Published October 1998 in Bioinformatics **14**(9), pp 807–814

Zomit: biological data visualization and browsing

Stuart Pook[*†] Guy Vaysseix[*‡] Emmanuel Barillot[*‡]

Received on April 16, 1998; revised on July 7, 1998; accepted on July 9, 1998

Abstract

Motivation: *The problems caused by the difficulty in visualizing and browsing biological databases have become crucial. Scientists can no longer interact directly with the huge amount of available data. However, future breakthroughs in biology depend on this interaction. We propose a new metaphor for biological data visualization and browsing that allows navigation in very large databases in an intuitive way. The concepts underlying our approach are based on navigation and visualization with zooming, semantic zooming and portals; and on data transformation via magic lenses. We think that these new visualization and navigation techniques should be applied globally to a federation of biological databases.*

Results: *We have implemented a generic tool, called Zomit, that provides an application programming interface for developing servers for such navigation and visualization, and a generic architecture-independent client (Java™ applet) that queries such servers. As an illustration of the capabilities of our approach, we have developed ZoomMap, a prototype browser for the HuGeMap human genome map database.*

Availability: *Zomit and ZoomMap are available at the URL http://www.infobiogen.fr/services/zomit.*

Contact: *stuart@infobiogen.fr*

Introduction

The volume of data produced in molecular biology has been growing exponentially for several years and all signs indicate that this process will continue during the next decade. This growth applies to mapping and sequence data on everything from microorganisms to humans.

For several years now it has been difficult for biologists to interact directly with the data concerning their species of particular interest. The huge volumes and complexity of the data, and the numerous links between them, make it hard to maintain a global view of the data. Researchers are restricted to local views of their data, and long-distance relationships are not visible. Basically, there is much more information available than researchers can interact with. Therefore, the selection of information is based on interactive, but rather arbitrary decisions, or on the results of an algorithm that processes the data according to a pre-established and rigid model.

With the large-scale sequencing projects now producing data at a rapid pace, the time has come to work on extracting knowledge from rich sources of data, i.e. to discover new models. Data mining can be used, but is not sufficient in the sense that it can only exploit the data to test hypotheses. Researchers need to interact globally with the data, to make incidental discoveries, find unexpected regularities and test new putative models.

Finally, the scientific community suffers from a lack of interaction with the data, making the large-scale mapping and sequencing projects not as fruitful as they should be.

The traditional data visualization and browsing tools are not successful in addressing these problems because:

- they are based on the limited WIMP (windows, icons, menus and a pointer device) concept (van Dam, 1997);
- they do not offer a global view of the data;
- the links between distant pieces of data are not handled in a way that intuitively reflects their semantic proximity; and
- data are represented under a single perspective, although they could often be seen under several.

For example, when using a World Wide Web (WWW) browsing tool applied to molecular biology databases, following a link from a page generates a view of the new page that is visually independent from the view of the first page. There is no visual track of the semantic relation between the two pages. Therefore the user quickly forgets the logic of his sequence of different views, and gets lost. Often visualization software offers a view of the data according to a given perspective and modifying this perspective is not easy. These problems are general in most visualization and browsing software.

In this paper we propose a new metaphor for data visualization and browsing in molecular biology. The principles are presented in the next section. Some of them have already

[*]GIS Infobiogen, 7 rue Guy Môquet – BP 8, 94801 Villejuif cedex, France
[†]École Nationale Supérieure des Télécommunications, Département Informatique et Réseaux, 46 rue Barrault, 75634 Paris cedex 13, France
[‡]Généthon, 1bis rue de l'Internationale, 91000 Évry, France

been used in other domains, such as hypermedia representation [see Leung and Apperley (1994) for a review]. An application to the visualization and browsing of a molecular biology database is presented in the section on a virtual world to visualize HuGeMap. The implementation of the software is described in the section on the generic tool Zomit.

