)

Contents

1 Introduction

In general, molecular interactions are governed by the interaction between various charge distributions in space. Today, the most sophisticated theory for explaining these interactions is non-relativistic quantum theory. However, for large macromolecules methods based on classical pair potentials can be used, too. In particular the mutual interactions in molecular systems (especially in biological macromolecules) are dominated by electrostatic effects. The importance of the molecular electrostatic potential (MEP) is widly recognized, and it is therefore not surprising that the MEP can be computed easily using today's standard molecular modeling programs. To interprete the MEP in molecular context 3D visualization methods are available in most commercial and academic software tools.

Whereas the energetics are determined by the (electrostatic) potentials, favoured mutual orientations and the direction of attachment processes can be better understood by looking directly at the local molecular electrostatic field (MEF), which in contrast to the potential is a vector quantity. One example is the interaction between plastocyanin and cytochrome c as described in [1]. It was shown that in this case electrostatic steering, i.e. the forcing of specific orientations prior short range interactions, plays an essential role. Further phenomena such as preferred hydrations sites in DNA, hydrophobicity, or properties of hydrogen bonds can be explained by analysing the electrostatic field pattern (see [2] and references therein). There is also some evidence that the inclusion of MEF into QSAR and similarity analysis improves correlation with experimental results [2].

In contrast to MEP, the visualization of the MEF is not widly used in current studies of biomolecular interactions. Whereas the scalar property MEP can be easly displayed by color coding or other mapping techniques, it is much more difficult to visualize both magnitude and direction of the electric field vectors simultaneously in a human acceptable form. In the past common approaches were either to color-code vector magnitude and to map it onto the molecular surface, or to display simple 2D slices of the molecular situation combined with projected field vectors, i.e. arrow plots. On the other hand, a number of novel visualization methods for depicting vector fields have been developed in the computer graphics and visualization community in recent years. These methods comprise of icon based techniques, generation of directional textures, or utilization of special hardware features of modern graphics computers. Mainly developed in the field of computational fluid dynamics, these methods have not yet been applied thoroughly to vector fields arising in chemical simulations.

This paper is intended to show the application of modern methods for vector field visualization in the area of molecular modeling and quantum chemistry. In particular the MEF of different molecules is shown and some characteristics are discussed. In section 2 we shortly outline how the data for the examples shown in this paper has been computed, namely by use of a semi-empirical NDDO method. In section 3 we give a survey of different methods for vector field visualization. These methods are applied to different molecular fields in section 4. Results and concluding remarks are given in section 5.

2 Electrostatic Fields from Wavefunctions

MEF data can be computed by any method dealing either with static and/or parametrized atomic charges (classical force field methods) or electronic charge distributions (quantum chemical methods). We have used the semi–empirical NDDO approach [3] enhanced with the NAO/PC [5] technique to reduce the electronic charge distribution to a discrete set of point charges. The basic theory is shown below. We have implemented the computation of the MEP and MEF on a regular 3D grid into our local VAMP code [4].

NDDO based methods solve the stationary non–relativistic Schrödinger equation within the Born–Oppenheimer approximation

$$\hat{H}(\{\vec{r}\}; \{\vec{R}\})\ \Psi_k(\{\vec{r}\}) = E_k\ \Psi_k, \tag{1}$$

where $\hat{H}$ is the spin free Hamilton operator which depends on the coordinates of all electrons $\{\vec{r}\}$ and parametrically on the positions of the nucleii $\{\vec{R}\}$, Ψ_k is the wavefunction of a given state k (usually the ground state of the system, $k = 0$), and E_k the electronic energy (the expectation value of $\hat{H}$ in state k). Within the Hartree-Fock model the wavefunction Ψ is represented by a Slater determinant consisting of one-particle functions, i.e. canonical molecular orbitals (MO). The MOs $\{\phi\}$ are built by a linear combination of atomic orbitals $\{\chi\}$ (LCAO formalism)

$$\phi_i = \sum_{\mu} c_{\mu i}\ \chi_{\mu}. \tag{2}$$

The coefficients $\{c\}$ are determined by means of a variational principle which leads to the Roothaan–Hall type equation

$$\boldsymbol{F}\,\boldsymbol{C} = \boldsymbol{E}\,\boldsymbol{S}\,\boldsymbol{C}, \tag{3}$$

where $\boldsymbol{F}$ is the Fock matrix, $\boldsymbol{C}$ is the matrix of eigenvectors (the coefficients in the atomic orbital expansion), $\boldsymbol{E}$ is a diagonal matrix of the eigenvalues

ϵ_i, and $\boldsymbol{S}$ is the overlap matrix resulting from the non-orthogonality of the atomic orbitals.

