Dynamically Binding Image to Text for Information Communication

Francis T. Marchese
Department of Computer Science, Pace University, New York, NY 10038
fmarchese@pace.edu

Abstract

The purpose of this paper is to demonstrate that a tight dynamical connection may be made between text and interactive visualization imagery. It shows that a bi-directional linkage may be created between the image space of a visualization program and hypertext space so that dynamical image and text representations of a data object are synchronized, thus maintaining the consistency of the visual information and information context. The paper begins with an analysis of the relationship between text and image, drawing upon examples from the history of art. It then discusses how text and imagery may be bound to improve the communication of information. Finally, a simple mapping application is created employing XML, HTML, and Scalable Vector Graphics (SVG) demonstrating these principles.

1. Introduction

Traditionally, text-oriented media, such as journals or monographs, are the final resting place for images created during the scientific visualization process. It is here that concise, expository text, commands supreme authority, responsible for communicating the details of the research process and its significant results. Static imagery is relegated a subservient role, ideally *illustrating* the essential points set forth within the paper. In contrast, visualization systems create interactive, dynamic imagery, displayable on any conforming output device from laptop to video wall, providing a level of visual description and detail not found in the written word. But these systems may not be able to display, annotate, and format the large body of descriptive text required to thoroughly document the imagery.

Hypermedia is a natural environment for integrating text with imagery because graphical content and text not only coexist but may be dynamically linked. Indeed, the navigational capabilities of hypertext combined with multimedia content have made it a complementary delivery mechanism for scholarly journals [1]. Yet, a problem arises when visualization software is embedded within a hypertext document.

Let us assume that a scientific communication is translated into an HTML document in which static images are replaced with an embedded interactive visualization program that allows the viewer to manipulate graphical depictions of the data. Also assume that the initial imagery correspond to snapshots from the manuscript. Finally, assume that the reader interactively explores the data, making substantive changes to its appearance.

Such a scenario is possible in the HTML document shown in Figure 1 [2]. This web page describes the structure of HIV-1 reverse transcriptase complexed with FAB-28 monoclonal antibody fragments. The left frame of the document contains an embedded link to a chemical data file that is read by a plug-in visualization program called Chime [3]. The descriptive text is loaded into the right frame. Chime is a molecular visualization program that can read protein structure files and allows users to directly manipulate the three dimensional molecular data with mouse control over geometric transformations such as rotation, scaling, and translation; and change the rendering style for all or part of the molecule.

When the web page is initially loaded, text and graphical descriptions of the data are synchronized. The viewer may change the visual representations at any time through a pop-up menu. But, in so doing, the dynamic image may no longer accurately reflect the contents of the static text. Chime can correct for these changes with the use of its scripting capabilities. By clicking on embedded buttons within the text (c.f. Figure 1), Chime loads scripts that change structural representations and invoke animation commands. And in this web page, these buttons are used to synchronize imagery to text by invoking scripts that illustrate what the text is communicating.

Proceedings of the Eighth International Conference on Information Visualisation (IV'04)
1093-9547/04 $ 20.00 IEEE

