Icons++: An Interface that Enables Quick File Operations Using Icons

Xiangping Xie

Department of Computer Science
University of Tsukuba
Tsukuba, Ibaraki, Japan
Email: xie@iplab.cs.tsukuba.ac.jp

Jiro Tanaka

Department of Computer Science
University of Tsukuba
Tsukuba, Ibaraki, Japan
Email: jiro@cs.tsukuba.ac.jp

***Abstract*—In graphical user interfaces, when users want to operate on a file, they usually double-click the file icon to launch the associated application and open the file. Several file operations are available in the context menu by right-clicking, such as printing and deleting. However, if the user wants to perform some simple operations, such as copying the file contents, opening the file by an application and then selecting a menu, can be cumbersome. Furthermore, operations in the context menu are limited. Thus, in this paper we present Icons++, a user interface which allows users to perform the file operations they want in a quick way by using icons. Through the use of Icons++, users can take a quick look at the file contents, and at the same time they can perform often-used file operations with only one click, without opening the file by a relevant application. In this paper we present our design of Icons++ and the user studies we performed in order to evaluate it. Studies' results show that using Icons++ is 53% faster than using an application to execute the same task, and our interface is preferred by participants.**

Keywords— icon interface; preview; file operation; file manager.

I. Introduction

Clicking a file icon to launch an application that opens the file is commonly used for accessing files in Graphical User Interfaces (GUIs). File icons allow users to browse files [1], and figure out the file types. Thus, it can be said that icons have an excellent browsability. On the other hand, applications allow users to perform a wide variety of file operations, so users can edit files using those applications. It can be said that applications have an excellent operability.

As can be seen from shortcut keys or mouse gestures, users have a constant need for carrying out often-used operations in an easy manner [2]. As a method of taking a shortcut to operations on menu selection, Appert et al. investigated using stroke gestures as command shortcuts [3]. However, it imposes memorizing gestures' patterns on users. Some researches accentuate items in a menu, e.g., Ephemeral Adaptation [4] and Bubbling Menus [5], though these techniques require tracking many steps on menu selection to reach the desired item. In addition, the following instances show that users also have a need for knowing the file contents without actually opening the file: first, using thumbnails of file contents as file icons is common. For example, a thumbnail for the first page of a PDF file is used as the icon of that PDF file. Also, in the current User Interface (UI) of Gmail, there is a preview function, which shows the contents of the attached files without using an application to open the files.

Therefore, we propose a new user interface which displays the file contents and allows users to execute operations on files, so that they can perform common file operations in a quick way. In this paper, we introduce an interface called Icons++ that has both browsability and operability. Icons++ is located between the icon and the application (Figure 1).

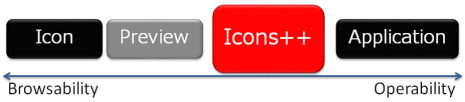

Fig. 1. Icons++ is placed between the icon and the application

As shown in Figure 1, we assign browsability and operability to the horizontal axis. File icons allow users to look at many files at the same time and know the file types, so the use of icons increases the browsability. Preview function shows further file information and allows users to scroll pages, so Preview is next to Icon on our axis. Since applications allow users to change file contents and operate on files significantly, the Application contributes to a high operability. Our proposed interface, Icons++, lies midway between the Icon and the Application.

The following section describes our Icons++ interface and interaction. Section 3 shows a usage scenario to reveal the usefulness of Icons++. In Section 4, we explain how we implemented Icons++. We then present two studies and a questionnaire about Icons++. The studies show that Icons++ is useful on file operations. Finally, we review related work on hovering, using icons for operations, and visual feedback on icons and thumbnails.

II. Interface Design for Icons++

In order to allow users to operate on files in a quick way, we first display the file contents in what we called "Contents View", so that users can confirm whether it is the right file. In Contents View, there are "Operation Icons" which make file operations possible with only one-click. Moreover, in order to find the file contents easier, users can add a mark on thumbnails in Contents View.