Visualization concepts

Our visualization scheme is based on the concept that the data are organized in a two-dimensional virtual world in which the user can travel and where he can focus on his area of interest (Furnas and Bederson, 1995). When the user goes closer to an object, details appear and the representation is modified. He can also easily go to and return from semantically related objects. He can focus on a specific area or zoom out to have a global view of the data. This is achieved using the techniques detailed in this section. A more complete presentation of zooming, portals and lenses is given in Bederson et al. (1996).

Zooming

The technique of navigation by zooming is easily applied to molecular biology data for two main reasons. Firstly, the large number of links between objects in molecular biology leads to a complex graph of semantic relations which is best navigated by zooming (e.g. when focusing on a given node, the neighbouring nodes can be represented with edges decreasing with the minimal path length). Secondly, biological databases frequently contain a well-defined hierarchy of information, from a genome to its sequence for example.

Traditionally, the displacements in graphical user interfaces are made by panning with a scroll bar to reach the area of interest and then zooming by entering the scale or by using another scroll bar.

We propose a different approach where a more natural zoom is the main method of displacement. The user zooms by clicking the first button of the mouse, and pans by moving the mouse while keeping the first button depressed. In our system, displacements are done by zooming out and in, and it is expected that the user will naturally only use panning to recentre his view. This is a deliberate choice because this navigation has been shown to be more efficient than extensive use of panning (Furnas and Bederson, 1995; Bederson et al., 1996).

An early version of Zomit recentred the view of the virtual world around the cursor with each zoom. This was found to be counterintuitive so the current version keeps the same point under the cursor, allowing the user to zoom rapidly on a point by continuous clicking.

When the user zooms in on an object, it is natural to see a more and more detailed view; i.e. as the size of the object increases, its representation is made more detailed. Conversely, the details get smaller and eventually vanish when the user zooms out. However, when zooming, it may be more appropriate to see a new representation more adapted to the scale. For example, at low scale, a chromosome should be represented according to its cytogenetic description, while at a very large scale the sequence is more pertinent. This is called semantic zooming.

Portals

A portal is a special graphical object that gives a view of another part of the virtual world. It is generally static (as in our current implementation); i.e. the user cannot create a new portal, only those provided by the system are available. It consists of a rectangle put at a given place in the world, in which another part of the world is displayed, usually at a different scale. As with other objects, portals grow as the user zooms, and when a portal almost fills the user's view, the main view can be said to be transferred to the view in the portal. The user can manipulate (zoom, dezoom, or pan) the view seen through a portal simply by clicking inside the portal.

Portals can be used to express the semantic relationship between two objects that are widely separated in the virtual world, even though they are related. A portal can be placed near the first object, pointing to the second one. If necessary, a reciprocal portal can complete the symmetry.

Portals are useful to avoid the duplication of objects, a technique that is convenient in graph representation but complicates the understanding of the graph structure.

Lenses

The user may be interested in transforming some part of the world; i.e. in viewing it under another representation. Magic lenses are dynamic items that allow the user to apply temporarily a predefined transformation to the area of their choice. These Magic Lens™ filters originated at Xerox PARC (Stone et al., 1994). A typical example of a magic lens is a rectangle which converts a table of numbers into the corresponding scatter plot.

Magic lenses can also be stacked to combine several transformations. This can be viewed as a particular and restricted case of visual programming. It is a very intuitive way to program, and thus is useful to biologists who master the biological concepts underlying the data transformation but are not familiar with programming.

Lenses can also be seen as portals that point to another part in the world where the representation is different. This adds a temporary fourth dimension to the two spatial dimensions and the scale 'dimension'.

A lens can also be a magnifying glass. This provides additional space in order to be able to display extra information on the objects. The disadvantage of this magnification is that the lens does not transform the entire region covered, but only

the objects in the centre of this region. An example of such a lens is shown in Figure 2a5.

An example of possible use of lenses in molecular biology (on protein data) data visualization is given in Robinson and Flores (1997).

A virtual world to visualize HuGeMap

As a demonstration of the use of the concepts outlined above, we developed a server, called ZoomMap, that queries the HuGeMap database. It provides a virtual world containing the human genome, chromosomes, maps, markers and sequences. ZoomMap was built using the generic tool Zomit. Other genome browsers are listed at http://www.cherwell.com/javagenomes and http://www.ebi.ac.uk/oib97.