Within semi-empirical methods a minimal set of Slater orbitals are used as atomic orbitals throughout. Only the valence electrons are taken into account. The computational cost is reduced by applying approximate schemes during the computation of integrals over the atomic orbitals or by using fitted parameters based on experimental results for integrals itself. Additionally, within the Roothaan-Hall equation the overlap matrix is replaced by the identity matrix $\mathbf{1}$, i.e. the general eigen problem is reduced to the special eigen problem

$$\boldsymbol{F}\boldsymbol{C} = \boldsymbol{E}\boldsymbol{C}. \tag{4}$$

After finishing the SCF iterations a set of point charges representing the electronic charge distribution is build based on *pre* natural atomic orbitals (NAO) [5]. The *pre* NAOs for a non-hydrogen atom are computed by diagonalizing each one-center block of the density matrix $\boldsymbol{P}$,

$$P_{\mu\nu} = 2 \sum_{i}^{occ} c_{\mu i} c_{\nu i} \quad \text{(closed shell case)} \tag{5}$$

and a subsequent estimation of magnitudes and positions in space of charge centers for each lobe of a sp^x NAO. Therefore, for each heavy atom 8 point charges and for each hydrogen 1 point charge is obtained. Using these point charges and the remaining core charges of the atomic nucleii the MEP and MEF can be easily calculated applying Coulomb's law. Both the MEP and the MEF are calculated on a 3D regular grid having a reasonable resolution in space. The set of point charges can be clearly identified in some of the images presented in section 4.

3 Visualization Methods

In this section we will review some visualization techniques, which can be used to gain insight into the structure of the electrostatic field within a molecular arrangement. The discussion mainly focuses on technical issues. In section 4 the different methods are applied to real examples and pros and cons are discussed in comparison.

3.1 Iso-Potential Surfaces

The molecular electrostatic field E is related to the molecular electrostatic potential ϕ by $E = -\nabla\Phi$. This means that field vectors and surfaces of equal potential value are always oriented perpendicular to each other. Therefore the structure of an electrostatic field can be reconstructed from pictures of iso-potential surfaces. Although this reconstruction process sometimes is difficult to perform mentally, in many molecular modeling software packages this is the only way to get at least an indirect understanding of the MEF structure. Nevertheless, visualization of iso-potential surfaces has some advantages. It allows one to quickly get an overview about the molecular situation and it clearly reveals possible symmetries.

In addition to iso-potential surfaces there are a number of other methods which are commonly used for visualizing the electrostatic potential. In particular these are color coding applied to cutting planes or molecular surfaces, as well as different kinds of direct volume rendering. With these methods it is much harder to imagine how the gradient field, i.e. the MEF, looks like. Therefore we will not discuss them further here.

Isosurface algorithms can be divided into two general classes. The one are methods which generate an intermediate polygonal representation of the surface, the other are methods which directly render images for a given point of view. Modern graphics computers are optimized for rendering large numbers of triangles. Therefore polygonal methods have become the favoured choice in many applications. Once a polygonal model has been created the surface can be zoomed and rotated interactively. However, depending on the rendering performance of the graphics hardware this approach is limited to model sizes somewhere between 50 000 and 500 000 triangles. If the model size exceeds these limits it is often preferable to generate an isosurface image directly without having to store a huge intermediate representation.

Let us first take a look at polygonal methods for generating isosurfaces. Almost any method used today is derived from the famous marching cubes algorithm proposed by Lorensen and Cline in 1987 [10]. This algorithm works for a scalar function, e.g. the molecular electrostatic potential, which is defined at the vertices of a regular cartesian grid. Other grid types can be handled as well using simple modifications. Assuming trilinear interpolation within a cubic grid cell, it is easy to check whether a particular cell is intersected by an isosurface or not. Intersected cells have corners with data values both below and above the isosurface threshold. For these cells a local triangular approximation of the isosurface is constructed. If the triangulations of neighbouring cells exactly fit together all cells can be processed independently and the final polygonal model is simply the sum of the local approximations. For performance reasons the cell triangulations are read from a lookup table. Since each of the 8 corners of a cell may lie above or below the isosurface threshold, there are 256 different cases in total. These can be divided into 14 groups of different topology.

