IOWA STATE UNIVERSITY
Digital Repository

2006

Interactive visualization of metabolic networks using virtual reality

Yuting Yang
Iowa State University

www.manaraa.com

Interactive visualization of metabolic networks using virtual reality

by

Yuting Yang

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering

Program of Study Committee:
Julie A. Dickerson, Major Professor
Carolina Cruz-Neira
Dirk Reiners
Eve Syrkin Wurtele
Manimaran Govindarasu

Iowa State University

Ames, Iowa

2006

UMI Number: 3229141

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Graduate College
Iowa State University

This is to certify that the doctoral dissertation of

Yuting Yang

has met the dissertation requirements of Iowa State University

Signature was redacted for privacy.

Major Professor

Signature was redacted for privacy.

For the Major Program

Dedication

To my beloved wife, Ying Feng, my parents Lianying Ye and Guozhen Yang, and my

brother Jinting Yang

TABLE OF CONTENTS

viii

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

A combination of graph layouts in 3D space, interactive computer graphics, and virtual reality (VR) can increase the size of understandable networks for metabolic network visualization. Two models, the directed graph and the compound graph, were used to represent a metabolic network. The directed graph, or nonhierarchical visualization, considers the adjacency relationships. For the nonhierarchical visualization, the weighted GEM-3D layout was adopted to emphasize the reactions among metabolite nodes. The compound graph, or hierarchical visualization, explicitly takes the hierarchical relationships like the pathway-molecule hierarchy or the compartment-molecule hierarchy into consideration to improve the performance and perception. An algorithm was designed, which combines the hierarchical force model with the simulated annealing method, to efficiently generate an effective layout for the compound graph. A detail-on-demand method improved the rendering performance and perception of the hierarchical visualization. The directed graph was also used to represent a sub-network composed of reactions of interest (ROIs), which reveal reactions involving a specific node. The fan layout was proposed for ROIs focusing on a metabolite node. The radial layout was adopted for ROIs focusing on a gene node. Graphics scenes were constructed for the network. The shapes and material properties of geometric objects, such as colors, transparencies, and textures, can encode biological properties, such as node names, reaction edge types, etc. Graphics animations like color morphing, shape morphing, and edge vibration were used to superimpose gene expression profiling data to the network. Interactions for an effective visualization were defined and implemented using VR interfaces. A pilot usability study and some qualitative comparisons were conducted to show potential advantages of stereoscopic VR for metabolic network visualization.

CHAPTER 1.   INTRODUCTION

1.1 Motivation

Metabolic networks are composed of two interwoven processes, *metabolism* and *regulation*. Within all organisms, chemical reactions and energy transformations occur that are essential to nutrition, growth and repair of cells, and conversion of energy into usable forms. The sum of all the biochemical activities of the organism is its metabolism [Solomon et al. 1999]. The essential purpose of metabolism is the flow of mass and energy. Regulating mechanisms precisely control the metabolic flows. Since metabolism is fundamental to life processes, understanding metabolic networks and profiling data is of utmost importance. A better understanding of metabolism may help to predict the effects of a given drug on human metabolism, the consequences of changes in a single gene on the composition of seed, or the effect of a given mutation in a pre-cancerous cell. A metabolic network can be expressed as a series of *pathways*. A pathway is composed of a series of interconnected chemical *reactions* among various molecules. Each reaction can be further classified as a *metabolic reaction* or a *regulatory reaction*.

These networks are complex and difficult to interpret. Visualization can aid the understanding of complex metabolic networks, gene expression profiling data, and their relationships. The human genome contains at least 28,000 genes [Lander and Linton 2001]. Genomes of this size create very complex metabolic networks, exceeding people's ability to understand text descriptions of the networks. An effective visualization can reveal global structures and local details that are not otherwise apparent.

The motivation of this research is to exploit 3D space, computer graphics, and VR for large scale metabolic network visualization. Metabolic network visualization belongs to the field of graph drawing in the area of information visualization. Information

visualization presents information that consists of entities and the relationships among them [Chen 2004]. In a graph, entities are nodes; relationships are edges. The process of graph drawing includes three stages [Kamada 1989]: modeling, layout, and rendering. In the modeling stage, a graph model is extracted from the original data. The layout stage assigns positions to nodes and edges in two-dimensional (2D) or three-dimensional (3D) space. The rendering stage produces an image of the graph. Most current research on metabolic network visualization exploits 2D space in the stage of layout and produce flowchart-like displays in the stage of rendering [Karp and Paley 1994; Becker and Rojas 2001; Dickerson et al. 2001; Schreiber 2002; Schreiber 2003; Shannon et al. 2003; Toyoda et al. 2003]. Current graph drawing methods and current metabolic network visualization systems and are described in Section 2.1 and Section 2.2 respectively.

Computer graphics uses computers to generate visual images for a real world or a synthetic world. It includes a *scene*, the set of geometric representations for the real world or the synthetic world and the rendering of the scene. The resultant images are usually displayed in 2D media, such as monoscopic computer monitors or paper. Virtual reality (VR) extends the display into the third dimension by introducing stereoscopic display. VR also uses advanced devices, such as head and wand tracking devices and various input devices, to enable richer interactions than convention computers. Computer graphics and VR are briefly introduced in Section 2.4.

The potential advantages of 3D space and VR for general graph visualization have been noted by some researchers. Ware's studies [Ware et al. 1993; Ware and Franck 1994; Ware and Franck 1996] compared 3D space with 2D space for graph visualization. Different configurations were tested for a path tracking task, including the 2D parallel display, the 2D perspective display, the 2D perspective display with the head tracking, the 2D perspective display with user interactions to translate and rotate the graph, the

stereoscopic display, the stereoscopic display with the head tracking, and the stereoscopic display with user interactions to translate and rotate the graph. The response times were relatively uniform across the configurations. The main difference was found in error rates. The last two configurations had the lowest error rate. The 2D display configuration had the highest error rate. Ware found that for a given error rate, the size of a graph viewed in stereoscopic VR with the manipulation of the viewpoint can be three times as large as that in a 2D plane. Using stereo alone appears to increase the graph size by a factor of 1.6 and using the head tracking alone appears to increase the graph size by a factor of 2.2. 3D space and VR have been adopted for software visualization [Parker et al. 1998; Maletic et al. 2001; Maletic et al. 2001] and statistical data visualization[Nelson et al. 1999].

This research is to exploit 3D space, computer graphics, and VR for large scale metabolic network visualization. It is a significant challenge to convey large amounts of information in metabolic networks. Layouts of metabolic networks in 3D space offer various benefits. The extra dimension gives greater flexibility for placing nodes and edges. Computer graphics improves the display of the network by generating and rendering geometric representations for nodes and edges. Computer graphics rendering can be an interactive process by enabling the user to move the viewpoint and change the shape and material properties (such as color, transparency, and texture) of geometric representations. VR technologies enable the user's interaction to change the viewpoint position and many other interactions such as selecting nodes. VR also features stereoscopic displays. Layouts in 3D space, material properties in computer graphics, and virtual reality environment will help people to explore high dimensional networks. This thesis proposes an interactive visualization of large scale metabolic networks in VR. The data flowchart of the visualization is in Figure 1.1.
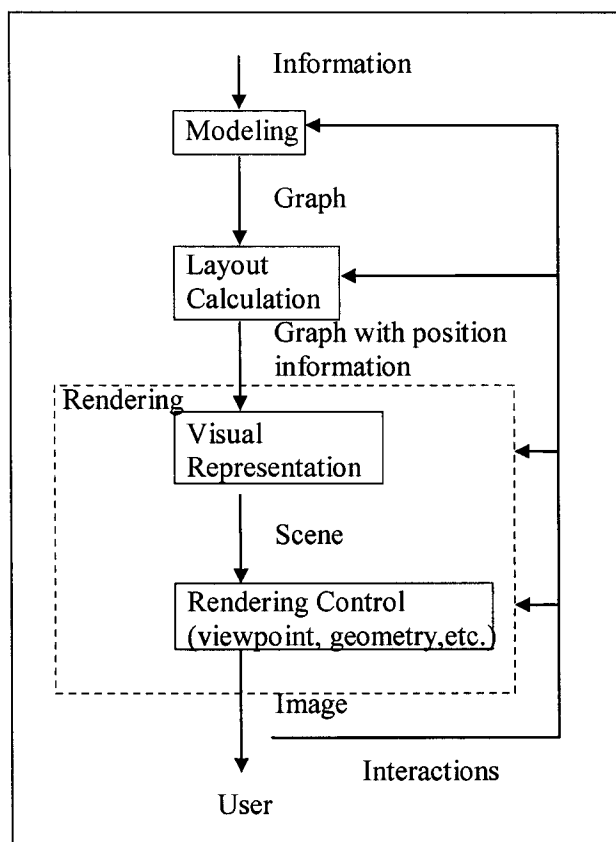
Figure 1.1 Data flowchart of interactive visualization of metabolic networks

1.2 Major contributions of this dissertation

This dissertation presents five major contributions in metabolic network visualization. I here:

- Proposed and implemented a layout algorithm for reactions of interest focusing on a metabolite node (Section 5.1).

- Adapted the radial layout highlighting reactions of interest focusing on a gene node (Section 5.2).

- Designed a layout algorithm taking both hierarchical relationships and adjacency relationships in the metabolic network into consideration (Section 4.3).

- Developed detail-on-demand interactions, which exploits the hierarchical relationships, to improve the performance and the perception of the metabolic network visualization (Section 4.2).

- Conducted a pilot usability study testing the effectiveness of virtual reality for the visualization of large scale metabolic networks (Section 7.2).

1.3 Organization of this thesis

Chapter 2 introduces the basic concepts of metabolic networks, gene expression profiling data, existing metabolic networks visualization systems, graph drawing algorithms, computer graphics, and virtual reality. Chapter 3 discusses how to achieve a global view of a metabolic network and gene expression profiling data, including the visual representations for both of them, a layout algorithm for the network, and the combination of them in virtual reality. Chapter 4 introduces the compound graph model for metabolic networks containing adjacency relationships and hierarchical relationships, a layout algorithm for the compound model, and a detail-on-demand display method. Chapter 5 proposes the concept of reactions of interest and the display methods, including special layouts and layout animation. Chapter 6 presents some methods to integrate the metabolic network with gene expression profiling data. Chapter 7 integrates these pieces into MetNetVR, a cross-platform system for metabolic network visualization, and describes a usability test with the hypothesis that the combination of 3D space, computer graphics rendering, and VR, are helpful to visualize large scale metabolic networks. A qualitative comparison between 3D space and 2D space for metabolic network visualization for metabolic network visualization and a qualitative comparison between the hierarchical visualization and the nonhierarchical visualization are also presented in Chapter 7. Chapter 8 gives some conclusions and lists future work.

CHAPTER 2.   BACKGROUND

Pathways are conventionally drawn as diagrams. Figure 2.1 shows the diagram of a pathway depicting the citric acid cycle from a standard text book. Today, there are many databases of static pathway diagrams published online at sites such as KEGG [Kanehisa and Goto 2000] and TAIR (http://www.arabidopsis.org/tools/aracyc/).
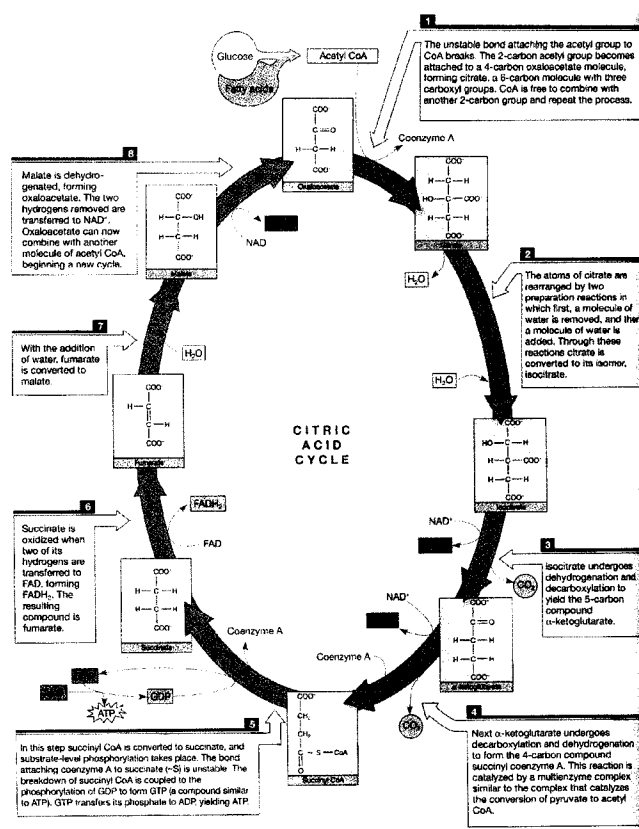
Figure 2.1 Citric acid cycle [Solomon et al. 1999]

Static visualization of metabolic networks suffers from disadvantages such as the high cost of manual drawing, the difficulty of modifying the network, and the absence of interactions. Research efforts are now focusing on visualizing metabolic networks dynamically. Dynamic visualization is the automatic generation of diagrams that represent user-selected portions of the entire network at the time they are needed.

Dynamic visualization of metabolic networks requires a mathematical model to describe metabolic networks. A metabolic network can be modeled as a directed graph, G (N, E), where N is the set of nodes representing various molecule nodes in the network; and E is the set of reaction edges connecting nodes. In this research, nodes in the metabolic network can be classified into metabolite nodes, gene nodes, RNA nodes, polypeptide nodes, protein complex nodes, and reaction nodes. A reaction node is introduced to convert a hyper reaction edge in the network to multiple edges in the directed graph. For example, reaction 8 in Figure 2.1 is a reaction generating oxaloacetate and NADH from malate and NAD. The relationship between reactants and products in the reaction is a hyper edge, which has multiple tails or multiple heads. The hyper edge is converted to multiple edges in the directed graph model by introducing a reaction node. After the conversion, the relationship is represented as the following reaction edges, malate->reaction node, NAD->reaction node, reaction node-> oxaloacetate, and reaction node->NADH. Reaction edges can be classified into enzymatic edges, catalysis edges, assembly edges, transcription edges, translation edges, negative regulation edges, and positive regulation edges [Wurtele et al. 2003].

2.1 Graph layouts

Current metabolic network visualization methods belong to the field of graph drawing. Graph layout problem is one of the most important issues in graph drawing. Graph layout algorithms are to automatically decide node placements and edge routings in graph visualization.

2.1.1 Conventions and rules for graph layouts

It is very difficult to accurately define a 'good' graph layout. Sugiyama [Sugiyama 2002] lists the conventions and rules for good graph layouts, which reflect the

characteristics of a graph. Conventions are constraints that must be fulfilled. Rules are optimization criteria. Conventions are rules are listed in Table 2-1, 2-2. and 2-3.

There are dependencies and competitions between conventions and rules. Although different applications have individual priorities concerning convention and rules, Sugiyama gives some general priority relationships, i.e., conventions have higher priority than rules; placement conventions have higher priority than routing conventions; semantic rules has higher priority than structural rules [Sugiyama 2002].

Table 2.1 Placement conventions for nodes [Sugiyama 2002]
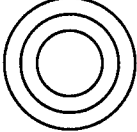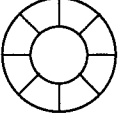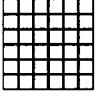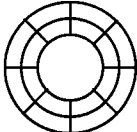
Node Placement Conventions	Coordinate System
Free placement	
Parallel line placement	
Concentric circle placement	
Radial line placement	
Orthogonal grid placement	
On the interaction of a polar grid	

Table 2.2 Routing conventions for edges [Sugiyama 2002]

Type	Routing Conventions
Line Type	Straight edge routing
	Poly-line routing
	Curve Routing
Relationship with coordinate system	Routing is independent of coordinate system axes
	Routing is parallel or perpendicular to coordinate system axes

Table 2.3 Graph drawing rules [Sugiyama 2002]

Type	Rules
Semantic Rules (Rules derived from the meaning of nodes or edges)	1. Place specified nodes on a straight line
	2. Place specified nodes on a curve
	3. Draw nodes with specified sizes
	4. Place specified nodes at the boundary of the layout
	5. Place specified nodes nearly
	6. Place specified node near the center
	7. Specify the upper limit to the number of edge crossings
	8. Specify the upper limit to the number of edge bends
	9. Specify the upper limit to the lengths of edges
Structural Rules (Rules concerned with the graph-theoretic features of a graph)	1. Central placement of high degree nodes
	2. Identical layout of isomorphic sub-graphs
	3. Hierarchical structure is shown vertically or horizontally
	4. Minimizing edge crossings
	5. Balance of the length and the breath of the layout
	6. Symmetry is clearly shown
	7. Minimize edge bends
	8. Draw faces as convex polygons
	9. Place children symmetrically
	10. Avoid crossings among outlines
	11. The density of the placement and the routing is uniform
	12. Minimize drawing area
	13. Minimize the total edge length
	14. Minimize the difference in node sizes
	15. Minimize the average length of edges
	16. Minimize the difference between the length of node contours and the length of edges
	17. Minimize the differences in edge lengths
	18. Minimize the length of the longest edge
	19. Nodes on the boundary are placed with uniform density

2.1.2 Graph layout algorithms

Layout algorithms for directed graphs can be grouped into three categories: physical based layouts, layered layouts, and orthogonal layouts. Physical based methods draw a formal analogy between the layout problem and the behavior of some idealized physical systems. The key features are a randomized initial embedding, a force model or an energy model, and an algorithm searches for an equilibrium state where the total force on each node is zero or a state with minimum total energy. Since force is the negative gradient of energy, the equilibrium state in the force model corresponds to the state of minimum total energy in the energy model. Figure 2.2 shows the analogy between a spring network and a spring force directed layout. Physical based methods are comparatively easy to implement, adaptable to different drawing criteria and give satisfactory results for many applications. The spring force directed layout [Eades 1984] is the first one of physical based layouts. Various heuristics [Kamada and S. 1989; Fruchtermann and Reingold 1991; Frick et al. 1994; Tunkelang 1994; Davidson and Harel 1996] are later presented to accelerate the convergence toward the equilibrium state. Although the physical based methods were presented for 2D layouts, they don't have any limitation on dimensions. It is very naturally to extend them into 3D [Bruss and Frick 1995; Monien et al. 1995].
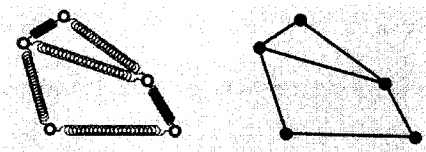
Figure 2.2 Spring force directed layout

Layered layouts partition nodes into layers and order the nodes within each layer so that the number of edge crossings is minimized. The first layered layout method was

introduced in [Sugiyama et al. 1981]. It is thus called Sugiyama method or STT method.

Variations of STT methods have been proposed [Rowe et al. 1987; Sander 1995]. Ostry

[Ostry 1996] extended the layering methods into 3D space.
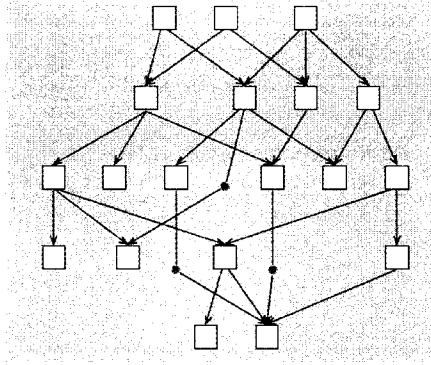
Figure 2.3 Layered layout. Green dots represent virtual nodes added for edges crossing

multiple layers.

An orthogonal layout [Batini et al. 1984; Tamassia et al. 1988] places all nodes at

the intersections of the orthogonal grid. The edges are routed parallel to, or perpendicular

to the grid lines. Edge routes are allowed to contain bends, but are not allowed to cross or

to overlap. A survey of 3D orthogonal layout algorithms [Landgraf 2001] points out that

existing 3D orthogonal drawing methods are less applicable in practice compared with

other two groups of methods. When applied to large graph, they only produce satisfactory

drawing for special applications like VLSI design.
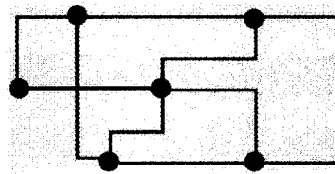
Figure 2.4 Orthogonal layout

Munzner [Munzner 2000] compared the scalability of general purpose graph drawing systems using three basic layout algorithms. More than one-half of them can only handle less than one hundred nodes. Only the Gem-3D layout [Bruss and Frick 1995] and the dot layout [Gansner et al. 1993] can handle hundreds or even a thousand nodes. The former belongs to physical based layouts. The latter is based on the STT method.

2.2 Existing metabolic network visualization systems

2.2.1 2D visualization of metabolic networks

Many researchers have been applying two-dimensional (2D) graph layout algorithms for metabolic network visualization. Paley [Karp and Paley 1994] uses the 'divide and conquer' method. It extracts the longest cycle from the graph. The rest of the graph is grouped into several strongly connected components, each of which is substituted for by a temporary node. It then uses the basic snake layout, tree layout and circular layout to locate all temporary nodes and all cycle nodes. Finally, each temporary node stretches to the strongly connected component and is recursively laid out using the same method. Becker [Becker and Rojas 2001] improves this method by introducing a spring embedding algorithm into the more complex layout problems where basic snake layout, tree layout and circular layout cannot fit. Schreiber [Schreiber 2002] uses the well-known Sugiyama method for hierarchical drawing [Sugiyama et al. 1981] and adds application-specific constraints on the layer assignment of a vertex and its position within the layer. Schreiber[Schreiber 2003] extends his previous work in order to enable the comparison of similar pathways across species by drawing similar pathways side by side, and placing vertices representing same substances in similar pathways on the same horizontal layer. Toyoda [Toyoda et al. 2003] applies the fisheye view method [Sarkar and Brown 1992] to increase the size of understandable networks. Instead of using a single layout method, FCModeler [Dickerson et al. 2003] integrates rank and cluster, dot

(http://www.research.att.com/sw/tools/graphviz/) and Graph-Embedder [Frick et al. 1994]

layouts in one package for dynamic display and fuzzy modeling of regulatory and

metabolic networks (http://clue.eng.iastate.edu/~julied/research/fcmodeler/index.html).
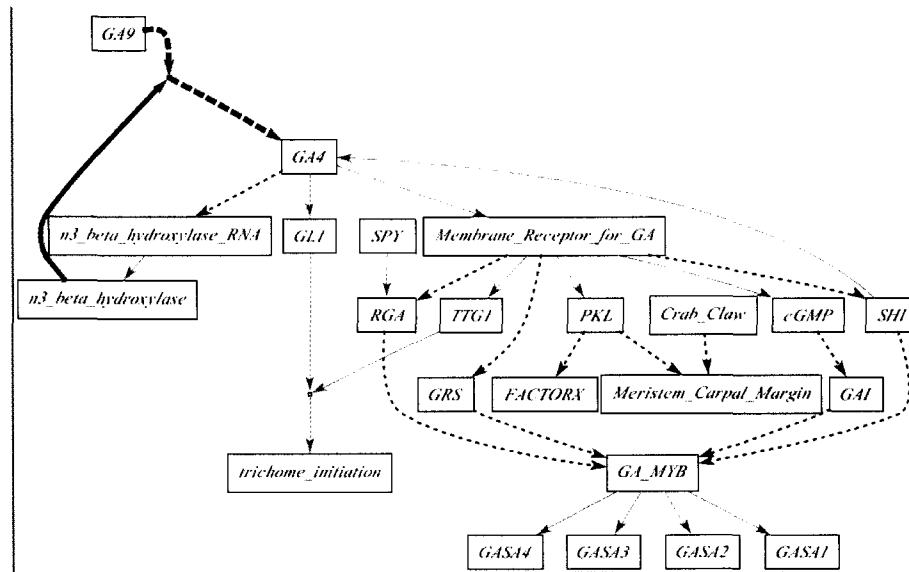
Figure 2.5 A snapshot of FCModeler

2.2.2 3D visualization of metabolic networks

Researchers have started to use 3D graph layout algorithms to visualization

metabolic networks in 3D space. One method, developed by Brandes [Brandes et al. 2003;

Brandes et al. 2004], exploits 3D space to compare similar pathways across species. Each

pathway across species is drawn in one 'stratum'. In the third dimension, all strata are

ordered so that the most analogous pathways are adjacent. The pathway in each stratum is

drawn using the layered method (dot from

http://www.research.att.com/sw/tools/graphviz/) to minimize edge crossings. The authors

call their method 'visualizing related metabolic networks in two and a half dimensions'.

The system uses the 3D rotation or the stereoscopic display to enhance the depth

perception (Figure 2.6). An advanced interaction is available to select and display a cut

place along the third dimension to get detailed information about a pathway (the right snapshot in Figure 2.6). The usage of 3D space here is specific (that is why it is called two and a half dimensions); can not be generalized.
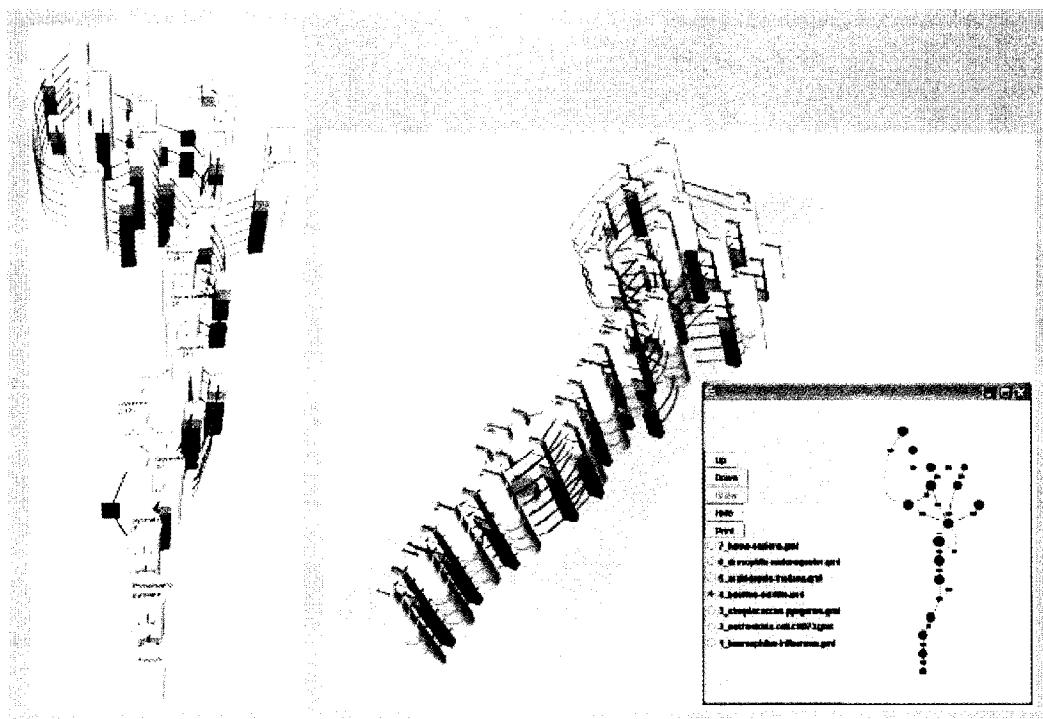
Figure 2.6 Visualizing related metabolic pathways in two and a half dimensions [Brandes et al. 2003; Brandes et al. 2004].

