ProteinShop: A Tool for Interactive Protein Manipulation and Steering

Silvia Crivelli[1], Oliver Kreylos[2], Bernd Hamann[2], Nelson Max[2], Wes Bethel[1]

[1] Computational Research Division, Lawrence Berkeley National Laboratory

[2]Department of Computer Science, University of California, Davis

Keywords: molecular visualization, protein manipulation, inverse kinematics, interactive methods.

Abstract

We describe ProteinShop, a new visualization tool that streamlines and simplifies the process of determining optimal protein folds. ProteinShop may be used at different stages of a protein structure prediction process. First, it can create protein configurations containing secondary structures specified by the user. Second, it can interactively manipulate protein fragments to achieve desired folds by adjusting the dihedral angles of selected coil regions using an Inverse Kinematics method. Last, it serves as a visual framework to monitor and steer a protein structure prediction process that may be running on a remote machine. ProteinShop was used to create initial configurations for a protein structure prediction method developed by a team that competed in CASP5. ProteinShop's use accelerated the process of generating initial configurations, reducing the time required from days to hours. This paper describes the structure of ProteinShop and discusses its main features.

Corresponding author:

Silvia Crivelli

Computational Research Division,

Lawrence Berkeley National Laboratory,

Mail Stop 50D,

Berkeley, CA 94720

SNCrivelli@lbl.gov

510-486-5061 (phone); 510-486-5812 (Fax)

1. Introduction

A main bottleneck in practical simulation of complicated physical phenomena, including protein folding, is the specification of the complex geometry involved. Today it is possible to numerically simulate extremely complex physical phenomena with an increasingly high degree of accuracy and level of detail. However, current tools used to interactively define complex geometrical structures, which are used as input to numerical simulations, are still immature. At a minimum, they have not evolved at the same speed as our ability to simulate increasingly large-scale phenomena. While ProteinShop was developed for the specific purpose of manipulating protein structures, many of the general paradigms of our framework are applicable to similar problems.

In the context of protein structure prediction via numerical optimization, a significant number of new fold approaches have emerged that incorporate some knowledge from the protein databases. These methods begin with a large set of initial structures and then perform numerical optimizations to improve those structures. Although generating initial configurations is not nearly as time-consuming as the optimization process itself, it is still a tedious process that may take several days to complete. An example of a new fold approach is the stochastic perturbation (SP) method (described in Section 3) that uses secondary structure predictions obtained from servers to generate starting configurations [1]. This method begins with the extended sequence of amino acids and builds secondary structures through local minimizations with soft constraints. Because proteins have thousands of atoms, these local optimizations may take hours or even days to converge. The problem is compounded by the presence of β-strands for which there is a combinatorial number of possible strand alignments, each of which needs to be created with a different constrained minimization. ProteinShop allows a scientist to significantly reduce the time to create a set of structures to a matter of hours. This fact is a significant achievement.

Furthermore, protein structure simulation programs based on energy minimization principles typically were executed in "batch mode," where it was not possible to observe or guide the evolution of the optimization process. ProteinShop makes it possible, when used in combination with an optimization system, to interact with and guide the numerical optimization process. In summary, ProteinShop substantially reduces set-up times for initial protein configurations and allows us to effectively monitor and interact with the optimization process, which allows us in turn to obtain in reduced time significantly better results than with existing methods.

Although several packages exist that can display protein structures [2-11], they do not provide support for the interactive manipulation of protein structures and their subsequent minimization. Among the most well-known packages are RasMol and its derivative Protein Explorer, MOLSCRIPT, Swiss-PdbViewer, Qmol, PyMol, Molden, Chimera, O, and VMD. RasMol, MOLSCRIPT, SWISS-PdbViewer, and Qmol are molecular graphics programs intended for the visualization of proteins, DNA, and/or small molecules [2-6]. They focus on the display of molecules, teaching and generation of publication quality images. The displayed molecule may be rotated, translated, and zoomed interactively and the rendered image may be saved in a variety of formats.

Molden, PyMol, Chimera and O are visualization *and* modeling tools [7-10]. However, the modeling capabilities they provide are intended to deal with small, localized changes of the protein structure rather than with the molecule as a whole. For instance, they can mutate a protein molecule by replacing or adding residues one at a time. Molden supports the creation of a 3D protein from scratch by adding one amino acid at a time and selecting the desired secondary structure. It also allows for the optimization of the protein structure using either the AMBER [12] or CHARMM [13] force fields. PyMol has a list of capabilities that include coarse interactive modeling using “molecular sculpting.” However, the building and sculpting features are not yet available.

VMD was originally designed for the visualization and analysis of biological systems such as proteins [11]. VMD can be used to animate and analyze the trajectory of a molecular dynamics (MD) simulation. In particular, VMD can serve as a graphical front-end for an external MD program by displaying and animating a molecule undergoing simulation on a remote computer [14]. VMD implements a steering system that allows it to be coupled with the molecular dynamics package. The system, called IMD (Interactive Molecular Dynamics), supports manipulation of protein molecules in molecular dynamics simulations with real-time force-feedback (provided by a haptic device) and graphical display [15]. However, the functionalities provided in VMD are not aimed at protein 3D structure prediction research. VMD does not support interactive manipulation of protein fragments, automatic alignment of beta-strands to form beta-sheets, or the visualization of energy functions.

In this paper we present ProteinShop, an interactive visualization and molecular manipulation tool designed to permit users to determine optimal tertiary structure of proteins based on an initial sequence of amino acids. ProteinShop is designed for two purposes: 1) to quickly generate a diverse set of protein structures that can be used as initial configurations for an optimization method for protein structure prediction, and 2) to provide a framework used to interact with and guide an optimization process that may be running on a remote machine. ProteinShop provides support for the following:

- Creation of proteins "from scratch" using the amino acid sequence and secondary structure specifications as input.
- Display of protein molecules in different rendering styles selected by the user.
- Interactive manipulation of the 3D protein structure using a mouse-

based user interface and an inverse kinematics method that changes dihedral angles along the backbone.

- Interactive reassignment of secondary structure types to individual residues.
- Semi-automatic β-sheet formation and global β-sheet adjustments.
- Display of an energy function provided by the user that is recomputed and displayed while the protein is manipulated.
- Monitoring the progress of a protein structure prediction process running on a remote machine.
- Steering a protein structure prediction process.

The following sections describe the structure and main features of ProteinShop. In Section two, we describe the methodology. In Section three, we analyze ProteinShop in the context of the application for which it was originally designed. In Section four, we discuss the current developments to expand ProteinShop's functionality. Finally, in Section five we provide a summary and discuss future research.

2. Methodology

ProteinShop can be used in two phases of the prediction process: the *protein structure creation phase* and the *optimization phase*. For the protein creation phase, ProteinShop interactively creates and manipulates protein structures. It does so using a combination of a build up procedure, which creates the structures adding one amino acid at a time, and an interactive procedure, which manipulates those structures using an inverse kinematics algorithm. The protein creation phase begins with either a prediction file that contains the sequence of amino acids and secondary structure predictions or with a PDB file that specifies atoms and their positions. An example of a prediction file for 1PGX [16] is given by:

```
Pred:CCCCCCEEEEEEEECCCCEEEEEEECCCHHHHHHHHHHHHHHCCCEEEEEEECCCEEEEEEECCCCCCC
 AA :ELTPAVTTYKLVINGKTLKGETTTKAVDAETAEKAFKQYANDNGVDGVWTYDDATKTFTVTEMVTEVPV
```

where the second line corresponds to the initial sequence of amino acids and the first line contains the predictions of whether each amino is part of an α-helix, "H", a β-strand, "E", or a coil, "C", region.

Two modules assist the user during the creation phase. The *geometry generation module*, used when reading a prediction file, builds a *pre-configuration,* which is an extended structure containing α-helices and β-strands according to the predictions. An example of a pre-configuration is shown in Figure 1. One of the uses of the *direct manipulation and visualization module* is to manipulate β-strands into β-sheets producing new folds. This module is used to manipulate a pre-configuration by aligning protein fragments either manually or semi-automatically. Structures created with ProteinShop can be saved at any time in PDB format and used as input in a later session.

ProteinShop is designed to create tertiary structures by allowing users to freely manipulate the backbone dihedral angles φ and ψ.

