UNI
FR

UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

Tobii:
Managing multiple Windows

Semester Project
Future User Interfaces (FUI): multimodal, plastic and natural
30.05.2018
University of Fribourg

Students:
Kevin Meister, kevin.meister@students.unibe.ch
Flurin Truebner, flurin.turebner@students.unibe.ch

Teacher
Prof. Dr. Denis Lalanne

Contents

1 Introduction

1.1 Motivation

The increase in resolution and screen size of monitors supports the use of multiple windows, both on mobile devices and desktops. Working with multiple windows is routine for many people. But the managing of multiple windows isn't very fast or intuitive, especially on mobile devices. The goal of this semester project is to simplify the work with multiple windows by using the Tobii Eye Tracker. We therefore introduce an additional modality to the mouse and keyboard by using the eyes to manage windows. This should lead to a faster and easier execution of these tasks to justify its use. Several use cases will be discussed and the benefit in terms of time or simplicity by using the Eye Tracker will be determined in a user evaluation.
Our goal with this modality was to offer another possibility to facilitate the daily work with several windows. Because we ourselves often work with several windows, we thought about whether we would really use this modality for the tasks. Therefore the user evaluation was important to us because we wanted to see if we could do the tasks faster or easier and would have an advantage over the mouse.

1.2 Technologies

For the development of the application the following technologies were used:

- Tobii Core SDK to get the screen coordinates of the gaze.
- WinAPI for handling the windows
- C# to use and connect both API's
- Visual Studio as the main IDE for programming
- The library MouseKeyHook to detect keyboard inputs
- Git for version management

2 Use-Cases

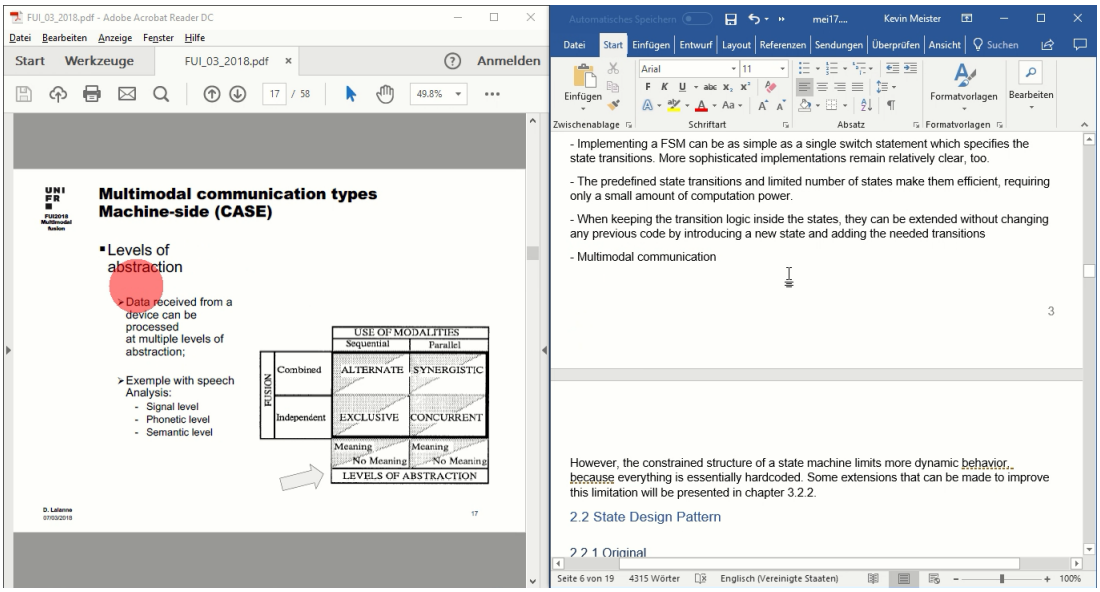

Figure 1: Windows splitscreen view with gaze trace

This is the starting point for all our use cases. Two or more windows are open on one screen. The keyboard and mouse are used to work with the windows. We will discuss several use cases where eyetracking facilitates the work with the windows.

2.1 Visual Feedback

The first goal is to give a visual feedback to the user whenever he wants to make use of the Eye Tracker. A simple way to provide feedback is to simply highlight the position that's currently gazed at. First, this was implemented in all of the subsequent use-cases. Whenever a keyboard shortcut for the Eye Tracker is used, the currently gazed at position gets highlighted so the user knows what to expect from that command. Even though this was planned to greatly help in the user evaluation, certain user were distracted by it. Therefore it was changed to an optional feature, so users can decide by themselves if they want to use this feature or not. After the first few tasks almost all test users disabled it.

