Automated Illustration of Molecular Flexibility

Aaron Bryden, George N. Phillips Jr., Michael Gleicher, *Member, IEEE*

Abstract—In this paper, we present an approach to creating illustrations of molecular flexibility using normal mode analysis (NMA). The output of NMA is a collection of points corresponding to the locations of atoms and associated motion vectors, where a vector for each point is known. Our approach abstracts the complex object and its motion by grouping the points, models the motion of each group as an affine velocity, and depicts the motion of each group by automatically choosing glyphs such as arrows. Affine exponentials allow the extrapolation of non-linear effects such as near rotations and spirals from the linear velocities. Our approach automatically groups points by finding sets of neighboring points whose motions fit the motion model. The geometry and motion models for each group are used to determine glyphs that depict the motion, with various aspects of the motion mapped to each glyph. We evaluated the utility of our system in real work done by structural biologists both by utilizing it in our own structural biology work and quantitatively measuring its usefulness on a set of known protein conformation changes. Additionally, in order to allow ourselves and our collaborators to effectively use our techniques we integrated our system with commonly used tools for molecular visualization.

Index Terms—Illustration, Motion, Molecular Visualization.

1 Introduction

Static images illustrating the motion or flexibility of molecules are valuable, but difficult to create. Our core application is depicting the results of normal mode analysis (NMA) of protein structures through automated illustration. Background on NMA is provided in Section 2.1, but briefly, it provides information on how segments of a protein molecule may move, indicating a velocity vector for each atom in a molecule. Other types of molecular flexibility analysis, including hinge finding and factoring of molecular dynamics traces, provide similar data, and can also be visualized with our approach. In fact, data where velocity information is known for a collection of points (a form of vector field data) is common across science and engineering, and our approach may have broad applicability. In Section 2.2 we describe why traditional vector field visualization approaches are ineffective for the types of problems we consider.

Displaying the flexibility or the potential motion of a molecule is challenging. It may have hundreds or thousands of distinct points (atoms), all moving in different, but coordinated ways. The specifics of each atom's movement are rarely of interest, as the analysis models do not provide that degree of fidelity. When fine details are of interest, they are studied only after understanding the more coarse-grained movement of the overall molecule. Therefore, our approach to illustrating these complex motions is to focus on creating abstractions that convey the key aspects of the potential changes. Specifically we are looking at large potential conformational changes of large fractions of a given molecule which are indicated by the vector field and our visualizations must reflect this.

- *Aaron Bryden is with the Department of Computer Sciences, University of Wisconsin, Madison, E-mail: abryden@cs.wisc.com.*
- *George N. Phillips, Jr., is with the Department of Biochemistry, University of Wisconsin, Madison, E-mail: phillips@biochem.wisc.com.*
- *Michael Gleicher is with the Department of Computer Sciences, University of Wisconsin, Madison, E-mail: gleicher@cs.wisc.com.*

Our approach to motion depiction is to augment the geometry of the object with 3D glyphs, such as arrows, as shown in Figure 1. Each glyph summarizes the motion of a portion of the object. Our approach provides techniques for automatically dividing the object into parts that move coherently, modeling these coherent motions, and synthesizing the appropriate glyphs. The 3D glyphs are created automatically. We have implemented the approach and integrated it into a system for normal mode analysis of proteins, however our system can read and display other types of velocity field data such as the results of principal component analysis performed on molecular dynamics.

Motivation: The most common method for showing motion is animation. Moving imagery is great for many applications but it is impractical in print, which still dominates scientific dissemination. Even when animated display is possible, static depiction is still valuable. A static display can provide a quick summary of movement, without requiring the viewer to wait through a long sequence. Furthermore, static depictions can be much more effective than animations when it is desirable to quantify how different pieces of a moving object move in relation to each other or how different objects move differently because static illustrations facilitate direct comparisons rather than requiring the viewer to remember aspects of the motion. In cases where comparisons between multiple conformations of a protein or multiple different modes are desired, illustrations are more useful than animations because they allow quick visual comparisons rather than requiring remembering details of one animation while watching another.

Sometimes the output of NMA is thought of as a set of displacement vectors. More precisely, it is a local linearization about the current state of the molecule. If the state of the molecule changes, the vectors will also change. These velocities are commonly used to create the displacement based animations which are used to interpret normal mode results. However, because these velocities necessarily must change as the molecule changes state, any actual displacements of the

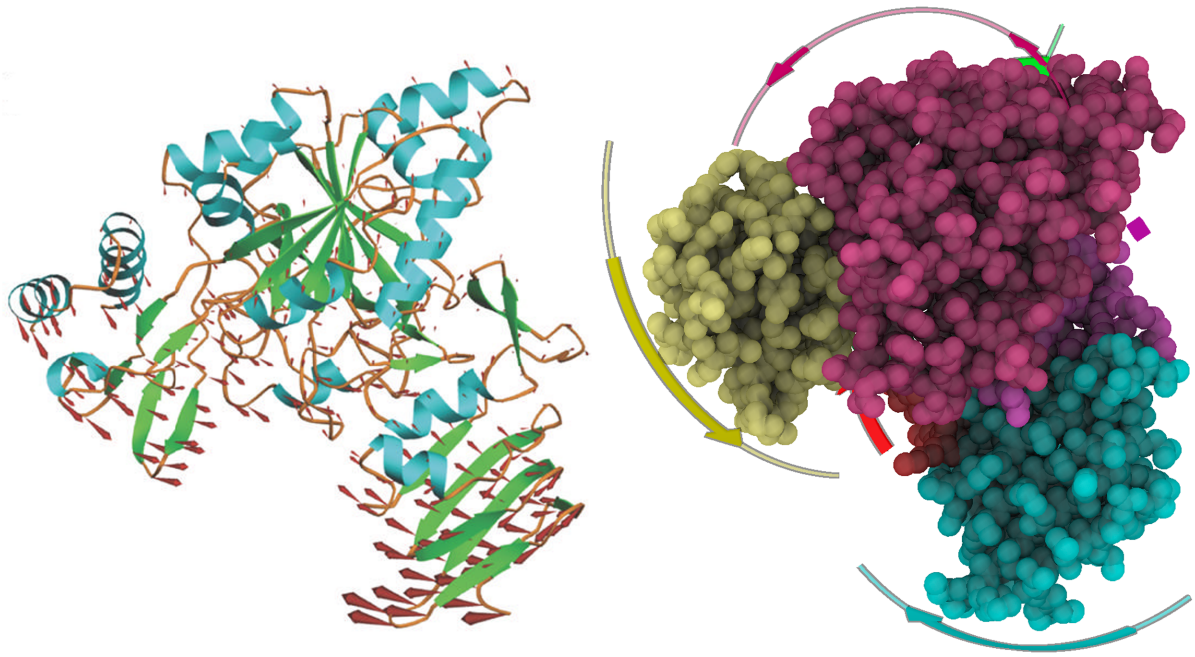

Fig. 1. Left: illustration of a normal mode of a protein taken from a journal publication [24]. Right: the same data, illustrated automatically by our approach. Note that we display all atoms as spheres because this depiction is effective at showing the groupings, whereas the journal figure shows a cartoon drawing of the protein (PDB:2ICY), showing the secondary structure. Our approach can be combined with most molecular depictions, including cartoon drawings (see Figure 10 for an example).

objects are extrapolations because the given velocities are only correct for the exact configuration which they are generated in. In current applications[1], these extrapolations are done in a simple way that distorts the objects and may give incorrect impressions of the motions. Structural biologists and others who use normal mode analysis can be trained to understand these limitations and interpret the animations correctly. It is our intent that our illustrations effectively complement existing visualization techniques using animation for understanding normal mode analysis.

1.1 Overview

NMA provides a collection of points, each with a position and a vector. This input to our system is the result of most commonly used normal mode analysis techniques. These techniques are commonly used among structural biologists. This collection of points and associated vectors representation is how most normal mode analysis convey molecular flexibility. In our approach, each illustration represents one basis vector or mode. A naïve (but common) approach to visualizing this vector field is to show an arrow for each point (or a subset of the points) indicating its vector. An example of such an illustration, taken from a chemistry publication [24], is shown in Figure 1. There are many issues with such depictions. They quickly become cluttered, relatively small flexible groups are hard to see, the significant features of the flexibility, particularly how groups move in a coordinated fashion, are difficult to discern, and the depiction of linear velocities misrepresents the likely trajectories.

These problems can be seen at a smaller scale in the simple example of Figure 2. It is important to note that the interpretation of this as two rigid objects rotating is only one possibility. However, given knowledge of the science and other assumptions, such interpretations are likely to be preferred. In our core molecular application, scientists often seek to identify near rigid components within proteins. Also, the interpretation of a number of points as a rotation provides a concise and easy to depict description of the motion.

