No-click browsing of large hierarchical data

Toshiyuki Masui
Keio University
5322 Endo, Fujisawa, Kanagawa 252-0882, Japan
masui@pitecan.com

***Abstract*—We introduce a novel simple interaction technique for finding and browsing data in a large hierarchical database using only two keys.**

Finding data from a huge hierarchical database takes time because users should traverse the tree using a mouse or a keyboard, and click a "select" button to check the content. If the data was not appropriate, they should go to the next entry and perform the selection task again.

In the past, when only a small number of "channels" were available from broadcast TV stations, users could simply rotate a television dial to select a channel and watch a program. We introduce a simple interaction technique called "*Gear*", with which users can select an entry from a large hierarchical database and browse it instantly, just like we could select a channel and enjoy a program using old television dials.

***Index Terms*—Input device, Information navigation, No-click browsing, Hierarchical data, Gear, Serencast**

I. Introduction

Huge amount of movies, musics, and other programs are available on the web, but finding and viewing a program is not easy because (1) finding a program takes time, and (2) a "confirm" operation (clicking something or typing a key) is required for viewing the content. The confirm operations was not required on old TVs and radios, because we could enjoy TV/radio programs as soon as we rotated a TV channel dial or a tuning dial of the radio. It is always convenient to view the contents of the selected item as soon as we find it. We call such viewing style as "*no-click browsing*".

We can perform no-click browsing on the "column view" mode of Finder.app on MacOS using four keys. In the column view mode, the contents of the file is shown at the right side of the window as soon as we select a file in a folder (Figure 1). When we type an arrow key and select another file, the contents is shown instantly without a confirm operation.

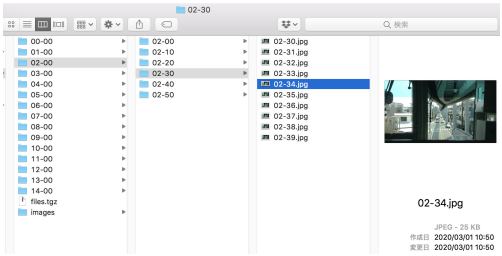

Fig. 1. No-click browsing on the "column view" of Finder.app.

978-1-7281-6901-9/20/$31.00 ©2020 IEEE

We could use no-click browsing on old TVs and radios because the number of broadcast channels were limited. Likewise, no-click browsing on Finder.app is useful when the number of files in a folder is not very large.

No-click browsing is not popular on modern GUI of personal computers. People are used to click-based browsing on web browsers and other information visualization systems. People can navigate through a huge information space by clicking links, zooming and scrolling by dragging, and using other GUI techniques. These techniques are useful for finding an item in a large information space, but many clicks and drag operations are required, and the contents of the items are usually not shown during the search.

It would be better if we could perform the navigation task using devices that have only two keys. We have developed a new navigation technique called "*Gear*" for no-click browsing, where users can explore a large hierarchical database by using only two keys.

II. Navigation method of Gear

Interaction with Gear is based on the following simple principles. Two operations, ▲ (up) and ▼ (down), are used for the navigation.

1) A portion of hierarchical data is shown to the user.
2) One element in the list is selected and highlighted.
3) When an element is selected, its siblings, its ancestors and their siblings are displayed around the selected element.
4) Users can issue ▲ to select the element above the currently selected element, or issue ▼ to select the element below the selected element. Whenever the selection is changed, 3) is performed. If the depth of the newly selected element is different from the currently selected element, siblings of the currently selected element disappear.
5) When a newly selected element has children and the user performs no further action, the first child of the selected element is newly selected, and 3) is performed. As a result, all the siblings of the newly selected element (the first child of the currently selected element) appear in the list.

We show how Gear works, using a hierarchical data structure of a shop list in a shopping mall shown in Figure 2 as example data. A rectangles with a thick border represents a shop category, and other rectangles represent individual shops.

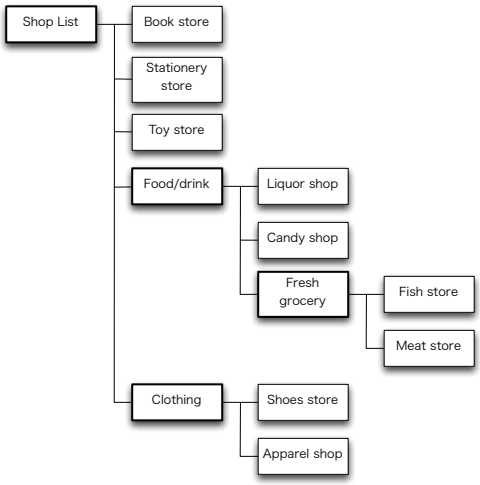

Fig. 2. Sample data: shop list in a shopping mall.