However, it does not allow users to manipulate the side-chains. Thus, structures created with ProteinShop should be fine-tuned by subsequent local optimizations in order to correct possible close range interactions that may have been created during manipulation. To guide the manipulation with an energy function, ProteinShop can be used with the AMBER [12] energy function that is provided or coupled with a user-supplied energy function.

During the optimization phase, the *control module* provides interactive and user-driven steering of the optimization process. In this context, ProteinShop displays protein configurations produced by the optimization process while running on a remote machine. The user can manipulate any of those structures to achieve a desired fold, and return the manipulated structures back to the protein structure prediction process for further optimization.

ProteinShop is written in C++. It uses the OpenGL library for three-dimensional graphics rendering and FLTK for the graphical user interface. In addition, the program offers standard methods for protein visualization that include Van-der-Waals spheres rendering, bond-stick rendering, and cartoon rendering. Next, we discuss the main capabilities of ProteinShop in the context of the protein creation and control phases.

2.1. The Geometry Generation Module

The purpose of this module is to create pre-configurations --extended protein configurations that do not contain any tertiary structure. These pre-configurations can be used as starting points when creating more complex folds with the manipulation module, which we discuss in Section 2.2.

2.1.1 Creating pre-configurations

ProteinShop creates pre-configurations from an input file, called a prediction file, that contains the amino acid sequence of the protein and a specification of whether each residue is part of an α-helix, a β-strand, or a coil. Secondary structure estimates may be obtained from secondary structure prediction servers [17,18]. ProteinShop creates protein structures one amino acid at a time by reading the sequence of amino acids from the input file, and the atom positions from residue template files. Each template file contains the position of all the atoms in a residue along with their connectivity information. There is one template file for each residue type. A template file models a residue in a local coordinate system. As the chain of amino acids is assembled, ProteinShop applies an "end-of-chain" transformation to reposition local template coordinates to the end of the partial chain. While adding a new residue, ProteinShop sets the dihedral angles φ and ψ of each incremental residue to the “ideal values” that correspond to the specified secondary structure type, then updates the end-of-chain transformation accordingly. Thus, configurations are created with secondary structures already assembled. Creation of a pre-configuration is not computational expensive, and is accomplished in a very short amount of time.

2.1.2 Changing secondary structures

A user may want to change part of the secondary structure of the protein to create new conformations without having to start from scratch with a modified prediction file. To address that need, ProteinShop allows users to interactively change the secondary structure specifications for individual residues "on the fly". Each amino acid residue can be set to any of the three basic structure types and its dihedral angles can be changed to the ideal values corresponding to the new type. Also, a user may use this feature to enlarge short coil regions that may be difficult to manipulate or to "break" long coils into shorter ones for extra control over the manipulation.

2.2 The Manipulation and Modeling Module

With ProteinShop, users may manipulate a pre-configuration or any configuration read in PDB format. ProteinShop supports several manipulation styles: interactive manipulation using an inverse kinematics algorithm [19], Ramachandran plots [20], alignment guides, global β-strand adjustments, and semi-automatic β-sheet formation. Each manipulation style is tailored to achieve a very specific type of result. These styles constitute one of ProteinShop’s primary contributions, and are discussed in this section.

2.2.1 Interactive Manipulation

The interactive manipulation module allows a user to select elements of secondary structure then arrange them to form a desired tertiary structure. Chemical bonds along the backbone between those elements are maintained during movement. An Inverse Kinematics (IK) method [19] is used to compute dihedral angles to achieve the molecular shape created by interactive manipulation. In fact, IK allows ProteinShop to translate 3D motions into bond rotations, which in turn allows the user to manipulate proteins in a very intuitive manner.

To begin manipulation, the user selects an element of secondary structure, i.e., an α-helix or a β-strand. The selected structure will be surrounded by a *3D direct manipulation widget*, depicted as a translucent green box. The 3D manipulation widget can be translated or rotated by dragging it with the mouse, (see Figure 2.)

After a secondary structure has been selected, the user picks one or more coil regions so that all the dihedral angles in those regions will be adjusted as the selected structure is dragged. The picked coil regions are called "active" coil regions and they constitute the *movement buffer* of the IK manipulation. Figure 2 shows the active coil regions highlighted in yellow. This figure illustrates a sequence of manipulations of the pre-configuration shown in Figure 1 to form a 3D structure for 1PGX.

2.2.1.1 Inverse Kinematics for Protein Modeling

Inverse Kinematics is a well-researched subject in the areas of robotics and computer animation. In robotics, it is mainly used to calculate a vector of joint angles for a robot arm that will move the robot's end actuator, its *hand*, to an intended position and orientation in space. Forward Kinematics, the related problem of calculating an end actuator's position and orientation given a vector of joint angles, is a straightforward arithmetic problem that can be solved by basic trigonometric operations. Thus, IK involves solving a set of non-linear equations containing trigonometric terms and general-purpose non-linear solver algorithms can be used, e.g., Newton-Raphson iteration [21].

However, using IK for protein modeling poses several additional problems that narrow the choice of available algorithms. The first problem is one of scale. In robotics, linked assemblies, e.g., robot arms, typically have a small number of joints, usually between six and twelve. In the protein modeling case, activating several long coil regions for manipulation can easily lead to assemblies with more than eighty joints, since each active amino acid residue (except Proline) contributes two rotational joints -- its dihedral angles φ and ψ. Thus, the chosen IK method must be highly efficient to support interactive manipulation of large proteins. The second problem arises from the fact that assemblies with more than six joints have more degrees of freedom than required to obtain an intended position/orientation of the end effector. A user interacting with a large protein expects an intuitive relationship between transformations of a selected structure and the behavior of the active coil regions. More precisely, a small movement of a selected structure should lead to a small change of the active coil regions, and the latter change should be predictable by the user to enable planning of manipulations.

We found that the best IK method for protein manipulation is based on the principle of transposed Jacobians [19]. Although this method is computationally similar to the gradient descent method of multidimensional optimization [21], and thus does not converge as quickly as other methods, it has two major benefits. First, each iteration step can be computed very efficiently; and second, it leads to intuitive coil behavior when extra degrees of freedom are present. The method can be interpreted as a physical simulation, where a user dragging the end effector applies forces and torques to each joint along the linked assembly so that they change smoothly.

The algorithm to calculate dihedral angle changes along a coil region under manipulation can be described as follows:

- Calculate initial vector of dihedral angles (φ_i,ψ_i) from a 3D protein structure.
- While user is dragging the selected structure:
    1. Calculate Jacobian matrix of Forward Kinematics function, i.e., partial derivatives of position/orientation functions for each φ and ψ.
    2. Calculate difference between current position/orientation and

intended position/orientations of the selected structure.

3. Multiply transpose of Jacobian matrix with position/orientation difference to get vector of dihedral angle increments ($\Delta\varphi_i$, $\Delta\psi_i$).
4. Multiply the difference vector with a prescribed step size, and add it to the current vector of dihedral angles.
5. Calculate new position and orientation of selected structure using Forward Kinematics based on changed vector of dihedral angles.

- Apply final vector of dihedral angles to initial 3D protein model.

2.2.1.2 Adaptive Step Size

The algorithm described in the previous section has one drawback -- its dependency on the prescribed fixed step size. If the step size chosen is too small, the iteration will make little progress towards convergence in each step, the manipulation will have very slow update rates, and it will appear to be lagging the user's mouse motion. If, on the other hand, the step size is too large, the iteration can become unstable, and the manipulation will appear to ``thrash'' or fail to converge. Choosing a good step size is a difficult problem since it depends both on the number of joints in the linked assembly and on the current conformation of the assembly with respect to an intended movement.

To address this problem, we enhanced the basic IK algorithm with automatic step size calculation, and adaptive step size control. Each iteration step is split into two steps with half the step size. Thus, we compute the linked assembly's Jacobian matrix twice, and calculate two vectors of dihedral angle increments. After both sub-steps are completed, we calculate the maximum difference between the two dihedral angle increment vectors. If this difference is larger than a prescribed maximum tolerance value, the computed step is rejected, and the step size is multiplied with a fixed factor smaller than one. If the difference is smaller than this value, the step is accepted, and the step size is multiplied with a fixed factor slightly larger than one. The initial step size is still prescribed, but tests showed that the step size adapts to the conformation of the linked assembly very quickly. This simple improvement of the algorithm fixed its stability problems completely, even for linked assemblies with very large numbers of joints, and has the added benefit that the IK algorithm takes larger steps, and converges quickly, when the linked assembly is short.