Under the interpretation of Figure 2 as two rotations issues in the individual arrows can be seen. The arrows are not really the trajectories of the points, if nothing else they lack the circular arc paths of a rotation. Extrapolation of the positions of the points is impossible. Noticing that the velocities form a rotation may be easy in this example, but might be difficult in a more complicated case of a more subtle or complex motion intermixed within other points. Noticing the groupings, or even the significant feature of rigid structures, may be difficult because the members of the group do not have similar velocities.

Our approach to visualizing vector fields addresses these issues. To help manage the complexity, we abstract the complex flexibility of the entire object by dividing the object into groups, each of which has a coherent instantaneous motion. Grouping not only reduces clutter but also highlights coordinated movements which are important features.

1. All tools for generating molecular animations from normal mode analysis we are aware of, including the very common elNémo [32], perform simple linear extrapolation.

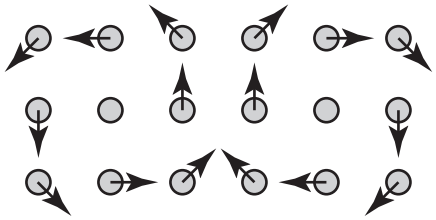

Fig. 2. A simple example of a 2D velocity field illustrates several of the issues faced by our approach. The movement can be grouped into two rotating subsets of particles. When the instantaneous velocities are drawn as straight lines, they fail to convey the curved trajectories of the rotations. The motion of a each group includes a variety of different directions, and the most logical grouping for a particle may not necessarily be with other particles with similar velocities. In more complex situations, it can be difficult to see the logical grouping, however our approach is able to find and depict them.

Furthermore, in our core application, grouping into domains that move together is a commonly used technique in analysis and is relevant and useful to the underlying science that the model is serving. The instantaneous motion of each group is approximated using a relatively simple model, however, the model is capable of capturing many important motions. Specifically, Section 3 describes how our approach employs affine velocity models. These models can exactly represent important categories of motion, such as rotations, and approximate others. By finding groups that fit these models, our approach can identify nearly rigid structures and extrapolate the curved trajectories they create as they rotate. Even if the model is only an approximation to the actual motion of the group, it provides a concise description that is convenient for depiction. It should be noted that our model cannot fully capture some potential conformational changes of proteins indicated by normal mode analysis such as twist or screw motions but in these cases the grouping mechanism will still effectively find affine groups that together can approximate these motions.

In Section 4 we describe how our approach automatically finds groups, by seeking collections of points that fit a common affine velocity model and are spatially compact. This first step of our approach takes the collection of point positions and velocities and produces a segmentation of the points where each segment has a fit motion model. The next step of our approach automatically selects glyphs to indicate each of these motion models. The glyphs are 3D objects designed to convey the motion that the models extrapolate to, for example showing the arc that a rotating point will follow, not just its vector. Section 5 describes the design issues in creating glyphs as well as an algorithm that automatically creates effective glyphs.

We have created a prototype implementation of our approach within our system for performing and visualizing normal mode analysis of proteins. Section 6 describes this system and some of the results we have obtained with it. We have also integrated this system into the PyMOL[2] molecular visualization software package, allowing us to get our system into the workflow of our structural biologist collaborators. The system demonstrates that we can automatically create depictions of motions, as well as provide for user adjustment to tune the results. To date, we have only applied our approach to molecular visualizations, however, our approach and implementation are more general.

Contributions: The overall contribution of this paper is to provide a comprehensive, automated approach for creating static depictions of the coordinated, complex motions of molecules. To our knowledge, it is the first approach that can integrate abstraction, modeling, and depiction of motions for objects as complex as proteins. The key insight of our approach is that by using affine models for the vector fields of groups, we can have compact descriptions of group motions that are rich enough to capture important categories of movements, yet are simple and efficient enough for fitting to be used within clustering algorithms. Specific algorithms are provided for clustering with group fitting and for creating glyphs from the groups' motion. The result is a system that not only creates illustrations of normal mode motions automatically, but is able to create illustrations with features (such as curved arrows) that are challenging to create manually.

2 Background and Related Work

2.1 Target Domain: Normal Mode Analysis

Our motivating application is in helping biochemists understand the large scale motions of proteins. The motion of proteins includes small, sub-nanometer conformational changes that lower the energy barriers in catalytic processes, intermediate scale conformational changes that support switching between functional states, or large scale unfolding/refolding transitions.

Large motions of proteins are difficult to simulate using detailed models. Therefore, scientists often study them through the use of coarse-grained models. These are models for protein motion that simplify either the structure of a protein or the forces that act upon in a way that makes computation and interpretation of results easier, for example replacing groups of atoms with a single group. Normal Mode Analysis (NMA) is a popular tool for coarse-grained examination of protein motion. In NMA, an energy model is used to determine the plausible configurations of the molecule [4]. The models range from simple mass-spring systems to detailed computations considering complex force fields. In NMA, the Hessian of the energy function is evaluated and diagonalized, resulting in a set of directions (in the configuration space of the molecule) that the molecule can move without a significant increase in its energy. Actual motions of the molecule are most likely to contain significant components in these directions. Specifically, each direction, or mode, consists of a set of vectors for each block (commonly a single amino acid in a protein) used in the NMA. These vectors represent a local linearization of the energy function at the molecule's current state. These

2. http://www.pymol.org

vectors represent the relative direction and relative magnitude of motion of the different pieces of the protein. They do not specify meaningful magnitudes of actual displacement, only relative (to other pieces of the molecule) magnitude and direction of movement.

Coarse grained NMA often uses individual, low-energy modes or individual (as opposed to aggregated) sets of vectors to understand potential diffusional behavior over a large (in molecular terms) timescale. In this use of NMA, structural biologists typically use individual modes or sets of vectors as the basis for understanding how different pieces of protein potentially move to allow or disallow access to potential binding pockets or how those potential changes support or don't support other functionally relevant changes in the proteins conformation. In our experience, this examination of specific modes consists of either viewing plots of each individual vector attached to its block or viewing animations of the molecule moving in the direction of the vectors. Typically, the 2 or 3 lowest energy modes are examined. It is this examination of individual modes or sets of vectors that our system is intended to enhance. Other analysis that are performed using the results of NMA include finding the potential large groups or dynamic domains that the molecule consists of. An introduction to the theory and applications of NMA is provided by Cui et al. [9].

The analysis of proteins using differential motions of nearly-rigid blocks has been considered by crystallographers for a long time [8]. Recent algorithms automatically segment proteins into nearly rigid groups [27]. However, this work does not consider the depiction of large-scale motions.

There has been considerable work on displaying the motion of proteins through animation, and effective systems are common. For example, Daae Lampe et al. [11] use the GPU to accelerate interactive animation of protein structures using NMA. Work on static depiction of protein motion [21], [29] has focused on conveying the positional uncertainty from vibrations, rather than the larger scale movements we consider.

In our system we use our own implementation of an anisotropic network model as introduced by Eyal et al. [13]. In our implementation each amino acid is viewed as its own block and we use the C-alpha atom position as the block location. For particularly large molecules it is possible to block by larger chains of amino acids but this produces less accurate results. These block locations are used to create a mass-spring network where each amino acid is a mass and springs at equilibrium length are created between each pair of amino acids within a specified cutoff distance. Our system evaluates the Hessian of the energy function derived from this mass-spring network and diagonalizes it. The result is a set of directions or vectors that all of the amino acids can move in. Other tools use the results of this analysis to displace the molecule along these vectors in order to make animations. Like these systems, we make the assumption that all the atoms in an amino acid move together. In addition to the anisotropic network model, we are able to import basis vectors from other NMA systems such as elNémo [32] and NMA systems created by our collaborators.

2.2 Vector Field Visualization

Abstractly, the problem we consider is a variant of vector field visualization. Because of its importance in fields such as fluid dynamics, weather, and medical data interpolation, vector field visualization has developed as an academic discipline, with a rich literature. Most of the work in this area focuses on flow visualization, where the vector field is tied to spatial positions and a stream of particles is considered to flow through the field, with new particles entering and exiting as necessary. Molecules (and other structural analysis applications) do not have these attributes. There is a fixed set of particles and the velocity field is not stationary (i.e. it moves with the particles).

A simple vector field visualization technique is to display the vector for each sample/particle, or a subset of them. The display of the vectors, as small lines or arrows, are sometimes referred to as hedgehogs, and the flow visualization community has explored issues in their design, including using attributes of the hedgehogs to portray other information such as uncertainty [28] in addition to the vector field. The hedgehog approach is dominant in current molecular motion depiction practice, and can be seen in Figures 1 and 2. Hedgehog approaches do not scale well to complex motions, but they do inspire us to consider glyphs design as a mechanism to convey information about a motion.

More sophisticated vector field visualizations recover trajectories of sample particles by solving the differential equations governing their motions numerically [25]. Well known methods include particle advection, streamlines, and line-integral convolutions. Such methods are difficult to apply when the field is non-stationary, as accurate integration would require many re-computations of the vector field. Also, integration is problematic on irregular grids, and rigid structures give rise to fields that are notoriously prone to integration error. While numerical integration-based flow visualization methods do not apply to our applications, they do inspire us to consider using trajectories extrapolation by integration for depiction. However, our approach creates per-group models whose equations of motion can be integrated symbolically.

