

# Context and Interaction in Zoomable User Interfaces

Stuart Pook<sup>1,2,4</sup>

Eric Lecolinet<sup>1</sup>

Guy Vaysseix<sup>2,3</sup>

Emmanuel Barillot<sup>2,3</sup>

<sup>1</sup> École Nationale Supérieure des Télécommunications, CNRS URA 820, 46 rue Barrault, 75013 Paris, France

<sup>2</sup> Infobiogen, 7 rue Guy Môquet – BP 8, 94801 Villejuif cedex, France

<sup>3</sup> Généthon, 1 bis rue de l'Internationale, 91000 Évry, France

<sup>4</sup> stuart@acm.org

## ABSTRACT

Zoomable User Interfaces (ZUIs) are difficult to use on large information spaces in part because they provide insufficient context. Even after a short period of navigation users no longer know where they are in the information space nor where to find the information they are looking for. We propose a temporary in-place context aid that helps users position themselves in ZUIs. This *context layer* is a transparent view of the context that is drawn over the users' focus of attention. A second temporary in-place aid is proposed that can be used to view already visited regions of the information space. This *history layer* is an overlapping transparent layer that adds a history mechanism to ZUIs. We complete these orientation aids with an additional window, a *hierarchy tree*, that shows users the structure of the information space and their current position within it. Context layers show users their position, history layers show them how they got there, and hierarchy trees show what information is available and where it is.

ZUIs, especially those that include these new orientation aids, are difficult to use with standard interaction techniques. They provide a large number of commands which must be used frequently and on a changing image. The mouse and its buttons cannot provide a rapid access to all these commands without new interaction techniques. We propose a new type of menu, a *control menu*, that facilitates the use of ZUIs and which we feel can also be useful in other types of applications.

## 1. INTRODUCTION

Zoomable User Interfaces (ZUIs) are no longer new and their theoretical bases [7] and practical applications [1] have been discussed in various papers. When using a ZUI, the users are presented with a view of an information space. The initial (or top level) view shows the entire information space at a scale which allows it to fit on the users' screen. The users can then zoom (or enlarge) a section of the view that they find interesting. The graphical objects will get bigger until, as soon as there is enough room on the screen, they are replaced by other graphical objects showing the underlying information in more detail. This is called semantic zooming. We developed a ZUI for browsing the HuGeMap [14] database of the

major genetic and physical maps of the human genome. This ZUI was used as a testbed for the new techniques described in this paper. For the purposes of this paper it is sufficient to know that the top level view in the ZUI shows 24 chromosomes, these chromosomes each have three genetic maps, and these maps consist of a large number of genetic markers positioned along an axis. The first three images on the Colour Plate show what a user would see in zooming from the top level view (image 1), to a view of the bands and maps on chromosome 10 (image 2), and then to a view of the Généthon map on this chromosome (image 3).

ZUIs are used to present an information space to users. One of the reasons that users are unable to successfully use ZUIs is that the view of the information space shown to users, or the focus, does not always contain the context needed by users to position this focus in the information space. Once users are in this situation they are disoriented, sometimes to the point of not understanding what they are looking at and not knowing whether they should pan, zoom, or dezoom to find what they are looking for. It could be said that they are 'lost in hyperspace.'

We present two new temporary aids that the system can display at the users' command if they arrive in this situation. The first, a *context layer*, allows users to position the focus with respect to more global views of the information space. The second, a *history layer*, allows users to revisit the route they took through the information space to arrive at their present position. We also present a third aid, a *hierarchy tree*, that is always visible. This aid is a second window in the ZUI and shows at all times the structure of the information and the users' current position within this structure. Users can also use this window to change their position in the information space.

