DOCKside II: ADDING FUNCTIONALITY TO CURRENT MOLECULAR DOCKING SYSTEM

Ian Kitley
ikitley@cs.uct.ac.za

Johannes Jansen van Vuuren
jjansenv@cs.uct.ac.za

Dumisani Campbell
dcampbel@cs.uct.ac.za

Department of Computer Science
University of Cape Town

ABSTRACT

The DOCKside program makes use of Electron Microscopy (EM) maps and X-Ray Crystallography (XRC) maps to dock molecules. This can be achieved either manually or automatically. The automated functions include rigid- and flexible- docking functions. A solution space viewer was also designed to enable users to view the complete solution space that is generated by the automated docking feature of DOCKside.

Testing revealed that the rigid docking procedures of DOCKside, especially Vector Quantization (VQ) performs just as well as the leading docking software packages like Situs. It was further revealed that the visualization tool was successful in conveying seven-dimensional data in an intuitive manner using three-dimensional graphics such as glyphs and starmaps.

In pursuit of a novel flexible docking method, boundary constraints on a simple model, such as a sphere, were investigated. Successful simulations were obtained where external forces acted upon the molecular structure pushing it back into the boundary constraints.

1. INTRODUCTION

Through understanding complex molecular and chemical processes, it is possible to advance the fields of medicine and science. This is achieved by modeling the structures of these molecules in as much detail as possible, since structure defines function. This can be accomplished using techniques such as X-Ray Crystallography (XRC), but often larger molecules are impossible to view at near atomic levels, as their complexity and flexibility cause problems. Techniques such as Cryo-electron microscopy (Cryo-EM) are often then used to image these molecules, but do so at a much lower resolution, providing us with at least an idea of their shape and density.

To circumvent these problems, the practice of mapping or docking smaller, high resolution images of parts of these molecules into the low resolution electron density maps has been developed [11]. This provides the detailed structural information needed to analyze the molecules. Done often manually, but also with the help of automatic computational algorithms, these dockings can be produced quickly and easily, providing researchers with the information they need to perform breakthroughs.

Numerous programs, such as Situs [10], that help researchers to perform docking are presently available and in use. In 2005, the University of Cape Town, in association with the National Bioinformatics Network, began development of their own implementation of a docking program. Known as DOCKside [7], it provided the user with an interactive, graphical interface that allowed the use and implementation of a number of automatic docking algorithms, as well as the option to perform the docking manually.

1.1. Comparison and Testing

As the latest addition to the ranks of computational docking systems, it is essential that DOCKside be compared against these systems to determine how usable and efficient it is. This type of comparison would also provide avenues of exploration for further advancement in later versions. However, for this to occur, it is also vital that DOCKside be properly tested to determine its accuracy in predicting dockings. Without this it will not be possible to determine its applicability within the wider community.

1.2. Solution space

The solution set generated by DOCKside is a seven-dimensional solution set. It contains a three-dimensional matrix representing the x, y and z translations that must be applied to the molecular structure (described in a PDB file) to center it on the electron Micrograph (EM) map. For each of these translations, the solution set contains multiple x, y and z rotations that were applied to the PDB model as well as the fitness correlation of the translation and rotation that serves as an indication of how well the two models overlap at that specific solution.

The concept of this solution space can be modelled in the real world by using an analogy of a store room. The store room contains a large amount of boxes and each of these boxes contains multiple items. When a specific item needs to be retrieved, the specific box that contains the item must be isolated, and then each of the items in the box has to be analyzed to see if it fits the description of the item that needs to be retrieved. In the same way, the ‘outer solution’ subset of the solution space serves as boxes that store different ‘inner solutions’ and a value that indicates how well that ‘inner solution’ fits the criteria of the program. In the case of DOCKside, the criterion is a perfect fit of the PDB file onto the EM map.

1.3. Visualization

The term visualization refers to the task of displaying sets of data in a visually pleasing manner. When dealing with information that consists of less than three variables, i.e. is less than three dimensional, these visualizations are fairly straight forward. Methods like graphs, plots, histograms and tables provide adequate methods to visualize data in these formats.

However, when one has to visualize data that has a complexity larger than three dimensions, one has to manipulate the data so that it can be displayed in a format that is either two- or three-dimensional. Finding appropriate ways of visualizing multidimensional data becomes complex since even if one finds a way of breaking the information up into some smaller sets of data, the representation of the data in 2D or 3D must still be appropriate for the specific solution.

Various methods have been adopted to provide visualization tools for viewing multidimensional information. The tools that have been developed can be split up into two main trends. In the early years of computer science, most of the visualization techniques were two dimensional but as technology advanced and the hardware became more powerful, it became more feasible to explore three dimensional visualization techniques.

The DOCKside project requires an investigation into visualization to create a visual representation of the seven-dimensional solution sets that are generated.

1.4. Structure

To this end, the remaining body of the report delves into previous work within the fields mentioned above in chapter 2, before moving on to the design of each section in chapters 3 and 4. Chapter 5 examines how some of these designs were implemented, while chapter 6 examines the results of user and algorithmic testing. Finally, conclusions are drawn in chapter 7 and future work examined in chapter 8.

2. BACKGROUND

Multi resolution molecular modeling has emerged as a powerful strategy to bridge the resolution gap between crystallographic high resolution structures and EM maps. As noted already, the atomic structure of the entire assembly is constructed by docking high resolution data obtained from XRC into the low resolution EM density maps (EDM). The two most commonly used techniques for building these pseudo atomic resolution structures are rigid and flexible docking. Rigid docking offers much simpler methods to achieve this. However, in many cases, these methods do not apply, as the molecular structure may undergo conformational rearrangements upon forming the assembly. Flexible docking methods, on the other hand, allow for conformational rearrangements upon forming the assembly. While both these methods have proven to be useful, there still exits a need to create more innovative docking solutions.