Our vector field depiction approach is most akin to the approaches of simplified representation using arrows [34], iconic visualization [36] and streamarrows [22], less common approaches to vector field visualization. In these approaches, key features of a field are identified or the field is segmented, and glyphs (or icons) are used to convey them. While we build on their use of sparse glyphs to convey a vector field, we use the approach to convey the overall sense of the motion, rather than specific details. Specifically, the goal of these methods is to show the flow field itself while our goal is to show near-rigid motions of pieces of a flexible object. Our methodology, application, and visual style is different than theirs.

2.3 Approaches to Motion Depiction

The depiction of motion in static images has a long tradition in art, Cutting provides a survey of the various approaches that have been used over the past few millennia [10]. Comic book artists have a particularly rich vocabulary of motion depiction [23].

The computer graphics community has explored many of the artistic approaches in both illustration applications [3], [30] and video applications [20]. For instance, Collomosse et al. analyzed the motion of tracked features in a video sequence and added cartoon-style motion cues in post-production video such as lines, ghosting and deformation cues [7]. In the area of illustration, Götzelmann et al. analyzed 3D instructional animations in order to automatically produce accompanying illustrative instructions [17], drawing arrows to show animation paths and Joshi and Rheingans provide a good overview of illustration inspired techniques to visualize time-varying data [19].

We are not the first to explore the use of expressive arrows to depict motion in a computer graphics application. Inverse storyboarding systems [12], [16] generate motion arrows to indicate camera movements in frame, Telea et al. used arrows to show vector fields at different levels of simplification [34] and recent work [6] has considered the use of arrows to indicate human movements in images created from motion capture. However, these works consider very different domains with longer movements of simpler objects than in our application or of pure flow visualization on regular grids.

2.4 Mathematical Foundations

A key insight in our approach is the use of an affine model to determine vectors. The applicability of such a model in the case of rigid transformations is well studied. Chapter 2 of Murray et al. provides a lucid explanation of rigid motions, how the differential models are derived, and the insights behind the exponential maps required to find the trajectories from the vector models [26].

The more general case of using matrix exponentials for motion models was introduced to the computer graphics community by Alexa [1]. They have been applied to numerous applications, including camera control in 2D [15] and 3D [18]. Our use of the techniques differs from these other applications as we compute the differential transformations directly, rather than relying on the computation of matrix logarithms.

3 Motion Modeling

The overall goal of our system is to effectively illustrate molecular flexibility using individual modes or sets of vectors obtained using normal mode analysis. The input to our system is a single set of vectors or mode and the position of the molecule and the output is an illustration. The two key steps in creating this illustration are grouping for abstraction and effective illustration of the motion or flexibility of these groups. The key to both of these steps is a motion model that can be fit to groups of point velocity pairs with measurable error and that can be effectively extrapolated for illustration purposes. This section describes a motion model that fits these requirements. In this and other sections we refer to groups of points and associated velocities. These points are the positions of the blocks (typically amino acids) used by the coarse-grained NMA.

We model the motion of groups of points/particles given the positions and vectors. Our approach considers a collection of n particles, each provided with a position and vector, denoted $\mathbf{x_i}$ and $\dot{\mathbf{x}}_\mathbf{i}$ respectively.

A group is a set of particles whose motions (i.e. their vectors) fit (or are at least well-approximated by) a common model. We denote group j as $\mathscr{G}_j$. That is for all points $i \in \mathscr{G}_j$,

$$\dot{\mathbf{x}}_\mathbf{i} \simeq \mathbf{f_j}(\mathbf{x_i}),$$

where $\mathbf{f_j}$ is some function that determines vectors from positions, specific for each group. We seek to find simple forms for the functions to provide concise descriptions that capture the gross description of the movement.

We can measure how well the model fits the provided data in a least squares sense by:

$$e_j = \sum_{i \in \mathscr{G}_j} \|\dot{\mathbf{x}}_\mathbf{i} - \mathbf{f_j}(\mathbf{x_i})\|^2. \tag{1}$$

Our approach is to model the vectors using an affine model. That is,

$$\mathbf{f_j}(\mathbf{x}) = \mathbf{M_j}\mathbf{x} + \mathbf{t_j}, \tag{2}$$

where $\mathbf{M_j}$ is a 3×3 matrix, and $\mathbf{t_j}$ is a 3-vector. These 12 parameters of the model are most conveniently viewed as a $4x4$ matrix in homogeneous coordinates,

$$\mathbf{A_j} = \begin{bmatrix} \mathbf{M_j} & \mathbf{t_j} \\ \mathbf{0} & 1 \end{bmatrix}$$

so that the function can be applied by matrix multiplication.

Given the set of points in a group, finding the 12 parameters of the affine model involves minimizing Equation 1, which is a linear least squares problem. Each point provides a constraint on the model, specifically, each dimension of each point provides a linear constraint on a row of $\mathbf{A_j}$. Using homogeneous coordinates (e.g. extended 3-vectors with a fourth component with value 1 [31]), the linear constraints all have the form

$$w_i \ \mathbf{x_i} \cdot \left[{}_\mathbf{0}\mathbf{A_j}\right]^\mathbf{T} = w_i \ {}_0\dot{\mathbf{x}}_\mathbf{i}$$

where w_i is a weight of how much this constraint should contribute to the solution, and the notation uses the leading subscript to indicate an element (i.e. the leading zero refers the top row of $\mathbf{A_j}$ and the first element of $\dot{\mathbf{x}}_\mathbf{i}$). Note that this equation is repeated for each of the 3 dimensions, but the left most part is common. Collecting these constraints leads to sets of linear equations with the same matrix, 4 unknowns corresponding to the entries in the row of $\mathbf{A_j}$, and a right hand side corresponding to the collection of the appropriate components of the vectors.

The sets of linear constraints are solved in a least squares fashion and there is a separate constraint for each dimension. Because the systems are small (4 unknowns), we solve them by forming the normal equations, which leads to the 4×4 linear system

$$\left[\sum_{i \in \mathscr{G}_j} w_i \ \mathbf{x_i} \otimes \mathbf{x_i}\right] \left[{}_\mathbf{0}\mathbf{A_j}\right]^\mathbf{T} = \sum_{i \in \mathscr{G}_j} w_i \ {}_0\dot{\mathbf{x}}_\mathbf{i} \ \mathbf{x_i} \tag{3}$$

shown here for the first component where $\otimes$ is the vector outer product here producing a 4×4 matrix. Note that the matrix, formed as the weighted outer products of the positions,

is constant across all dimensions so it can be factored once. Because these linear systems are positive definite and symmetric, our preferred implementation uses Cholesky factorization [35] to provide an efficient mechanism for determining the model given a set of points, vectors, and weights.

3.1 Affine Exponential Motion Models

The affine motion model of the previous section is convenient because it is linear, and can be fit efficiently by solving small linear least squares problems. However, its power comes from the fact that for many important types of motions, the vectors are affine functions of the positions and the model fits exactly.

Recall that the model determines the vectors as affine functions of position. The motions, i.e. the trajectories of the points, must be determined as the solution of the ordinary differential equation given by the vectors. The actual motions (trajectories) seem more intuitive than the vector fields: it is more common to talk of "an object translating" than "all points have the same constant velocity."

Affine motion models are sufficient to model the motions of a rigid body. If a set of points ($\mathscr{G}_j$) is undergoing a rigid transformation (i.e. a combination of rotation and translation), then Equation 2 holds exactly, with $\mathbf{M_j}$ being a skew-symmetric matrix with its diagonal elements being zero. A lucid introduction to these concepts is provided in Chapter 2 of Murray et al. [26]. The elements of the skew-symmetric $\mathbf{M_j}$ are the angular velocity. A more general class of transformations, such as scales and shears, can be created by removing the restrictions on $\mathbf{M_j}$. For our work depicting the flexibility of molecules we use the full, unrestricted affine model because it effectively captures the kind of near-rigid motions are commonly encoded in the analysis that we are illustrating.

It is straightforward to encode the skew symmetry constraints on $\mathbf{M_j}$ in the least-squares fitting process. Rather than finding all 12 entries in $\mathbf{A_j}$, the fitting process would determine the 6 constrained degrees of freedom. In practice, we have found this unnecessary. All examples in this paper, and in our work on real domain problems, have used the fully general solution.