The VRML Metabolic Network Visualizer [Rojdestvenski and Cottam 2002; Rojdestvenski 2003] translates the XML file that represents the metabolic network into a Virtual Reality Modeling Language (VRML) file so that users can browse the network on-line. A spring embedding algorithm is used to calculate the layout of the graph. The interactions are enabled by the VRML browser. The scalability of VRML Metabolic Network Visualizer isn't discussed in the paper. On its website (http://www.patronov.net/sciencevr/mnv/screenshots.html), users can select a single pathway to visualize.
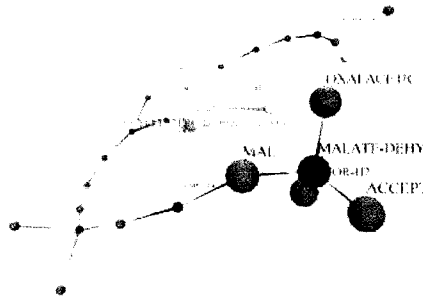
Figure 2.7 A snapshot of VRML Metabolic Network Visualizer

MetNetVR combines graph layouts in 3D space, computer graphics, and VR technologies for interactive visualization of high dimensional metabolic networks and gene expression profiling data [Dickerson et al. 2003; Yang et al. 2005; Yang et al. 2006].

2.3 Metabolic networks and large-scale profiling experimental data

Systems biology is an academic field that seeks to integrate different levels of information to understand how biological systems function. System biology includes genomics, transcriptomics, proteomics and metabolomics. Genomics is the study of an organism's genome and gene functions. Transcriptomics data depicts the level of the gene transcripts, often using techniques capable of sampling tens of thousands of different mRNA molecules at a time (e.g., DNA microarrays). Proteomics is the large-scale study of proteins, particularly obtaining data on the proteins present in a biological sample [Oliver et al. 2002]. Metabolomics defines the concentration of metabolites and identifies the biochemical mechanisms that regulate the flux through metabolic networks.

The integration of the data from all sub-fields is very important to understand biological systems. Some sub-fields are relatively more mature than others. This research specially addresses the integration of metabolic networks and the transcriptomics data. However, the tools presented here will also apply to other data.

An organism constantly responds to external and internal stimuli by altering the flux though its metabolic networks. Levels about this flux can be revealed by multi-dimensional genome-wide transcriptomics data, sometimes called *gene expression profiling data* or microrray data, which provide a rough measure of the cellular accumulation of different messenger RNAs. Using recent gene probe tools such as Affymetrix's Genechip® has made available large scale transcriptomics data for different species. These tools are semiconductor devices that contain hundreds of thousands of tiny segments designed to reach with a particular transcript. Figure 2.8 shows a color map of a microarray where different colors indicate relative expression of different genes (http://science.nasa.gov/headlines/y2004/images/radmicrobe/microarray.jpg).
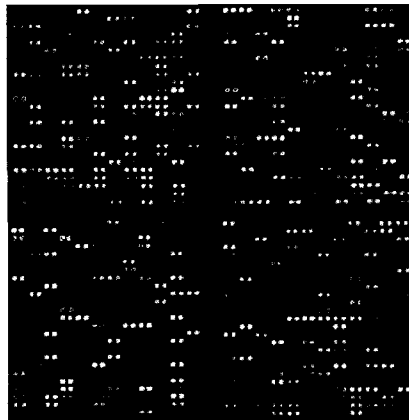
Figure 2.8 Different colors indicate relative expression levels of different genes.

TreeView (http://rana.lbl.gov/EisenSoftware.htm) is one of the most commonly used methods for visualizing transcriptomics data [Eisen et al. 1998]. Different colors indicate relative expression levels of different genes. The genes are clustered in the X axis. The change of gene expression levels along the time is displayed along the Y axis (Figure 2.9).
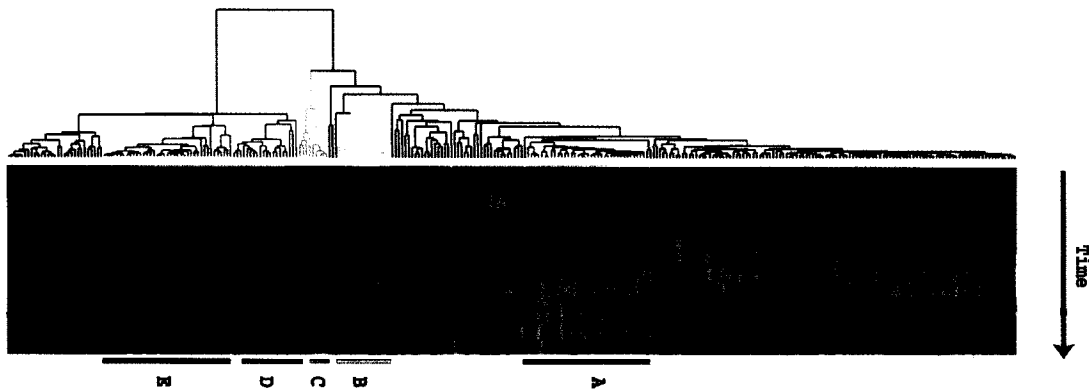
Figure 2.9 Treeview of microarry data [Eisen et al. 1998]

2.4 3D computer graphics and virtual reality

The final stage of graph drawing is rendering, which produces an image of the graph. Most current research on metabolic network visualization systems only produce flowchart-like displays in this stage. This research exploits 3D computer graphics and virtual reality to enhance the visualization.

2.4.1 Rendering pipeline in computer graphics

The primary objective of 3D computer graphics is to generate realistic images of a real world or a synthetic world. Figure 2.10 lists the rendering pipeline in computer graphics [Angel 2000]. In a computer graphics application, a *scene* includes all the geometric representations for objects to visualize. The scene is composite of a combination of various primitives and transformations. Geometrics primitives in computer graphics include points, lines and polygons. Primitives are composites of a number of vertices with position, orientation, and material properties. Primitives are initially in the *object space*. Each object has its own object space where it is constructed. The *modeling transformation* assigns the placement of an object in the *world space*. Each object has its own modeling transformation. There is only one world space. Rendering a scene is analogous to taking a photograph of the scene, and modeling transformations are

analogous to arranging objects in the scene. After the modeling transformations, the primitives are in the world space. The *viewing transformation* defines the position and the orientation of the viewpoint. It is analogous to setting up the tripod and pointing the camera at the scene. After the viewing transformation, primitives are in the *eye space*, where the origin is the viewpoint position (eye position). Next, the projection transformation is applied. The projection transformation defines a viewing volume. The parts of the scene located within the viewing volume will be visible in the image. The other parts are clipped. After the projection transformation, primitives are the clip space where the clipping happens. The projection transformation is analogous to choosing a camera lens or adjusting the zoom. *Viewport Transformation* controls the size of the image. It is analogous to determining the size of the photograph. *Rasterization* converts primitives into pixels in the image.

Two kinds of projections are typically used in computer graphics, orthographic parallel and perspective. Under an orthographic parallel projection, the viewpoint is infinitely far from the scene in the world space. Parallel lines in the scene are projected to parallel lines in the projection plane. Under a perspective projection, the viewpoint is at a finite point in the world space. Parallel lines in the scene are foreshortened to produce a vanishing point in the projection plane. The orthographic parallel projection is used in the applications where the image needs to reflect the measurements of objects such as blueprint plans. The perspective projection makes objects that are farther way appears smaller, generating realistic images. Stereoscopic rendering uses the perspective projection.
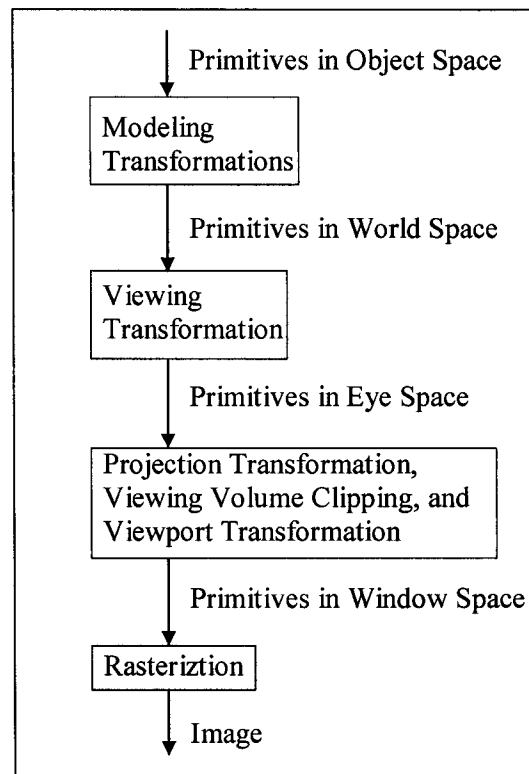
Figure 2.10 Rendering pipeline of computer graphics[Shreiner et al. 2005]

2.4.2 Lighting model and material properties

Besides the transformations discussed above, lighting model and texture mapping are two important issues that affect the image. The Phong lighting model is widely used in computer graphics. It calculates the color of each vertex in each primitive as the combination of four elements, i.e. diffuse color (C_d), specular color (C_s), ambient color (C_a) and emissive color C_e. They are defined as following,

- $C_d = M_d L_d \sin(\alpha)$

- $C_s = M_s L_s \cos^n(\beta)$

- $C_a = M_a L_a$

- $C_e = M_e$

where M is the color of the vertex and L is the color of a light source. The meanings of α

and β are illustrated in Figure 2.11, where $\vec{N}$ is the normal vector of vertex P. $\vec{L}$ is the

direction vector of the light source; $\vec{R}$ is the reflection vector of $\vec{L}$ along $\vec{N}$; $\vec{V}$ is the

viewing direction; α is the angle between $\vec{N}$ and $\vec{L}$; and β is the angle between $\vec{R}$ and $\vec{V}$.

Color multiplication and addition are component-wise.
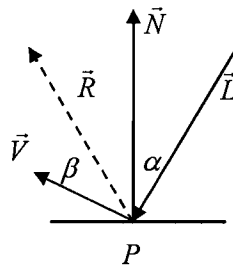
Figure 2.11 Phong lighting model

Multiple light sources are allowed in computer graphics. Diffuse component,

specular component, and ambient component of the final color of the vertex are the sums

of all light sources. The emissive component is only dependent on vertex itself.

Color is one of the most important material properties that affect the appearance

of the image. The other two are the texture map and transparency. Texture mapping

allows an image (texture) glued on the surface of a polygon. Texture images can be either

preloaded or generated on the fly. Texture mapping increases the developer's flexibility

to control the appearance of objects in the image. The transparency property makes the

translucent effect possible. An object behind another object whose transparency is zero is

totally blocked. An object behind another object whose transparency is larger than zero

still contributes to the image.

2.4.3 Virtual Reality

In traditional computer graphics, rendering is basically the projection of a 3D scene into a 2D image. The absence of depth information causes ambiguities in interpretation. Computer graphics has some techniques to remedy this problem, such as the user of the perspective projection, hidden-surface removal, atmospheric effects, and shadowing. While all of these techniques add monocular depth cues to the image, they still provide limited information. VR provides two additional techniques, binocular disparity and motion parallax, to add depth information. Binocular disparity is the difference in images projected on the left and right eyes in the viewing of a 3D Scene. Binocular disparity is modeled in computer graphics with eyes considered as two off-axis centers of perspective projection. Binocular disparity is enabled by stereoscopic displays and liquid crystal shutter glasses or polarizing glasses in VR. Motion parallax provides different views of a scene in responds to the movement of the viewpoint with the fixed scene placement or the movement of the scene with the fixed viewpoint position. It can be enabled by head tracking devices and input devices in VR.

The origin of VR traces back to the Head Mounted Display (HMD) proposed by Sutherland[Sutherland 1968]. Some VR platforms cross a very wide range, from fully immersive and room size CAVEs[Cruz-Neira et al. 1993], to HMD, to wall size projection screens, and to desktops with stereoscopic displays. VR also features 3D input devices like wands, space balls, and data gloves.

CHAPTER 3. GLOBAL VIEW OF METABOLIC NETWORKS AND GENE EXPRESSION PROFILING DATA

The display of an entire network gives users a good understanding of the network structure and serves as the starting point of the exploration. Given the graph model of a network, visual representation for nodes and edges, layout calculation, and rendering are needed to get an image of the network. Section 3.1 describes the visual representation for metabolic networks. Section 3.2 presents a modified GEM-3D layout algorithm that takes the biological knowledge into consideration. Section 3.3 describes the visual representation for gene expression profiling data. Section 3.4 introduces the rendering of scenes for both metabolic networks and gene expression profiling data in virtual reality.

3.1 Visual representation for networks
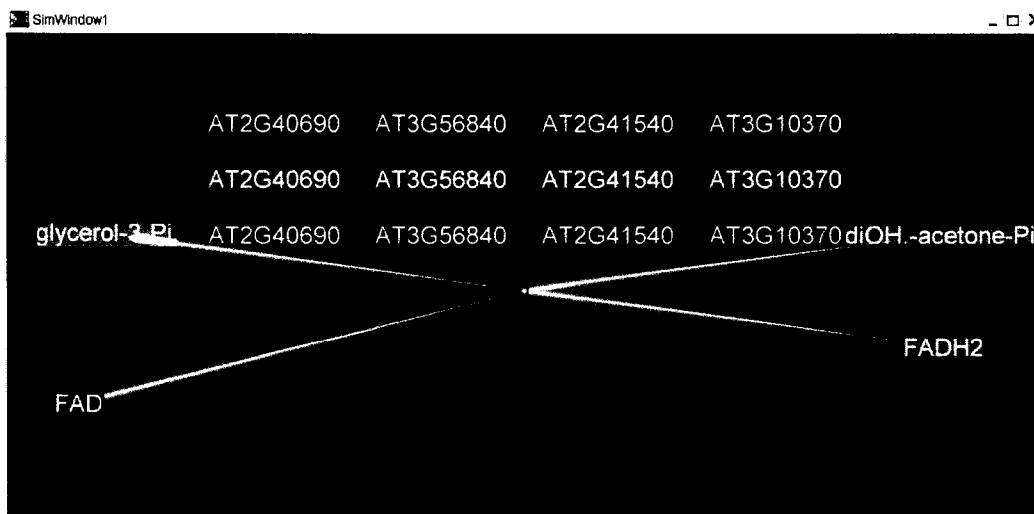
Figure 3.1 Visual representation of nodes and edges

Computer graphics rendering draws geometric objects with different shapes and material properties, such as color, texture maps, and transparency. Shapes and material properties are helpful to reveal useful information, such as molecule type, molecule names, and reaction edge directions in the visualization of high dimensional metabolic

networks. Metabolite nodes, gene nodes, RNA nodes, polypeptide nodes, and protein complex nodes are represented as labels indicating theirs names. A label has a white and opaque foreground as name texts and a translucent and colorful background. The background is used to reduce the confusion of letters from different labels visually overlapping. The translucent background is designed to avoid the visual blocking of the scene behind the label[Irani and Iturriaga 2002]. The labels are implemented by applying the *texture mapping* technique to a rectangle. Reaction nodes are represented as small spheres. It is not necessary to display names for reaction nodes since they are introduced to convert a hyper reaction edge in the network to multiple edges in the directed graph. Different types of nodes are assigned different colors. An edge is represented as a cone whose button indicates the edge tail and whose top indicates the edge head. Different edge types are assigned different colors.

Usually metabolic networks contain many nodes and edges. The scene representing the network contains many geometric primitives (points, lines, and polygons) composing the nodes and edges. Rendering is a computation-consuming task. To achieve an interactive frame rate, a technique in computer graphics called *level of detail* is used to reduce the number of the displayed primitives in some frames. The geometric objects (spheres and cones) representing nodes and edges have multiple levels of detail. Objects which are far away from the viewpoint are drawn using simplified models; only when an object is close enough to the viewer for all the details to be visible should a full, complex model be drawn. For example, a sphere is approximated using a polyhedron. An icositetrahedron is more detailed and contains more polygons than a tetrahedron.

Two more techniques in computer graphics are used to increase the label readability and the rendering quality in the network visualization. Labels have the highest readability when they are perpendicular to the viewing direction. Navigation through the

network, i.e., moving the viewpoint or moving the entire network, breaks this relationship. The *billboarding* technique is used to solve the problem. Billboarding was originally designed to simplify the rendering of a geometrically complex object, such as a tree, by drawing a polygon in the shape a tree, applying a texture map of foliage, and rotating the polygon around the viewing direction. It is much faster than creating the tree out of polygons. Billboarding is applied to the rectangles representing node labels to make them always perpendicular to the viewing direction in the navigation. The other technique, *mipmapping*[Williams 1983], is applied on the texture maps for the node labels. In the network navigation, as a label moves farther from the viewpoint, the texture map must decrease in size along with the size of the perspective projected label. Otherwise, visually disturbing artifacts, such as shimmering, flashing, and scintillation will be introduced. Mipmaps are a series of prefiltered texture maps of decreased resolutions. The mipmapping technique automatically determines which texture map to used based on the size of the rectangle (label) being mapped.

The visual representation is suitable to select an individual node or edge in the network. A technique in computer graphics called *ray casting* is used for selection. A virtual ray is sent into the scene and intersects geometric primitives of some nodes or edges. The node or edge object first hit is the selected node or edge.

3.2 Weighted GEM-3D layout

The 3D Graph-Embedder (GEM-3D) algorithm [Bruss and Frick 1995] belongs to the class of physical based methods (Section 2.3.2) . It is a spring-embedder approach, in which each edge acts as a spring and exerts a repulsive or attractive force upon the two nodes attached to it, dependent on the distance between them. The spring system converges to an equilibrium state with minimum energy. The computation of the repulsive force between any two nodes (μ and υ) is according to the equation of $\Delta / \left| \Delta \right| \times$

L, where Δ is the vector from the current position of μ to the current position of υ, $|\Delta|$ is the length of the vector, and L is the desired edge length. The computation of the attractive force of edge (μ, υ) is according to the equation of $\Delta \times |\Delta| / (L \times \Phi(\upsilon))$, where $\Phi(\upsilon)$ is a function that grows with the degree of υ. The GEM-3D method adds a gravitational force and several heuristics to speed up the convergence. It is one of the two methods that can handle hundreds or even a thousand nodes (The other one is the dot layout in 2D). It is adopted in this research to accommodate large scale metabolic networks.

The weighted GEM-3D layout modified the GEM-3D layout by taking the reaction edge type into consideration. The GEM-3D layout follows the drawing rule of edge length uniformity (Structural Rules 18 in Table 2-3). However, biologists prefer that enzymatic edges (from metabolite nodes to reaction nodes, or from reaction nodes to metabolite nodes) are longer than other types of edges to emphasize the relationships among metabolite nodes. The algorithm is modified by adding weight factors to the attractive forces of edges,

$$\Delta \times |\Delta|^2 \div (E^2 \times w \times \Phi(\upsilon))$$

where $w > 1$ for enzymatic edges and $w = 1$ for other edges. Figure 3.2(a) shows that enzymatic edges are longer than other edges in the final layout.

(a) A weighted GEM-3D layout. The weighting factors of other types of edges larger than

enzymatic edges (in yellow) in the calculation of attractive forces of edges. Node labels

are intentionally turned off to give a clear view of the layout. The green gadget is a

simulated wand interface in VR (Section 3.4)
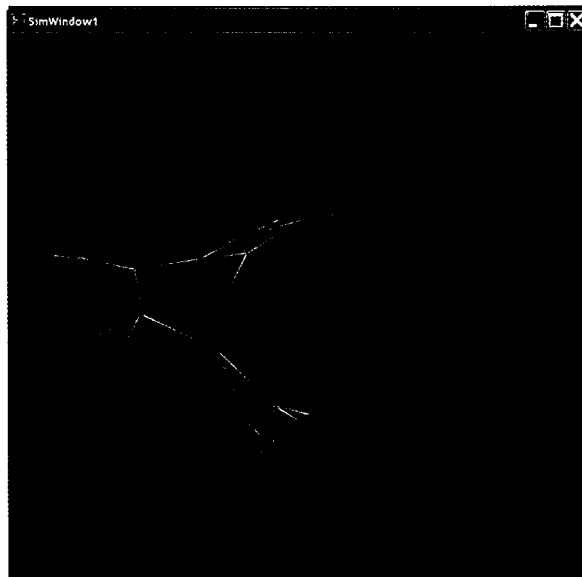
(b) A standard GEM-3D layout

Figure 3.2 A comparison between the weighted GEM-3D layout and the GEM-3D layout.

3.3 Visual representation for gene expression profiling data

Gene expression profiling data are represented as 3D plots as shown in Figure 3.3(a)[Yang et al. 2005]. Each sphere represents an expression level of a given gene as reflected by the accumulation of its corresponding RNA. Values are indicated by both color and height. The cylinder connecting two spheres shows the change in expression values. The series of spheres connected by cylinders form a connected line in 3D, showing the expression levels of a gene under different conditions (the plots on the right side in Figure 3.3). The consistency between color and height are implemented using the texture mapping technique. The color scale in Figure 3.3(b) is used as the texture map. The leftmost color in the scale indicates the lowest expression level of all genes under all conditions. The rightmost color indicates the highest expression level. A label is available for each connected line, indicating gene names and their availability in the currently displayed metabolic network. A label is also available for each expression level if the gene is selected (using the ray casting technique), indicating the corresponding condition. Labels are implemented the same as the labels in metabolic networks.

There are about 800 genes whose profiling data were used in this research. It is too much to display them at the same time. So, these genes were first clustered using the k-means method. The plots on the left side show the profiling data of clusters of genes that behave similarly. The bars through the spheres show the amount of variation within the cluster. The plots on the right side show the individual genes that make up a currently selected cluster. Genes with similar expression patterns are close to each other in the plots. The individual cluster or gene is selected using the ray casting technique.
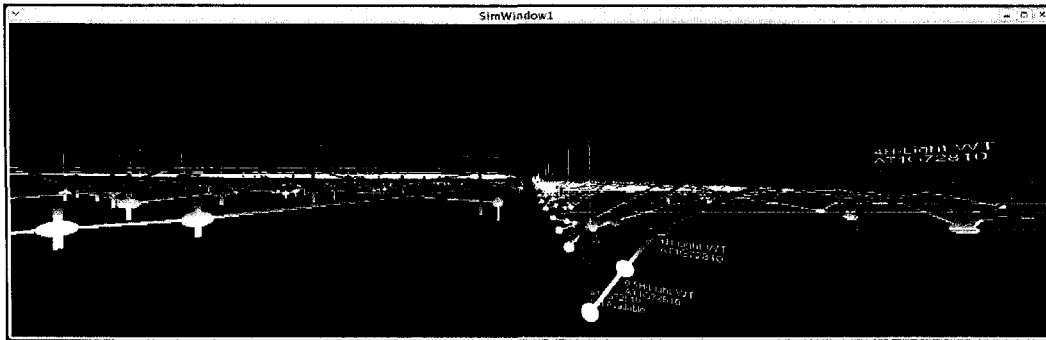
(a) 3D plots for clusters and genes

(b) Color scale for gene expression levels.

Figure 3.3 Visual representation for gene expression profiling data

3.4 Combining metabolic networks and gene expression profiling data in virtual reality

Metabolic networks and gene expression profiling data need to combine together for a better understanding of the dynamic behaviors of metabolic network. They are represented as scenes uniformly and rendered in virtual reality environments (Figure 3.4). Users navigate through the scenes. The relative position between two scenes is adjustable.
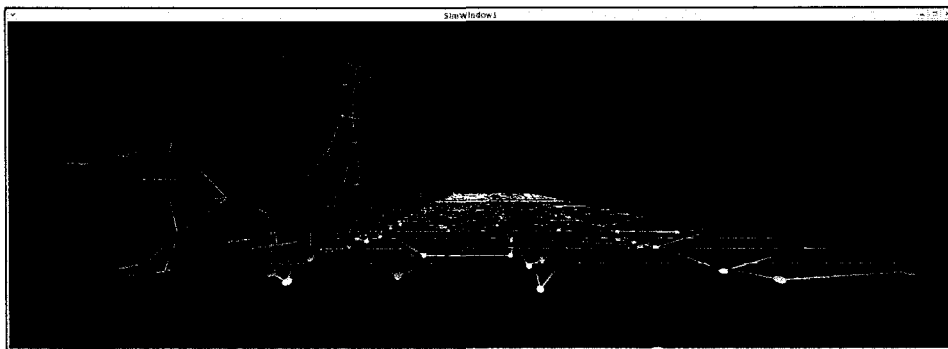
Figure 3.4 Combining a metabolic network and gene expression profiling data together in VR

Metabolic networks and gene expression profiling data communicate with each other. Both of them contain the same gene names (Not all genes in the expression data link to the networks since even in model organisms such as Arabidopsis, the function of most genes is still unknown). The communication between two scenes are achieved though the gene names. If a gene node is selected in the network, it is easy to locate it in the profiling data, and vice versa. The communication is important for understanding the network behaviors. The user may want to know its expression profile when he is looking at a gene node in the network. The visual representation the expression profile will be brought to the front of the user. He may also want to see the reactions the gene takes part in (Section 4.2) if its expression profile is very special. The interactions among the user, the metabolic network, and gene expression profiling data are illustrated in Figure 3.5. The fan layout of reactions of interest, the radial layout of reactions of interest, and the layout animation are discussed in Chapter 4.
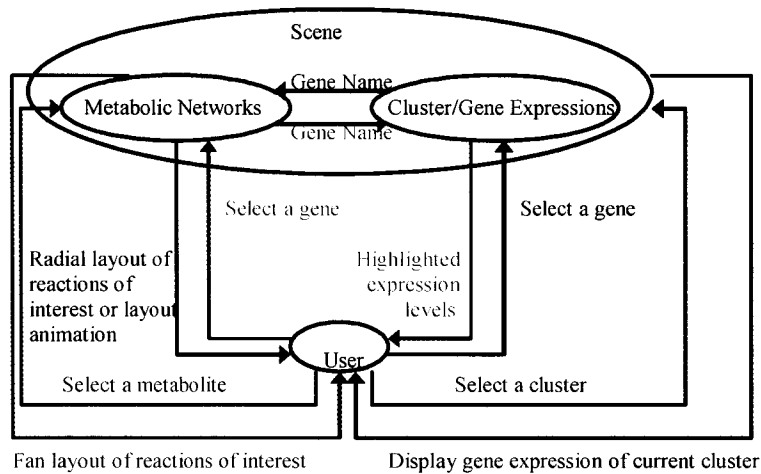
Figure 3.5 Interactions among users, metabolic networks and gene expressions

3.5 Discussion

This chapter introduces the visual representations for metabolic networks and gene expression profiling data, the weighted GEM-3D layout, and how to combine two scenes in virtual reality.