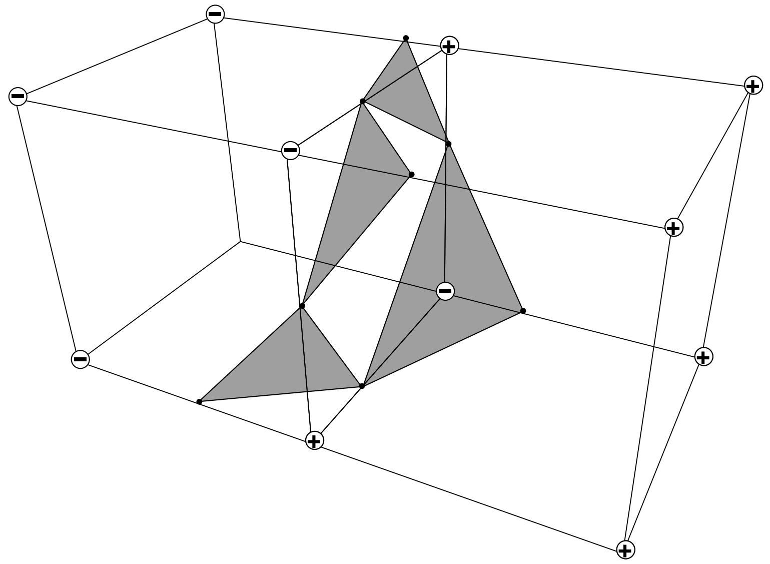

Figure 1: Example of an inconsistent triangulation in the original marching cubes algorithm.

For most configurations it is quite clear how to construct a valid triangulation. However, ambiguities arise for example if opposite vertices of a rectangular face have equal sign. In this case the correct topology cannot be determined from a binary classification alone. The original marching cubes algorithm did not take into account these ambiguities. As a result, sometimes incorrect surfaces were produced, i.e. surfaces with holes in it. An example of such a configuration is shown in Figure 1. These holes are caused by triangulating neighbouring cells in an inconsistent way. Such inconsistencies can be avoided by using a modified lookup table [13]. However, in this case not the same isosurface would be obtained anymore after changing the sign of all data values. An alternative method which preserves this kind of symmetry is to generate special patches to fill the holes [15]. While this approach produces a consistent triangulation, it may not represent the shape of the exact isosurface correctly. Assuming trilinear interpolation, an additional scalar value at the center of a cell has to be evaluated to determine the correct shape [14]. However, in many applications still the more simple original algorithm is used, often equipped with a modified lookup table. The reason might be that ambigious configurations do not occur very often if the data has been sampled at a sufficiently high resolution.

As mentioned above, a more serious problem with polygonal isosurface algorithms is the huge number of triangles they produce. To reduce this number standard methods for polygon reduction may be applied in a postprocessing step. The obvious disadvantages are that still a huge intermediate representation has to be computed and that information inherent to the isosurface problem cannot enter anymore. Therefore ploygon reduction should be integrated into the isosurface algorithm. An example of such a method is the so-called compact cubes algorithm proposed by Warren and Moore [12]. Starting with a marching cubes like lookup table, it works by coalescing tiny triangles into points, and long thin ones into edges. Using this simple technique the number of polygons can be reduced to up to 50% compared to traditional marching cubes. We used this method to generate the examples in section 4.

For very high resolution data sets it becomes favourable to render images of an isosurface directly without generating an intermediate polygonal representation. One particular technique is so-called dividing cubes, also proposed by Lorensen [11]. Again, all cells of a regular cartesian grid are checked for intersection with the isosurface. Intersected cells then are subdivided into smaller cubes. The smaller cubes are checked for intersection again and are subdivided recursively until the projection of a subcube into screen space is smaller than a pixel. In this case a single pixel is drawn with ordinary shading and z-buffering applied. After all cells have been processed an image of the isosurface from a given point of view is obtained. For other viewing directions the algorithm has to be rerun.

Direct iso-surfacing can also be implemented via ray casting. Instead of iterating in object space, in ray casting the main loop is in image space. For each image pixel a ray is sent into the volume and the first intersection with the isosurface is computed. The gradient vector at this location determines how the pixel is shaded. In direct methods usually the shading calculations are performed in software. Therefore one has greater flexibility in choosing a proper illumination model. However, it is more difficult to combine direct methods with the display of other objects, e.g. text, line primitives, or polygons.

3.2 Arrows and Icons

Probably the most simple method for depicting a vector field is to draw an arrow plot. An arrow can be used to indicate field direction at a unique point. Proper scaling and coloring allows one to encode a scalar quantity, e.g. vector magnitude, as well. Arrow plots are easy to implement and easy to interprete. These are their main advantages. On the other hand, they are restricted to a rather coarse spatial resolution. If individual symbols overlap each other, the image rapidly becomes cluttered. Therefore in general arrows are not well suited to resolve small details of a field.