In order to effectively depict the motion of a group with similar motion, it is necessary to produce a path of motion that can be used to create a glyph. In order to do this, we extrapolate a path using the affine motion model starting from an initial position. Given only the initial positions and vectors for each point, we assume that the vectors remain constant during extrapolation. The simple approach would be to consider points independently, so their constant vectors lead to linear paths. Instead, we consider the vectors of groups, given by the velocity model, assuming that the model computed for the initial configuration remains constant. For a rotational motion this would be equivalent to computing the angular velocity in the initial configuration and assuming that the angular (as opposed to the point-wise) velocity remains constant. More generally, we assume that the model fit to a group's vectors $\mathbf{A_j}$ remains constant. The vectors of the points depend on the current position of the points, so that their trajectory is determined by:

$$\mathbf{x_i}(t) = \int_{u=0}^{t} \dot{\mathbf{x}}_\mathbf{i}(u)du = \int_{u=0}^{t} \mathbf{A_j}\mathbf{x_i}(u)du = e^{t\mathbf{A_j}}\mathbf{x_i}(0) \quad (4)$$

if point i is in group j so that its vector is determined by Equation 2. Note that the derivation in Equation 4 uses only basic calculus, albeit with the slightly unusual aspect of having a matrix form for the exponential.

If the transformation is rigid, the upper left part of $\mathbf{A_j}$ ($\mathbf{M_j}$) will be skew-symmetric with a zero diagonal, and the matrix exponential $e^{t\mathbf{A_j}}$ can be computed in closed from by Rodrigues' formula [26]. For the more general case of $\mathbf{A_j}$ being an arbitrary matrix, we use the efficient, iterative algorithm presented by Alexa [1].

If the initial vectors of a group suggest a twisting motion such as a rotation or spiral, the affine model (12 entries of the matrix) fit to these vectors will encode it, and the use of this matrix in the exponential mapping of Equation 4 will give the expected curved trajectories.

4 Finding Groups

Combining the particles into groups is a central component of our approach to illustrating molecular flexibility. Grouping is the primary mechanism we use for abstraction in order to reduce the amount of information that needs to be presented to the viewer. Groups are also important since we will define the motion for them, rather than for individual points.

In some cases, groupings may be provided with the positions and vectors. For example, if a motion clustering algorithm's results or hypotheses about motion coordination are to be examined, the grouping would come from a domain science-specific preliminary step. In such cases, motion models are fit to each group, and the grouping step described in this section is skipped. Our implementation supports using provided groupings. However, we have noted that groupings that are meaningful for the domain science are not always effective for motion depiction. This observation has lead to the automated grouping methods we now describe.

Our approach seeks to segment the collection of points into groups in a way that will best lead to effective depiction. There are many criteria for good groups:

1) Each group should fit its model well, that is the sum of squared errors from the velocity model should be minimized. Note that this does not necessarily mean that the group has similar vectors.
2) Larger groups are generally preferred, as this leads to a greater degree of abstraction.
3) Groups should be connected, not having sub-parts scattered around.
4) Groups should convey the level of abstraction desired by the user.

These criteria often conflict: for example, larger groups often have less perfect fits to their models (and therefore more error).

Grouping for motion abstraction provides a clustering problem that is challenging for a number of reasons. First, we have many potentially conflicting goals to consider and cannot cluster based solely on similarity, which rules out many

standard clustering algorithms (such as the common k-means [37]). Second, groups must be of at least a certain size, otherwise there are insufficient constraints to fit a motion model, which again rules out a number of standard algorithms that build groups up from individual elements (e.g. standard hierarchical clustering). Third, the affinity of a cluster is not based on the values themselves, but rather on the quality of the fit of a model to the values, which requires model fitting to be interleaved within the clustering process as in an Expectation/Maximization (EM) style approach. Fourth, the connectedness criterion is not well handled by most clustering algorithms.

Our solution to motion abstraction combines hierarchical clustering and EM algorithms, with an explicit bootstrapping component to handle the small groups. Like hierarchical clustering, it greedily assembles groups by combining smaller ones. Like EM algorithms, the approach interleaves model fitting and cluster formation. We chose an approach based on hierarchical clustering because it allows the user to vary the level of abstraction.

A key step within our algorithm is to fit a motion model to a group as described at the end of Section 3, solving Equation 3. However, to accommodate smaller groups, as well as to provide more spatial coherence in the fitting process, we include points within 10 angstroms of those in the group in the fitting process. Points that are members of the group are given a weight of 1, while the added neighbors are given weights of $1-d/10$ where d is the distance from the closest point the group. The neighbor weights are also reduced by a factor of $1/k$ where k is the number of points in the group. This prevents the degenerate cases that can result from smaller groups. We taper off the influence of surrounding points as the groups grow rather than simply not using them after a certain group size because their is no definite cutoff for a safe group size.

Our algorithm proceeds as follows:

1) The algorithm is initialized by creating a group for each point.
2) For all pairs of neighboring groups, a temporary group is created, a motion model is fit to each temporary group and the error is calculated using Equation 1
3) The pair with the least squared error in the fit model is merged, and its error is stored.
4) Steps 2 and 3 repeat until only 1 group remains and the result is a binary tree containing all nodes
5) All the clusters that remain unmerged at the desired error level (default parameter is .5) are filtered by size such that only clusters of size 5 or greater are made final groups.

As in traditional hierarchical clustering, our algorithm produces a binary tree of groups. This allows for rapidly changing the number of desired clusters without re-running the clustering process. It is also possible to tune the algorithm by using different criteria for group quality other than just least squared error. For example, penalizing larger groups will cause the algorithm to find more uniformly sized groups. Similarly, a metric of model suitability can be used to encourage groups that are meaningful in the scientific domain, for example, a measure of deviation from skew-symmetry can be incorporated to encourage rigid groups. In practice we have found simply using least squared error to be quite effective. Because our clustering algorithm takes n^3 time to complete we have used several acceleration strategies to make it usable on medium to large sized proteins. For small proteins (400 amino acids and less) the clustering takes less than 5 seconds and for large proteins (2000-3000 amino acids) the clustering takes around 30 seconds and the results can be cached. More details on performance are given in the results section. Because hierarchical clustering produces a tree of groupings, we allow the user to select the level of abstraction that they desire by either selecting a desired error level or moving up and down the level of abstraction by either merges or splits of existing visible clusters without recomputation of the clustering (for a given motion our system only needs to cluster once for varying levels of abstraction).

5 Depicting Motions

Given the abstraction of the motion by grouping the points and fitting velocity models to these groups, the remaining step of our approach determines how to depict the motion of each group. Our observation is that showing the trajectory of a small number of points is sufficient to convey the velocity of a group. It is important to note that while we are showing instantaneous velocities of groups, we do this using trajectories because that is what is available in a static figure. This use is well understood in the community that utilizes flexibility analysis. A single trajectory is sufficient to convey a rigid velocity (or the near rigid aspects of a velocity) well. Therefore, our depiction strategy is to choose appropriate points and indicate their trajectories. Note that since we can compute the trajectory from an arbitrary starting position (from Equation 4) using a group's motion, we are not limited to only computing points within the group. In fact, it is preferable to choose points away from the objects so that the drawn trajectories do not interfere with the object itself.

There are several issues in indicating the trajectories, that we have resolved through the design of the glyphs used to depict them. Figure 3 shows visual examples of each of these criteria.

1) Connection between a glyph and its group is important. We address this by using color to indicate the distinct groups and using matching colors for the glyphs. The glyphs are also designed to remain close to their associated group. We use a small set of distinct colors in order to make sure that users can distinguish between and reference groups by name.
2) If the trajectory is short, it may not curve sufficiently to suggest the rotation. This is problematic for groups moving slowly. We address this by decoupling the depiction of motion "shape" and magnitude. Our glyph design uses a "ribbon" that extrapolates the trajectory of the starting position for a sufficient distance to convey its shape. A smaller arrow that is a subset of the ribbon is used to indicate magnitude.
3) The magnitudes of the motions are relative. We address this by choosing the arrow lengths so that they show rel-

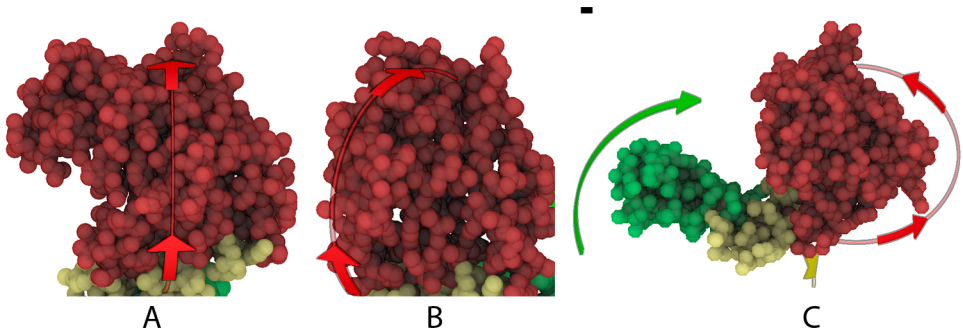

Fig. 3. A. Glyphs and their associated groups are assigned the same color and glyphs are placed close to their group. B. The red group has a small velocity and therefore a short arrow. In order to effectively show the path of the red group, a ribbon that extends beyond the arrow is used. C. The green group has a larger velocity than the red group. To show this, the green arrow is longer than the red arrow. Note that these velocities are relative velocities and assume the same frame of reference as the normal mode analysis.

ative velocities. The longest arrow is the fastest motion, and the other arrows are sized relative to that.