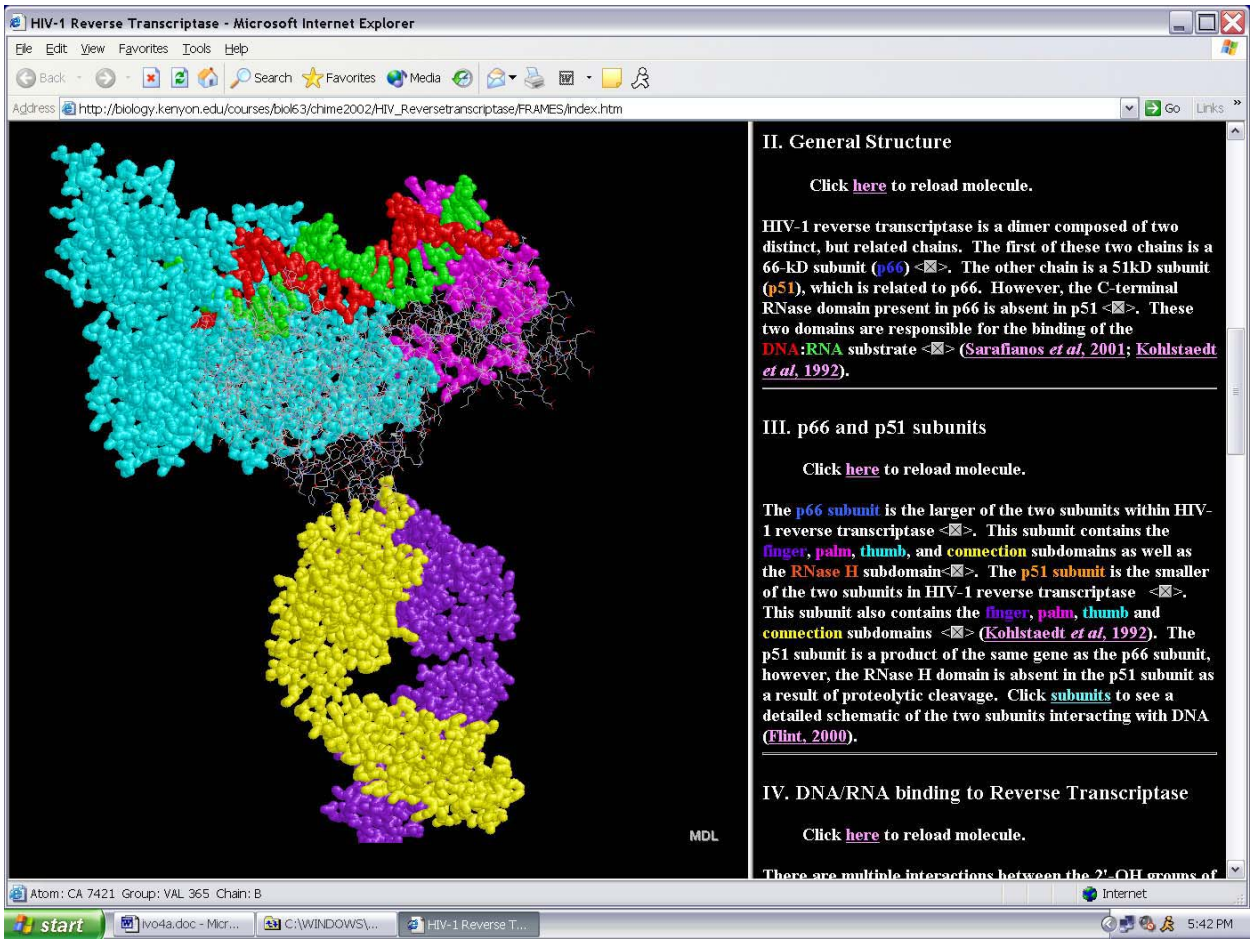

Figure 1. Demonstration of linking text with real-time visualization through scripts that are launched by clicking on embedded buttons within text.

This extension of Chime's user interface into the text body through the use of buttons gives the appearance that the text has the same direct manipulation control over the data as the visualization program. This is not true, because the static text has no link to the data objects. Moreover, operations within Chime have no effect on the text displayed. Hence, when a component of the visual representation is selected, the text does not communicate any information about it. And that leads to the problem addressed in this paper: how can a bi-directional linking be made between text and interactive visualization so as to dynamically synchronize each representation of data.

The solution put forward herein relies on XML, its linking capabilities, and SVG (Scalable Vector Graphics). In the following section we will present the problems of placing text within imagery. Section 3 contains a discussion of binding text to imagery. Section 4 briefly covers SVG. The demonstration is found in Section 5, with discussion following in Section 6.

2. Text within Image

The hypermedia example presented in Section 1 assumed that text and image occupied neither the same program nor physical space. As a result, the text was unable to gain access to the data objects to which the visualization software was manipulating freely. If the text were a graphical object, it could be placed within the image space occupied by the molecular model and logically linked to the data object. However, text placement within an image remains problematic.

For centuries, artists have been challenged by placing text within images. The ancient Egyptians immersed their images in hieroglyphics, many following the contours of representations of pharaohs and dei-

ties. This text defined the shape of the visual space the imagery occupied. In early medieval manuscripts the initial letters were given flourishes, shaped like animals or humans. Some words were enhanced to give the appearance of carved forms. Other words were represented as though they were placed within scrolls held by figures. In many portrayals of saints, especially those who wrote scripture, images of books covered with text or text-like marks were included within the composition. Sometimes the text was arranged so the viewer of the art could read the words. In other works the text was inverted so that it appeared to be read by figures. A novel compositional convention is found in Fra Angelico's Annunciation (c.1432-1434, Museo Diocesano, Cortona, Italy) that also appears within medieval illuminations. The painting represents the New Testament account of the angel Gabriel announcing to the Virgin Mary that she will give birth to a son, Jesus. It is composed with Gabriel on the left facing Mary on the right. The words spoken by Mary and the Angel appear to be coming out of their mouths. Gabriel's words are arranged in sequence from left-to-right, while Mary's response is arranged from right-to-left.