3D Shapes and material properties in computer graphics, such as colors, texture maps and transparency of geometric objects, can reveal useful information in metabolic networks, such as molecule node type, reaction edge type, and molecule names, more effectively than 2D flowchart-like displays. In the 3D plot representing gene expression profiling data, shapes (the connected lines) and colors are used together to indicate the changes of expression levels. The color changes consistently with the height. The color and the height change continuously, while the color map method shows the change of expression levels discretely. 3D space gives the plot more room so that it is possible to add labels for each gene and descriptions for each expression level.

Stereoscopic VR supplies an effective platform to network visualization. Ware's studies [Ware et al. 1993; Ware and Franck 1994] found that graphs (with a 3D layout) viewed in stereoscopic VR can be three times as large, in terms of the number of nodes, as graphs viewed in a 2D plane. VR features tracked 3D input devices such as wands to enable the navigation through two scenes and selecting along the device orientation. Navigation through the scenes is actually the rendering of scenes from continuous viewpoints implemented by translating and rotating the scenes. VR also features head tracking, which enables the movements and rotations of the viewpoint in a small range. The 3D layout, navigation, and stereoscopic display help to increase the size of understandable networks. For example, two edges may look like they cross each other from one viewpoint in a monoscopic display. The apparent crossing will disappear when

viewed from another viewpoint, or the depth difference shown by the stereoscopic display helps distinguish the edges.

Heading tracking in VR also helps to the combination of metabolic networks and gene expression profiling data. It detects the current position of the viewpoint. The expression profile of a gene node can be brought to the position when the user (viewpoint) is looking at the same gene node in the network.

It is a time-consuming task to render large scenes in computer graphics. Multiple techniques are used to speed the rendering (and also improve the image quality), such as level of detail, billboarding, and mipmapping, in order to achieve an interactive rate for large scale metabolic networks in available hardware. The fast development of the low-cost graphics rendering hardware also popularizes the adoption of computer graphics. Another advantage of computer graphics comes from its vector-based essential. When the scene is rendered in a 2D window, it has a better quality for the window size scaling than bitmap-based drawings.

For an effective visualization, it is necessary to interactively change the visual metaphors, for example, the color for metabolite nodes. The change of shapes, colors, and transparency applies no performance penalty in the current rendering hardware. The change of texture maps is an exception. However, there is no need to change them since they indicate node names.

CHAPTER 4.   EXPLOITING HIERARCHICAL RELATIONSHIPS IN METABOLIC NETWORKS

4.1 Compound graph model for metabolic networks

Besides adjacency relationships among molecule nodes, many metabolic networks contain some hierarchical relationships. For example, molecules in a network can be grouped according to their location within the cell or sub-cellular compartments. One compartment may contain multiple molecules. Molecules that appear in different compartments are modeled as different nodes. These hierarchical relationships show different levels-of-detail in the metabolic network (Figure 4.1). A graph model containing both adjacency relationships and hierarchical relationships is called a *compound graph* (Sugiyama and Misue 1991). The compound graph model for the metabolic network with compartment information is $\{G(N, E), T(N_t, E_t)\}$. $G(N, E)$ is a directed graph. N is the set of nodes representing various molecule nodes in the network. E is the set of reaction edges connecting nodes. $T(N_t, E_t)$ is a free tree (a tree without a root). N_t is the union of N and the set of nodes representing various compartment nodes. E_t is the set of inclusion edges representing the hierarchical relationships between compartment nodes and molecule nodes.

Metabolic networks may also contain some quasi-hierarchical relationships. A quasi-hierarchical relationship is a loose hierarchical relationship where a child may belongs to more than one parent in the hierarchy. For example, a metabolic network may include several pathways. A pathway contains molecules. One pathway involves many molecules. Some molecules take part in more than one pathway. A quasi-hierarchical relationship converts to a hierarchical relationship using duplication. In Figure 4.2(a), molecule 'A' appears in two pathways. In Figure 4.2(b), 'A' is duplicated so that each

pathway has a copy. There is a duplicate edge between two copies of 'A'. The duplicate edges are treated the same as the adjacency edges in the detail-on-demand method discussed in the next section.
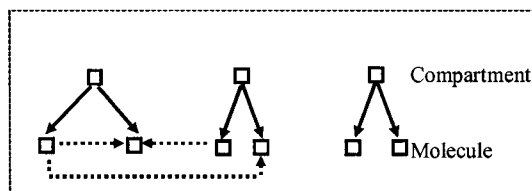
(a) The compound graph model {G(N, E), T(N$_t$, E$_t$)}
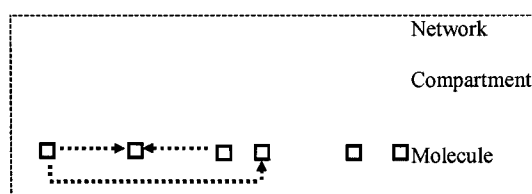
(b) G(N, E)
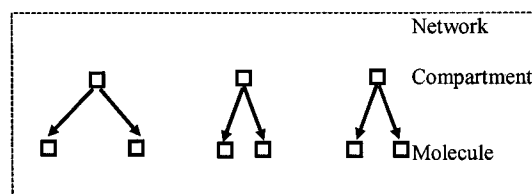
(c) T(N$_t$, E$_t$)
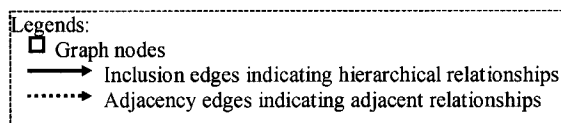
Figure 4.1 A compound network contains hierarchical and adjacency relationships.
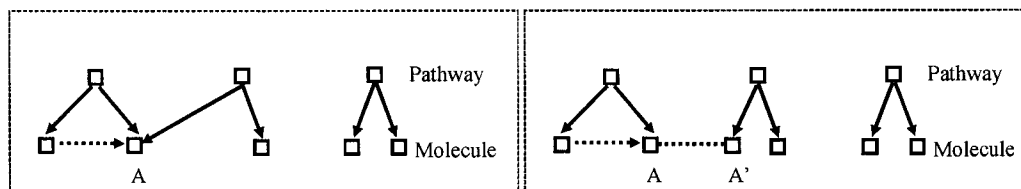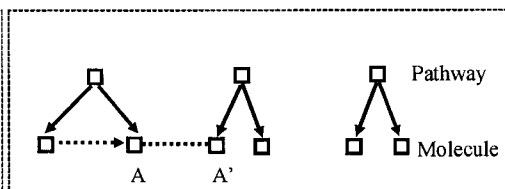
(a)   (b)

Figure 4.2 Node duplication converts a quasi-hierarchical relationship to a hierarchical

relationship

A metabolic network is called a *compound network* if it is represented as a compound graph. It is called a *standard network* if represented as a (normal) directed graph. A compound network is reduced to a standard network if all pathway (or compartment) nodes and hierarchical relationships are removed.

4.2 Detail-on-demand visualization method

4.2.1 Methodology

Displaying the whole graph representing a large standard network shows the overall structure of the network. The drawback is that details such as node and edge names are too small to be seen. Zooming into a part of the network and panning to other parts show local details but make it difficult to see the global structure. The detail-on-demand method uses hierarchical relationships in a compound network to allow users to dynamically change the visual level-of-detail. The method helps users to understand both the global relationships and local details of a large network simultaneously, which was observed as key to successful information visualization [Risden et al. 2000]. Figure 4.3 shows how the detail-on-demand method reduces the number of displayed nodes and still maintains the correct relationships in a compound network.

A snapshot of a compound network, $\{G(N, E), T(N_t, E_t)\}$, is another compound network $\{G_s(N_s, E_s), T_s(N_{st}, E_{st})\}$. $G_s(N_s, E_s)$ is a directed graph. N_s is the node set including two types of nodes:

Nodes of Type 1. Molecule nodes in expanded compartments (or pathways)

Nodes of Type 2. Unexpanded compartment nodes

E_s is the edge set including three type of edges:

Edges of Type 1. Edges connecting nodes of type 1

Edges of Type 2. Edges connecting nodes of type 1 and nodes of 2

Edges of Type 3. edges connecting nodes of type 2
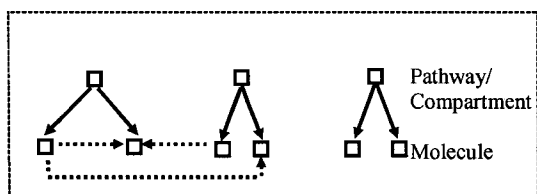
(a) A compound network contains hierarchical and adjacency relationships.
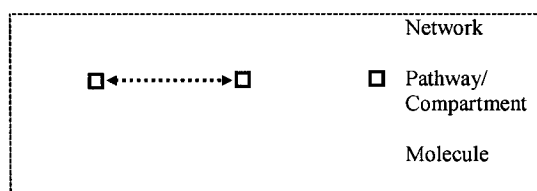
(b) The snapshot in the level of least details. Adjacency edges in more detailed levels are

converted to this level.
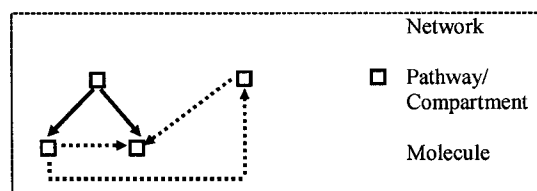
(c) Expand the leftmost pathway (or compartment) node in (b), the adjacency edges are

converted to the currently deepest level accordingly
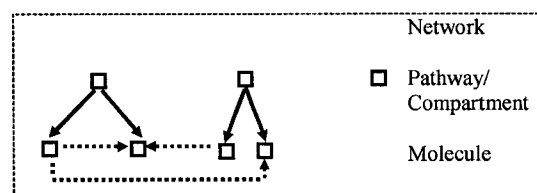
(d) Expand one more node

Figure 4.3 The detail-on-demand method reduces the number of displayed nodes and still

maintains the correct adjacency relationships in a compound network.

An edge of Type 2 (an edge between one molecule node and one unexpanded

compartment node) exists if there is at least one edge in the original compound network

between the molecule node and one molecule node belonging to the compartment. An

edge of Type 3 (an edge between two unexpanded compartment nodes) exists if there is at least one edge in the original compound network between two molecule nodes belonging to the two compartments respectively. A property called *edge density* is added for any edge of Type 1 or 2 indicating the number of edges between molecule nodes it represents.

$T_s(N_{st}, E_{st})$ is a free tree (a tree without a root). N_{st} is the set of expanded compartment nodes. E_{st} is the set of inclusion edges representing the hierarchical relationships between expanded compartment nodes and molecule nodes in them.
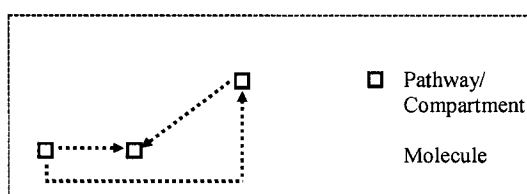
(a) The directed graph mode $G_s(N_s, E_s)$ in the compound network $\{G_s(N_s, E_s),$

$T_s(N_{st}, E_{st})\}$ representing a snapshot in Figure 4.3(c)

(b) A tree model $T_s(N_{st}, E_{st})$ in the compound network

Figure 4.4 The compound model representing a snapshot

A snapshot is generated out of the original compound graph according to the definition and the current status (expanded or unexpanded) of the compartments. The snapshot with all compartments expanded is the same as the original compound graph.

4.2.2 Visual representation for compound networks

Unexpanded compartment (or pathway) nodes are initially represented as translucent cubes with their names on the each surface (Figure 4.5(a)). The size of a cube is proportional to the number of molecule nodes it contains. Once a compartment node is

expanded, hierarchical relationships are represented as a geometric inclusion in 3D space, i.e., the cube size increases; all the visual representations of molecules belonging to it appear; and the molecules are bounded by the compartment node. The cube can turn off for a clear view of molecules inside (Figure 4.5(b-d)).

An edge between two compartments or between a compartment and a molecule is represented as a cylinder. The edge density is visually represented by the radius of the cylinder (Figure 4.5).

4.2.3 Detail-on-demand interactions

Detail-on-demand interactions are enabled by an abstract interface for a tracked wand-like input device in VR, which supports both real and simulated wands. A ray (the red line in Figure 4.4) is cast from the wand (the green gadget in Figure 4.4) according to its position and orientation. When the ray intersects with the visual representation of a pathway (or compartment), a button click expands the pathway. When the ray intersects any one of molecules in the expanded pathway, a button click will shrink the pathway back into the pathway node.

Figure 4.5 shows some snapshots in the exploration of a network in Arabidopsis from the MetNet Database (Wurtele, Dickerson et al. 2003) using detail-on-demand interactions. The video clip that shows the exploration is available at (http://www.vrac.iastate.edu/research/sites/metnet/Thesis/Yuting/Video/Chapter-4/Video 4-1.mpg). There are 572 molecules and 648 reaction edges among molecules in this network. The molecules in the network belong to the following sub-cellular compartments: plastid, mitochondrion, nucleus, cytosol, plastid stroma, and unknown. The network contains three pathways, Acetyl-CoA biotin, starch degradation, and starch synthesis.
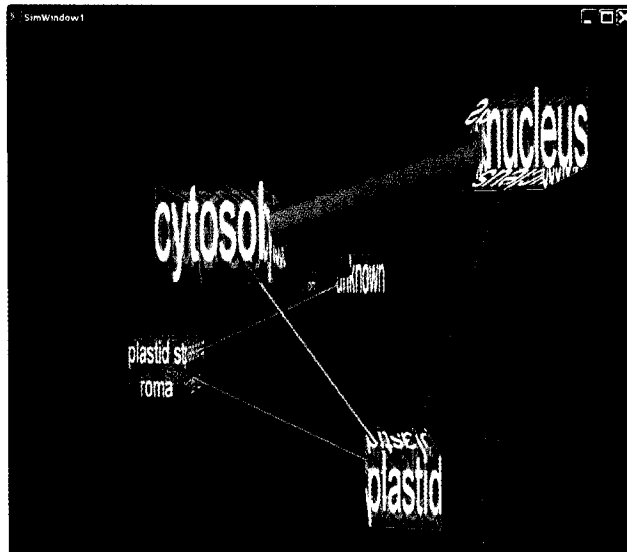
(a) A network whose molecules in the network belong to six compartments
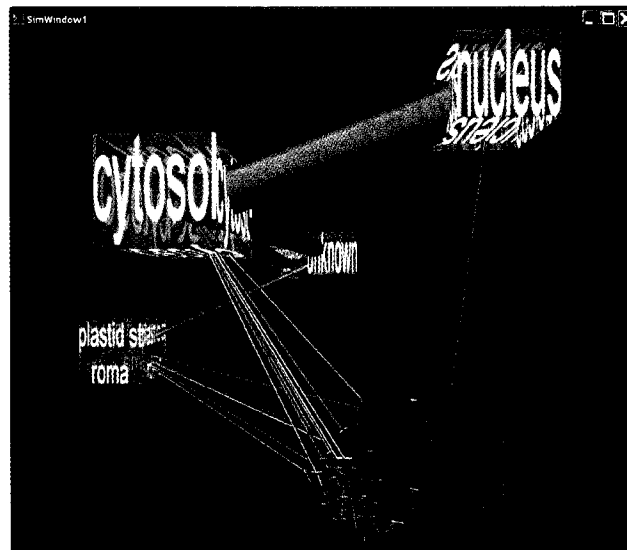
(b) Expand the plastid compartment. After the expansion, the representation of plastid

(a translucent boundary box) is turned off to show the nodes and edges inside.

Figure 4.5 Snapshots illustrating the exploration of a metabolic network using

hierarchical relationships and detail-on-demand interactions.
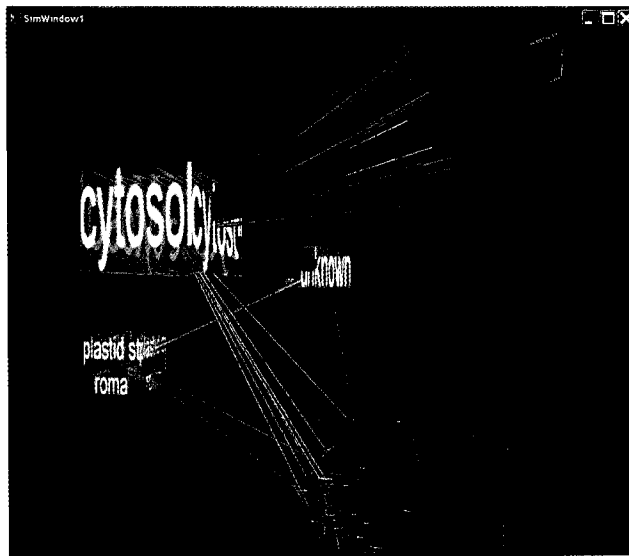
(c) Expand the nucleus compartment.

(d) Expand the plastid stroma compartment.

Figure 4.5 (continued)

4.3 Layout algorithm for compound networks

4.3.1 Existing algorithms

Even in 2D space, there are few works on the layout of compound graphs [Sugiyama and Misue 1991; Sander 1995; Eades and Feng 1996; Eades et al. 1996; Sander 1996; Eades and Huang 2000] , probably due to the difficult nature of the problem. Most of them [Sugiyama and Misue 1991; Sander 1995; Eades et al. 1996; Sander 1996] converted a cluster node to one or multiple layers and applied layered layout algorithms to the converted graphs.

The 2D layered layout algorithms first reverse the directions of a set of edges to make the graph temporarily acyclic (for assigning nodes to different layers in the next stage). It is a NP-complete problem to find such a set of edges, the reversion of whose directions makes the graph acyclic, as small as possible. In the second stage, nodes are assigned to layers, which correspond to parallel equally-spaced lines in the final layout. It is also a NP-complete to find a layered partition of nodes that satisfies the constraints of same oriented edges directions (e.g. downwards) with the bounded drawing area. In the third stage, nodes are ordered in each layer. It is NP-complete to find an order of nodes to minimize the number of edge crossings between two consecutive layers. In the final stage, the positions of nodes in each layer are defined. There are different heuristics for these NP-hard problems [Sugiyama et al. 1981].

When extending the 2D layered method to 3D space [Ostry 1996], the barycentric heuristic for finding the order of nodes in each layer in 2D space does not work. The barycentric heuristic calculates the position of a node in one layer as the bary center of its adjacent nodes in the consecutive layer. There is a primary layer, where the node positions are arbitrarily assigned. The barycentric heuristic causes collisions when two or more nodes in one layer have the same set of adjacent nodes in the consecutive layer. In

2D space, the collision can be resolved by separate the colliding node to a predefined distance in the one dimensional layer. In 3D space, the separation is in a two dimensional plane and an arbitrary separation direction may cause further collisions. Ostry resorted to the force directed method for the node position in each layer, using the layout generated by barycentric heuristic as initial embedding.

[Ware et al. 1997] implemented a 3D layout based on the layered method for visualizing complex software structures, which contain hierarchical relationships, like the inclusion of multiple classes in a single source file, and adjacency relationships, like the method calls among different classes. The layered method is recursively used in different hierarchical levels. The longest path layering is used in the assignment of layers to nodes in different hierarchical levels, which layers nodes in the breadth first order. The force directed model is used to taking into consideration the edge connecting nodes nested within one node to nodes nested within another node in layering. Each layer is further divided into a grid. The nodes in the layer are initially located in the random grid points. The force directed method is used to move the nodes to different grid points to reduce the edge crossings between two consecutive layers.

[Eades and Huang 2000] introduced a force directed model for the compound graph layout in 2D space. There is a force for each node in one cluster to push it toward the cluster node and there are different weights for the attractions of edges within a cluster and for the attractions of edges crossing clusters. User interventions are needed to avoid the overlapping among different clusters.

[Demir et al. 2002; Dogrusoz et al. 2004] proposed a layout method for the compound graph, which represents nesting relationships such as molecular complexes and pathway abstractions in 2D space. The force directed method is recursively used in different hierarchical levels.

4.3.2 3D force directed layout algorithm for compound graphs

Layered methods are not adopted due to two reasons. The existing heuristics for the NP-complete problems may not generate satisfactory results for the layout of metabolic networks. For example, the longest path layering is used in the assignment of layers, which layers nodes in the breadth first order. Metabolic networks may contain very long paths, generating too many layers. It may also generate edges crossing multiple layers. When extending from 2D space to 3D space, the existing heuristics for the NP-complete problems may be not enough to generate a solution. For example, the barycentric heuristic has to work together with a force directed model to reduce the edge crossing between two consecutive edges in the stage of deciding the nodes positions in a layer.

The layout method proposed here for a compound network $\{G(N, E), T(N_t, E_t)\}$ combines the hierarchical force model with the simulated annealing method of the GEM-3D algorithm[Bruss and Frick 1995], which is most effective to accelerate the convergence [Brandenburg et al. 1995]. The pseudo code of the algorithm is available at Appendix A. Below are the descriptions about the algorithm.

In the initial embedder, a randomized position is used for each node that belongs to N (i.e. leaf nodes in T). The barycenter of its children in T is used for each node that belongs to $N_t - N$ (i.e. non leaf nodes in T, or cluster nodes).

The algorithm calculates the layout in a loop. In each round of the loop, all nodes update their positions once according to the hierarchical force model. The loop will stop in an equilibrium state where the total force on each node is zero or when the rounds of the loops beyond a predefined number, which is usually the number of nodes.

The hierarchical force model includes the following items,

- Node Repulsion. For any node, i, belonging to N, there is a repulse between i and any other node, j, belonging to N. The repulse is calculated as $\Delta/|\Delta| \times L$, where Δ is the vector from the current position of j to the current position of i, $|\Delta|$ is the length of the vector, and L is the desired edge length.

- Cluster Repulsion. For any node, i, belonging to N_t - N, there is a repulse between i and any other node, j, belonging to N-N_t. The calculation of the cluster repulsion compares the distance of i and j, $|\Delta|$, with the sum of the longest distance of i to its children in T, L1, and the longest distance of j to its children in T, L2. If $|\Delta|$ is less than the sum of L1 and L2, there is a repulse calculated as $\Delta/|\Delta| \times (L1 + L2)$ so that the overlapping between i and j can be avoided.

- Adjacency Attraction. For each edge of i, e_{ij} or e_{ji}, belonging to E in G(N,E), there is an attraction between nodes i and j. The computation of the attraction is according to the equation of $-w \times \Delta \times |\Delta|/(L \times \Phi(i))$, where Δ is the vector from the current position of j to the current position of i, $\Phi(i)$ is a function that linearly grows with the degree of i, and w is a weight factor. The value of w is larger when the parent of i is the same as the parent of j (i.e., i and j belong to the same cluster) than the value of w when the parent of i is different from the parent of j.

- Hierarchy Attraction. For any node, i, belonging to N, there is an attraction between i and its parent, j, in $T(N_t, E_t)$. For any node, i, belonging to N_t-N, there is an attraction between i and each of its children, j, in $T(N_t, E_t)$. The attraction is calculated according to the equation of $-w \times \Delta \times |\Delta|/(L \times \Phi(i))$, where Δ is the vector from the current position of j to the current position

of i, $\Phi(i)$ is a function that linearly grows with the degree of i, and w is a weight factor.

- Gravity. For any node, i, belonging to N_t, there is a gravity toward the barycenter of the current layout, c. The gravity is calculated as $w \times \Delta$, where Δ is the vector from the current position of i to the current position of c, and w is a weight factor.

- Brownian motion. For any node, i, belonging to N_t, There is a random impulse (x, y, z). x, y, z are random values between -L/2 and L/2, where L is the desired edge length.

The same simulated annealing method of as in the GEM-3D algorithm[Bruss and Frick 1995] is used to accelerate the convergence to the equilibrium stage. Nodes only move if the movement will result in a reduction in energy. Each node has a temperature. The initial temperature is high. The impulse of a node is first scaled by its current temperature and then used to update its position. If the current impulse and the last impulse of a node are in opposite directions, oscillations occur. The temperature is lowered to depress the oscillation. If the current impulse of a node is always perpendicular to its last impulse and the angles always have the same direction, rotations occur. The temperature is lowered to depress the rotation.