ZUIs are complex programs with complex user interfaces that are normally controlled using the mouse, buttons and standard menus. When navigating in a ZUI, users zoom, dezoom, scroll, create magic lenses, move and resize magic lenses, move and scrolls portal, etc. Some of these actions are executed very frequently. A user zooms until the desired scale has been obtained and scrolls until the object looked for has been found. The graphical objects used to present the information space change frequently; a zoom or dezoom can completely change the objects visible. Making these objects active and using them to control the ZUI is problematic because these objects change and move too frequently. We propose a new type of menu, a *control menu*, that allows users to control ZUIs in a consistent and rapid fashion. A control menu can also include up to two scroll bars; a single interactor can thus control a complex operation. A ZUI incorporating these new techniques can be tested over the Web at the URL <http://www.infobiogen.fr/services/zomit/>.

## 2. CONTEXT LAYER

In ZUIs users can only see one view at a time: the focus. Users often

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.  
AVI 2000, Palermo, Italy.

© 2000 ACM 1-58113-252-2/00/0005..\$5.00

cannot understand where the focus fits into the information space because it only shows a limited region of this space and ignores the surrounding context. Fisheye views [6] are one way of integrating the context and focus into a single view. Some of the information surrounding the focus is shown following the rule: the greater the distance of the information from the focus the more interesting it must be for it to be shown. The Document Lens [15] allows the user to focus on a part of a document while keeping the surrounding pages (the context) visible. Three dimensional pliable surfaces [2] and hyperbolic displays [12] relate the screen area given to showing non-focus information to the distance it is from the focus. The further information is from the focus the less interesting it is assumed to be and thus the smaller it is shown. These methods deform the information space by eliminating information or by changing the size and position of the representations of information.

In contrast to the above methods that integrate the context and focus by deformation, we integrate the context and the focus by drawing the context (the context layer) over the focus. These two views are transparent so that they can both be seen at the same time. User studies [8] have shown that transparent views and overlays are well accepted by users.

When using a context layer two orthogonal controls are available: the scale of the view shown in the layer and the relative levels of transparency of the context layer and the focal view. The scale of the context layer can be chosen so that the context layer shows any view between the initial view and the focus. Users can also control the relative transparencies of the context layer and the focus. This allows users to concentrate on either the context or the focus by making the chosen view be drawn solid and the other as transparent as desired. The interactor used to control the scale and transparency is described in section 5.

The context layer is positioned so that the region of this layer that corresponds to the focus is in the centre of the ZUI's window. This region is indicated by a rectangle drawn in the centre of the window that shows the size and position of the focus relative to the view currently visible in the context layer.

This method of combining the focus and the context avoids the deformation of the other methods described above, which often makes images difficult to recognize and understand. The advantage of deformation, that more information can be represented, is maintained by a very fluid and rapid control of the scale of the context layer. The users can quickly find the context that they need to identify the focus. Changing the scale of the context layer changes the position and often the form of the objects in the layer. This movement helps users understand which objects belong to the context layer and which to the focus. The context layer is temporary and only shown during the gesture used to create and control it. This avoids overloading the screen with context information or allocating valuable screen real-estate to context information when it is not needed. The view of the focus shown in image 3 on the Colour Plate shows a view that the user might see after having navigated for a while in the ZUI. This view does not contain any clues that would let the user know what map or chromosome is visible. In this situation the user can ask the ZUI to show the context layer. The context layer contains the initial view or context (image 1) and is drawn on top of the focus (image 3) giving image 6. The position of the focus is indicated on the context layer by a green rectangle. In image 6 this rectangle covers the text '10q' and tells the user that the focus is showing the chromosome 10. In image 7 the user has zoomed the context layer so that it shows the names of the genetic maps (the focus never changes during the use of the context layer). The green rectangle showing the position of the focus covers the name of the Généthon genetic map. The focus is thus showing this map. Using

the context layer the user has been able to position the focus in two different contexts.

When the context layer is visible the user can choose to concentrate on either the focus or the context by changing the relative transparency of these two views. The relative transparency can be continuously adjusted from a state where only the focus is visible to a state where only the context is visible. Image 8 is similar to image 7 except that the user is now concentrating on the context and has faded out the focus. The rectangle that shows the position of the focus is always visible and the user can see more clearly that the focus is currently showing the Généthon genetic map.