Artists like Manet used their signatures to indicate a horizontal plane beneath their subjects' feet. Or they signed their work on a book or piece of paper portrayed within the painting. Georgia O'Keefe painted the last name of her husband Alfred Stieglitz as though it were a neon sign in a night time cityscape. Charles Demuth's painting "I Saw the Figure **5** in Gold" (Metropolitan Museum of Art, New York), not only has three repetitions of the figure **5** diminishing into the vortex of the work, it also includes the words Bill and initials W.C.W., the name and initials standing for William Carlos Williams the author of the poem upon which the image is based. In this painting, the numbers and words establish the major planes of the work, create perspective, and relate strongly to the content and spare style of the poem.

Cubist artists incorporated painted letters and words, and later pieces of newspaper or labels into their works. Sometimes words were used as the sole means for indicating a plane in space. Conversely, the text would subvert the geometric sense of the plane. According to Meyer Shapiro, these artists were freed from the bookish aspect of using words and experienced no conflict between modeled forms and the use of words [4]. Hence, bodies of text became graphical elements, part of the total composition of the visual representation, complementing or complicating its meaning.

Psychological research sets these observations within a scientific context [5]. For example, according to Paivio's [6] dual coding model of working memory, there are separate but interconnected processing subsystems for verbal and visual stimuli. And although visual text is initially processed by the visual subsystem, it is passed onto the verbal subsystem for integration with language structures found there. In addition, researchers have shown that images and text are better at communicating different concepts [7,8]. In general, text is better at communicating abstract concepts, procedural information, and logic; while imagery is better at conveying spatial and structural relationships, a sense of place, and detail about an objects appearance.

In summary, whether the visualization is static or dynamic, outside of sparingly inserting labels and annotations into an illustration, the inclusion of larger bodies of text affects the overall composition and visual sense of the image. Therefore, text and image spaces are best treated separately.

3. Image-Text Binding in Hypertext

HTML supports a binding between text and image through the action of linking such that text selection exposes or launches a static image, an animation, or an independent interactive application. The reverse is possible when image-maps are linked to text. However, standard HTML neither provides an interactive graphics language nor the linking capabilities to support a bi-directional communication between text and image. One way around this problem is to create a client-server application in which text and image clients access the data objects through a server. The disadvantage of this approach is that the data and its context are hidden behind the server. Such a system maintains the authority of the author, as does a print document, presenting only the text and visual representation the author allows the reader to see. As a result, there is not only a loss of transparency to the information communication but a diminishment in the sense of collaboration the active visualization affords.

Ideally, a hypermedia document used for information communication should contain data, metadata, the descriptive information context, and methods for data transformation, interactive rendering, text formatting, linking, and dynamic binding of text with visualization. Today, this is possible with XML and its dynamic graphical language SVG (Scalable Vector Graphics) [9].

4. SVG

SVG is language created by Adobe and supported by the W3C Consortium that describes two-dimensional vector and mixed raster/vector graphics in XML. It is designed to integrate with XLink, XML Namespaces, DOM, CSS and XSL. Because SVG is an XML format, SVG graphics can be parsed using any XML compliant parser. SVG marks-up graphical data in the same way that XML and HTML mark-up text. And SVG can be generated dynamically using CGI Scripts, Java Servlets, and Server Pages.

SVG is similar to Java2D graphics [10], supporting rich graphical content with three types of graphic objects: vector shapes, images, and text. These objects can be grouped, transformed, composited, clipped, masked, and filtered. SVG drawings can be interactive and dynamic, with animations defined and triggered either by directly embedding SVG animation elements within SVG content or by means of scripts. Event handlers can be assigned to any SVG graphical object. Finally, because SVG is compatible with other Web standards, scripting can be done on XHTML and SVG elements simultaneously within the same Web page.

Because of SVG's newness and competiton from other technologies such as Macromedia Flash, few web browsers currently convert SVG code into images; Mozilla is an exception. As a result, a stand-alone viewer or browser plug-in is required. Adobe's SVG Viewer plug-in is one such program [11].

5. Demonstration

To demonstrate how image and text may be bound, a hypermedia document was created using SVG, CSS, HTML, and JavaScript. Figure 2 shows a simple mapping application containing a map and bar charts specified in SVG, and HTML-formatted text. The SVG map of the Middle East, shown in Figure 2, was created with GeoClient [12] and adapted to this application. The bar charts were created from an SVG demonstration at the Goldthorp Graphics website [13]. Text was gathered from the US embassy websites of the respective countries and statistics from the Students of the World website [14]. Data was selected for its availability. GeoClient was selected because it could output SVG code, and the Middle East map was one that was immediately available from the GeoClient website.