A. Contents View by Hovering

Users can activate Icons++'s Contents View by using hovering. Hovering is the action that occurs when the mouse cursor

is placed over the object. Using hover, users do not need to memorize complicated operations like mouse gestures. Moreover, it has the potential to make the user notice Icons++'s functionalities naturally. Hover is uncompetitive with other mouse events, such as a right-click to open a pop-up menu or a double-click to open the file in the application. Thus, it allows users to use existing mouse events as usual.

In Icons++, when users hover over a file icon for about one second, Contents View is automatically displayed near the file icon (Figure 2a). Users can look at each page of the file in Contents View. Contents View disappears when the mouse cursor leaves the file icon. Contents View becomes fixed when users click the file icon. Application icons become available when users move the mouse cursor on top of Contents View (Figure 2b left). Selecting one application icon, the file will be opened by the application (Figure 2b right).

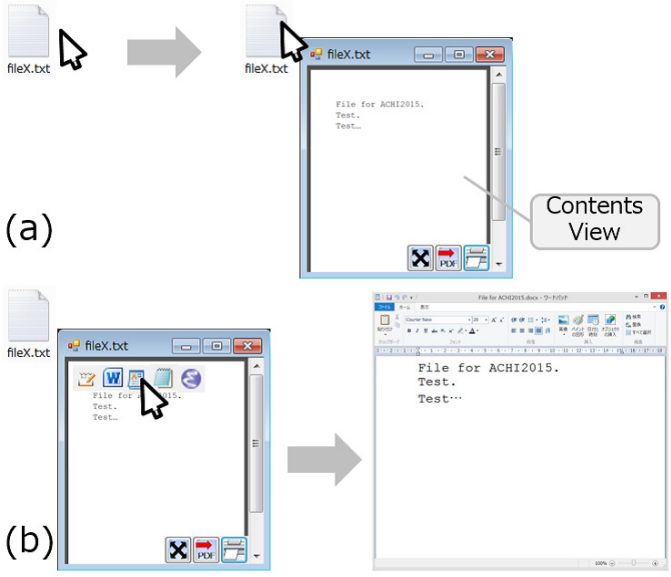

Fig. 2. (a) Hover over the file icon to display Contents View (b) Hover over the top of Contents View to show application icons and open the file by an application

Thus, by using hover, users can access Icons++ from a file icon, and can then access applications from Icons++. Therefore, it can be said that Icons++ is located between file icons and applications, and it links the file icons with applications. Unlike the traditional way, one file type can be linked with multiple applications.

B. File Operations by Using Operation Icons

Using Operation Icons in Contents View of Icons++, users can perform some often-used file operations with only one-click, as opposed to using many steps when using an application. As shown in Figure 3, there are three Operation Icons at the bottom of Contents View. Users click an operation icon, then the file operation is executed (Figure 3).

File operations include making a different type of file (e.g., PDF or plain text) based on the original file, showing in full-screen, displaying as slide show, printing, etc. Figure 4 shows some examples for Operation Icons. Operation Icons

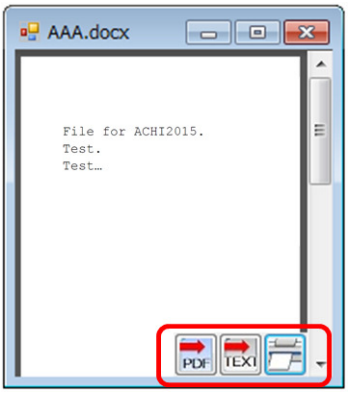

Fig. 3. Operation Icons on the bottom of Contents View

Fig. 4. (a)Full-screen (b)Slide show (c)Make PDF (d)Make TEXT (e)Print

in Contents View are dynamically changed according to the file type. With regard to visibility for the file contents, the maximum number of Operation Icons in Contents View is 6, which is the number of icons that can be shown in one line.