2.2 Finding the mouse cursor

Whenever the mouse isn't used for a longer period of time, it can be difficult to locate the position of the mouse cursor. This is especially true for monitors with a higher resolution. Without the Eye Tracker, the only solution to retrieve the position of the cursor is to move the mouse

around until you find its position. With the Eye Tracker, a short keyboard command can be used instead, which instantly moves the cursor to the currently gazed at location and highlights it briefly. Using the same approach, we implemented a similar keyboard command that also makes a mouseclick on the gazed at position, for example to directly clicking a (larger) button or setting the cursor into a specific textfield like in a Skype window.

2.3 Switching to another window on the screen

The current way to switch the active window is to either click on it with the mouse, or press the keyboard shortcut Alt+Tab as many times as needed until you get to it. When working with a mouse, switching to another window that's already in the foreground doesn't take a lot of time. It can take additional time however when the user is currently writing something because one hand needs to be taken off the keyboard or when the cursor isn't already near the window. The alternative, to use Alt+Tab solves this problem, but introduces a new one: If the desired window wasn't the last one active, Alt+Tab needs to be pressed repeatedly until the correct window is found.
The Eye Tracker allows for an even better solution, by just switching to the window that's currently gazed at with a dedicated keyboard shortcut.

2.4 Switching window in the multi-window view

The current way to switch to an active window that's in the background is either to click on it in the windows taskbar with the mouse, or press the keyboard shortcut Alt+Tab as many times as needed until you get to it. The first way works fairly well for Desktop setups (with the same drawbacks as the previous use-case), but for mobile devices with a track pad it's usually faster to use Alt+Tab. When too many applications are open at the same time, Alt+Tab needs to be pressed many times until the desired window is in focus.
With the Eye Tracker, there is an easier way to solve this task. After pressing Alt+Tab, Alt can simply be held down while the desired window is being found, and by confirming with another hotkey the gazed at window is automatically selected and set as active.

2.5 Making inputs to a non-active window in the split-screen view

When working with only one monitor, but more than one window needs to be open and looked at simultaneously, the Windows splitscreen view can be used. For example, on the left half of the screen a pdf window with the course slides can be open, while on the right side we have a word document used for writing a summary of said lecture. When working on a laptop, it can be cumbersome to skip to the next slide without interrupting the current work on the summary. Either the mouse can be used which isn't the optimal solution since the hands leave the keyboard, or Alt+Tab can be used. But this takes an extra step since it needs to be done twice to go back to the original window. A better solution for this is to have a dedicated keyboard command which sends all keyboard shortcuts to the window that's currently gazed at and switches back to the previous window without any extra steps.

3 Multimodality

3.1 Multimodal systems Usability properties - CARE

3.1.1 Complementarity

One of our use-cases, making inputs to a non-active window in the split-screen view, is complementary. It isn't possible with just a keyboard and mouse to make an input to a window that's not currently active. With the Tobii Eye Tracker we reach a special state, since we're able to do that without the user having to switch the active window.

3.1.2 Redundancy

In all of our other use-cases we have redundancy. It's possible to solve the same tasks with either the keyboard or the mouse, even though it's faster and more efficient with the Tobii Eye Ttracker.

3.2 Multimodal communication types Machine-side - CASE

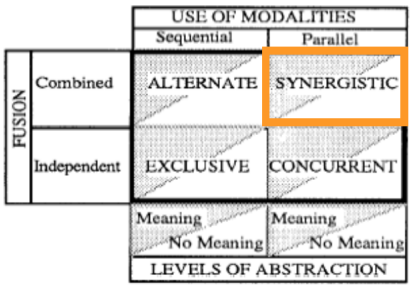

Figure 2: Communication type synergistic [1]

All of our applications use a synergistic approach of multimodal communication. The reason for this is to let the user be in control at all times, so he can always chose to not use the Eye Tracker for a certain task. The most intuitive way we found was the use of keyboard shortcuts whenever the user wants to use the Eye Tracker.

4 User Evaluation

4.1 Description

For the user evaluation we measured the time needed to fulfil the tasks described in the use cases. Therefore we did the same procedure with eight probands. The probands are using Notebooks or Desktop computers mostly daily. First every proband gets to know the Eye Tracker by calibrating it. Then the proband is told the needed shortcut to solve the task and has to solve the task eight times. At the beginning of every try the proband starts with both hands placed on the keyboard. The first five tries are trials to get used to the shortcut and the Eye Tracker, only for the last three the time is measured. This is done twice. One time with the Eye Tracker and one time without it. To compare the modalities we compare the average time needed to complete the tasks among all the probands using the 3 time measurements for each task.
Errors in the execution of the tasks were not evaluated and filtered out by asking the user for an additional try. Most of the mistakes were pressing the wrong keys on the keyboard. Only very rarely was the gazed at position next to the desired point. Besides that, the only problem with the mouse was that the click did not press anything interactive.