All components are organized within a table. All graphical (SVG) and text (HTML) objects in the document are active. For example, clicking on any country name (HTML) will cause the corresponding country (SVG) to be highlighted in yellow and surrounding countries to be rendered in blue (Figure 2b). Conversely, clicking on a country will cause the text corresponding to its name to be highlight in yellow and the loading of supplementary descriptive text below the country list. This new text is formatted in HTML and may contain additional links, as seen in Figure 2c. Here, embedded within a discussion of Jordan's climate, is a hotlink labeled "temperature" that, upon clicking, will load an interactive bar chart displaying the average monthly temperatures throughout the year. Placing the mouse over any one of these bars changes its rendering style from transparent to opaque and displays text within the chart specifying the name of the month and its average temperature. The map may be redrawn by clicking on any of the country names or the "Countries and Territories" heading.

Communication between HTML and SVG components is negotiated using JavaScript functionality contained within the SVG code. For example, to inform the map graphical object that the county "Jordan" has been selected from the list, a function called "setCountry('Jordan')" is attached to the "onclick" mouse event and included within the HTML table data tag <td> such that

```
<td onclick =
"maps.window.setCountry('Jordan')"
id="t6"> <em>Jordan</em> </td> .
```

As a result, when the "Jordan" list item is selected, the setCountry() function is invoked, receiving the name of the country. SetCountry() then goes through its list of graphical objects and finds the item with this country name, changes its fill attribute to highlight it in yellow, and changes the fill attributes of the remainder to render them in blue. In addition, setCountry() sends a text string back to this data tag (id="t6"), replacing the original content " Jordan " with "Jordan". Because the style tag's background color attribute has been modified in the HTML document's cascading style sheet (CSS), Jordan is highlighted in the country list.

A similar process occurs when a country is selected from the map. Each graphical object is linked with a mouse-click event that invokes the function notify('id') when the event is captured. Notify() highlights the country name in the same fashion as setCountry(), and delivers a block of HTML formatted text to a row in the table below the country list that is formatted with a style sheet that appends a scroll bar,