The following subsections detail our methods for automatically creating glyphs that fit this design.

5.1 Depicting The Trajectory of Groups

The main choice in creating a glyph for a group's motion is the choice of the starting position for the trajectory that represents the group's velocity. A second choice is determining the length of the trajectory, effectively deciding the interval of time over which the the equation of motion will be integrated. These two choices are intertwined: a good starting point is defined by being a position that generates a good path. Our procedure for identifying the best point and path is to sample a collection of possible starting points, generate the paths from each, and assess each one to choose the best.

We make three requirements for a valid trajectory for a glyphs (Figure 4 shows a set of valid trajectories for a group) :

1) The closest particle to any point in the path must be in its associated group. This criterion ensures that glyphs are visually associated with their group and prevents glyphs from interfering with each other.
2) Every point in the path must be more than a specified distance from all particles. This criterion prevents the path from occluding the object too severely.
3) Every point in the path must be less than a specified distance from some particle. This criterion prevents the glyphs from significantly changing the shape of the viewing area, or becoming too disconnected from the object.

Given a starting position $\mathbf{x}(0)$, a path can be generated by using the motion model to generate a trajectory of new points until that trajectory violates one of the three criteria. A path is the longest possible trajectory through the starting point.

Our algorithm generates a set of starting positions by starting in the geometrical center of points of each group and finding the point p in each direction for a number of directional samples that lie a certain distance outside all of the particles in all of the clusters. If the closest particle to p belongs to group $\mathscr{G}_j$, then p is added to the list of potential starting points. An illustration of the sampling is shown on the left of Figure 4.

Our algorithm scores each candidate starting position's path using the following criteria:

1) The arc-length of the path. This criteria prefers longer paths because they are usually easier to see and visible from more viewpoints.
2) The inverse of the distance between the plane the path lies in and the center of its associated group. This criteria prefers paths that loop around the center of the group, as such paths are easier to associate with the group.

Our selection algorithm has 3 parameters: the weighting of the two scoring criteria, and the two distance cutoffs. The values of the distance cuttoffs are 2.75 units and 12 units, and the two scoring criteria are equally weighted and normalized to the range of possible values. We chose to weight the scoring parameters equally because we found this to be an effective heuristic because in almost all cases their are large central regions of the exposed piece of a motion group that fit both criteria reasonably well but if we allow one criteria or the other to dominate, it finds edge cases that are undesirable.

Paths are depicted as thin semi-transparent ribbons, with their edges emphasized by outlining. The orientation of the ribbon is chosen to be parallel to the axis of rotation as extracted from the affine model, $\mathbf{M_j}$. We do not directly show the axis of rotation because of occlusion issues and also because the exact axis of rotation is less important than the general direction of relative movement or rotation between different groups in the molecule and the boundaries between the groups.

5.2 Depicting Direction and Velocity

While ribbons are an excellent tool for depicting the trajectory of the motion for the groups, they cannot show the direction of velocity or relative magnitude between groups. For each group, a large variable-length arrow is drawn overlayed on the ribbon, to indicate the direction and relative magnitude of its velocity. The arrows are drawn to lie along the surface of the

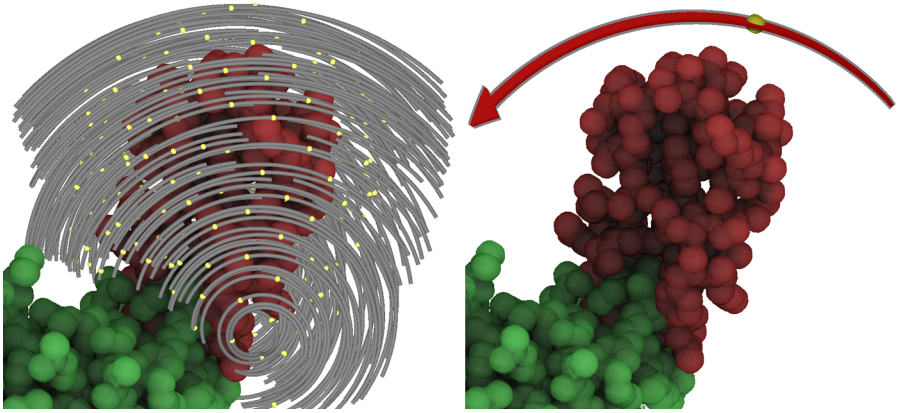

Fig. 4. (PDB:4AKE) On the left the set of possible starting positions (small yellow spheres) and associated paths (grey tubes) is shown for the red group. On the right the selected best starting point (light yellow sphere) and path (red glyph) is shown.

ribbon, but have a slight thickness so that appear to have some mass, and are more easily visible from a variety of directions. The arrow is usually placed in the center of the ribbon except when the arrows are much shorter than the ribbon and then 2 arrows are placed at each third of the ribbon. We use two arrows in this case so that the relative velocity of the group can be understood from a large variety of potential viewpoints.

The length of the arrows are chosen such that they convey the relative velocities of the different groups. Effectively, a duration of time is chosen and the arrow length is how far the starting position moves in that period of time, so faster moving groups have longer arrows. The amount of time is chosen such that no arrow goes beyond its path, but at least one arrow fills its path. An example can be seen in the right hand side of Figure 1. Notice that the length of all arrows are dictated by the length of the green arrow. The lengths are chosen such that the green arrow fills its path, and the other arrows' lengths are proportional to this. In the case of multiple arrows, each individual arrow's length is proportional to the group's velocity.

5.3 Depicting Reliability of the Illustration

In order to ensure that users understand how closely the motion that they believe a glyph depicts matches the actual velocity of a group, we developed techniques for communicating how reliably a given glyph depicts the motion of its associated group. These are optional and in practice are rarely used.

The two issues that are important in communicating the reliability of our illustrations are communicating the general level of error in fitting the motion model to a group and communicating whether or not fit motion model was in fact close to a rigid motion because our glyphs are designed to communicate near-rigid motions and the domain science also assumes a degree of rigidity in groups that move together. We chose to convey the level of error in fitting the motion model to the group by simply adding optional error bars to the glyphs. Rather than using continuous error bars to show the exact level of error we instead chose to either show no error bars in the case where 90% or more of the velocity of a group could be explained with the model, small error bars in the case where 70% or more of the velocity could be explained with the model and large error bars where 70% or less could be explained by the model fit to a group. Figure 5 shows an example of these glyphs.

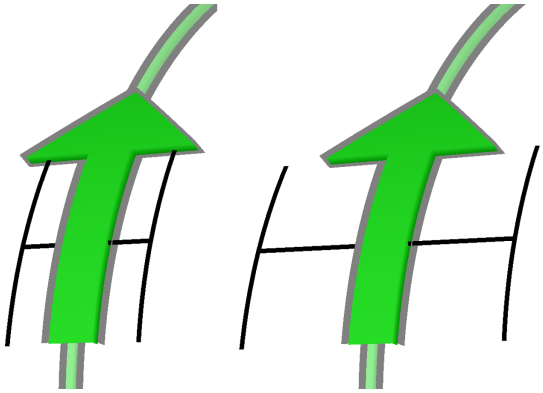

Fig. 5. Depictions of differing amounts of error. The arrow on the left indicates a moderate amount of error in the fit of the motion model and the arrow on the right indicates a high amount of error in the motion model.

The second issue, communicating whether or not a given group's model depicts a near-rigid motion is somewhat more difficult. First it is necessary to determine whether a groups model was near-rigid or not. We perform this measurement by first finding the nearest pure-rigid motion instantaneous affine model to the fit model and factoring the pure-rigid component and a remainder using matrix subtraction (instantaneous velocity matrices compose by addition and substraction as discussed by Alexa [1]).

Once the instantaneous affine model is separated into a rigid and non-rigid component we measure the energy of each component by recomputing the velocities for each point in the group for both the rigid and non-rigid components. Once

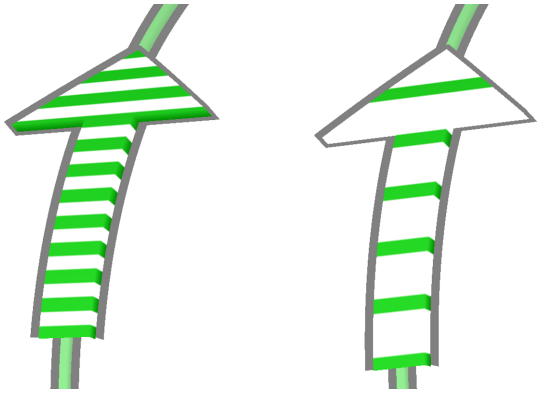

Fig. 6. Depictions of differing amounts of non-rigid energy. The arrow on the left inidicates a moderate amount of non-rigid energy and the arrow on the right indicates a high amount of non-rigid energy.