2.2.1.3 Manipulation Modes

There are two modes of manipulation depending on whether the active coil regions are all on one side of the selected structure or not. In one manipulation mode, depicted in Figure 2 a-d, all the active coil regions are on the same side of the selected structure (in chain order). Let us assume that all active coil regions are located before the selected structure. Then, when dragging the manipulation widget, the protein will be treated as three parts. Dragging the selected structure will cause the protein to move as follows. Part 1, the protein segment to the left of all active coil regions, will remain fixed during dragging. Part 2, the protein segment from the first active coil region to the last active coil region, will be modified during dragging so that the active coil regions change shape and the structures in-between undergo rigid body transformations (i.e., they will be rotated or translated but they will not change shape themselves.) Part 3, the protein segment to the right of the last active region, will undergo rigid body transformations according to the manipulations made by the user. Thus, dragging the manipulation widget will transform the selected structure and the segment of the protein that follows in chain order with respect to the segment of the protein that lies before the first active coil. This mode of interaction is mainly used to align β-strands to form β-sheets.

In the other manipulation mode, depicted in Figure 2 e-h, active coil regions are present on both sides of the selected structure. When dragging the manipulation widget, the protein will be treated as five parts. Part 1, the segment to the left of the first active coil region, will remain fixed during dragging. Part 2, the protein segment from the first active coil to the last active coil to the left of the selected structure, will be modified during dragging so that the active coil regions change and the structures in-between undergo rigid body transformations. Part 3, the segment between the last active coil region to the left of the selected structure and the first active coil region to the right of the selected structure, will undergo rigid body transformations according to the manipulations made by the user. Part 4, the segment from the first active coil region to the last active coil region to the right of the selected structure, will be modified during dragging in the same way as part 2. Part 5 will stay fixed during dragging. Thus, dragging the widget will move the selected structure with respect to the two segments of the protein that are preceding and following the active coil regions. Those two segments will not move with respect to each other. As in the other manipulation mode, shape changes of active coil regions and transformations of intermediate structures are constrained by the IK algorithm. This mode of interaction allows one to move a structure without disturbing the alignments between protein parts on both sides. Therefore, it is used to "fine tune" an already assembled structure.

2.2.2 Ramachandran Plots

The IK algorithm assumes that the backbone bonds can freely rotate and the backbone dihedral angles can adopt any value. However, some combinations of φ and ψ values are more likely to occur than others. The internal energy of a single residue can be represented as a function of φ and ψ and graphs of this function are called Ramachandran plots [20]. ProteinShop visualizes the dihedral angles of active coil regions in the style of a Ramachandran plot to guide the user during manipulation to adopt those combinations that seem more favorable.

2.2.3 Alignment Guides

One of the main uses of our program is the alignment of β-strands to form β-sheets. ProteinShop offers alignment guides that are specifically designed for the purpose of decreasing the interaction time required to form β-sheets. These guides are updated in real time during manipulation. An example of these guides is the hydrogen-bond site-rendering feature that shows hydrogen-bonding sites along the backbone. A hydrogen bond site is the midpoint between the hydrogen and oxygen atoms involved in a hypothetical hydrogen bond. ProteinShop renders a line segment connecting the hydrogen or oxygen atom of each backbone N-H or C=O group to its respective bond site. The bond site itself is highlighted by a dot, which along with the segments, are colored according to polarity as shown in Figure 3 -light blue for N-H, red for C=O. To form a hydrogen bond, a user aligns a blue and red segment by overlaying their "end dots". As the protein changes shape during manipulation, ProteinShop constantly monitors the position and orientation of the hydrogen bonding sites and renders a dashed yellow line when a hydrogen bond is detected.

2.2.4 Beta-strand Adjustments

Structure alignment via IK manipulations will change dihedral angles only inside coil regions, but never inside α-helices or β-strands. To change the shape of β-strands, ProteinShop supports some coherent changes of the φ and ψ angles inside the strands. These coherent changes are used to adjust a strand's shape to form more hydrogen bonds between adjacent strands in a β-sheet. Most of these changes are discussed in Richardson and Richardson [23]. They are called *twist*, *pleat*, *curl*, and *braid*. Figure 4 depicts the effects of twisting, pleating, curling and braiding, respectively.

2.2.5 Semi-Automatic β-Sheet Formation

Even with the manipulation tools described so far, creating β-sheets can be a time-consuming process. To decrease the time required to create β-sheets, ProteinShop supports the automatic formation of hydrogen bonds between two amino acid residues selected by the user. The automatic-bonding feature supports parallel and anti-parallel β-sheet hydrogen bonding. To invoke this feature, the user selects a β-strand and activates coil regions as usual. Next, the user selects two residues, the first from the selected β-strand, and a bonding type. In the anti-parallel case, ProteinShop computes a transformation that moves the selected β-strand so that a double hydrogen bond is formed between the selected residues. Thus, if R_i and R_j are the first and second selected residues, then R_i's N-H group bonds with R_j's C=O group and vice versa. In the parallel case, the transformation will form bonds between the first selected residue and both neighbors of the second one. Thus, R_i's C=O group bonds with R_{j-1}'s N-H group, and R_i's N-H group bonds with R_{j+1}'s C=O group. The IK algorithm automatically updates dihedral angles in all active coil regions to realize the desired transformation. After the two β-strands have been automatically aligned to form these two hydrogen bonds, the user can interactively fine-tune the alignments to form more hydrogen bonds between the two strands, using the interactive manipulation of section

2.2.1 and/or the β-strand adjustment of section 2.2.4.

2.2.6 Saving Motifs

Complex β-motifs can be quickly created by first assembling simple arrangements, and then saving the dihedral angles associated with them. The saved results can be later reused as templates. Thus, if the user wants to design many structures containing, for instance, an antiparallel arrangement of strands 1, 2, and 3, he can save a file containing their dihedral angles and then load it every time he wants to repeat that motif.

2.2.7 Collision Detection and Visualization

The IK algorithm used for constrained protein manipulation allows for the collision of atoms inside a protein in order to give the user more flexibility when performing manipulations. However, atom intersections must then be “cleaned up” to obtain plausible protein structures. ProteinShop calculates and visualizes atom intersections in real time during manipulation. We use a simple grid-based algorithm to quickly find all pair of atoms whose van-der-Waals spheres intersect by more than a threshold value. To visualize intersections, ProteinShop renders red spheres of radii proportional to the depth of an intersection at the midpoint between two intersecting atoms. The collision-detection algorithm is sufficiently fast for use on proteins of any size. Figure 5 illustrates the atom collision-rendering feature of ProteinShop.

2.2.8 Energy Calculation

Initially, ProteinShop was designed for creating and manipulating initial structures used as input for global internal energy minimization. Therefore, it was desirable to interactively calculate a protein structure's internal energy using the same energy function employed by the optimization algorithm. ProteinShop provides a plug-in system that allows dynamically loading of energy computation modules at program startup. This feature allows for coupling of ProteinShop with any energy computation module that takes a set of 3D Cartesian coordinates of atom positions in a protein and computes energy value(s). Currently, the ProteinShop package provides a module that calculates the AMBER [12] energy function.

Furthermore, ProteinShop supports visualization of energy values. To take advantage of this feature, the energy computation module must be able to compute the contribution of each atom inside the protein to the total energy value. ProteinShop visualizes these partial energy values by mapping colors to atoms' van-der-Waals spheres as shown in Figure 6. The ProteinShop's Energy Visualization Dialog is used to color-code atoms' van-der-Waals spheres according to the selected settings. Also, it can be used to define the minimum and maximum per-atom energy values that will be mapped to ProteinShop's green-yellow-red color ramp, and to select any combination of energy components to be visualized.

To provide even more insight, a computation module could keep track of separate partial energies for each atom, based on the separate computations it performs. For example, an AMBER energy computation module considering bond distances, bond angles, dihedral angles, and bonded and non-bonded interactions could store the five components separately per atom. By sending these partial energy components to ProteinShop via a software interface, a user can toggle energy components separately for visualization. Such visualizations help to find and understand energy “hot spots” in a protein. For example, energy “hot spots” can occur with undesirable dihedral angles or for atom collisions.