Transparent overview layers [4] are a different type of display that differs from ours in that: (1) their layer is permanent while ours is a temporary orientation aid; (2) the transparency level of their layer cannot be changed by the user; (3) their layer always shows the top level view of the information space while ours can be used even if there is no top level view; and, (4) their layer can be used to move or modify the objects in the information space while our ZUI does not allow objects to be manipulated in this way.

### 3. HISTORY LAYER

Context layers allow the user to find the answer to the question 'where am I?' Another important question is 'how did I get here?' ZUIs need a history so that the user can return to previously visited regions of the information space and see these regions in relation to the focus and the top level view. We propose a transparent and temporary *history layer* that allows the user to move interactively along the path taken in the ZUI. As with the context layer, the history layer is temporary so as not to overload the screen and it disappears when the user releases the mouse button at the end of the gesture used to create it.

The path taken by the user in the ZUI is a sequence of views of the information space. The first view is the initial (or top level) view (image 1 on the Colour Plate) and the view on the screen is the last current view (image 5). All the views (called the historical views) seen by users are stored in this sequence. Images 2 and 3 are historical views that the user has seen in going from the top level view to the last current view. The history layer is drawn over the top level view (giving image 4) and contains a view that can be varied by users from the last current view, via the all the historical views in order, to the initial view (and back again). The user can thus interactively 'go back in time' and see the evolution of the current view in relation to the top level view. The comparison is done directly because transparent views are used so as to show the top level view and the historical view simultaneously. This comparison is also aided by the rectangles, drawn in two different colours, that show the sizes and positions, relative to the top level view, of the last current view and the historical view. The interactor used to control the history layer is described in section 5.

As with the context layer, the relative transparency levels of the history layer and the top level view can be adjusted so users can concentrate on the history layer or on the top level view. The two rectangles showing the position of the current view and the last current view are always drawn solid and are not affected by the level of transparency.

The current implementation of history overlays requires a top level view. This may not exist in systems where users can dezoom from a view of their own files to a view of, potentially and for example, the whole Internet. The scale of the top level view could also be so different from that where the user is currently working that users are unable to see changes to the positions of the current and historical views. We are currently investigating whether the system should

choose a different view to replace the top level view and whether (and how) users can control this choice.

#### 4. HIERARCHY TREES

ZUIs are often used on hierarchically structured datasets. Creating an information space for a ZUI requires the developer to provide graphical objects visible in the top level view of the space that summarize those objects found when users zoom. These objects will then summarize those objects to be found as users continue to zoom. A hierarchy is thus created. Objects in the information space that are not accessible via objects visible in the top level view will be hard to find by users because they will have no way of knowing where to zoom to find them. The critical zones technique [10] can be used to automatically provide the clues that there are objects to be found by further zooming.

The techniques presented in the previous section help users to understand the information space from the top level view to the current position. However, users do not know what is in other parts of the information space and in particular what is to be found by further zooming. The user is also unable to use the hierarchy to navigate in the information space. Users looking at the details of an object are unable to rapidly dezoom to see the entire object and cannot easily move from a sub-object to another sub-object of the same type.

ZUIs are three dimensional spaces [7] and the scale is the vertical dimension. The main view in a ZUI shows a horizontal slice through the information space. We propose a second orthogonal view, called a *hierarchy tree*, that is a view of a flattened vertical slice through the information space. The names of those objects above the current position of the user in the space are shown. Objects also have types (an object can be a chromosome, a map, a sequence, etc). If the information space is highly regular, the entire hierarchy of types can be displayed in the *hierarchy tree*, otherwise just that part of the hierarchy centred on the user's current position. This hierarchy tree will thus show users the structure of the information space, what information is available, where that information is located and how to find it.