2.1. Testing

At present, there is no formalized procedure for testing docking algorithms with the scientific community, but investigation shows that the testing that does take place falls into two main categories. The first, which will be referred to as the experimental approach, makes use of EDM generated through experimental techniques. Often containing noise, these maps provide a real-world challenge for the systems, testing their ability to operate under the conditions they were designed for.

The second technique, which will be referred to as the simulation technique, generates EDMs from atomic structures with the help of various algorithms and filters. This technique is instrumental in the testing of newer algorithms, as it is used to test the accuracy of the algorithms. This comes about due to our knowledge that we definitely have the correct structure and know how it should dock into our simulated EDM.

2.2. DOCKside [7]

Produced in 2005 by students at the University of Cape Town, this system provides a graphical interface for the docking of EDMs and atomic structures. The system concentrates on implementing a user-friendly interface and manual and automatic docking algorithms. The system is geared towards a component architecture, allowing new modules and docking algorithms to be added and removed as needed. At present, three algorithms are implemented, namely global and local correlation, as well as contour-based fitting.

2.3. Situs [10]

Developed by members of the research group BioMachina at the University of Texas, this system is considered to be one of the pioneering programs in the area of computerized docking tools. Providing a range of docking tools and algorithms within a text based implementation, the system can handle multiple file formats and allows the user much leeway in how the tests are structured. At present, it implements rigid contour-based fitting and vector quantization (VQ) algorithms, as well as a flexible implementation of the VQ algorithm.

2.4. Vector Quantization [12]

Previously used as a lossy compression technique in image processing, VQ has been extended to be used in docking algorithms over the last decade. Conventionally, this technique uses a set of code-book vectors to describe the topology of an object. These vectors are initialized randomly to begin with and are then, over the course of a number of iterations, refined until they produce the required topology.

The conventional technique for accomplishing this clusters the points or vectors that initially describe the object around the code vectors closest to them. These clusters are referred to as Voronoi cells. These code vectors are calculated by taking the average of the points surrounding them. However, a more recent for calculating these vectors has been developed and referred to as a topology representation network (TRN).

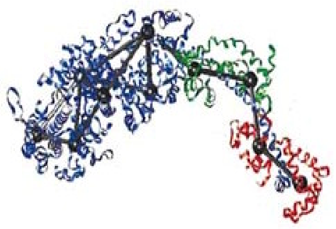


Figure 1:Vector quantization of a molecule

TRNs use a Hebbian rule, which alters the code-book vectors by a certain amount proportional to the distance between the code-book and a chosen input vector, applying equations that affected by the number of iterations that have passed and the ranking of the code-book vector in comparison to the rest of the vectors.

However, when used in a docking setting, the VQ algorithm is in fact a method for producing vector sets for both the EDMs and atomic structures so that they can be matched. These sets must then be matched, resulting in a combinatorial problem. Due to this, the algorithm does not work very well with vector sets greater than 9, but below this produces matches in real time.

2.5. Visualization

Visualization techniques are currently split into two distinct categories: two- and three- dimensional techniques. This background section explores the different techniques used to visualize multi-dimensional data under each of these categories.

2.5.1. Two dimensional visualization techniques

The two-dimensional visualization techniques map information onto two dimensions and display the information on two independent axes.

Assa et al. [1] describes one method of achieving two-dimensional visualization of high dimensional data sets. In [1], each data item is evaluated according to different relations and a correlation factor is assigned to indicate how well each data item and relation match up. Assa et al. [1] proposes a method of displaying the multidimensional data using relevance maps. The relations are converted into gravitation nodes. A gravitation node is mapped onto a two-dimensional plane and serves as an attractor to the data. The data items are represented by points placed between the nodes, using the values associated with each node as an attraction value for that node. The points are dispersed in a similar fashion to that used in Venn diagrams. A point will be placed close to a node if it has a high relevance to that node, and between two nodes if it has relevance to both; the relative distance indicating the relevance to each node. It will be placed far from both if it does not have relevance to either node.

However, when the mapping takes place, there are various ambiguities that can be introduced.

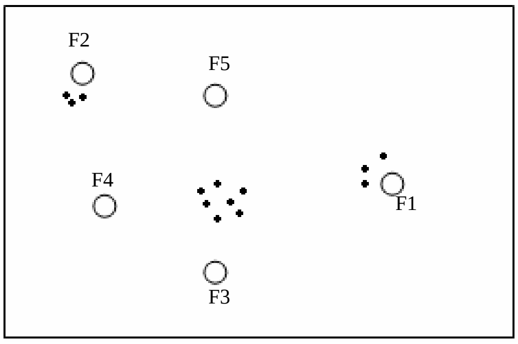


Figure 2: Representation of the relevance map.

Figure 2 represents a naïve solution using relevance maps to visualize the data sets. Ambiguities occur, because when one looks at Figure 1, there is no way that one can tell whether the data found in the middle of the figure represents data that is relevant to all of the relations (F1 – F5) or just relevant to nodes that are diagonal to each other.

The paper proposes a solution for these ambiguities. The proposal is to create new gravitation nodes that would represent multiple relations. The composed nodes are created by searching through the data and finding sets of relations where a certain amount of data items have high relevance to all of the nodes. The new node is then placed onto the map in the relevant position. This method enables the users to get an overview of the information at a glance. They are able to see the distribution of the information between the different nodes and also to see which data items are highly relevant to all the relations.