2.3. The Control Phase

Although ProteinShop does not perform the global optimization process that is usually associated with protein structure prediction, it provides a framework that can be used to interact with an external optimization process that may be running on a remote machine. ProteinShop’s *control module* provides interactive, user-driven monitoring (visual feedback) and steering of an optimization process. In this context, the control module visualizes protein configurations retrieved from a remote optimization code. The control module shows the entire tree of possible solutions and allows the user to select any node from the tree for visualization and manipulation. Furthermore, the control module allows the user to transfer a manipulated structure back to the remote process for further optimization. This approach helps reduce the time to find the solution by rejecting or eliminating large, unproductive subsets of the search space.

The control module is composed of two primary components: a user-provided protein structure prediction program and the visualization/manipulation/communication capabilities provided by ProteinShop. Communication is achieved through an efficient socket connection between ProteinShop and the structure prediction code running on single or parallel processors.

ProteinShop has a server dialog box that allows users to connect/disconnect to a server running an optimization code. In Section 4, we describe how the steering mechanism of ProteinShop helps us to obtain important qualitative insights into the optimization process, to focus the global search on areas of the protein that seem most relevant, and to change secondary structures that prefer to unfold or form a different type of structure.

3. An Application

The stochastic perturbation (SP) method for protein structure prediction developed by a group of researchers in LBNL and the University of Colorado (SP group) is a physics-based method that uses secondary structure information [1]. Such information is used to create initial protein configurations that contain secondary structure according to the predictions. The SP method is composed of two phases. The first phase, called setup phase, generates starting structures that are local minima containing predicted secondary structure. The second phase, called optimization phase, improves upon the starting structures using both global and local minimizations. The setup phase begins with the primary sequence of amino acids and builds secondary structures through local minimizations with soft constraints. These local optimizations may take hours or even days to converge.

The global optimization phase improves the initial configurations through global minimizations in subspaces of the dihedral angles of amino acids predicted to be part of a coil region by the secondary structure prediction servers. The large-scale global optimization problem is solved as a series of small-dimensional ones. The idea is to randomly select subsets from the set of dihedral angles predicted to be coil and then perform global optimizations on those subsets. Thus, the algorithm selects a local minimum from a list of minima and a subset of dihedral angles and performs a small-scale global optimization on that subset using the selected angles as variables while keeping the remaining ones temporarily fixed at their current values. The small-scale optimization produces a number of local minima in the subspace of chosen angles. A number of those conformations are selected for local minimizations on the full variable space. The new local minima are merged into the current list ordered by energy value. The process iteratively repeats until no further lowering of energy is observed between consecutive iterations. Figure 7 shows a block diagram of the setup and optimization phases.

Computationally, the global optimization process can be viewed as a search through a huge tree of possible solutions where the root of the tree corresponds to the primary sequence of amino acids, a sub-tree consists of an initial local minimum and all the local minima generated from it, and the leaves correspond to the local minima most recently found.

The SP method was tested in CASP4, a competition in which groups from around the world submit blind predictions of protein structures. The CASP4 results showed that the SP method has potential for predicting new folds. However, because of the high computational cost of the method, the team predicted only eight targets, the largest of which was 240 amino acids long. In addition, the SP method showed a lack of variety in the configurations created and as a result, produced poor predictions. The team did not attempt proteins containing more than four predicted β-strands.

The high cost of the SP method motivated us to create ProteinShop with the overarching goal of increasing the SP method's efficiency by means of an interactive system that could be used as a front-end for both the setup and the optimization phases. Rather than running lengthy constrained minimizations to form α-helices and β-sheets according to secondary structure predictions, ProteinShop streamlines the process of creating initial configurations. Its geometry generation module forms α-helices and β-strands instantaneously, and its manipulation module permits users to interactively align β-strands into different β-sheets. Its capabilities and usefulness stem from tools not typically found in existing systems. These tools, discussed in Section 2, allow us to rapidly create different conformations and select only the most promising ones. ProteinShop was instrumental in the participation of the SP group in CASP5. The total time to create a set of initial configurations for a given sequence of amino acids was reduced from days (for CASP4) to hours (for CASP5), allowing the SP group to substantially increase the number of targets submitted. Furthermore, ProteinShop allowed them to quickly generate a diverse population of initial configurations regardless of the size and topology of the targets. The group submitted predictions of 20 targets ranging in size from 53 to 417 amino acids and containing more than ten β-strands. This group ranked 13th or 15th (depending on the evaluation metric used) in the new fold/fold recognition category out of 154 groups evaluated (see Groups 271 and 272 at http://www.russell.embl.de/casp5).

4. Current Research

Currently, ProteinShop is being used for the optimization phase of the SP method. The control module provides interaction with the configuration and subspace selection module of the global optimization process while it is running and provides access to its internal data structures. By using this data, the control module can create a graph of the entire tree of possible configurations generated by the global optimization process thus far and make them accessible for viewing and manipulation by the ProteinShop user. The user can download the entire tree or the 3D atom coordinates of any configuration from the tree at any time during the computation. The server dialog box of ProteinShop offers several options. Through this dialog box the user can connect/disconnect to a computer running the global optimization code. The user can obtain the entire tree created thus far and visualize it as shown in Figure 8. Red nodes in the tree correspond to structures that are currently being expanded via sub-space global minimizations, blue nodes correspond to structures that have been expanded once, and green nodes correspond to structures that have been manipulated with ProteinShop and then inserted back in the tree structure for further minimization. The user can select a particular node in the tree for visualization or select the best configuration found thus far according to energy value. Finally, the user can insert the modified configuration back into the global optimization process.

Our goal is to provide users with a tool to examine configurations created during the optimization phase, so that users can make a decision about which areas of the search space are promising or which areas should be eliminated. In addition, the combination of the control module with the manipulation module gives users a powerful method to steer the optimization process towards promising regions of the search space. In fact, a user can download a configuration, then manually "improve" it with the interactive manipulator, and upload it to the optimization process for further minimization. These improvements may range from re-forming hydrogen bonds that were broken during the minimization process, to changing pieces of secondary structures, to changing entire pieces of tertiary structure. The user can add configurations to the list that were originally overlooked or remove configurations that do not make biological sense. This interplay between the physics-based global optimization code and the knowledgeable user can substantially increase the overall performance of the SP method.

Currently, the SP algorithm uses a random selection of dihedral angles for sub-space minimization, mainly because no heuristic is known that favors some angles over others. We are working on a new feature for the control module that allows us to keep track of the most successful topologies and subspaces of dihedral angles chosen so far with the goal of defining new heuristics for the selection of dihedral angles that are guided either by a history of previous selections, or by knowledge from known proteins, or both. Visually debugging the optimization process should reveal fundamental insight into the efficiency and performance of the process and enable us to formulate better heuristics.

5. Conclusions

We have described ProteinShop, a program that creates and manipulates protein structures interactively. ProteinShop provides new capabilities for manipulating protein molecules and for visualizing their properties. It has been used to reduce the time required to determine protein folds from days to hours, and has provided the ability to solve realistic-sized problems.

ProteinShop has been deployed for use by computational biologists and computers scientists at the Lawrence Berkeley National Laboratory, the University of California at Berkeley, and the University of Colorado at Boulder. These scientists competed in CASP5 with a protein structure prediction method based on stochastic global optimization. ProteinShop was used to create the initial configurations for subsequent optimizations. ProteinShop helped the CASP5 team to tackle proteins of sizes and topologies that were not possible before. Recently, we have integrated ProteinShop with the global optimization code. ProteinShop is being used as a front-end for monitoring and steering massively parallel optimizations.

In addition, we are exploring different parallel methodologies for local minimization so that ProteinShop can quickly provide the local minimum corresponding to the protein structure on display. It would be valuable to display both protein structure and (evolving) energy values - based on possibly local and/or global energy terms. We plan to develop additional meaningful energy visualization methods, which will enable a user to easily comprehend the relationship between structure and energy, and thus perform focused steering operations in the most appropriate regions of a protein.

We will also investigate ways to measure the "difference" between protein structures. The optimization process itself operates in an extremely high-dimensional parameter space, where a relatively large number of quite different parameter values may lead to relatively similar 3D protein structures - or where quite similar parameter values may lead to highly dissimilar 3D protein structures. Understanding the interplay between difference of parameter values and difference of 3D proteins is important, and for this purpose we plan to devise meaningful definitions and efficient methods for the numerical and visual analysis of structure deviation.