For a given snapshot, $\{G_s(N_s, E_s), T_s(N_{st}, E_{st})\}$, molecule nodes in the expanded compartment (or pathway) nodes use the positions of corresponding molecule nodes in $G(N, E)$ in the original compound graph; unexpanded compartment nodes use the positions of corresponding compartment nodes in $T(N_t, E_t)$ in the original compound graph. Figure 4.6 shows the whole procedure for interactive visualization of compound metabolic networks.
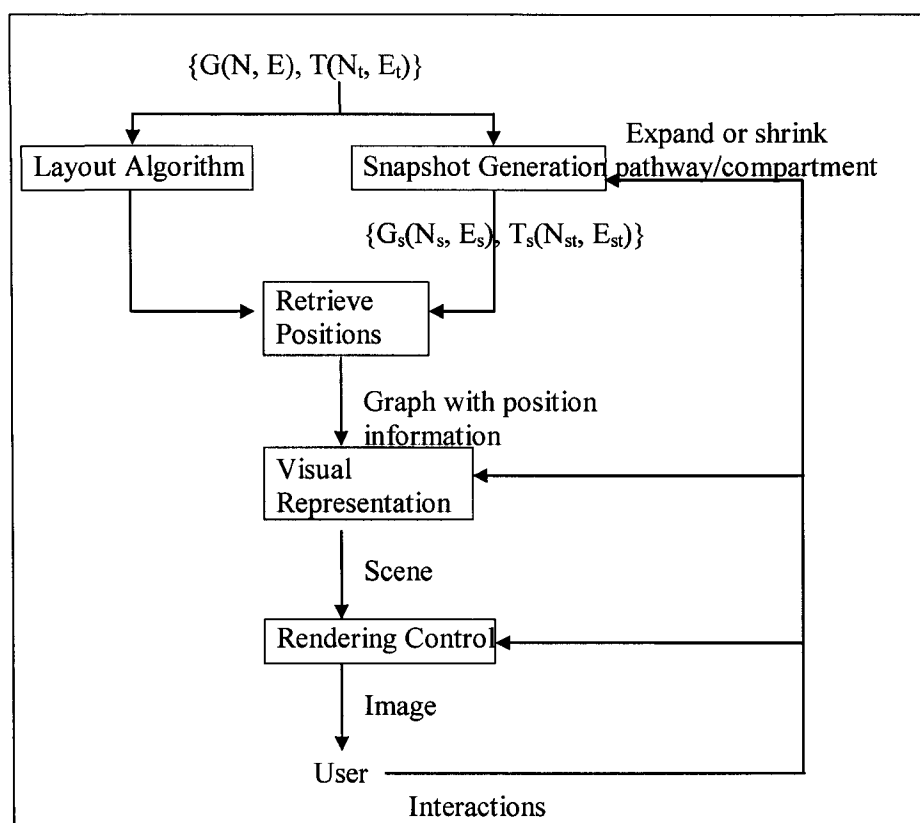
Figure 4.6 Data flowchart of interactive visualization of compound networks

4.4 Discussion

This chapter presents a hierarchical visualization method for metabolic networks. The hierarchical relationships in the metabolic networks construct different levels-of-detail, which can be used to reduce the number of displayed nodes and edges and still maintain the correct relationships in large scale metabolic networks. It is easier for users to perceive when less information is displayed at one time.

The metabolic networks containing hierarchical relationships are models a compound graph model $\{G(N, E), T(N_t, E_t)\}$. $G(N, E)$ is a directed graph model describing the adjacency relationships among molecules. T is a tree describing the inclusion relationships between molecules and pathways or between molecules and compartments. A snapshot of the compound graph model indicates the network at a

specific level-of-detail, where some of pathways are expanded, i.e., molecules in these pathways appear, and other pathways are unexpanded, i.e., only pathway nodes appears. The snapshots can also be models as a compound graph model $\{G(N_s, E_s), T(N_{st}, E_{st})\}$. The algorithm to generate a snapshot from the original compound graph is straightforward from the definition and its time complexity is $O(|N_t| + |E|)$.

Instead of recalculating the layout for each snapshot, the layout calculation only occurs once for the snapshot with all pathway (or compartment) nodes expanded, i.e., the original compound graph. The layout algorithm combines the hierarchical force model with the simulated annealing method of the original GEM-3D algorithm[Bruss and Frick 1995], which is used to accelerate the convergence. The hierarchical force model can also apply to the multiple-level hierarchical structure. In metabolic network, only three levels exist, i.e. the network, the pathways or compartments, and the molecule nodes. The time complexity of the layout algorithm is $O(k*|N_t|^2)$, where k is the number of rounds to converge. For a given snapshot, $\{G_s(N_s, E_s), T_s(N_{st}, E_{st})\}$, molecule nodes in the expanded pathway nodes use the positions of corresponding molecule nodes in $G(N, E)$ in the original compound graph; unexpanded pathway nodes use the positions of corresponding compartment nodes in $T(N_t, E_t)$ in the original compound graph. Besides the efficiency, this method also keeps a better mental image for users. Except for the node being expanded or shrunk, all other nodes stay in their previous positions. Two expansions of one pathway node are guaranteed to have the same internal layout.

The detail-on-demand method exploits the different levels-of-detail in metabolic networks dynamically. The network is initially displayed in the least level-of-detail. All pathway (or compartments) nodes are unexpanded. The user can select any pathway of interest and trigger the expansion. He can also select any molecule in the expanded pathway and trigger the shrinkage. Selecting and triggering are called detail-on-demand

interactions. In virtual reality, detail-on-demand interactions are enabled by a position-tracked wand-like input device. It is very comfortable to point the wand towards a node and press a button on the wand to trigger the expansion or shrinkage of the selected node.

CHAPTER 5.   REACTIONS OF INTEREST

The global layout reveals the overall structure of the metabolic network to users. Users may also be interested in the reactions that a specific molecule node takes part in. The term *Reactions of interest (ROI)* is used to call the set of all reactions (in the network in study) that a molecule takes part in. The molecule node is called the focus node. Given a focus node, ROI is automatically extracted from the whole network. The abstraction is done by exploiting some uniform properties of metabolic reactions, i.e., the substrates (metabolite nodes) are on one side of the reaction node; products (also metabolite nodes) are on the side of the reaction node; and genes, RNA, polypeptides, and protein complex form a quasi-tree rooted at the reaction node (Figure 3.1). A quasi-tree is a loose defined tree where a node may have more than one parent. For example, a polypeptide may assemble more than one protein complex in a reaction. A depth-first search [Cormen et al.] starting from the reaction node extracts the quasi-tree. The substrates and products are retrieved through the incoming and outgoing enzymatic edges the reaction node (A enzymatic edge is either from a substrate node to a reaction node or from a reaction node to a product node).

5.1 Layout for reactions of interest from the network of enzyme-catalyzed reactions focusing on a metabolite

The ROI focusing on a metabolite is the set of metabolic reactions the metabolite takes part in. A layout algorithm, called a *fan layout*, is proposed to draw the ROI. In the fan layout, the reactions are around the focus node (Figure 5.1), like leaves of a fan. Each reaction is drawn following the conventions that all substrates are on one side of the reaction node, all products are on the other side, and all other nodes form a quasi-tree rooted at the reaction node (Figure 3.1). The conventions are for an isolated reaction. A

node (other than the focus node) may take part in N ($N>1$) reactions in a ROI, causing a conflict between the drawing of N isolated reactions. To resolve this conflict, the node splits into N copies. Each reaction has a copy of the node. This procedure is called a *node split*. The algorithms for the fan layout are available at Appendix B.

The fan layout shows how the focus node participates in metabolic reactions as shown in Figure 5.1. The fan layout can be rotated so that at each time there is a single reaction facing the user, giving a better view of each reaction. Choosing a node other than the current focus node in the fan layout generates another ROI. A series of ROIs let the user navigate through a pathway.
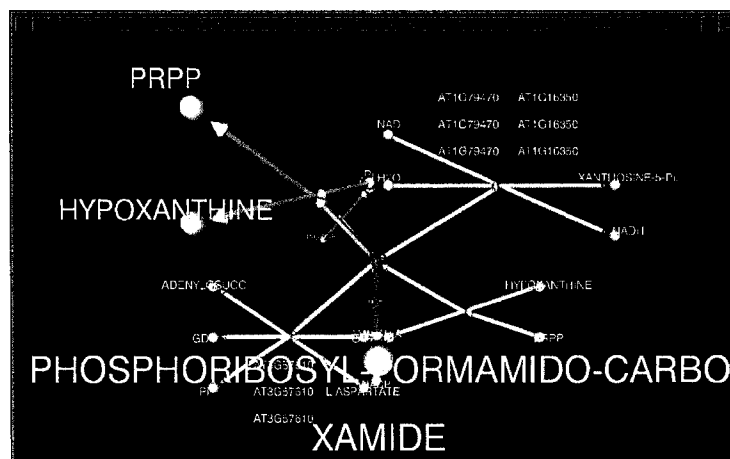
(a) A fan layout of reactions of interest focusing on 'IMP' (the node in the center with a red label)

Figure 5.1 Fan layout
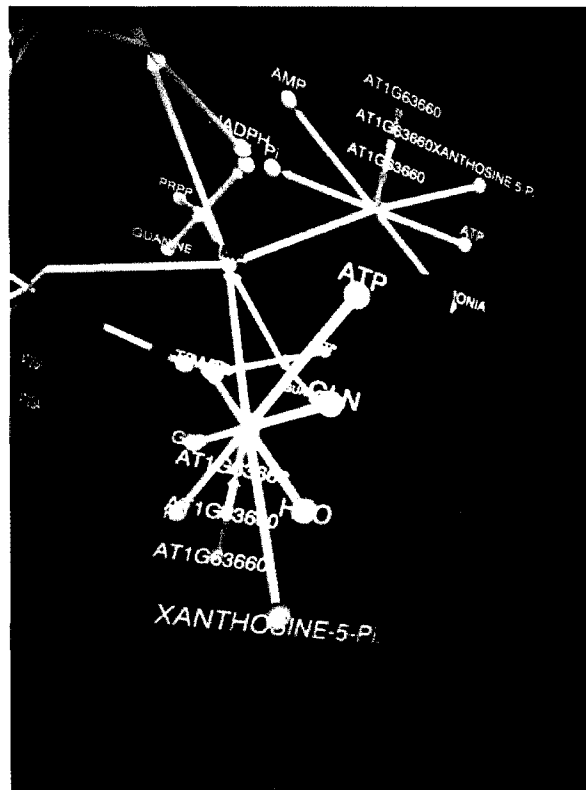
(b) Users discuss the reactions of interest focusing on 'GMP' in a virtual environment

Figure 5.1 (continued)

5.2 Layout for reactions of interest from the network of enzyme-catalyzed reactions focusing on a gene

A key role of genes in metabolic networks is to encode proteins that catalyze reactions. In the graph model, the edges go out from the genes to the reactions that their encoded proteins catalyze. This property makes the radial layout [Eades 1992; Di Battista et al. 1999; Herman et al. 2000] applicable for the ROI focusing on a gene. The radial layout arranges all the children nodes on a circle around the parent node. In the ROI focusing on a gene, the radial layout algorithm arranges the reaction nodes on a circle around the gene (Figure 5.2). The radial layout shows which reactions a gene product catalyzes. Drawing radial layouts centered on different genes side-by-side helps to

compare these reactions and visualize any differential regulation of the catalysis. Rotating the radial layout along the center of the circle, such that at each time one reaction is horizontal, gives the viewer a better view of each reaction. The procedure of node split (Section 5.1) is needed before applying the layout method to avoid edge crossings. The algorithms for the fan layout are available at Appendix C.
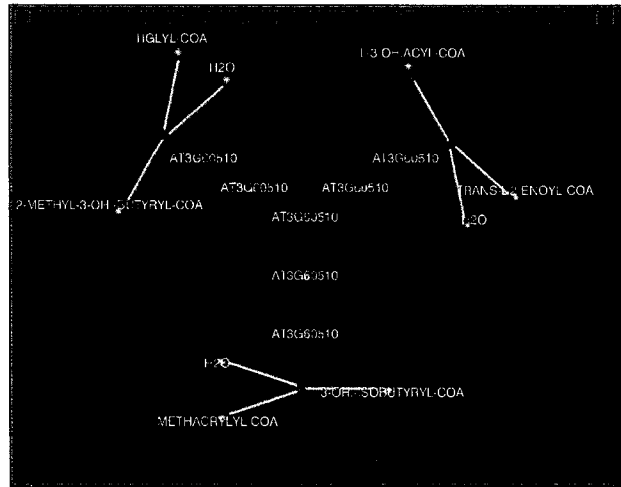
(a) A Radial layout of the reactions of interest focusing on gene 'AT3G60510'
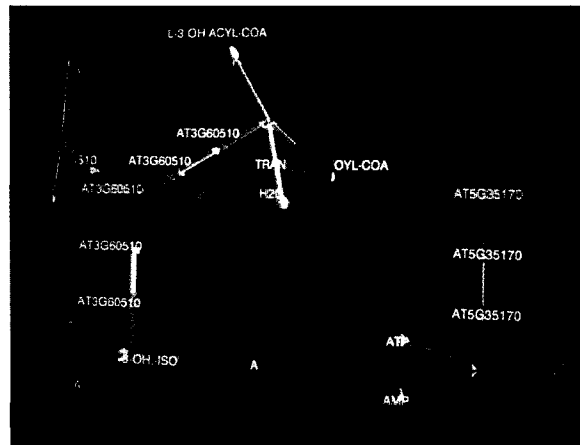
(b) Users compare the reactions of interest focusing on gene 'AT5G35170' and gene 'AT3G60510'; these two genes have similar expression profiles

Figure 5.2 Radial Layout

5.3 Layout Animation

Layout animation changes the position of a specific portion of the network while keeping the rest unchanged. The goal of the layout animation is to highlight specific areas of the metabolic network such as the reactions of interest, or specific pathways. Step by step animation, instead of a sudden change, helps users to keep their mental map of the network intact.

The key points in designing layout animation include the initial layout of the whole pathway network, the final layout of the sub-network, and the interpolation between the initial position and the final position of the sub-network. The weighted GEM-3D method calculates the initial layout for the whole network and the final layout of the sub-network. The final layout of the sub-graph is translated in front of the user. The animation is a simple linear interpolation between the positions of the nodes of the sub-network in the initial layout and their positions in the final layout. The edges between the nodes either stretch or shrink according to the node positions.

The video clip of an animation, which is available at (http://www.vrac.iastate.edu/research/sites/metnet/Thesis/Yuting/Video/Chapter-5/Video 5-1.mov), shows how 'AT2G22190' the putative gene for trehalose-6-phosphate phosphatase fits into the known pathways in the MetNet database [Wurtele et al. 2003]. The animated portion pulled out of the main graph consists of all the reactions that gene AT2G22190 takes part in from the pathways: trehalose biosynthesis and trehalose degradation.

5.4 Discussion

This chapter describes some methods to explore the local details of a metabolic network, including two layouts and a layout animation method. Together with the automatic extraction of ROIs, they constitute query functions, by which users can easily

find information of interest. Query functions are very necessary for the study of large scale networks. The global layout is good to reveal the overall structure of a metabolic network. To find the details of an interesting molecule, one way is to navigate through the network. A more efficient way is to search the node and let the system bring out the detailed information. The user interface to search a node according to its name is Section 7.1.

Two layouts are tailor-made for the ROI focusing on a metabolite and the ROI focusing on a gene respectively. Addition scenes are generated for ROIs. Users can navigation through and selected nodes from the scenes. Since ROIs have less number of nodes and edges, it is easy to understand and compare them.

Layout animation pulls out the interested sub-network and also keeps their relationship with the other part of the network. Some research has focused on the interpolation for graph animation [Friedrich and Houle 2001; Friedrich and Eades 2002]. However, these algorithms apply to complex position changes in the whole graph or to different clusters in a graph. They are much more complex and computationally intensive than is needed for this application. In this research, only a small part of the network is pulled out (and then withdraws to the original position for the turn of the next sub-network) and the majority of the network is kept stationary to keep the user's mental model of the network intact. It is not necessary to apply a complex animation method to the whole network, which usually contains more than 1,000 nodes.

CHAPTER 6.   INTEGRATION OF METABOLIC NETWORKS AND GENE EXPRESSION PROFILING EXPERIMENTAL DATA

Gene expression profiling experimental data reveal the dynamic behavior of its metabolic network when the organism is responding to external or internal stimuli. The experimental data provide a rough measure of the cellular accumulation of corresponding messenger RNAs of different genes.

Few studies have been done to combine gene expression profiling data and metabolic pathways. Detlef [Detlef et al. 2000] adds gene expression information into (static) KEGG pathway diagrams [Kanehisa and Goto 2000] by coloring the gene nodes according to their expression. Kurhekar [Kurhekar et al. 2002] monitors the changing of gene expression data with time by adding 'next' and 'previous' buttons into the KEGG pathway diagrams. More recently, Ryoko [Ryoko and Toshiyuki 2003] reports the results of adding color to gene nodes in the 2D dot layout. Dwyer [Dwyer et al. 2004] extends the method of Brandes [Brandes et al. 2003] to map metabolite profiling data to pathways. Instead of drawing related pathways in each level of the stack, the new method draws the time series of one pathway in each level. The size of the same metabolite node in each level is proportional to its profiling amount at a specific time corresponding to this level.

Chapter 3 introduces a method to construct an extra scene for the experimental data and spatially relate them to the scene for the metabolic network (Figure 3.5). Using 3D space and computer graphics, a tighter integration of these two types of data is possible. Section 6.1 presents a color morphing method, Section 6.2 presents a shape morphing method, and Section 6.3 presents an edge vibration method.

6.1 Color Morphing

The experimental data are captured using recent gene probe tools such as Affymetrix's Genechip® and formatted as a 2D array using the gene and the condition under which the expression level are captured as dimensions. It is intuitive to represent the different expression levels of a gene under different conditions temporally, i.e., show one expression level at one time. The data used in this research come from an experiment in carbohydrate metabolism that used Affymetrix ATH1 GeneChip® data from the Wurtele Lab (Li, Foster et al. 2004). In this particular experiment, leaves were sampled 12 times over a diurnal cycle. Each sampling point has an expression level. The value in between of two ending sampling points is the linear interpolation of them. If the experimental data are sampled temporally, the sampling intervals are used for interpolation; otherwise a fixed interval is used.

Humans are very sensitive to the change of colors. Using different colors for different expression levels highlights dynamic behaviors of the metabolic network. The color morphing will apply upon gene nodes, RNA nodes and the transcription edges between them. Two colors are defined for the highest level and the lowest level. The color for any other level in between is the linear interpolated color of them. The user interface to set two colors and control the color morphing is showed in Section 7.1.

Four snapshots of a video clip, which shows the color morphing of a metabolic network, are listed in Figure 6.1. The video clip is available at http://www.vrac.iastate.edu/research/sites/metnet/Thesis/Yuting/Video/Chapter-6/Video 6-1.mpeg. The network contains two pathways in Arabidopsis from the AraCyc Database (Mueller, Zhang et al. 2003): glycerol biosynthesis and glycerol metabolism. A clock is available to indicate time.
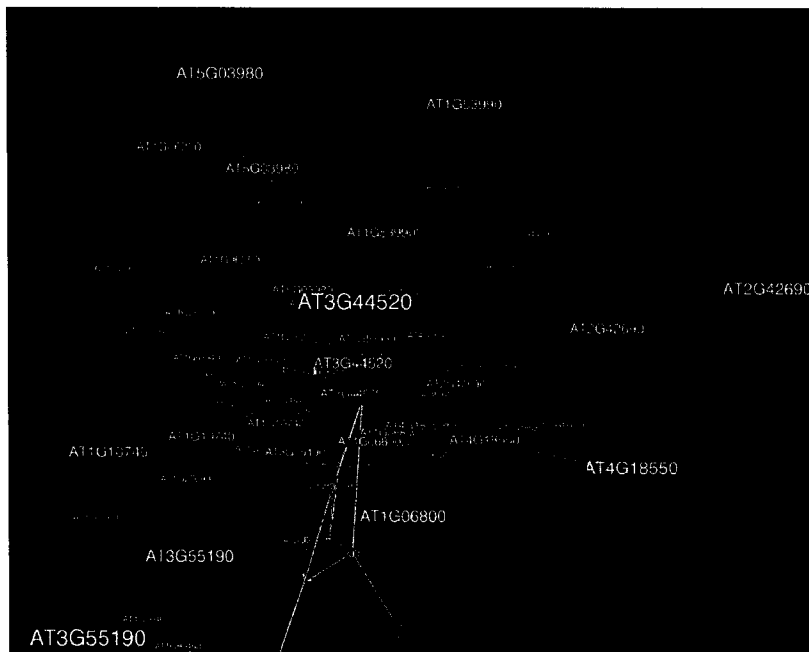
(a)
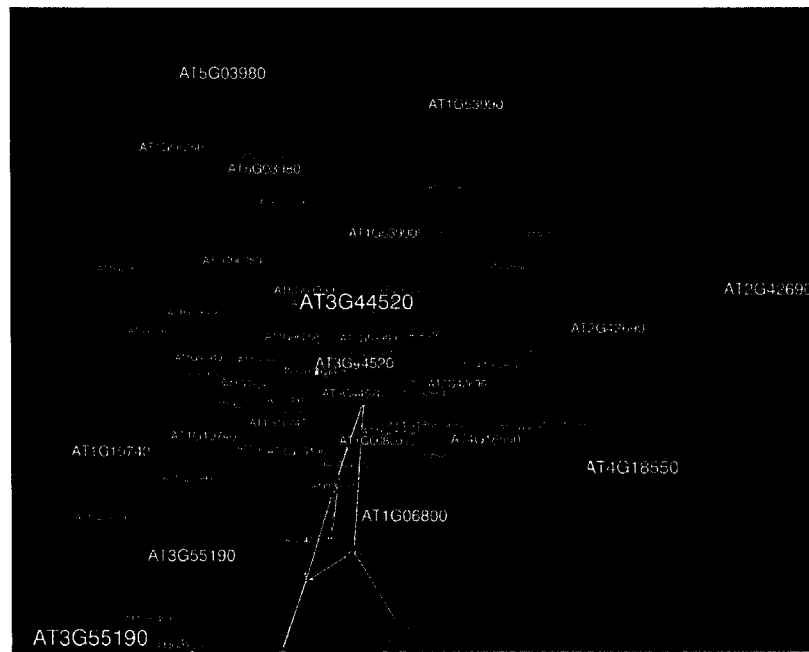
(b)

Figure 6.1 Color morphing of a metabolic network
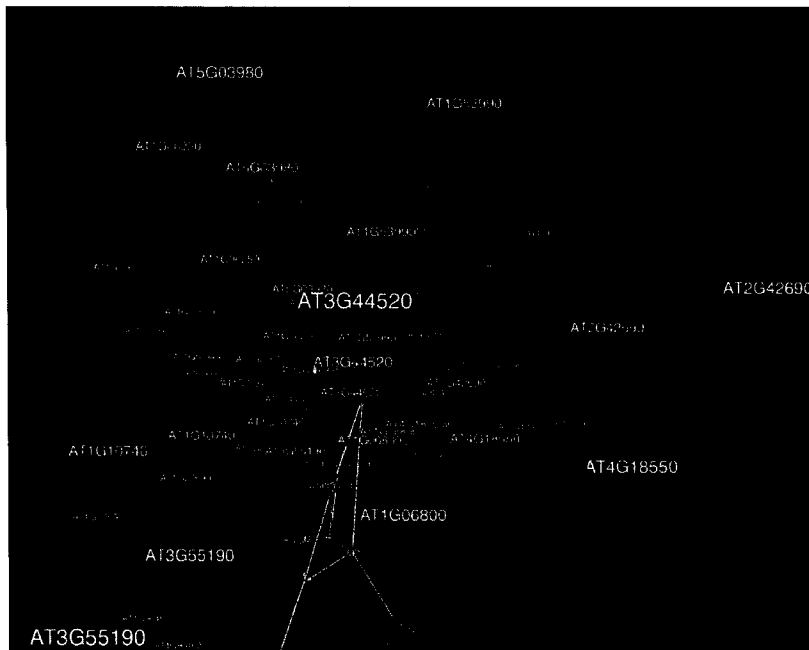
(c)
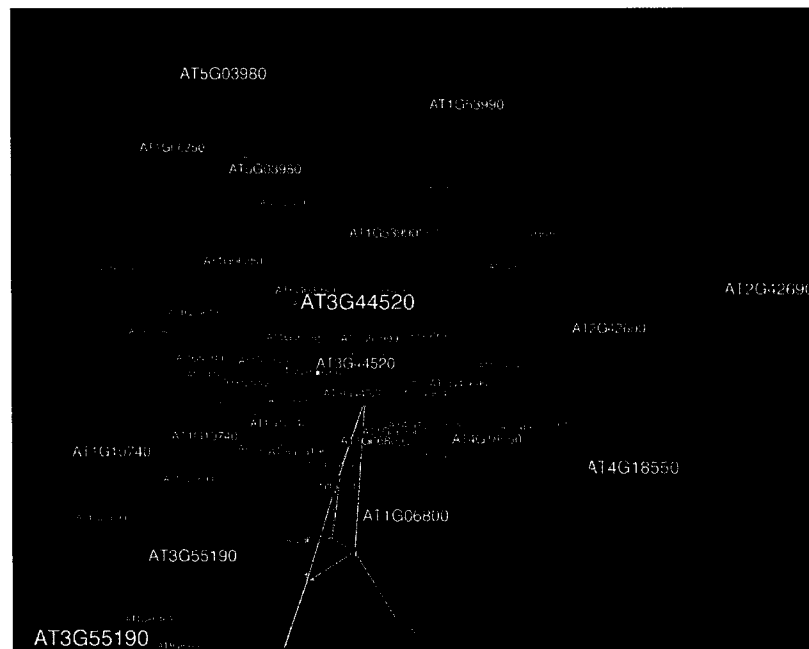
(d)

Figure 6.1 (continued)

Applying color morphing to the network shows its dynamic behaviors as a whole.