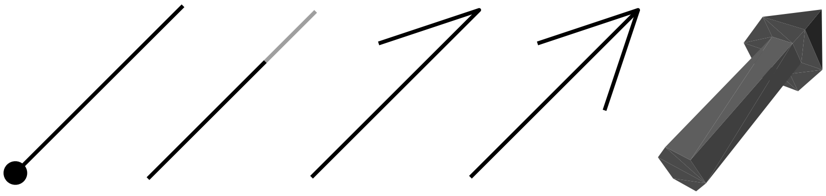

Figure 2: Different graphical representations of an arrow.

While it is quite obvious how to draw arrows in a plane, some care has to be taken in 3D graphics. Definitely the nicest results are obtained by representing arrows polygonally. This makes proper illumination and shading of individual symbols possible. However, in many cases a polygonal representation is far too expensive. A single 3D arrow can easily contain 30 or more triangles. In this case even a single slice sampled rather coarsely at 50^2 points gets too big to be manipulated interactively on a standard graphics workstation. Some alternative representations are shown in Figure 2. The simpler symbols are built of simple line and point primitives only. Usually these primitives are colored, but no shading is applied.

A straight-forward extensions of a simple arrow has been proposed by de Leeuw and van Wijk [8]. Their *vector field probe* or *icon* not only indicates field magnitude and direction, but also properties obtained from a first order field expansion. At a particular point all first order derivatives are computed, yielding the field's Jacobian matrix. This matrix is decomposed into characteristic components like curvature and torsion, which are then represented by a set of geometric symbols. An example of the de Lueew–van Wijk–icon is shown in Figure 3. Because of its size and geometric complexity only few icons can be displayed in a single image. However, by interactively moving a single probe in space valueable insight into the field's local structure can be obtained. The method has received considerable attention in the past. Originally it has been developed for visualizing fields from computaional fluid dynamics. While in such fields velocity usually varies only within a small range, in the vicinity of a point charge field strength may become arbitrary high. Therefore it is not a good idea to visualize field strength by geometric scaling. This argument is true for ordinary arrows as well. As a workaround, a logarithmic scale might be used or field strength might be encoded by color. Also, since a Coulomb field has vanishing rotation, the encoding of torsion in the original icon is superfluous in our application.

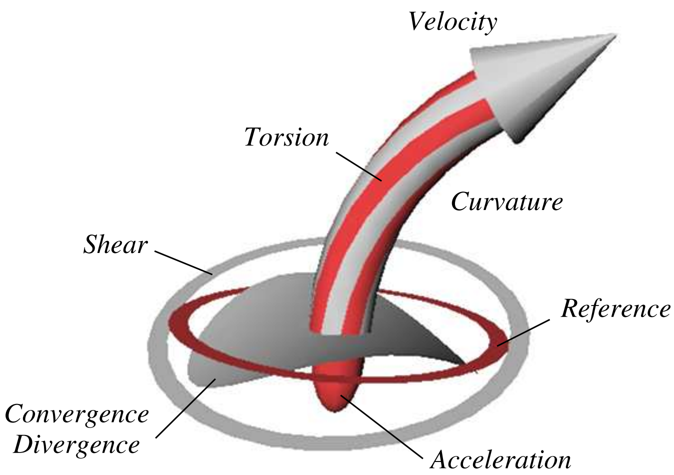

Figure 3: Vector field probe of de Lueew and van Wijk.

3.3 Field Lines

Field lines or integral curves have been used to depict the directional structure of a vector field since the introduction of the field concept to science more than 150 years ago. While arrows or icons only reveal local structure of a field, integral curves encode global information. This makes them a powerful method in visualization and, more generally, an important concept in mathematical analysis. For a stationary vector field $\boldsymbol{v}$ an integral curve $\boldsymbol{\sigma}$ is defined by an autonomous ordinary differential equation

$$\frac{d\boldsymbol{\sigma}(u)}{du} = \boldsymbol{f}(\boldsymbol{\sigma}(u)), \quad \text{with } \boldsymbol{\sigma}(0) = \boldsymbol{x}. \tag{6}$$

Note, that for time-dependent fields these curves do not correspond to particle trajectories. In contrast to field lines, trajectories of a time dependent vector field may intersect. Trajectories show at once the integral effect of the field variation over time, while field lines represent a snapshot of the field at one instance. In the following we will concentrate on stationary fields, since these are most relevant in the context of quantum chemistry.