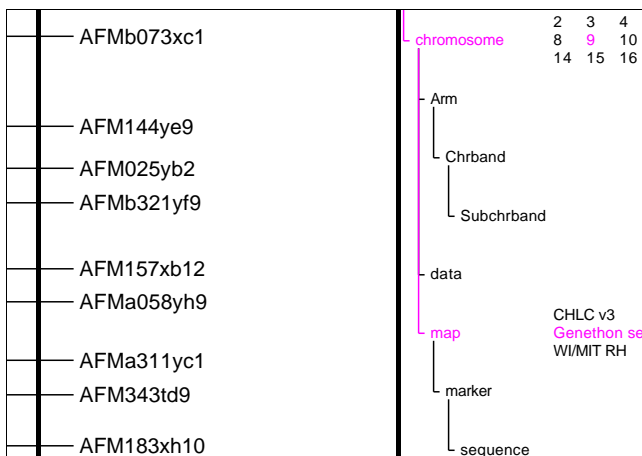


Figure 1: hierarchy tree

Figure 1 shows part of the main view of the information space plus the hierarchy tree. In this regular information space, the chromosomes are visible on the top level view of the information space. The chromosomes consist of arms, data and maps. This structure is displayed as soon as the ZUI starts. The user's current position in the structure is shown in magenta (or light gray); in this example

the user is currently looking at the 'Généthon' map on the chromosome 9. The structure indicates that if the user continues to zoom on the map, it will be possible to find the markers' sequences.

#### 4.1 Navigation

This technique offers an efficient method for rapid navigation as users can use the hierarchy view to navigate in the information space and to directly access related but currently invisible objects. If the user clicks on the string 'chromosome' shown in Figure 1, the ZUI will dezoom sufficiently to show all of chromosome 9. Users can also click on the string 'CHLC v3' (the name of the map next to the 'Généthon' map on the chromosome 9) to move to map CHLC v3. In this case, the ZUI will show the same relative position on the new map.

#### 4.2 Similar techniques

Excentric Labeling [5] offers a way of identifying objects on the screen. This technique labels, with 'tool tips' in the main view, those objects located around the cursor. We propose a different, non intrusive way, of identifying the object currently under the cursor. As users move the cursor across the main view the hierarchy tree is updated to show the type and name of the object under the cursor. If the cursor leaves the window or is not on an object, the hierarchy view indicates the lowest level in the hierarchy to which all the objects in the main view belong. Our method of labelling remains similar to 'tool tips' in that the user does not have to ask for the information to appear.

The gIBIS system [3] provides a global view of the displayed IBIS graph structure that is in some ways similar to our hierarchy trees. Their global view shows the subject of all the nodes in the network organised by their primary link. As users zoom or pan the local view of the network, the global view scrolls to show the users' their current position in the network. The global view can also be used to navigate in the network. Their global view does however differ from our hierarchy trees in that it shows all the nodes in the network. This means that at any moment only a small proportion of its contents are visible and a scroll bar must be used to navigate within the global view. ZUIs typically contain a very large number of objects and so a global view of all the objects in the information space would be so big as to be ineffective. As discussed above, our hierarchy trees are designed to make use of the structure present in many ZUIs and thus show the types of the objects in the information space and the names of only that object under the cursor and its ancestors in the hierarchy.

#### 4.3 Evaluation

To aid the evaluation of the visualization techniques proposed in this paper we created a modified version of the ZUI without the hierarchy trees. Eight subjects, chosen from our colleagues at Infobiogen, were taught how to use our ZUI. The subjects were asked to answer 22 multiple choice questions. A training session explained how to answer these question with and without the hierarchy trees. This experimental design allowed us to study, using two interfaces otherwise as similar as possible, whether the hierarchy trees were of assistance.

Ordering effects were taken into account. Half of the subjects answered their first 11 questions with the hierarchy trees and the other half of the subjects answered their first 11 questions without the hierarchy trees. Half of the subjects were given the first 11 questions in the list of 22 to do first while the other half of the subjects did the second half of the questions first. This lead to four equally sized groups of subjects.