There are various constraints that one must be aware of when considering using a similar approach. This visualization technique is only applicable if the relations are independent. The method will also not work if the relations are not static. In other words, if the user changes the values of some of the variables, the map would have to be completely regenerated.

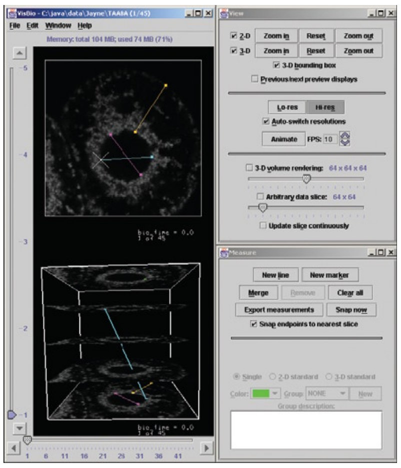
Another method of visualizing information that is constructed using relations is known as “Parallel” Coordinates, discussed by Inselberg [4]. Using “Parallel” Coordinates, the user is able to visualize data of any complexity (any number of relations or variables can be visualized.). This method is also able to give the user some hints as to whether the data follows certain statistical distributions, such as the Normal distribution or Gaussian distributions.

The parallel coordinate system is used to construct relations, as follows. The points in the data sets are mapped to lines between variables. These lines represent the relations between two different variables. This enables the user to make conclusions based on visual cues that they ascertain from the parallel coordinate representation. The user is also empowered to perform simple and complex queries on the visualized data, which will eliminate and highlight certain data items.

The parallel coordinate system method is a powerful tool when one wishes to do data mining but, as with [1], the technique is most useful when working with relations and not with dependant variables. However, Spears pointed out that rotations in Cartesian space can be mapped to translations using parallel coordinates [8].

VisBio is a software package that was released to enable the viewing of multidimensional solution spaces by using a slice viewer [5]. In a slice viewer, each slice represents a single step in the multidimensional data. For example, when a slice viewer is applied to the six dimensional solution space that is generated by DOCKside, each slice represents a single translation. The resulting image is a two-dimensional display of all the rotations that occur at that specific translation. Slice viewers indicate fitness using colour e.g. brighter colours indicating fitter solutions.

Although this method would be an appropriate application for a solution viewer in DOCKside and something similar was implemented in the previous version, this representation is somewhat limited. The slice viewer requires the user to scroll through each slice and only displays the information relevant to that specific slice.



File Edit Window Help
2-D Zoom in Reset Zoom out
3-D Zoom in Reset Zoom out
3-D bounding box
Previous/next preview displays
Lo-res Hi-res
Auto-switch resolutions
Animate FPS: 10
3-D volume rendering: 64 x 64 x 64
Arbitrary data slice: 64 x 64
Update slice continuously
New line New marker
Merge Remove Clear all
Export measurements Snap now
Snap endpoints to nearest slice
Single 2-D standard 3-D standard
Color: Group: NONE New
Group description

Figure 3: Example of a slice viewer in VisBio [5]

The viewer therefore does not supply a way for the user to view the entire solution space at once. Another drawback of this visualization for multidimensional data is that it does not reflect the changes that the information undergoes as a particular variable changes, in an intuitive fashion. In other words, each slice exist separately and thus, one has to switch between slices to see the changes that occur between the two, thus making it complicated for the user to compare two different solutions.

For this specific problem, one needs a solution whereby the user is able to manipulate certain variables and see the effect it has on other related variables. Therefore, an exploration into three-dimensional visualization is necessary.

2.5.2. *Three-dimensional visualization*

Three-dimensional visualization empowers the user to view multi-dimensional data in three dimensions instead of two. This review will look at some of the methods that have been suggested to make use of all three the physical dimensions to create visualization solutions.

Feiner et al. produced one such solution in their paper [3]. They propose a system whereby one slices away the higher dimensions (dimensions higher than three) and then adds them back in a controlled fashion. These dimensions are placed back into the system as three-dimensional subsets of the original three dimensions that were kept after discarding the higher dimensions. Each of these 3D sets are represented as height maps where higher values at those coordinates represent higher areas in the height map. The metaphor given to this method is one of *worlds within worlds.* This method of 3D visualization enables the user to view different inner dimensions of the data since they are related to the initial three dimensions.

Dos Santos et al. proposes another method of 3D visualization known as the *HyperCell* method [2]. The *HyperCell* concept enables the breakdown of the n-dimensions by enabling the user to specify which dimensions should serve as the axes to the cells that are generated. The concept is able to generate visualizations in 2D and 3D. If the user selects a single variable, that variable will become the x-axis of the graph and the y-axis will be assigned values where the other variables are represented as a function of the first variable. When the user selects two variables to serve as the axes, the rest of the variables are displayed as contour maps of those two functions and finally if the user selects three different variables to represent the axes, the rest of the variables will be converted into a 3D function of the three variables and be displayed in such a fashion.

Stump et al. discusses a third way of constructing three-dimensional representations for multidimensional data structures using glyphs [9]. The glyph structure is tied in with Parallel Coordinates (see above). The glyph system proves to be more applicable to systems where the variables are independent but can also be applicable to dependant variable sets.

Glyphs are points in either two-dimensional or three-dimensional space that represent data items. The different “tails” that extend from that point represent different relations or variables. Different characteristics of these tails represent certain information about the specific variable or relation. For instance, the length of a line will represent its magnitude and the in some instances the direction of the line might represent the direction in which a variable is pointing to.