For example, the expression status of the network is relatively higher if the brightness of

the network is higher (Figure 6.2(a) and (b)), or vice versa (Figure 6.2(c)). Since other

types of nodes and edges do not contribute the brightness change, they may be turned off

to emphasize the brightness change. To look at the dynamic behavior of a specific gene,

users may stay in front of the node and enable the color morphing.
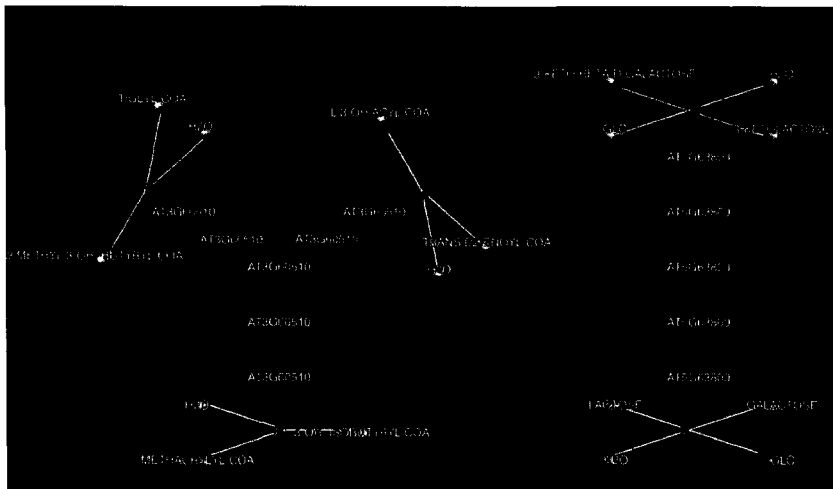
(a)
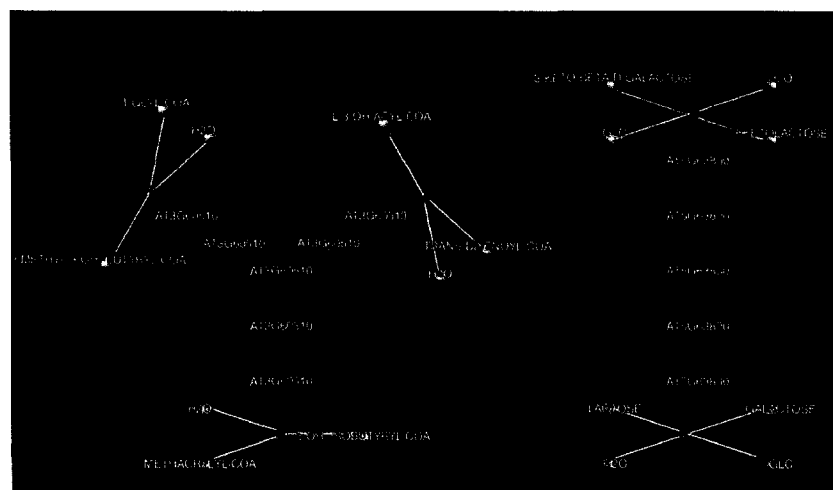
(b)

Figure 6.2 Color morphing of ROIs focusing on two genes in radial layout
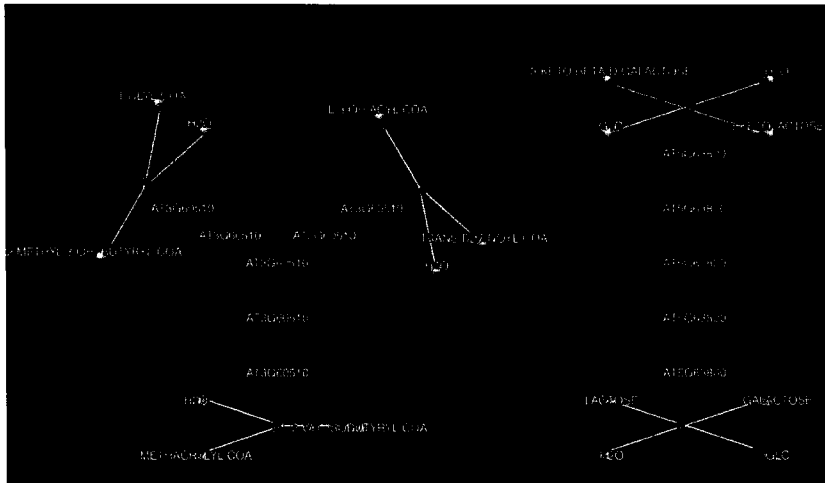
(c)
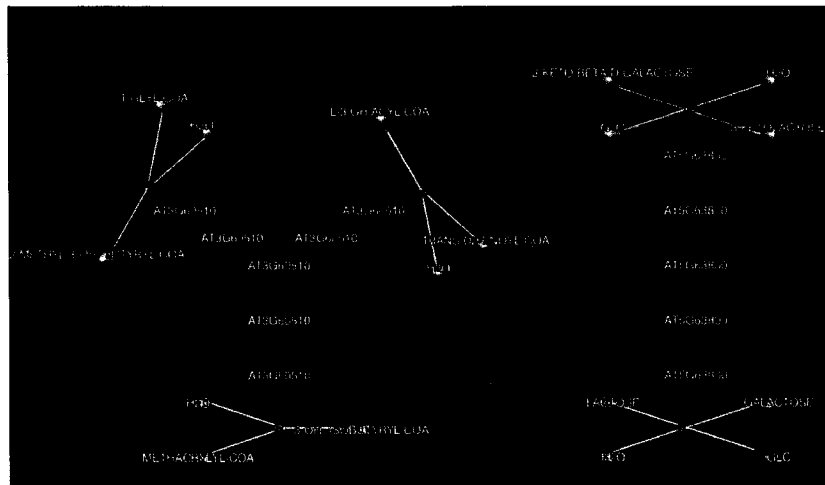
(d)

Figure 6.2 (continued)

A comparison of two genes about their expression status is of great interests for the network study. Applying the color morphing upon the reactions of interest in the radial layout (Section 5.2) makes the comparison possible. Figure 6.2 are the snapshots of a video clip, which shows the dynamic behaviors of two genes (AT3G60510 and AT5G63800) in a time series, along with the biological reactions they take part in according to a metabolic network map [Wurtele et al. 2003]. The video clip is available at

http://www.vrac.iastate.edu/research/sites/metnet/Thesis/Yuting/Video/Chapter-6/Video 6-2.mpeg. Figure 6.2 demonstrates that the expression status of AT3G60510 (the left one) declines earlier than AT5G63800 (The right one).

6.2 Shape Morphing

Shape is one of the important properties to convey information in 3D computer graphics. Shape morphing is good to reveal the dynamic information because human vision is sensitive to the changes. The dynamic behavior of a metabolic network can be visualized by using the shape morphing of edges in the scene for the network along the time. Figure 6.3 shows the shape morphing of ROIs focusing on 'AT4G24620' and 'AT5G49460' in the radial layout. The video clip is available at http://www.vrac.iastate.edu/research/sites/metnet/Thesis/Yuting/Video/Chapter-6/Video 6-3.avi. The bottom radius of the cone representing a transcription edge (between a gene node and a RNA node) at one time is proportional to the expression level at that time. The edge shape morphing propagates from the transcription edge to the translation edge, to the assembly edge if there is one, and to the catalysis edge. There is a time delay between any two consecutive edge types. Using shape morphing along the time, the dynamic behavior is represented as the flux through the network.
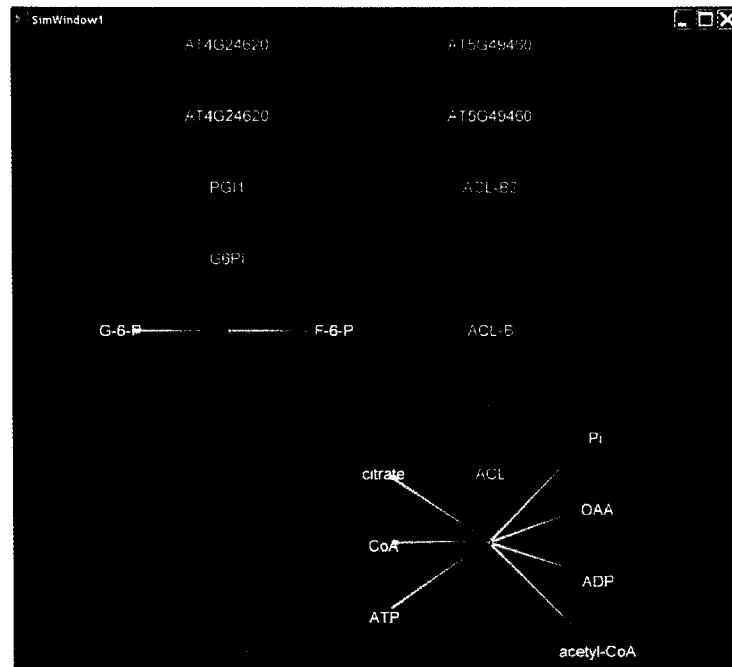
(a)
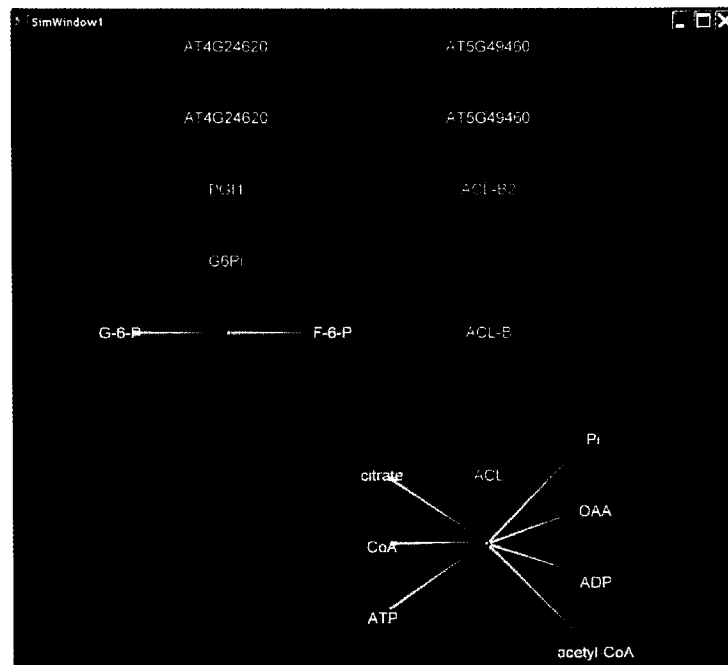
(b)

Figure 6.3 Shape morphing of ROIs focusing on 'AT4G24620' and 'AT5G49460' in

radial layout
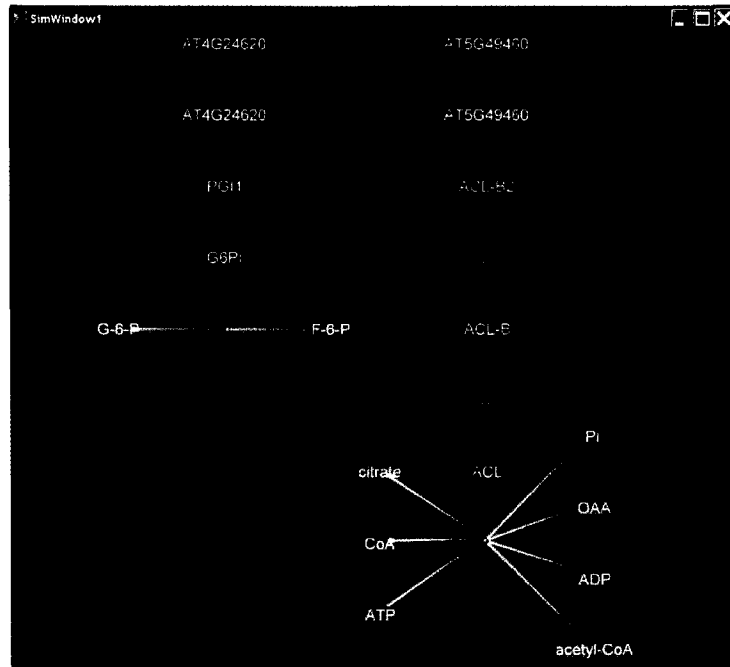
(c)
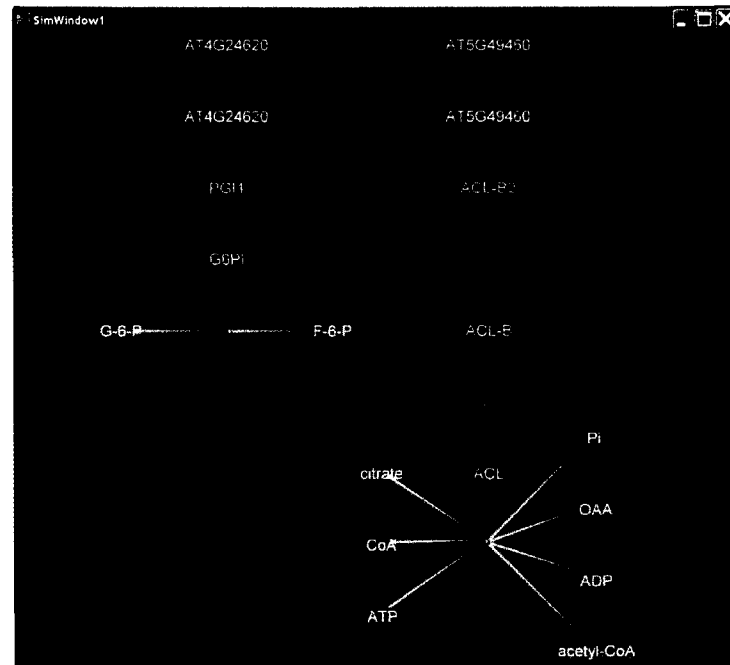
(d)

Figure 6.3 (continued)

6.3 Edge Vibration

Movement is another important property to convey dynamic behaviors in 3D computer graphics. It is indirect to use movement to display the different expression levels of genes in the network visualization because the topology and position information of the network need to remain unchanged. The vibration of the cone representing a transcription edge meets such requirements. The movement of the edge is perpendicular to the axis of the cone and is limited to a range, i.e., the same as the bottom radius of the cone, to avoid the potential collision of the edge with other edges and nodes. The vibration frequency is proportional to the current gene expression level. Figure 6.4 contains 2 snapshots from the video clip showing the edge vibration of two ROIs focusing on 'AT4G24620' and 'AT5G49460' in the radial layout. The video clip is available at http://www.vrac.iastate.edu/research/sites/metnet/Thesis/Yuting/Video/Chapter-6/Video 6-4.avi.

6.4 Discussion

This chapter presents some computer graphics based methods, including color morphing, shape morphing, and edge vibration, to integrate the gene expression levels into the metabolic network visualization. The color is a basic material property in computer graphics. The latter two can be simply implemented by the scale transformation and the translation transformation respectively in 3D computer graphics. The extra dimension of 3D space gives more room to shape change and movement than 2D space.

The potential problem is that it is difficult to follow the changes of the colors, shapes, or positions of multiple edges at the same time. The solution is to first apply the change to the reactions of interest (ROI) focusing on a gene node that contain only one gene node, and then look at multiple ROIs. It generates better results than apply the change to the whole network.
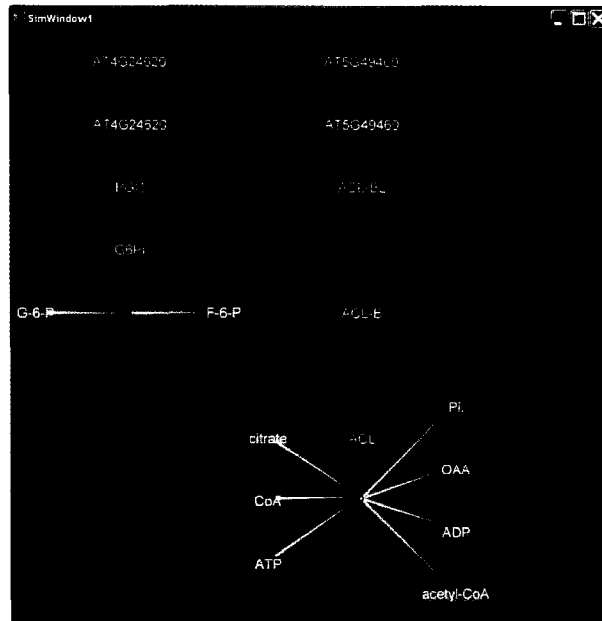
(a)
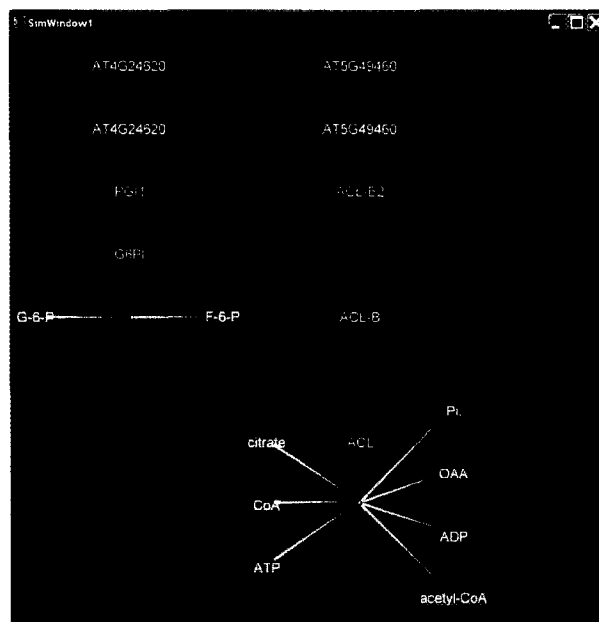
(b)

Figure 6.4 Edge vibration of ROIs focusing on 'AT4G24620' and 'AT5G49460' in radial

layout

CHAPTER 7.   METNETVR SYSTEM DESIGN

Traditional methods for the visualization of abstract information like metabolic networks use 2D space and a flowchart-like display. It is controversial to apply 3D space, computer graphics, and virtual reality (VR) for abstract information visualization. To investigate the advantage, a usability test was conducted for MetNetVR and MetNetVR Tweek.. MetNetVR implements the methodologies and algorithms described in the previous chapters to visualize high dimensional metabolic networks in 3D virtual reality. MetNetVR Tweek, a Java program with a 2D Graphical User Interface (GUI), communicates with MetNetVR in the real time, displays the detailed text information that is important to biologists but difficult to visualize, and enable complex interactions beyond the wand based interactions in VR. MetNetVR and MetNetVR Tweek are introduced in Section 7.1. The usability test is described in Section 7.2. Section 7.3 compares the exploration of a large network in MetNetVR and a 2D metabolic network visualization system. Section 7.4 compares the hierarchical visualization method with other nonhierarchical visualization methods.

7.1 MetNetVR and MetNetVR Tweek

7.1.1 System overview

Multiple global graph layout methods are implemented to automatically decide node and edge positions. The network is then visually represented by a scene. Shapes and material properties, such as color, transparency, and texture, are used as visual metaphors indicating basic properties of nodes and edges. Reactions of interest are automatically extracted from the whole network according to the user's selection of the focus gene or metabolite. Hierarchical relationships or quasi-hierarchical relationships, together with detail-on-demand interactions, help users to understand global relationships and local

details of a large scale metabolic network simultaneously. The interactions for changing the viewpoint position, selecting a node, and triggering various actions are enabled by head position tracking device and a tracked wand-like input device in virtual reality.

MetNetVR also provides a solution for linking text-rich information that is important to biologists but difficult to visualize, such as chemical reaction stoichiometry, sources of information, and synonyms, to network nodes and edges. Each node and edge is linked to a record in the database of text-based annotation data. MetNetVR Tweek, a Java program with a 2D Graphical User Interface (GUI), communicates with MetNetVR and displays the detailed text information.

The other function of MetNetVR Tweek is to contain complex interaction commands beyond the wand based interactions. These commands enable functions in MetNetVR to manipulate network visualization, such as graph theory operations, layout management, and visual metaphor controls.

MetNetVR is implemented in C++ and uses two APIs, VRJuggler (http://www.vrjuggler.org/) and OpenSG (http://www.opensg.org/). VRJuggler is a flexible development platform for VR applications (Bierbaum, Just et al. 2001). It enables MetNetVR to run in a range of platforms without changing source codes, from a fully immersive CAVE (Cruz-Neira, Sandin et al. 1993) to a conventional desktop. OpenSG is an OpenGL based API for graphics scene construction and rendering (Reiners, Vo et al. 2002). MetNetVR Tweek is implemented in C++ and Java and uses the Tweek API (Hartling, Bierbaum et al. 2002), which uses the Common Object Request Broker Architecture (http://www.corba.org/) for the cross-language communication. MetNetVR Tweek runs on a portable tablet computer and connects to MetNetVR wirelessly when the latter runs in an immersive CAVE. It can also run on the same desktop as MetNetVR when the latter runs on a conventional desktop. Figure 7.1 shows MetNetVR running in a

CAVE and MetNetVR Tweek running on a tablet computer. MetNetVR Tweek communicates with the visualization program wirelessly and displays the text rich information of the network.
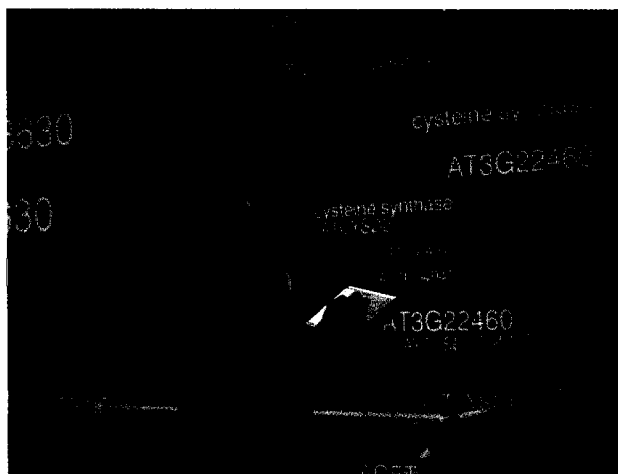
Figure 7.1 A user navigating through the metabolic network in a CAVE.

7.1.2 Interactions in MetNetVR Tweek

Wand-based interactions in MetNetVR include moving forward and backward along the wand orientation, rotating right and left along the center of the network, selecting the nodes, and triggering such actions as the generation of reactions of interest, the expansion of the selected pathway, etc. All other functions are called through menus in MetNetVR Tweek, including graph theory operations, layout management, and visual metaphor controls. These functions are necessary for an effective visualization of large metabolic networks. The main interaction commands in the MetNetVR Tweek menu fall into categories described in the following subsections.

7.1.2.1 Network management

In MetNetVR, both the standard graph model and the compound graph model exist for a metabolic network. They come from the same data source. The standard network filters out pathway (or compartment) nodes and all hierarchical relationships. The adjacency relationships among molecules are identical between a standard network and a compound network with all the compartments expanded. The layout of the compound network groups the molecules belonging to one compartment together. The layouts of the standard network only look at graph connectivity. Users toggle between these representations.

7.1.2.2 Standard network visualization

Users switch among different layouts for the standard network including a GEM-3D layout, a weighted GEM-3D layout (Chapter 3), and a multilevel force directed layout (Gajer, Goodrich et al. 2000). Ideal edge length is adjustable for each layout. The adjustment of the ideal edge length likes a zoom in/out function. However, it keeps the molecule node size fixed to maintain the label's readability. Small edge lengths are suitable for the overview of a network because they keep the network volume small. Large edge lengths work best for local details as they reduce the overlapping of node labels.

Users select a metabolite or a gene from displayed in a scrollable list which contains text-rich information for molecules in the network. The scene is automatically generated for the reactions of interest focusing on the selected metabolite (Section 4.1 and 4.2), or .the reactions of interest are pulled out using the layout animation (Section 4.3). Users can pause, resume, or stop the layout animation.