For each subject we calculated the time taken to answer 11 questions without the hierarchy trees divided by the time taken to an-

swer the other 11 questions with the hierarchy trees. A value greater than one from this calculation would mean that having the context aids was an advantage. The mean value was 1.58 with a standard deviation of 0.54. The high standard deviation was caused by the lack of familiarity of some of the subjects with ZUIs. For these people the training session was not long enough and they thus found the second set of questions easier. In general however the subjects were faster with the hierarchy trees and were positive in their comments regarding these aids: in fact those that started with the hierarchy trees were reluctant to continue the experiment without them. The other new techniques presented in this paper are currently being evaluated.

## 5. A NEW INTERACTOR

Standard menus such as pull-down, pop-up and marking menus [11] allow actions to be selected. Pop-up and marking menus are contextual: they are activated at a location chosen by users. The program can thus adapt the contents of the menu to the objects found at that position and apply the action chosen to those objects. Pop-up and marking menus do not allow the operation to be controlled once it has been chosen from the menu (a scroll operation cannot be completed using just a standard menu) and do not allow users to supply the parameters required by the operation. An operation such as a font size change in a word processor often requires a dialog box to provide the new size. The users must select the operation using the menu and then concentrate on a second interactor. Once the new size has been entered the dialog box disappears and the users must reconcentrate on the main task.

Panning in a ZUI requires either a dedicated mouse button so that users can drag the image or two scroll bars. It is not possible to pan using a standard menu except with difficult to use commands such as 'move a little to the left.' Zooming is also difficult to perform with a standard menu as users want to zoom until they reach the required scale. Standard menus only allow users to zoom by fixed steps and that only by repeated use of the menu.

### 5.1 Control menu

We propose a new form of pop-up menu, called a control menu [13], that can integrate up to two scroll bars or spin-boxes. With

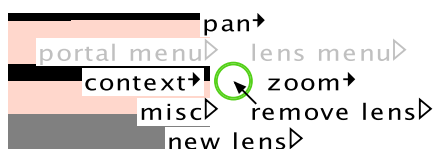


Figure 2: the control menu in our ZUI

this menu users can choose an operation and control it or supply parameters with a single gesture.

A control menu works somewhat like a marking menu. The novice user presses the mouse button and waits (0.3 seconds) until the menu appears under the cursor and then moves the cursor in the direction of the desired operation. The menu disappears and the operation starts as soon as the cursor has been moved the *activation distance* from the centre of the menu. (We have empirically chosen an activation distance of five times the radius of the circle in the centre of the menu.) The operation finishes when the user releases the mouse button. A user that has learnt the position of the desired operation does not have to wait to see the menu and can move the cursor immediately. The gesture is otherwise the same. This user has thus learnt to be an expert and is not distracted by the now unnecessary menu.

### 5.2 A control menu integrating a scroll bar

A control menu can be used to modify the scale in a ZUI. This operation is an example of the integration of a menu and a single scroll bar. Figure 3 shows the mouse movements during the use of

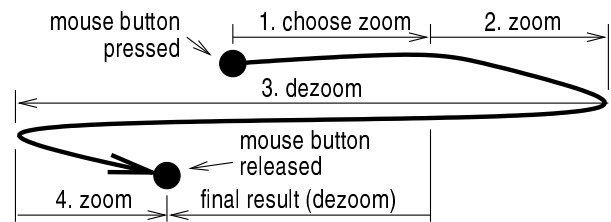


Figure 3: zooming with a control menu

a control menu to choose the zoom operation and to then control the zoom. The user presses the mouse button and moves it the activation distance (movement 1) towards the right (as the zoom operation is on the right of the control menu in Figure 2). This selects the operation and the cursor changes on the screen. From this moment until the mouse button is released, mouse movements to the right (movements 2 and 4) zoom the view and movements to the left (movement 3) dezoom it. The feedback is immediate: the view changes as the user moves the mouse. The user releases the mouse button once the desired scale has been obtained (a dezoom in this example).

During the zoom operation, the user can undo the current zoom by moving the mouse up or down a large distance. The user can then confirm the undo by releasing the mouse button or can undo the undo by moving the mouse back towards the centre of the display. In this case the user still has the mouse button pressed and can continue the zoom.