Besides the default Operation Icons in Icons++, users can make icons for custom operations. Custom Operation Icons are made when users record the operations they want, then press the custom operation icon to repeat the recorded operations.

Thus, Icons++ has an improved operability, because, with only one-click, Operation Icons enable users to perform file operations, which they would traditionally perform in multiple steps.

C. Mark File Pages by Dog-ear and Folded Page

In Contents View of Icons++, we make it possible for users to add a mark on the thumbnail of the page, so that they can easily find the file or page they want. As shown in Figure 5, Dog-earing is used for the thumbnails of important pages (first page in Contents View in Figure 5). Folded page is used for the thumbnails of insignificant pages (second page in Contents View in Figure 5). For example, if a report file has a part that the user wants to review, the user can add a dog-ear on the pages that have that part. On the other hand, if the title page of the report has little information, the user can fold the thumbnail of the title page to make it less noticeable.

We explain the creation of dog-ears and folded pages below.

1) Dog-eared Page: Dog-earing is an action that folds a corner of a page to create a triangle shape. In Icons++, clicking the upper right corner of the thumbnail adds a dog-ear. Single-clicking adds a small dog-ear, while double-clicking creates a big dog-ear. Clicking the area which already has a dog-ear can remove this dog-ear by flattening the dog-eared area. The page that has a dog-ear will be the page that users see first in Contents View, thus finding the important page becomes easier.

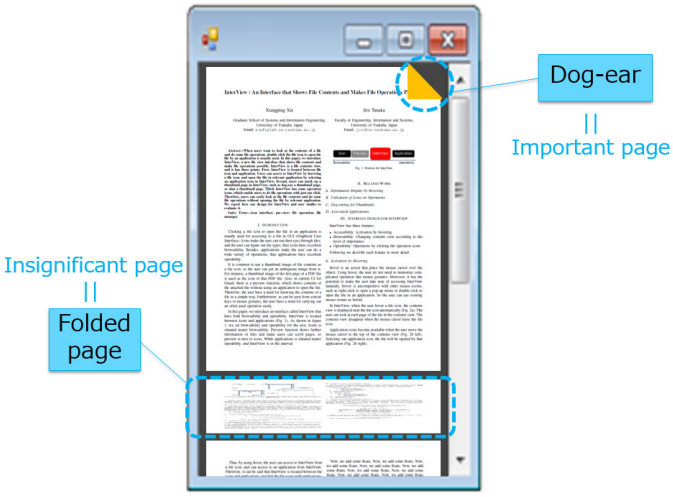

Fig. 5. Dog-ear and folded page for thumbnails in Contents View

2) Folded Page: Folded page is a thumbnail which is a quarter in size of the original size. Double-clicking the thumbnail of the page transforms it into a folded page. Similarly, double-clicking a folded page takes the thumbnail back to the original size.

We can thus state that Icons++ has an improved browsability because users can check the file contents at a glance in Contents View, and they can set the importance for any page using dog-ears and folded pages.

III. Usage Scenario

In this section we will introduce a scenario for using our interface. Let us assume that, after a meeting, user A wants to send the minutes of the meeting, recorded as plain text, via e-mail. Taking into account possible character corruption, user A is going to change the minutes written in plain text into a PDF file and attach it to the e-mail.

In the traditional way, user A should perform the following procedures: (1) use the context menu by right-clicking the minutes file icon in the file manager, (2) open the file in WORD, (3) click menu in WORD, (4) select "Save As", (5) change file type to PDF, (6) make a PDF file, (7) attach the PDF file to the e-mail.

When using our Icons++, the procedure becomes as follows: (1) hover over the minutes file icon in the file manager to show Contents View, (2) click the file icon to make Contents View fixed, (3) select "make PDF" Operation Icon, (4) make a PDF file, (5) attach the file to the e-mail. Thus, user A can perform the file operation in a shorter way without going through an application.

IV. Software Implementation

We created a prototype as a file manager for Icons++ in C# under Windows 7. We made it based on the close image of the file manager in Windows, hence it is expected to be easy for the user to adapt to Icons++.

