Assignment 6
Input Devices

Rasmus Kock Thygesen, 201909745
Timur Bas, 201906748

06/11/2020

In this assignment, we look at design interaction concepts with respect to two devices A, and B. The devices have the following input and output properties

Device A:

Output (visual)	Single-line alphanumeric display (max. 20 characters)
Output (auditiv)	Two different beeps
Input	2 Buttons

Device B:

Output (visual)	Single-line alphanumeric display (max. 20 characters)
Output (auditiv)	A beep
Input	5 Buttons and a scroll wheel

Additionally, both devices should support the following

- Entering a number between 0 and 10000
- Deleting the input
- Submitting the input

Note that

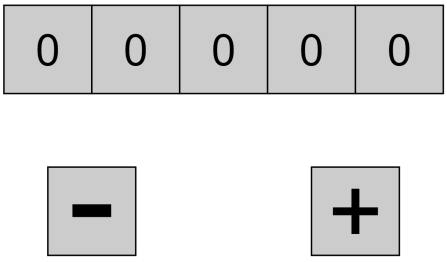
- Each key can be pressed once or several times
- Each button can be pressed for short or longer
- The scroll wheel can be moved in two directions (back and forward). It cannot be pressed.

Task 1: The creation of four interaction designs (two per device)

We will now explain the four different designs we have come up with and lightly discuss why we chose what we did.

The First Design of Device A

The first interaction design for Device A is shown below



The first design is what we thought was most obvious. We use one of the buttons to increment the number and the other button to decrement the number. When holding one of the buttons, the increments/decrements get bigger and bigger until you release the button again. This is what we usually see from ovens and similar electrocnics with a simple interface.

In order to delete the input on this device we click both buttons at the same time twice, and in order to submit the input we hold both buttons at the same time for two seconds. This ensures that the user does not accidentally delete or submit the input very often.

Examples:

a^c	means a clicked c times
a_c	means a pressed for c seconds
(a, b)	means a and b clicked at the same time

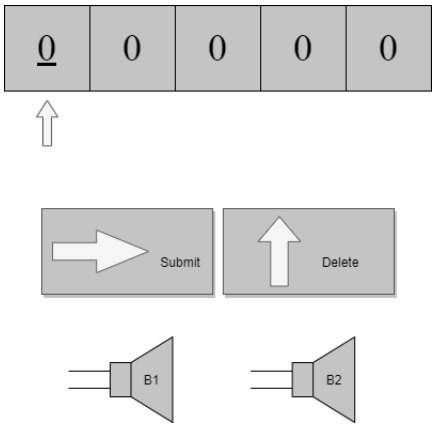
00000	
00003	$+^3$
02320	$+^{2317}$
00000	$(+,-)^2$
06723	$+^{6723}$
*****	$(+,-)_2$

These examples leave out the fact that holding one of the buttons starts incrementing faster and faster. When the device reaches the number 10,000, it loops back to 00000 at the next increment.

Since we change the number often, we think that it would be annoying to have auditive feedback on increments/decrements. This means we can use one beep for deletion and the other beep for submission. This also means the user will (hopefully) never be in doubt if they deleted or submitted the input. Another idea would be to have a short beep every time the increments/decrements get bigger and having a long beep on deletion. This would leave the second beep for submission.

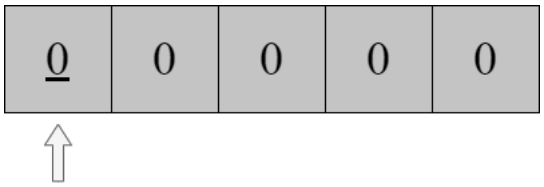
The Second Design of Device A

The second interaction design for Device A is shown below



The Layout of The Components

First off, we have a 1×5 input/output field consisting of 5 identical squares



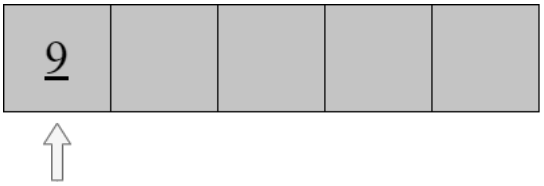
The pointer indicates which square is currently being selected, hence the underline in the number.