It can be proved that there is a unique solution of Equation 6 for all $\boldsymbol{x}$ if the field is differentiable, or more general, if it fullfills Lipshitz's condition. For the case of an electrostatic field defined by a set of point charges this condition holds everywhere, except for the point charges itself. At these singularities an infinite number of field lines begin or end.

For field line integration a number of methods can be used. Any numerical integrator has to approximate the continuous curve $\boldsymbol{\sigma}$ by a set of discrete samples. In order to guarantee the accuracy of this discrete approximation in areas of high curvature, the step-size should not remain fixed, but should be adapted to the local characteristics of the field by means of some error estimation scheme. For most of our examples we used a rather simple and robust embedded adaptive Runge-Kutta integrator of 4th order [7]. While an adaptive integrator usually returns sparse discrete samples with varying distances, in visualization often many regularly spaced samples are needed. To achieve this so-called *dense output* the field line has to be interpolated within a sparse interval. Usually the accuracy of the interpolation is one order less than the order of the integrator. For a 4th order integrator a cubic interpolation polynomial is obtained, which corresponds to straight-forward Hermite interpolation. To evaluate the interpolation polynomial very quickly, a forward differencing scheme should be used.

After the field lines have been computed they have to be displayed on a graphics computer. To get high quality images, individual lines are often represented using small cylindrical tubes. The tubes may be colored to visualize an additional scalar quantity, e.g. field strength or the electrostatic potential. Cylindrical tubes have the advantage that standard illumination models can be applied during rendering. Diffuse and specular light reflections very much improve the spatial impression of the resulting images. However, representing field lines polygonally is computationally quite expensive. Therefore only a very limited number of lines can be displayed and manipulated interactively. This means, that the seed points for the individual lines have to be selected very carefully to reveal the important features of a vector field.

This limitation has been relaxed somewhat by the introduction of a new rendering method which exploits the texture mapping features available on modern graphics computers [17]. Instead of drawing tubes made up of many little polygons, field lines are rendered as ordinary line primitives using standard scan-conversion algorithms. Unfortunately, proper illumination of such line primitives is not supported in standard graphics libraries like OpenGL. However, it turns out that a Phong-type reflection model for lines can be implemented using ordinary texture mapping combined with clever initialization of the so-called texture transformation matrix T. At each vertex of a field line the current tangent vector is specified as a 3D texture coordinate. During rendering this coordinate is multiplied by T. Via this matrix multiplication all necessary dot products contained in the illumination model are computed. The transformed texture coordinates are used to read the correct light intensity from a texture map. An example of such a texture map is shown in Figure 4. One can clearly identify the specular reflection on top of a diffuse background.

With the ability to render a huge number of field lines at interactive rates, new methods for distributing field lines become feasible. In particular

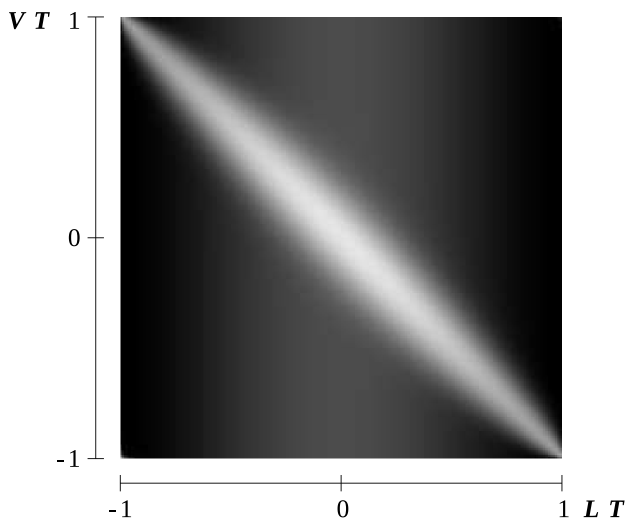

Figure 4: Texture map used for illuminating a line segment with tangent vector $\boldsymbol{T}$. $\boldsymbol{V}$ denotes the viewing direction, $\boldsymbol{L}$ the light vector.

we propose statistical methods. The user may select an interesting region of the field interactively. Within this region seed points may be distributed either homogeously or with a probability proportional to some scalar quantity, e.g. vector magnitude. This quantity loosely defines some degree of interest at a particular point. Other strategies are also possible, e.g. a combination of multiple interest functions or a non-linar transformation of a given interest function. In particular, if the original function varies over several orders of magnitude – like vector magnitude in electrostatic fields – transformations similar to histogram equalization in image processing turned out to be useful.