When a user starts the exploration, only the shops and categories at the top level are displayed (Figure 3). When the user issues ▼, the second element (Stationery store) is selected (Figure 4).

Fig. 3. Initial display.

Fig. 4. Typing ▼.

If the user issues ▼ two more times, Food/drink category is selected (Figure 5). If the user stops the operation and waits for a moment, the shops under the Food/drink category are automatically displayed, and the first entry (Liquor shop) is selected (Figure 6).

Fig. 5. Selecting Food/drink.

Fig. 6. Selecting Liquor shop.

When the user issues ▼ twice here, Fresh grocery category is selected (Figure 7). If the user keeps issuing ▼, the list will change to Figure 8, without expanding the children of Fresh grocery.

Fig. 7. Selecting Fresh grocery.

Fig. 8. Selecting Clothing.

If the use stops issuing ▼ at Figure 7, the shops under category Fresh grocery is automatically selected (Figure 9). When the user issues ▲ here, Fresh grocery is selected, and the shops under Fresh grocery disappears, resulting in the same state as Figure 7. If the user issues ▲ two more times, the display changes to the state shown in Figure 6, and one more ▲ will set the system to the state of Figure 5.

If the user issues ▼ in Figure 5, the next visible entry (Clothing) is selected (Figure 8), and the state changes to Figure 10.

Fig. 9. Selecting Fish store.

Fig. 10. Selecting Shoes store.

When the user issues ▼ twice in Figure 9, Clothing is selected, and the category under Food/drink will shrink (Figure 8).

In this way, users can explore the hierarchical structure only by issuing ▲ and ▼ at the right timing.

III. Serencast: A no-click browsing service

To prove that Gear is useful for selecting an entry from a large hierarchical database, we developed the "Serencast" service[1], where users can enjoy movies, musics, and other web services on the browser with the Gear interface without clicking an entry. Serencast is implemented in JavaScript and runs on modern web browsers. Serencast uses the "Scrapbox" wiki service[2] for listing the contents of the hierarchical database. Scrapbox is an interactive wiki system where users can share and edit texts on the web browser, just like the Google Docs system.

When we list the URLs of Web pages of movies and musics on a Scrapbox page, we can use Serencast service for using the Gear interface to select a program from the list on Scrapbox pages and automatically play it on the browser.

Figure 11 shows the movie list defined in a Scrapbox page.

[1]https://Serencast.com/
[2]https://Scrapbox.io/

```
Movies and videos
 [https://www.netflix.com/watch/70105600 Gran Torino]
 [https://www.netflix.com/watch/80047616 My Intern]
 [https://www.netflix.com/watch/330201 Breakfast at Tiffany's]
 [https://www.netflix.com/watch/70080178 Heroes]
 [https://www.netflix.com/watch/70174781 Sherlock]
 [https://www.happyon.jp/watch/100048550 Mad Max]
 [https://www.netflix.com/watch/60010351 Flashdance]
 [https://www.netflix.com/watch/80095365 La La Land]
 [https://www.netflix.com/watch/81191988 Don't stop the camera]
 [https://youtube.com/embed/AtZr9g2vwrI?autoplay=1 Fireplace]
```

Fig. 11. A movie list page on Scrapbox.

When we access this Scrapbox page from Serencast, "Gran Trino" from Netflix automatically starts in the right side of the screen, just like a TV program starts when we rotate the channel dial of an old television. When the user pushes the ▼ key twice, "Breakfast at Tiffany's" is selected and the movie starts automatically at the right side of the screen (Figure 12).

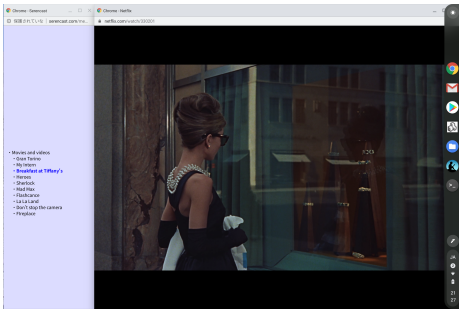

Fig. 12. Watching "Breakfast at Tiffany's" on Serencast.

In the above example, we can select only one of the 10 movies from the movie list. But we can add more data by hierarchically providing contents in Scrapbox pages.

Since anybody can edit Scrapbox pages and construct the hierarchical data, people can enjoy creating their own database by editing their Scrapbox pages. When someone wants to advertise their works, he can create his Scrapbox pages and give them to other people so that they can watch the pages.