The HuGeMap database

The HuGeMap database (Barillot et al., 1998) stores the major genetic and physical maps of the human genome. It includes (i) the genetic maps from Généthon (Dib et al., 1996) and the Cooperative Human Linkage Consortium (Sheffield et al., 1995), (ii) the physical maps from CEPH-Généthon (Bellanné-Chantelot et al., 1992; Cohen et al., 1993; Chumakov et al., 1995) and from the Whitehead Institute-MIT (Hudson et al., 1995), and (iii) the ISCN cytogenetic description.

HuGeMap is interconnected with the RHdb database (Rodriguez-Tomé and Lijnzaad, 1997). The interconnection is based on CORBA servers built on top of the two databases. CORBA is a specification of the Object Management Group (http://www.omg.org) that allows applications to communicate with one another no matter where they are located or who has designed them. This communication is defined by a single implementation language-independent specification, the IDL.

The schema of HuGeMap has a natural and strong hierarchy and contains a large number of links. On the top of the hierarchy is the human genome, then the chromosomes, their cytogenetic elements, the maps, the markers and finally the nucleotide sequences. Markers may belong to several maps. This database is, therefore, a good testbed for data visualization and browsing.

ZoomMap: zooming and portals

The world presented in ZoomMap consists at the lower scale of a karyotype of the human genome (Figure 1). The 24 chromosomes are depicted with no details at the top level (Figure 1a). As the user zooms on a chromosome, their arms and names appear, followed by the banding (Figure 1b and 1c). The names of the cytogenetic bands are shown when the bands are big enough to contain readable text. These steps are examples of semantic zooming (modification of the visualized object according to the scale). At the same time, more and more information on the chromosomes appears as text beside each chromosome. The names of the different maps associated with each chromosome are displayed and, if one continues to zoom, the maps themselves are drawn (Figure 1d) using their usual representation (markers positioned on an axis). Only those markers for which space is available are shown, another example of semantic zooming. When there is sufficient space a series of portals are shown opposite each marker (Figure 1e). They point to the other maps in which the corresponding marker is present. After further zooming on a marker, the sequence appears under the name (Figure 1f).

ZoomMap: magic lenses

Several different types of magic lenses have been implemented (Figure 2a1–5).

- Lenses that display a map when applied to a cytogenetic description of a chromosome (Figure 2a1 and 2a2). This representation of a chromosome is also given when zooming in further from the chromosome name. This illustrates the fact that a problem of data transformation can be addressed with different techniques: here magic lenses or semantic zooming. Lenses present the advantage of offering numerous possibilities simultaneously (one can have as many lenses as necessary and choose among them) while with semantic zooming there is only one possible transformation.
- A lens that, when applied to markers, uses the colour of each marker to show the level of heterozygosity (Figure 2a3). Here, a marker's heterozygosity is coded on a scale from black (100% heterozygous) to white (0% heterozygous).
- A lens that displays only those markers with high heterozygosity (>65%; Figure 2a4). Here the lens is used as a filter to select some data. Typically, such lenses can be combined to produce new filters on a conjunctive (logical AND) basis.
- A lens that gives further information on each marker: name, D-number, EMBL access number, and heterosygosity. This lens is a magnifying glass so as to have sufficient space to display the additional information (Figure 2a5).

As discussed in the previous section on lenses, these lenses can be combined as shown in Figure 2a3 and 2a4.

Zooming on indices

Our purpose is to visualize biological data, generally stored in structured databases, for which indices exist. (Indices are sorted lists of object identifiers.) Though the logic of navigation by zooming is based on a hierarchically organized data schema, it may be useful to browse through these indices to retrieve data.