3.4 Surface Texturing

A common problem with standard methods for vector field visualization is limited spatial resolution. If arrows or icons or even field lines are placed too densely, the image rapidly gets cluttered and no information at all can be encoded anymore. Recently a number of so-called texture based methods have been proposed to tackle this problem. The idea is to synthesize a texture or a textured surface, which faithfully represents the directional information contained in a vector field. In principle texture resolution is only limited by the size of a single pixel. The most successful texture-based methods used in vector field visualization are spot noise [18, 9] and line integral convolution (LIC) [6, 16]. In both methods a random input texture or white noise is blurred along local vector field direction. While in spot noise often 2D filter

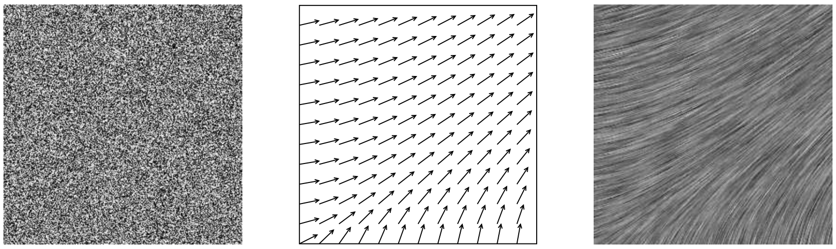

Figure 5: In line integral convolution a random input noise is blurred along the field lines of a vector field.

kernels of elliptical cross-section are used, in line integral convolution 1D filter kernels are applied, which follow exactly the field line pattern of a vector field. In addition, spot noise is an iterative process, while in LIC all pixels of an image are processed step by step. In the following we will concentrate on LIC, since this is a somewhat more accurate method. In particular we will outline an algorithm for computing the convolution very efficiently for the whole image.

In line integral convolution a noise function T is blurred along the field lines or integral curves of a vector field $\boldsymbol{v}$. Blurring results in highly correlated pixel values along the field lines, whereas perpendicular to them almost no correlation appears. Therefore LIC images provide a clear visual impression of the directional structure of $\boldsymbol{v}$. This is illustrated in Figure 5. Mathematically the blurring is described by the convolution integral

$$I(\boldsymbol{x}) = \int_{s_0-L_f}^{s_0+L_f} k(s-s_0)\, T(\boldsymbol{\sigma}(s))\, ds, \tag{7}$$

with $\boldsymbol{x} = \boldsymbol{\sigma}(s_0)$. In this equation k denotes a one-dimensional filter kernel of length $2L_f$, and $\boldsymbol{\sigma}(s)$ denotes a field line parametrized by arc-length s.

The convolution integral has to be calculated for every pixel at least once – a prohibitively expensive task if no coherences can be exploited. In case of a constant filter kernel k a fast evaluation strategy can be established by calculating only the change of the integral while stepping along a field line. This exploits the coherence between neighbouring points on a single field line, while not affecting visual appearance. The convolution integral is approximated by $2N_f+1$ equidistant discrete samples

$$I(\boldsymbol{x}_0) = \frac{1}{2N_f+1} \sum_{i=-N_f}^{N_f} T(\boldsymbol{x}_i), \quad \boldsymbol{x}_i = \boldsymbol{\sigma}(s_0 + i\,\frac{L_f}{N_f}). \tag{8}$$

Typical values of L_f comprise 10 to 40 pixel lengths. Samples should be taken with an increment of half a pixel. Again we use Runge-Kutta methods with adaptive step-size control and error monitoring for field line integration. After having computed $I(\boldsymbol{x}_0)$ it is easy to update intensity along a field line via

$$I(\boldsymbol{x}_{i\pm1}) = I(\boldsymbol{x}_i) - \frac{T(\boldsymbol{x}_{i\mp N_f})}{2N_f+1} + \frac{T(\boldsymbol{x}_{i\pm(N_f+1)})}{2N_f+1}\,. \tag{9}$$

This fast intensity update is the main idea of the so-called fast LIC algorithm. The equation is evaluated for N_s samples in both directions. Usually a single pixel will be covered by multiple field lines. Therefore all samples falling into that pixel have to be averaged. At the end accumulated intensities have to be normalized against the actual number of hits per pixel. There are several strategies which can be applied to obtain an favourable field line distribution, i.e. a distribution where every pixel is covered by as little field lines as possible. Obviously this is an important criterium for the overall performance of the algorithm.