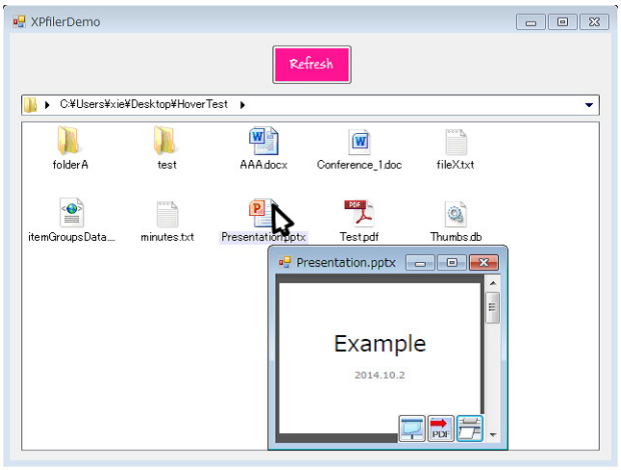

Fig. 6. UI for the prototype of Icons++

A. UI for the Prototype

The user interface for the prototype of Icons++ is shown in Figure 6.

Figure 6 shows the user hovering over a PowerPoint file icon with the mouse cursor; the Contents View for this file becomes visible near the file icon.

B. Contents View

Icons++ keeps JPEG files corresponding to each page of the user's files. Therefore, when the user hovers over a file icon, the corresponding JPEG files are immediately displayed, so that the user can instantly check the contents of that file.

When there is a newly-created file, and the user hovers over that file icon in Icons++, our system will generate JPEG files corresponding to that file before Contents View is displayed. If the file is large, e.g., a PowerPoint file that has 20 slides, then it takes a little while to generate JPEG files corresponding to that file.

In Icons++, we also have a folder which stores dog-eared JPEG files and folded JPEG files. If a file is hovered over in Icons++, the system first checks the contents of this folder. If there is a dog-eared JPEG file for the hovered file, Icons++ displays the dog-eared page as the starting page in Contents View.

Contents View has four layers, as shown in Figure 7.

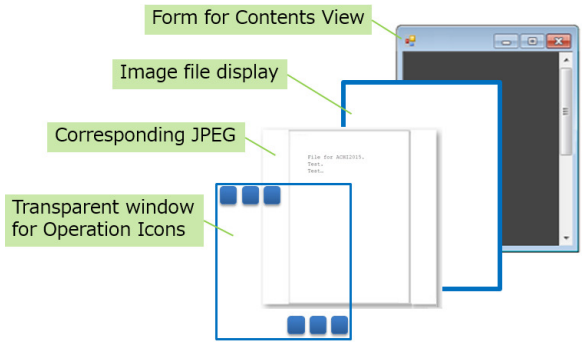

Fig. 7. Four layers for Contents View

When Icons++ shows a JPEG file in Contents View, the system first shows a form window. Then it adds a layer of image file display. It then adds a JPEG file corresponding to the hovered file on the image file display. Finally, the system adds a transparent window for Operation Icons on the top in a way that allows the user to perform file operations.

Because of the use of JPEG for showing the contents of the file, Contents View of Icons++ can prevent character corruption, regardless of the format of the original file.

C. File Operation

We conducted a user survey on file usage before implementing the prototype. We asked 9 students (1 female, 8 males, ranging from 20 to 24 years old) about the functions in file operations that they often use and that they would like to use easily. Based on their responses, we implemented 6 operations that had the highest number of responses (some are shown in Figure 4).

We preliminarily registered commands which correspond to Operation Icons in Icons++. When users click an Operation Icon in Contents View, the system activates the pre-registered command which corresponds to that Operation Icon, and then executes the file operation.

In the prototype of Icons++, the targeted file type is a document file, such as plain text, source code (e.g., program file, LaTex file), Microsoft office file, PDF, etc.