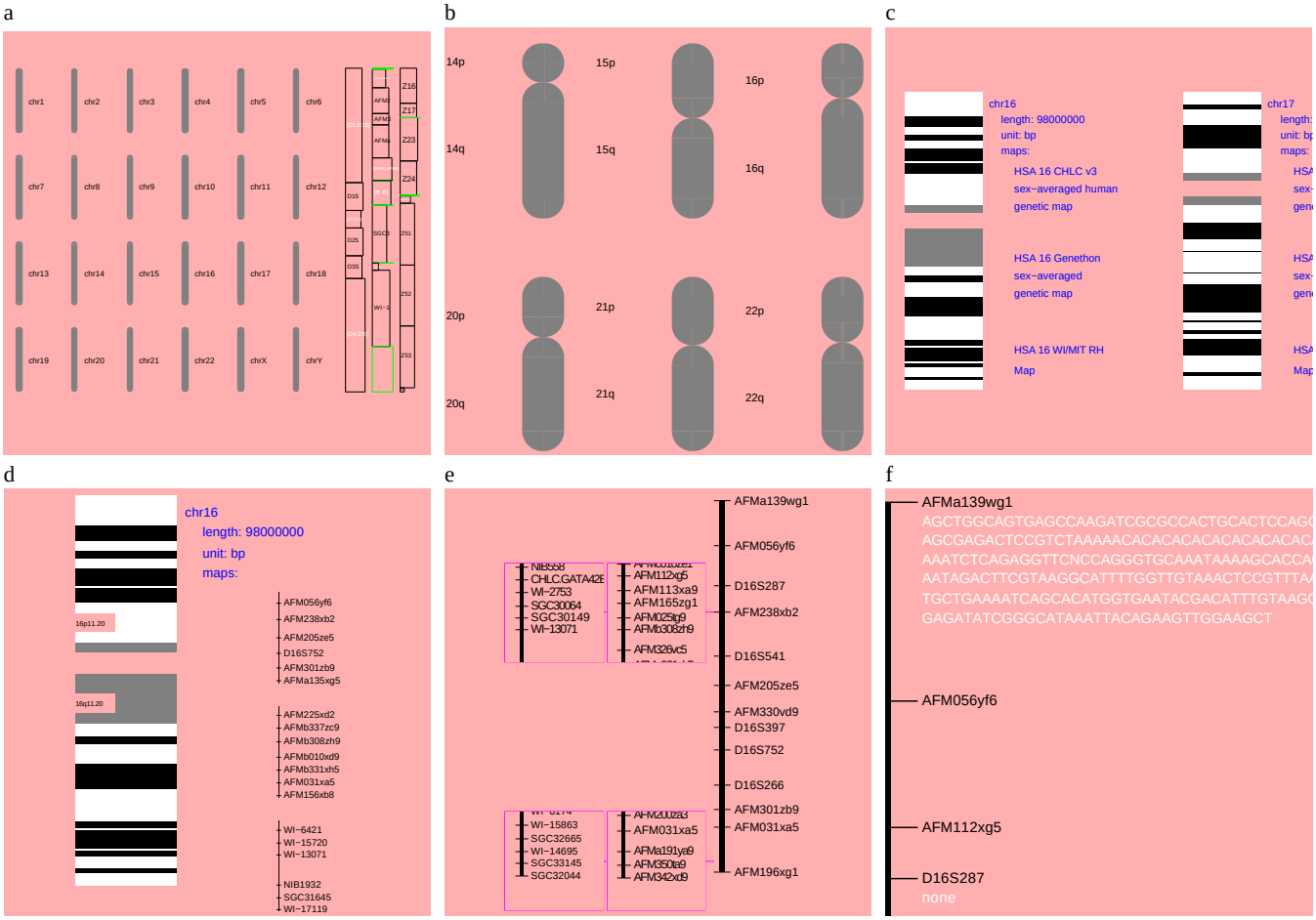

Fig. 1. ZoomMap: zooming from genome to sequence. (**a**) Outline karyotype; (**b**) chromosome arms appear; (**c**) banding and chromosome textual information visible; (**d**) and (**e**) maps are displayed, with marker names and portals when there is enough space; (**f**) marker sequence is shown.