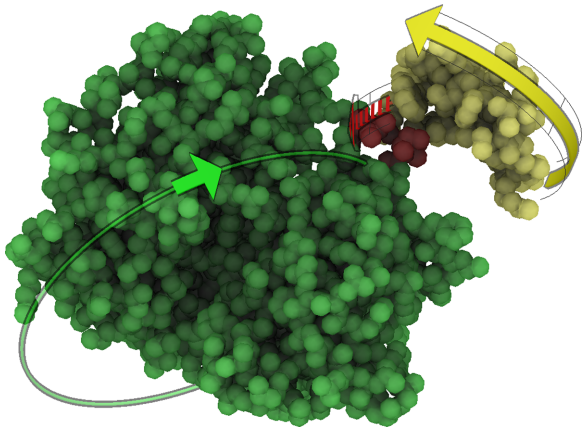

Fig. 7. An example of our reliability depiction techniques. The green group has low error and fits a rigid motion model well. The red and yellow groups have a medium amount of error and the red group does not fit a rigid model well.

we have the rigid energy and the non-rigid energy we then add an optional component to the motion glyphs that lets the user know whether or not a given glyph contains substantial non-rigid motion. Specifically, if a user adjustable threshold (default 25%) or more of energy in the group comes from non-rigid motion we modify the arrow portion of the glyph to have large transparent stripes indicating that the arrow may not tell the whole story of the motion for the group. If a very large amount of the energy (default 75%) comes from non-rigid motion we modify the arrow to have very sparse stripes. Figure 6 demonstrates these modifications to the arrows.

In practice, structural biologists are looking for large, general motions without a high degree of accuracy. Therefore, the reliability is much less important than the abstraction. For this reason we designed our depictions of the reliability such that they can be turned on and off easily and do not show them by default. The main case where they are potentially useful is when looking at the movement of a specific piece and verifying that the claims made by the illustration are accurate. Figure 7 shows an example of our error illustration technique.

6 Results

We have implemented our approach to motion depiction within our system for normal mode analysis of proteins. The images in this paper and the experiments described were created with this system or with PyMOL. The system can perform various types of normal mode analysis to generate potential motions, or can read in the results generated by other tools. All molecular motion examples in this paper are based on normal mode analysis performed by our system.

At present, the system supports displaying molecules using a space-filling representation (showing solid spheres for atoms) and the ribbon representation (depicting the nucleotide backbone of the protein as a ribbon). For space filling representations we use ambient occlusion shading [33] to enhance depth perception. These representations are the fundamental depictions used in domain journals but in some cases, it is desirable to combine the different depictions as well as using a stick and balls depiction. For illustrations that require fine-grained choices of illustration styles and for integration into our collaborators workflow we have implemented PyMOL integration of the system and have also developed a prototype of Jmol [3] integration. An advantage of our illustration technique over traditional per amino acid arrows is that a wider range of molecule depictions can be used without occlusion issues.

The system can allow for user input to improve the quality of the illustrations. However, for the examples shown in the paper, the only manual adjustments were to alter the threshold to select the number of groups in the illustration. All other parameters for depiction, including arrow generation, used their default values. All examples in the paper use the automatic grouping algorithm.

Figures throughout the paper show examples of the illustrations created by our system. These examples show the advantages of our approach. For example in Figure 1 we see that our technique clearly demonstrates how different parts of the molecule move at different rates but also shows the trajectory of motion in the parts with little movement far more effectively than a hedgehog plot can. In Figure 8, the static illustrations allow several motions of the same molecule to be viewed simultaneously and compared. Figures 9 and 13 show additional examples of illustrations created using our system.

In order to assess the effectiveness of our system for creating illustrations of protein flexibility we used two methods of evaluation. First, we used the system to assist our collaborators in understanding the potential large scale changes of several proteins studied by them and their collaborators. Second, we evaluated the effectiveness of our automated illustration technique on a selection of protein crystal structures for which normal mode analysis is either known to be or likely to be interesting. We chose these methods for evaluation because they address the core of our contribution which is abstracting a set of displacement vectors down to a small number of groups and techniques for automatically illustrating the motion

3. http://jmol.sourceforge.net

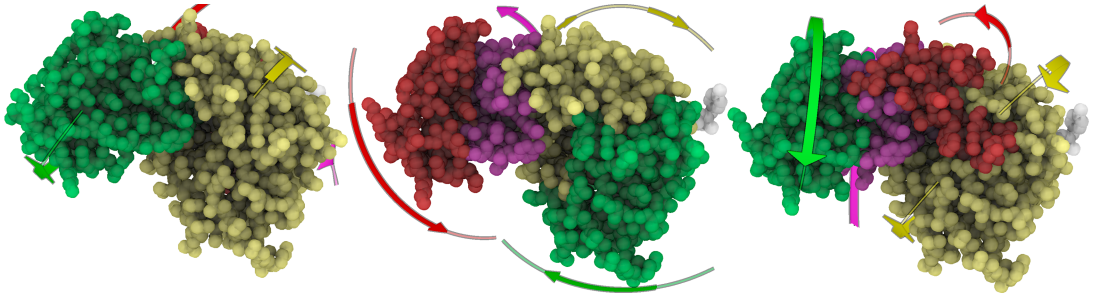

Fig. 8. An illustration of 3 different motions of a single molecule (PDB:1HMY), all taken from a similar viewpoint.

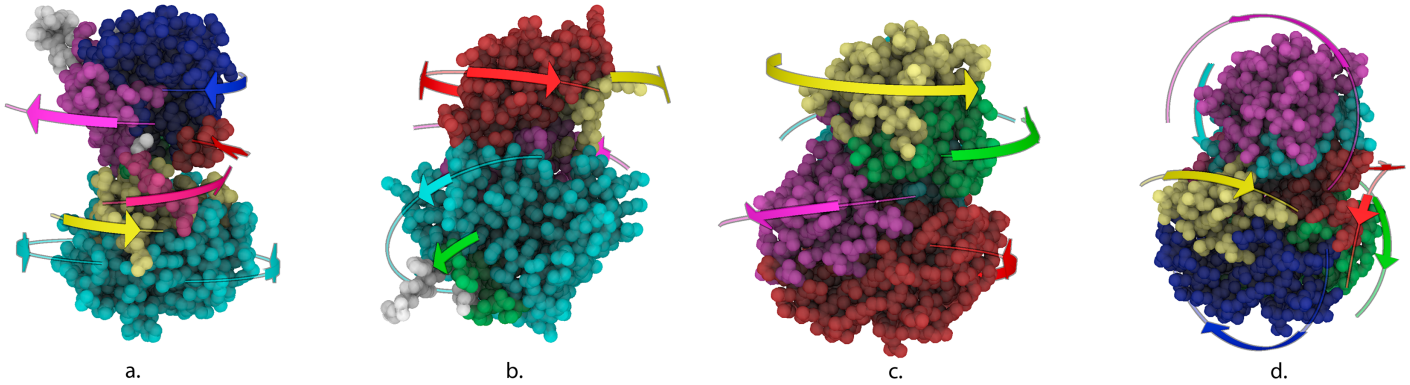

Fig. 9. Additional examples of our illustration technique. a) PDB:1X13 b) PDB:8MHT c) and d) 2 different motions of CalG2 (unpublished) an enzyme that opens and closes its active site while adding sugar substituents to a bacterial antibiotic compound. d) illustrates this open and closing motion effectively.

of those groups in a way that is as effective or more effective than animations created from the displacement vectors.

6.1 Informal Use and Assessment of Our System by Structural Biologists

In order to evaluate the effectiveness of our technique we used our system to examine the flexibility of several proteins in a group setting with our collaborators. At first, while our collaborators responded well to our illustrations they were unable to use them to gain insights into the function of the proteins they were working on because of the limitations of our system's general molecular visualization capabilities. They needed several features present in software such as PyMOL including cartoon renderings, and sequence browsing. In order to address these issues we developed PyMOL and Jmol export capabilities in our system. Specifically in the case of PyMOL export capability we implemented our groupings as colored selection groups in PyMOL and imported the geometry for the arrows into PyMOL. Our Jmol export capability is similar but is integrated into an exported web page that loads Jmol. These capabilities allowed our collaborators to use the visualization methods of their choice to examine the protein structures while maintaining the colorings of our groupings and displaying our arrows.

After this integration with PyMOL was implemented the response was overwhelmingly positive. In all cases where our technique did not catastrophically and detectably fail, our collaborators agreed that the grouping and illustration capabilities of our system made normal mode analysis much more accessible and were able to see which pieces moved together much more quickly than using animation techniques. In most cases, after using our illustrations, our collaborators also wanted to see traditional animations to verify that our illustrations were correct and in all cases they agreed that our illustrations effectively abstracted the motion shown using animation of the protein along the displacement vectors. Figure 10 shows one such protein in PyMOL.