To enable the customization of file operations, Icons++ allows users to record operations they want to easily use. A sequence of shortcut keys and menu selections in applications will be recorded in the system. When users select a custom operation icon in Contents View, recorded operations will be repeated automatically.

V. Evaluation

To better understand the benefits of Icons++ and compare Icons++ with the UI of an existing file manager, we conducted two user studies and a questionnaire survey.

Seven computer users (2 females, 5 males), ranging in age from 22 to 27 (mean 24), participated in the studies.

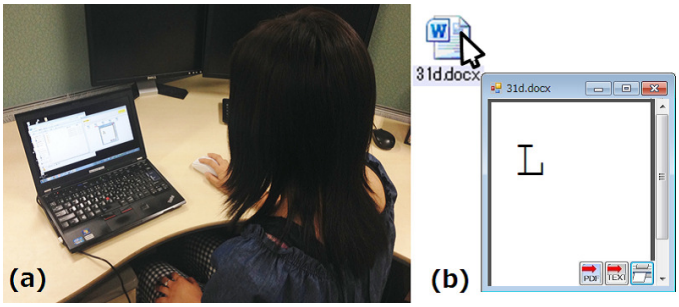

Fig. 8. (a)A participant taking part in the study (b)File with just one alphabet letter

A. Procedure

We first asked the participants to use Icons++ for 3 minutes without any explanation. Then we explained the use of Icons++ and all of its functionalities. After that, we let participants use Icons++ for about 3 more minutes so that they get familiarized with its usage. After the participants got used to Icons++, we asked them to perform two user studies (Figure 8a). Finally, we asked them to fill out a questionnaire survey. We will describe the two user studies and the questionnaire survey in detail in the following subsections.

B. User Study 1: Finding a File Containing a Designated Alphabet Letter

The goal of this study is to compare Icons++ with the file manager of Windows 7. The task of this study is to find a file containing an alphabet letter designated by the system.

We prepared files containing just one alphabet letter (Figure 8b). In total, we had 3 approaches to find the file: (1) using double-click to open the file in the application, (2) using the preview function in the file manager of Windows 7, and (3) hovering over the file icon in Icons++. Participants had to perform 5 trials for each approach. In each trial, participants had to find one file with an alphabet letter designated by the system amongst 5 files. Thus, participants performed the file finding task in 5 files $\times$ 5 trials $\times$ 3 approaches (in total, 75 trials per participant). The system calculated the time for each finding task.

C. User Study 2: Look at a PowerPoint File in Slide Show

The goal of this study is to compare Icons++ against the traditional model for file operation. In this study, participants performed the same operation using two approaches: (1) using the method they usually use, (2) using Operation Icons in Icons++. The task for this study is to look at a particular PowerPoint file in slide show. There was no limitation on the methods they usually use. These methods could be using the menu or the icon in PowerPoint, using a shortcut key, using the context menu by right-clicking on the file icon, etc. Each participant performed this task once for each approach. The system calculated the time for each approach.

D. Questionnaire Survey

After all user studies were completed, we asked participants to answer a questionnaire about the usability of Icons++. We used a five-point Likert scale, in which 1 corresponds to "strongly disagree" and 5 corresponds to "strongly agree".

First, we asked the participants about accessibility in each approach of User Study 1 (i.e., "whether they know how to use the interface without explanation"). Next, we asked the participants about the usability of Icons++ and whether they want to continue to use Icons++: (1) Is Contents View by hovering easy to use? (2) Are file operations by Operation Icons easy to use? (3) Is the overall UI for Icons++ easy to use? (4) Do you want to continue to use Contents View by hovering? (5) Do you want to continue to use file operations by Operation Icons? (6) Do you want to continue to use the overall UI for Icons++? In addition, we offered the participants the opportunity to write their comments freely.