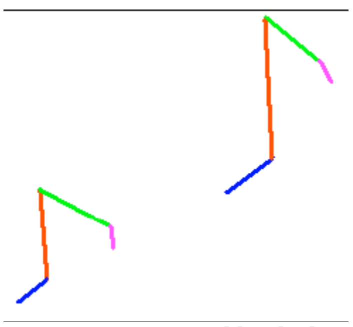


Figure 4: Customizable Glyphs

2.6. Advances in Flexible Docking Techniques

Recent investigations have shown that molecular modeling (MM) can be used to create innovative solutions to the problem of flexible body fitting. Of particular interest, the use of molecular dynamics (MD) has been proposed due to its ability to model large system. Furthermore, MD simulations are able to model the dynamics of the system and the method allows for conformational rearrangements upon forming the assesmbly. In this way, EDMs can be seen as acting as a restraining potential on MD simulations.

2.6.1. *MD Algorithm*

In MD, sets atomic positions are derived in sequence by applying Newton’s equation of motion *(1)* and *(2)*. The equations are integrated, using different integration algorithm [9], by breaking the calculation into a series of very short time steps (typically between 1 femtosecond and 10 femtoseconds; 10^{-15} s to 10^{-14} s). At each step, the force on the atoms are computed and combined with the current positions and velocities to generate new positions and velocities a short time ahead. The atoms are then moved to the new positions, an updated set of forces is computed, and so on. In this way, a MD simulation generates trajectories that describe how the dynamics variables change with time.

$$d^2r_i(t)/dt^2 = m_i^{-1}F_i \quad (1)$$

$$F_i = -\partial V(r_i, \ldots r_N)/\partial r_i \; (2)$$

The force on atom *i* is denoted by F_i and *t* denotes the time. The gradient of the potential energy *U(r)* are the forces, and therefore *U(r)* must be a differentiable function of the atomic coordinates r_i. The integration of equation *(2)* is performed in small time steps mentioned above.

2.6.2. Force Field Functions

Force fields contain the necessary building blocks for the calculation of energy and forces. Essentially, force fields refer to the parameterization of a system, in particular, the systems atoms. That is, parameter sets are used to describe the potential energy of a system. The parameterized energy functions, or the defined force field for each type of atom, are basically idealized bond and angle geometry. The better the parameterized force fields are, the more accurately the properties of the molecular system can be reproduced. As potential functions are empirical, the individual parameters for one force field are not generally transferable to other force fields.

2.6.3. Software Tools for MD

To perform MD simulation we require appropriate computer programs. Also, due to size of molecules, efficient computation is necessary. The software used by molecular modelers ranges from simple programs that perform just a single task to highly complex packages that integrate many different methods. In order to conduct MD simulations, various computer programs were originally developed; these include CHARMM [12] and X-PLOR [13]. These programs are able to give us a better understanding of the system mainly due to the fact that they allow us to examine the molecule as a 3D model. Although proven to be very useful, they were however, developed to run on serial machines.

In the case of simulation of large systems, enormous computing power is required. One way to achieve such simulations is to utilize parallel computers. In recent years, distributed memory parallel computers have been offering cost-effective computational power. NAMD [11] is one of recent software packages build to run MD simulations efficiently and to utilize parallel machines.

2.7. Discussion

The aim of the project is to test and update DOCKside to the point where it can be provide an adequate facility to the larger scientific community. This requires that the system be tested and adapted. We have chosen to use Situs as a template for our update, as well as an adequate benchmark system to compare DOCKside to. In order to do this DOCKside needs to be able to simulate EDMs for testing and implement an advanced docking algorithm such as VQ.

The research explored a wide variety of visualization methods that might be applied to the solution space that is generated by the DOCKside system.

None of the visualization techniques discussed enables the user to view all of the solution space at once, but it would seem that the three-dimensional techniques can be expanded with more ease to support some functionality to visualize the change in fitness that the data undergoes as any of the data sets are manipulated. It was therefore decided to undertake the development of a novel approach that makes use of some of the ideas discussed above. The approach will make use of glyphs and star-maps to visualize the information.

In terms of the MD simulations, before any utilization of EDM as a restraints potential, thorough testing of MD simulations on a simplified model needs to be conducted. This simplified model was chosen as a sphere.

3. RIGID DOCKING AND TESTING DESIGN

3.1. File Type

Presently, DOCKside is only capable of reading SPIDER and PDB file and writes to PDB file. An EDM file format similar to ASCII was developed for DOCKside in order to write EDMs to file. Referred to as ISO, it was not fully implemented, but provides a easy to use format and therefore was chosen for storage of the simulated EDMs.

3.2. Testing

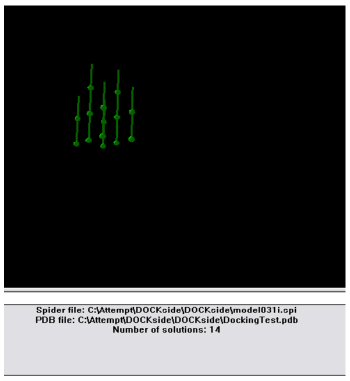
In order to fully test DOCKside, it was felt that both the experimental and simulation techniques should be applied to provide as much data as possible. These tests would make use of data provided to us by the Electron Microscopy Lab and five atomic structures used in previous tests by Kovacs et al. on Situs algorithms.