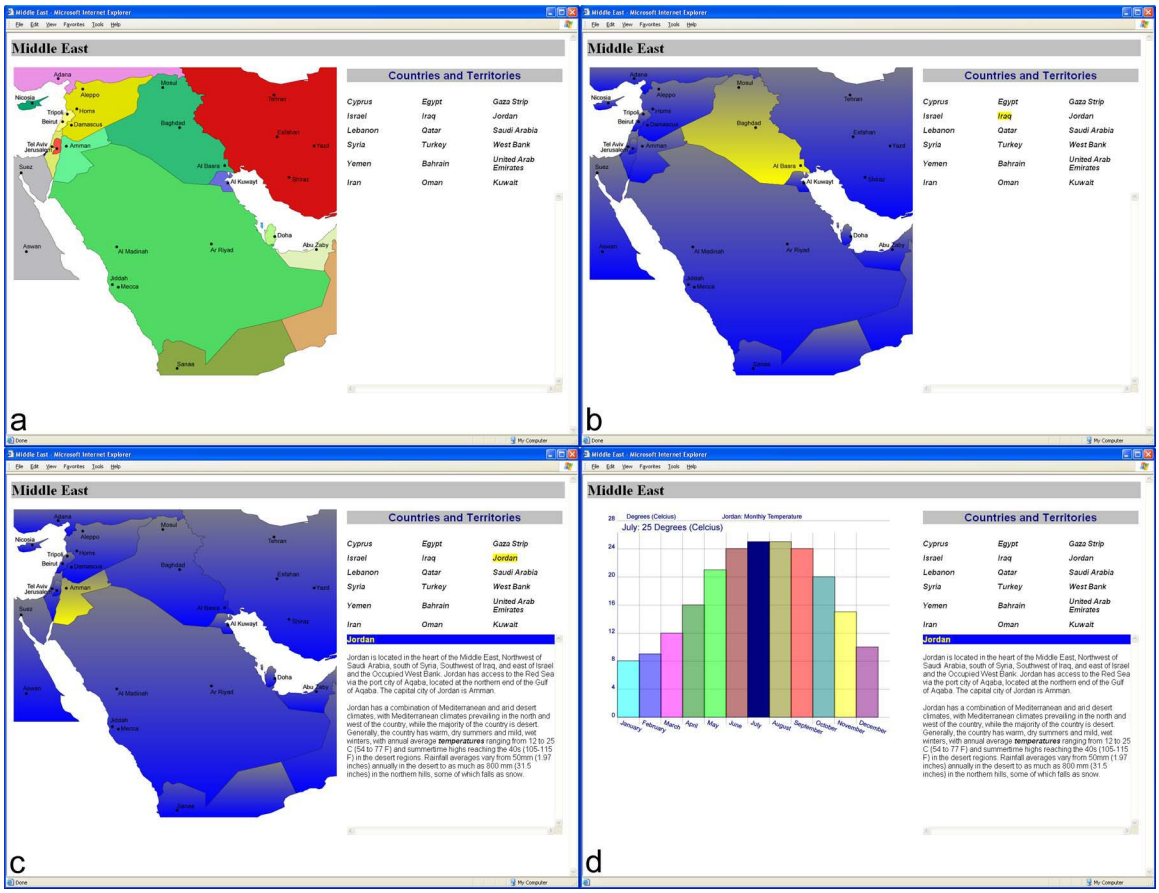

Figure 2. Simple mapping example: a) system state upon loading, b) country highlighted when country name is clicked, c) country name highlighted and descriptive text loaded when country is clicked, d) bar chart appears when the word "temperature" is clicked within loaded text.

sets text styles, etc. An active link may be embedded within the text. For example, within this text that appears below the country list (c.f. Figure 2c) is the following:

```
"…with annual average <em onclick=
\"bars.window.setPlot('bar')\" >tem-
peratures</em> ranging from…"
```

Here the emphasis tag has been modified to include clickable text ("temperature") that will load the temperature bar chart by invoking the setPlot() function. Once clicked, setPlot() will change the SVG map's visibility attribute to "hidden" and the bar chart's to "visible".

7. Discussion and Conclusions

The technical narrative found in journals and conference proceedings is a structured form in which text is responsible for communicating the research process and its significant results with static imagery providing a supporting role. By using hypermedia technology, in particular HTML, XML and SVG, it is possible to transform these documents so that dynamically changing text, visualizations, and the active linking among them should provide a richness and complexity of exposition not found in static media.

The simple mapping application demonstrates that a dynamic bi-directional binding between image and text spaces may be constructed employing SVG and XML that synchronizes representations so visual information and information context are consistent. This implies that data objects may be equivalently accessed and manipulated either by image or text. Hence, the concept of direct manipulation extends to the text representation of object data. In addition, by providing means for individual graphical components of a visualization to modify, enhance, or augment descriptive text, it is possible to create in-depth annotations beyond labels and pop-ups that do not intrude

on the graphical space of the visualization. Moreover, as shown in Figures 2b&c, these annotations can have their own formatting and links, including those, which control the original visualization.

Finally, by using SVG to construct the visualization component of the hypermedia document, the entire document's data, metadata, transformations, and informational content become reviewable. As a result, the communication component of the research process is enhanced.

8. Acknowledgements

The author would like to thank Mary Curtin for helpful discussions. This project was supported by Pace University's Center for Advanced Media.

9. References

[1] A.E Treloar, "Applying Hypertext and Hypermedia to Scholarly Journals Enables Both Product and Process Innovation," *ACM Computing Surveys* 31 (4es), 1999, pp.1-5.

[2] "Bimolecules at Kenyon," *biology.kenyon.edu/BMB/chime.htm* (current 28 March 2004).

[3] "Chime," *www.mdli.com/products/framework/chime/index.js,* (current May 3, 2004).

[4] M. Schapiro, *Words, Scripts, and Pictures: Semiotics of Visual Language*, George Braziller, New York, 1996.

[5] C. Ware, *Information Visualization: Perception for Design*, 2nd Edition, Morgan Kaufmann Publishers, San Francisco, 2004, pp. 297 – 316.

[6] A. Paivio, *Mental Representations - A Dual Coding Approach*. Oxford University Press, Oxford, 1986.

[7] L.J. Najjar, "Principles of Educational Multimedia User Interface Design, *Human Factors*, 40(2), 1998, pp. 311-323.

[8] C. Strothotte and T. Strothotte, *Seeing Between the Pixels*, Springer Verlag, Berlin, 1997.

[9] A.H. Watt, *Designing SVG Web Graphics*, New Riders, Indianapolis, IN, 2002.

[10] V.J. Hardy, *Java2D API Graphics*, Prentice-Hall, Palo Alto, CA, 2000.

[11] "SVG Viewer," *www.adobe.com/svg* (current 5 May 2004).

[12] "GeoClient," *www.mycgiserver.com/~amri/ geoclient.cocoon.xml* (current 28 March 2004).

[13] "Goldthorp Graphics," *www.goldthorp.com/pls/svg/gldthp_demo_form.show_form* (current 28 March 2004).

[14] "Students of the World," *www.studentsoftheworld.info,* (current 28 March 2004).