Users select a molecule of any type from the node list and set the step size. The sub-network composed of the selected molecule and its neighbors within the distance of the step size is automatically generated. Users switch the display between the whole network and the sub-network.
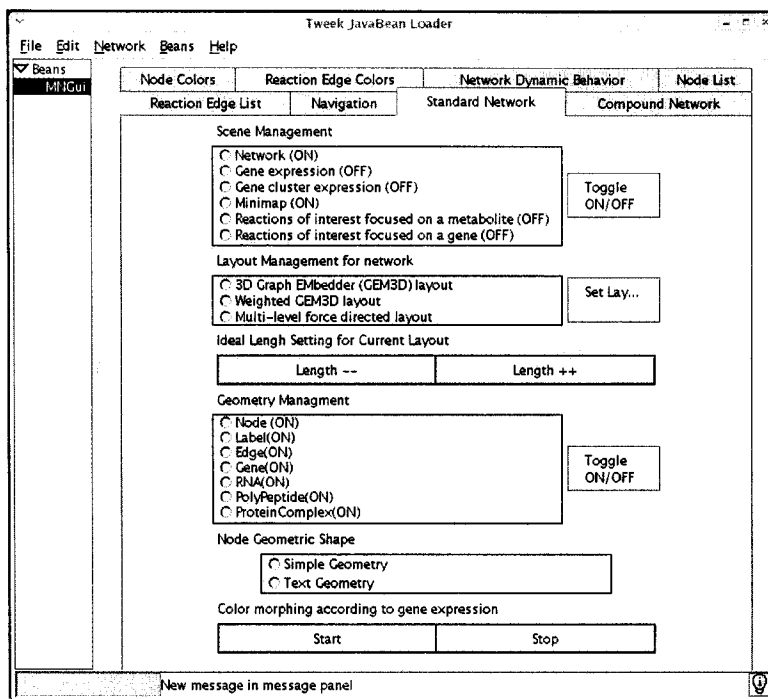
Figure 7.2 GUI for interactions controlling standard network visualization

7.1.2.3 Compound network visualization

Users select the type of hierarchical relationships to explore, including the relationships between molecules and pathways and the relationships between molecules and compartments (Chapter 5).
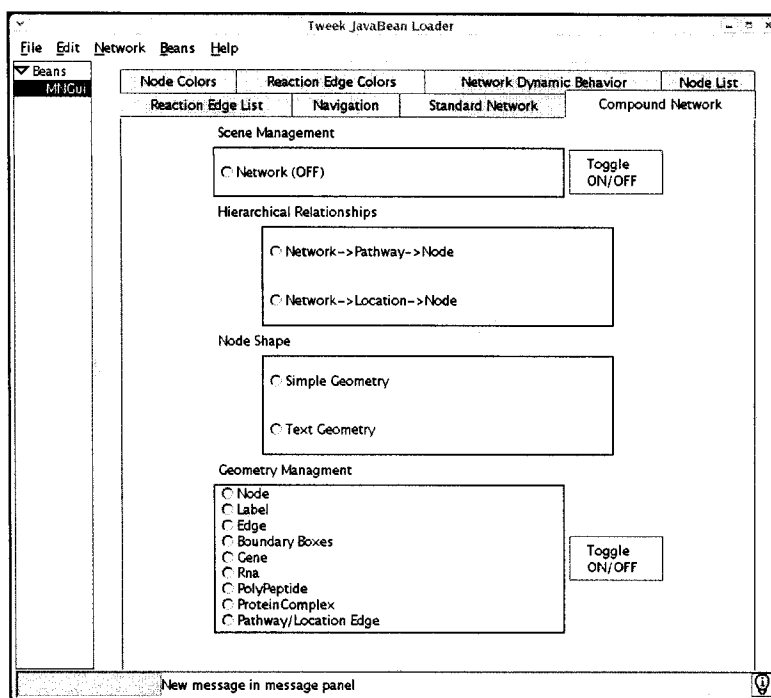
Figure 7.3 GUI for interactions controlling compound network visualization

7.1.2.4 Visual metaphor control

Users select colors for different types of nodes and different types of reaction edges. Different colors for different types of nodes and edges help users to easily recognize the type of specific nodes and edges.
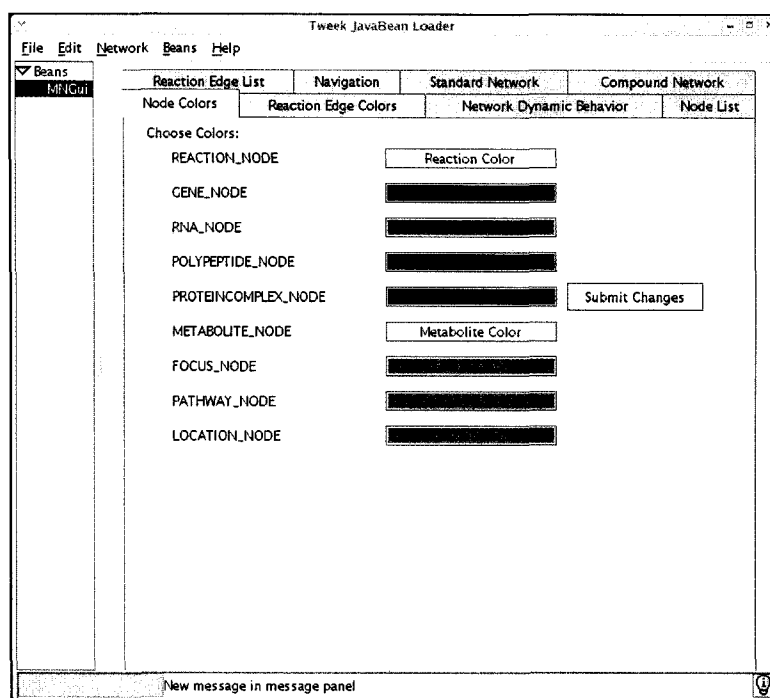
Figure 7.4 GUI for interactions controlling node colors

Users toggle the display of molecule nodes or reaction edges on and off to increase readability. For example, the user can temporarily turn off edges to read a node label because one or two letters may be blocked by the incoming/outgoing edges of this node. The user can also temporarily turn off node labels for a better vision when trying to find a cycle structure in the network.

Users can select the visual representation for nodes. Besides the visual representation introduced in Section 3.1, an inexpensive option is available regarding the needed computation for rendering. The node is represented as a geometric object indicating the node name. A translucent boundary box is available for each node. The boundary boxes of the nodes of the same type have the same color. The background is used to reduce the confusion of letters from different nodes visually overlapping. Making the background translucent is designed to avoid the visual blocking of the scene behind the node. All the nodes share the same white and opaque material. All the nodes of the

same type share the same translucent material for the boundary box. Few materials make the rendering faster. The disadvantage of the inexpensive solution is the dimming of the node name due to the translucent boundary box. Users select the type of the visual representation for nodes according to the network size and the quality requirement for the rendering.

7.1.2.5 Network dynamic behavior management

Users set the colors for the highest and the lowest expression levels and enable the color morphing to show the dynamic behaviors of the whole network or reactions of interest (Chapter 6).

7.1.2.6 Navigation management

Users select a specific node in the node list. The viewpoint in MetNetVR moves to the selected node automatically. Users select to reset the navigation. The viewpoint in MetNetVR moves back to the original position.

7.1.3 Discussion

MetNetVR and MetNetVR Tweek together implement the proposed interactive visualization of metabolic networks (Section 1.1). The implemented interactions are supposed to be necessary and sufficient for an effective and efficient visualization.

MetNetVR Tweek has the potential to be a good collaboration tool. One user is in charge of the navigation and intersection using a wand in a virtual environment. Another user uses MetNetVR Tweek to control network visualization. They two can work together to explore the network. For example, the former selects a metabolite node using the wand. The node record will be automatically highlighted in the node list in MetNetVR Tweek. The latter can use MetNetVR Tweek to enable a scene in MetNetVR, which represents the reactions of interest focusing on the selected metabolite.

7.2 Pilot Usability Test

7.2.1 Test setting

An initial usability test is conducted to see whether 3D space, computer graphics, and VR technologies are helpful to study high dimensional metabolic networks. The test is also designed to check whether the functions of MetNetVR enabled by both wand based interactions and the menu in MetNetVR Tweek are necessary and sufficient for effective visualization. The hypotheses for the usability test are:

I. Shapes and material properties in computer graphics, such as color, textures and transparency, are helpful for revealing useful information, such as molecule type, reaction edge type, and molecule names in the visualization of high dimensional metabolic networks.

II. Metabolic networks with 3D graph layouts and displayed in a stereoscopic VR environment are helpful for finding various structures contained in large networks, such as routing, cycles, groups, highly connected nodes, etc.

Six graduate students took part in the test as subjects. Table 7-1 describes their backgrounds related to MetNetVR.

Table 7.1 Subject backgrounds

Major	Bioinformatics and Computational Biology
Age	23-30
Gender	Female: 1 Male: 5
Computer Experiences:	5-16 years
3D video game experiences	Very often: 2 Some: 2 Seldom: 1 Never: 1
Education	Ph.D. student: 5 Master student: 1
Color Blindness	None: 5 Red and Green Color Blindness: 1

The subjects were asked to finish the following nine tasks corresponding to the two hypotheses.

I. Find the name and the type of two randomly selected molecule nodes (Tests first hypothesis).

II. Find any two molecule nodes whose connectivity is larger than or equal to five (Tests second hypothesis).

III. Find the routing path between two given molecule nodes (Tests second hypothesis).

IV. Find whether there is a cycle or an alternate path in the network (Tests second hypothesis).

V. Find all pathways in the network by exploring the compound network (Tests second hypothesis).

VI. Find all cellular compartments in the network by exploring the compound network (Tests second hypothesis and the hierarchical layout described in Chapter 5).

VII. Find the overlapping molecules between two given pathways, i.e., molecules appear in two pathways, by exploring the compound network (Tests second hypothesis and the hierarchical layout described in Chapter 5).

VIII. Find the reactions that a specific metabolite node is involved in (Test the functionality of the fan layout describe in Section 5.1)

IX. Find the reactions that a gene node is involved in (Test the functionality of the radial layout described in Section 5.2).

A two-hour training section was conducted before the test to introduce the functions and interactions needed to finish each task. Each subject answered two questions for each task. The first question was an effectiveness question to find whether the subject completed the task correctly. The second was an attitude question asking how

helpful the corresponding functions and interfaces supplied by MetNetVR and MetNetVR Tweek were to finish the task. The subject selected one among four answers, 'Very Helpful', 'Helpful', 'Not Helpful', and 'Other'. The subject gave an explanation if 'Other' was selected (Table 7-2).

The test was carried out in a desktop with dual monitors. MetNetVR ran in a full screen passive stereo mode in one monitor. The subjects wore LCD shutter glasses to see the stereo display. Head tracking was not used. The wand was simulated by the mouse and the keyboard in the desktop. The simulated wand has the full six degrees of freedom (The subjects were observed to only use yaw and pitch in the test). The MetNetVR Tweek program was displayed on the second monitor. This simple configuration tested MetNetVR as a tool for daily use.

7.2.2 Results and analysis

All the subjects successfully completed all the tasks and gave the correct responses to all effectiveness questions except for one subject who failed Task VII. There were two main observations about the functioning of MetNetVR during the test:

- The current rotation method, which is along the center of the network, is effective when the network is not too large and the viewpoint is out of the network. When the network is large, the user should be within the network to get details. New rotation methods, the rotations along the viewpoint and the rotation along an arbitrary position, for example, a selected node, may help users explore large graphs. Interactions will be needed to switch among different rotation centers.

- Setting the mapping between nodes/edges and colors are crucial for users with partial color blindness. One of the test subjects had red and green color blindness. He had to set the colors for all types of nodes and edges before the first task.

The analysis focuses on attitude questions. Below is a summary of the answers of all subjects to the attitude questions.

Table 7.2 Summary of the user responses to the attitude questions in the seven tasks.

Task Number	Very Helpful	Helpful	Not Helpful	Other
1	4	1	0	1 (Couldn't see stereo)
2	3	2	0	1 (Couldn't see stereo)
3	1	2	3	0
4	3	2	1	0
5	3	3	0	0
6	6	0	0	0
7	2	3	1	0
8	5	1	0	0
9	5	1	0	0

The analysis of testing results supports the hypothesis that the 3D space, computer graphics, and stereoscope are helpful for studying high dimensional metabolic networks given the conditions that users have both biological knowledge and computer experiences. Subjects' 3D experiences and gender difference don't make large difference in their performance. Since all subjects are graduate students and in the ages from 23 to 30. The conclusion may only apply for users in the same education level and age range.

The usability testing suggests that stereo virtual reality and hierarchical viewing can be used to create an effective analysis environment and reveal hierarchical relationships in metabolic networks.

The analysis also shows that simulated wand interface using the mouse and the keyboard is not comfortable for navigation and control (for example, in Task III) for most of the subjects, although a real wand device in an immersive CAVE is very intuitive.

7.2.3 Discussion

Usability testing suggests that 3D computer graphics is suitable for visualizing abstract information like metabolic networks. Metabolic networks are modeled as graphs

whose nodes and edges represent biological properties. Material properties in computer graphics, which were originally used for photorealistic rendering of the real world, are effective visual metaphors for basic biological properties.

Metabolic networks usually have a large number of nodes and edges. The combination of 3D space, rendering from different viewpoints, and stereoscopic display helps to increase the size of understandable networks. For example, two edges may look like they cross each other from one viewpoint in a monoscopic display. The apparent crossing will disappear when viewed from another viewpoint, or the depth difference shown by the stereoscopic display helps distinguish the edges.

Besides enabling binocular disparity through stereoscopic display, VR has the option of head tracking to enable the rendering of the metabolic network from different and continuous viewpoints. Moving the viewpoint and fixing the network position have the same effect as fixing the viewpoint and moving the network position in an opposite way. Both of them bring about motion parallax, which is helpful to understand the complex structures of high dimensional metabolic networks. VR also features tracked 3D input devices such as wands to enable movement and find intersections along the device orientation. Intersecting with visual representation to pick a node is of great importance for interactive visualization. For example, in the detail-on-demand method described in Section 2, user needs to pick the pathway (or compartment) node in order to expand it. 3D input devices can be simulated by mice and keyboards with full six degrees of freedom in conventional desktops. However, controlling the position and orientation in 3D space using mice and keyboards requires considerable practice.

The detail-on-demand interactions enable different hierarchical views of a metabolic network. 3D geometric inclusions, together with the transparency material property and stereoscopic display, are effective to represent hierarchical relationships in

the network. Hierarchical visualization reduces the amount of information displayed at one time and helps users understand both the global relationships and local details of a large network simultaneously.

Text-rich information display methods and complex interaction methods are needed for an effective visualization. In traditional 2D visualization systems, it is very natural to add popup windows and menu items for this purpose. In a 3D computer graphics based visualization systems for metabolic networks, an auxiliary 2D GUI program can be used to display text-rich and domain-related information and enable interactions to manipulate visualization.

7.3 Qualitative comparison between 3D space and 2D space for metabolic network visualization

FCModeler [Dickerson et al. 2003] is a metabolic network visualization system in 2D space. It shares the same input file structure of metabolic networks with MetNetVR. For the sake of comparison, the 2D GEM layout is used in FCModeler and the 3D GEM layout is used in MetNetVR.

MetNetVR was run in the simulation mode, in which the rendering is monoscopic but perspective, the position and orientation of the viewpoint is controlled with the mouse and the keyboard. The monoscopic rendering is for the purpose of viewing without any aids.

To compare 3D with 2D visualizations directly, only simple navigation methods are used in two systems. In MetNetVR, the nonhierarchical visualization is used. The navigation includes the network movement along the wand orientation, the network rotation along its center, the change of the viewpoint position, and the change of the viewpoint orientation. In FCModeler, navigation includes zooming in and out of the network and changing of the view port using the horizontal and vertical scrollbars.

Figure 7.5 shows some snapshots of the exploration of a network in Arabidopsis from the MetNet Database [Wurtele et al. 2003] using FCModeler. Figure 7.6 shows the exploration of the same network using MetNetVR. A video clip is available at (http://www.vrac.iastate.edu/research/sites/metnet/Thesis/Yuting/Video/Chapter-7/Video 7-1.avi) for the exploration using MetNetVR. There are 572 molecules and 648 reaction edges among molecules in this network. The snapshots are caught when both systems are running in the full screen mode.

7.3.1 Comparison of global views

A global view helps the user to find some structures like cycles, alternative paths, etc., in the network. To look at the global structure of the network, FCModeler zooms out the network till it fits into the current window size, which is the full screen size, as in Figure 7.5(a). For the same purpose, MetNetVR moves the network away from the viewpoint and rotate the network (Figure 7.6(a-c)). The network should be far away enough from the viewpoint so that the whole network appears in the viewing volume. Rotation along the network brings about the motion parallax, which adds depth information. The depth information plays an important role to reveal the global structure when the stereoscopic display is not available. The node label size of in Figure 7.5(a) is much smaller than that in Figure 7.6(a, c) (The labels in Figure 7.6(b) are turned off while the network rotates). The comparison result can be applied to the node label sizes in the original snapshots of MetNetVR and FCModeler because the node label sizes in the Figure 7.5and Figure 7.6 share the same proportion to the node label sizes in the original snapshots (The snapshots are caught when both systems are running in the full screen mode and now they are displayed at the same size in this document). The comparison result, together with the fact that the edge lengths are proportional to the largest node

label size, implies that for the same display window, 3D space can display more nodes than 2D space; the nodes have the same size in the display window.

7.3.2 Comparison of local views

The local view of some part of the network gives detailed information like the node name, edge types, etc. Figure 7.5(b-d) and Figure 7.6(d) show the procedure where a tree structure rooted at the node 'BAM3-5-6-7-8 Complex' (the red rectangle area in Figure 7.5(a)) is explored. In FCModeler, if the user notices from the global view that within the red rectangle area there is a tree structure, he needs to zoom in the network to a readable size and then use scrollbar to locate and browse the tree structure (Figure 7.5(b-d)). In MetNetVR, the user notices the tree structure when he rotates the whole network (the red rectangle area in Figure 7.6(c)). He moves the tree structure toward the viewpoint till the label 'BAM3-5-6-7-8 Complex' is readable (Figure 7.6(d)). The labels of 'BAM3-5-6-7-8 Complex' in MetNetVR and FCModeler are assumed to have the same readability because they have similar size in Figure 7.5(d) and Figure 7.6(d). MetNetVR keeps the tree structure within the display window. FCModeler resorts to the sliding of scrollbars to cover the whole tree structure (Figure 7.5(b-d)). MetNetVR keeps better mental images of the tree structure for the user. Another observation is that there are some edge crossings near the node 'BAM3-5-6-7-8 Complex' in FCModeler (Figure 7.5(d)). The edge crossing issue is related to the layout algorithm. However, MetNetVR and FCModeler use the same type of layout algorithm, i.e. the force directed layout, and share the same force model. The only difference is the dimension of force directions and node positions. 3D space has more rooms to avoid edge crossings. There may appear some edge crossings in MetNetVR when the network is viewed from some directions. The edge crossings will disappear when the network viewed from other directions.
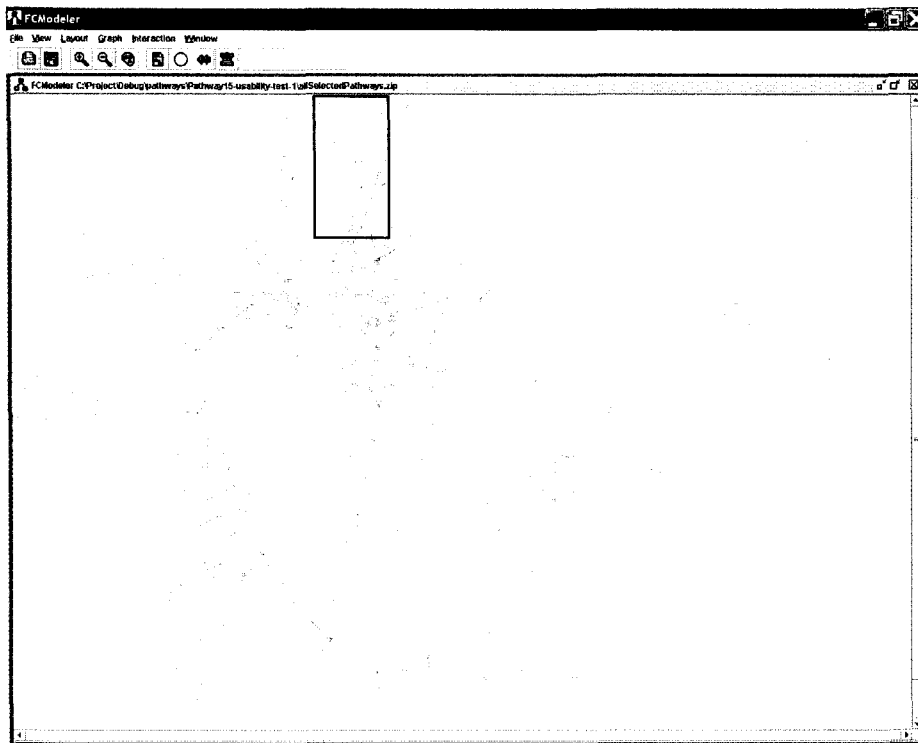
(a)
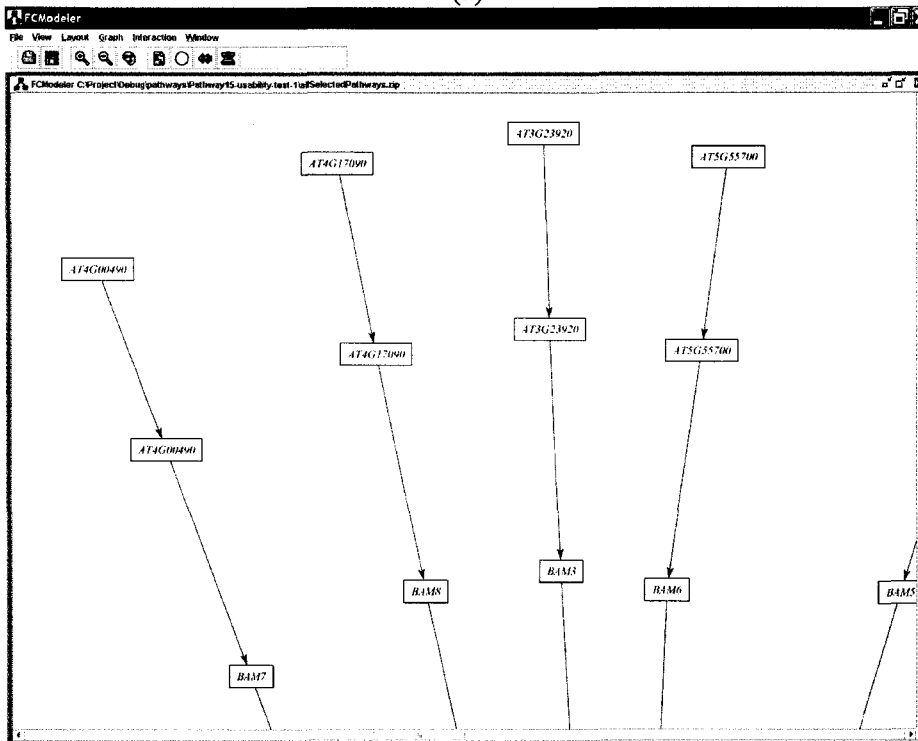
(b)

Figure 7.5 Snapshots of FCModeler

(c)

(d)
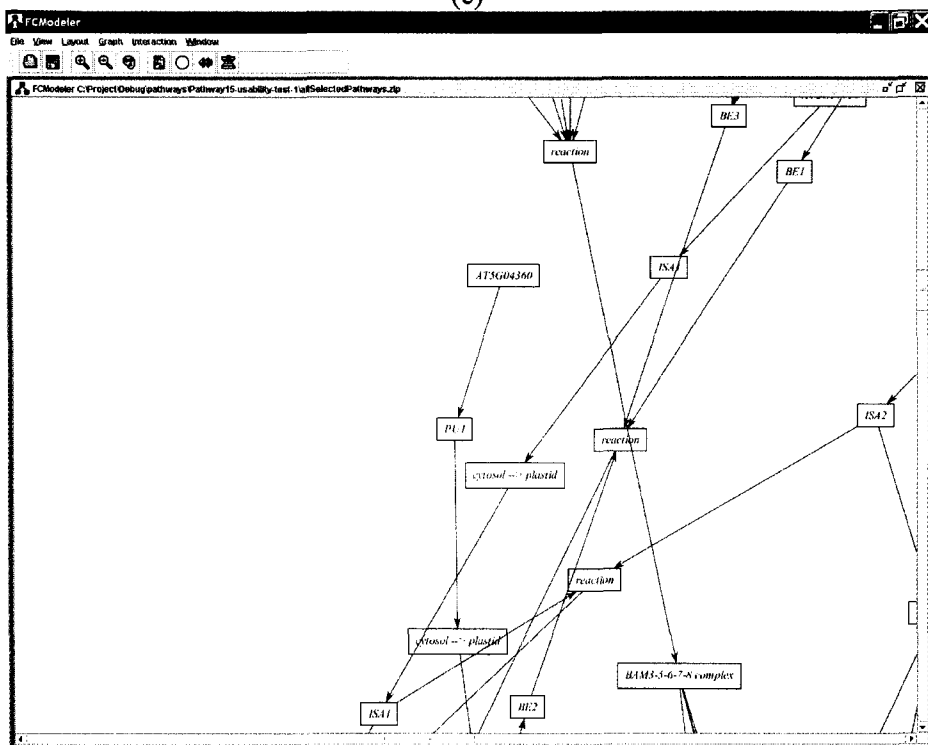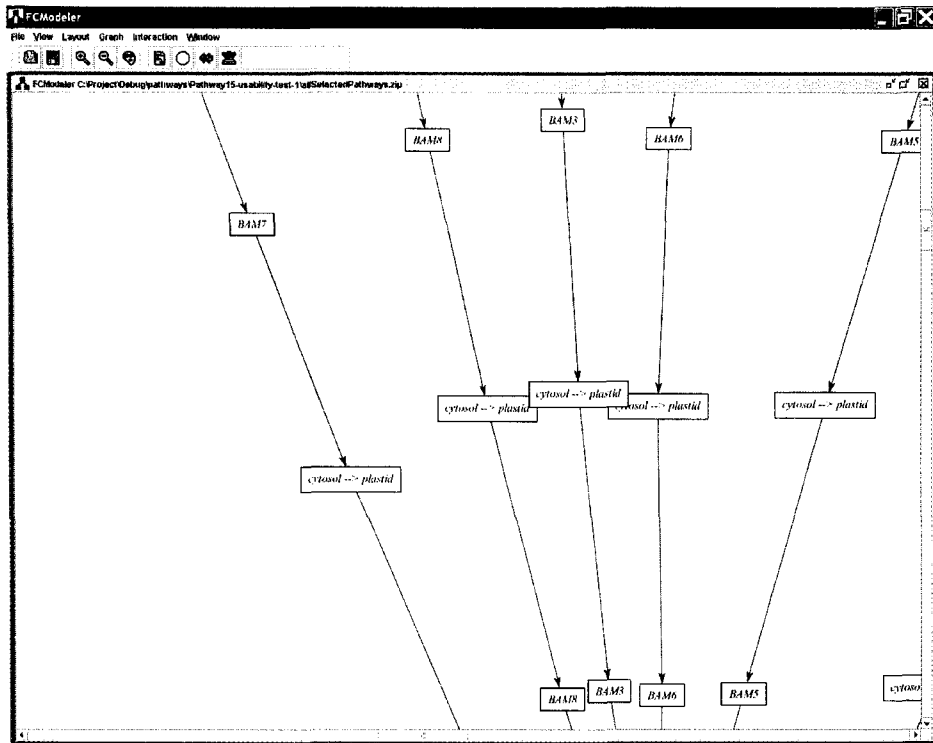
Figure 7.5 (continued)
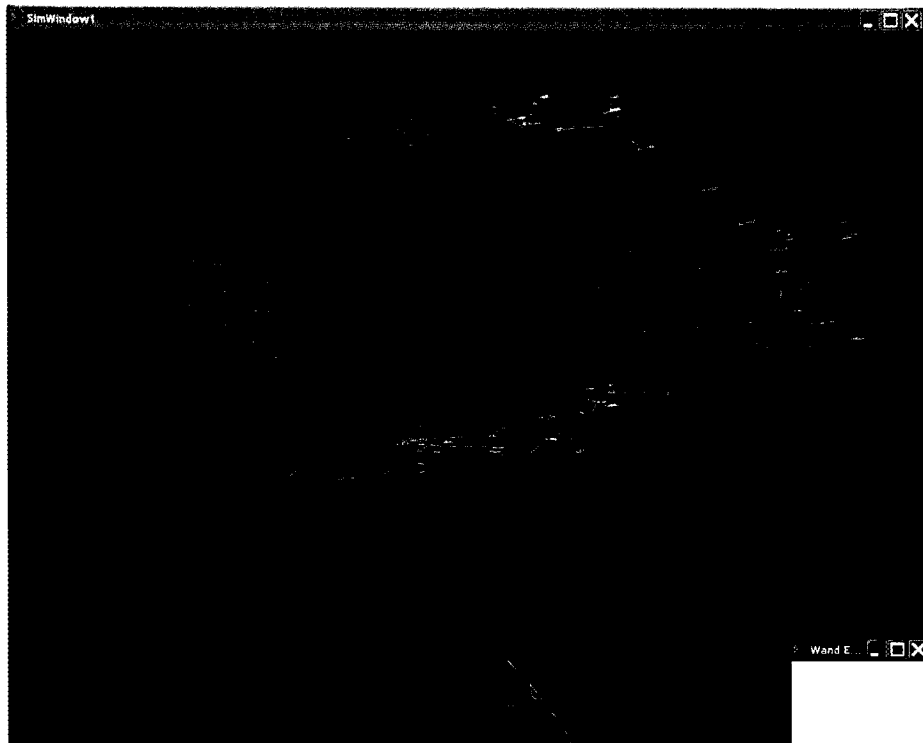
(a)