In addition, we plan to incorporate a feature for the fully automatic formation of complex β-sheet motifs.

Finally, while our focus in the near future will remain on improving ProteinShop for protein structure prediction applications, we will also investigate how the underlying interaction paradigms can be applied to or adapted for other computational science and engineering domains where definition and manipulation of complicated geometry are crucial.

Acknowledgements

This work was supported by the U.S. Department of Energy under contract DE-AC03-76SF00098, the National Science Foundation under contracts CHE-0205170 and ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, through the National Partnership for Advanced Computational Infrastructure (NPACI) and a large Information Technology Research (ITR) grant; and the National Institute of Health under contract P20 MH60975-06A2, funded by the National Institute of Mental Health and the National Science Foundation. We thank Prof. Head-Gordon, Prof. Schnabel, Prof. Byrd, and Elizabeth Eskow for their helpful comments.

References

1. S. Crivelli, E. Eskow, B. Bader, V. Lamberti, R. Byrd, R. Schnabel, and T. Head-Gordon, *A Physical Approach to Protein Structure Prediction*, Biophys. J. **82** (2002) 36-49.
2. R.A. Sayle and E.J. Milner-White, *RasMol: Biomolecular Graphics for All*, Trends in Bilchemical Sciences **20** (1995) 374-376.
3. E. Martz, *Protein Explorer: Easy Yet Powerful Macromolecular Visualization,* Trends in Biochemical Sciences **27** (2002) 107-109.
4. Per J. Kraulis, *MOLSCRIPT: A Program to Produce Both Detailed and Schematic Plots of Protein Structures*. Journal of Applied Crystallography **24** (1991) 946-950.
5. N. Guex and M.C. Peitsch, *SWISS-MODEL and the Swiss-PdbViewer: An environment for comparative protein modeling***.** Electrophoresis **18** (1997) 2714-2723.
6. J. Gans and D. Shalloway, *Qmol: A Program for Molecular Visualization on Windows-Based PCs*, J. of Molecular Graphics and Modeling **19** (2001) 557-559.
7. G. Schaftenaar and J.H. Noordik, *Molden: A Pre- and Post-Processing Program for Molecular and Electronic Structures*, JCAMD **14** (2000) 123-134.
8. PyMol, DeLano Scientific, San Carlos, CA, USA. http://www.pymol.org.
9. C.C. Huang, G.S. Couch, E.F. Pettersen, and T.E. Ferrin, *Chimera:*

An Extensible Molecular Modeling Application Constructed Using Standard Components. Pacific Symposium on Biocomputing **1** (1996) 724.

10. T.A. Jones and M. Kjeldgaard, *Making the First Trace with O*, Proc. From the CCP4 Study Weekend in Chester, UK, (1994).

11. W. Humphrey, A. Dalke, and Klauss Schulten, *VMD: Visual Molecular Dynamics,* J. of Molecular Graphics **14**, 33-38 (1996).

12. W.D. Cornell, P. Cieplak, I. Bayly, I.R. Gould, K.M. Merz, D.M. Ferguson, D.C. Spellmeyer, T. Fox, J.W. Caldwell, and P.A. Kollman. *A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids and Organic Molecules.* J. of the American Chemical Society **117** (1995) 5179-5197.

13. A.D. MacKerell, Jr., D. Bashford, M. Bellott, R.L. Dunbrack Jr., J.D. Evanseck, M.J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F.T.K. Lau, C. Mattos, S. Michnick, T. Ngo, D.T. Nguyen, B. Prodhom, W.E. Reiher, III, B. Roux, M. Schlenkrich, J.C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiorkiewicz-Kuczera, D. Yin, M. Karplus. *All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins*. Journal of Physical Chemistry B **102** (1998) 3586-3616.

14. M.T. Nelson, W.F. Humphrey, A. Gursoy, A. Dalke, L. Kale, R.D. Skeel, and K. Schulten. *NAMD: A Parallel Object-Oriented*

Molecular Dynamics Program. Intl. J. of Supercomputer Applications and High-Performance Computing **10(4)** (1996) 251-268.

15. J.E. Stone, J. Gullingsrud, and K. Schulten. *A System for Interactive Molecular Dynamics Simulation.* 2001 ACM Symposium on Interactive 3D Graphics, ACM SIGGRAPH (2001) 191-194.

16. A. Achari, S.P. Hale, A.J. Howard, G.M. Clore, A.M. Gronenborn, K.D. Hardman, and M. Whitlow. 1.67-*A X-Ray Structure of the B2 Immunoglobulin-Binding Domain of Streptococcal Protein G and Comparison to the NMR Structure of the B1 Domain*. Biochemistry **31** (1992) 10449.

17. L.J. McGuffin, K. Bryson, and D.T. Jones. *PSIPRED: A protein structure prediction server*. http://www.psipred.net.

18. W. Kabsch and C. Sander. *Dictionary of protein secondary structures: pattern recognition of hydrogen-bonded and geometrical features*. Biopolymers **22** (1983).

19. C. Welman. *Inverse kinematics and geometric constraints for articulated figure manipulation*, Master's Thesis, Simon Fraser University, Vancouver, Canada (1993).

20. A.L. Lehninger, D.L. Nelson, and M.M. Cox. *Principles of Biochemistry*, 2nd ed., Worth, New York, New York (1993).

21. W.H. Teukolsky, S.A. Vetterling, W.T. and B.P. Flannery. *Numerical Recipes in C*, 2nd ed. (1992), Cambridge University Press, Cambridge, MA

22. O. Kreylos, N. Max, B. Hamann, S. Crivelli, and W. Bethel. *Interactive Protein Manipulation*. Proceedings of IEEE Visualization 2003.

23. J.S. Richardson and D.C. Richardson. *Principles and Patterns of Protein Conformation.* Prediction of Protein Structure and the Principles of Protein Conformation, G.D. Fassman ed., Plenum Press, New York, New York, 1989.

Captions to Figures

Figure 1: A pre-configuration created with ProteinShop.

Figure 2: ProteinShop's IK manipulations to generate an initial configuration for 1PGX:

- **a.** Select first β-strand and activate a coil region;
- **b.** Form a β-sheet between first and second β-strands;
- **c.** Select fourth β-strand and activate a coil region;
- **d.** Form a β-sheet between third and fourth β-strands;
- **e.** With fourth β-strand selected, activate coil regions on both sides of the α-helix;
- **f.** Form a β-sheet between first and fourth β-strands;
- **g.** Select the α-helix and coils on both sides.
- **h.** Manipulate the α-helix without disturbing the β-sheet.

Figure 3: Alignment guides and hydrogen bond rendering in ProteinShop.

Figure 4: A β-sheet and the effects of twisting, pleating, curling, and braiding.

Figure 5: Atom collision rendering in ProteinShop.

Figure 6: Visualization of per-atom partial energy values.

Figure 7: Block diagram of the setup and optimization phases of the SP method.

Figure 8: A tree of configurations generated with the SP method.