In addition to helping structural biologists understand the protein structures they work with we used our system to illustrate the motion of several molecules where the authors have already used normal mode analysis to make hypothesis about a proteins motion or function in order to demonstrate that our illustrations support these hypothesis. Figure 11 is an illustration of the motion of the domains surrounding the binding pocket in Adenylate Kinase that facilitate catalysis [5]. Figure 12 shows the coordinated motion of the nucleotide binding loop in UDP-glucose Pyrophosphorylase [24]. Our collaborators have confirmed that these illustrations effectively show these motions.

6.2 Quantitative Evaluation on an External Dataset

In order to determine how often our technique succeeds or fails and whether or not it is able to clearly illustrate the motion on a wide variety of proteins we evaluated its effectiveness on an external dataset. We chose to use all known hinge movements for which multiple crystal structures were

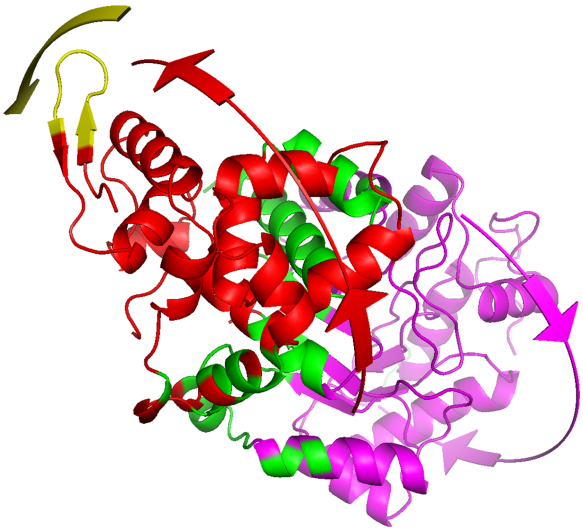

Fig. 10. Our illustration technique integrated into PyMOL. This allows structural biologists to use our illustrations in a familiar, powerful molecular visualization tool. This protein (PDB:4AKE) shows the same motion as Figure 11 with a lower degree of abstraction and from a different viewpoint.

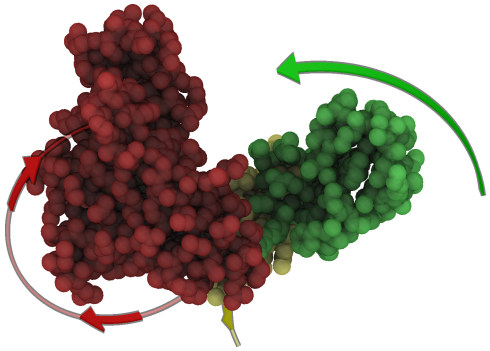

Fig. 11. The lowest energy mode shows the 'clamping motion' of the domains around that binding pocket that occurs to facilitate catalysis as discussed in [5] (PDB:4AKE). Our collaborators have confirmed that this illustration conveys this motion much more effectively than traditional methods.

available in Molmovdb [14], a webserver of curated known protein motions which are based on the availability of crystal structures in different conformations. We chose this dataset because at least one of every two crystal structures in the dataset is likely to have interesting low frequency modes. For each of these crystal structures we determined the following with the guidance of our collaborators who are domain experts: 1. Did our technique work (i.e did the groupings it found make sense)? 2. Does the illustration show the same motion as an animation of the protein along the displacement vectors?

Our technique was effective on 22 out of 33 of the crystal structures we tested it on, and in all cases where it was effective the illustration showed the same potential conformational change as the animation did. The failure case that we encountered is a failure of the clustering algorithm to generate effective clusterings and instead to snake along the backbone of the protein. We found that this occurred in cases where there was a gradual transition between pieces of the protein that move together rather than definite regions of coordinated motion. The failure of our technique in these cases is obvious and does not produce potentially misleading figures. We believe that in these cases, an illustration technique other than groupings with motion arrows may be necessary.

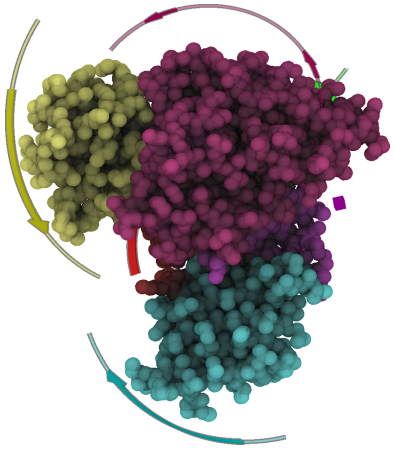

Fig. 12. Using normal mode analysis, the authors of McCoy et al. [24] hypothesized that the motion of the nucleotide binding loop induced by the binding of UDP-glucose or UTP is coupled with the motion of the carboxyl-terminal domain. The yellow group consists of the nucleotide binding loop and the carboxyl domain. Our collaborators have told us that this illustration effectively supports this hypothesis.

6.3 Performance

Our non-optimized prototype can generate illustrations in a reasonable amount of time: clustering is nearly instantaneous on a small molecule of less than 300 amino acids, and about 10 seconds on a medium sized molecule (500-1000 amino acids). For larger molecules, clustering is time consuming. The largest molecule we segmented had 2537 amino acids and took 183 seconds to cluster. The majority of proteins in the PDB are of a size that present our clustering technique with no problems and the clustering is completed with very little wait time. All of the proteins used in our quantitative evaluation finished clustering in less than 10 seconds and the majority completed the clustering process in less than a second. Our performance measurements were done on an iMac with a 3.06 GHz Intel Core 2 Duo processor. The clustering computation time is dominated by the finding of nearby points/groups at each iteration and we have improved this dramatically using approximate nearest neighbor search [2] and expect that improved caching will result in additional speedups. In order to handle larger molecules we would need to add an acceleration strategy, such as blocking, into the clustering.

7 Discussion

In this paper, we have presented an approach for illustrating the complex, instantaneous motions of complex objects. Our

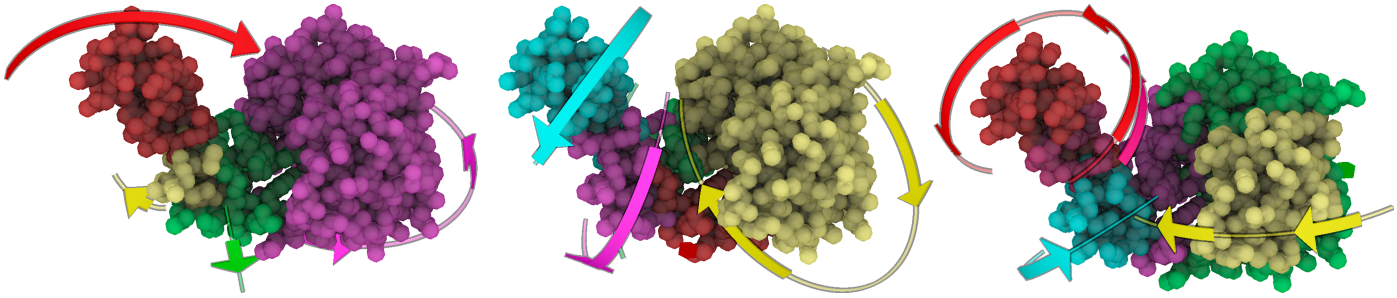

Fig. 13. Illustration of 3 different motions of adenylate kinase (PDB:4AKE).

approach is targeted toward the display of molecular flexibility computations, particularly Normal Mode Analysis. However, its input can be any collection of points with associated vectors. Our approach reduces the complexity of the motion by abstracting it. The moving points are divided into groups such that each group has a consistent motion. Glyphs are generated for each group to convey these motions.

Our conversations with biochemists and professional illustrators who create images for biochemistry publications have suggested the attractiveness of the kinds of illustration created by our approach and when using our tool to view potential motions of new crystal structures, our collaborators affirmed that our tool was much more clear in the presented motion than the traditional illustrative style when used to illustrate both known and unknown motions. Curved arrows in 3D are generally considered too difficult to create manually.

Our method is limited in the complexity of motions it can handle in several ways:

- We consider individual normal mode vectors, and are, therefore, limited to the motions that can be described by a single mode vector. Extending our approach to richer motion models, such as multi-step NMA or combinations/spaces of modes is a potential avenue for future work.
- Affine models are not capable of capturing all possible motions that normal mode analysis can potentially represent. While richer or different motion models could potentially capture these motions as coherent groups, there is a tradeoff between models that are more complex and capable of capturing a wider range of motions, and the efficiency and ease of depiction afforded by simpler models. Piecewise affine velocity models seem to be a sweet spot in this tradeoff in part because even in cases where affine models poorly model a motion, our model is capable of capturing more complexity by using a larger number of groups where the local motion can be effectively described using an affine model.
- Our illustration techniques cannot effectively depict all potential affine motions, in particular those involving large scaling components. In our application this limitation is not problematic because molecules do not undergo significant scaling motions.