3.3. Evaluation

Once the testing results have been received, it will be required that they be evaluated and compared. In order to achieve this, criteria needed to be identified as critical to an algorithms performance. As such, we identified accuracy, which encompassed the ability of the algorithm to produce a correct or near correct solution and supply the same solutions every time, and time taken, which deals with the amount of time an algorithm takes to compute a solution set. The first of these was felt to be integral within all scientific fields, while the second was chosen to represent the need of a researcher for solutions to be supplied in the least amount of time.

4. VISUALIZATION DESIGN

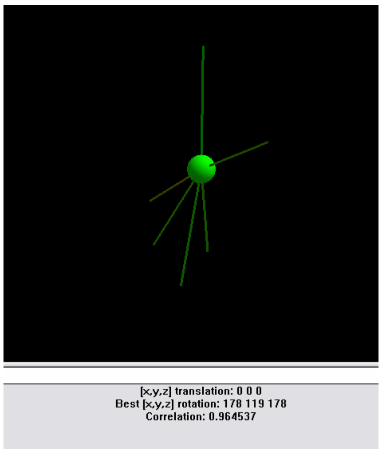
A solution viewer was written using the Microsoft Foundation Classes libraries (MFC) and C++. The program supports its own file format and is able to render three-dimensional glyph sets using OpenGL.



```
Spider file: C:\Attempt\DOCKside\DOCKside\model031i.spi
PDB file: C:\Attempt\DOCKside\DOCKside\DockingTest.pdb
Number of solutions: 14
```

Figure 5: Program view of outer solutions

Figure 5 above shows the design of the initial view that is supported by the program. The program displays the whole outer solution set (consisting of translations) and vectors extend from each solution, indicating the best inner solution for each of the outer solutions. It also contains an information window below the three-dimensional visualization that is used to convey the information in text form. The window displays the two files that were used to generate the solution set and the number of outer solutions that are present in the solution set.



```
[x,y,z] translation: 0 0 0
Best [x,y,z] rotation: 178 119 178
Correlation: 0.964537
```

Figure 6: A star map of a single outer solution with multiple inner solutions

The user is able to select any one of the solutions by double clicking on the solution. The user is then presented with the view that can be seen in Figure 6. This view is known as a star map and is an extended form of the Glyph notation. The user is able to view all the inner solutions that are associated with the specific outer solution that was selected. Each vector indicates a solution, the colour of the vector indicates the fitness correlation and the length also serves as an indication of the fitness correlation. The direction of the vector serves as an indication of the orientation of the solution.

Initially vectors were drawn from the solution to the specific point and scaled according to the fitness correlation but this posed various problems. The vectors that indicated the same inner solution would not be pointing in the same direction for different solutions, thus it might serve only to confuse the user. Solutions that contain the same values for both rotations and translations can also not be displayed since the point the vector should point to, is actually the center of the sphere that represents the outer solution.

To overcome such problems as mentioned above, the up vector [0, 1, 0] was chosen to represent no rotations [0, 0, 0]. This would mean that the vector would point directly up if no rotations have been applied. This seems to be a good visualization approach since it serves a good real world analogy where one could view the model as 'standing' in its original position.

4.1. File format

The file format was designed to be as compact as possible, see Figure 7. It requires the program to output the spider and PDB files used to generate the solution set in the first two lines of the file. The rest of the file is used to store the solution space. The inner solutions are listed first, followed by the outer solutions. Each line starts with either a 't' or 'r' that serves to indicate whether the entry represents a translation or rotation. The following three values indicate the x, y and z values for that entry.

The outer solutions contain extra pairs of entries after the initial x, y and z values. The first value in the pair indicates the reference to the position of the inner solution, i.e. if the reference should point to the first inner solution specified in the file, the reference would be 0. The second value in the pair indicates the fitness correlation associated with that inner solution.

Thus an example file format would look something like:

```
<spider file>
<pdb file>
R 0 0 0
R 10 50 20
T 3 2 6 0 0.9 1 0.5
```

Figure 7: File format for solution viewer

From the above example one can conclude that most of the information that is required to visualize the solution space. The first thing one can conclude is that the rotations are considered as inner solutions, therefore one should initially use a translation biased visualization approach. One can also conclude that there are two rotations and one translation in the example. The translation contains both rotations and the fitness correlation values associated with each of the rotations.

5. IMPLEMENTATION

5.1. Simulation

In order to simulate EDMs, a trilinear mass-interpolation algorithm, similar to that used in the three already implemented docking algorithms, must be used to project the atomic structure onto a regular grid. However, this method does not take into account decaying densities around an atoms center, which greatly influence the structure of the EDM.

This rate of decay can be simulated using a Gaussian filter, which averages the density values around this center. This provides the system with much more information that can be used during the docking process, but more can be added by further using a Laplacian filter to emphasize the edges of the molecules. This should reduce the chances of false positives by reducing the likelihood of high density areas promoting high correlations regardless of the corresponding data.

However, since we are not certain as to the actual performance of any of these approaches, we determined it would be best to initially test each of the three techniques (no filter, Gaussian filtered, and Gaussian and Laplacian filtered) to determine their performance and choose the best for subsequent use in testing.

5.2. Vector Quantization

Both VQ techniques that can be used in rigid docking make use of an unsupervised neural network. The training data used in these types of networks do not have a solution result to compare the network derived result against in order to adjust the node weightings. The type of network used in both VQ algorithms makes use of a Hebbian rule that adjusts the weightings dependent on a competitive algorithm that selects vectors based on the distance between them and the input vector.