In ZoomMap the user can zoom on three different indices (D-number, EMBL access number, and name) that are present on the right-hand side of the world at the beginning of a session (Figure 1a).

At the beginning, all the possible first letters of the indexed fields are displayed in order and with a size reflecting the number of objects starting with each letter. After zooming in on the desired first letter, the second and following letters appear as soon as space is available, or a range of strings if all the possibilities would span too much space (Figure 2b1). Different colours encode the type of the string shown; i.e. the current position in the index, string ranges, and complete marker names.

This repeated zooming process leads to the complete name of the desired object (Figure 2b2 and 2b3), and then to a series of portals that point towards the position of the marker in the maps in which they are present (Figure 2b4).

The ZoomMap server

To display the data in the way described above, the ZoomMap server queries HuGeMap through a CORBA server. Any genome map database implementing the same CORBA interface can be visualized by the current version of ZoomMap. ZoomMap must be considered a prototype and not as an industrial product. Zooming or using lenses in ZoomMap may be relatively slow because we gave preference to functionalities and did not optimize the run time.

The generic tool Zomit

The Zomit package is a generic tool that allows the developer to create visualization environments (virtual worlds) incorporating the concepts described in the section on visualization concepts.

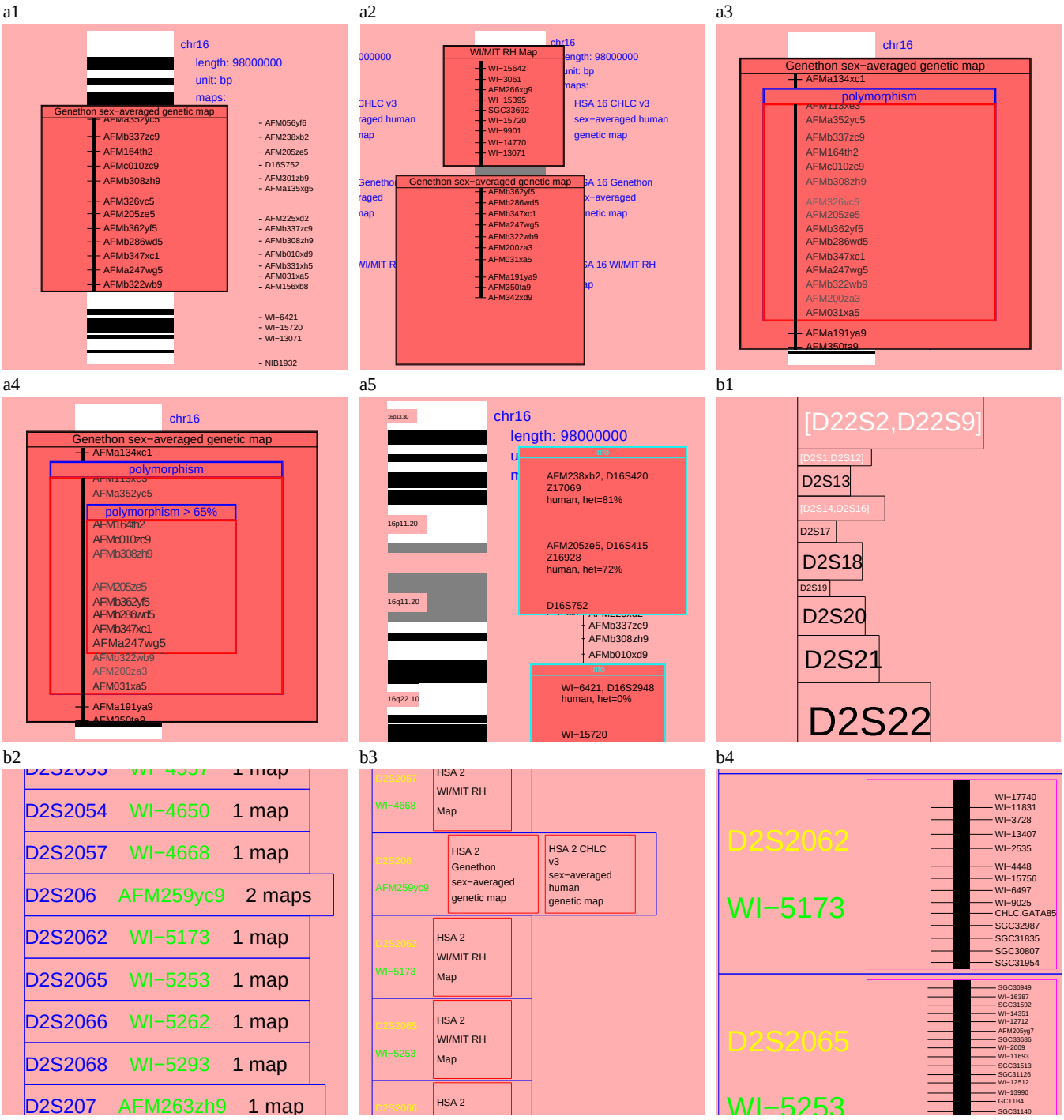

Fig. 2. ZoomMap: several lenses. (**a1**) Transformation of the cytogenetic representation of a chromosome into its Généthon genetic map; (**a2**) as for (a1) with two lenses; (**a3**) combination of the first lens and a lens that encodes the marker heterozygosity on a grey scale; (**a4**) as for (a3) with a third lens that selects only the markers with heterozygosity $> 65\%$ (the marker AFM326cv5 is no longer shown); (**a5**) magnifying lens giving more information on a marker. ZoomMap: zooming on indices. (**b1**) Choosing the first letters; (**b2**) the name is complete; (**b3**) marker names and related maps; (**b4**) portals to the positions of the markers on each map.