Figure 1

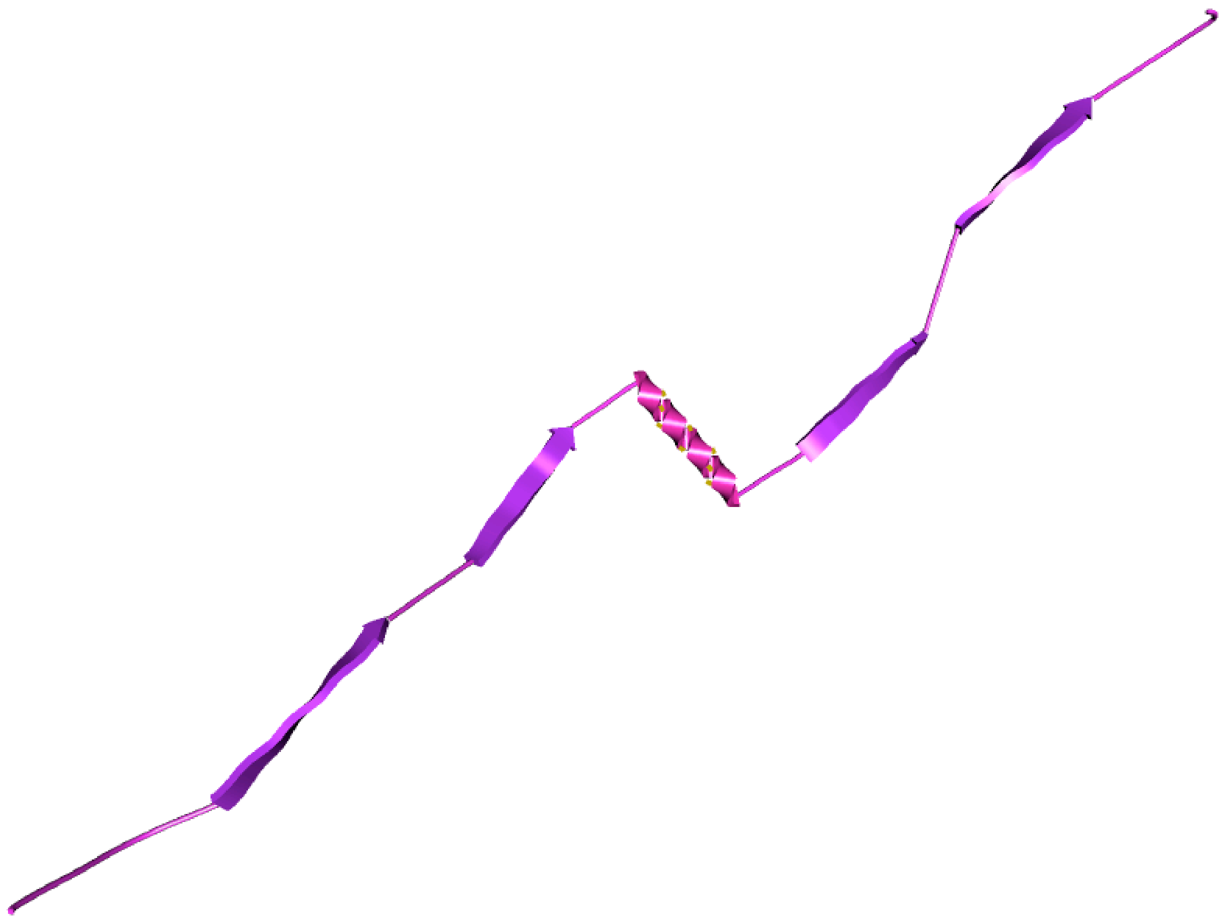
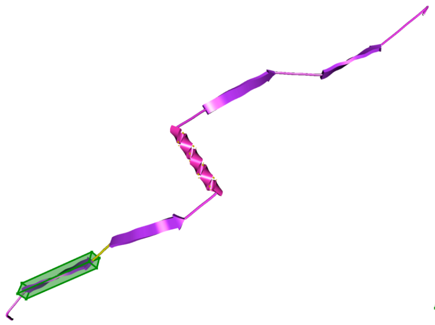
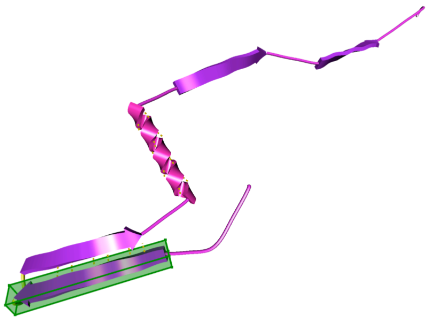
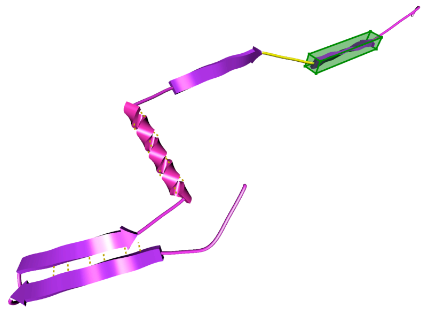
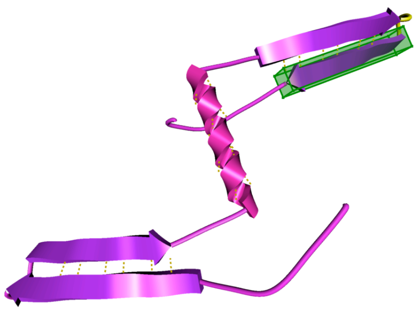


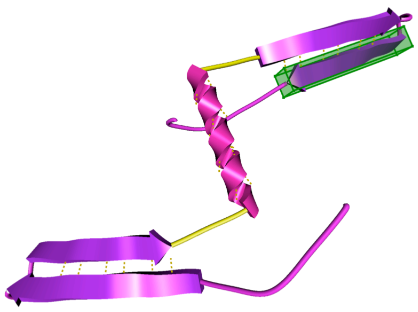
Figure 2

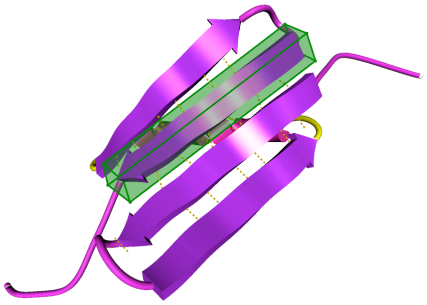

a)

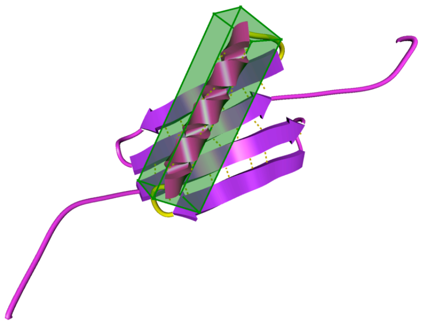

b)


c)


d)


e)


f)


g)

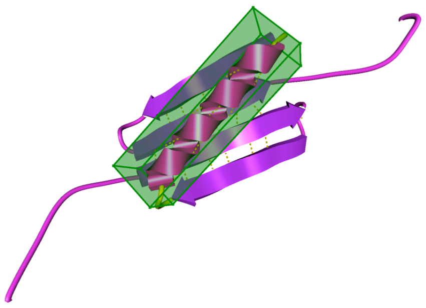

h)