4.2 Switching Active Window

In this task we prepared multiple windows on the same desktop for the user to switch between them. For each try, we told the user to which window he has to switch to.
With the Eye Tracker, the task was to look at the desired window and press the dedicated shortcut. We didn't give any further instructions on how to solve the task without the Eye Tracker, since both the use of the touchpad and Alt+Tab would've been possible and we wanted our testusers to use the method they would naturally use. All of our testusers solved it with the touchpad and none used the keyboard.

Person	Average w/o Eye Tracker [s]	Average w/ Eye Tracker [s]
# 1	2.26	2.39
# 2	3.24	2.51
# 3	3.75	1.85
# 4	4.03	2.41
# 5	2.41	1.65
# 6	2.28	1.52
# 7	3.29	1.76
# 8	2.55	2.12
Mean	**2.98**	**2.03**
SD	0.69	0.38
SEM	0.25	0.14

The two-tailed P value equals 0.0056
By conventional criteria, this difference is considered to be very statistically significant.

Confidence interval:
The mean of Group One minus Group Two equals 0.9500
95% confidence interval of this difference: From 0.3808 to 1.5192

Intermediate values used in calculations:
t = 3.9464
df = 7
standard error of difference = 0.241

4.3 Input to inactive Window

For this task two windows in the Windows split-screen view were side by side. The left one was a PDF document of multiple slides, the right one was a word document used for writing which was the active window in the beginning.
With the Eye Tracker, the users were asked to press the dedicated shortcut to change to the next slide in the PDF window and write any word on the word document (switching back wasn't necessary with the Eye Tracker).
Without the Eye Tracker, the users had to switch the active window, change to next slide, switch back to the word document and again write any word.

Person	Average w/o Eye Tracker [s]	Average w/ Eye Tracker [s]
# 1	4.33	1.61
# 2	5.06	1.43
# 3	5.92	1.71
# 4	2.96	1.69
# 5	3.59	2.82
# 6	3.18	1.95
# 7	4.01	1.99
# 8	6.54	2.11
Mean	**4.45**	**1.91**
SD	1.29	0.43
SEM	0.46	0.15

The two-tailed P value equals 0.0015
By conventional criteria, this difference is considered to be very statistically significant.

Confidence interval:
The mean of Group One minus Group Two equals 2.535
95% confidence interval of this difference: From 1.3395 to 3.7305

Intermediate values used in calculations:
t = 5.0142
df = 7
standard error of difference = 0.506

4.4 Switching Window by Alt-Tab

To test this use case, we prepared 8 opened windows all in full screen. The task for our users was to switch to the window we told them to. We had a predefined list for both the Eye Tracker and the keyboard/touchpad setup which defined the windows they needed to switch to. We used different windows for both tests to mitigate the learning effect, but in both cases the windows were the same distance apart.
When using the Eye Tracker, they needed to press Alt+Tab, then while holding the Alt key look at the respective window and confirm with another dedicated key.
Without the Eye Tracker, they needed to press Alt+Tab as many times as needed to get to the desired window, or while holding Alt to click on it (again, we wanted to let them solve the task as they naturally would).

Person	Average w/o Eye Tracker [s]	Average w/ Eye Tracker [s]
# 1	4.52	2.70
# 2	3.27	1.53
# 3	4.35	2.78
# 4	2.41	1.18
# 5	2.80	2.36
# 6	2.90	3.30
# 7	3.46	1.92
# 8	3.01	2.01
Mean	**3.34**	**2.22**
SD	0.75	0.70
SEM	0.26	0.25

The two-tailed P value equals 0.0043
By conventional criteria, this difference is considered to be very statistically significant.

Confidence interval:
The mean of Group One minus Group Two equals 1.12
95% confidence interval of this difference: From 0.4814 to 1.75

Intermediate values used in calculations:
t = 4.15
df = 7
standard error of difference = 0.269

4.5 Move Cursor to Gaze Position

For our final use case the task was to simply move the mouse cursor to a predefined location. We hid the mouse cursor in one of the corners or on the screen border, so the users didn't know where it was for each test run.
The Eye Tracker mitigated the unknown position of the mouse cursor, since it directly jumps to the gazed at position with the press of a keyboard shortcut.
Using the touchpad, the users first had to locate the cursor to move it to the desired position afterwards.