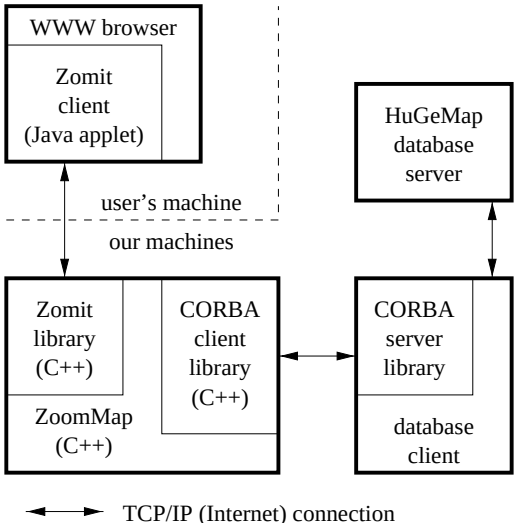

Fig. 3. How Zomit is used by ZoomMap.

Architecture

Zomit is based on a client-server architecture as illustrated in Figure 3.

The Zomit client is a Java applet that is executed by the user's browser. Therefore, Zomit is independent of the computer architecture from the user's point of view. Some of the most common browsers are supported, including the most recent versions of Netscape on personal computers and Unix systems. The client handles all the user's interactions and communicates with the server to fulfil the user's requests. It sends the server the user's position in the virtual world and the server responds with the graphical objects that should be visible in this area. These graphical objects are simple for the client to draw and compact in transmission time. The graphical items currently provided are lines, (possibly rounded) rectangles, text, and portals. The list of available lenses is communicated by the server to the client. The client handles the creation and display of the lenses.

The Zomit client is completely generic. The same client can be used to talk to any server and thus display any virtual world. The only restriction is that imposed by the browsers: an applet can only communicate with the server from which it was downloaded. If the user installs the applet as an application, this restriction no longer applies and the same client can be used to communicate with any Zomit server on any machine.

The server is a C++ program in two distinct parts. The first part is the generic Zomit library that handles all communication with the Zomit client, stores graphical objects that are not yet required by the client, and calls the code provided by the implementer of the virtual world. The implementer has to provide functions that are registered with the library as covering certain regions of the virtual world. When the user enters into a region covered by a function, the library calls the function. The function can generate graphical objects that are to be communicated to the client and can register other functions associated with sub-regions of the region covered by the function.