IV. Discussion

A. Input devices for Gear

Since only two switches are required in the Gear interface, almost any kind of input devices can be used for Gear. We have tried various input devices and collected experiences.

1) Keys and buttons: The simplest input device for Gear may be keyboards and push buttons. Gear works pretty comfortably with arrow keys on a PC. Various HID devices[3] are available on the market, and all such devices can be used as input devices for Gear.

2) 2-way lever: We have also tried to use a paddle-like device for generating ▲ and ▼ (Figure 13). Users can push the puddle to the right to generate ▲ and left to generate ▼. Pressure sensors are installed on the puddle, and many ▲'s are generated when the user pushes the puddle strongly.

[3]https://en.wikipedia.org/wiki/Human_interface_device

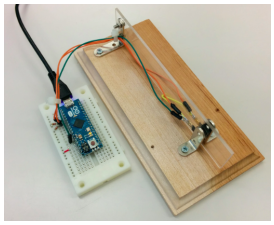

Fig. 13. Paddle device for Gear.

Fig. 14. Using a jog dial.

Controlling the speed of the scrolling of Gear by pressure is convenient, since sometimes users want to jump to an entry far from current position. That behavior is similar to the key-repeat feature of computer keyboards.

3) Rotating devices: We then tried various rotating devices for Gear. We tried mouse wheels first, and found that they are very suitable for Gear, because we are good at controlling our finger with precision.

Many kinds of rotating input devices have been used for controlling TVs and VCRs so far. For example, "jog dials" have been widely used for controlling VCRs, and we tried the device for Gear (Figure 14).

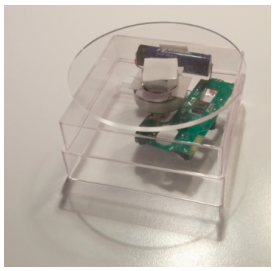

Fig. 15. A disk-based device for Gear.

Fig. 16. Using a roller.

We have also tried using a cylinder for Gear. The rotating cylinder shown in Figure 16 is touching the mouse wheel, and mouse wheel events are generated as the user rotates the cylinder using his arm or hand.

After trying these devices, we found that controlling rotating devices like jog dials and cylinders are not as easy as we expected, because precise control of an arm or a hand is fairly difficult. People are good at controlling fingers, but people are usually not very good at controlling arms and hands precisely.

After trying various devices, it became clear that simple buttons or wheels can be put at anywhere like on the table or under a chair, and we felt that the integration of furniture and input devices will be important in the IoT age [1]. If we can install the devices neatly on furniture and beds, people with disabilities can easily use them.

B. Comparison with existing navigation methods

To compare the speed of Gear with conventional navigation method, we created two applications for checking how fast a user can select an entry from a large hierarchical database. Both applications behaves the same way: a user can set the

date to a specific time to see a photo in a 15-minute monorail trip. Using the first one (App1), a user can set the time using Gear with up/down arrow keys for ▲ and ▼ (Figure 17).

Using the second one, a user can set the time using the <input type="time"> tag of HTML (Figure 18). Users can first set the value of the "minute" value at the top-left using up/down cursor keys, and move to the right with the right-arrow key and set the "second" value with up/down arrow keys.

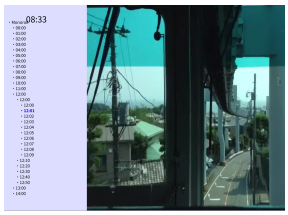

Fig. 17. App1: Gear-based application for finding a monorail picture.

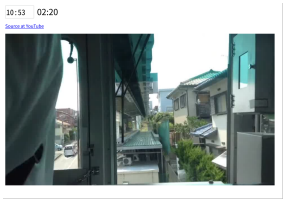

Fig. 18. App2: using input element.

On both applications, a random target time is displayed on the screen and the subjects are asked to set the time to the displayed target time. After repeating the task 5 times, the average time is shown to the user, and the value was reported.

We asked 5 people to try these tasks 10 times, and the result of the experiment is shown in Figure 19.

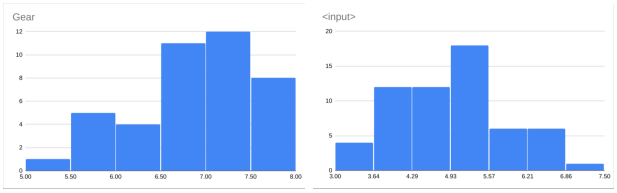

Fig. 19. Average search time of App1 and App2.