Person	Average w/o Eye Tracker [s]	Average w/ Eye Tracker [s]
# 1	4.94	1.26
# 2	2.88	1.56
# 3	2.71	1.41
# 4	2.90	1.18
# 5	2.54	1.02
# 6	2.19	1.31
# 7	3.58	1.87
# 8	1.95	2.08
Mean	**2.96**	**1.46**
SD	0.94	0.36
SEM	0.33	0.13

The two-tailed P value equals 0.0053
By conventional criteria, this difference is considered to be very statistically significant.

Confidence interval:
The mean of Group One minus Group Two equals 1.5
95% confidence interval of this difference: From 0.61 to 2.39

Intermediate values used in calculations:
t = 3.98
df = 7
standard error of difference = 0.377

4.6 Conclusion

The result is shown in the following diagram:

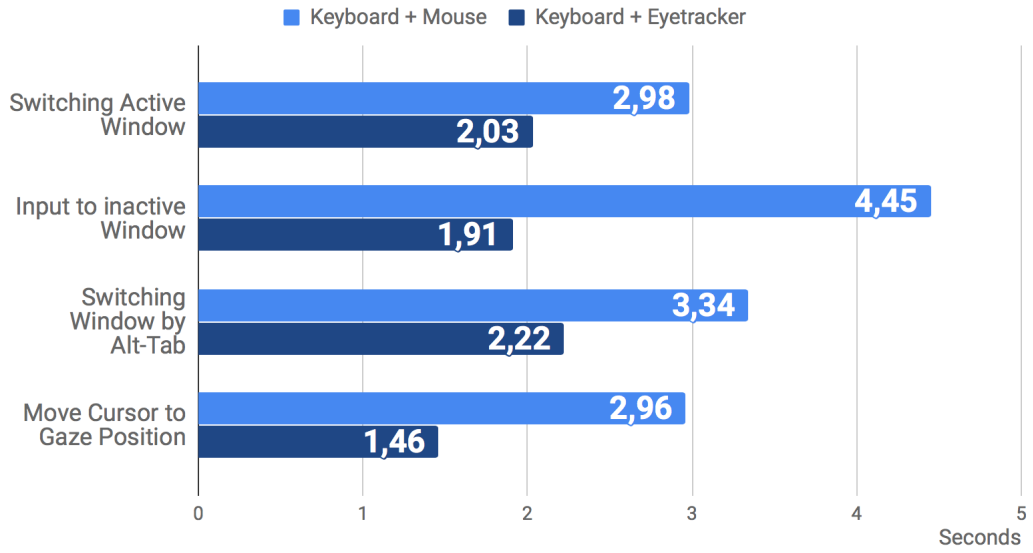

Figure 3: Results of the User Evaluation

The Keyboard combined with the Eye Tracker is faster for every task. There are further optimizations which could decrease the time needed. At the moment of the user evaluation the shortcuts consisted of three keys, which is a lot for shortcuts that haven't been used before. The probands are also much more used to the keyboard and the mouse than to the Eye Tracker. Therefore we expect a decrease in the needed time by providing dedicated keys or shorter hotkeys and more time to get used to them.
The time needed for "Move Cursor to Gaze Position" depends mostly on the time needed to click all buttons of the keyboard shortcut. "Switching Active Window" and "Input to inactive Window" only need one different key for the Eye Tracker solution. But the increase in the needed time for "Input to inactive Window" with a mouse and a keyboard is explained by the additional steps needed. First the proband must click on the window to press the desired shortcut (e.g. page down) or even just use the mouse. Then he needs to click on the previous active window or to Alt+Tab to make it active again. There were no instructions concerning the use of the cursor or Alt+Tab (we wanted the probands to solve the task the same they would outside of test-conditions). Finally there is an increase in the time needed while using the mouse if the position of the cursor is unknown before the start of the try.
The test persons enjoyed working with the Eye Tracker and the functionality offered by our program. One reason could be that everyone used an Eye Tracker for the first time. A long-term test would be important to see if the test persons would continue to use the Eye Tracker and for which scenarios it would be used most.

5 Conclusion and future work

The goal of this project to improve and simplify the use of multiple windows was certainly a success. As we saw in our user evaluation, each of our use-cases resulted in time-saving, even if the mouse is still more intuitive to use in the beginning.
An improvement could be made for our visual feedback, so that the gazed at window gets highlighted instead of just the gazed at point. This would be more intuitive, since all of our applications (except the relocating of the mouse cursor) are about the handling of windows. This proved itself to be a difficult task to implement, since all the windows consist of several subwindows which get highlighted instead, and some windows didn't support the highlighting at all.
This project could also be extended for the use of multiple monitors in the future, but the Tobii Eye Tracker doesn't support this feature yet.

6 References

[1] FUI 03, 2018, page 20