Developing a Zomit server

The Zomit client and the Zomit library allow the developer to create a two-dimensional multiscale world for the visualization of their data. At any moment, the user sees a certain region of the world at a certain scale. The user can pan, in other words move to another region while keeping the same scale, or zoom or dezoom. When the user zooms, the scale increases and he sees a smaller area. When he dezooms, the scale decreases and a larger area of the world is visible.

The developer of the user's world has complete control over what is shown to the user at each point and at each scale. He will normally create graphical objects that implement a semantic zoom possibly including portals with their view of a different part of the world. Each graphical object has a colour, a position and a range of scales in which the object is to be visible. This range of scales allows the developer to create an object, to have it grow on the screen as the user zooms on it, and then to have the object disappear as it is, presumably, replaced by other objects. The position of a piece of text is just its starting position, and that of a rectangle, the coordinates of its corners.

The names of the lenses available to the user are defined by the developer. When the graphical objects for a region are being created, the developer can specify that a given graphical object should be visible in lens zero. They can also specify a different graphical object for lens one, another object for the combination of lenses zero and one, and yet another object for the combination of lenses one and zero. It is also possible to specify simplifying rules, indicating, for example, that the stacking order of lenses zero and one is not important, or that the combination of lenses zero and two should show the contents of lens zero.

Portals have the attributes of a simple graphical object plus an initial position and scale for the region that they show. The client controls the display of the portals, no further programming by the developer is required. The user can zoom, dezoom, and pan in the portals, and the current position is remembered by the client even when the portal is not visible. Lenses can also be applied to the objects visible in portals, but cannot themselves contain portals.

Conclusion and perspectives

Improvements

The prototype ZoomMap gives an idea of what can be done with the concepts outlined in this paper. There are three main directions in which Zomit can be improved rapidly (i.e.

without developing additional ideas).

The Zomit client and library need to be optimized. After the user has navigated in the world for a time, the client stores a large number of graphical objects and must search through this list when redrawing the screen. It may even be necessary for the client to discard objects. In this case the server will have to be notified so that the objects can be resent if required.

The Zomit library is currently single threaded. Once a request for the objects in a region is received, the server does not look for new requests until the current request has been completely satisfied. This is inefficient because a new request might indicate that the user has moved on (normally by zooming) and is thus no longer interested in the results of the original request.

At the moment, zooming is by discrete steps to reduce the computation load on the user's machine. When faster hardware is more universally available, a more fluid and natural continuous zoom will be implemented.

Other improvements will come from a better use of the concepts already implemented. Currently, lenses are empty when placed over a region where no transformation has been defined. Lenses should be defined so that when in such a region they display the lens's instructions, including where they should be used. In addition, when the user arrives in a region that is empty, he should be directed to something interesting.

In the current implementation, the user has only one view of the world. He needs to be able to clone his view of the world so that he can keep a view of something that he has found whilst looking for something else, or to be able to search in two regions simultaneously.

Minor improvements, such as allowing the user to resize lenses, will add to the user's comfort of utilisation.

New visualization concepts

The metaphors used by Zomit are in contrast with those provided by standard WWW browsers. Zomit could, however, be used to write a WWW browser by presenting links as portals containing a glimpse of each link's destination. This glimpse of the destination of the link is a feature missing in standard browsers.

One of the main aims of the Zomit project was to allow the user to keep the context of the area that he is currently focused on. An additional screen needs to be provided. This screen will show the entire three dimensional world (two dimensions plus the scale) that the user has at his disposal and the path that he has taken while traversing it. The screen will be a space-scale diagram that shows the world as a cone (Furnas and Bederson, 1995).

We plan to allow the user to create portals fixed to his screen. These sticky user portals will allow him to create a view of an area that interests him and keep it on the screen while he moves or zooms to other areas. These portals will be an alternative to cloning the current window and different to those described in Bederson et al. (1996), where the sticky portals are created by the system and become sticky only when used.