In order to increment the number or/and move the pointer, we look at the design for the buttons

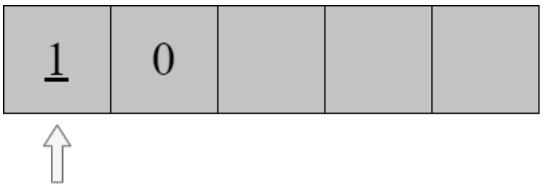
The button on the left (with the right arrow) when pressed once, moves the pointer one square to the right. When the pointer reaches the last square (the rightmost square), and the user presses the button again, then the pointer moves back to the leftmost square. The pointer always start at the leftmost square in case of deletion of the input etc.

The button on the right (with the up arrow) when pressed once, increments the current number by one.

To get a feel about how we interpret the output when incrementing we demonstrate the following example

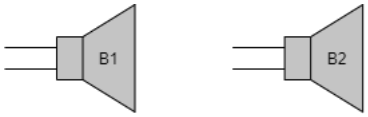


The above image shows the state where we have incremented the number at the first square 9 times



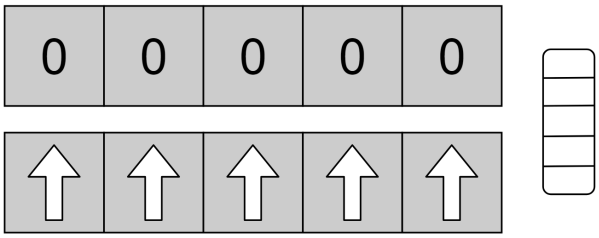
and the last image shows the state after we have incremented by 1. Therefore, as the user of the device, we do not see the other squares and the numbers until it gets relevant. Note that you can select a square without a number and increment it, and it will follow logically e.g. if the user selects the third square, and there is not a number in the second square, and the user increments by 1, then the second square will also show a number, since the third square shows a number.

In order to talk about submitting and deleting we look at the design for auditive output



When we press and hold the left button, we submit and the audio $B1$ is played. Likewise, when we press and hold the right button, we delete the input and the audio $B2$ is played.

The First Design of Device B



Our first design for device B is to use the fact that we have the same amount of buttons as we need characters. We use each button to select the corresponding digit in the number, and we use the scroll wheel afterwards to increment/decrement the digit. The currently selected digit will be underlined by the device.

In order to delete the input we hold the leftmost button for two seconds, and in order to submit the input we hold the leftmost and the rightmost button for two seconds. These two actions are far enough away from each other that the user will probably not delete or submit the input by accident.

Since we have plenty of inputs for controlling the number, we probably do not need auditive feedback for setting the number. Since the number should also be fairly quick to set, the user might be prone to deleting the number fairly often. This means auditive feedback for deleting the input could be annoying, which leaves the beep for submissions.

Examples:

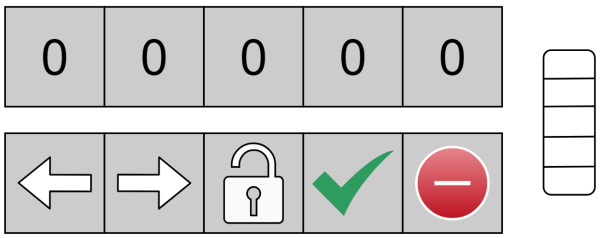
$[n]$	means button number n from left is clicked (zero indexed)
$([a], [b])$	means button number a from left and button number b from left are clicked at the same time
$[n]_c$	means button number n from left is held for c seconds
$+^n$	means the scroll wheel is scrolled up n times
$-^n$	means the scroll wheel is scrolled down n times

I will underline which digit is currently selected to make it easy to see.

$\underline{0}0000$	
$00\underline{0}00$	$[2]$
$00\underline{8}00$	$+^8$
$\underline{0}0000$	$[0]_2$
$0\underline{0}000$	$[1]$
$0\underline{5}000$	$+^5$
$050\underline{0}0$	$[3]$
$050\underline{6}0$	$+^6$
*****	$([0], [4])_2$

The Second Design of Device B

The second interaction design for Device B is shown below



The left and right arrows indicate which direction we want to go, i.e. the left button selects the number to the left of the current number, when pressed once, and likewise for the right button but selects the number to the right.

The button with the open lock only works together with the button with a check mark or the button with a delete icon. The button with the check mark submits the input, and the button with the delete icon deletes the input. When the user presses both the lock button and the submit/delete button simultaneously and holds them down for 2 seconds, then the action will occur. When the action occurs the system will give auditive feedback by a beep, to let the user know that the action has happened.