VI. Results

Figure 9 shows the average completion time for 7 participants with each approach in User Study 1. The left side in Figure 9 shows the mean of total time for 5 trials by 7 participants for each approach, and the right side in Figure 9 shows the mean time of each trial by 7 participants for each approach.

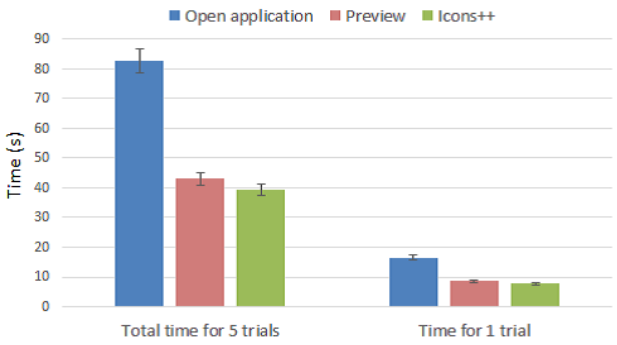

Fig. 9. Average completion time by 7 participants for trials with opening file by an application, preview, and Icons++

The mean time to perform 5 trials with hovering over the file icon in Icons++ is 39.2 seconds (s.d. 5.7), 53% faster than the time opening the file by an application (mean 82.7, s.d. 18.1): $T(12) = 5.6$, $p < .001$. Thus, based on the results, the time for trials with Icons++ shows a significant difference compared with the method of using double-click to open in an application.

On the other hand, the mean time to perform 5 trials with hovering over the file icon in Icons++ is 9% faster than the time using preview in the file manager of Windows (mean 43.1, s.d. 8.6): $T(12) = 0.7$, $p = 0.2$. Hence, there is no significant difference in time with Icons++ compared with the method of using preview function in the file manager.

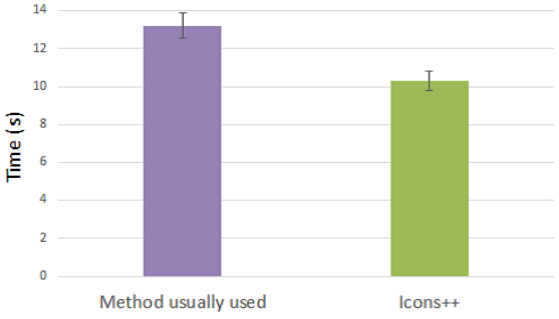

Fig. 10. Average completion time by 7 participants to open a PowerPoint file in slide show with the method they usually used and with Icons++

Figure 10 shows the average completion time for 7 participants for User Study 2. The time to open a PowerPoint file in slide show with Icons++ is 10.3 seconds, 22% faster than the time for using the method they usually used, which is 13.2 seconds: $T(12) = 0.9$, $p = 0.19$. In terms of methods the participants used, some of them opted for menu selection on PowerPoint, some used the icon at the bottom of the window of PowerPoint, and some used a shortcut key (i.e., F5). No one used the context menu by right-clicking on the file manager.

Based on the results for the questionnaire survey, participants responded more favorably to the question "Hovering over the file icon in Icons++ is a more intuitive manipulation" (7 above neutral, none below) than for the equivalent question with double-click to open in an application (6 above neutral, 1 below) and with preview function (4 above neutral, 3 below). Thus, although the time for the same task shows little difference in using the preview function and using hover over in Icons++ (as shown in the result of User Study 1), participants preferred Icons++ to the preview function in terms of operability.

Regarding the UI of Icons++, participants also felt it was easy to use Contents View by hovering over (6 above neutral, 1 below), file operation by Operation Icons (7 above neutral, none below), and the overall UI for Icons++ (6 above neutral, 1 below). When asked whether they want to continue the use, almost all participants agreed to Contents View by hovering over (6 above neutral, 1 below), file operation by Operation Icons (7 above neutral, none below), and the overall UI for Icons++ (7 above neutral, none below).