### 5.3 A control menu integrating two scroll bars

A control menu can also be used to perform two dimensional pans. It thus replaces two scroll bars. The pan operation is selected by pressing the mouse button and by moving the mouse up (the pan operation is at the top of the control menu in Figure 2). The view is dragged by the mouse during the operation. It is not possible to undo this operation as it is possible to undo a zoom because all the movements of the cursor already have a meaning.

### 5.4 Control menus and simple commands

A control menu can contain simple commands that do not have any parameters. Commands that can be cancelled are executed as soon as the cursor has been moved the activation distance from where the mouse button was pressed. As the user still has the mouse button pressed, a movement in the other direction undoes the operation. The undo can be undone by moving the cursor back again. The user releases the button when the desired result has been obtained.

### 5.5 Marking versus control menus

When an expert uses a marking menu the distance moved during the selection of an operation is not important. Only the form of the gesture is important and is analyzed once the user releases the mouse button or stops moving the cursor (so as to see a menu). A control menu is different in that the distance moved by the cursor is important. The position of the cursor is constantly analyzed and as soon as the cursor has been moved the activation distance from where the mouse button was pressed, the operation indicated by the direction of the movement is started.

## 5.6 Control menus with context and history layers

A control menu is used to control the history and context layers presented in sections 2 and 3. The context layer has two parameters: the scale and the relative transparency. The scale is controlled by horizontal movements of the cursor and the transparency by vertical movements. The control of the history layer is similar; the choice of current view is controlled by horizontal movements of the mouse and the transparency by vertical movements. These parameters are not integral [9] (a diagonal movement of the cursor has no simple meaning) and we are currently investigating whether this poses problems for users.

## 6. IMPLEMENTATION

The ideas presented in the previous sections were implemented and tested in a client/server system designed to be used over the Internet (Figure 4). The client is a Java applet that communicates with

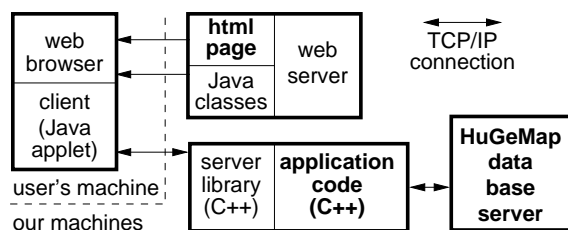


Figure 4: client server implementation

a C++ server over a TCP/IP connection. The client communicates the user's current position to the server and the server responds with all the objects visible at the position in the information space. The client stocks these objects and can thus respond rapidly to user commands; only the arrival and display of new objects is delayed by the latency of the network connection to the server. The server remains close to the data base so that it can rapidly read the large quantities of information required to construct the objects to be sent to the client. The client and the server library that it communicates with are standard and do not need to be changed to provide a system to visualize a completely different information space. Only the database code and the code that calls the server library, shown in bold text in Figure 4, need to be changed.

## 7. CONCLUSION

We have attempted to address one of the major problems of ZUIs: the lack of context and thus the users' difficulties in understanding where they are in the information space and where they can find the information that they are looking for. We have developed new visualization techniques that provide a new way of combining the focus and context in ZUIs. One of our new context aids, hierarchy trees, is always visible and provides a constant reminder of the user's position in the information space. The other context aids, context and history layers, are only visible when the user is lost and requires orientation help. We will continue to compare these quite different aids and try to understand when and how the users find them useful. The interaction complexities that exist in ZUIs and especially in our ZUI with its extra context aids lead us to develop a new type of menu that combines the selection and control of operations. This new menu has been integrated into our ZUI but we believe that it will be useful in other visualization programs.

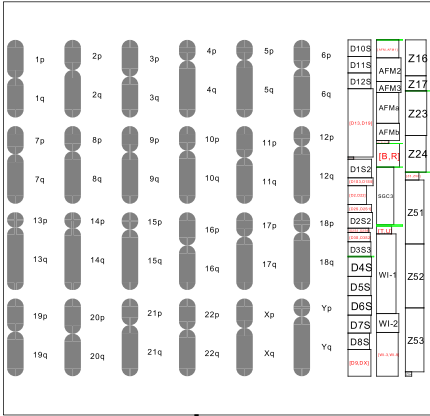
We hope that the combination of our new context aids and our new interaction methods will give users better control over a better interface.