The average time for finding an entry using Gear was 6.84s, and the average time using standard <input> was 4.96s. It takes about 40% longer than using conventional method. The standard deviations were 0.71s and 0.93s, respectively. This is a little bit disappointing, but we are happy to see that no-click browsing with Gear is not too slower than conventional methods using four keys. We have also tried using Finder.app for the same data (Figure 1), and got similar results.

Using a conventional hierarchical menu, child elements are displayed automatically when their parent element is selected by a mouse. The behavior is well understood by computer users, and Gear's automatic transition (e.g. from Figure 5 to Figure 6) looks familiar to users.

C. Comparison with InfoVis techniques

Various information visualization techniques like Treemap [2], Hyperbolic Tree [3], and Sunburst [4] have been proposed for visualizing large hierarchical data. Zooming user interface (ZUI) systems like Pad [5] and Pad++ [6] can also be used for handling large hierarchical data laid out in a 2D space. These visualization techniques and descendent technologies have become popular these days and all of these systems are useful for understanding the structure of large hierarchical data. However, users of these systems have to use a pointing device to take full advantage of these methods, and no-click browsing is not supported.

It seems to be a good idea to use Gear on more conventional visualization techniques like TreeView[4], where four keys are used for navigation. We can also use Gear on a 1D zooming system like LensBar [7], where pointing devices are mainly used for navigation.

D. Using Gear in the wild

The author have been using Gear in his living room for several years with a small personal computer connected to a 60-inch TV monitor. The database is managed on Scrapbox, and a mouse wheel of a wireless Bluetooth mouse is used for Gear interface. With Gear, the user can enjoy all available contents on the web with the mouse wheel without selecting the source of the movies, animes, and musics from Amazon, Netflix, etc.

We can enjoy using Gear everywhere using a stick PC and a wireless mouse, just like Amazon FireTV can be used for the same purpose. Using Serencast is more fun for the author, since no-click browsing is more comfortable than selection-based interface on existing systems.

We demonstrated Gear at an exhibition and asked more than 100 people to try it (Figure 20). Since the only thing a user could do with the Gear device was to rotate a wheel, users seemed to be able to understand the behavior of Gear with trials and errors. Although using only one rotating device for navigation was a new experience for the visitors, they seemed to have felt it natural in a short while.

Fig. 20. People at the right using a wheel device at an exhibition.

V. Conclusion

We have developed a new simple interaction method "Gear" for exploring a large hierarchical data structure. A Gear user can easily find an entry in a huge hierarchical database only by using two keys or a rotating device that can be installed at wide ranges of locations where conventional devices do not fit. We are hoping to install various implementations of Gear and try them at various places like kitchens, restrooms, etc.

[4]http://en.wikipedia.org/wiki/Tree_view

REFERENCES

[1] S. Hashimoto and T. Masui, "The furniture of ubiquitous computing," in *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*. New York, NY, USA: Association for Computing Machinery, 2013, p. 845–852. [Online]. Available: https://doi.org/10.1145/2494091.2497326

[2] B. Johnson and B. Shneiderman, "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in *Proceedings of the 2nd Conference on Visualization '91*, ser. VIS '91. IEEE Computer Society Press, 1991, pp. 284–291. [Online]. Available: http://dl.acm.org/citation.cfm?id=949607.949654

[3] J. Lamping, R. Rao, and P. Pirolli, "A focus+context technique based on hyperbolic geometry for visualizing large hierarchies," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '95. ACM Press/Addison-Wesley Publishing Co., 1995, pp. 401–408. [Online]. Available: http://dx.doi.org/10.1145/223904.223956

[4] J. Stasko and E. Zhang, "Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations," in *Proceedings of the IEEE Symposium on Information Vizualization 2000*, ser. INFOVIS '00. IEEE Computer Society, 2000, pp. 57–65. [Online]. Available: http://dl.acm.org/citation.cfm?id=857190.857683

[5] K. Perlin and D. Fox, "Pad: An alternative approach to the computer interface," in *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '93. ACM, 1993, pp. 57–64. [Online]. Available: http://doi.acm.org/10.1145/166117.166125

[6] B. B. Bederson and J. D. Hollan, "Pad++: A zooming graphical interface for exploring alternate interface physics," in *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '94. ACM, 1994, pp. 17–26. [Online]. Available: http://doi.acm.org/10.1145/192426.192435

[7] T. Masui, "LensBar - visualization for browsing and filtering large lists of data," in *Proceedings of the 1998 IEEE Symposium on Information Visualization*, ser. INFOVIS '98. IEEE Computer Society, 1998, pp. 113–120. [Online]. Available: http://dl.acm.org/citation.cfm?id=647341.721215