In the case of the conventional algorithm, this rule assigns each input vector in the training file to a code vector, computing a new set of code vector once all vectors have been allocated. This then continues until the newly produced set does not differ from that produced in the previous iteration.

With regards to the TRN algorithm, the rule adjusts each code vector by a certain percentage of the difference between the given code vector and the input vector. This percentage is dependent on both the amount of time before the algorithm will end and the distance between the input and code vector in comparison to the same distance between the other code vectors and the input vector.

5.3. Vector Rotation

Once the set of code vectors has been computed for both the EDM and the atomic structure, one of these needs to be rotated and translated such that it can be docked to the other. In order to accomplish this, the angles between the two need to be calculated. This can be accomplished by calculating the average planes on which each of the EDM and atomic structure lies on and using this to calculate the angles between the planes.

In order to accomplish this, the normals for each plane constructed be the code vector is calculated using the cross product and an average of these normals is constructed. This then can be used within the conventional angle calculation algorithm to produce the rotations needed. The translation effect is computed similarly by finding the center of both objects and calculating the distance components between the two

5.4. Boundary Constraints

Boundary constraints were implemented by utilizing the tcl scripting interface for applying forces [14]. The created script essentially pushes all the atoms that reside outside the boundary of the sphere back into the boundary. To achieve this, the algorithm defines sphere constraints, i.e. the sphere center and radius. In the calcforces{} procedure of the tcl script, the algorithm goes through every atom and checks whether or not the calculated radius for that particular atom exceeds the given constraints. If that is the case, a force of 7.2 kcal/mol·$Å^2$ is applied to the atoms and thus the atom is “pushed” back into the boundary.

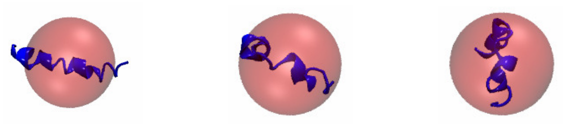


Figure 8: Force Application using Tcl Scripts

6. FINDINGS

6.1. Simulation

Tests showed that while the time taken to compute each algorithm using the different types of simulated data differed in no significant way, the accuracy of these results did. It could be seen that though no filtering produced a correct result for all the original algorithms, it only provided one or two results. On the other end of the spectrum, simulating with both Gaussian and Laplacian filters resulted in numerous varying and often incorrect results for both local and global docking, and a few results that produced at least one correct solution for contour-based fitting.

In the middle, Gaussian filtered EDMs produced a small number, usually no more than five, solutions that were correct in rotation and similar in translation across all algorithms. The differences can be attributed to too little and too much information being present in some of the simulations, resulting in both under and over fitting.

6.2. Vector Quantization

It became quickly apparent that neither of the implemented algorithms presented produced consistent solutions. It was also apparent early on that these algorithms did not always produce a correct solution, often taking two or three runs to do so. This is most likely due to the randomization of initial code vector selection, resulting in final code vector sets differing wildly in their results. It was further felt that the algorithm used in calculating the rotations was not sufficient for our uses, producing only a partially correct matrix due to the varied code vector sets.

6.3. Algorithm Testing

The results produced by our testing provided us with a number of insights regarding the efficiency of the algorithms. It was found that global was between 1.5 and 2 times quicker than contour-based fitting, with contour-based fitting producing a smaller, slightly more accurate set of solutions than global. These differences can both be attributed to the use of Gaussian and Laplacian filters in the contour-based fitting, which reduces the number of false positives, while increasing the amount of time taken to compute.

On the other side of the equation, local correlation takes anywhere from 20 to 1700 times longer to compute than either of the other two algorithms. This is largely due firstly to its inability to use Fast Fourier Transforms for its computations and secondly to the differing sizes of the footprints used during computation of the algorithm. However, the accuracy of the solutions are more in line with those produced by contour-based fitting than that of global correlation, but this does not reduce the significance of an algorithm that takes hours to run, rather than the minutes that the others take.

6.4. Comparison

In comparing DOCKside to Situs, three important differences were observed. The first of these is that Situs provides the user with either the choice of which results to save (VQ) or automatically saves only the first x results (contour-based fitting). This was in contrast to DOCKside, where the user can observe all results with accuracy above a certain threshold and then save them as they wish.

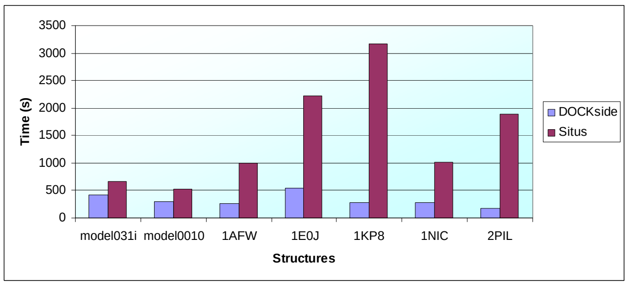


Figure 9: Map of times taken by Situs and DOCKside when computing the contour-based fitting algorithm

The second is that the time taken by DOCKside to compute the algorithms, with regards to those that they share, is much faster than that taken by Situs. With regards to contour-based fitting, DOCKside is anywhere from 1.5 to 15 times faster. This difference extends to VQ, and is most likely due to the way the algorithms were implemented more than anything else. It was also observed that the ratio between times stayed relatively similar when viewing experimental data, while that of simulated data was variable. This can be explained by DOCKside's attempts to maintain a relatively small and constant dimension size for their simulated data, whereas Situs produces EDMs of dimensions that best suited the structure being simulated.