This work was supported by the European Union (contract B104-CT96-0346) and by the CNET (contract 97 754 21).

## 8. REFERENCES

- [1] B. B. Bederson, J. D. Hollan, K. Perlin, J. Meyer, D. Bacon, and G. Furnas. Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. *J. Vis. Lang. Comput.*, 7:3–32, Mar. 1996.
- [2] M. S. T. Carpendale, D. J. Cowperthwaite, and F. D. Fracchia. 3-dimensional pliable surfaces: For the effective presentation of visual information. In *UIST '95*, pages 217–226, Pittsburgh PA, USA, Nov. 1995. ACM Press.
- [3] J. Conklin and M. L. Begeman. gIBIS: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems*, 6(4):303–331, Oct. 1988. Selected Papers from CSCW '88.
- [4] D. A. Cox, J. S. Chugh, C. Gutwin, and S. Greenberg. The usability of transparent overview layers. In *CHI '98 Summary*, pages 301–302, Los Angeles, CA USA, Apr. 1998. ACM Press.
- [5] J.-D. Fekete and C. Plaisant. Excentric labeling: Dynamic neighborhood labeling for data visualization. In *CHI '99*, pages 512–519, Pittsburgh, PA, USA, May 1999. ACM Press.
- [6] G. W. Furnas. Generalized fisheye views. In *CHI '86*, pages 16–23, Boston MA, USA, Apr. 1986. ACM Press.
- [7] G. W. Furnas and B. B. Bederson. Space-scale diagrams: understanding multiscale interfaces. In *CHI '95*, pages 234–241, Denver CO, USA, 1995. ACM Press.
- [8] B. L. Harrison, G. Kurtenbach, and K. J. Vicente. An experimental evaluation of transparent user interface tools and information content. In *UIST '95*, pages 81–90, Pittsburgh PA, USA, Nov. 1995. ACM Press.
- [9] R. J. K. Jacob and L. E. Sibert. The perceptual structure of multidimensional input device selection. In *CHI '92*, pages 211–218, Monterey, CA USA, May 1992. ACM Press.
- [10] S. Jul and G. W. Furnas. Critical zones in desert fog: Aids to multiscale navigation. In *UIST '98*, pages 97–106, San Francisco, CA, USA, Nov. 1998. ACM Press.
- [11] G. Kurtenbach and W. Buxton. User learning and performance with marking menus. In *CHI '94*, pages 258–264, Boston MA, USA, Apr. 1994. ACM Press.
- [12] J. Lamping and R. Rao. The hyperbolic browser: A focus+context technique for visualizing large hierarchies. *J. Vis. Lang. Comput.*, 7(1):33–35, Mar. 1996.
- [13] S. Pook, E. Lecolinet, G. Vaysseix, and E. Barillot. Control menus: execution and control in a single interactor. In *CHI 2000*, The Hague, The Netherlands, Apr. 2000. ACM Press. to appear.
- [14] S. Pook, G. Vaysseix, and E. Barillot. Zomit: biological data visualization and browsing. *Bioinformatics*, 14(9):807–814, Nov. 1998.
- [15] G. G. Robertson and J. D. Mackinlay. The document lens. In *UIST '93*, pages 101–108, Atlanta GA, USA, Nov. 1993. ACM Press.