Other limitations of our current approach that suggest avenues for future development include:

- At present, we have limited understanding of the perceptual issues in the design of arrow glyphs. Our anecdotal evidence suggests that curved arrows imply rotations, but it is unclear how curvature, radius, and arc length are interpreted to assess the parameters of the motion. For example, it appears that users connote the curvature of an arrow with the indicated velocity.
- We would like to adapt our approach to support static depiction of the range of motions in molecular dynamics trajectories.
- We believe our approach has applicability to applications beyond illustration of molecular flexibility and hope to explore these.

The feedback we have gotten on our results has been very positive and our technique has been used by our collaborators to successfully analyze the flexibility of several protein structures. The number one question we got when showing our technique to structural biologists was 'when will it work in my workflow' and we have integrated our system with the PyMOL molecular visualization software to make that workflow integration possible.

8 Acknowledgements

The authors wish to thank our collaborators in the Phillips lab. This work was supported in part by a grant from NSF award IIS-0946598 and CMMI-0941013 and National Library of Medicine grant R01LM008796 (P.I. Jude Shavlik) and National Library of Medicine grant NLM Grant 5T15LM007359

References

[1] M. Alexa. Linear combination of transformations. *ACM Transactions on Graphics*, 21(3):380–387, 2002.

[2] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.

[3] J. Assa, Y. Caspi, and D. Cohen-Or. Action synopsis: pose selection and illustration. *ACM Transactions on Graphics*, 24(3):667–676, 2005.

[4] I. Bahar and A. Rader. Coarse-grained normal mode analysis in structural biology. *Current Opinion in Structural Biology*, 15(5):586–592, Oct. 2005.

[5] M. B. Berry, E. Bae, T. R. Bilderback, M. Glaser, and G. N. Phillips Jr. Crystal structure of ADP/AMP complex of Escherichia Coli adenylate kinase. *Proteins*, 62(2):555–556, Feb. 2006.

[6] S. Bouvier-Zappa, V. Ostromoukhov, and P. Poulin. Motion cues for illustration of skeletal motion capture data. In *Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, pages 133–140, 2007.

[7] J. Collomosse, D. Rowntree, and P. Hall. Rendering cartoon-style motion cues in post-production video. *Graphical Models*, 67(6):549 – 564, 2005.

[8] D. W. J. Cruickshank. The analysis of the anisotropic thermal motion of molecules in crystals. *Acta Crystallographica*, 9:754–756, 1956.
[9] Q. Cui, I. Bahar, et al. *Normal Mode Analysis: Theory and Applications to Biological and Chemical Systems*. Taylor and Francis Group, 2006.
[10] J. E. Cutting. Representing motion in a static image: constraints and parallels in art, science, and popular culture. *Perception*, 31(10):1165–1193, 2002.
[11] O. Daae Lampe, I. Viola, N. Reuter, and H. Hauser. Two-level approach to efficient visualization of protein dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1616–1623, 2007.
[12] R. Dony, J. Mateer, and J. Robinson. Techniques for automated reverse storyboarding. *IEEE Journal of Vision, Image and Signal Processing*, 152(4):425–436, 2005.
[13] E. Eyal, L. Yang, and I. Bahar. Anisotropic network model: systematic evaluation and a new web interface. *Bioinformatics*, 22(21):2619–2627, Nov. 2006.
[14] S. Flores, N. Echols, D. Milburn, B. Hespenheide, K. Keating, J. Lu, S. Wells, E. Z. Yu, M. Thorpe, and M. Gerstein. The database of macromolecular motions: new features added at the decade mark. *Nucl. Acids Res.*, 34:D296–301, Oct. 2004.
[15] M. Gleicher and F. Liu. Re-cinematography: Improving the camerawork of casual video. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 5(1):1–28, 2008.
[16] D. B. Goldman, B. Curless, S. M. Seitz, and D. Salesin. Schematic storyboarding for video visualization and editing. *ACM Transactions on Graphics*, 25(3):862–871, 2006.
[17] T. Götzelmann, K. Hartmann, and T. Strothotte. Annotation of animated 3D-objects. In *Simulation and Visualization 2007*, 2007.
[18] A. Hawkins and C. M. Grimm. Camera keyframing using linear interpolation of matrices. *Journal of Graphics, GPU, and Game Tools*, 12(3):55–69, 2007.
[19] A. Joshi and P. Rheingans. Illustration-inspired techniques for visualizing time-varying data. In *Proc. IEEE Visualization*, pages 679 – 686, 2005.
[20] B. Kim and I. Essa. Video-based nonphotorealistic and expressive illustration of motion. *Proc. Computer Graphics International Conference*, pages 32–35, 2005.
[21] C. Lee and A. Varshney. Representing thermal vibrations and uncertainty in molecular surfaces. In *SPIE Conference on Visualization and Data Analysis*, pages 80–90, 2002.
[22] H. Löffelmann, L. Mroz, and E. Gröller. Hierarchical streamarrows for the visualization of dynamical systems. In *8th Eurographics Workshop on Visualization in Scientific Computing*, pages 155–164.
[23] S. McCloud. *Understanding Comics: The Invisible Art*. Harper Penrennial / Kitchen Sink Books, 1993.
[24] J. G. McCoy, E. Bitto, C. A. Bingman, G. E. Wesenberg, R. M. Bannen, D. A. Kondrashov, and G. N. Phillips Jr. Structure and dynamics of udp-glucose pyrophosphorylase from arabidopsis thaliana with bound udp-glucose and utp. *Journal of Molecular Biology*, 366(3):830 – 841, 2007.
[25] T. Mcloughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen. Over two decades of integration-based geometric flow visualization. *Computer Graphics Forum*, 29(6):1807–1820.
[26] R. M. Murray, S. S. Sastry, and L. Zexiang. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., 1994.
[27] J. Painter and E. A. Merritt. Optimal description of a protein structure in terms of multiple groups undergoing TLS motion. *Acta Crystallographica*, 62(4):D439–450, 2006.
[28] A. Pang, C. Wittenbrink, and S. Lodha. Approaches to uncertainty visualization. *The Visual Computer*, 13(8):370–390, 1997.
[29] J. Schmidt-Ehrenberg, D. Baum, and H. Hege. Visualizing dynamic molecular conformations. In *Proc. IEEE Visualization*, pages 235–242, 2002.
[30] D. D. Seligmann and S. Feiner. Automated generation of intent-based 3D illustrations. In *Proc. ACM SIGGRAPH*, volume 25, pages 123–132, 1991.
[31] P. Shirley and S. Marschner. *Fundamentals of Computer Graphics*. A K Peters, Natick, Massachusetts, USA, 2009.
[32] K. Suhre and Y.-H. Sanejouand. Elnemo: a normal mode web server for protein movement analysis and the generation of templates for molecular replacement. *Nucl. Acids Res.*, 32:W610–614, July 2004.
[33] M. Tarini, P. Cignoni, and C. Montani. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1237–1244, 2006.
[34] A. Telea and J. van Wijk. Simplified representation of vector fields. In *Proc. IEEE Visualization*. IEEE Computer Society, 1999.
[35] L. N. Trefethen and D. Bau III. *Numerical Linear Algebra*. SIAM: Society for Industrial and Applied Mathematics, 1997.
[36] T. Van Walsum, F. H. Post, D. Silver, and F. J. Post. Feature extraction and iconic visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):111–119, 1996.
[37] R. Xu and I. Wunsch, D. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, May 2005.

Aaron Bryden Aaron Bryden received his B.S. in from Iowa State University in 2004 in Computer Engineering. He is currently a Ph.D. candidate in the Department of Computer Sciences at the University of Wisconsin, Madison in the Department of Computer Sciences. His research interests are in the areas of visualization, and molecular flexibility.

George N. Phillips Jr. George N. Phillips Jr. received his B.A. at Rice University, majoring in both Chemistry and Biochemistry. He completed his Ph.D. also at Rice in Biochemistry in 1976. He was a postdoctoral trainee at in Structural Biology at Brandeis University. He is currently a professor at the University of Wisconsin, Madison in the Department of Biochemistry and an affiliate faculty member in Computer Sciences. His interests are in the connections between structure, dynamics, and function of proteins, using experimental methods and computational methods to establish mechanisms of action.

Michael Gleicher Michael Gleicher is a Professor in the Department of Computer Sciences at the University of Wisconsin, Madison. Prof. Gleicher is founder and leader of the Department's Computer Graphics group. His research interests include visualization, image and video processing tools, and character animation techniques for films and games. Prior to joining the university, Prof. Gleicher was a researcher at The Autodesk Vision Technology Center and in Apple Computer's Advanced Technology Group. He earned his Ph. D. in Computer Science from Carnegie Mellon University, and holds a B.S.E. in Electrical Engineering from Duke University.