Finally, Situs' implementation of VQ provides accurate and recurring results. When investigated, this was found to be due to three differences with the DOCKside implementation. The first involves the use of both VQ techniques, with TRN being used to construct eight code vector sets which were then fed into the conventional technique to produce a solution set. The second is due to the use of the Kabsch algorithm, a means of calculating the angle between objects, which greatly improves on that used in our implementation. Lastly, Situs uses a constant seed in calculating random numbers, resulting in the same solution.

Learning from this, we implemented the first difference, which resulted in our algorithm producing solutions after a maximum of two runs. However, we have yet to implement the Kabsch algorithm, as we are still investigating it, and we felt that the constant seed has the possibility of being unable to produce a correct solution when one does exist.

6.5. Methodology

Two different methodologies were used during the user testing of the system. While the users were interacting with the system, the researcher was observing their interaction and noting problems that they encountered. After the study, the users were asked to complete a questionnaire as a reflection of their experience of the system.

Naturalistic Observation – Naturalistic observation refers to the observation of people interacting in their natural environment without interfering. During this study, the researcher observed the interaction between the user and the program while being as inconspicuous as possible.

Questionnaire – A questionnaire was used to get feedback from the users on their experience of the system. The questionnaire contained questions that focused on the usability of the program and the effectiveness of the visualization technique. The users were supplied with a statement, as can be seen in Table 1, and were required to rate to which extend they agreed with the statement.

Since these two methodologies were adopted, it was clear from the beginning that the results would contain qualitative data only. Qualitative data refers to data that cannot be used for statistical calculations; rather, the data provides subjective information that would differ from user to user.

6.6. Summary of MD Simulations

Each structure was first minimized for 1000 steps. Thereafter, equilibration was carried out at constant temperature for 125000 time steps (250 ps). Tcl interface for applying forces was utilized to exert a force of on the atoms that were outside the boundary of the sphere.

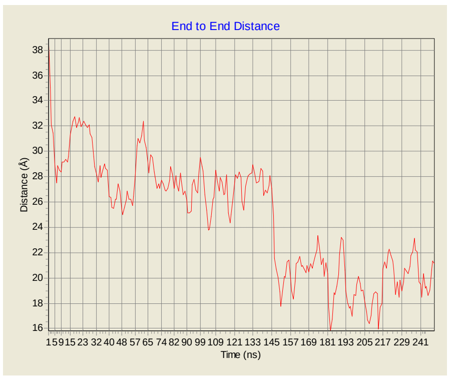


Figure 10: End-to-end distance of Simulated Protein

6.7. User testing

Initial user testing was performed where users were given a brief tutorial on how to operate the system. The users also received a brief explanation of what the system hoped to achieve, so as to inform them of what the information on the screen meant. The users were then allowed to explore the system and interact with it. Although the system is not complete, the aim of the initial testing was to extract information from the users as to their experience of the system.

A session was set up where each of the seven users received a handout explaining what the DOCKside [7] system does and what information the solution viewer aims at communicating to the user. The users that were tested did not have any previous knowledge of the system. Each user was asked to open a solution file and explore the solution space. After the user explored the space, s/he was required to complete a questionnaire. The questionnaire contained five statements and the user was asked to assign a number, on a scale of one to ten, indicating how much s/he agrees with the statement. At the end of the questionnaire, the user was able to note any comments they had about the system, enabling them to give some constructive feedback on their experience.

More sound testing methods will be adopted once a complete system has been developed. There is also the problem of not having a current system with which the solution viewer can be compared since the literature review indicated that most solution viewers in this area of visualization make use of two-dimensional slice viewers.

6.8. Results

The results of the testing revealed that the users found the program fairly intuitive to use. This indicates that using the MFC libraries was a good idea since the users were familiar with the environment. The users were able to open the test file without any trouble since the program reflected the open file command used in other Microsoft applications.

The users had a few complaints about the camera movement. The camera movement does not correlate. When the user pushes the up button, the information is translated upward and when the user pushes the down button, the information is translated downward. However, when the right button is pushed, the information is translated to the left and the other way round for pushing the left button. This can easily be corrected since it only requires the swapping around of mathematical equations that control the camera movement.

Since the users did not have any previous experience with docking procedures, they found it difficult to interpret what the information on the screen meant. Only after an in-depth discussion on how the automated docking system works, were the users able to interpret the information that was presented to them.

The users found the visualization method to serve as an effective way to communicate the information. None of the users were disoriented by the vectors that extend from the sphere. Some of the users did not realize that the text at the bottom of the screen displayed information that was relevant to the solution space until they had to answer the questionnaire. In future work, a way must be found to make the text more prominent, so that it would draw the user's attention.

Table 1: Results obtained from user testing

Question	Average
I found the program intuitive to use	7.14
The camera movement is intuitive and I found it easy to use	7.74
The information that is displayed, serves the purpose of conveying the needed data in a useful manner	7.28
I found the three-dimensional view represented the information in an intuitive fashion	8.28
I found the combination of text and graphics to be a good way to represent the data	7.86

One problem that was discovered during the testing and not addressed as of yet, is that users had difficulty selecting some of the vectors. It is not clear at this time whether this was caused by the fact that the users were working on a touch pad instead of using a mouse or due to the fact that a colour picking buffer was used and that the drawing thickness of the vectors were only three pixels wide. If the problem occurred because of the line thickness, one could increase the thickness in the picking buffer although that might cause occlusion errors.

7. CONCLUSION

DOCKside has proven to be able to accurately and efficiently produce a docking solution in a way that is easy to view and use. However, it still requires some adjustments and improvements to elevate itself to the level of similar systems within the field.