(b)

Figure 7.6 Snapshots of MetNetVR
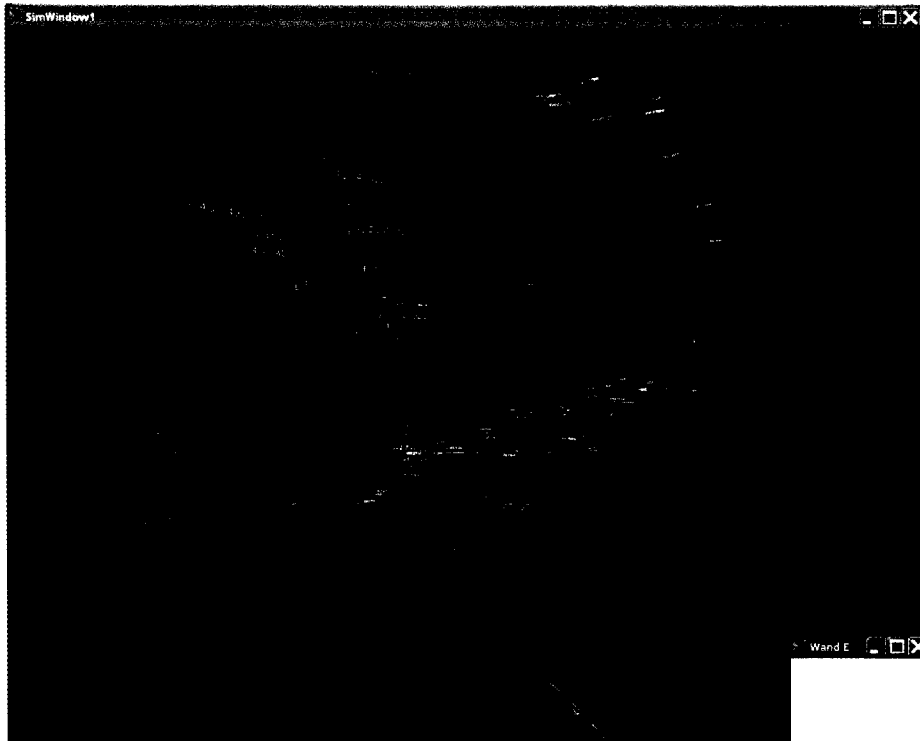
(c)
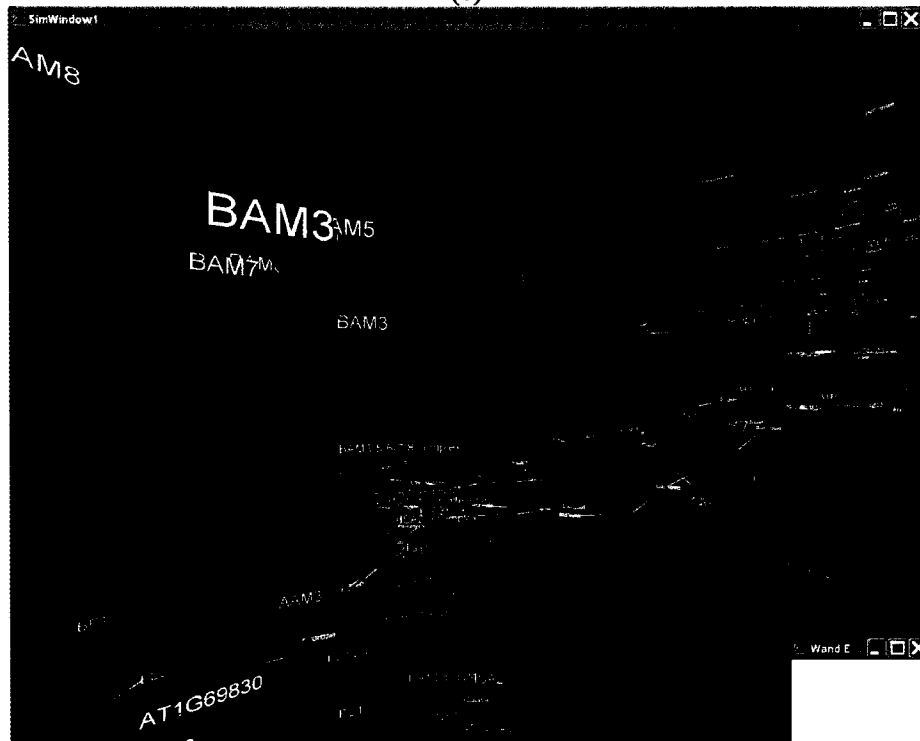
(d)

Figure 7.6 (continued)

7.4 Qualitative comparison between hierarchical visualization and nonhierarchical visualization

To explore metabolic interactions in a specific compartment, the hierarchical visualization is obviously better than the nonhierarchical visualization because the interactions are grouped together in the hierarchical layout. The comparison below is about an exploration involving multiple compartments.

Figure 7.7 shows MetNetVR exploring the node 'BAM3-5-6-7-8 Complex' (the green rectangle area) in the same network as in Figure 7.6 using the hierarchical visualization method introduced in Chapter 4. The expanded compartment is 'plastid' in Figure 7.7(a). There are such edges as BAM3->reaction node, BAM5->reaction node, BAM6->reaction node, BAM7->reaction node, BAM8->reaction node, and reaction node-> BAM3-5-6-7-8 Complex in the plastid. The knowledge that these edges and nodes belong to the plastid can be perceived from the visualization. Other knowledge about the nodes and edges within the plastid is easy to perceive.

The cross-compartment edges (edge in the blue color) also show connections between compartments. In Figure 7.7(b), the unknown compartment is expanded. For each one of BAM3, BAM5, BAM6, BAM7, and BAM8, there is a node whose compartment is unknown, which acts as a transition to the cytosol. These nodes are reaction nodes. In Figure 7.7(c) the cytosol is expanded. Each corresponding node has the same node name as the one in the plastid. This means that the same molecule appears in both compartments. The two copies are connected through a reaction node whose location is unknown. In Figure 7.7(d), nodes are explored in the cytosol.
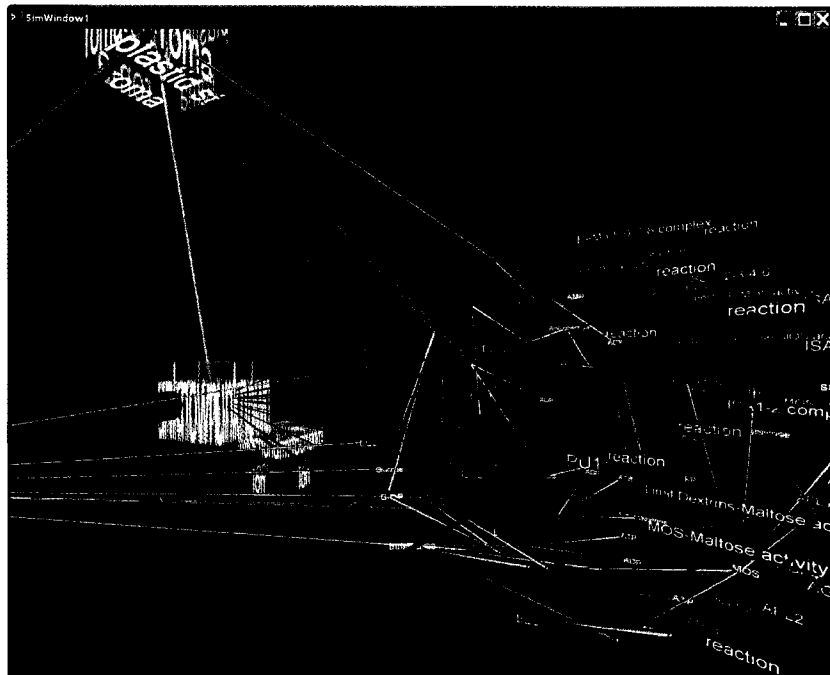
(a)
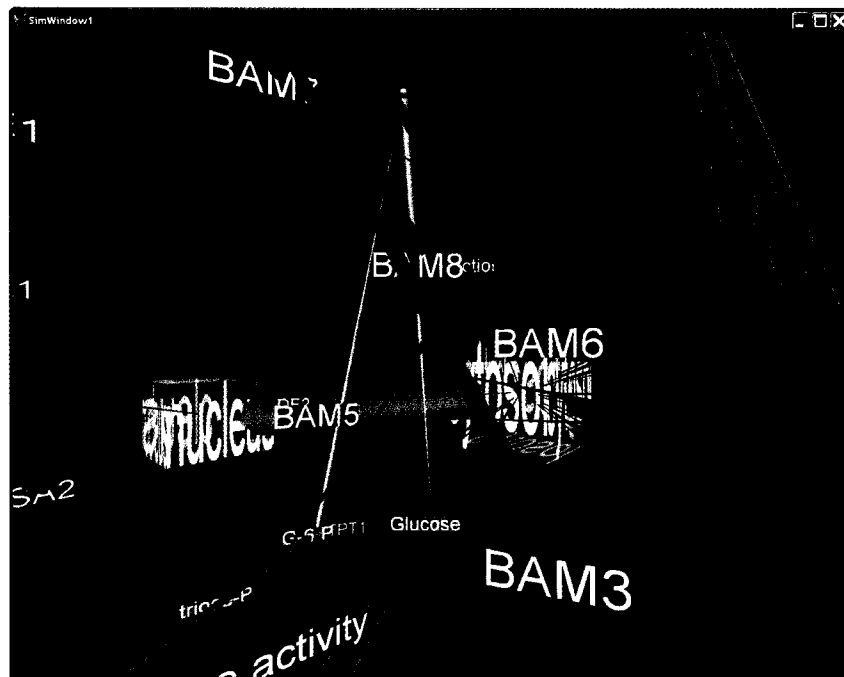
(b)

Figure 7.7 Snapshots of MetNetVR hierarchically exploring the same network as in
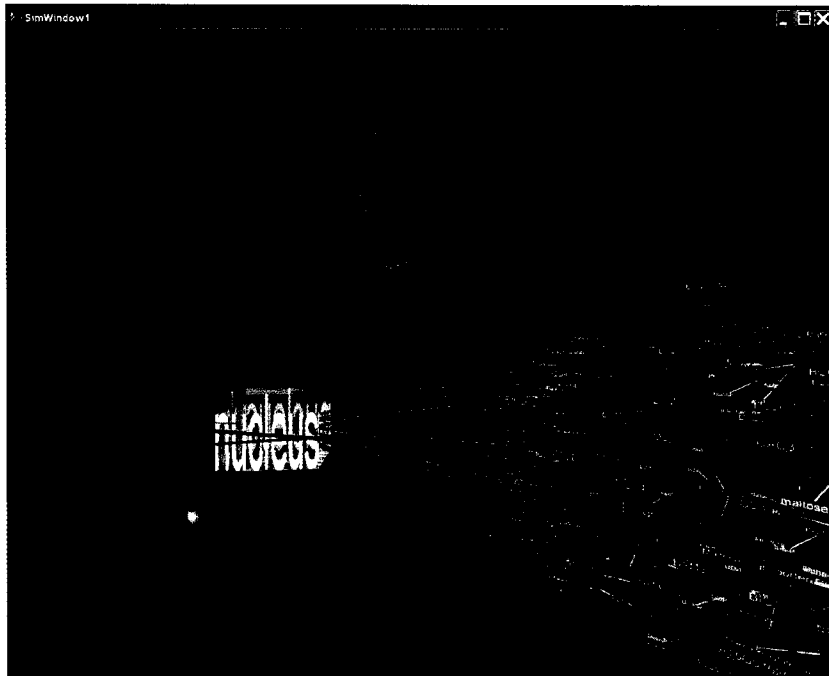
(c)

(d)

Figure 7.7 (continued)

Comparing the procedure with the nonhierarchical visualization procedure described in Figure 7.6, which excludes the hierarchical relationships, there are following observations,

1) The hierarchical visualization reveals the relationships of molecule nodes and edges to pathways or compartments.

2) The information is displayed in a more organized way in the hierarchical visualization. A single compartment or pathway is easier to layout and fit into a display window than the whole network. For example, almost all node and edges within the plastid are perceivable in Figure 7.7(a).

3) The hierarchical visualization also reveals the global relationships. The cross-compartment (or cross-pathway) adjacency relationships are easy to perceive at the compartment level and the molecule-compartment level. In the former level, a single and thick edge indicates the cross-compartment adjacency relationships. In the latter level, the edges are in the shape of cone whose top is the compartment node (e.g. the cross-compartment edges between the plastid stroma and the molecule nodes in the expanded plastid in Figure 7.7(a)).

CHAPTER 8.   CONCLUSIONS

8.1 Summary

In this work, we combined graph layouts in 3D space, computer graphics, and virtual reality to increase the size of understandable metabolic networks. A framework for an interactive visualization was proposed and implemented in MetNetVR. The user can control all the stages of graph drawing, including modeling, layout, and rendering.

In the modeling stage, the user can choose between the directed graph model and the compound graph model to represent the metabolic network. The directed graph model only considers the adjacency relationships in the network. The compound graph model also takes the hierarchical relationships into consideration. We call the visualization using the former model a nonhierarchical visualization and the visualization using the latter model a hierarchical visualization. For the hierarchical visualization, the user can choose the pathway-molecule hierarchical relationships or the compartment-molecule hierarchical relationships. The user can also extract reactions of interest according to their selection of the focusing node in this stage.

In the layout stage, for the nonhierarchical visualization, the weighted GEM-3D layout was adopted for the graph. The edge weights are decided according to the edge types so that enzymatic edges (from metabolite nodes to reaction nodes, or from reaction nodes to metabolite nodes) are emphasized by their lengths. Some important parameters, e.g. the ideal edge length, of the layout algorithm are adjustable by the user. For the hierarchical visualization, a layout algorithm was proposed, which combines the hierarchical force model with the simulated annealing method. The algorithm only runs once for the snapshot of the compound graph, where all the cluster nodes are expanded. The positions for all nodes are recorded and retrievable for different snapshots generated

by the detail-on-demand interactions. The fan layout was proposed for the reactions of interest focusing on a metabolite node. The radial layout was adopted for the reactions of interest focusing on a gene node.

The rendering stage includes two sub-stages, visual representation and rendering control. In the visual representation sub-stage, geometric objects are constructed for nodes and edges. The shapes and material properties of geometric objects, such as colors, transparencies, and textures, are used to reveal biological properties of nodes and edges, such as names and types. In the hierarchical visualization, the spatial inclusion of molecule nodes within the cluster nodes like pathway nodes or compartment nodes in 3D space reflects the hierarchical relationships. The transparency of the cluster nodes make it easy to perceive the molecule nodes inside. Multiple computer graphics techniques are used to increase the rendering performance and quality, including level of detail, billboarding, and mipmapping. In the rendering control sub-stage, the user can move the viewpoint, move the network and move the reactions of interest using head tracking devices and tracked input devices. The user can also turn on or off different part of the network. The detail-on-demand interactions in the hierarchical visualization happen in this sub-stage to expand or shrink cluster nodes. The users can also enable animations like color morphing, shape morphing, and edge vibration according to the gene expression profiling data, which presenting the dynamic behaviors of the metabolic network.

A pilot usability study was conducted to investigate the applicability of 3D space, computer graphics, and virtual reality for metabolic network visualization. A qualitative comparison between 3D space and 2D space for visualizing metabolic networks was presented. The pilot usability study and the qualitative comparison suggest that the combination of 3D space, computer graphics and virtual reality has the potential for large scale metabolic network visualization. A qualitative comparison between the

nonhierarchical visualization and the hierarchical visualization was also presented. It shows that the hierarchical layout and detail-on-demand methods can increase the amount of the perceivable content in a fixed display size.

8.2 Future work

The perception of abstract information visualization is subjective. Different users have different criteria. More functions will increase the overall satisfactions of different users. Future work for effective metabolic network visualization can be done in the following directions.

- Exploit two kinds of hierarchical relationships at the same time. Currently a single hierarchical relationship is represented using the 3D spatial inclusion. A future work can be a method to superimpose the other hierarchical relationship on this representation. The combination of two hierarchical relationships may make the whole network more organized and easier to perceive.

- Layout algorithms should take gene expression profiling data and other biological data into consideration. It is very often to compare expressions of multiple genes in the context of their connections. Current method is to pull out the reactions of interest focusing one multiple genes and look at the change of expressions using color morphing, shape morphing or edge vibration. A potential method is to take gene expression data into consideration in the calculation of node positions for the whole network. With such a layout, comparing all genes in a network in the context of their connections may be possible.

- Combine the abstract information visualization with the scientific visualization. For example, the network is displayed in a digital model of a cell; nodes and edges are located in sub-cellular compartments in the cell model. The layout algorithm will be constrained by the shapes and locations of the sub-cellular compartment

models in the cell. The combination can generate vivid rendering effects, which

may improve the perception.

Another important effort is a formal usability test of MetNetVR. The pilot usability

test suggests that the 3D space, computer graphics, and virtual reality may have the

advantage over 2D space and the flowchart-like display for metabolic network

visualization. The formal test needs to quantitatively compare MetNetVR with a 2D

visualization system of similar functionalities. Time for each task is one of the

comparable quantities. The best configuration of VR needs to be decided. The

decision should be made between large scale virtual environment like CAVE and

desktop virtual environment. The former has the best effect and the latter is suitable

for widespread and daily usage. There should be a tradeoff between the stereoscopic

display of the network and the monoscopic display of MetNetVR Tweek. Decisions

should also be made for the option of head tracking for the desktop VR.

93

APPENDIX A. COMPOUND LAYOUT ALGORITHMS

Below are the pseudo codes of the layout algorithm for the compound graph {G(N, E), T(N_t, E_t)} described in Section 4.3. Each node contains the following information,

- Position. Indicating the current node position.
- Impulse. Indicating the last impulse applied on the node. An impulse applied on a node include the brown motion of the node, gravity of the node towards the bary center of the current layout, repulsive forces from all other nodes, attractions of its adjacency edges in G(N, E), and attractions of its inclusion edges in T(N_t, E_t).
- Local temperature. Indicating the current node temperature. The temperature is used to scale the impulse applied on the node in order to accelerate the convergence towards an equilibrium state.
- Direction. Used to detect rotation (see the comments in the algorithm)

Besides the parameters for the function **compoundLayout()**, the following variables are accessible through the functions below.

edgeLen /* Ideal edge length. */
minEdgeLen /* Minimum length */
curTemp /* Current global temperature */
stopTemp /* Temperature for convergence */
stopRound /* Number of rounds to force exit */
impulse /* Currently calculated impulse. It is a 3D vector */
sum /* Sum of positions of all nodes. It is used to calculate the bary center of the current layout. It is a 3D vector */

compoundLayout(G(N, E), T(N_t, E_t), nodeDim, k_1, k_2, k_3, k_4, k_5, k_6)
/* nodeDim is the maximum dimension of all nodes. k_1 is a constant to control stopTemp, $0<k_1<1$. k_2 is a constant to control stopRound. $k_2>=1$. k_3 is the weight factor for gravitation forces. k_4 is the weight factor for the attractions of edges in G(N, E) whose ending nodes belongs to the same compartment/pathway. k_5 is the weight factor for the attractions of edges in G(N, E) whose ending nodes belongs to different compartments/pathways. k_6 is the scale factor for the attractions of edges in T(N_t, E_t)*/
{
  init();
  round = 0;
  while ((curTemp > stopTemp) and (round < stopRound))
  {
    for (i = 0; i < size(N_t); i ++)
    {
      randomly choose a node, n, from N_t;
      impulse = (0, 0, 0);
      brownMotion();
      gravity(n);
      nodeRepulsion(n);
      clusterRepulsion(n);
      adjacencyAttraction(n);
      hierarchyAttraction(n);

```
      update(n);
      }
    round ++;
  }
}
```

init()
```
{
```
factor = max (1, (size(E) + size(E_t)) / size(N_t)); /* Ideal edge length is dependent on the density of edges and at least the same as the maximum dimension of all nodes */

edgeLen = factor * nodeDim;

minEdge = nodeDim; /* Minimal edge length is the same as the maximum dimension of all nodes */

scale = size(N_t)$^{1/3}$ * edgeLen / 2;

for each node n ∈ N
```
  {
```
   n.temp = edgeLen; /* The initial local temperature */

   n.impulse = (0,0,0) /* The initial impulse*/

   n.dir = (0,0,0)

   n.position = (x, y, z) /* The initial position. x, y, and z are the random numbers between –scale and +scale */
```
  }
```
for each node n ∈ N_t–N
```
  {
```
   n.temp = edgeLen; /* The initial local temperature */

   n.impulse = (0,0,0) /* The initial impulse*/

   n.dir = (0,0,0)

   n.position = bary center of n's children in T(N_t, E_t)
```
  }
```
curTemp = edgeLen * size(N_t)

stopTemp = k_1 * curTemp; /* k_1 is a constant to control stopTemp, $0<k_1<1$ */

stopRound = k_2 * size(N_t); /* k_2 is a constant to control stopRound. $k_2>=1$ */

sum = $\sum_{n \in N_t} n.position$
```
}
```

brownMotion()
```
{
```
impulse += (x,y,z); /* x,y,z are random number between -edgeLen/2 and edgeLen/2 */
```
}
```

gravity(n)
```
{
```
center = sum/size(N_t); /* center is the center of the current layout */

impulse += (center – n.position) * k3 /* k_3 is the weight factor for gravitation forces. */
```
}
```

nodeRepulsion(n)

```
{
  if (n ∈ N)
  {
    for each node i ∈ N,
    {
      if (i != n)
      {
        Δ = i.position - n.position; /* Δ is the vector from the current position of i to the
          current position of n */
        if ( | Δ | > 0 ) /* | Δ | is the length of the vector */
          impulse += Δ/ | Δ | * edgeLen;
      }
    }
  }
}
```

clusterRepulsion(n)

```
{
  if (n ∈ Nₜ−N)
  {
    for each node i ∈ Nₜ−N,
    {
      if (i != n)
      {
        Δ = i.position - n.position; /* Δ is the vector from the current position of i to the
          current position of n */
        length1 = longest distance of n to its children in T(Nₜ, Eₜ)
        length2 = longest distance of i to its children in T(Nₜ, Eₜ)
        length = length1 + lengh2
        if ( | Δ | < length ) /* | Δ | is the length of the vector */
          impulse += Δ/ | Δ | * length;
      }
    }
  }
}
```

adjacencyAttraction(n) /* Attractions of adjacency edges of n in G(N, E) */

```
{
  for each edge of n, e , e ∈ E in G(N,E)
  {
    Δ = j.position - n.position; // j is the other node of e
    if ( | Δ |  > minEdgeLen)
    {
      //Calculate the negative impulse pnj, between n and j, the other end of e
      Δ ×= | Δ | /edgeLen * Φ(n)  /* Φ(n) is a function that grows with the degree of n. */
      if (parent(n) == parent(j) ) /* parent(n) is a function return the parent node of a
        molecule node n in tree T(Nₜ, Eₜ)*/
        Δ ×= k₄; /* k₄ is the weight factor for the attractions of edges in G(N. E) whose
          ending nodes belongs to the same compartment/pathway */
```

```
        else
          Δ ×= k₅  /* k₅ is the weight factor for the attractions of edges in G(N, E) whose
              ending nodes belongs to different compartments/pathways. */
          impulse -= Δ;
      }
    } // for
}
```

hierarchyAlttraction(n)

```
{
  for each edge of n, e, e ∈ Eₜ in T(Nₜ, Eₜ)
  {
    Δ = j.position − n.position; /* j is the other node of e */
    Δ *= k₆ * | Δ | /edgeLen * Φ(n) /* k₆ is the scale factor for the attractions of edges
        in T(Nt, Et). Φ(n) is a function that grows with the degree of n. */
    impulse -= Δ;
  }
}
```

update(n)

```
{
  if ( | impluse | > 0)
  {
    temp = n.temp;
    impulse *= temp/| impluse |  /* scale the impulse with current temperature*/
    n.position += impulse /* position update */
    sum += impulse /* update for the calculation of the bary center of the layout */
    if ( | n.impulse | > 0)  /* n.impulse is the last impulse recorded*/
    {
      α = the angle between impulse and n.impulse /* compare the direction of last
          impulse and the current impulse */
```

```
      /* Oscillation or acceleration happens when the last impulse and the current one are
      parallel*/
      if ( | cos α | > cos α₀ ) //α₀ is the opening angle for oscillation detection, α₀ ∈ [0, π/4]
      {
        /* Oscillation happens if cos α < 0, i.e. the last impulse and the current one are in
        the opposite direction. Decrease the temperature to depress the oscillation. In
        crease the temperature to accerelate if cos α > 0, i.e. two impulses have the same
        direction*/
        temp += σ₀ * cos α /* σ₀ is the sensitivity towards oscillation σ₀ >=1 */
      }
```

```
      /* Rotation may happens when the last impulse and the current one are orthogonal*/
      if ( | cos α | < cos αᵣ ) //αᵣ is the opening angle for rotation detection, αᵣ ∈ [0, π/2]
      {
        n.dir += (
          σᵣ * sign(n.impulse.x * impulse.y - n.impulse.y * impulse.x),
```

```
        /* σ_r is the sensitivity towards oscillation σ_r ∈ (0,1]. sign(x) is a function that
        returns 1 when x >=0 and -1 when x <0. Sign() return different values when
        the angle between the projections of n.impulse and impulse in XY plane has
        different signs. The absolute value of each element of n.dir will increase if the
        angle always has the same sign. */
        σ_r * sign(n.impulse.y * impulse.z - n.impulse.z * impulse.y),
        σ_r * sign(n.impulse.z * impulse.x - n.impulse.x * impulse.z)
    }

    /* Scale down the temperature to depress rotation. The more | n.dir.x | , | n.dir.x | ,
    or | n.dir.x | approachs 1, the more unbalance the node is, i.e. a rotation happens.
    Decrease the temperature to depress the rotation */
    temp *= ((1 - | n.dir.x |) *(1 - | n.dir.y |) *(1 - | n.dir.z |) )^{1/3}
    /* Update current global temperature */
    curTemp -= n.temp;
    curTemp += temp;
    /* Update the current temperature of the node*/
    n.temp = temp;
    }
    n.impulse = impulse /* record the current impulse */
    }
}
```