Figure 3

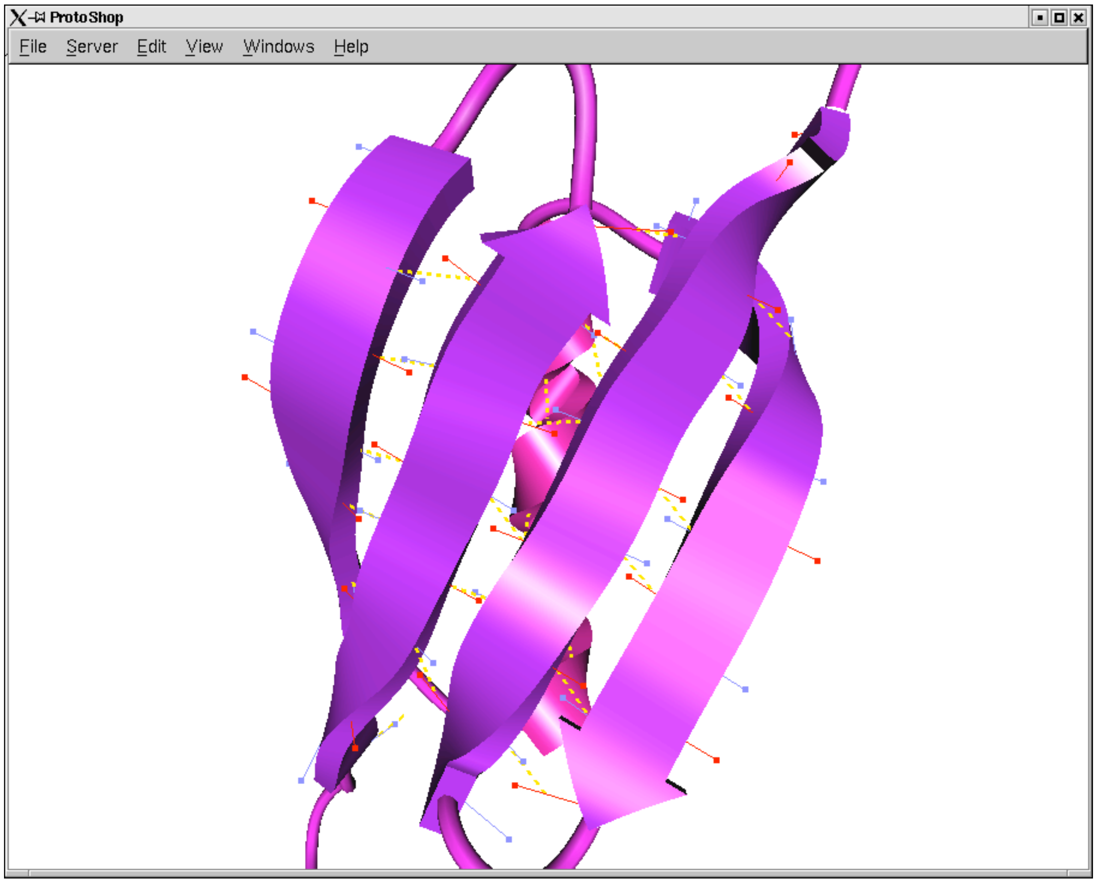


Figure 4

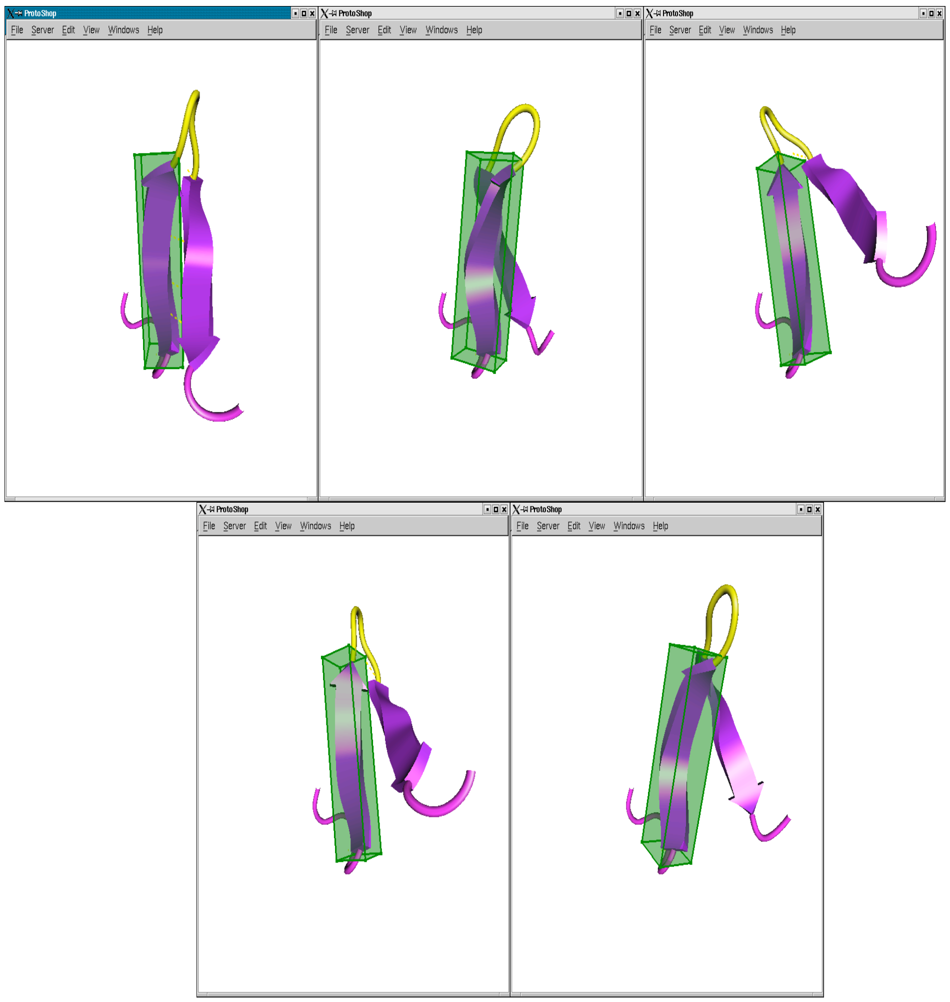


Figure 5

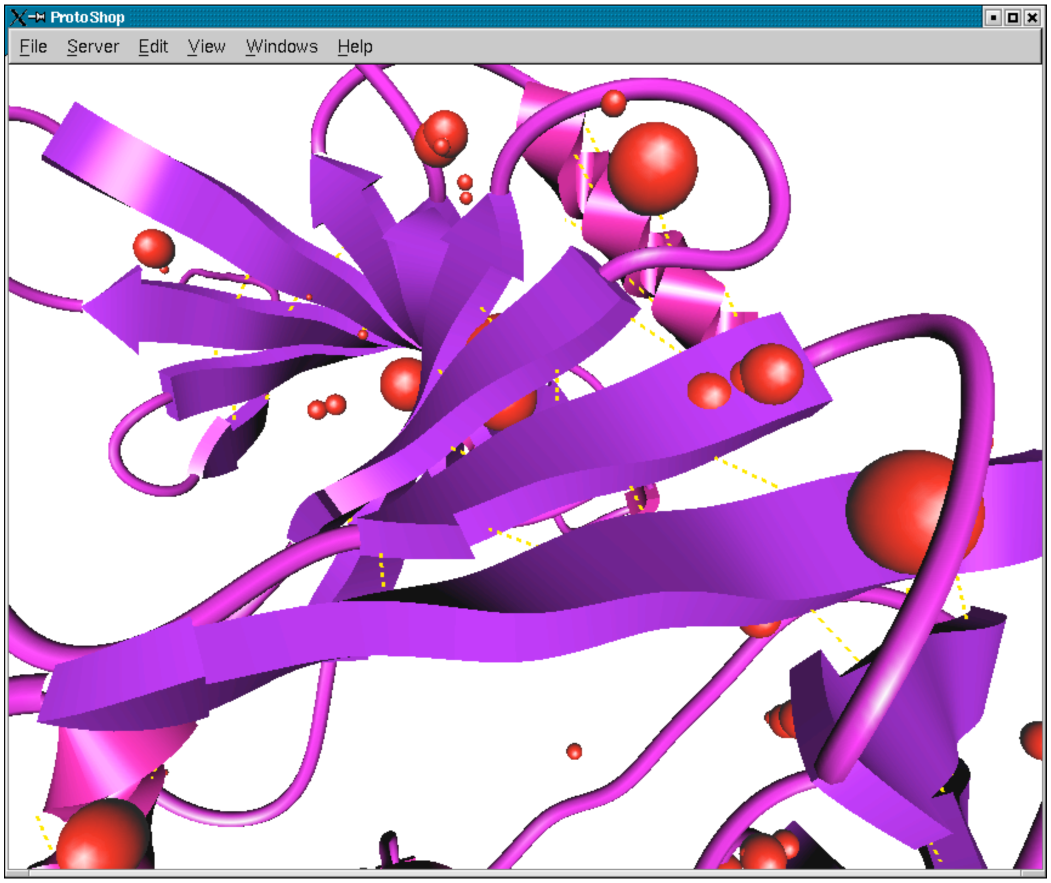


Figure 6

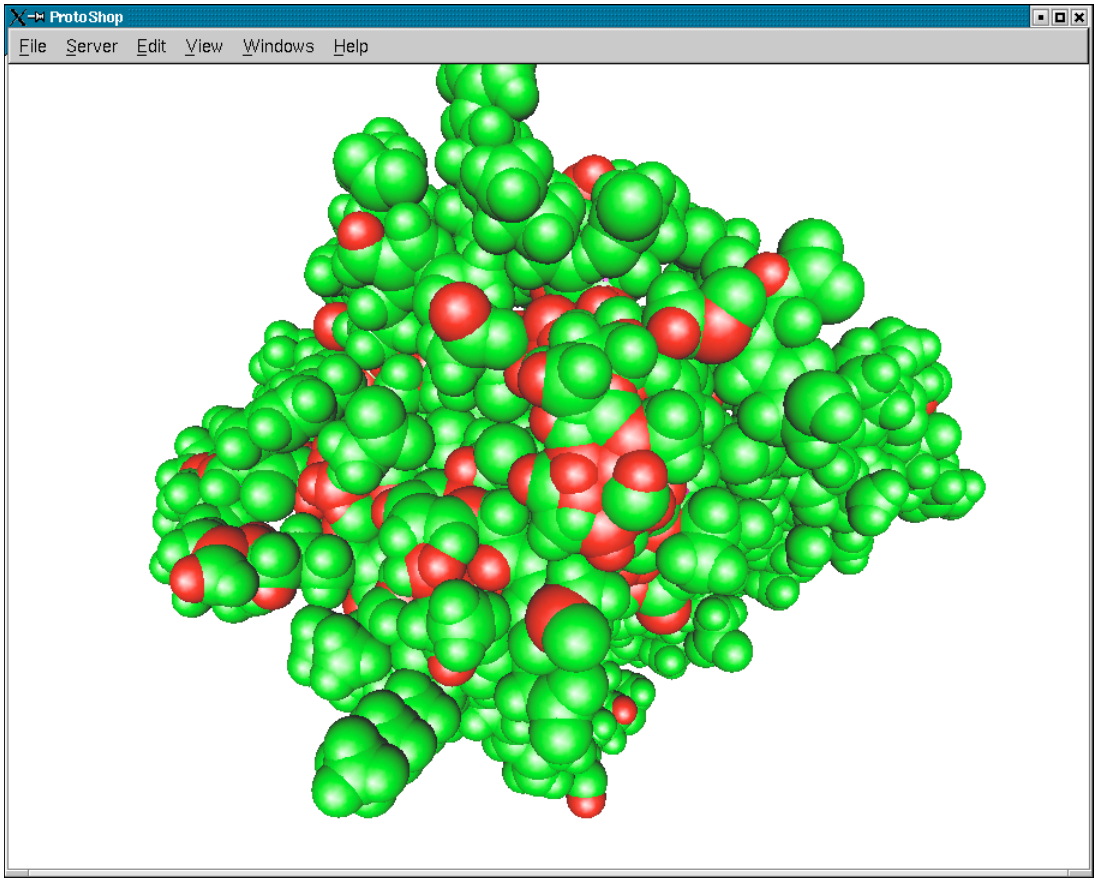


Figure 7

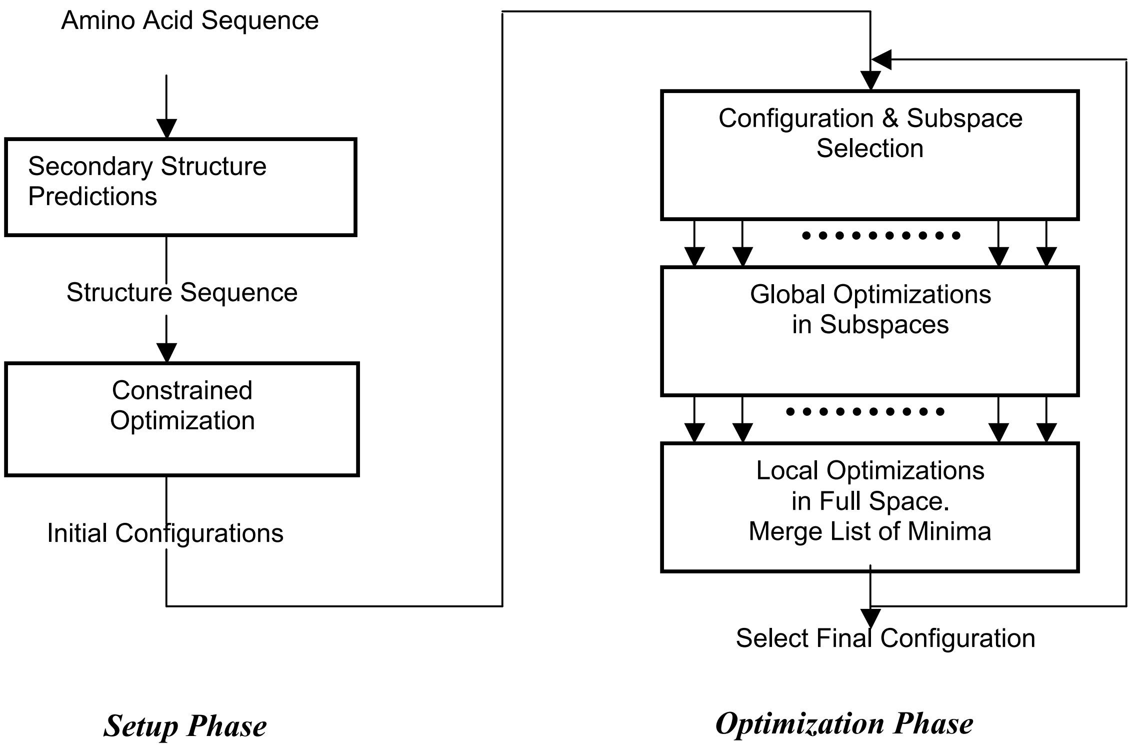
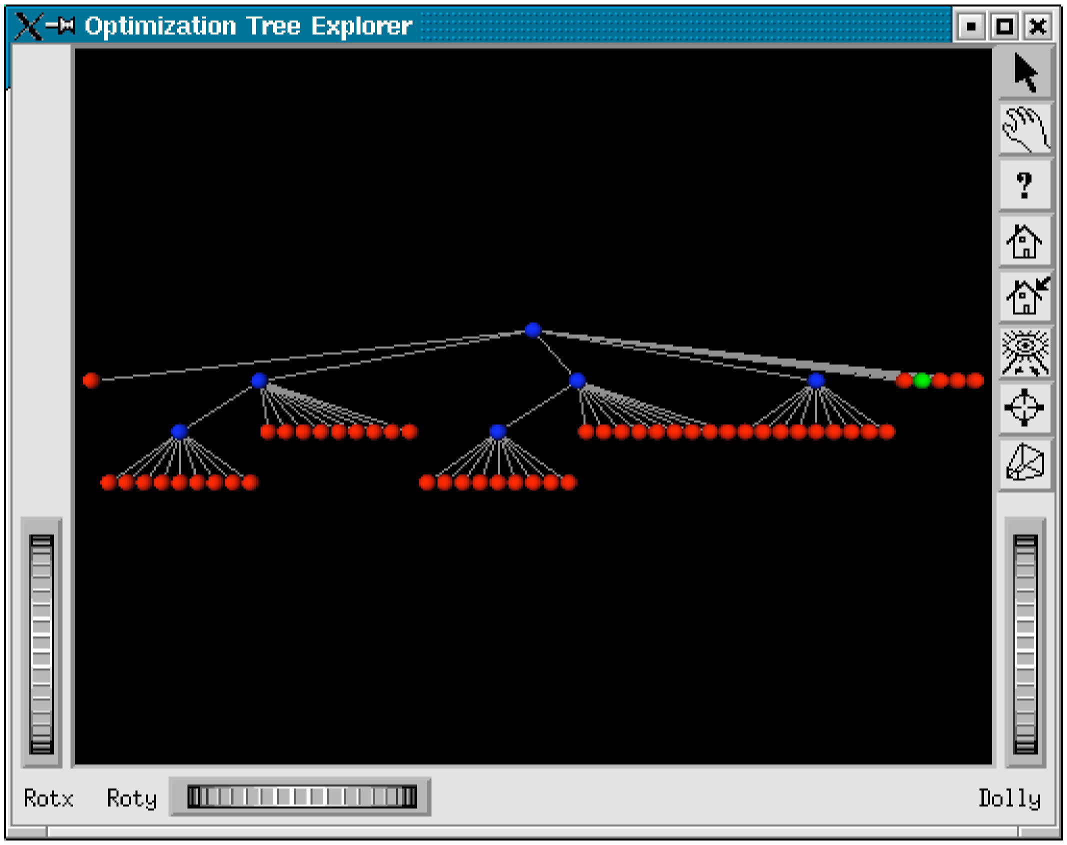

Amino Acid Sequence
Secondary Structure Predictions
Structure Sequence
Constrained Optimization
Initial Configurations
Configuration & Subspace Selection
Global Optimizations in Subspaces
Local Optimizations in Full Space. Merge List of Minima
Select Final Configuration
Setup Phase
Optimization Phase

Figure 8


Optimization Tree Explorer
Rotx
Roty
Dolly