4 Applications

We will now apply the visualization methods discussed in the previous section for analysing electrostatic fields obtained from quantum chemical simulations. The fields have been computed using the NAO/PC technique described in section 2. This method approximates the electronic charge distribution of a molecule by a set of non-fractional point charges. The electrostatic field defined by these point charges has been resampled on a uniform cartesian grid of sufficiently high resolution (101^3 nodes). Within the grid cells we interpolate all vector components trilinearly. Resampling is much faster than evaluating Coulomb's law directly for a huge number of point charges, although it means some loss of accuracy. All visualization methods have been implemented within the framework of an highly interactive modular software system originally designed for medical applications. The system makes use of the OpenGL and Open Inventor graphics libraries. For all examples truely interactive frame rates can be obtained of a fast desktop graphics workstation supporting hardware texture mapping, e.g. SGI Indigo 2 High Impact.

4.1 Benzene

The benzene molecule is a highly symmetric cyclic carbon compound. Its six carbon atoms and six hydrogens lie in a plane. Below and above the plane the charge centers representing the π–orbitals are located. In Figure 6a the zero potential surface is shown in white. This surface separates the areas of negative potential from the areas of positive potential. Electrostatic field lines connect positive hydrogens and positive carbon cores with the negative electronic charges. In Figure 6 the electrostatic field in the symmetry plane is depicted using small arrows, which are scaled and colored according to field strength. While the structure of the field is hard to understand from this image, line integral convolution (Figure 6d) provides much more detail information. Again color denotes field strength. However, the information contained in the LIC image is only two-dimensional. The spatial structure of the field becomes somewhat more obvious by rendering three-dimensional field lines using proper illumination, cf. Figure 6c. The spatial understanding can be improved very much by investigating the field lines interactively from different points of view. Compared to illuminated field lines or line integral convolution the use of the vector field probe has been reported quite poor by most users. Since this method gives no global view of the field, it should only be used to highlight certain features in combination with other methods.

4.2 Thymine Nucleosid

The nucleic acid thymine is contained in DNA molecules. Beside adenine, guanine, and cytosine, it is one of the four base compounds used to encode genetic information in a cell. In Figure 7 the thymine is visible as the six ring in the lower left part of the image. It is coupled to a D-ribose molecule, which is the five ring structure in the upper right. Together both components form a so-called nucleosid.

Since the molecule does not have a simple planar symmetry like the benzene, the structure of the electrostatic field is much more difficult to understand. In a single image only parts of the whole field can be visualized. In particular in Figure 7 the interaction between two OH-groups of D-ribose and a free oxygene of thymine is highlighted. While the potential around the single oxygene is negative, the potential around two hydrogens of the OH-groups is positive. Electrostatic field lines directly connect these areas. The projection of the field lines into a plane is depicted in Figure 7a using line integral convolution. Again the spatial structure of the field becomes more clear if three-dimensional field lines are drawn using proper illumination. In Figure 7b the field lines are colored according to electrostatic potential. To get a better understanding of the location of positive and negative areas, in Figure 7c transparent iso-potential surfaces are shown in addition to the field lines. From this image it is obvious that the field lines are oriented perpendicular to the iso-potential surfaces.

5 Conclusions

In this paper we surveyed a number of methods designed for vector field visualization in general and for visualization of molecular electrostatic fields (MEF) in particular. Because the MEF is always oriented perpendicular to the electrostatic potential (MEP), the display of iso-potential surfaces can help to get a better understanding of the field structure, too. Standard methods like arrow plots or more advanced variants like the vector field probe are not well suited for visualizing the complex vector fields occuring in a molecular situation. Texture based methods like line integral convolution are able to encode directional information at a much higher resolution. However, these methods are inherently two-dimensional. Applying them to simple cutting planes only reveals the structure of the projected field. In principle it would be possible to apply textures to non-planar surfaces too, although it is not quite clear what surfaces should be selected. A better method for understanding the three-dimensional structure of a MEF is to draw a large number of field lines, taking into account diffuse and specular light reflection. The ability to investigate the resulting scenes interactively from different view points is very helpful. In some sense the iso-potential surface, illuminated field lines, and line integral convolution visualize different aspects of the field. Therefore we propose to use these methods simultaneously in an integrated visualization environment. In future, means for highlighting certain features of molecular electrostatic fields would be desireable. In additon, since these fields tend to be very complex, methods for visual data reduction are needed.

References

[1] L. F. Ten Eyck, J. Mandell, V. A. Roberts, M. E. Pique: *Surveying Molecular Interactions With DOT*, Supercomputing '95; ACM 1995; `http://www.sdsc.edu/CCMS/Papers/DOT_sc95.html`

[2] P. C. Mishra, A. Kumar: *Mapping of Molecular Electric Potentials and Fields*, in: *Molecular Similarity II*, (ed. K. Sen), Springer, 1995.