According to the participants' comments, we found that they felt positive about Icons++. Some participants said that Icons++ is access-friendly because they do not need to learn how to use it. Some participants also pointed out that they found Operation Icons useful because they could perform operations easily. However, there were a few comments about the design improvement of Icons++. Two participants indicated that Icons++ had a drawback: Contents View covered up the lower file icons when it appeared. One participant mentioned that if he really wanted to look at the file contents, Contents View of Icons++ is too small to be legible. Another participant expressed that although Icons++ is useful, the automatic appearance of Contents View is a nuisance if he does not have the intention of knowing the file contents.

VII. Discussion

Compared with existing approaches, such as launching applications, or using the preview function, Icons++ is more intuitive. Participants were able to find out how to use Icons++ without any explanation. One of the main reasons why Icons++ has a better usability is the quick display of the file contents by hovering, which is faster than opening files by an application.

Since hovering is an action that naturally precedes a click (to select) or a double-click (to open the file), Icons++ has an advantage: participants can naturally discover the display of the file contents when hovering over an icon. Results for two user studies and a questionnaire indicated that users preferred simple ways of use, such as hovering, or icons with only one click.

However, the fast display of Contents View is likely to be impeditive when the user has no intention of knowing the file contents. Therefore, there is still some room for improving the design of Icons++ to make it more practical. Some possible solutions could be setting a time limit on the appearance of

Contents View, or making Contents View translucent. In our approach, we appended a slider to Icons++ window, which can turn On or Off the display of Contents View, and which can allow setting its on-screen time.

VIII. Related Work

Many researches use hovering for displaying information near the hovered object. For example, Fitchett et al. proposed a UI named Hover Menu [6] in the file manager, which shows the list of commonly accessed items inside hovered folders. Terry et al. presented Side Views [7], which provided a dynamic, on-demand preview of a command applied to a copy of the data when a menu was hovered over. CommunityCommands [8] by Matejka et al. and Share and Share Alike [9] by Voida et al. display information about the hovered file, such as notes written by co-workers or members who share the file. These researches are related with Icons++ in terms of displaying information by hovering. However, Icons++ is different in that Icons++ allows users to perform file operations and to access an application in Contents View.

Several researchers utilize icons on operations. Touch-Display Keyboards [10] by Block et al. shows an environment using icons for file operations by projection on the keyboard of the computer. Users can select operations by pressing a key. Sikuli [11] by Yeh et al. and the research by Chang [12] used graphical icons in screenshot-based scripts. Users can use screenshot patterns, shown as icons in the scripts, to perform mouse and keyboard events. Unlike these, Icons++ uses the mouse action, which is a conventional method. In addition, Icons++ does not only perform actions on one file, but also allows the users to perform file operations on two or more files in the same file manager's window.

In order to allow users to quickly find an item or a file they want, there are some techniques proposed in existing research. One of these techniques is to add some visual feedbacks on the file icon or file thumbnails for accentuating the target. For example, Fitchett et al. showed Finder Highlights [13], a file browser which has a function of highlighting file icons. There are proposals that use dog-ears to emphasize the target, such as NiCEBook [14], WebView [17], research by Kaasten et al. [15], and research by Hoeben et al. [16]. Unlike these works, we not only add a visual feedback (i.e., dog-ear) for accentuating the object, but we also propose a method to make information less noticeable (i.e., folded page).

Some applications and functions are related to our approach in Icons++. For example, the Quick Look function on Mac OS is similar to Icons++ in that it shows file contents. However, there is no Operation Icon in Quick Look function; it exists for browsing only. Furthermore, the way to access Quick Look is to press the space bar on the keyboard. We cannot state that pressing the space bar is an intuitive manipulation, because, if unknown beforehand, it is difficult to find it naturally. Another example is the preview function in Gmail. The difference between Gmail preview and Icons++ is the tackling of character corruption. In Gmail preview, sometimes characters are not displayed correctly, while this does not happen in Icons++, because Icons++ uses JPEG files.