The solution viewer was developed in the attempt to create a visual representation of the data that is generated by DOCKside. The solution viewer allows the user to explore a three-dimensional space that contains all the solutions. The space allows the user to translate and rotate the camera, giving them complete freedom in exploring the data. The users are also able to select any one of the outer solutions to get a more detailed view of the selected solution. The detailed view allows the user to view all the inner solutions available for the selected solution.

Since the user testing was done with a small group of testers, the testers were unfamiliar with the complete system and the fact that the questionnaire contained qualitative information; the results are preliminary and inconclusive but the testing did reveal some important insights to the future development of the program. The testing enabled the developers to discover possible pitfalls that would only be brought to light when users that are unfamiliar with the system interacted with it.

8. FUTURE WORK

The improvement of DOCKside is a high priority in the future, requiring the vector quantization be adapted to handle the Kabsch algorithm and the possible implementation of a flexible docking algorithm. It is also essential to optimize some of the algorithms, such as the filtering algorithms, and provide vector quantization with the ability to handle larger vector sizes.

An implementation of the novel flexible docking method is a task that can be taken up in the near future. To go about attempting this, flexible fitting of high resolution atomic structures into low resolution EM maps using MD simulation can be achieved using a method called difference mapping [15]. Difference mapping between the density calculated from the fitted structure and the EM map can locate portions of the structure not present in the EM map. External forces can then be applied to these portions, pushing them back into the EM map

9. REFERENCES

[1] Assa J, Kohen-Or D, Milo T**.** *Displaying data in multidimensional relevance space with 2D visualization maps* Proceedings Visualization '97 (Phoenix, AZ) Los Alamitos, CA: IEEE Computer Society Press, 1997, 127-134

[2] Dos Santos S R, Brodlie K W. *Visualizing and investigating multidimensional functions.* Proceedings of the Symposium on Data Visualization 2002 (Barcelona, Spain), 2002, 173 – ff, ISBN: 1-58113-536-X

[3] Feiner S, Beshers C. *Worlds within worlds: Metaphors for exploring n-Dimensional Virtual Worlds.* Readings in

Information Visualization: Using vision to think. 1999, 96 – 103. ISBN: 1-55860-533-9

[4] Inselberg A, *Multidimensional Detective.* Readings in Information Visualization: Using vision to think. 1999. 107 – 114. ISBN: 1-55860-533-9

[5] Rueden C, Eliceiri K W, White J G. *VisBio: A computational tool for visualization of multidimensional biological image data.* Traffic 2004 5:6, 411. DOI: 10.1111/j.1600-0854.2004.00189.x

[6] Schneiderman, B. *The eyes have it: a task by data taxonomy for information visualizations.* In Proceedings of 1996 IEEE conference on Visual Language, IEEE comp. soc. Press, 1996 pp. 336-343

[7] Snowden A., Stern G., Kuttel M., Gain J. *DOCKside - A Tool for Docking Atomic Molecular Structures into Low-Resolution Electron Microscopy Graphs*. Technical Report CS05-08-00, 2005. Department of Computer Science, University of Cape Town.

[8] Spears W M. *An overview of multidimensional visualization techniques.* In Collins T D (editor) Evolutionary Computation Visualization, 104-105, Orlando, Florida, 1999

[9] Stump G M, Yukish M, Simpson T W, Harris E N. *Design space visualization and its applications to a design by shopping paradigm.* ASME 2003 Design Engineering Technical Conferences and Computer and Information in Engineering Conference. September 2 – 6, 2003 (Chicago, Illinois)

[10] Wriggers W., Milligan R. A., McCammon J. A. *Situs: A Package for Docking Crystal Structures into Low-Resolution Maps from Electron Microscopy*. J. Structural Biology, 1999, Vol. 125, pp. 185-195.

[11] TS Baker, JE Johnson. *Low resolution meets high: towards a resolution continuum from cells to atoms.* Curr. Opin. Struct. Biol, 1996. Vol. 6, pp. 585-594.

[12] W Wriggers, RA. Milligan, K Schulten, and JA McCammon. *Self-Organizing Neural Networks Bridge the Biomolecular Resolution Gap. J. Molecular Biology,* 1998, Vol. 284, pp. 1247-1254.

[11] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kale and K. Schulten. *Scalable molecular dynamics with NAMD*. Journal of Computational Chemistry, 2005, Vol. 26, pp 1781-1802.

[12] M. Nelson, W. Humphrey, A. Gursoy, A. Dalke, L. Kalé, R. D. Skeel, and K. Schulten. *NAMD - A parallel, object-oriented molecular dynamics program*. International Journal of Supercomputer Applications and High Performance Computing, 1996, Vol. 10, pp 251-268.

[13] M. Nelson, W. Humphrey, A. Gursoy, A. Dalke, L. Kalé, R. Skeel, K. Schulten, and R. Kufrin. *MDScope - A visual computing environment for structural biology*. In S.N. Atluri, G. Yagawa, and T.A. Cruse, editors, Computational Mechanics *95*, 1995, Vol. 1, pp. 476-481

[14] *NAMD User's Guide*.
http://www.ks.uiuc.edu/Research/namd/2.6/ug/

[15] A. Hoenger, S. Sack, M. Thormählen, A. Marx, J. Müller, H. Gross and E. Mandelko
Image Reconstructions of Microtubules Decorated with Monomeric and Dimeric Kinesins: Comparison with X-Ray Structure and Implications for Motility. Journal of Cell Biology, Vol 141, 1998, pp 419-430.