The virtual world made available to the user will normally be large and thus the user will want to be able to return to interesting regions that he finds. We plan to allow the user to create his own world by copying interesting regions into it or by making portals that point to interesting regions. These regions will, of course, remain zoomable. The user will then be able to save this personalized world.

Application to a federation of biological databases

One of the main challenges of bioinformatics is the interoperation of molecular biology databases. Envisaged solutions range from data warehouses to object request brokers with a communication layer on top of each database (Davidson et al., 1995; Achard and Barillot, 1997). This interoperation is crucial for the advancement of genetics and molecular biology.

Our intention is to implement a new Zomit server that will allow the visualization of a federation of biological databases. This server will query the databases of the federation through a CORBA interface. It will also be capable of communicating with other Zomit servers, and to include in its world some parts of worlds from these other servers.

Then the new concepts presented in this paper are expected to show their relevance.

Acknowledgments

We would like to thank all our colleagues at the GIS Infobiogen for their support and help in our work. We would particularly like to thank Christophe Cussat-Blanc, who implemented our CORBA server, and Philippe Gesnouin our system engineer. Thanks also to Eric Lecolinet for his helpful comments on the text. This work was supported in part by the European Union contract BIO4-CT96-0346.

References

Achard, F. and Barillot, E. (1997). Ubiquitous distributed objects with CORBA. In Altman, R., Dunker, K., Hunter, L., and Klein, T., editors, *Pacific Symposium on Biocomputing '97*, pages 39–50, Maui HI, USA. World Scientific, Singapore.

Barillot, E., Guyon, F., Cussat-Blanc, C., Viara, E., and Vaysseix, G. (1998). HuGeMap: a distributed and integrated human genome map database. *Nucleic Acids Res.*, 26:106–107.

Bederson, B. B., Hollan, J. D., Perlin, K., Meyer, J., Bacon, D., and Furnas, G. (1996). Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. *J. Vis. Lang. Comput.*, 7:3–32.

Bellanné-Chantelot, C. et al. (1992). Mapping the whole human genome by fingerprinting yeast artificial chromosomes. *Cell*, 70:1059–1068.

Chumakov, I. M. et al. (1995). A YAC contig map of the human genome. *Nature*, 377(Suppl.):175–298.

Cohen, D., Chumakov, I., and Weissenbach, J. (1993). A first generation physical map of the human genome. *Nature*, 336:698–701.

Davidson, S. B., Overton, C., and Buneman, P. (1995). Challenges in integrating biological data sources. *J. Comput. Biol.*, 2:557–572.

Dib, C. et al. (1996). A comprehensive genetic map of the human genome based on 5,264 microsatellites. *Nature*, 380:152–154.

Furnas, G. W. and Bederson, B. B. (1995). Space-scale diagrams: understanding multiscale interfaces. In *CHI '95 Human factors in computing systems*, pages 234–241, Denver CO, USA. ACM Press.

Hudson, T. J. et al. (1995). An STS-based map of the human genome. *Science*, 270:1945–1954.

Leung, Y. K. and Apperley, M. D. (1994). A review and taxonomy of distortion-oriented presentation techniques. *ACM Trans. Computer-Human Interaction*, 1:126–160.

Robinson, A. J. and Flores, T. P. (1997). Novel techniques for visualising biological information. In *ISMB 97*, pages 241–249, Halkidiki, Greece. AAAI Press, Menlo Park CA, USA.

Rodriguez-Tomé, P. and Lijnzaad, P. (1997). The radiation hybrid database. *Nucleic Acids Res.*, 25:81–84.

Sheffield, V. C. et al. (1995). A collection of tri- and tetranucleotide repeat markers used to generate high quality, high resolution human genome-wide linkage maps. *Hum. Mol. Genet.*, 4:1837–1844.

Stone, M. C., Fishkin, K., and Bier, E. A. (1994). The movable filter as a user interface tool. In *CHI '94 Human factors in computing systems*, pages 306–312, Boston MA, USA. ACM Press.

van Dam, A. (1997). Post-WIMP user interfaces. *Commun. ACM*, 40:63–67.