IX. Conclusions and Future Work

We presented Icons++, an icon based user interface for viewing file contents and manipulating files. Icons++ is able to show file contents quickly when users hover over a file icon. Moreover, it allows users to perform file operations by Operation Icons with only one click. We conducted two user studies and a questionnaire survey for comparing Icons++ against existing approaches. Based on these results, it was found that, first, using Icons++ for checking the file contents was 53% faster than using an application to open the file. Second, using Operation Icons to perform the same file operation was 22% faster than the common methods employed by users. Furthermore, the participants' comments pointed out several issues needing improvement.

In our future work, we expect to address the issue of the file contents not being clearly viewed because of the relatively small scale. Also, we will assess some solutions for the problem of the lower file icons being covered up by Contents View.

References

[1] D. Barreau and B. A. Nardi, Finding and Reminding: File Orgamization from the Desktop. ACM SIGCHI Bulletin 27.3, 1995, pp. 39-43.

[2] J. Tidwell, Designing interfaces. O'Reilly Media, Inc., 2010.

[3] C. Appert and S. Zhai, Using Strokes as Command Shortcuts: Cognitive Benefits and Toolkit Support. In Proc. CHI '09, 2009, pp. 2289-2298.

[4] L. Findlater, K. Moffatt, J. McGrenere, and J. Dawson, Ephemeral Adaptation: The Use of Gradual Onset to Improve Menu Selection Performance. In Proc. CHI '09, 2009, pp. 1655-1664.

[5] T. Tsandilas and M. C. Schraefel, Bubbling Menus: A Selective Mechanism for Accessing Hierarchical Drop-Down Menus. In Proc. CHI '07, 2007, pp. 1195-1204.

[6] S. Fitchett, A. Cockburn, and C. Gutwin, Improving Navigation-Based File Retrieval. In Proc. CHI '13, 2013, pp. 2329-2338.

[7] M. Terry and E. D. Mynatt, Side Views: Persistent, On-Demand Previews for Open-Ended Tasks. In Proc. UIST '02, 2002, pp. 71-80.

[8] J. Matejka, W. Li, T. Grossman, and G. Fitzmaurice, CommunityCommands: Command Recommendations for Software Applications. In Proc. UIST '09, 2009, pp. 193-202.

[9] S. Voida, W. K. Edwards, M. W. Newman, R. E. Grinter, and N. Ducheneaut, Share and Share Alike: Exploring the User Interface Affordance of File Sharing. In Proc. CHI '06, 2006, pp. 221-230.

[10] F. Block, H. Gellersen, and N. Villar, Touch-Display Keyboards: Transforming Keyboards into Interactive Surfaces. In Proc. CHI '10, 2010, pp. 1145-1154.

[11] T. Yeh, T. Chang, and R. C. Miller, Sikuli: Using GUI Screenshots for Search and Automation. In Proc. UIST '09, 2009, pp. 183-192.

[12] T. Chang, Using Graphical Representation of User Interfaces ad Visual References. In Proc. UIST '11, 2011, pp. 27-30.

[13] S. Fitchett, A. Cockburn, and C, Gutwin, Finder Highlights: Field Evaluation and Design of an Augmented File Browser. In Proc. CHI '14, 2014, pp. 3685-3694.

[14] P. Brandl, C. Richter, and M. Haller, NiCEBook-Supporting Natural Note Taking. In Proc. CHI '10, 2010, pp. 599-608.

[15] S. Kaasten and S. Greenberg, Integrating Back, History and Bookmarks in Web Browsers. In Proc. CHI '01, 2001, pp. 379-380.

[16] A. Hoeben and P. J. Stappers, Flicking through Page-based Documents with Thumbnail Sliders and Electronic Dog-ears. In Proc. CHI '00, 2000, pp. 191-192.

[17] A. Cockburn, S. Greenberg, B. McKenzie, M. Jasonsmith, and S. Kaasten, WebView: A Graphical Aid for Revisiting Web Pages. In Proc. OzCHI '99, 1999, pp. 15-22.