Our scroll is for incrementing and decrementing number values.

Task 2: Discussion of The Interaction Designs

In the following few sections we critically discuss the advantages and disadvantages of each of the interaction designs we came up with in Task 1

Discussion of Device A, Design 1

For this device, the precision of the input would go down the longer you hold the increment/decrement button. This is due to the increments/decrements getting bigger the longer you hold any of the buttons. So if the device starts at 0 and you want to enter $5,000$, then you will have to stop somewhere close to $5,000$ and then use smaller increments/decrements to get the exact value you want. This is the approach for many small devices that we use in our everyday such as the interface for ovens, so many people are already accustomed to doing this. Deleting and submitting the number is probably going to have to be something you just know, since it is not particularly intuitive. A lot of people would have an idea that there is probably some combination to do each of the two actions, but guessing this combination is not particularly easy, especially when you do not want to accidentally delete the number while trying to submit it.

One way to fix this problem would be to not increment more and more the longer you hold any of the buttons, but instead have some number a that the number is incremented by every time you press the increment button. You could then increase how much it increments by holding the increment button for two seconds. This would mean all changes in the number would be using single clicks and the amount you increment/decrement is changed by holding one of the buttons. This, however seems like an awful lot of work, and would probably need som time to get used to as well.

Discussion of Device A, Design 2

In this interaction design, the user would be slow to pick a certain number, because you have to operate with the button, select different squares all the time, and increment them individually. Additionally, if the user accidentally increments a number that was not supposed to be incremented, then the user has to increment the current number 9 times to get the desired number. This is an execution error, also known as a slip. However, we would have high accuracy and precision (if the user does not make too many slips), because the user is under control at all times what the individual numbers should be. An error that could also occur is that when the user would rapidly press e.g. the left button, such that the device feels that the button is pressed and hold, then the input would be submitted, or even if the user wants to switch digit a lot of times in a row and tries to hold the button to achieve this, which would accidentally submit or delete the number.

This problem with accidentally deleting/submitting while trying to switch digit could be resolved by using a combination of the two buttons to delete or submit, possibly the same way as we do it in our first design where clicking both buttons twice deletes the number and holding both buttons for two seconds submits the number.

Discussion of Device B, Design 1

For this design, the input efficiency is fairly good since the maximum amount of scrolls to a desired digit is at most 5 (you can both scroll back and forward). This coupled with the 4 clicks to get to each digit gives a maximum of 29 actions to get any given number assuming you do not make any slipses and/or lapses and have to go back. A maximum of 29 actions for any number and high accuracy and precision gives a really efficient input, especially considering that the scroll wheel is an analogue input, which means you can estimate how far you need to scroll your scroll wheel, so in order to scroll for instance 5 up, it does not really feel like 5 individual actions, but rather one estimate and one or two scrolls to adjust to your desired number. However, the user of the interface might do slips during incrementation/decrementation, because they use the scroll wheel continuously and not discretely, i.e. they scroll too far. This is not

a huge problem since a scroll wheel is a physically intuitive way of adjusting a number, which makes correcting mistakes rather quick. Furthermore, the design is somewhat intuitive and straightforward, but as in the first design of Device A, the user has to know the different combinations for submitting and deleting.

The main problem with this design is figuring out how to delete and submit. We could fix this problem by havign designated buttons for deletion and submission, and this is exactly what we do in our last design of device B which you can find below.

We believe this design could be very efficient for an experienced user since they get the ability to jump in digits, for instance if you are on the first digit and you just want to change the last digit, you do not have to click multiple times to get to the last digit. Furthermore, the scroll wheel is an analogue input as opposed to the buttons from device A, which allows the user to scroll in different amounts at a time as explained in previous paragraph.

Discussion of Device B, Design 2

For this design, the input effiency is also great for the same reasons as the previous design, but it is not quite as good as the previous design, since we cannot skip digits as explained for the previous design. Furthermore, the ways of submitting and deleteting are more explicit than in the previous design since we have labeled the two rightmost buttons with icons, so we can logically identify them and their meaning. Also, we prevent slips, because we only can submit or delete if we hold both the middle button and the respective button for two seconds. Thus, it is more easy and intuitive to use and learn the design of the user interface. This design is prone to scrolling too far, however, we covered why this is not a big deal in the discussion of the previous design.