APPENDIX B. FAN LAYOUT ALGORITHMS

Below are the pseudo codes of the fan layout algorithm described in Section 5.1.

fanLayout(*focus_node*)
{

    /* Generate multiple copies for each other node than *focus_node* if it takes part in multiple ones of the reactions that *focus_node* takes part in. */
    split(*focus_node*);
    /* Lay out reactions one by one. */
    num = the number of reactions that *focus_node* takes part in;
    half_num = **floor**(*num*÷2); /* the function **floor** is to round down value */
    i = 0;
    for each reaction *r* that *focus_node* takes part in
    {

        /* Lay out reaction *r*, let the position of *reaction_node* to be (0, 0, 0). */
        reactionLayout(*r*);
        /* Translate *focus_node* along the –X axis with the distance of *offset* to avoid potential overlap of nodes in different reactions (Figure B.1(b)). */
        focus_node.position.x = *focus_node.position.x* – *offset*;
        /* Translate the layout of reaction *r*, let the position of *focus_node* to be (0, 0, 0). */
        for each node *n* that belongs to *r*
            n.position = *n.position* – *focus_node.position*;
        /* Rotate the layout of reaction *r* so that all reactions are evenly located in 3D space around *focus_node* (Figure B.1(a)) */
        if (*i* < *half_num*)
            Rotate the layout of reaction *r* along the Y axis with the degree of 360÷ *half_num* ×*i*;
        else
        {
            Rotate the layout of reaction *r* along the X axis with the degree of 180;
            Rotate the layout of reaction *r* along the Y axis with the degree of 360÷(*num*– *half_num*) ×(*i*– *half_num*);
        }
    }
}

split(*focus_node*)
{
    for each pair *(i, j)* of the reactions that *focus_node* takes parts in
    {
        /* Find the common nodes that reaction *i* and reaction *j* have. *i.nodes* and *j.nodes* are sets of all nodes in reaction *i* and reaction *j* respectively. */
        Φ= *i.nodes* ∩ *j.nodes*;
        /* Generate another copy for each common node except *focus_node*. */
        for each *n*∈ *Φ*
        {

```
        if (n == focus_node) continue;
        make n', a copy of n;   /*   Split n */
        substitute n' for n in reaction j;
    }
  }
}
```

reactionLayout(r)
```
{
    reaction_node.position = (0, 0, 0);
    lay out the tree rooted at reaction_node;
    calculate (α_left, α_right);  /*  (α_left, α_right) is the pie-shape boundary of the tree (Figure B.1
                                        (b))*/
```
/* For each metabolite node, decide whether to draw it at the left side of *reaction_node* or at the right side. The *focus_node* is always at the left side. Φ_{left} is the set of nodes to be drawn at the left side. Φ_{right} is the set of nodes to be drawn at the right side. */

if (*focus_node* is a substrate of reaction *r*)
```
    {
        Φ_left = the set of all substrates of reaction r;
        Φ_right = the set of all products of reaction r;
    }
```
else /* *focus_node* is a product of reaction *r* */
```
    {
        Φ_left = the set of all products of reaction r;
        Φ_right = the set of all substrates of reaction r;
    }
```
/* Lay out the nodes at the left side of *reaction_node*. Make sure that edges between any of these nodes and *reaction_node* will not intersect the pie-shape boundary (α_left, α_right). */
```
    maxHeight = unitHeight × (Φ_left.size − 1) ÷ 2;   /* unitHeight is a constant distance */
    x = maxHeight ÷ tag(α_left − δ);   /* δ is a small constant angle. */
    i = 1;
    for each n ∈ Φ_left
    {
        if (n == focus_node )
            n.position = (x, −maxHeight, 0);
        else
        {
            n.position = (x, −maxHeight + unitHeight × i, 0);
            i ++;
        }
    }
```
/* Lay out the nodes at the right side of *reaction_node*. Make sure that edges between any of these nodes and *reaction_node* will not intersect the pie-shape boundary (α_left, α_right). */
```
    maxHeight = unitHeight × (Φ_right.size − 1) ÷ 2;
    x = maxHeight ÷ tag(α_right − δ);
```

```
i = 0;
for each n ∈ Φ_right
{
    n.position = (x, −maxHeight + unitHeight × i, 0);
    i ++;
}
}
```
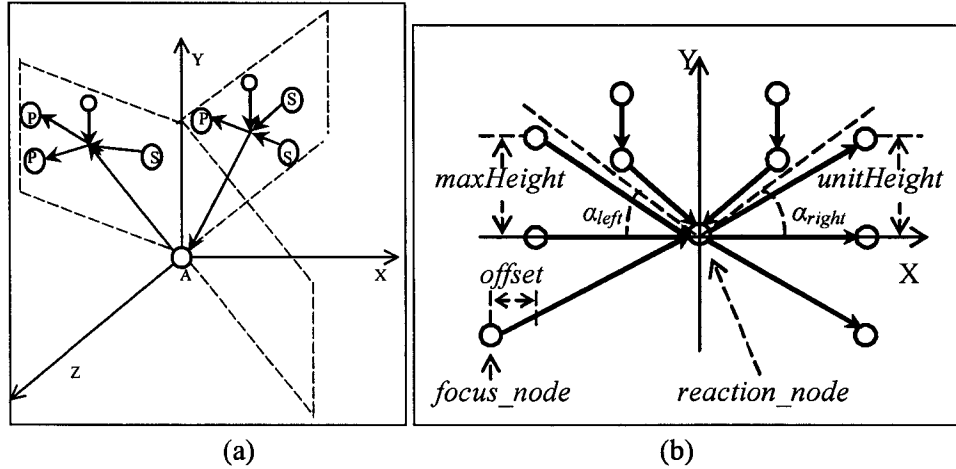

(a)   (b)

Figure B.1 (a) An illustration of a fan layout of reactions of interest focusing on node 'A'

(b) An illustration of how to draw a reaction

APPENDIX C. RADIAL LAYOUT ALGORITHMS

Below are the pseudo codes of the fan layout algorithm described in Section 5.2.

RadialLayout*(gene_node)*
{
    for each reaction *r* that *gene_node* takes part in
    {
        /* Lay out reaction *r* using any one of metabolites in *r* as *focus_node*, let the position of *reaction_node* to be $(0, 0, 0)$. */
        reactionLayout(*r*).
        /* Translate the layout of reaction *r*, let the position of *gene_node* to be $(0, 0, 0)$. */
        for each node *n* that belongs to *r*
            n.position = n.position − gene_node.position;
        calculate $(\alpha_{r,left}, \alpha_{r,right})$; /* $(\alpha_{r,left}, \alpha_{r,right})$ is the pie-shape boundary of the layout (Figure C.1(b)) */
        $\alpha_r = \mathbf{max}(\alpha_{r,left}, \alpha_{r,right})$.
    }

    $k = 2 \times \pi / \Sigma_r(\alpha_r)$;
    for each reaction *r* that *gene_node* takes part in
    {
        /* Scale the Y positions of all nodes in *r* so that angular sum of all reactions is 2π (Figure C.1(b)). */
        $H' = W_r / tan(k \times \alpha_r)$;
        $s = H'_r / H_r$;
        for each node *n* that belongs to *r*
            n.position.y = n.position.y × s;
        $R_r = \mathbf{abs}(reaction\_node.position.y)$;
    }

    $R = \mathbf{max}_r(R_r)$;   /* *R* is the radius for the final radial layout; */
    $\alpha = 0$;
    $i = 0$;
    for each reaction *r* that *gene_node* takes part in
    {
        /* Scale the X and Y positions of all nodes in *r* so that all reactions has the same radius *R* */
        $s = R / R_r$;
        for each node *n* that belongs to *r*
        {
            n.position.x = n.position.x × s;
            n.position.y = n.position.y × s;
        }
        /* Rotate *r* around *gene_node* so that the reaction_nodes for all reactions on the circle around *gene_node* */
        if ($i > 0$) $\alpha = \alpha + \alpha_r/2$;

rotate r around gene_node with the degree of α;
$\alpha = \alpha + \alpha_r/2$;
i ++;
}

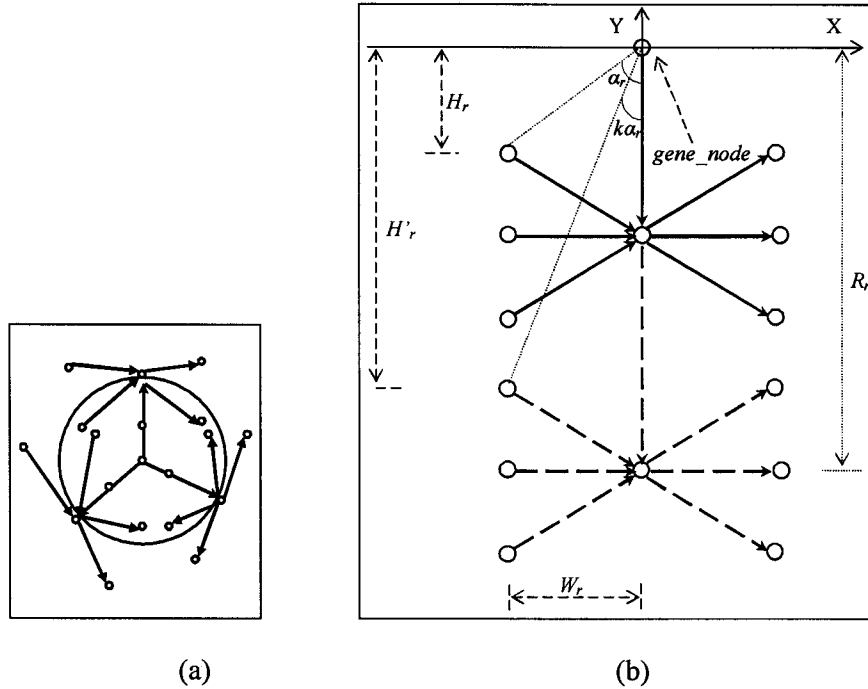(a)   (b)

Figure C.1 (a) An illustration of radial layout. (b) An illustration of how to scale the Y

position so as to narrow down the boundary angle of a reaction.

REFERENCE CITED

Angel, E. 2000. *Interactive Computer Graphics*, Addison Wesley.

Batini, C., Talamo, M. and Tamassia, R. 1984. An Algorithm for Automatic Layout of Entity-Relationship Diagrams. *Journal of Systems and Software 4*: 163-173.

Becker, M. Y. and Rojas, I. 2001. A graph layout algorithm for drawing metabolic pathways. *Bioinformatics 17*(5): 461-467.

Brandenburg, F.-J., Himsholt, M. and Rohrer, C. 1995. An Experimental Comparison of Force-Directed and Randomized Graph Drawing Algorithms. *Graph Drawing*, 76-87.

Brandes, U., Dwyer, T. and Schreiber, F. 2003. Visualizing Related Metabolic Pathways in Two and a Half Dimensions. *11th International Symposium on Graph Drawing (GD '03)*, Springer-Verlag, LNCS 2912, 111-122.

Brandes, U., Dwyer, T. and Schreiber, F. 2004. Visual Understanding of Metabolic Pathways across Organisms Using Layout in Two and a Half Dimensions. *Journal of Integrative Bioinformatics 0002*.

Bruss, I. and Frick, A. 1995. Fast Interactive 3-D Graph Visualization. *Proceedings of Graph Drawing*, Springer-Verlag, Passau, Germany, LNCS 1027, 99-110.

Chen, C. 2004. *Information Visualization - Beyond The Horizon*, Springer.

Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein, C. 2001. Elementary Graph Algorithms. *Introduction to Algorithms*, The MIT Press.

Cruz-Neira, C., Sandin, D. J. and Defanti, T. A. 1993. Surround-Screen Projection-based Virtual Reality: The Design and Implementation of the CAVE. *Proceedings of ACM SIGGRAPH 1993*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 135-142.

Davidson, R. and Harel, D. 1996. Drawing graphs nicely using simulated annealing. *ACM Transactions on Graphics 15*(4): 301--331.

Demir, E., Babur, O., Dogrusoz, U., Gursoy, A., Nisanci, G., Cetin-Atalay, R. and Ozturk, M. 2002. PATIKA: An Integrated Visual Environment for Collaborative Construction and Analysis of Cellular Pathways. *Bioinformatics 18*(7): 996-1003.

Detlef, W., Christopher, P. G. and Antoine, D. S. 2000. Visualising gene expression in its metabolic context. *Briefings in Bioinformatics 1*(3): 297-304.

Di Battista, G., Eades, P., Tamassia, R. and Tollis, I. G. 1999. *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall.

Dickerson, J. A., Berleant, D., Cox, Z., Qi, W., Ashlock, D., Wurtele, E. S. and Fulmer, A. W. 2003. Creating and Modeling Metabolic and Regulatory Networks Using Text Mining and Fuzzy Expert Systems. *Computational Biology and Genome Informatics*. C. H. W. Jason T. L. Wang, Paul Wang, World Scientific Publishing: 207-238.

Dickerson, J. A., Cox, Z., Wurtele, E. S. and Fulmer, A. W. 2001. Creating Metabolic and Regulatory Network Models using Fuzzy Cognitive Maps. *North American Fuzzy Information Processing Conference (NAFIPS)*, Vancouver B.C., 2171-2176.

Dickerson, J. A., Yang, Y., Blom, K., Reinot, A., J., L., Cruz-Neira, C. and S., W. E. 2003. Using Virtual Reality to Understand Complex Metabolic Networks. *Atlantic Symposium on Computational Biology and Genomic Information Systems and Technology*, 950-953.

Dogrusoz, U., Giral, E., Cetintas, A., Civril, A. and Demir, E. 2004. A Compound Graph Layout Algorithm for Biological Pathways. *12th International Symposium on Graph Drawing*, NYC, USA

Dwyer, T., Rolletschek, H. and Schreiber, F. 2004. Representing Experimental Biological Data in Metabolic Networks. *Second Asia-Pacific Bioinformatics Conference (APBC2004)*, Australian Computer Society, CRPIT, 13-20.

Eades, P. 1984. A heuristic for graph drawing. *Congressus Numerantium 42*: 146-160.

Eades, P. 1992. Drawing Free Trees. *Bulletin of the Institute of Combinatorics and its Applications*: 10-36.

Eades, P. and Feng, Q.-W. 1996. Multilevel Visualization of Clustered Graphs. *Proc. Graph Drawing, {GD}*, 101--112.

Eades, P., Feng, Q.-w. and Lin, X. 1996. Straight-Line Drawing Algorithms for Hierarchical Graphs and Clustered Graphs. *The 4th Symposium on Graph Drawing (GD '96)*, 113-128.

Eades, P. and Huang, M. L. 2000. Navigating Clustered Graphs using Force-Directed Methods. *J. Graph Algorithms and Applications: Special Issue on Selected Papers from 1998 Symp. Graph Drawing 4*(3): 157--181.

Eisen, M. B., Spellman, P. T., Brown, P. O. and Botstein, D. 1998. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences 95*(25): 14863–14868.

Frick, A., Ludwig, A. and Mehldau, H. 1994. A Fast Adaptive Layout Algorithm for Undirected Graphs. *Graph Drawing*, LNCS 894, 388-403.

Friedrich, C. and Eades, P. 2002. Graph Drawing in Motion. *Journal of Graph Algorithms and Applications 6*(3): 353-370.

Friedrich, C. and Houle, M. E. 2001. Graph Drawing in Motion II. *9th International Symposium on Graph Drawing*, Vienna, Austria, LNCS 2265, 220-231.

Fruchtermann, T. M. J. and Reingold, E. M. 1991. Graph drawing by force-directed placement. *Software--- Practice and Experience 21*: 1129-1164.

Gansner, E. R., Koutsofios, E., North, S. C. and Vo, K.-P. 1993. A Technique for Drawing Directed Graphs. *Software Engineering 19*(3): 214-230.

Herman, I., Melancon, G. and Marshall, M. S. 2000. Graph Visualization and Navigation in Information Visualization: a Survey (2000). *IEEE Transactions on Visualization and Computer Graphics 6*(1): 24-43.

Irani, P. P. and Iturriaga, C. 2002. Labeling Nodes in 3D Diagrams: Using Transparency for Text Legibility and Node Visibility. Technical Report TR02-152, University of New Brunswick.

Kamada, T. 1989. *Visualizing Abstract Objects and Relations*. Singapore, World Scientific.

Kamada, T. and S., K. 1989. An algorithm for drawing general undirected graphs. *Information Processing Letters 31*(1): 7-15.

Kanehisa, M. and Goto, S. 2000. KEGG: kyoto encyclopedia of genes and genomes. *Nucleic Acids Research 28*(1): 27-30.

Karp, P. D. and Paley, S. M. 1994. Automated drawing of metabolic pathways. *3rd Int. Conf. on Bioinformatics and Genome Research*, 225-238.

Kurhekar, M. P., Adak, S., Jhunjhunwala, S. and Raghupathy, K. 2002. Genome-Wide Pathway Analysis And Visualization Using Gene Expression Data. *Pacific Symposium on Biocomputing 2002*, 462-473.

Lander, E. S. and Linton, L. M. 2001. Initial sequencing and analysis of human genome. *Nature 409*: 860-921.

Landgraf, B. 2001. 3D Graph Drawing. *Drawing Graphs: Methods and Models, Lecture Notes in Computer Science*, Springer-Verlag GmbH. **2025**: 172-191.

Maletic, J. I., Leigh, J. and Marcus, A. 2001. Visualizing Software in an Immersive Virtual Reality Environment. *ICSE'01 Workshop on Software Visualization*, Toronto, Canada, 49-54.

Maletic, J. I., Leigh, J., Marcus, A. and Dunlap, G. 2001. Visualizing Object Oriented Software in Virtual Reality. *The 9th IEEE International Workshop on Program Comprehension (IWPC'01)*, Toronto, 26-35.

Monien, B., Ramme, F. and Salmen, H. 1995. A Parallel Simulated Annealing Algorithm for Generating 3D Layouts of Undirected Graphs. *3rd Int. Symp. Graph Drawing*, Springer-Verlag, LNCS 1027, 396-408.

Munzner, T. 2000. Interactive Visualization of Large Graphs and Networks. <u>Department of Computer Science</u>, Stanford University.

Nelson, L., Cook, D. and Cruz-Neira, C. 1999. XGobi vs. the C2: Results of an Experiment Comparing Data Visualization in a 3-D Immersive Virtual Reality Environment with a 2-D Workstation Display. *Computational Statistics: Special Issue on Interactive Graphical Data Analysis 14*(1): 39-52.

Oliver, D. J., Nikolau, B. J. and Wurtele, E. S. 2002. Functional Genomics: High throughput mRNA, protein, and metabolite analyses. *Metabolic engineering 4*: 98-108.

Ostry, D. I. 1996. Some Three-Dimensional Graph Drawing Algorithms. <u>Department of Computer Sciences and Software Engineering</u>, The University of Newcastle.

Parker, G., Franck, G. and Ware, C. 1998. Visualization of Large Nested Graphs in 3D: Navigation and Interaction. *Journal of Visual Languages and Computing 9*(3): 299-317.

Risden, K., Czerwinski, M. P., Munzner, T. and Cook, D. 2000. An Initial Examination of Ease of Use for 2D and 3D Information Visualizations of Web Content. *International Journal of Human Computer Studies 53*(5): 695-714.

Rojdestvenski, I. 2003. Metabolic pathways in three dimensions. *Bioinformatics 19*(18): 2436-2441.

Rojdestvenski, I. and Cottam, M. 2002. Vsualizing Metabolic Networks in VRML. *Sxth International Conference on Information Visualisation*, 175-180.

Rowe, L. A., Davis, M., Messinger, E., Meyer, C., Spirakis, C. and Tuan, A. 1987. A browser for directed graphs. *Software--Practice and Experience 17*(1): 61-76.

Ryoko, T. and Toshiyuki, A. 2003. A System for Visualizing Gene Expressions Using Metabolic Networks. *Genome Informatics 14*: 340-341.

Sander, G. 1995. A Fast Heuristic for Hierarchical Manhattan Layout. *International Symposium on Graph Drawing*, LNCS 1027, 447-458.

Sander, G. 1996. Layout of Compound Directed Graphs, Universit at des Saarlandes.

Sarkar, M. and Brown, M. H. 1992. Graphical Fisheye Views of Graphs. *Human Factors in Computing Systems, CHI'92 Conference Proceedings*, 83-91.

Schreiber, F. 2002. High quality visualization of biochemical pathways in BioPath. *Silico Biology 2*(2): 59-73.

Schreiber, F. 2003. Comparison of metabolic pathways using constraint graph drawing. *Conferences in Research and Practice in Information Technology Series*, Australian Computer Society, Inc, Adelaide, Australia, Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics 2003, 105-110.

Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B. and Ideker, T. 2003. Cytoscape: A Software Environment for

Integrated Models of Biomolecular Interaction Networks. *Genome Research 13*: 2498-2504.

Shreiner, D., Woo, M., Neider, J. and Davis, T. 2005. *OpenGL Programming Guide : The Official Guide to Learning OpenGL, Version 2 (5th Edition)*.

Solomon, E. P., Berg, L. R. and Martin, D. W. 1999. *Biology*.

Sugiyama, K. 2002. *Graph Drawing and Applications for Software and Knowledge Engineers*, World Scientific.

Sugiyama, K. and Misue, K. 1991. Visualization of structural information: Automatic drawing of compound digraphs. *IEEE Trans. Systems, Man and Cybernetics, 21*(4): 876-892.

Sugiyama, K., Tagawa, S. and Toda, M. 1981. Methods for Visual Understanding of Hierarchical Systems. *IEEE Transactions on Systems, Man and Cybernetics 11*(2): 109-125.

Sutherland, I. E. 1968. A head-mounted threedimensional display. *Proc. of the Fall Joint Computer Conference*, 757-764.

Tamassia, R., Di Battista, G. and Batini, C. 1988. Automatic Graph Drawing and Readability of Diagrams. *IEEE Transactions on Systems, Man and Cybernetics SMC18*(1): 61-79.

Toyoda, T., Mochizuki, Y. and Konagaya, A. 2003. GSCope: a clipped fisheye viewer effective for highly complicated biomolecular network graphs. *Bioinformatics 19*(3): 437-438.

Tunkelang, D. 1994. A Practical Approach to Drawing Undirected Graphs.

Ware, C. and Franck, G. 1994. Viewing a Graph in a Virtual Reality Display is Three Times as Good as a 2D Diagram. *IEEE Symposium on Visual Languages*, 182-183.

Ware, C. and Franck, G. 1996. Evaluating Stereo and Motion Cues for Visualizing Information Nets in Three Dimensions. *ACM Transactions on Graphics 15*: 121-140.

Ware, C., Franck, G., Parkhi, M. and Dudley, T. 1997. Layout for visualizing large software structures in 3D. *The Second International Conference on Visual Information Systems (Visual'97)*, San Diego, 215-225.

Ware, C., Hui, D. and Franck, G. 1993. Visualizing Object Oriented Software in Three Dimensions. *IBM Centre for Advanced Studies Conference, Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research: software engineering*, Toronto, 612 - 620.

Williams, L. 1983. Pyramidal Parametrics. *ACM SIGGRAPH '83*, ACM, 1-11.

Wurtele, E. S., Dickerson, J. A., Cook, D., Hofmann, H., Li, J. and Diao, L. 2003. MetNet: software to build and model the biogenetic lattice of Arabidopsis. *Comp Funct Genom 4*: 239-245.

Yang, Y., Engin, L., Wurtele, E. S., Cruz-Neira, C. and Dickerson, J. A. 2005. Integration of metabolic networks and gene expression in virtual reality. *Bioinformatics 21*(18): 3645-3650.

Yang, Y., Wurtele, E. S., Cruz-Neira, C. and Dickerson, J. A. 2006. Hierarchical Visualization of Metabolic Networks using Virtual Reality. *ACM International Conference on Virtual Reality Continuum and Its Applications*

ACKNOWLEDGMENTS

Many people have helped me during my Ph.D. studies. I would like to thank all of them for their constant support.

I am sincerely indebted to my major professor, Dr. Julie Dickerson. She has led me into the fantastic research area of information visualization and given me so much guidance and encouragement through the years. I would also like to express gratitude to Dr. Carolina Cruz-Neira for her advice through the whole project, especially in the design of complex software systems for virtual reality and in the usability test. I am indebted to Dr. Eve Wurtele for her advice in my Ph.D. research, especially in writing and biology. I would also like to thank Dr. Dirk Reiners for his help in computer graphics programming and Dr. Manimaran Govindarasu for his advice on the dissertation.

I would like to thank Virtual Reality Applications Center (VRAC) for providing me with a good work environment. Especially, I would like to thank Kevin Teske, Glen Galvin, Karen Koppenhaver, and Beth Hageman for their daily support. Also I would like to thank Dr. Derrick Parkhurst, Dr. Josette Etzel, Dr. Pan Du, Michael Lawrence, Jie Li, and many other people in VRAC for the discussions and help during the years.

This research makes use of a lot open source software packages, including VRJuggler, OpenSG, R, and Rserve. I am grateful to people in the mailing lists of these open source projects for promptly answering my questions.

Finally, I owe my best gratitude to my beloved wife, Ying Feng, my parents, my brother, and other family members. Their love and patience have been always encouraging me during my life.