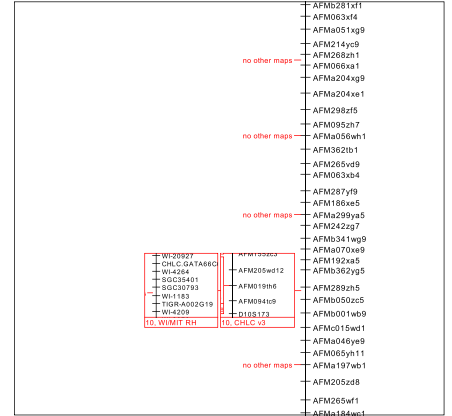
(1) the initial view of the 24 chromosomes



(2) chromosomal bands and map names

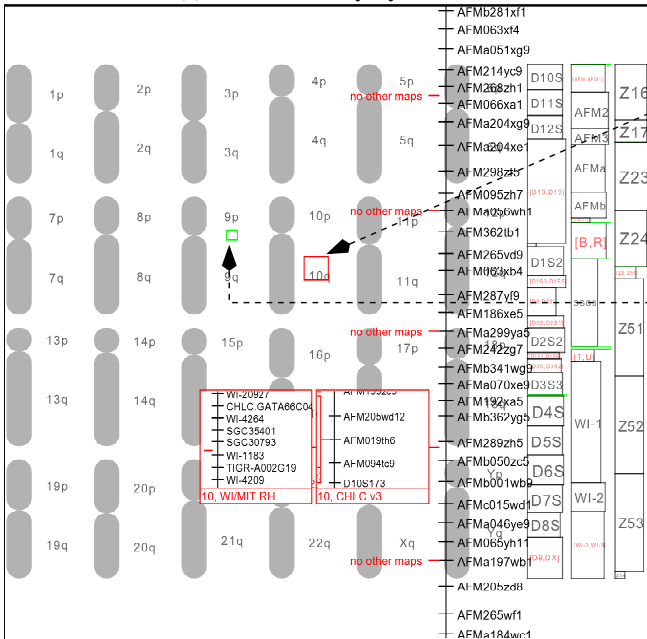


(3) markers on the axis of a map



(initial view + historical view)

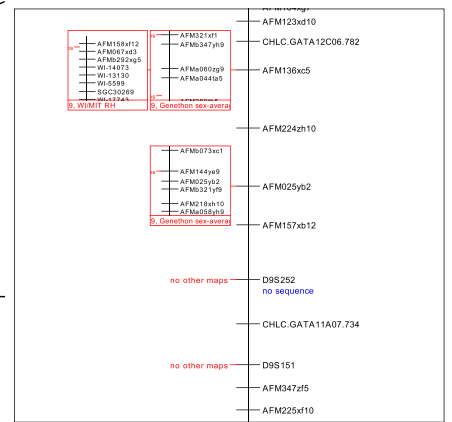
(4) view with history layer



position and size of the historical view

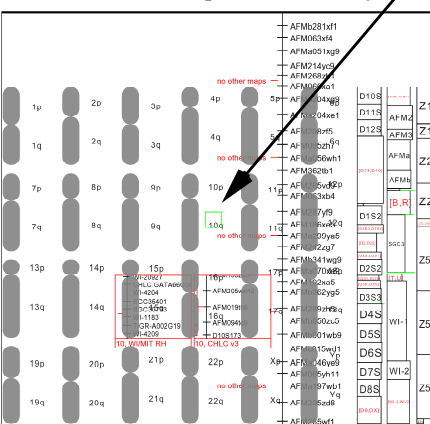
position and size of the last current view (relative to the history layer)

(5) last current view

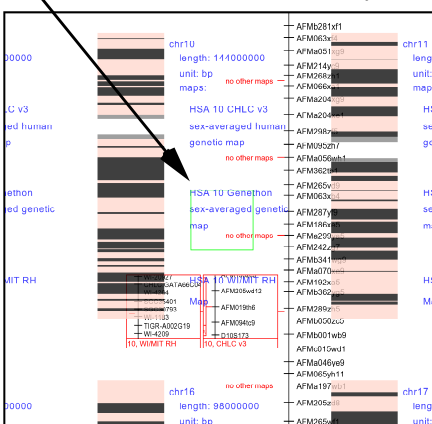


the position and size of the focus (relative to the context layer)

(6) view with (a top level) context layer



(7) view with (a zoomed) context layer



(8) view concentrating on the context layer