[3] *Neglect of Diatomic Differential Overlap*, for details see:
J. A. Pople, D. L. Beveridge: *Approximate Molecular Orbital Theory*, McGraw–Hill, New York, 1970; J. A. Pople, D. P. Santry, *J. Chem. Phys.* **43** (1965) S129; M. J. S. Dewar, E. G. Zoebisch, E. F. Healy, J. J. P. Stewart, *J. Am. Chem. Soc.* **107** (1985) 3902; J. J. P. Stewart, *J. Comp. Chem.* **10** (1989) 209,221

[4] VAMP 5.6: G. Rauhut, A. Alex, J. Chandrasekhar, T. Steinke, W. Sauer, B. Beck and T. Clark, Erlangen 1995

[5] G. Rauhut, T. Clark, *J. Comput. Chem.* **14** (1993) 503

[6] B. Cabral, L. Leedom, *Imaging vector fields using line integral convolution*, Proceedings of SIGGRAPH '93 (Anaheim, California, August 1-6, 1993). In *Computer Graphics* 27, 1993, ACM SIGGRAPH, pp. 263-272.

[7] P. Deuflhard, F. Bornemann, *Numerische Mathematik II: Integration gewöhnlicher Differentialgleichungen,* Verlag de Gruyter, Berlin, 1994.

[8] W.C. de Leeuw, J.J. van Wijk, *A probe for local flow field visualization*, Proceedings of Visualization '93, Nielson and Bergeron, Eds., IEEE Computer Society Press, 1993, 39-45.

[9] W.C. de Leeuw, J.J. van Wijk, *Enhanced Spot Noise for Vector Field Visualization*, Proceedings of Visualization '95, Nielson and Silver, Eds., IEEE Computer Society Press, 1995, pp. 233-239.

[10] W.E. Lorensen, H.E. Cline, *Marching cubes: A high resolution 3D surface construction algorithm*, Computer Graphics 21:4 (1987), pp. 163-169.

[11] W.E. Lorensen, *Extracting Surfaces from Medical Volumes*, In Visualization'94 Course Notes: Volume Visualization Algorithms and Applications, pp. 26-45, 1994

[12] D. Moore, J. Warren, *Mesh Displacement: An Improved Contouring Method for Trivariate Data*, Technical Report TR-91-166, Rice University, Department of Computer Science, 1991.

[13] C. Montani, R. Scateni, R. Scopigno, *A modified look-up table for implicit disambiguation of Marching Cubes*, The Visual Computer, 10(6), pp. 353-355, 1994.

[14] B.K. Natarajan, *On generating topologically consistent isosurfaces from uniform samples*, The Visual Computer, 11(1), pp. 52-62, 1994.

[15] S. Röll, A. Haase, M. von Kienlin, *Fast Generation of Leakproof Surfaces from Well-Defined Objects by a modified Marching Cubes Algorithm*, Computer Graphics Forum, 14(2), pp. 127-138, 1995.

[16] D. Stalling, H.C. Hege, *Fast and Resolution Independent Line Integral Convolution*, Proceedings of SIGGRAPH '95 (Los Angeles, California, August 6-11, 1995). In *Computer Graphics* Annual Conference Series, 1995, ACM SIGGRAPH, pp. 249-256.

[17] M. Zöckler, D. Stalling, H.-C. Hege, *Interactive Visualization of 3D-Vector Fields using Illuminated Streamlines*, Proc. IEEE Visualization '96, Oct./Nov. 1996, San Francisco, pp. 107-113, 1996.

[18] J.J. van Wijk, *Spot noise-texture synthesis for data visualization*, Proceedings of SIGGRAPH '91 (Las Vegas, Nevada, 28 July - 2 August, 1991). In *Computer Graphics*, 25, pp. 309-318.

Figure 6: (page 18) Benzene molecule. In the upper left the surface of zero potential is shown in white. The red surface encloses areas of high positive potential. In the other images the molecular electrostatic field is shown directly by means of an arrow plot, illuminated field lines, and line integral convolution.

Figure 7: (page 19) Thymine nucleosid. The images highlight parts of the molecular electrostatic field. In the upper image line integral convolution is applied to a planar cutting plane through the molecule. In the middle image illuminated field lines are shown. Red color means positive electrostatic potential, green negative potential. In the lower image transparent iso-potential surfaces have been blended in.